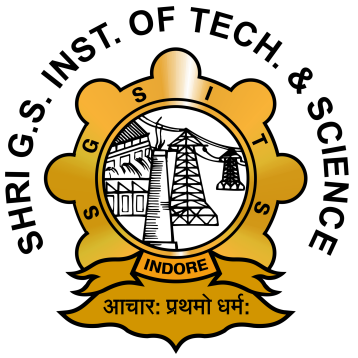
Human Anomaly Detection In Thermal Image using Deep Learning



2023-2025

A Dissertation Submitted to
Rajiv Gandhi Proudyogiki Vishwavidyalaya,
Bhopal towards the partial fulfillment of the degree of

Master of Technology

(Information Technology)

Supervised by:
Mrs. Sonu Airen
Associate Professor

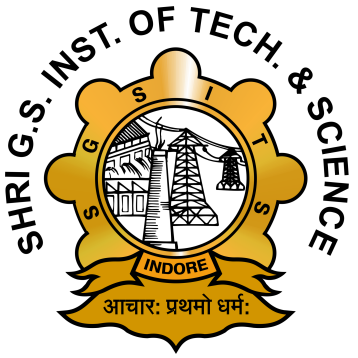
Submitted by:
Dhananjay Kumar Prasad
0801IT23MT03

Co-Supervised by:
Dr. Chandra Prakash Singar
Assistant Professor

DEPARTMENT OF INFORMATION TECHNOLOGY

SHRI GOVINDRAM SEKSARIA INSTITUTE OF TECHNOLOGY AND SCIENCE, INDORE(M.P.)

Human Anomaly Detection In Thermal Image using Deep Learning



2023-2025

A Dissertation Submitted to
Rajiv Gandhi Proudyogiki Vishwavidyalaya,
Bhopal towards the partial fulfillment of the degree of

Master of Technology

(Information Technology)

Supervised by:
Mrs. Sonu Airen
Associate Professor

Submitted by:
Dhananjay Kumar Prasad
0801IT23MT03

Co-Supervised by:
Dr. Chandra Prakash Singar
Assistant Professor

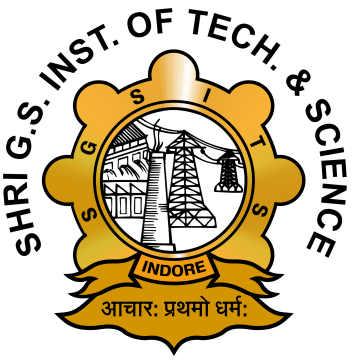
DEPARTMENT OF INFORMATION TECHNOLOGY

SHRI GOVINDRAM SEKSARIA INSTITUTE OF TECHNOLOGY AND SCIENCE, INDORE(M.P.)

SHRI GOVINDRAM SEKSARIA INSTITUTE OF TECHNOLOGY AND SCIENCE, INDORE(M.P.)

A Govt. Aided Autonomous Institute, Affiliated to RGPV, Bhopal

DEPARTMENT OF INFORMATION TECHNOLOGY



2023-2025

RECOMMENDATION

We are pleased to recommend that the dissertation work entitled **Human Anomaly Detection In Thermal Image using Deep Learning** submitted by **Dhananjay Kumar Prasad** may be accepted in partial fulfillment of the degree of **Master of Technology, Information Technology** of Rajiv Gandhi Proudyogiki Vishwavidyalaya, Bhopal (M.P.) during the **2023-2025**.

Mrs. Sonu Airen
Associate Professor
Information Technology

Dr. Anjulata Yadav
Professor
Electronics and Telecommunication

Dr. Chandra Prakash Singar
Assistant Professor
Information Technology

Dr. K. K. Sharma
TPREC Member and HOD
Information Technology

Forwarded By
Dean ARSD, SGSITS, Indore

SHRI GOVINDRAM SEKSARIA INSTITUTE OF TECHNOLOGY AND SCIENCE, INDORE (M.P.)

A Govt. Aided Autonomous Institute, Affiliated to RGPV, Bhopal

DEPARTMENT OF INFORMATION TECHNOLOGY

CERTIFICATE

This is to certify that the dissertation entitled **"Human Anomaly Detection In Thermal Image using Deep Learning"** submitted by **Dhananajy Kumar Prasad** is accepted in partial fulfillment of the degree of **Master of Technology, Information Technology** of Rajiv Gandhi Proudyogiki Vishwavidyalaya, Bhopal during the **2023-2025**.

Internal Examiner
Date:

External Examiner
Date:

DECLARATION

I, **Dhananjay Kumar Prasad**, student of **M.Tech in Information Technology** at the **Department of Information Technology, SGSITS, Indore**, hereby declare that the dissertation titled **"Human Anomaly Detection In Thermal Image using Deep Learning"** is the result of my own work carried out under the supervision of **Mrs. Sonu Airen**, Associate Professor, and **Dr. Chandra Prakash Singar**, Assistant Professor, Department of Information Technology, SGSITS, Indore.

I further declare that to the best of my knowledge, this dissertation work does not contain any part of any work which has been submitted for the award of any degree or any other work either in this University or in any other University/website without proper citation.

Signature of the candidate :

Name of the candidate : Dhananjay Kumar Prasad

Enrollment No. : 0801IT23MT03

Date :

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to my guide **Mrs. Sonu Airen** and co-guide **Dr. Chandra Prakash Singar** for their constant encouragement, guidance, and valuable insights throughout the course of this project. Their constructive feedback and continuous support helped me stay focused and motivated. I am also thankful to the faculty and staff members of the **Department of Information Technology,** SGSITS, for their support and for providing the necessary facilities to complete this work. I would like to extend my appreciation to my fellow researchers, classmates, and friends who provided a stimulating and fun environment in which to learn and grow. Lastly, I express my heartfelt thanks to my family for their endless love, patience, and belief in me throughout my academic journey.

Dhananjay Kumar Prasad
0801IT23MT03
Date:

ABSTRACT

This thesis presents a deep learning-based framework for human action recognition and anomaly detection from thermal images, with a specific emphasis on pose estimation. The framework we proposed processes thermal images in stages. First, we extracted frames from the thermal video, followed by preprocessing the thermal frames, which included resizing, augmenting, and labelling action classes; labelling bounding boxes, and labelling 17 COCO-like keypoints. We developed a custom dataset with nine human actions including walking, sitting, lying, and an abnormal behaviour class. Lastly, we trained a YOLOv8-Pose model on the IM-Thermal dataset to both detect humans and estimate pose. Among the tested variants, the YOLOv8n-pose had the best accuracy-efficiency tradeoff. When evaluated on the IM-Thermal validation set, the YOLOv8n-pose achieved bounding box and pose mAP@0.5 average precision scores of 0.98 with mAP@0.5:0.95 scores of 0.96–0.97. It also achieved bounding box precision and recall values of 0.94 and 0.96, respectively, and pose precision and recall values of 0.93 and 0.96, respectively. We evaluated the anomaly detection action class mAP@0.5 scores and obtained average scores of 0.99 for the normal actions and 0.98 for the abnormal actions, while the average pose mAP@0.5:0.95 score for the abnormal behaviour class was 0.82. The results show that the Deep Learning model can be effective for reliably detecting slight changes in human poses from thermal imagery in infinitely variable and difficult thermal conditions. Additionally, the proposed system can be built to detect abnormal behaviours in real-time based on distributions of pose and action classes. Overall, the results confirm that pose-based analysis using thermal imagery is an appropriate, privacy-respecting and illumination-independent, method for automated human behavior monitoring in complex indoor scenarios, with direct relevance for applications in surveillance, healthcare, and security fields of study.

Table of Contents

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

CNN Convolutional Neural Network

DL Deep Learning

FPS Frames Per Second

HAR Human Action Recognition

HPE Human Pose Estimation

FPS Frames Per Second

GFLOPs Giga Floating Point Operations per Second

GPU Graphics Processing Unit

IR Infrared

IoU Intersection over Union

ML Machine Learning

mAP Mean Average Precision

LR Learning Rate

ReLU Rectified Linear Unit

Val Validation

RGB Red Green Blue

TPU Tensor Processing Unit

YOLO You Only Look Once

Chapter 1

INTRODUCTION

1.1 Background and Motivation

Human Action Recognition (HAR) refers to the detection and interpretation of human actions from vision data. Recently, it has gained prominence as a key research area because of its application in a wide range of fields like healthcare monitoring, sports analysis, smart surveillance, self-driving cars, assisted living spaces, and human-machine interaction. HPE systems are predominantly dependent on deep learning models, such as convolutional neural networks (CNNs), recurrent neural networks (RNNs), and transformer models [2]. These models have pushed the state of the art by allowing accurate detection of keypoints that are human joints. Estimating poses for more than one person adds complexity, as the models need to detect multiple persons and properly attribute each joint to the corresponding person. Two main techniques are utilized here: bottom-up and top-down.

The bottom-up approach starts by localizing all of the body joints in an image and then clusters them into separate skeletons. Models like OpenPose [3], HRNet [4], and CenterNet [5] adhere to this approach. Top-down methods, on the other hand, initially detect every person and afterwards estimate their pose in the detected area. Although more computationally expensive, this process tends to provide better accuracy, such as in the cases of ViTPose [6], YOLOv8-Pose [7], and AlphaPose [8]. Although there has been impressive development here, most of the existing work and development has been aimed at action recognition by utilizing visible spectrum imagery. However, at low-light or obscured conditions, visible cameras can fail and thermal imaging then becomes an attractive alternative. Realizing this, some recent works have begun to leverage the promise of deep learning techniques on thermal video streams for the extraction of spatial as well as temporal information for reliable action recognition [9].

Infrared thermal sensors operate by detecting the heat that objects emit or reflect, rather than depending on visible light. This emitted energy, called infrared (IR) radiation, is part of the electromagnetic spectrum and is directly tied to an object's temperature. Since IR wavelengths are longer than those of visible light, they are invisible to the human eye. The IR spectrum is commonly categorized into bands: Near-Infrared (0.7–1 μm), Short-Wave Infrared (1–3 μm), Mid-Wave Infrared (3–5 μm), Long-Wave Infrared (8–14 μm), and Very Long-Wave Infrared (more than 14 μm). The shorter bands (NIR and SWIR) record reflected IR light, whereas MWIR and LWIR detect the natural radiation emitted by objects no other light source is needed. That is why thermal imaging is extremely effective in total darkness and under changing lighting or weather conditions [10].

The aim of this thesis is to develop a deep learning system capable of detecting and classifying human actions, including anomalies, from thermal images. The research intends to close the performance gap by fine-tuning state-of-the-art pose estimation models on thermal datasets to enhance real-time decision making in safety-critical applications.

1.2 Need of this Project

As surveillance systems become more demanded in critical environments like healthcare centers, old age homes, and residential spaces, there is an increasing requirement for privacy-preserving technologies that can ensure safety. Conventional light-based surveillance systems record detailed visual data, which tends to create privacy issues when used in personal or private areas. Conversely, thermal imaging offers a privacy-friendly solution by recording heat signatures instead of clear visual identities. Additionally, thermal imaging-based systems perform well even in adverse lighting environments, like night or dark settings, where traditional light-based cameras cannot perform. Even with these benefits, thermal image utilization in human action recognition and anomaly detection is still an untapped field because of issues like low-resolution images, insufficiency of detailed features, and restricted datasets.

The necessity for this project stems from the discrepancy between the promise of utilizing thermal imaging in privacy-sensitive and low-light settings and available limitations in resilient human action recognition algorithms. Filling such gaps, this work would aspire to contribute towards the establishment of robust, accurate, and privacy-conscious systems for human detection, action recognition, and anomaly detection from thermal images.

1.3 Problem Statement

Most work in this area has focused on standard, light-based cameras, but that approach struggles with poor lighting, blocked views, and can create privacy issues. A practical alternative is thermal imaging, which senses heat instead of light. This gives it a major advantage in dark conditions and naturally keeps subjects anonymous.

The challenge with thermal images, however, is their general lack of texture and color. This scarcity of detail makes it tough to distinguish between fine-grained human postures and specific activities. On top of that, existing pose estimation models are almost exclusively trained on datasets from conventional cameras, so they fail to perform reliably when used on thermal footage.

This research tackles the problem of spotting human anomalies in thermal video by using deep learning for both action recognition and pose estimation. The ultimate aim is to engineer a system that can effectively detect and classify actions and abnormalities from thermal frames. The goal is for this system to outperform existing methods based on standard cameras while providing the same level of reliability in real-world thermal scenarios.

1.4 Project Objectives

The main goal of this project is to create a robust and efficient system for human action recognition and anomaly detection based on thermal imaging, prioritizing privacy-protecting settings and harsh low-light conditions. To address this, the project establishes the following specific goals:

- To investigate and examine the limitations of current techniques for human detection, action recognition, and anomaly detection in thermal images.
- To develop a deep learning-based system that is able to effectively detect human presence and identify actions from thermal data.
- To create a method to detect abnormal behaviors from thermal images by learning normal activity patterns.
- To use suitable datasets and, where required, create more annotated data ready for thermal-based human action recognition tasks.
- To compare the performance of the proposed method under different conditions to ensure robustness in privacy-sensitive, low-light, and diverse environments.

1.5 Proposed Approach

The proposed approach is to create a deep learning-based framework for human detection, action recognition, and anomaly detection with thermal imaging data. The system is intended to perform well in privacy-sensitive conditions and low-light environment.

Preprocessing operations will be performed on the thermal image data initially to improve the quality of the input and prepare it for deep learning models. A pose estimation model, such as YOLO-based keypoint detection, will be used to extract human skeletal keypoints from thermal images. These keypoints will be used as a privacy-preserving and compact representation of human actions. Then, a temporal analysis of the keypoints will

be performed to recognize patterns of normal actions and detect deviations that can indicate abnormal behaviors. The developed system will be assessed with suitable datasets that are annotated with actions and anomalies in thermal images. Performance will be quantified in terms of accuracy, resilience, and extensibility in different environmental conditions for its applicability to real-world deployment in healthcare, home, and surveillance scenarios.

The developed methodology is based on the following main components:

- **Data Capture and Preprocessing:** Thermal image frames are captured and extended to create a complete dataset of human behavior.
- **Object Detection and Pose Estimation:** Yolopose or other deep learning architectures detection of human bodies and main body joints in thermal images.
- **Anomaly Classification:** Deep neural networks classify actions as normal or abnormal according to pose and movement patterns.
- **Model Evaluation:** Performance is measured by standard classification metrics and visual inspection tools to achieve reliability and robustness.

1.6 Organization of the Thesis

This thesis is organized in five chapters, which all intend to lead the reader through the research process from defining the problem to drawing final conclusions:

- **Chapter 1: Introduction**
  This chapter sets the stage for the research by presenting the issue of human anomaly detection in thermal images. It sets out the motivation, the research goals, and the general outline of the study.
- **Chapter 2: Literature Review**
  This chapter summarises current work in thermal imaging, pose estimation, and action recognition. It explains the strengths and weaknesses of previous techniques and specifies the gaps in existing research that this thesis aims to fill.
- **Chapter 3: Case Study and Problem Identification**
  In this chapter, a practical scenario is explored to better understand the real-world challenges of recognizing human actions using thermal images. By studying current systems and technologies, we identify key limitations especially in low-light and privacy-sensitive situations. These insights help define the core problems that this project aims to solve.

- **Chapter 4: Methodology and Data Collection**

  This chapter explains how the project was carried out from the initial planning to the techniques used in building the system. It covers how thermal data was collected or prepared, how preprocessing was handled, and how models were selected and trained. The goal is to show a clear and logical process that supports accurate and reliable human action recognition using thermal imagery.

- **Chapter 5: Implementation and Evaluation Metrics**

  This section gives a clear description of the proposed system, such as the dataset properties, preprocessing, model structures (YOLOv8-Pose variants), and training. Evaluation metrics utilized in assessing performance are also explained.

- **Chapter 6: Results and Analysis**

  Experimental setup and results are given in this chapter. The performance of various models over thermal image data is analyzed with the help of common metrics. Comparative analysis is performed based on visualizations and quantitative outcomes.

- **Chapter 7: Conclusion and Future Work**

  This last chapter recapitulates the most important results and contributions of the work. It also defines the limitations of the proposed approach and some possible directions for further work, such as model improvement and real-time deployment potentials.

Chapter 2

LITERATURE SURVEY

2.1 Detailed Literature Review

Human action recognition and detection in thermal and infrared imagery has attracted growing research interest due to its potential in night-time surveillance and security applications. A variety of approaches leveraging deep learning, sensor fusion, and innovative feature extraction have been developed to address the inherent challenges of limited illumination and low-contrast thermal data.

Manssor et al. [11] solved the problems of pedestrian detection in thermal infrared images by enhancing the Tiny-YOLOv3 model. They augmented it with channel-wise contrast enforcement and paired it with a hybrid architecture consisting of PDM-Net and TIE-Net. Darknet-53, in this configuration, was tasked with extracting strong feature representations, while PDL-Net carried out classification operations. The approach effectively minimized loss of information during the early stages of processing, leading to more consistent detections, particularly in low-light environments where visible-spectrum detectors perform poorly.

Imran et al. [12] proposed a four-stream deep learning architecture that pairs CNN and BiLSTM networks for detecting global and local motion patterns. Their approach infused dense optical flow-based features in the forms of SSDI and SDFDI, which allowed for encoding spatial and temporal information more holistically. By dividing video clips into segments and processing them through parallel CNN-BiLSTM streams, their system was able to fuse complementary features and deliver better action recognition across a wide range of activities.

Krišto et al. [13] compared some of the top object detection models, including YOLOv3, by retraining them on thermal images recorded under different weather conditions such as rain, fog, and clear nights. Their findings suggested that YOLOv3 represented a good accuracy-speed trade-off and therefore was an efficient choice for real-time surveillance systems.

Batchuluun et al. [14] sought to recover skeletal keypoints from thermal video. They did so by converting single-channel thermal frames into three-channel inputs appropriate for a Joint-GAN model and subsequently generating joint and skeleton data with it. These were then passed through a CNN-LSTM architecture, allowing it to recognize complex human actions accurately.

Ding et al. [15] designed a thermal infrared system that was aimed at recognizing airport apron activities. Their pipeline began with the use of tracking algorithms to identify moving

individuals from the background and subsequently extract spatiotemporal features within short time windows. These were input into a deep network with stacked LSTM layers to identify longer-term temporal patterns in order to aid the classification of walking, standing, and operational movements behaviors.

A major contribution in this space is the work by Liu and Ostadabbas [16], who created the SLP dataset. This resource includes thermal, visible, depth, and pressure images gathered from 109 participants lying in bed under different conditions, such as uncovered, thinly covered, and fully covered scenarios. Each person performed multiple poses across three main categories supine, left side, and right side leading to a total of 14,715 images. Their findings showed that visible images produced strong results when no cover was present, but performance dropped considerably when blankets were used. In such cases, thermal images were more effective for pose detection. Despite its usefulness, the SLP dataset has low variability since all data were recorded in the same environment with identical sensor settings, which limits its applicability to other contexts.

Building on this dataset, Liu et al. [17] examined how combining different sensing modalities including visible, thermal, depth, and pressure information could enhance pose estimation accuracy. Their experiments confirmed that multimodal input improves performance. Nevertheless, their work remained restricted to in-bed monitoring due to the dataset's narrow scope.

In an effort to further improve thermal pose estimation, Chen et al. [18] compiled a large dataset containing 24,000 pairs of thermal and visible images recorded indoors. The visible images were high-resolution, whereas the thermal images had much lower resolution (80 × 60 pixels). To generate labels, OpenPose was used on the visible images in the training set, while a subset of 2,000 test images was manually annotated. They proposed the ThermalPose model, which adapts OpenPose for thermal data. Experimental results showed that visible-based models achieved better accuracy under good lighting conditions, but in low-light or dark settings, ThermalPose outperformed all other approaches because visible cameras failed to detect people. However, the dataset's limited manually labeled subset and its indoor-only nature pose challenges for broader application.

To help address the lack of thermal data, Kniaz et al. [19] developed ThermalGAN, a generative adversarial network capable of translating visible images into thermal images support tasks like person re-identification. Their approach incorporated segmentation masks to estimate average temperatures for each object and to model variations within them. They introduced the ThermalWorld dataset, which includes over 15,000 visible–thermal

image pairs with corresponding object annotations. Although this work shows promise in supplementing training data, its evaluation relied mostly on subjective judgments of image realism rather than objective performance metrics, making its practical impact on pose estimation uncertain.

Mehra et al. [20] also explored the benefits of fusing thermal and depth information for pose estimation. They developed a smaller dataset of 1,000 labeled images divided into training, validation, and testing sets. Using a modified version of the part affinity fields detector, they demonstrated that combining thermal and depth modalities resulted in more accurate detection than thermal input alone. However, the dataset's annotations covered only five keypoints per person, which limits its ability to support more detailed pose estimation. Several benchmark datasets have been introduced to facilitate research in thermal human detection and pose estimation. Each offers distinct characteristics suited for different application scenarios.

The CAMEL dataset [21] contains 26 sequences of paired color and thermal videos, totaling over 23,000 annotated frames, with around 7,775 precisely aligned pairs. Captured at 336 × 256 resolution and 30 fps in the LWIR spectrum, it features both indoor and outdoor urban settings under diverse lighting and weather conditions.

The KAIST dataset [22] provides 95,000 aligned color and thermal image pairs over 103,000 annotated pedestrian bounding boxes. Acquired at 640 × 480 resolution and 20 fps, it includes dynamic outdoor scenes captured from a moving vehicle across varying times of day and weather.

The OTP dataset [23] offers 6,090 thermal images with bounding boxes and 17 keypoints per person, covering over 14,000 human instances in challenging outdoor conditions. It includes a range of activities, occlusions, scale variations, and environmental diversity. The LLVIP dataset [24] supports pedestrian detection and color–thermal fusion in low-light conditions, featuring 15,438 aligned image pairs from nighttime scenes. Its extension, LLVIP-POSE (LLVIP-P) [25], is the largest thermal pose estimation dataset to date, with over 26,000 annotated poses across training and test sets.

2.2 Summary of Literature Review

Research on human action recognition using thermal imagery has grown steadily, especially as conventional methods struggle in low-light or privacy-sensitive environments. To improve performance, researchers have adapted models like YOLOv3 and Tiny-YOLOv3 for thermal inputs. Others have explored combining CNNs with BiLSTMs to better capture both spatial features and motion over time. For understanding body movement, thermal keypoints have been estimated using LSTM-based and GAN-driven approaches. Multi-modal setups where thermal data is paired with visible, depth, or pressure inputs have shown improved accuracy when standard vision fails. Due to the lack of large, labeled thermal datasets, tools like ThermalGAN are being used to generate realistic synthetic data. Public datasets like CAMEL, KAIST, OTP, LLVIP, and LLVIP-POSE are now playing a key role in pushing the field forward.

Chapter 3

CASE STUDY AND PROBLEM IDENTIFICATION

This chapter presents a detailed case study and identifies the specific problems addressed in this research. It explores prior studies relevant to various components of the proposed work, such as preprocessing, feature extraction, model selection, and evaluation strategies.

3.1 Previous Studies in Proposed Work

Several previous studies have provided valuable insights that form the basis of the proposed work, particularly in the areas of human detection, action recognition, and anomaly detection using thermal images. While much existing research has focused on RGB images, recent work highlights the potential of thermal imaging in environments where privacy and low-light conditions are important concerns.

For example, Manssor et al. [11] improved the Tiny-YOLOv3 model for pedestrian detection in thermal images by enhancing contrast and using a hybrid architecture. This approach showed better performance, especially in poor lighting conditions where traditional visible light methods often fail. Similarly, Krišto et al. [13] retrained YOLO models with thermal images collected under different weather conditions, confirming YOLO's effectiveness for thermal based detection tasks. In terms of action recognition, Imran et al. [12] introduced a combination of CNN and BiLSTM networks to capture both spatial and temporal features from video data, which is highly relevant for analyzing human actions in thermal images. Batchuluun et al. [14] worked on extracting skeletal keypoints from thermal images using GAN-based models and further processed this information with CNN-LSTM architectures to recognize complex human activities. Datasets like SLP, LLVIP, CAMEL, and OTP have also played a significant role in advancing research in thermal-based human detection and pose estimation. These studies and resources collectively support the direction of the present work and highlight the potential of thermal imaging combined with deep learning techniques for action recognition and anomaly detection.

3.2 Study Used in Preprocessing of Data

Preprocessing is an important step when working with thermal images, as these images often suffer from low resolution, poor contrast, and limited texture information. To address these challenges, several studies have focused on various preprocessing techniques to improve the quality and usability of thermal data for deep learning applications. Previous research commonly applied image resizing, normalization, and noise reduction techniques to standardize the input data and enhance the visual quality of thermal images. In particular, image augmentation methods such as flipping, rotation, scaling, and adding slight noise

have been widely adopted to increase the diversity of training data, reduce overfitting, and improve the robustness of models during testing. Studies involving thermal-based human detection and pose estimation, such as those by Liu et al. [16] and Chen et al. [18], have emphasized the importance of careful preprocessing. These works highlight how preprocessing not only helps in improving model accuracy but also ensures the consistency of data fed into deep learning pipelines. In this project, the same preprocessing steps have been utilized. The thermal images are resized to the input size that the detection and pose estimation models require. Augmentation methods are utilized for the purpose of model generalization, and normalization is employed to scale the data to an appropriate range for neural network processing.

3.3 Study Used in Feature Extraction

Feature extraction is especially important for human action understanding from thermal images, particularly for low-resolution and low-contrast images. A number of research works have investigated efficient feature extraction techniques from thermal images, with the majority of them targeting human body keypoints and pose estimation. Pose estimation models are traditionally utilized by researchers to extract skeletal keypoints as a compact and privacy-preserving representation of human activities. The keypoints are useful structural information regarding body posture and movement, which is fundamental for precise action recognition. For example, Batchuluun et al. [14] used GAN-based models to transform thermal images into appropriate inputs for joint and skeleton information extraction, which were subsequently applied for action recognition using CNN-LSTM frameworks. Likewise, Chen et al. [18] proposed the ThermalPose model, which adopts pose estimation techniques such as OpenPose to operate efficiently with thermal information, the significance of skeletal keypoints in gathering human posture under poor visual conditions. Feature extraction in this project depends on pose estimation through YOLO-based keypoint detection algorithms adapted for thermal images. The extracted keypoints form the core features for subsequent action recognition and anomaly detection operations, enabling the system to concentrate on human body structure and motion and not raw pixel values.

3.4 Study Used in Model Selection

Model choice is an important phase of planning an efficient system for human action recognition and anomaly detection, particularly when dealing with thermal images. A number of research studies have investigated and proven the application of deep models

for this task, informing the choice of appropriate architectures for the current research. Other research has shown that convolutional neural networks (CNNs) can efficiently extract spatial features from images, whereas recurrent neural networks (RNNs), especially Long Short-Term Memory (LSTM) networks, have been utilized extensively to capture temporal dependencies in sequential data. For instance, Imran et al. suggested a CNN-BiLSTM architecture to combine both spatial and temporal data, with enhanced performance in action recognition tasks for thermal data. Transformer models have also gained interest for their capacity to model long-range dependencies and capture intricate temporal patterns. Moreover, YOLO-based models have been preferred due to their efficiency and effectiveness in human detection and keypoint estimation tasks, as evident in research by Manssor et al. [11] and Krišto et al. [13] On the basis of these findings, this project uses a hybrid framework with YOLO-based pose estimation for feature extraction and CNN-LSTM structures for modeling the temporal nature of actions. The choice is driven by the established capabilities of these models in the ability to deal with specific challenges of thermal imaging, like low resolution and absence of fine textures, without compromising on accuracy and efficiency.

3.5 Study Used in Evaluation

Evaluation is an important component of verification of performance of any human action recognition and anomaly detection system, particularly when working with thermal imaging data. Past research works have always employed standard evaluation measures to verify the efficacy and reliability of their models. These measures are accuracy, precision, recall, F1-score, and mean Average Precision (mAP), all of which contribute towards an overall insight into a model's performance across different tasks. For example, object detection and human pose estimation studies like those conducted by Krišto et al. [13] and Manssor et al. [11] have used mAP and detection rate to analyze how accurately the models can detect human figures from thermal imagery. Likewise, action recognition studies like those by Imran et al. [12] and Batchuluun et al. [14] employed precision, recall, and F1-score to quantify the systems performance in classifying actions correctly over time. In this project, evaluation adheres to these common practices. Detection accuracy and mAP are utilized to test the performance of the pose estimation phase, and precision, recall are used to test the system performance in identifying actions and detecting anomalies. This makes sure that the outcomes are similar to previous research and gives a concise measurement of the system's robustness and reliability under various circumstances.

Chapter 4

METHODOLOGY AND DATA COLLECTION

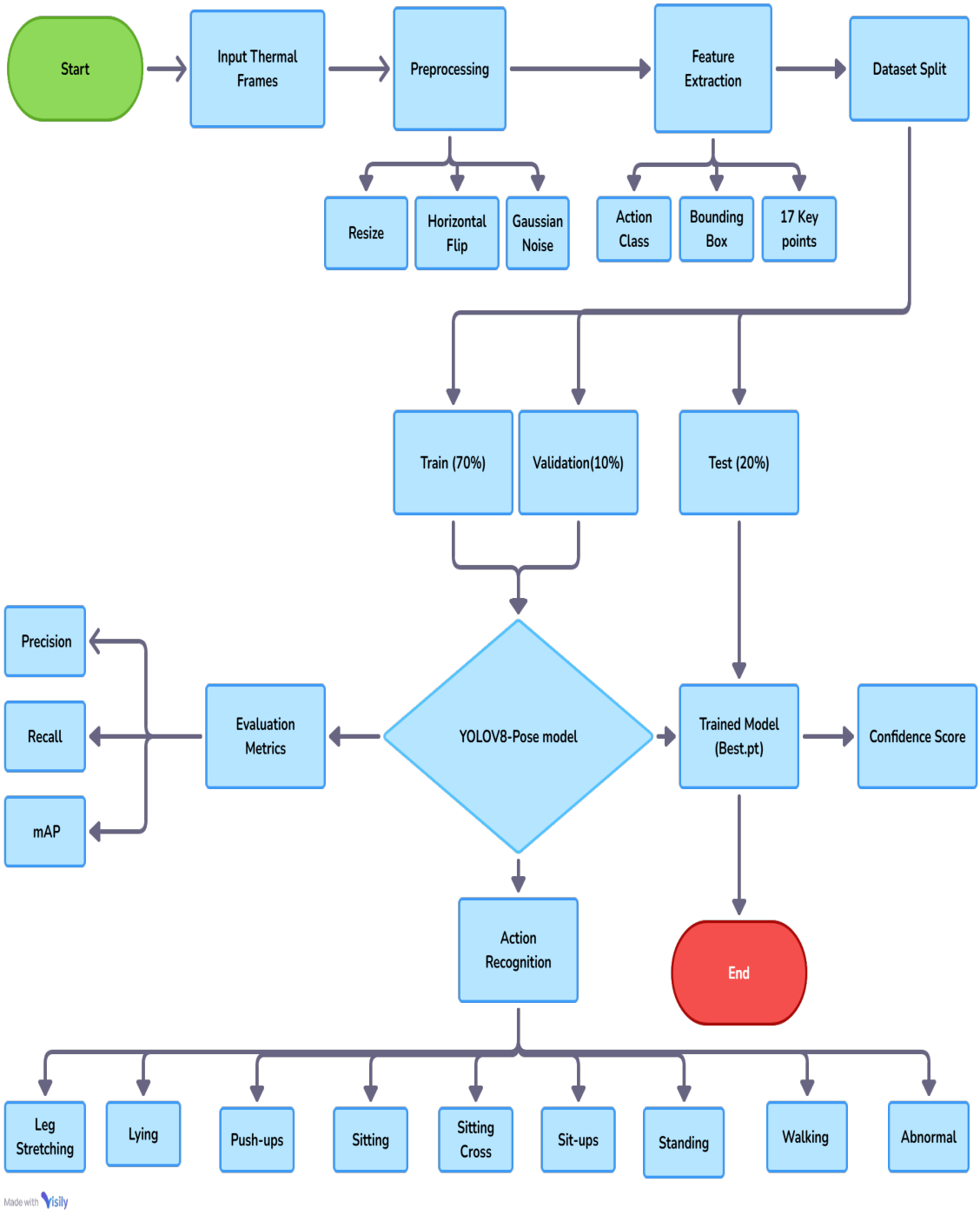


Figure 4.1: Flowchart:Training and Testing Human Pose Estimation on Thermal Images using YOLOv8-pose

4.1 Data Acquisition

This work used the thermal component of the Thermal Indoor Motion (Thermal-IM) dataset [1] to develop a human action detection system specifically designed for indoor environments. The thermal sequences were recorded with a Hikvision DS-2TD4237T-10 camera at a frame rate of 15 frames per second and a resolution of 288 × 384 pixels. To overcome challenges caused by low illumination and background clutter, only the

thermal data were considered, even though the dataset also includes RGB and depth channels. The dataset comprises 783 video segments totaling over 560,000 thermal frames (approximately 10.4 hours) and captures actors performing everyday activities across various room configurations and camera positions. As seen in Fig. 4.2, this diverse and practical thermal-only dataset enables robust training and evaluation of models for pose estimation and action recognition.

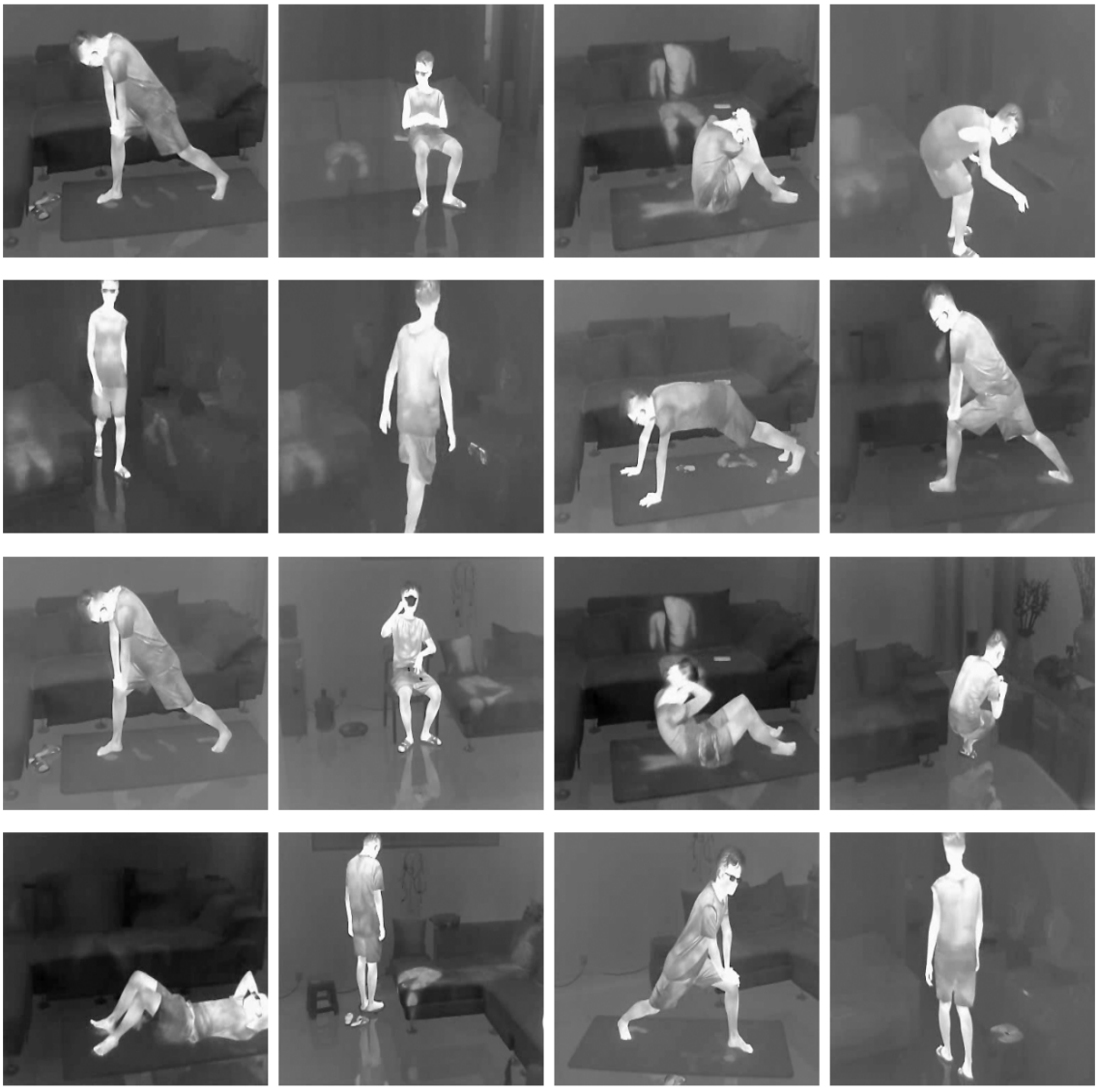


Figure 4.2: Sample images from the Thermal-IM dataset [1].

4.1.1 Data Extraction

53 thermal video samples were processed to construct a structured dataset for Human Action Recognition (HAR). Each video was accompanied by a corresponding JSON annotation file with temporal labels defining the start and end of various human actions. These actions were annotated into nine pre-defined classes: *Abnormal*, *Leg_stretching*, *Lying*, *Push-ups*, *Sitting*,

Sitting_crosslegs, *Sit-ups*, *Standing*, and *Walking*. A Python script was designed to extract frames and corresponding short video samples automatically from the annotated areas. The source videos were captured using a thermal infrared camera in MPEG-4 (.mp4) format, resolution 288 $\times$ 384 pixels, and captured at 15 frames per second (FPS) in the 7.5–14 μm spectral band. The output was organized into class-specific directories to ensure a clean and well-annotated dataset appropriate for keypoint extraction and action classification. This processed dataset was used as the foundation for training and testing the YOLOv8-Pose model on thermal human action sequences. The video input was broken down into separate frames based on its frame rate, which is the number of frames recorded per second.

4.2 Data Preprocessing

To standardize the input for the YOLOv8-Pose model, all thermal images were resized to a fixed resolution of 640×640 pixels using a custom script built with the Pillow library. This resizing ensured uniform spatial dimensions across the dataset, facilitating efficient training and inference. To enhance model generalizability, standard data augmentation techniques are applied to the thermal images. Horizontal flipping simulates mirrored movement, while Gaussian noise is introduced to mimic real-world thermal sensor distortions as seen in Fig. 4.3. These augmentations were implemented using the PyTorch and torchvision libraries.

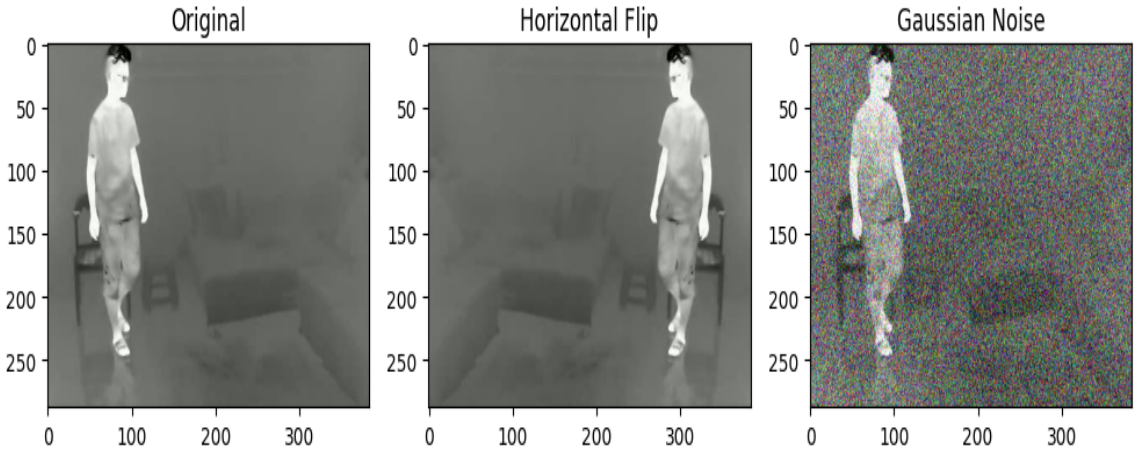


Figure 4.3: Example of data augmentation: (a) original, (b) horizontally flipped, (c) with Gaussian noise.

4.2.1 Dataset Summary and Class Distribution

After preprocessing, the final dataset consisted of 9,414 thermal images categorized into nine distinct human action classes as shown in Table 4.1, including both simple and complex movements. The dataset was organized into class-specific directories, making it suitable for supervised learning tasks. The preprocessing steps helped reduce overfitting, increased

robustness to real-world conditions, and provided a consistent and diverse foundation for pose estimation and action recognition in thermal environments.

Table 4.1: Summary of Image Distribution Across Action Classes in the Thermal Dataset

Class	**Number of Images**
Abnormal	873
Leg_streching	1,749
Lying	414
Push-ups	777
Sit-ups	1,692
Sitting	1,350
Sitting_crosslegs	468
Standing	312
Walking	1,761
Total	**9,414**

4.3 Keypoint and Bounding Box Annotation

In this work, we manually prepared a dataset consisting of 9,414 thermal images, each annotated with bounding boxes and 17 human keypoints according to the widely adopted COCO keypoint format. These keypoints capture critical anatomical landmarks such as the nose, eyes, shoulders, elbows, wrists, hips, knees, and ankles. Due to the nature of thermal imagery, where body outlines and joint positions are often less distinct, precise annotation proved to be a challenging and time-intensive task. For initial annotations, we utilized a YOLOv8m-pose model pre-trained on the COCO dataset to automatically detect bounding boxes and estimate keypoint positions. These initial predictions were then carefully reviewed and corrected through manual refinement to ensure accuracy, especially in cases where keypoints were missed or incorrectly positioned. All reliable detections were filtered using a confidence level of 0.7. Annotations were stored as.txt files for each image after the results were translated into normalized YOLO format. Only frames with legitimate detections were kept, and the dataset was arranged into distinct directories for images and labels. The pose estimation model may be trained and evaluated efficiently because to this standardized framework.

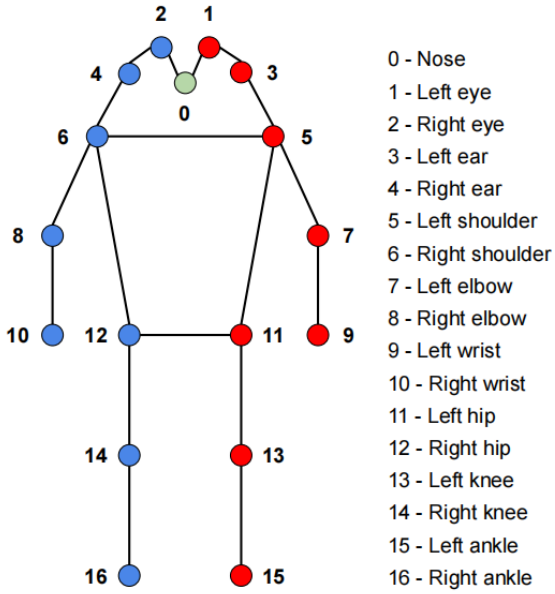


Figure 4.4: Keypoints annotation format used for Thermal-IM dataset

Algorithm 1 Algorithm for Thermal Image YOLOPose Annotation

Input: YOLOPose results on 640×640 image
Output: YOLO-style .txt file with bbox + 17 keypoints

Start
Set image width $W = 640$, height $H = 640$
for each detection in results **do**
    Extract bounding box $\rightarrow$ (x_{center}, y_{center}, $width$, $height$)
    Normalize:
        $x_{center} = x_{center}/640$
        $y_{center} = y_{center}/640$
        $width = width/640$
        $height = height/640$
    for each of 17 keypoints (x_i, y_i) **do**
        Normalize $x_i = x_i/640$, $y_i = y_i/640$
        Set visibility $v_i = 2$
    Format output line as:
        $\rightarrow$ class_id x_{center} y_{center} $width$ $height$ x_1 y_1 v_1 ... x_{17} y_{17} v_{17}
    Save the line to .txt file
Stop

4.4 YOLOv8-Pose Models: Architecture and Comparison

Yolo model consists of many elements including: stem, downsampling layers, stages composed of primary building blocks, and head. The initial component of the network known as the stem, maintains the responsibility of taking in and processing the raw input data, typically images and video. The downsampling layer is tasked with reducing the spatial resolution of the features, thereby making the overall model computationally efficient, while simultaneously increasing the receptive field of the model.Typically, each level of YOLO's

structure consists of numerous convolutional blocks, referred to as basic building blocks or structures. Generally, each level produces features at higher levels of abstraction and processes data at a different resolution.

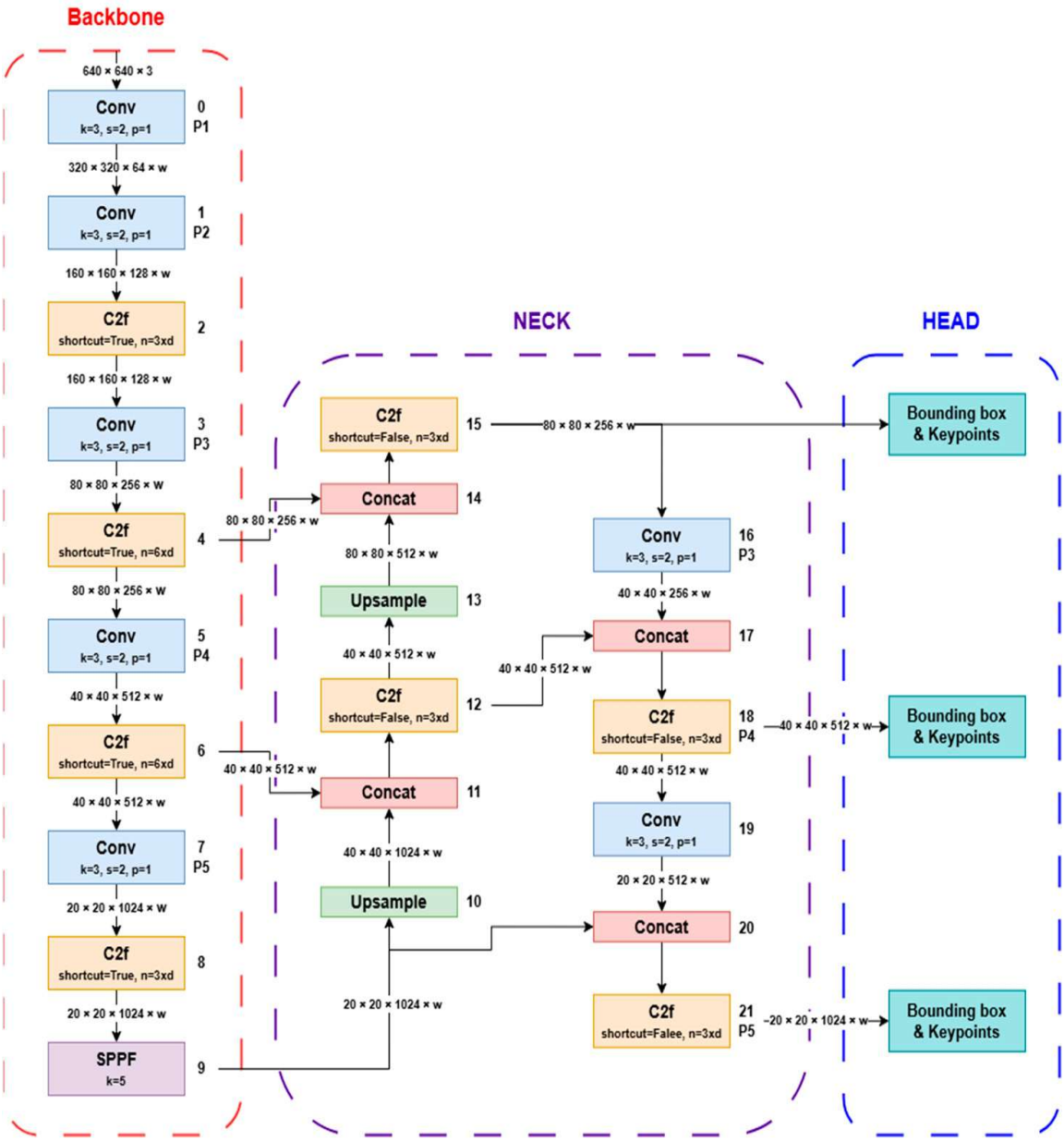


Figure 4.5: YOLOv8Pose architecture used for thermal human pose estimation.

4.4.1 YOLOv8-Pose Overview

There are three parameters of YOLOv8-pose that determine the version: depth_multiple, width_multiple, and max_channels. The depth_multiple parameters decide how many bottleneck blocks are in the C2f block. The width_multiple and max_channels parameters define the output channels. The yolov8 stem is comprised of two convolution blocks with stride 2, kernel size 3. These two blocks create the origins of features and reduce the input

resolution. The stage component in YOLOv8 is structured using the C2f block. The 8 stages are blocks no. 2, 4, 6, 8, 12, 15, 18, and 21. The stages in the backbone (blocks no. 2, 4, 6, and 8) utilize shortcuts while the neck (blocks 12, 15, 18, and 21) does not. Using shortcuts or not is based on seemingly sensible, valid results obtained from trial and error to try to achieve optimal. Downsampling for YOLOv8 is accomplished using a convolution block with a stride of 2 and a kernel size of 3. A stride of 2 will yield an output spatial resolution that is half the size.

After the final block on the backbone, SPPF (Spatial Pyramid Pooling Fast) is used at the neck to give a multi-scale representation from the feature map. When pooling features at different scales, SPPF allows the model to capture features at different levels of abstraction. There are a few concat and upsample blocks on the neck. Upsampling increases the resolution of the feature map. YOLOv8 uses the nearest neighbor technique to conduct upsampling. This method fills the new pixels in a larger feature map by copying the value of neighboring pixels. Feature maps are concatenated with concat. The resolution does not change, however, the number of channels will increase when concatenating feature maps.

YOLOv8 has three heads. The first head is connected to block No. 15 and detects small objects. The second head is connected to block No. 18 and detects medium objects. The third head is connected to block No. 21 and detects large objects. After these predictions, the model applies *Non-Maximum Suppression (NMS)* to remove overlapping boxes and discard low-confidence results. This produces clean and reliable detections.

The overall design of YOLOv8-Pose makes it well-suited for real-time human analysis in thermal imagery, especially in applications like activity monitoring, surveillance, and anomaly detection.

4.4.2 YOLOv8n-pose (Nano)

YOLOv8n-pose is the most compact model within the YOLOv8-Pose family. It is designed for edge devices and applications where speed and low latency are paramount. It has a shallow network with 144 layers and around 3.3 million parameters and provides the highest inference speed but lowest accuracy among the four. It can be used as a real-world baseline for quick prototyping. The model is composed of a backbone, neck, and a pose head tuned specifically for keypoint detection. Its lightweight architecture, having fewer layers carrying fewer parameters, provides faster inference, hence can be used for real-time usage in low-resource settings. The architecture is designed to handle 640×640 thermal frames and produces 17 COCO-style keypoints and bounding box predictions.

```
Ultralytics 8.3.133 🚀 Python-3.10.10 torch-2.7.0+cu128 CUDA:0 (Tesla T4, 14931MiB)
engine/trainer: agnostic_nms=False, amp=True, augment=False, auto_augment=randaugment, batch=32, bgr=0.0, box=7.5,
Overriding model.yaml nc=1 with nc=9

                   from  n    params  module                                       arguments
  0                  -1  1       464  ultralytics.nn.modules.conv.Conv             [3, 16, 3, 2]
  1                  -1  1      4672  ultralytics.nn.modules.conv.Conv             [16, 32, 3, 2]
  2                  -1  1      7360  ultralytics.nn.modules.block.C2f             [32, 32, 1, True]
  3                  -1  1     18560  ultralytics.nn.modules.conv.Conv             [32, 64, 3, 2]
  4                  -1  2     49664  ultralytics.nn.modules.block.C2f             [64, 64, 2, True]
  5                  -1  1     73984  ultralytics.nn.modules.conv.Conv             [64, 128, 3, 2]
  6                  -1  2    197632  ultralytics.nn.modules.block.C2f             [128, 128, 2, True]
  7                  -1  1    295424  ultralytics.nn.modules.conv.Conv             [128, 256, 3, 2]
  8                  -1  1    460288  ultralytics.nn.modules.block.C2f             [256, 256, 1, True]
  9                  -1  1    164608  ultralytics.nn.modules.block.SPPF            [256, 256, 5]
 10                  -1  1         0  torch.nn.modules.upsampling.Upsample         [None, 2, 'nearest']
 11            [-1, 6]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 12                  -1  1    148224  ultralytics.nn.modules.block.C2f             [384, 128, 1]
 13                  -1  1         0  torch.nn.modules.upsampling.Upsample         [None, 2, 'nearest']
 14            [-1, 4]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 15                  -1  1     37248  ultralytics.nn.modules.block.C2f             [192, 64, 1]
 16                  -1  1     36992  ultralytics.nn.modules.conv.Conv             [64, 64, 3, 2]
 17           [-1, 12]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 18                  -1  1    123648  ultralytics.nn.modules.block.C2f             [192, 128, 1]
 19                  -1  1    147712  ultralytics.nn.modules.conv.Conv             [128, 128, 3, 2]
 20            [-1, 9]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 21                  -1  1    493056  ultralytics.nn.modules.block.C2f             [384, 256, 1]
 22       [15, 18, 21]  1   1037494  ultralytics.nn.modules.head.Pose             [9, [17, 3], [64, 128, 256]]
YOLOv8n-pose summary: 144 layers, 3,297,030 parameters, 3,297,014 gradients, 9.3 GFLOPs
```

Figure 4.6: YOLOv8n-Pose architecture used for thermal human pose estimation.

4.4.3 YOLOv8s-pose (Small)

The YOLOv8s-pose model represents a small-scale variant in the YOLOv8-pose family, designed to deliver an effective trade-off between computational efficiency and pose estimation accuracy. This model builds upon the YOLOv8n-pose (nano) architecture by incorporating a deeper structure comprising 144 layers and approximately 11.6 million parameters. These enhancements allow the model to achieve better generalization and robustness, especially in applications involving diverse human poses.

Due to its relatively low computational footprint, YOLOv8s-pose is particularly suitable for real-time pose estimation tasks on resource-constrained hardware. In thermal imaging scenarios, where data often lacks texture and contrast, the additional depth of this model contributes to improved spatial feature extraction and more stable keypoint localization. Compared to larger variants in the YOLOv8 series, the small model maintains a lightweight structure while retaining sufficient capacity to manage typical pose variations in thermal datasets. Its use in this work is motivated by the need for fast, responsive human pose estimation in low-light or no-light conditions, which are characteristic of thermal imaging

environments.

```
Ultralytics 8.3.133 🚀 Python-3.10.10 torch-2.7.0+cu128 CUDA:0 (Tesla T4, 14931MiB)
engine/trainer: agnostic_nms=False, amp=True, augment=False, auto_augment=randaugment, batch=32, bgr=0.0, box=7.5,
Overriding model.yaml nc=1 with nc=9

                   from  n    params  module                                       arguments
  0                  -1  1       928  ultralytics.nn.modules.conv.Conv             [3, 32, 3, 2]
  1                  -1  1     18560  ultralytics.nn.modules.conv.Conv             [32, 64, 3, 2]
  2                  -1  1     29056  ultralytics.nn.modules.block.C2f             [64, 64, 1, True]
  3                  -1  1     73984  ultralytics.nn.modules.conv.Conv             [64, 128, 3, 2]
  4                  -1  2    197632  ultralytics.nn.modules.block.C2f             [128, 128, 2, True]
  5                  -1  1    295424  ultralytics.nn.modules.conv.Conv             [128, 256, 3, 2]
  6                  -1  2    788480  ultralytics.nn.modules.block.C2f             [256, 256, 2, True]
  7                  -1  1   1180672  ultralytics.nn.modules.conv.Conv             [256, 512, 3, 2]
  8                  -1  1   1838080  ultralytics.nn.modules.block.C2f             [512, 512, 1, True]
  9                  -1  1    656896  ultralytics.nn.modules.block.SPPF            [512, 512, 5]
 10                  -1  1         0  torch.nn.modules.upsampling.Upsample         [None, 2, 'nearest']
 11             [-1, 6]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 12                  -1  1    591360  ultralytics.nn.modules.block.C2f             [768, 256, 1]
 13                  -1  1         0  torch.nn.modules.upsampling.Upsample         [None, 2, 'nearest']
 14             [-1, 4]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 15                  -1  1    148224  ultralytics.nn.modules.block.C2f             [384, 128, 1]
 16                  -1  1    147712  ultralytics.nn.modules.conv.Conv             [128, 128, 3, 2]
 17            [-1, 12]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 18                  -1  1    493056  ultralytics.nn.modules.block.C2f             [384, 256, 1]
 19                  -1  1    590336  ultralytics.nn.modules.conv.Conv             [256, 256, 3, 2]
 20             [-1, 9]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 21                  -1  1   1969152  ultralytics.nn.modules.block.C2f             [768, 512, 1]
 22       [15, 18, 21]  1   2609590  ultralytics.nn.modules.head.Pose             [9, [17, 3], [128, 256, 512]]
YOLOv8s-pose summary: 144 layers, 11,629,142 parameters, 11,629,126 gradients, 30.4 GFLOPs
```

Figure 4.7: YOLOv8s-Pose architecture used for thermal human pose estimation.

4.4.4 YOLOv8m-pose (Medium)

The YOLOv8m-pose model is the medium-level architecture in the YOLOv8-pose variants, providing an optimal trade-off between model complexity and performance. It contains 184 layers and around 26.5 million parameters, much more than the nano and small versions. This extra depth and number of parameters allow the model to attain higher accuracy in object detection and keypoint localization problems. For thermal human pose estimation, the medium version is ideally suited for scenarios that require greater accuracy while maintaining a reasonable level of computationally feasible computations. Its advanced architecture enables it to capture more nuanced spatial relationships, which is particularly useful when dealing with low-texture thermal images where pose estimation can prove to be difficult. YOLOv8m-pose is well-suited to systems with mid-range GPUs, where there is sufficient scope for more elaborate computation without making real-time inference impossible. Its generalizability to a broader spectrum of human pose and motion patterns renders it a good option for use in applications like behavior monitoring, anomaly detection, or physical activity recognition within thermal scenarios. This model was used in this work

to investigate the performance benefits that can be realized on thermal data when going past light-weight architectures.

```
Ultralytics 8.3.133 🚀 Python-3.10.10 torch-2.7.0+cu128 CUDA:0 (Tesla T4, 14931MiB)
engine/trainer: agnostic_nms=False, amp=True, augment=False, auto_augment=randaugment, batch=32, bgr=0.0, box=7.5,
Overriding model.yaml nc=1 with nc=9

                   from  n    params  module                                       arguments
  0                  -1  1      1392  ultralytics.nn.modules.conv.Conv             [3, 48, 3, 2]
  1                  -1  1     41664  ultralytics.nn.modules.conv.Conv             [48, 96, 3, 2]
  2                  -1  2    111360  ultralytics.nn.modules.block.C2f             [96, 96, 2, True]
  3                  -1  1    166272  ultralytics.nn.modules.conv.Conv             [96, 192, 3, 2]
  4                  -1  4    813312  ultralytics.nn.modules.block.C2f             [192, 192, 4, True]
  5                  -1  1    664320  ultralytics.nn.modules.conv.Conv             [192, 384, 3, 2]
  6                  -1  4   3248640  ultralytics.nn.modules.block.C2f             [384, 384, 4, True]
  7                  -1  1   1991808  ultralytics.nn.modules.conv.Conv             [384, 576, 3, 2]
  8                  -1  2   3985920  ultralytics.nn.modules.block.C2f             [576, 576, 2, True]
  9                  -1  1    831168  ultralytics.nn.modules.block.SPPF            [576, 576, 5]
 10                  -1  1         0  torch.nn.modules.upsampling.Upsample         [None, 2, 'nearest']
 11             [-1, 6]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 12                  -1  2   1993728  ultralytics.nn.modules.block.C2f             [960, 384, 2]
 13                  -1  1         0  torch.nn.modules.upsampling.Upsample         [None, 2, 'nearest']
 14             [-1, 4]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 15                  -1  2    517632  ultralytics.nn.modules.block.C2f             [576, 192, 2]
 16                  -1  1    332160  ultralytics.nn.modules.conv.Conv             [192, 192, 3, 2]
 17            [-1, 12]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 18                  -1  2   1846272  ultralytics.nn.modules.block.C2f             [576, 384, 2]
 19                  -1  1   1327872  ultralytics.nn.modules.conv.Conv             [384, 384, 3, 2]
 20             [-1, 9]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 21                  -1  2   4207104  ultralytics.nn.modules.block.C2f             [960, 576, 2]
 22       [15, 18, 21]  1   4388470  ultralytics.nn.modules.head.Pose             [9, [17, 3], [192, 384, 576]]
YOLOv8m-pose summary: 184 layers, 26,469,094 parameters, 26,469,078 gradients, 81.4 GFLOPs
```

Figure 4.8: YOLOv8m-Pose architecture used for thermal human pose estimation.

4.4.5 YOLOv8l-pose (Large)

The YOLOv8l-pose model is the biggest and strongest architecture in the YOLOv8-pose series. It has 224 layers and around 44.5 million parameters, leading to a much higher computational requirement of about 168.6 GFLOPs. Although this increases inference time, it also allows the model to have the highest accuracy out of all the variants, especially in keypoint detection as well as complex human poses. Because of its high representational capability, YOLOv8l-pose is particularly suited for use cases where accuracy comes at the cost of speed. In the in the context of thermal imaging, where visual clues are scarce and pose estimation is inherently more difficult, the big model exhibits excellent generalization and accurate keypoint localization.

```
New https://pypi.org/project/ultralytics/8.3.160 available 😃 Update with 'pip install -U ultralytics'
Ultralytics 8.3.133 🚀 Python-3.10.10 torch-2.7.0+cu128 CUDA:0 (NVIDIA L4, 22491MiB)
engine/trainer: agnostic_nms=False, amp=True, augment=False, auto_augment=randaugment, batch=32, bgr=0.0, box=7.5,
Overriding model.yaml nc=1 with nc=9

                   from  n    params  module                                       arguments
  0                  -1  1      1856  ultralytics.nn.modules.conv.Conv             [3, 64, 3, 2]
  1                  -1  1     73984  ultralytics.nn.modules.conv.Conv             [64, 128, 3, 2]
  2                  -1  3    279808  ultralytics.nn.modules.block.C2f             [128, 128, 3, True]
  3                  -1  1    295424  ultralytics.nn.modules.conv.Conv             [128, 256, 3, 2]
  4                  -1  6   2101248  ultralytics.nn.modules.block.C2f             [256, 256, 6, True]
  5                  -1  1   1180672  ultralytics.nn.modules.conv.Conv             [256, 512, 3, 2]
  6                  -1  6   8396800  ultralytics.nn.modules.block.C2f             [512, 512, 6, True]
  7                  -1  1   2360320  ultralytics.nn.modules.conv.Conv             [512, 512, 3, 2]
  8                  -1  3   4461568  ultralytics.nn.modules.block.C2f             [512, 512, 3, True]
  9                  -1  1    656896  ultralytics.nn.modules.block.SPPF            [512, 512, 5]
 10                  -1  1         0  torch.nn.modules.upsampling.Upsample         [None, 2, 'nearest']
 11            [-1, 6]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 12                  -1  3   4723712  ultralytics.nn.modules.block.C2f             [1024, 512, 3]
 13                  -1  1         0  torch.nn.modules.upsampling.Upsample         [None, 2, 'nearest']
 14            [-1, 4]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 15                  -1  3   1247744  ultralytics.nn.modules.block.C2f             [768, 256, 3]
 16                  -1  1    590336  ultralytics.nn.modules.conv.Conv             [256, 256, 3, 2]
 17           [-1, 12]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 18                  -1  3   4592640  ultralytics.nn.modules.block.C2f             [768, 512, 3]
 19                  -1  1   2360320  ultralytics.nn.modules.conv.Conv             [512, 512, 3, 2]
 20            [-1, 9]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 21                  -1  3   4723712  ultralytics.nn.modules.block.C2f             [1024, 512, 3]
 22       [15, 18, 21]  1   6448324  ultralytics.nn.modules.head.Pose             [9, [17, 3], [256, 512, 512]]
YOLOv8l-pose summary: 224 layers, 44,495,364 parameters, 44,495,348 gradients, 169.2 GFLOPs
```

Figure 4.9: YOLOv8l-Pose architecture used for thermal human pose estimation.

All four YOLOv8-Pose models exhibit good performance on the dataset of thermal human pose. Both the small and nano versions are good choices for lightweight deployments, but they sacrifice some accuracy for speed. There is a good balance between the two ends in the medium version. Due to the fact that accuracy gains taper off from m to l, the medium model might be the most effective in many practical situations.

4.4.6 Comparison of Model Variants

All models were evaluated using an input resolution of 640 $\times$ 640 pixels. The table below highlights the key architectural and performance differences between them.

Model	Layers	Params (M)	FLOPs (G)	Training Time (hrs)	Weights Size (MB)
YOLOv8n-pose	144	3.3	9.3	1.97	6.9
YOLOv8s-pose	144	11.6	30.4	3.33	23.5
YOLOv8m-pose	184	26.5	81.4	6.69	53.3
YOLOv8l-pose	224	44.5	169.2	6.28	89.4

Table 4.2: Comparison of validation performance across YOLOv8-Pose models on the Thermal Dataset.

4.4.7 Action Recognition

Once trained, the models are used to infer keypoints from test samples. These keypoint coordinates serve as skeletal descriptors for the underlying human action. By analyzing the spatial configuration and temporal evolution of keypoints, actions are classified based on posture and movement patterns. This stage completes the HAR pipeline, enabling automated recognition of actions in thermal video sequences.

4.5 Comparison of Different Thermal Datasets

Table 4.3: Comparison of Thermal Human Pose Datasets

Dataset	**# Keypoints**	**BBox**	**# Images**	**# People**	**Single/Multi**	**Environment**	**Type**
Mehra et al. [26]	5	+	1,000	N/A	Multi	Indoor	Person Detection
ThermalPose [27]	19	-	2,000	N/A	Single, Multi	Indoor	Person Detection
SLP [28]	14	-	14,715	14,715	Single	Indoor	Person Detection
UCH-Thermal-Pose [29]	17	+	904	1,378	Single, Multi	Indoor, Outdoor	Person Detection
OpenThermalPose [23]	17	+	6,090	14,315	Single, Multi	Indoor, Outdoor	Action Classification
IM-Thermal (custom) [1]	17	+	3,138	3,138	Single	Indoor	Action Classification

The table 4.3 provides a comparative view of thermal human pose datasets publicly available along with the IM-Thermal dataset that has been created. Most datasets, e.g. Mehra et al., ThermalPose, SLP, in particular only focus on person detection or single pose estimation in controlled indoor environments. UCH-Thermal-Pose and OpenThermalPose datasets have extended these features with multi-person annotations and outdoor approaches to image acquisition. OpenThermalPose aims to classify actions related to exercise type activity, but IM-Thermal is developed for general human action recognition. IM-Thermal uses the 17 keypoints from OpenPose, bounding box annotations, and 9 differing action classes, and uses a larger breadth of action's than any particular thermal dataset. There are 3,138 labeled images in the IM-Thermal dataset and the corresponding annotations for the 3,138 and only single person instances, also all collected in indoor environments. IM-Thermal is unlike any thermal dataset as it looks at 9 different classes for general human action recognition, and distinguishes human pose estimation from human behavior classification in a thermal imaging context.

4.6 Training Setup for Anomaly Detection

The YOLOv8n-Pose model was trained using a thermal human pose dataset with 17 keypoints per subject. The model architecture consisted of 144 layers and approximately 3.3 million parameters, with a computational complexity of 9.3 GFLOPs. Pretrained weights were partially loaded from the base model `yolo11n.pt`, transferring layers. The layer `model.22.dfl.conv.weight` was explicitly frozen to preserve its original behavior during training. Training was conducted over 100 epochs with an input image size of 640×640 for both training and validation. Four dataloader workers were used to parallelize data loading. Automatic Mixed Precision (AMP) was enabled to reduce memory consumption and improve speed.

The training set contained 6589 samples and the validation set consisted of 941 images. Optimization was performed using the AdamW optimizer with a learning rate of 0.0001. Parameter groups were divided into 63 weights (no decay), 73 weights (with decay 0.0001), and 72 biases (no decay) to apply selective regularization. All outputs, logs, and label visualizations were saved in the corresponding training directory for further evaluation.

4.7 Libraries and Packages

The implementation of the proposed thermal human pose estimation and action recognition system was carried out using Python, with several essential libraries supporting the entire pipeline. The core detection and pose estimation model was developed using the Ultralytics YOLOv8 framework, known for its efficiency and accuracy. OpenCV was employed for image and video processing tasks, such as frame extraction, resizing, and visualization. Data augmentation techniques like horizontal flipping and Gaussian noise were implemented using the Albumentations library. Numerical computations and label data processing were made easier with NumPy and Pandas. Matplotlib and Seaborn were used to visualize training curves and performance indicators. PyTorch supplied the deep learning backbone, allowing for GPU-accelerated model inference and training. Furthermore, Weights and Biases (WandB) was incorporated to track, log, and monitor experiment performance in real time. These libraries worked together to facilitate effective model training, data preprocessing, and YOLOv8-pose framework evaluation on the thermal dataset.

Chapter 5

IMPLEMENTATION AND EVALUATION METRICS

5.1 Implementation and Parameter Settings

To ensure stable performance, the model hyperparameters needed to be correctly adjusted before the training process began. In order to balance detection accuracy and computational effort across all experiments, an input resolution of 640 × 640 pixels was chosen during this investigation. A mini-batch size of 32 was employed, which strikes a fair balance between model convergence and GPU memory usage. The optimizer utilized Stochastic Gradient Descent (SGD), with weight decay equal to 0.0005 to encourage generalization and momentum equal to 0.9 to aid in learning speed. Each model was trained for 100 epochs until the validation measures converged, with an initial learning rate of 0.01. Several data augmentation techniques, such as random horizontal flip, scaling, mosaic augmentation, and color changes, were used to strengthen the models and reduce overfitting. During training, Automatic Mixed Precision (AMP) was turned on to speed up computation and use less memory.

The NVIDIA Tesla T4 and NVIDIA L4 GPUs with CUDA acceleration were used for all of the tests on the Lightning AI cloud platform. The performance of four YOLOv8 Pose models, YOLOv8n-pose (nano), YOLOv8s-pose (small), YOLOv8m-pose (medium), and YOLOv8l-pose (large) was assessed in this study. The Ultralytics YOLOv8 framework (version 8.3.133) was used to run the models in Python 3.10.10 with PyTorch 2.7.0. 32 GB of RAM and 64-bit Intel Xeon-class CPUs made up the computing environment. During training, essential variables such as precision, recall, mean Average Precision (mAP), and posture estimate accuracy were tracked through automated checkpointing and validation.

5.2 Training and Validation on Thermal-IM Dataset

To guarantee a strong model evaluation and equitable comparison across various YOLOv8 pose variations, the IM-Thermal dataset was meticulously divided. Three subsets of the dataset were specifically created: 70% for training (6,589 photos), 10% for validation (941 images), and 20% for testing (1,884 images). All nine human action classes abnormal, leg stretching, lying, pushing up, sitting, sitting cross-legged, sitting erect, standing, and walking were evenly represented in this stratified category. The models were trained to assess human position keypoints in thermal pictures and identify bounding boxes. After each epoch, the validation set was used to evaluate intermediate performance, adjust hyperparameters, and track learning progress. This method was crucial for reducing overfitting and guaranteeing that the models maintained their strong capacity to generalize to new data. Only the final

evaluation, which measured accuracy and robustness in real-world circumstances, used the reserved test set.

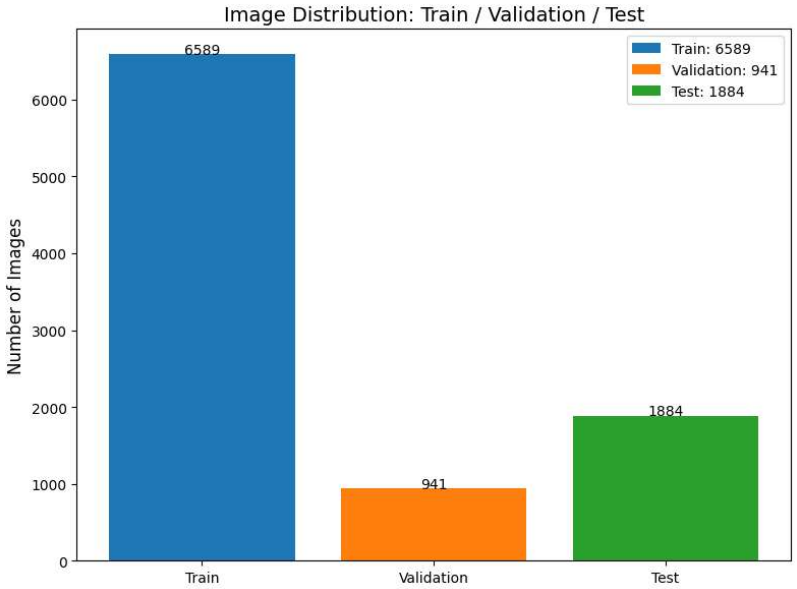


Figure 5.1: Bar Chart of Dataset Split Counts

The distribution of the entire dataset among the test, validation, and training subsets is displayed in Figure 5.1. The way the dataset was arranged to facilitate testing and performance comparison between the YOLOv8n-pose, YOLOv8s-pose, YOLOv8m-pose and YOLOv8l-pose models is made clearer by these visual summary.

5.3 Evaluation Metrics

The quantitative performance of the suggested YOLOv8-Pose model was evaluated with typical object detection and human pose estimation metrics. These metrics give a general idea about the accuracy of the model to detect people and predict anatomical keypoints from thermal images and the computational complexity in real-time applications.

5.3.1 Object Detection Evaluation

The performance of the suggested YOLOv8-Pose model was evaluated based on typical object detection and human pose estimation metrics. The metrics provide a comprehensive assessment of the model's accuracy in detecting humans and estimating anatomical keypoints from thermal images and its computational cost for potential real-time application.

Precision (P) is defined as the number of true positive detections among all the predicted positives:

$$P = \frac{TP}{TP + FP} \tag{5.1}$$

Recall (R) is the ratio of correct positive detections out of all actual ground-truth instances:

$$R = \frac{TP}{TP + FN} \tag{5.2}$$

Where:

• TP: True Positives (correctly detected persons),

• FP: False Positives (incorrect detections),

• FN: False Negatives (Missed Detection).

Intersection over Union (IoU) :To evaluate the localization accuracy of predicted bounding boxes in human detection and pose estimation, the *Intersection over Union (IoU)* metric was employed. IoU measures the degree of overlap between a predicted bounding box and its corresponding ground truth bounding box and is defined as:

$$\text{IoU} = \frac{\text{Area}(B_p \cap B_{gt})}{\text{Area}(B_p \cup B_{gt})} \tag{5.3}$$

where B_p denotes the predicted bounding box and B_{gt} denotes the ground truth bounding box. The numerator represents the area of intersection, and the denominator represents the area of union of both bounding boxes.

IoU values range from 0 to 1. A value of 1 indicates perfect overlap, while 0 indicates no overlap. In this study, mean Average Precision (mAP) was computed over multiple IoU thresholds from 0.50 to 0.95, with an increment of 0.05, to comprehensively assess detection performance under varying degrees of localization strictness.

$$\text{mAP} = \frac{1}{n} \sum_{i=1}^{n} AP_{IoU_i} \tag{5.4}$$

5.3.2 Pose Estimation Evaluation

The accuracy of 17-keypoint human pose estimation was evaluated using the Object Keypoint Similarity (OKS) metric, which measures the similarity between predicted and ground-truth keypoints while accounting for object scale and keypoint visibility.

$$\text{OKS} = \frac{\sum_i \exp\left(-\frac{d_i^2}{2s^2k_i^2}\right) \delta(v_i > 0)}{\sum_i \delta(v_i > 0)} \tag{5.5}$$

To evaluate the model's effectiveness in human pose estimation, the following assessment metrics were calculated based on the Object Keypoint Similarity (OKS): Mean Average Precision (mAP), calculated across different OKS thresholds; Average Recall (AR), which

quantifies the percentage of visible keypoints accurately predicted by the model; and AP_{50} and AP_{95}, which stand for Average Precision at OKS thresholds of 0.50 and 0.95, respectively. d is the Euclidean distance between the predicted and ground-truth keypoints in the OKS formulation, s stands for the object scale (bounding box area), $\delta(\cdot)$ is the indicator function, and k_i is a keypoint-specific falloff constant, v_i indicates the visibility of the keypoint. Together, these measures offer a thorough assessment of the model's precision and reliability in identifying human keypoints in thermal pictures.

5.4 Loss Function

The total loss function combined bounding box loss, classification loss, distribution focal loss, pose keypoint loss, and keypoint objectness loss. The final loss was computed as:

$$\text{Total Loss} = \lambda_{\text{box}} \cdot \text{BoxLoss} + \lambda_{\text{cls}} \cdot \text{clsLoss} + \lambda_{\text{dfl}} \cdot \text{dflLoss} + \lambda_{\text{poseLoss}} \cdot \text{pose} + \lambda_{\text{kobj}} \cdot \text{kobjLoss} \tag{5.6}$$

Where, λ represents the loss weights used to balance each component in the total loss: λ_{box} for bounding box loss, λ_{cls} for classification loss, λ_{dfl} for distribution focal loss, λ_{pose} pose keypoint loss, and λ_{kobj} for keypoint objectness loss.

Chapter 6

RESULT AND ANALYSIS

This chapter provides a thorough evaluation of the proposed human action recognition and anomaly detection on the Thermal dataset. The evaluation uses various common metrics to evaluate performance in identifying human actions and to evaluate the performance of the method for subject localization.

6.1 Dataset Characteristics

Before evaluating the model's performance, it is crucial to understand the underlying data distribution, as this can significantly influence the results. The dataset consists of nine classes: **Abnormal**, **Leg_streching**, **Lying**, **Push-ups**, **Sitting**, **Sitting_crosslegs**, **Sit-ups**, **Standing**, **Walking**.

6.1.1 Class Instance Distribution

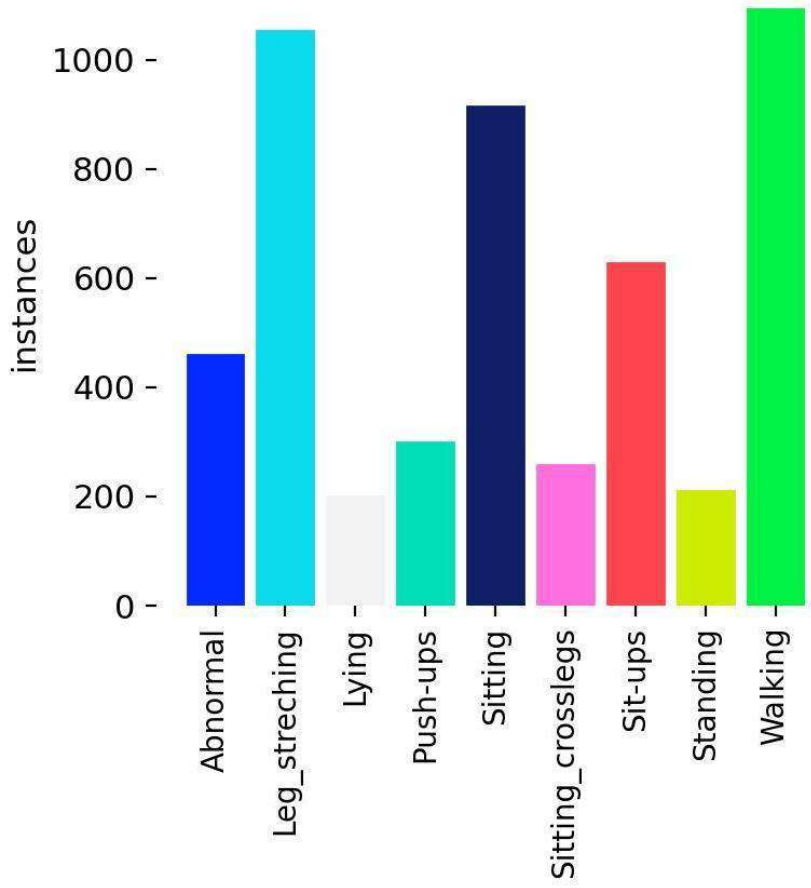


Figure 6.1: Analysis of Number of instances per class.

As shown in Figure 6.1, the dataset exhibits a notable class imbalance. The classes **Leg_streching** and **Walking** are the most represented, each with over 1000 instances. In contrast, classes like **Push-ups** and **Sit-ups** have significantly fewer instances (around

200-300). This imbalance presents a challenge for the model, as it could potentially develop a bias towards the majority classes. The model's ability to perform well on minority classes will be a key indicator of its learning capacity.

6.2 Confusion Matrix Analysis

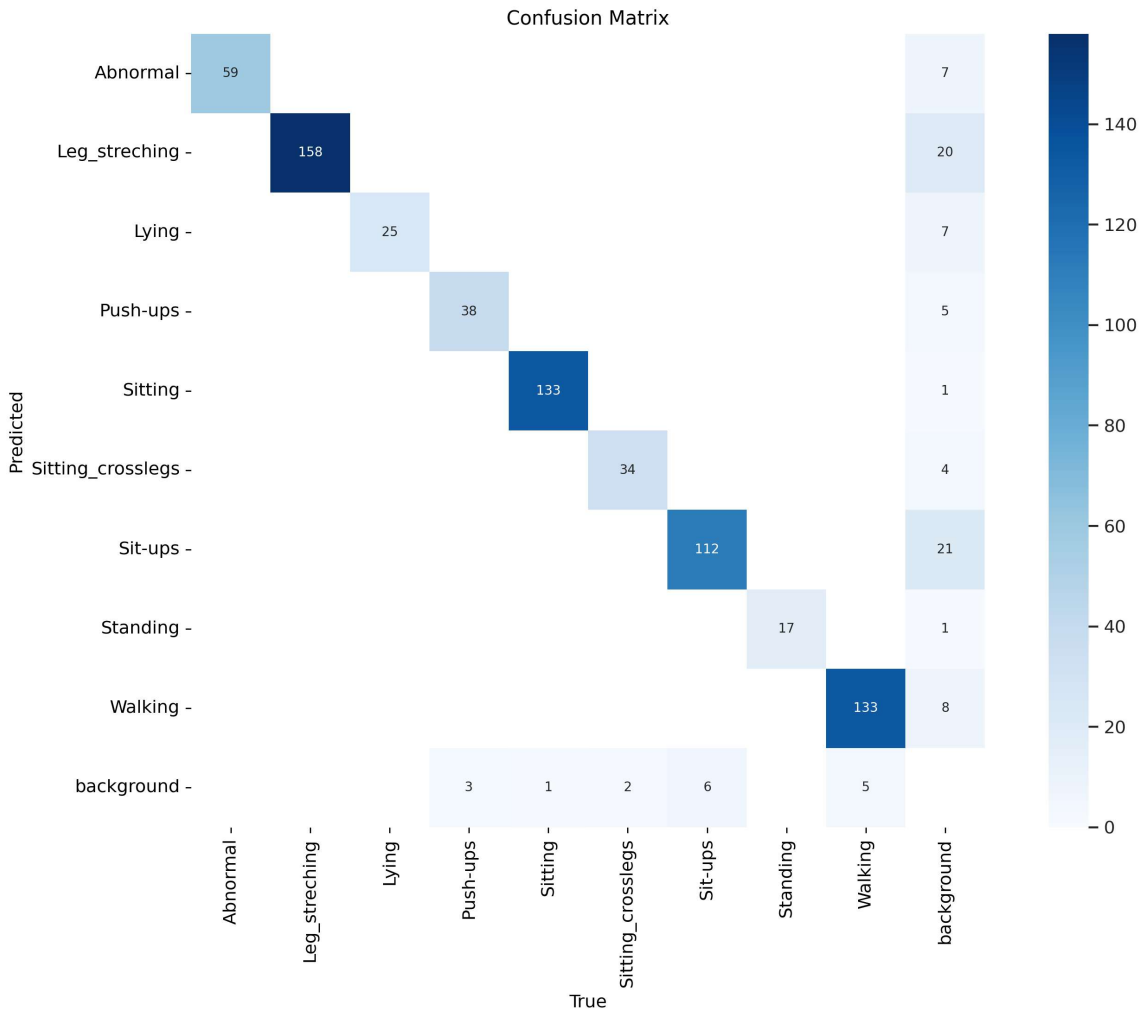


Figure 6.2: The confusion matrix with prediction counts.

The confusion matrix provides a detailed breakdown of classification performance, showing which classes are correctly identified and which are confused with others. Figure 6.2 shows the absolute number of predictions made by the model for each class. The **True** labels are on the x-axis, and the **Predicted** labels are on the y-axis. The numbers on the main diagonal represent **correct predictions**. The model exhibits outstanding performance for the majority of classes. The dark blue diagonal shows the percentage of correct predictions for each class and the lighter-colored cells off the diagonal reveal the model's mistakes.

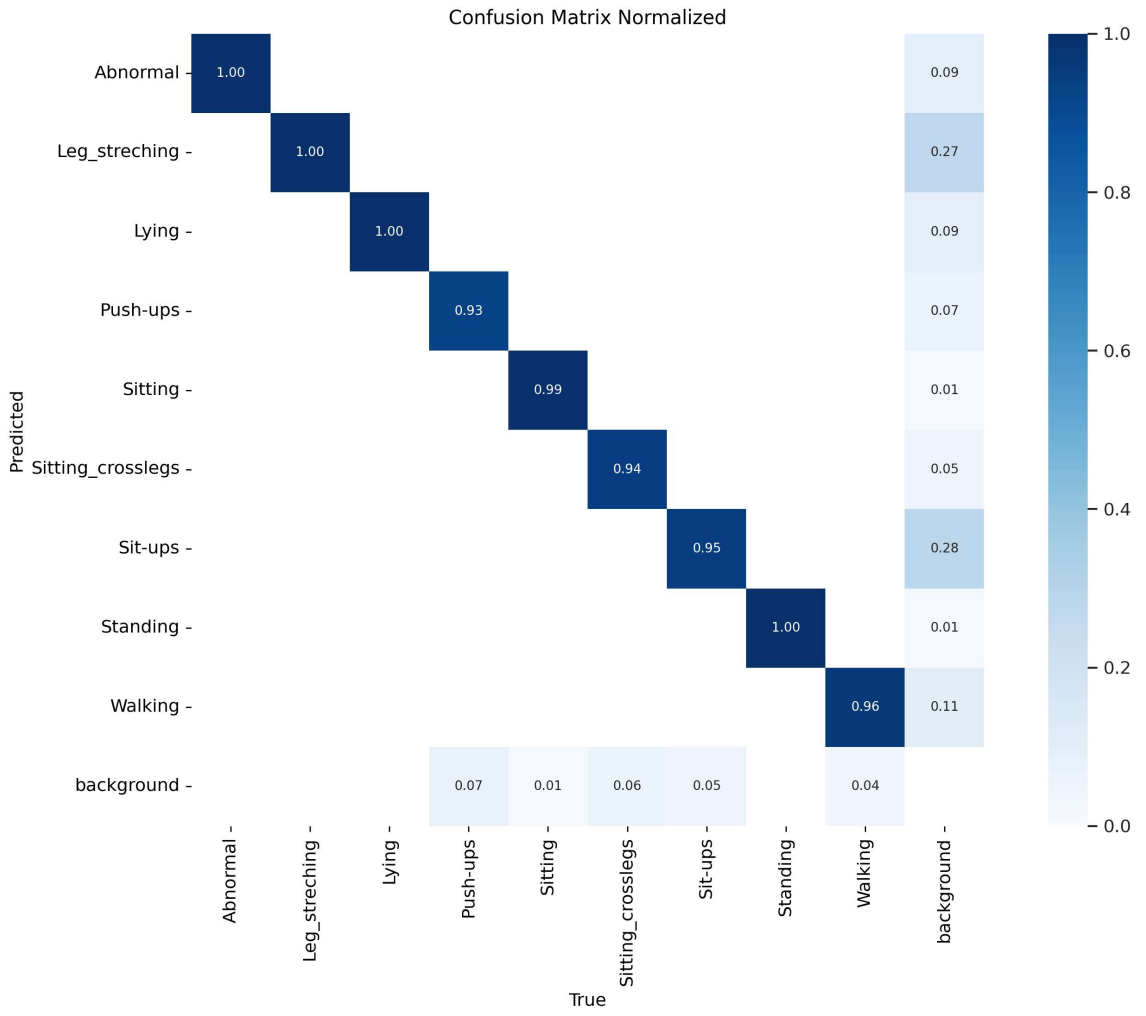


Figure 6.3: The normalized confusion matrix with prediction count.

6.3 Model Training and Validation

The model was trained for 100 epochs, and its performance was monitored on both the training and validation datasets to ensure successful learning and prevent overfitting. Figure 6.4 shows the evolution of key loss components and performance metrics over the course of the training process. The top row shows training metrics, and the bottom row shows validation metrics. Loss components decrease while performance metrics increase and converge, indicating a successful training process. The total loss combines **box_loss**, **pose_loss**, and **cls_loss**, along with auxiliary terms like **dfl_loss** and **kobj_loss**, which guide the model in learning accurate bounding boxes, keypoints, and action classifications.

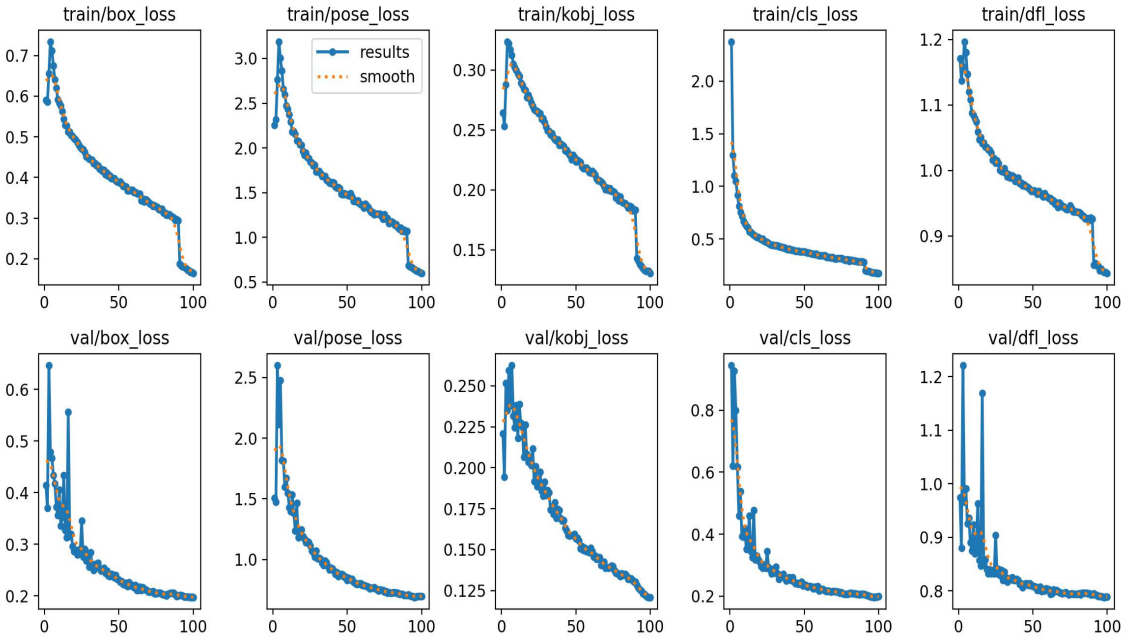


Figure 6.4: Training and validation history over 100 epochs.

Alongside the loss, performance metrics were tracked throughout the training process. The graphs show the evolution of precision, recall, and mean Average Precision (mAP). On both the training and validation sets, these metrics exhibit a rapid increase during the early epochs and then plateau at high performance levels.

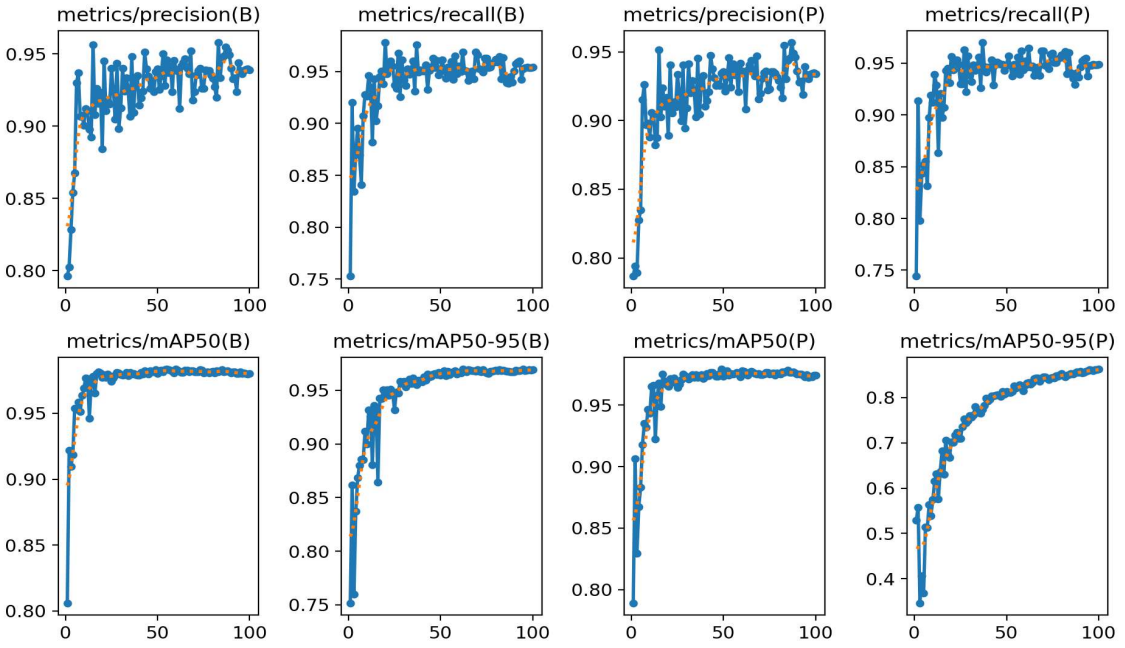


Figure 6.5: evaluation metrics for the YOLOv8l-pose model over 100 epoch

6.4 Experimental Results

This section presents the performance evaluation of four YOLOv8-Pose models (YOLOv8n, YOLOv8s, YOLOv8m, and YOLOv8l) over 100 training epochs. The evaluation is based on standard object detection and pose estimation metrics, including precision, recall, and mean Average Precision (mAP) for both bounding box detection and keypoint (pose) estimation.

6.4.1 Bounding Box Detection Metrics

The following figures illustrate the performance of the models in detecting bounding boxes.

Figure 6.6: Bounding Box Detection Precision vs. Epochs.

Figure 6.6 shows the precision for bounding box detection. All models exhibit some performance fluctuation during training but ultimately converge to a high precision score of approximately 0.93–0.94.

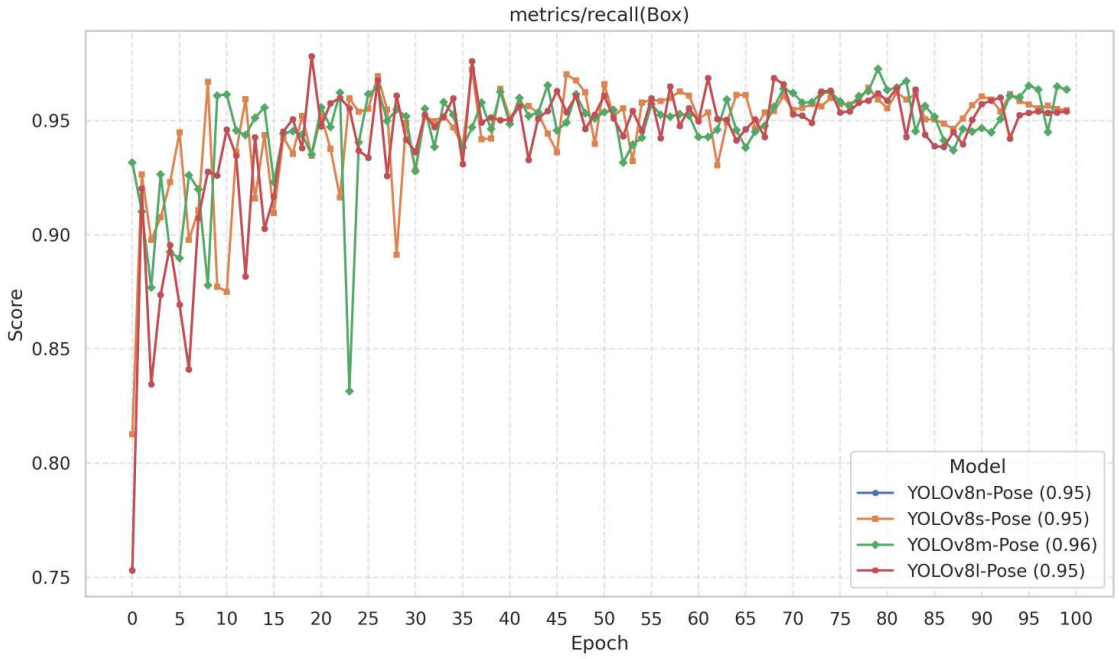


Figure 6.7: Bounding Box Detection Recall vs. Epochs.

Figure 6.7 illustrates the recall for bounding box detection. The models consistently achieve a high recall rate throughout the training process, stabilizing around a score of 0.95 to 0.96.

Figure 6.8: Bounding Box Detection mAP (IoU 0.50) vs. Epochs.

Figure 6.8 presents the mean Average Precision at an IoU threshold of 0.50. The models rapidly achieve and sustain an exceptional mAP50 score of approximately 0.98, indicating

strong performance from early in the training.

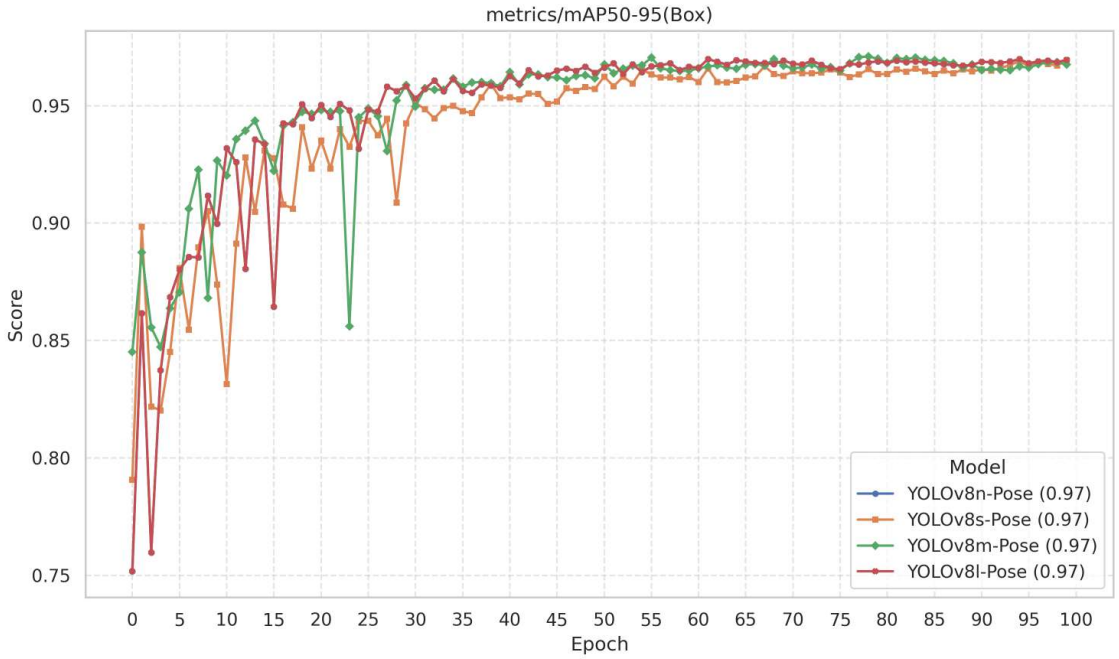


Figure 6.9: Bounding Box Detection mAP (IoU 0.50-0.95) vs. Epochs.

Figure 6.9 displays the primary mAP metric, averaged over IoU thresholds from 0.50 to 0.95. All models show consistent improvement and converge to a final score of about 0.97.

6.4.2 Pose Estimation Metrics

The following figures illustrate the performance of the models in the task of pose estimation.

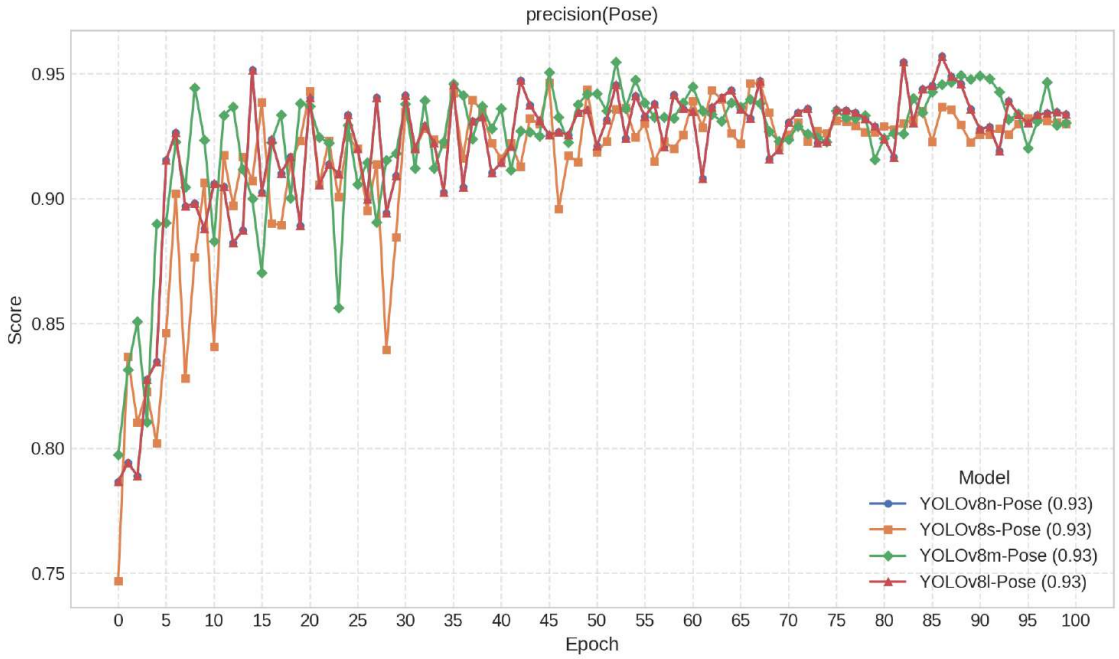


Figure 6.10: Pose Estimation Precision vs. Epochs.

Figure 6.10 shows the precision for pose estimation. The score for all models is volatile but trends towards a final value of approximately 0.93, indicating high accuracy for the detected keypoints.

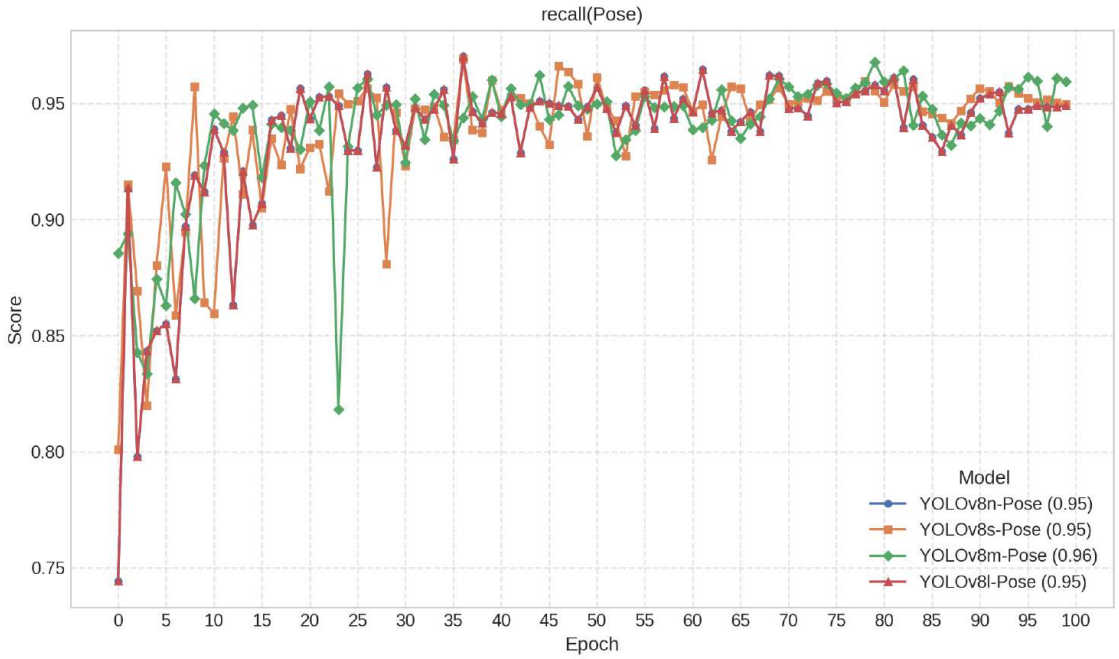


Figure 6.11: Pose Estimation Recall vs. Epochs.

Figure 6.11 illustrates the recall for pose estimation. After some initial instability, all four models stabilize and maintain a high recall score, fluctuating between 0.94 and 0.96.

Figure 6.12: Pose Estimation mAP (IoU 0.50) vs. Epochs.

Figure 6.12 presents the mean Average Precision for pose estimation at a 0.50 IoU threshold. All models quickly reach and maintain a high performance level, stabilizing at a score between 0.97 and 0.98.

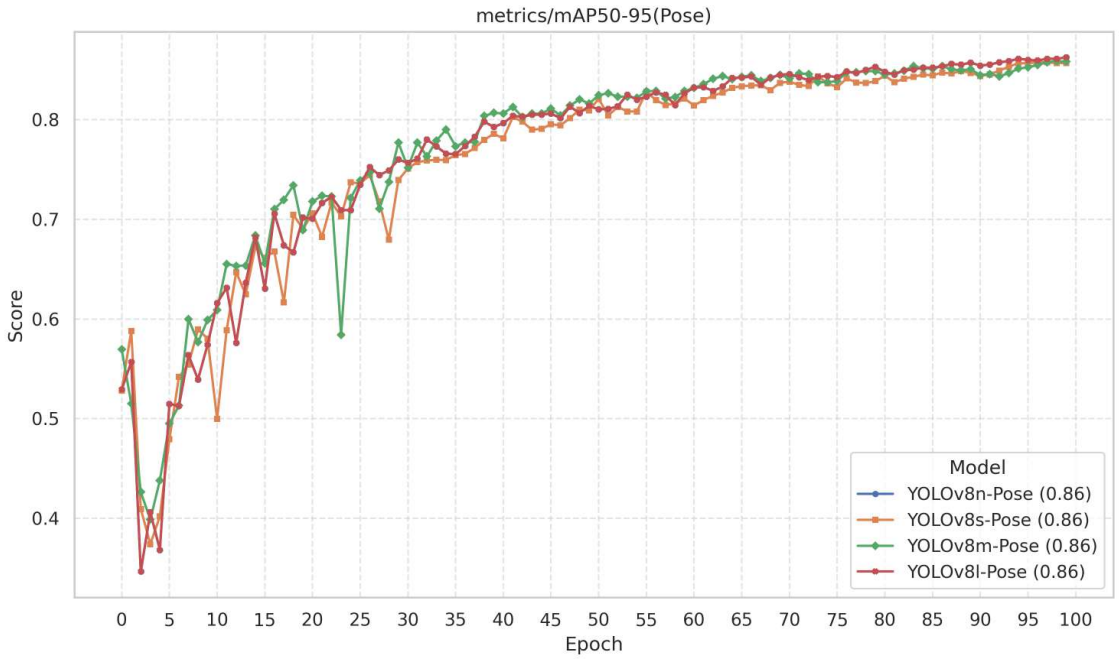


Figure 6.13: Pose Estimation mAP (IoU 0.50-0.95) vs. Epochs.

Figure 6.13 The models demonstrate steady and consistent improvement over 100 epochs converging at a final score of 0.86 mAP metric for pose estimation.

6.5 YOLOv8-Pose Validation Results on Thermal-IM Dataset

A comparative analysis of four YOLOv8-pose model variants (nano, small, medium, and large) was conducted. The results reveal a uniformly high level of performance with negligible differences between the model sizes. For the initial task of subject localization, all variants achieved an identical and excellent Box mAP@0.5 of **0.98**. Similarly, for the final Pose based action classification, accuracy remained outstanding, with a mAP@0.5 score between **0.97** and **0.98** for all models. A marginal advantage for the larger models was only observed in the stricter Pose mAP@0.5–0.95 metric. These results demonstrate the model's robustness and accuracy in detecting human poses from thermal imagery under challenging indoor conditions.

Table 6.1: Validation performance metrics of the YOLOv8n-pose model on the Thermal-IM dataset.

Class	Box				Pose			
	Precision	Recall	mAP@0.5	mAP@0.5–0.95	Precision	Recall	mAP@0.5	mAP@0.5–0.95
All	0.936	0.960	0.982	0.966	0.931	0.955	0.975	0.852
Abnormal	0.936	0.996	0.991	0.969	0.936	0.996	0.991	0.877
Leg streching	0.938	0.959	0.989	0.983	0.938	0.959	0.989	0.922
Lying	0.913	0.960	0.981	0.972	0.913	0.960	0.981	0.899
Push-ups	0.903	0.909	0.943	0.897	0.903	0.909	0.940	0.734
Sitting	0.993	0.994	0.995	0.985	0.978	0.979	0.988	0.914
Sitting crosslegs	0.914	0.944	0.985	0.985	0.914	0.944	0.974	0.850
Sit-ups	0.948	0.920	0.978	0.955	0.948	0.920	0.978	0.693
Standing	0.924	1.000	0.992	0.976	0.924	1.000	0.992	0.957
Walking	0.957	0.956	0.986	0.971	0.928	0.927	0.946	0.822

Table 6.2: Validation performance metrics of the YOLOv8s-pose model on Thermal-IM dataset.

Class	Box				Pose			
	Precision	Recall	mAP@0.5	mAP@0.5–0.95	Precision	Recall	mAP@0.5	mAP@0.5–0.95
All	0.935	0.955	0.981	0.969	0.930	0.950	0.974	0.857
Abnormal	0.936	0.986	0.992	0.978	0.936	0.986	0.992	0.895
Leg streching	0.936	0.968	0.989	0.982	0.936	0.968	0.989	0.923
Lying	0.908	0.960	0.980	0.972	0.908	0.960	0.980	0.878
Push-ups	0.922	0.861	0.934	0.907	0.922	0.861	0.924	0.733
Sitting	0.994	0.993	0.995	0.989	0.979	0.978	0.988	0.915
Sitting crosslegs	0.915	0.944	0.984	0.984	0.915	0.944	0.976	0.879
Sit-ups	0.936	0.915	0.973	0.951	0.936	0.915	0.973	0.699
Standing	0.920	1.000	0.995	0.985	0.920	1.000	0.995	0.956
Walking	0.945	0.964	0.987	0.975	0.916	0.935	0.952	0.834

Table 6.3: Validation performance metrics of the YOLOv8m-pose model on the Thermal-IM dataset.

Class	Box				Pose			
	Precision	Recall	mAP@0.5	mAP@0.5–0.95	Precision	Recall	mAP@0.5	mAP@0.5–0.95
All	0.934	0.965	0.983	0.969	0.929	0.961	0.976	0.859
Abnormal	0.913	1.000	0.991	0.976	0.913	1.000	0.991	0.901
Leg streching	0.933	0.968	0.988	0.981	0.933	0.968	0.988	0.929
Lying	0.925	0.985	0.990	0.981	0.925	0.985	0.990	0.897
Push-ups	0.919	0.902	0.941	0.907	0.919	0.902	0.930	0.720
Sitting	0.996	0.993	0.995	0.987	0.989	0.985	0.991	0.923
Sitting crosslegs	0.913	0.944	0.984	0.984	0.913	0.944	0.976	0.874
Sit-ups	0.931	0.915	0.975	0.950	0.931	0.915	0.975	0.704
Standing	0.920	1.000	0.995	0.983	0.920	1.000	0.995	0.955
Walking	0.951	0.976	0.983	0.968	0.922	0.947	0.945	0.831

Table 6.4: Validation performance metrics of the YOLOv8l-pose model on the Thermal-IM dataset.

Class	Box				Pose			
	Precision	Recall	mAP@0.5	mAP@0.5–0.95	Precision	Recall	mAP@0.5	mAP@0.5–0.95
All	0.939	0.954	0.980	0.969	0.934	0.949	0.974	0.862
Abnormal	0.935	0.983	0.991	0.976	0.935	0.983	0.991	0.887
Leg streching	0.940	0.975	0.989	0.985	0.940	0.975	0.989	0.929
Lying	0.920	0.960	0.981	0.973	0.920	0.960	0.981	0.915
Push-ups	0.922	0.861	0.934	0.912	0.922	0.861	0.934	0.735
Sitting	0.996	0.993	0.995	0.987	0.989	0.985	0.993	0.919
Sitting crosslegs	0.912	0.944	0.982	0.982	0.911	0.944	0.975	0.869
Sit-ups	0.952	0.907	0.970	0.946	0.942	0.898	0.967	0.698
Standing	0.922	1.000	0.992	0.988	0.922	1.000	0.992	0.961
Walking	0.950	0.964	0.986	0.974	0.921	0.935	0.946	0.845

Table 6.5: Comparison of Detection and Pose Estimation Performance Across Different YOLOv8-Pose Models on Thermal-IM dataset.

Model	Box				Pose			
	Precision	Recall	mAP@0.5	mAP@0.5–0.95	Precision	Recall	mAP@0.5	mAP@0.5–0.95
YOLOv8n-pose	0.94	0.96	0.98	0.97	0.93	0.96	0.98	0.85
YOLOv8s-pose	0.94	0.95	0.98	0.97	0.93	0.95	0.97	0.86
YOLOv8m-pose	0.93	0.97	0.98	0.97	0.93	0.96	0.98	0.86
YOLOv8l-pose	0.94	0.95	0.98	0.97	0.93	0.95	0.97	0.86

6.6 Trained Model Test Result on Thermal-IM dataset

The proposed human pose estimation framework was evaluated on a thermal image test set using the **YOLOv8l-Pose** model. The model consisted of 121 layers with 44.47 million parameters and a computational complexity of 168.6 GFLOPs. A total of 1884 thermal images were used for evaluation, including 1478 images containing human instances and 406 background-only images.

Table 6.6: YOLOv8l-Pose Evaluation Metrics on Thermal-IM Test Set

Class	Images	Box				Pose			
		Precision	Recall	mAP@0.5	mAP@0.5–0.95	Precision	Recall	mAP@0.5	mAP@0.5–0.95
All	1884	0.943	0.962	0.986	0.972	0.938	0.956	0.979	0.833
Abnormal	133	0.948	0.932	0.985	0.963	0.948	0.932	0.985	0.824
Leg_streching	308	0.938	0.981	0.988	0.983	0.931	0.974	0.987	0.905
Lying	57	0.950	0.994	0.990	0.976	0.950	0.994	0.990	0.904
Push-ups	87	0.928	0.884	0.974	0.944	0.928	0.884	0.970	0.736
Sitting	275	0.977	0.996	0.995	0.985	0.973	0.993	0.994	0.913
Sitting_crosslegs	71	0.914	0.972	0.990	0.988	0.901	0.958	0.974	0.851
Sit-ups	196	0.943	0.925	0.974	0.953	0.943	0.925	0.974	0.673
Standing	64	0.923	1.000	0.988	0.982	0.923	1.000	0.988	0.899
Walking	287	0.969	0.973	0.987	0.976	0.941	0.945	0.946	0.794

Table 6.6 summarizes the performance of the YOLOv8l-Pose model on the thermal

test dataset across nine action classes. The model achieved high overall accuracy, with a bounding box mAP@0.5 of **0.986** and pose mAP@0.5 of **0.979**, while the more stringent mAP@0.5–0.95 scores were **0.972** for box and **0.833** for pose estimation. Actions with less motion such as Sitting pose mAP@0.5–0.95 **0.913**, Lying **0.904**, and Leg_streching **0.905** showed strong results, indicating the model's ability to localize static poses accurately. On the other hand, more dynamic actions like Push-ups **0.736** and Sit-ups **0.673** had relatively lower pose accuracy, likely due to rapid motion and pose complexity. The model maintained consistent performance across most categories, confirming its robustness for pose estimation in thermal imagery.

Table 6.7: Best posture detection results per class, displaying predicted and original image together with their respective confidence scores

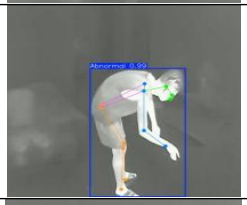
Original_Image	Predicted_Image	Class	Confidence Score
		**Lying**	**0.9902**
		**Sitting**	**0.9806**
		**Walking**	**0.9833**
		**Leg_streching**	**0.9808**
		**Abnormal**	**0.9878**
		**Sitting_crosslegs**	**0.9863**
		**Sit-ups**	**0.9769**
		**Push-ups**	**0.9895**
		**Standing**	**0.9868**

Table 6.7 shows a collection of thermal images that demonstrate the YOLOv8-Pose model's qualitative performance. In every example, the original input frame is accompanied with the expected output with bounding boxes and pose estimates superimposed, the predicted action class, and the associated confidence score. With confidence scores between 0.9769 and 0.9902, the model shows a high degree of confidence in its predictions across a variety of activities. Because of body occlusions in thermal imaging, the model is able to reliably identify complicated poses that are usually difficult. Highest confidence was found for the Lying class (0.9902), closely followed by Abnormal activities (0.9878) and Push-ups (0.9895). The model is a viable contender for real-world surveillance and anomaly detection in thermal imagery because of these strong predictions, which validate its accuracy in reading human behavior in low-visibility conditions.

6.7 YOLOv8n-Pose Model Evaluation Results for Anomaly Detection

6.7.1 Model Training and Evaluation

This section details the results from the training and validation of the model. The analysis includes a review of the performance metrics over 100 epochs, an examination of the training and validation loss curves, and an evaluation of the final classification performance using confusion matrices.

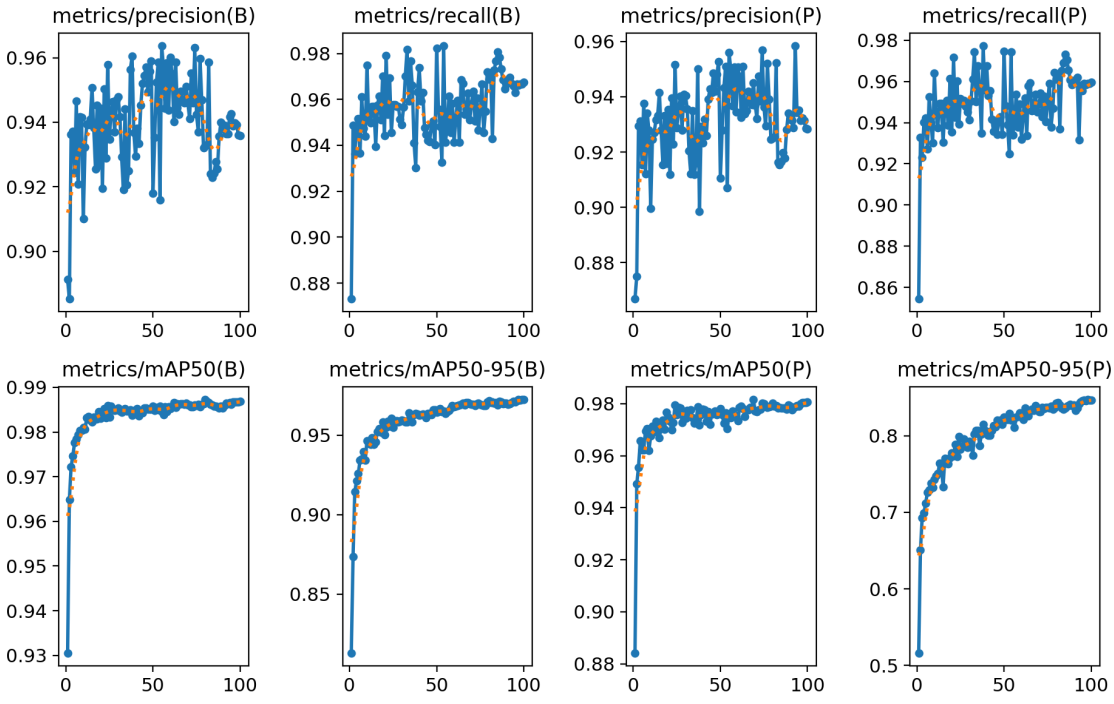


Figure 6.14: Model performance metrics for bounding box (B) and pose (P) estimation over 100 epochs.

Figure 6.14 shows the model's performance on key metrics. There is a clear trend of rapid improvement followed by stabilization at high values. Metrics such as precision, recall, and mAP@0.5 reach excellent final values (approx. 0.94, 0.96, and 0.98 respectively), while the stricter mAP@0.5-0.95 for pose estimation steadily improves to approximately 0.85.

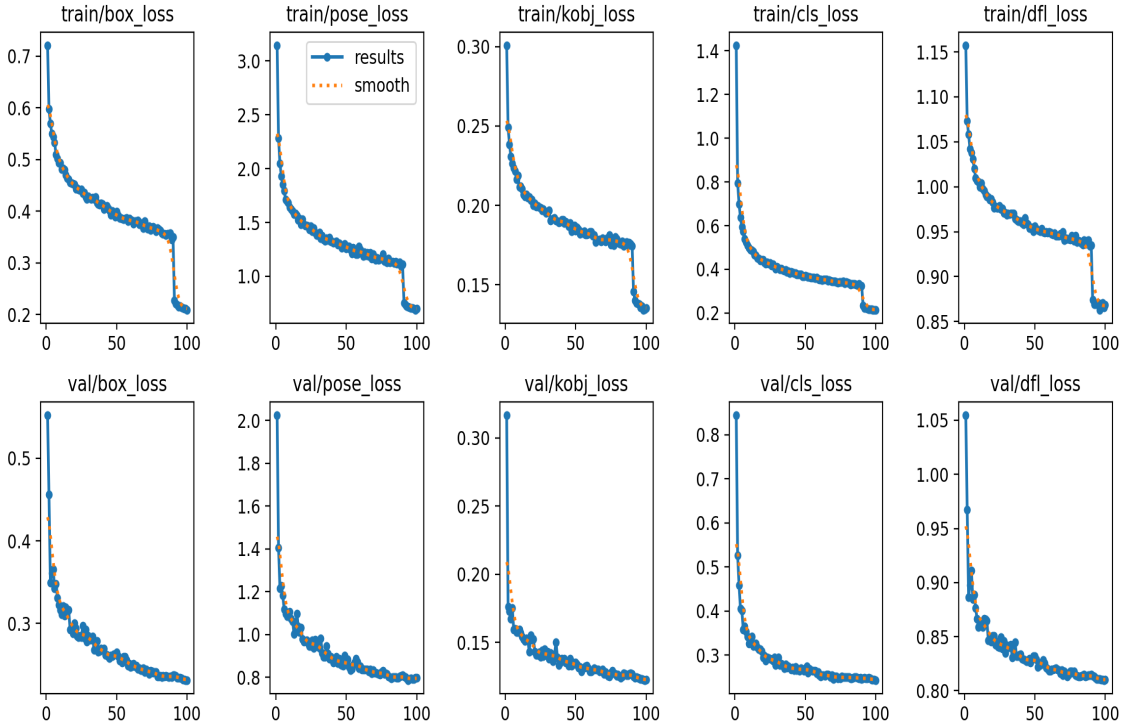


Figure 6.15: Training and validation loss curves for all model components over 100 epochs.

As shown in Figure 6.15, all training and validation loss curves display a sharp decrease in the initial epochs before converging steadily. This pattern indicates that the model learned the task effectively and generalized well to the validation data without significant overfitting.

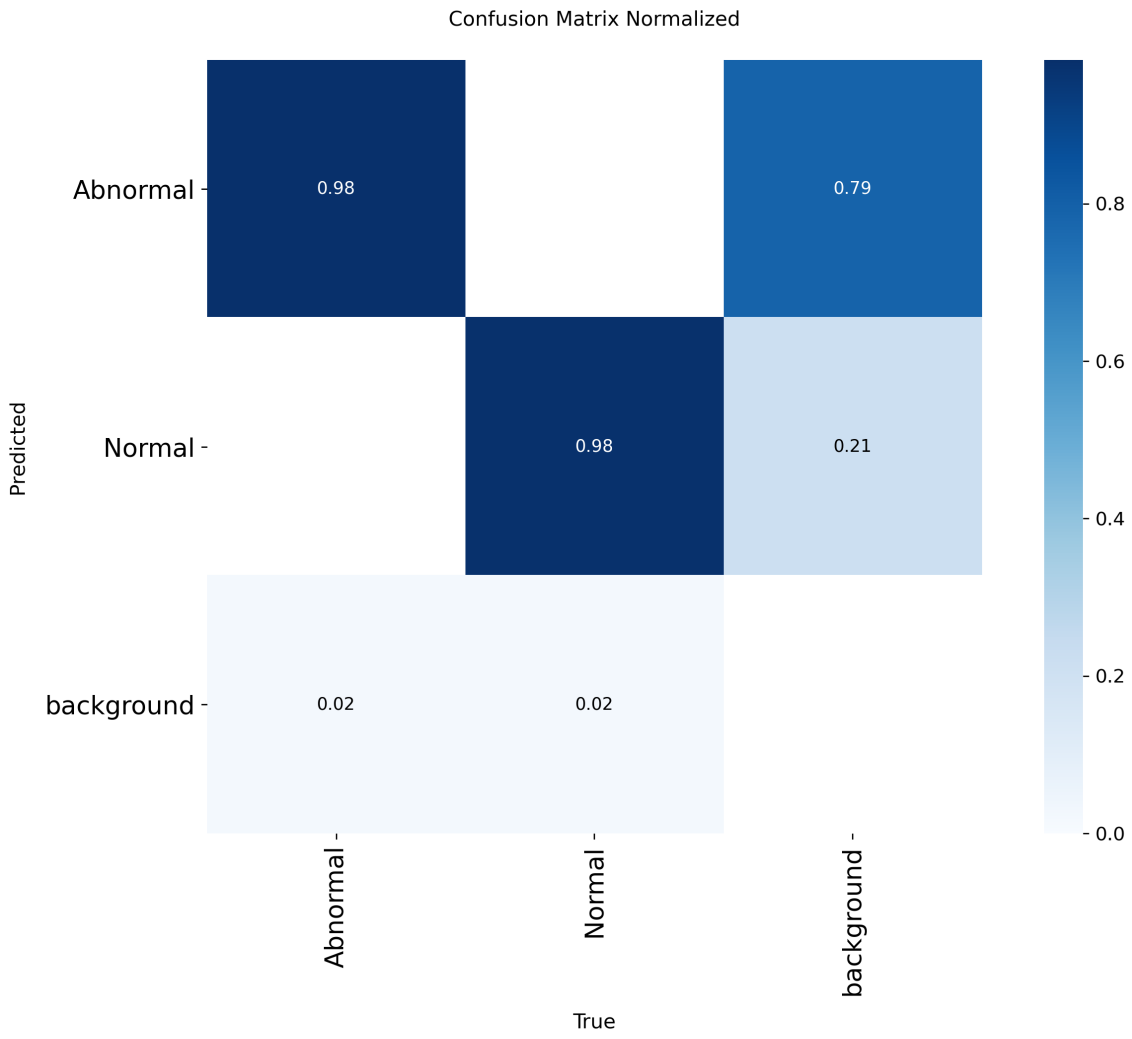


Figure 6.16: Normalized confusion matrix.

The model demonstrates high accuracy for the primary classes, correctly identifying 98% of true Abnormal and 98% of true "Normal" instances. However, the model struggles with the background class, misclassifying 79% of its instances as Abnormal.

Table 6.8: Evaluation results of YOLOv8n-Pose on the IM-Thermal validation set.

Class	**Box**				**Pose**			
	Precision	Recall	mAP@0.5	mAP@0.5:0.95	Precision	Recall	mAP@0.5	mAP@0.5:0.95
All	0.939	0.967	0.987	0.972	0.932	0.959	0.980	0.848
Abnormal	0.918	0.955	0.983	0.966	0.918	0.955	0.983	0.824
Normal	0.961	0.978	0.991	0.978	0.946	0.963	0.978	0.871

The model achieved a mean Average Precision (mAP) of **0.980** for keypoints at IoU threshold 0.5 and **0.848** at 0.5:0.95. For bounding boxes, it reached **0.987** and **0.972** respectively. Per-class metrics indicate consistent performance across both *Abnormal* and

Normal categories, demonstrating the model's robustness under thermal imaging conditions.

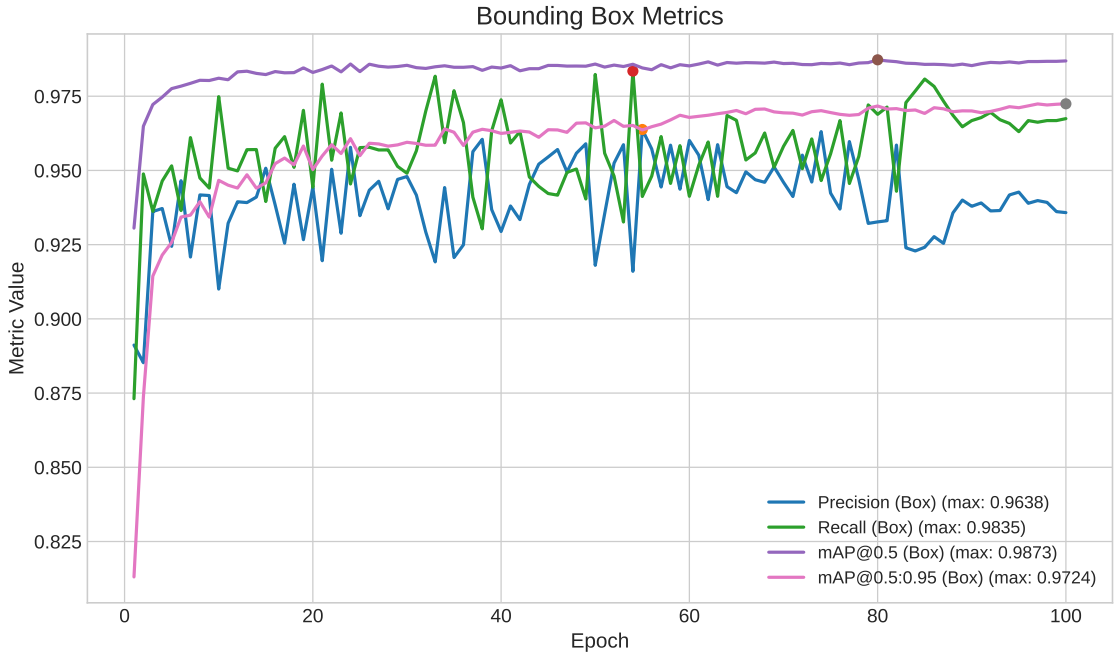


Figure 6.17: Bounding Box Metrics for YOLOv8n-Pose models on Thermal-IM dataset.

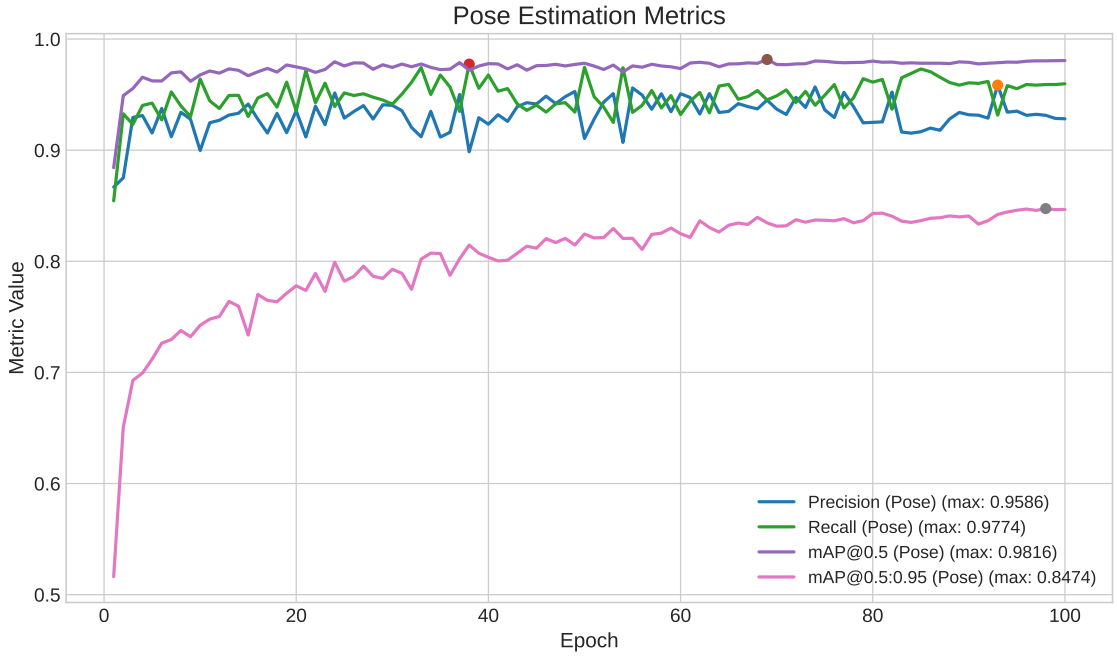
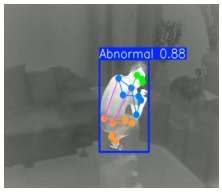
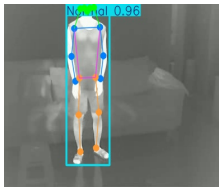
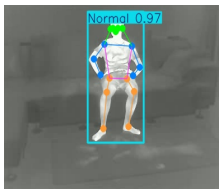
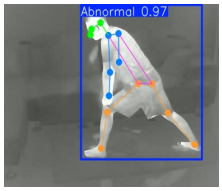
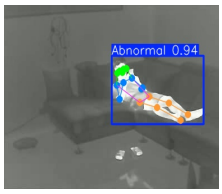
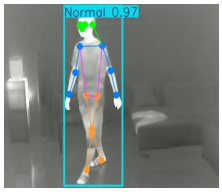


Figure 6.18: Pose Estimation Metrics for YOLOv8n-Pose models on Thermal-IM dataset.

The bounding box detection achieved a **precision of 0.9638**, **recall of 0.9835**, **mAP@0.5 of 0.9873**, and **mAP@0.5:0.95 of 0.9724** (Figure 6.17). Similarly,pose estimation metrics peaked at a **precision of 0.9586**, **recall of 0.9774**, **mAP@0.5 of 0.9816**, and **mAP@0.5:0.95 of 0.8474** indicating high confidence in joint localization.(Figure 6.18)

Table 6.9: Class-wise best pose detection results showing original and predicted images with corresponding confidence scores

Original Image	Predicted Image	Class	Confidence Score
		Abnormal	0.8797
		Normal	0.9485
		Normal	0.9633
		Normal	0.9722
		Abnormal	0.9657
		Abnormal	0.9364
		Normal	0.9742

The result images presented in Table 6.9 further validate these findings. Each entry in the table shows the original frame, predicted frame with keypoints, predicted class, and confidence score. These results confirm that the integration of pose estimation significantly enhances anomaly detection capabilities in thermal videos. The model proves to be a promising solution for surveillance scenarios.

Chapter 7

CONCLUSION AND FUTURE WORK

7.1 Conclusion

This thesis revealed that YOLOv8-Pose models have a tremendous potential for human action recognition and anomaly detection with thermal images derived from a custom dataset (IM-Thermal). Of the YOLOv8 variants that were considered YOLOv8n-pose offered the best performance to computational cost ratio, thus ideally suited for real-time, or other resource-limited applications. The action recognition task with the IM-Thermal dataset offered consistently high results, with the YOLO model achieving bounding box mAP@0.5 and pose mAP@0.5 metrics both at 0.98, and mAP@0.5:0.95 metrics both in the 0.96 to 0.97 range. The anomaly detection task accurately separated the human behaviors into normal or abnormal, achieving mAP@0.5 scores of 0.99 for the normal actions, and 0.98, for the abnormal actions, while maintaining a good level of pose estimation metrics. Although the pose mAP@0.5:0.95 metric for abnormal actions dropped to 0.824, there was still good overall detection quality. These performances show that utilizing thermal imaging with pose estimation found on the IM-Thermal custom dataset provides a sound, privacy-friendly, real-time method to monitor human behaviour (or other behaviours) in real time and with limited constraints, while still providing some degree of physical privacy regardless of the environment.

7.2 Future Work

While the current model performs well, there are several directions to improve and extend this work:

- **Temporal Analysis:** Integrate time-series models to better understand motion patterns and improve anomaly detection accuracy.
- **Real-Time Deployment:** Optimize the model for deployment on edge devices with limited resources by reducing model size and inference time.
- **Multi-Person Detection:** Extend the system to handle multiple individuals in a scene, which is critical for real-world surveillance scenarios.
- **Dataset Expansion:** Incorporate more diverse thermal data from different environments to improve model generalization.

APPENDIX

CODE USED IN THESIS

Step 1: Extract Frames and Clips From Thermal Dataset

```
import os
import json
import cv2

dataset_path = "/content/dataset/dataset/"
output_root = "/content/dataset1"
frames_output_dir = os.path.join(output_root, "frames")
videos_output_dir = os.path.join(output_root, "clips")
os.makedirs(frames_output_dir, exist_ok=True)
os.makedirs(videos_output_dir, exist_ok=True)
json_files = [f for f in os.listdir(dataset_path) if f.endswith(".json")]
for json_file in json_files:
    json_path = os.path.join(dataset_path, json_file)
    video_path = json_path.replace(".json", ".mp4")
    video_cap = cv2.VideoCapture(video_path)
    if not video_cap.isOpened():
        print(f"❌ Failed to open video: {video_path}")
        continue
    fps = video_cap.get(cv2.CAP_PROP_FPS)
    with open(json_path, 'r') as f:
        data = json.load(f)
    base_filename = os.path.splitext(json_file)[0]
    for frame_no in range(int(video_cap.get(cv2.CAP_PROP_FRAME_COUNT))):
        ret, frame = video_cap.read()
        if not ret:
            break
        frame_filename = f"{base_filename}_frame{frame_no}.jpg"
        frame_output_path = os.path.join(frames_output_dir, frame_filename)
        cv2.imwrite(frame_output_path, frame)
    for entry in data:
        label = entry["label"]
        start_frame = int(entry["segment"][0])
        end_frame = int(entry["segment"][1])
        clip_filename = f"{base_filename}_{label}_{start_frame}_{end_frame}.mp4"
        clip_output_path = os.path.join(videos_output_dir, clip_filename)
        fourcc = cv2.VideoWriter_fourcc(*'mp4v')
        height, width = int(video_cap.get(4)), int(video_cap.get(3))
        out = cv2.VideoWriter(clip_output_path, fourcc, fps, (width, height))
        video_cap.set(cv2.CAP_PROP_POS_FRAMES, start_frame)
        for _ in range(start_frame, end_frame + 1):
            ret, frame = video_cap.read()
            if not ret:
                break
            out.write(frame)
        out.release()
    video_cap.release()
print("\n    All frames and action videos extracted to /content/dataset1 successfully!")
```

 # Step 2: Data Preprocessing (Augmentation)

```
import os
from glob import glob
from PIL import Image
from torchvision import transforms
from torchvision.transforms.functional import to_pil_image
from tqdm import tqdm
import torch

original_dataset_path = '/content/.../Thermal_frames/frames'
augmented_output_path = '/content/.../Thermal_frames_augmented_combined'
output_size = (640, 640)

class AddGaussianNoise(object):
    def __init__(self, mean=0., std=0.1):
        self.mean = mean
        self.std = std
    def __call__(self, tensor):
        return tensor + torch.randn(tensor.size()) * self.std + self.mean

for class_name in tqdm(sorted(os.listdir(original_dataset_path))):
    ...
```

Step 2.1: Visualize Original and Augmented

```
import matplotlib.pyplot as plt
from PIL import Image
import torchvision.transforms as transforms
import torchvision.transforms.functional as F
import torch

image_path = '/content/.../Walking/11_frame375.jpg'
original = Image.open(image_path).convert("RGB")

def add_gaussian_noise(img, mean=0., std=0.2):
    tensor_img = F.to_tensor(img)
    noisy_tensor = tensor_img + torch.randn(tensor_img.size()) * std + mean
    noisy_tensor = torch.clamp(noisy_tensor, 0., 1.)
    return F.to_pil_image(noisy_tensor)

resized = F.resize(original, (640, 640))
noisy = add_gaussian_noise(resized)

fig, axs = plt.subplots(1, 3, figsize=(10, 6))
axs[0].imshow(original)
axs[0].set_title('Original')
axs[1].imshow(resized)
axs[1].set_title('Resized')
axs[2].imshow(noisy)
axs[2].set_title('Noisy + Resized')
```

```
for ax in axs: ax.axis('off')
plt.tight_layout()
plt.show()
```

Step 3: Feature Extraction

3.1: Resize All Frames

```
import os
from PIL import Image
from tqdm import tqdm
image_dir = '/content/.../Thermal_frames_combined'
output_size = (640, 640)
for action_folder in tqdm(os.listdir(image_dir)):
    folder_path = os.path.join(image_dir, action_folder)
    for img_file in os.listdir(folder_path):
        img_path = os.path.join(folder_path, img_file)
        img = Image.open(img_path)
        img_resized = img.resize(output_size)
        img_resized.save(img_path)
```

3.2: YOLOv8 Predict One Image

```
from ultralytics import YOLO
import cv2
import matplotlib.pyplot as plt
model = YOLO("yolov8m-pose.pt")
image_path = "/content/.../Sitting/11_frame110_noisy.jpg"
results = model.predict(source=image_path, conf=0.7, save=False)
result_img = results[0].plot()
plt.imshow(cv2.cvtColor(result_img, cv2.COLOR_BGR2RGB))
plt.title("YOLOv8 Detection Result")
plt.show()
```

3.3: Feature Extractions in YOLO Format

```
import os
from pathlib import Path
from ultralytics import YOLO
from PIL import Image
from tqdm import tqdm

source_dir = Path('/content/.../Thermal_frames_combined')
output_images_dir = Path('/content/.../dataset_thermal/images')
output_labels_dir = Path('/content/.../dataset_thermal/labels')
model = YOLO("yolov8m-pose.pt")

class_names = [...]
label_encoder = {name: idx for idx, name in enumerate(class_names)

def convert_to_yolo_format(bbox, keypoints, image_size):
    ...

for class_folder in tqdm(source_dir.iterdir()):
```

#Step 4: Dataset Split

4.1: Train-Val-Test Split on Thermal Dataset

```
import os
import shutil
import random
from pathlib import Path
from tqdm import tqdm
from concurrent.futures import ThreadPoolExecutor

base_path = '/content/.../dataset_thermal'
output_base = '/content/.../dataset_thermal_split'
splits = {'train': 0.7, 'val': 0.1, 'test': 0.2}
image_files = sorted([...])
random.shuffle(image_files)
split_map = { ... }

def copy_pair(filename, split):
    ...

for split, files in split_map.items():
    …
```

Step 5: Training Model on Yolov8Pose

5.1Train YOLOv8n-Pose Model on Thermal Human Pose Dataset

```
# Install required packages
!pip install ultralytics tensorboard
# Import
from ultralytics import YOLO
import os
# Load the model
model = YOLO('yolov8n-pose.yaml').load('yolov8n-pose.pt')
# Train the model
model.train(

data='/teamspace/studios/this_studio/dataset_thermal_split/dataset_thermal_split/data
set.yaml',
    epochs=100,
    imgsz=640,
    batch=32,
)
```

5.2: Train YOLOv8s-Pose Model on Thermal Human Pose Dataset

```
# Install required packages
!pip install ultralytics tensorboard

# Import
from ultralytics import YOLO
import os
```

```
# Load the model
model = YOLO('yolov8s-pose.yaml').load('yolov8s-pose.pt')

# Train the model
model.train(

data='/teamspace/studios/this_studio/dataset_thermal_split/dataset_thermal_split/data
set.yaml',
   epochs=100,
   imgsz=640,
   batch=32,
)
```

5.3: Train YOLOv8m-Pose Model on Thermal Human Pose Dataset

```
# Install required packages
!pip install ultralytics tensorboard

# Import
from ultralytics import YOLO
import os

# Load the model
model = YOLO('yolov8m-pose.yaml').load('yolov8m-pose.pt')

# Train the model
model.train(

data='/teamspace/studios/this_studio/dataset_thermal_split/dataset_thermal_split/data
set.yaml',
   epochs=100,
   imgsz=640,
   batch=32,
)
```

5.4: Train YOLOv8l-Pose Model on Thermal Human Pose Dataset

```
# Install required packages
!pip install ultralytics tensorboard

# Import
from ultralytics import YOLO
import os

# Load the model
model = YOLO('yolov8l-pose.yaml').load('yolov8l-pose.pt')

# Train the model
model.train(
```

```
data='/teamspace/studios/this_studio/dataset_thermal_split/dataset_thermal_split/data
set.yaml',
   epochs=100,
   imgsz=640,
   batch=32,
)
```

#Step6:Validation on Thermal Human Pose Dataset

```
from ultralytics import YOLO

# Load a model
# you can choose any version of yolov8-pose/yolo11-pose
model = YOLO('yolov8n-pose.pt')  # load an official model
model = YOLO('/teamspace/studios/this_studio/runs/pose/train4/weights/best.pt')  #
load a custom model

# Validate the model
metrics = model.val()  # no arguments needed, dataset and settings remembered
metrics.box.map    # map50-95
metrics.box.map50  # map50
metrics.box.map75  # map75
metrics.box.maps   # a list contains map50-95 of each category
```

Step 7: Evaluate YOLOv8-Pose Model on Test Set and Save Metrics

```
!pip install -q ultralytics
from ultralytics import YOLO
import pandas as pd
import os

# configuration
MODEL_PATH =
'/teamspace/studios/this_studio/runs/pose/train19/weights/best.pt'
DATA_YAML_PATH =
'/teamspace/studios/this_studio/dataset_thermal_split/dataset_thermal_spl
it/dataset.yaml'
CSV_OUTPUT_PATH =
'/teamspace/studios/this_studio/yolov8pose_test_metrics.csv'

# load model
model = YOLO(MODEL_PATH)

# evaluate on test set
metrics = model.val(data=DATA_YAML_PATH, split='test')
```

```
# extract metrics correctly using method calls
results_dict = {
    'Metric': [
        'Box mAP@50',
        'Box mAP@50-95',
        'Box Precision (mp)',
        'Box Recall (mr)',
        'Pose mAP@50',
        'Pose mAP@50-95',
        'Pose Precision (mp)',
        'Pose Recall (mr)'
    ],
    'Value': [
        round(metrics.box.map50(), 4),
        round(metrics.box.map(), 4),
        round(metrics.box.mp(), 4),
        round(metrics.box.mr(), 4),
        round(metrics.kpts.map50(), 4),
        round(metrics.kpts.map(), 4),
        round(metrics.kpts.mp(), 4),
        round(metrics.kpts.mr(), 4)
    ]
}

# save to csv
df_metrics = pd.DataFrame(results_dict)
df_metrics.to_csv(CSV_OUTPUT_PATH, index=False)

# print metrics table
print("\n✅ yolo pose evaluation metrics (test set):")
print(df_metrics)
```

 # Step 7: Predict on Full Video

```
from ultralytics import YOLO
import cv2

model_path = '/content/.../yolopose_v8n_thermal/weights/best.pt'
model = YOLO(model_path)

video_path = '/content/44.mp4'
output_path = '/content/.../1.mp4'

cap = cv2.VideoCapture(video_path)
```

```
out = cv2.VideoWriter(output_path, cv2.VideoWriter_fourcc(*'mp4v'), 20, (640, 640))

while True:
    ret, frame = cap.read()
    if not ret:
        break
    resized_frame = cv2.resize(frame, (640, 640))
    results = model.predict(source=resized_frame, conf=0.8, save=False,
verbose=False)
    annotated_frame = results[0].plot()
    out.write(annotated_frame)
cap.release()
out.release()
```

REFERENCES

[1] Z. Tang, W. Ye, W. Ma, and H. Zhao, "What happened 3 seconds ago? inferring the past with thermal imaging," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023, pp. 17 111–17 120.

[2] E. Samkari, M. Arif, M. Alghamdi, and M. A. A. Ghamdi, "Human pose estimation using deep learning: A systematic literature review," *Machine Learning and Knowledge Extraction*, vol. 5, no. 4, pp. 1612–1659, 2023.

[3] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, "Openpose: Realtime multi-person 2d pose estimation using part affinity fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 1, pp. 172–186, 2021.

[4] K. Sun, B. Xiao, D. Liu, and J. Wang, "Deep high-resolution representation learning for human pose estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 5686–5696.

[5] X. Zhou, D. Wang, and P. Krähenbühl, "Objects as points," *arXiv preprint arXiv:1904.07850*, 2019.

[6] Y. Xu, J. Zhang, Q. Zhang, and D. Tao, "Vitpose: Simple vision transformer baselines for human pose estimation," in *Advances in Neural Information Processing Systems*, 2022.

[7] G. Jocher, A. Chaurasia, and J. Qiu, "Ultralytics yolov8," https://github.com/ultralytics/ultralytics, 2023.

[8] H.-S. Fang, J. Li, H. Tang, C. Xu, H. Zhu, Y. Xiu, Y.-L. Li, and C. Lu, "Alphapose: Whole-body regional multi-person pose estimation and tracking in real-time," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 6, pp. 7157–7173, 2023.

[9] P. Srihari, "Spatio-temporal information for action recognition in thermal video using deep learning model," *International Journal of Electrical and Computer Engineering Systems*, vol. 13, no. 8, pp. 669–680, 2022.

[10] C. Solomon and T. Breckon, *Fundamentals of Digital Image Processing: A Practical Approach with Examples in Matlab*. John Wiley & Sons, 2011.

[11] S. Manssor, S. Sun, M. Abdalmajed, and S. Ali, "Real-time human detection in thermal infrared imaging at night using enhanced tiny-yolov3 network," *Journal of Real-Time Image Processing*, vol. 19, pp. 261–274, 2022.

[12] J. Imran and B. Raman, "Deep residual infrared action recognition by integrating local and global spatio-temporal cues," *Infrared Physics & Technology*, vol. 102, p. 103014, 2019.

[13] M. Krišto, M. Ivašić-Kos, and M. Pobar, "Thermal object detection in difficult weather conditions using yolo," *IEEE Access*, vol. 8, pp. 125 459–125 476, 2020.

[14] G. Batchuluun, J. Kang, D. Nguyen, T. Pham, M. Arsalan, and K. Park, "Action recognition from thermal videos using joint and skeleton information," *IEEE Access*, vol. 9, pp. 11 716–11 733, 2021.

[15] M. Ding, Y. Ding, X. Wu, X. Wang, and Y. Xu, "Action recognition of individuals on an airport apron based on tracking bounding boxes of the thermal infrared target," *Infrared Physics & Technology*, vol. 117, p. 103859, 2021.

[16] Y. Liu and S. Ostadabbas, "Slp: A dataset for in-bed pose estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2020.

[17] Y. Liu, Z. Shao, and S. Ostadabbas, "Multimodal in-bed human pose estimation under blankets," in *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2021, pp. 2249–2258.

[18] C. Chen, W. Xie, Y. Yang, and Y. Liu, "Thermalpose: Estimating human pose from thermal images using self-supervised multi-modal learning," arXiv preprint arXiv:2109.10199, 2021.

[19] V. Kniaz, R. Mizginov, and S. Afonin, "Thermalgan: Multimodal color-to-thermal image translation for person re-identification in multispectral dataset," in *Proceedings of the European Conference on Computer Vision Workshops (ECCVW)*, 2018.

[20] D. Mehra, S. Suri, and R. Gupta, "Fusion of thermal and depth data for enhanced human pose estimation," in *International Conference on Computer Vision Systems (ICVS)*, 2022.

[21] E. Gebhardt and M. Wolf, "Camel dataset for visual and thermal infrared multiple object detection and tracking."

[22] S. Hwang, J. Park, N. Kim, Y. Choi, and I. S. Kweon, "Multispectral pedestrian detection: Benchmark dataset and baselines," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[23] A. Kuzdeuov, D. Taratynova, A. Tleuliyev, and H. A. Varol, "Openthermalpose: An open-source annotated thermal human pose dataset and initial yolov8-pose baselines," in *2024 IEEE 18th International Conference on Automatic Face and Gesture Recognition (FG)*, 2024, pp. 1–8.

[24] X. Jia, C. Zhu, M. Li, W. Tang, and W. Zhou, "Llvip: A visible-infrared paired dataset for low-light vision," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 3496–3504.

[25] M. Cormier, C. N. Zhi Yi, A. Specker, B. Blaß, M. Heizmann, and J. Beyerer, "Leveraging thermal imaging for robust human pose estimation in low-light vision," in *Proceedings of the Asian Conference on Computer Vision*, 2024, pp. 67–83.

[26] R. Mehra, M. Chetty, and J. K. Kamalu, "Multiperson pose estimation using thermal and depth modalities," Stanford University, Tech. Rep., 2017.

[27] I.-C. Chen, C.-J. Wang, C.-K. Wen, and S.-J. Tzou, "Multi-person pose estimation using thermal images," *IEEE Access*, vol. 8, pp. 174 964–174 971, 2020.

[28] S. Liu, X. Huang, N. Fu, C. Li, Z. Su, and S. Ostadabbas, "Simultaneously-collected multimodal lying pose dataset: Enabling in-bed human pose monitoring," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 1, pp. 1106–1118, 2023.

[29] J. Smith, P. Loncomilla, and J. Ruiz-Del-Solar, "Human pose estimation using thermal images," *IEEE Access*, vol. 11, pp. 35 352–35 370, 2023.

PLAGIARISM REPORT

Human Anomaly Detection In Thermal Image Using Deep learning

By DHANANJAY KUMAR PRASAD

Submission Date: 31st July 2025, 10:48 AM (UTC+05:30)

Submission ID: 2723147576

File Name: Dhananjay_Dissertation_Final_Thesis31072025.pdf (3.44M)

Word Count: 11,999

Character Count: 62,915

14%	10%	8%	8%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Shri GS Institute of Technology and Science, Indore Student Paper	4%
2	www.mdpi.com Internet Source	3%
3	Submitted to Heriot-Watt University Student Paper	1%
4	Mickael Cormier, Caleb Ng Zhi Yi, Andreas Specker, Benjamin Blaß, Michael Heizmann, Jürgen Beyerer. "Chapter 5 Leveraging Thermal Imaging forRobust Human Pose Estimation inLow-Light Vision", Springer Science and Business Media LLC, 2025 Publication	<1%
5	Javier Smith, Patricio Loncomilla, Javier Ruiz-Del-Solar. "Human Pose Estimation Using Thermal Images", IEEE Access, 2023 Publication	<1%
6	Submitted to Liverpool John Moores University Student Paper	<1%
7	www.nature.com Internet Source	<1%
8	Submitted to Central University of Rajasthan Student Paper	<1%
9	Submitted to University of Lancaster Student Paper	<1%

Figure 1: Plagiarism Report