Eindhoven University of Technology

MASTER

Activity Recognition of Office Space Users using Thermopile Array Sensor

Agni, Shravan N

Award date:
2020

Link to publication

Department of Mathematics and Computer Science
System Architecture and Networking Research Group

Activity Recognition of Office Space Users using Thermopile Array Sensor

Master Thesis report

Shravan N Agni

Supervisors:
dr. Tanir Ozcelebi
dr. Qingzhi Liu (Wageningen University and Research)
Jan Ekkel (Signify Netherlands B.V.)
Harry Broers (Signify Netherlands B.V.)
Vladimir Osin (Signify Netherlands B.V.)

Version 1

Eindhoven, October 2020

Abstract

Human activity recognition is an interesting topic and has attracted a lot of researchers because of its applications in video surveillance, healthcare, human-computer interaction etc. In office spaces, human activity recognition system can assist facility managers to smartly utilize office spaces thereby reducing costs on space and energy. Present solutions for activity recognition include RGB camera-based systems which offer high accuracy but suffer from privacy issues. In addition to that processing high resolution RGB images is computation heavy. Another solution is wearable sensor-based which require users to wear the sensors all the time which may cause discomfort to some of the users.

Commercial availability of thermopile array sensors has made it possible to use them in activity recognition systems. With the use of these sensors, present limitations can be mitigated. Low-resolution sensors can be used to preserve privacy and also processing of low-resolution thermal images is computation friendly.

Most of the research in the human activity recognition domain using thermopile array sensors are concentrated on single subject and to the best of our knowledge, solutions to multiple people activity recognition are not available. The main challenge when considering more than one person is the simultaneous detection of each individual's activity. This challenge composes of two tasks i.e. localization of each individual and classification of each individual's activity. In this research, a deep learning solution is proposed by re-purposing YOLO framework to localize and detect each individual's activity. This solution achieves mAP of 0.76 which shows that it is possible to predict each individual's activity in multi-human setting using a thermopile array sensor. To the best of our knowledge, this is the first of its kind solution for such a problem based on low-resolution thermal images and hence we cannot compare it with any other solutions. Further, we extend our solution for a specific case where the heat profiles of people merge in the generated thermal images when they come close to each other. During this case, it is difficult to estimate the number of people and their postures. Our solution increases mAP from 0.38 to 0.70 for this specific case.

Acknowledgements

Throughout this thesis, I had a great deal of support and assistance (especially with the ongoing pandemic) without which this thesis would not have been completed.

First of all, I would like to thank my supervisor, Dr. Tanir Ozcelebi for his invaluable guidance throughout my thesis. Next, I would like to thank Dr. Qingzhi Liu for his insights and continuous support. Our weekly meetings always pushed my thinking capabilities and helped me take this work to next level.

I would like to thank Signify Netherlands B.V. for providing me this opportunity. I would like to thank Jan Ekkel, Harry Broers and Vladimir Osin for their continuous support, insights and ideas throughout the project. The discussions we had during our meetings and the suggestions I got proved monumental towards the success of this work. I also express my gratitude to Joanna Gibas for helping me to collect the required data.

Finally, I must express my very profound gratitude to my family and friends. Most importantly, to my parents, my sister and brother-in-law for providing me with unfailing support and continuous encouragement throughout my years of study. This accomplishment would not have been possible without them. Thank you.

Abbreviations

BECM Building Energy and Comfort Management. 1

CNN Convolutional Neural Network. 3, 11–13, 21–23, 35, 37

CSI Channel State Information. 10, 12

GRU Gated Recurrent Unit. 9

HAR Human Activity Recognition. 1, 2, 8, 12

HVAC Heating, Ventilation, and Air Conditioning. 1

IoT Internet of Things. 1

IoU Intersection-over-Union. 16, 19, 28–30, 35, 36, 40, 42, 43, 45

LSTM Long Short Term Memory. 9–11, 15, 33, 45, 46

mAP mean Average Precision. 35, 37, 42, 43, 46

MEMS Microelectromechanical Systems. 3

NMS Non-maximum Suppression. 18, 30, 42, 43, 45

RCNN Region Based Convolutional Neural Network. 15–18

RGB Red Green Blue. 2, 3, 8, 9

RNN Recurrent Neural Network. 14, 15

SSD Single Shot MultiBox Detector. 19, 20, 26

TOF Time-of-Flight. 9

YOLO You Only Look Once. 17–20, 25–28, 30, 32, 35, 41–43, 48, 49

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The digital ecosystem around us is expanding at a very rapid rate. The sensor components are becoming much more affordable and as a result building owners and investors are ready to invest in smart technologies to improve building performance. Advancements being made in Internet of Things (IoT) and sensor technology are enabling intelligence to be a core part of entities like homes, commercial buildings, etc. The intelligence is achieved by connecting all the electrical, mechanical and electromechanical systems via the internet and then monitoring and taking actions whenever necessary.

All these systems, the devices and the platforms used for communication form a Building Energy and Comfort Management (BECM) system. The main objective of a BECM system is to satisfy users' requirements for comfort while keeping energy consumption as low as possible. For such a system activity awareness plays a vital role in making decisions. In a survey performed by Nguyen et al.[35], they concluded that substantial energy savings is possible by making dynamic decisions about a building's usage. They analyzed studies made on occupancy-based controls which showed up to 40% reduction in energy consumption for Heating, Ventilation, and Air Conditioning (HVAC) system. Research also shows that automated control based on the activities improves comfort while reducing energy consumption [12] and activity-based control has the potential to save energy over using individual sensors or manual controls [32].

Apart from saving energy, activity awareness greatly impacts workspace utilization. Reducing costs and increasing the productivity is a company-wide responsibility. To estimate the costs of an organization, facility managers, use a rule-of-thumb given by real-estate investment firm Jones Lang LaSalle called 3:30:300 rule [23]. This states that for every square foot, $3 is spent on utilities (energy), $30 is spent on space and $300 is spent on employees' salaries and benefits by an organization. Following this rule, an organization can greatly reduce costs by optimizing its space utilization and also improving its employees' comfort in addition to energy savings. For instance, each desk or each room can be monitored to identify vacant spaces or presence and thereby enabling down-sizing or expansion. While this is one of the fundamental ways of effectively utilizing space, monitoring human activities, helps assess usage of work spaces mainly answering the question "why is this space used for this activity?". Answering such a question provides an understanding of the functioning of the workplace and its importance. In this context, Human Activity Recognition (HAR) has numerous applications in the office sector which is one of the largest consumers of floor space and energy.

Apart from smart buildings, HAR finds its applications in different areas as well. Smart environments are being developed to assist elderly people who wish to live independently in their homes. The activity recognition system can be used to monitor activities and detect anomalies in the daily activities performed by older people. Besides, ever-growing computer vision techniques has made it possible to use activity recognition in sports, human-computer interaction and security and surveillance purposes.

1.1 Approaches used in HAR system

Because of its wide range of applications new techniques are being developed in HAR especially in the field of computer vision. Although computer vision techniques benefit from their non-intrusive approach, researchers have also developed sensors based solutions. In literature, the approaches used in HAR systems are broadly classified into camera-based and non-camera based solutions [20].

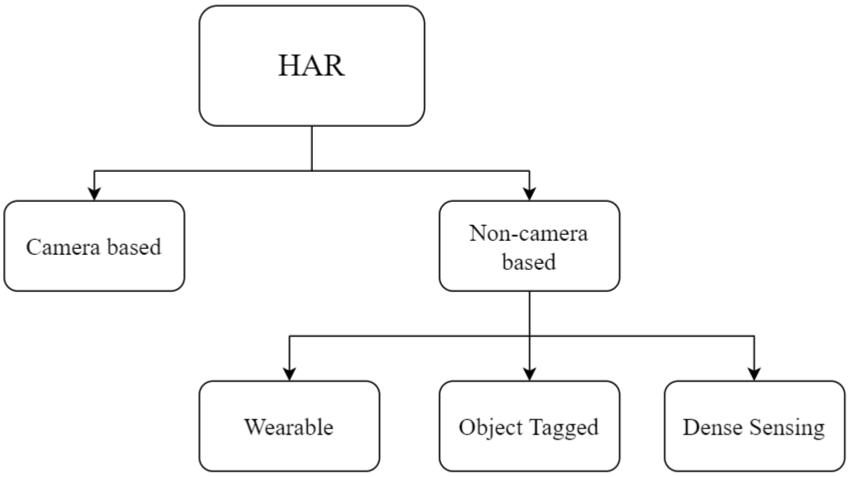


Figure 1.1: Classification of HAR approaches

As the name indicates camera-based approach utilizes the information captured from RGB cameras to recognize the activities. The details obtained from the cameras are highly informative and hence it allows the computer vision techniques to furnish good results. However, this approach is not suitable to be used in office spaces for two main reasons. One is the privacy of the users because of its highly informative data and the second one is the fact that cameras are light dependent. Poor lighting conditions easily affect the results of the recognition system. In addition, this approach also causes high computational overhead because of the size of the data.

In contrast, non-camera based approaches preserve the privacy of users and are not light-dependent. Due to low-cost various sensors are used to capture the activity data performed by humans. As shown in Figure 1.1 [20], sensor-based approach can be further classified into wearable, object tagged and dense sensing. In wearable approach, users wear sensors while performing the activity where the limitation is that a user has to wear the sensor while performing the activity. This may not be a long-term solution as the sensor may become worn out and also require batteries to be replaced/charged constantly. Additionally, sensors covering the entire human body have to be ergonomic and meet health requirements.

In the object tagged approach, sensors like RFID tags are attached to daily use objects to infer human activities [27]. In this case, deployment is difficult and users are required to use specific objects to recognize the activity.

Dense sensing approach corresponds to device-free approach where users are not required to wear any sensor or limit themselves to use specific object for activities to be recognized. The core idea is to deploy the sensor in an environment of interest and capture the data when a person is performing an activity. Later data can be analyzed by different methods to recognize the activity.

This analysis can also be real-time if required. Since the sensors are deployed in an environment sometimes there can be difficulties in deploying them. These difficulties can include blocking of the sensors by surrounding objects which makes recognition system to fail.

1.2 Proposed approach

Thermopile array sensors

Continuous development in Microelectromechanical Systems (MEMS) technology has produced remarkable progress in uncooled infrared sensors. Through these sensors thermal distribution of people and objects can be estimated by using the emitted IR radiations and mapped passively and non-intrusively. A thermopile array sensor consists of many coordinated sensing elements that allow them to measure absolute temperature as well as temperature gradient even from a distant location [39]. Due to their passive and non-intrusive characteristics thermopile arrays are used in several applications ranging from presence detection, location estimation [40] to medical applications [29]. The thermal images captured from thermopile array sensors are privacy-preserving due to their low-resolution. This makes it ideal to use these array sensors in office spaces for presence detection and activity recognition purposes. Hence in this work a thermopile array sensor is used to capture data and recognize activity.

Multiple people activity recognition

In most of the research, the focus is given on single subject activity recognition. To become practical, developed solutions should be able to handle multiple people coming into the scene. In addition, group activities such as meetings, discussions can also influence energy consumption of the buildings. Hence extending single user activity recognition to multi-user activity recognition is important. However, this task itself brings in several challenges like varying number of people, simultaneously classification multiple people actions, the similarity between different action classes, etc [21]. Research in this domain is less explored and the available studies are concentrated on RGB input data. Hence, in this research, the main goal is to develop a solution to monitor activities of each individual in office spaces using thermopile array sensors.

As explained, the privacy preserving nature of thermopile sensors makes it suitable to be used in office spaces. These office spaces normally are big with objects like desks, monitors etc. Placing sensors on wall in such spaces is not very useful due to occlusions from different objects and people. To avoid these occlusions, in this work, a ceiling based sensor is used.

Recognizing multiple people's actions is a difficult task. At a time people can be involved in multiple actions and a single image captured by sensor can have people performing different actions. Hence this can be seen as an object detection problem. To solve this, this research proposes a deep learning solution that can detect multiple people and their activities. In addition to this, the low-resolution images derived from the sensor are fuzzy in nature and it is difficult to predict if there is a single person or more people when they are close by. To deal with this, we try to analyze a sequence of images in time to tell if there is a single person or more than one person and predict each individual's activities.

1.3 Structure of the thesis

The remainder of the report is organized as follows. Chapter 2 outlines the motivation behind this research. In Chapter 3, previous works done in activity recognition are presented. Chapter 4 gives brief introduction to CNNs and object detection algorithms. Chapter 5 provides the implementation details and in Chapter 6 experiments and results are presented. Finally, Chapter 7 provides conclusions and future work.

Chapter 2

Research Motivation

As discussed earlier, activity recognition is a well-known problem in smart building/spaces domain. Improvising the present solutions or providing new solutions to this problem can greatly benefit the sectors related to health care, smart offices, assisted living etc. In most of the works related to activity recognition, the solutions are cameras-based. These solutions leverage high resolution data produced by the cameras and hence produce accurate results but the main concern is they suffer from privacy issues due to their higher resolution and details in the images. In addition, higher resolution images also require huge processing components and the recognition is limited to the field-of-view of cameras.

Generally, sensor-based approaches include wearable sensors for activity recognition. As the user wears the sensors the recognition will not be limited to a particular area. But here the limitation is that the user is required to wear these sensors which may not be preferred by some users and also if worn by the users, then their placement plays an important role in accurate prediction.

Commercial availability of thermopile array sensors has made it possible to use these sensors for applications like people counting, posture recognition. The advantages of these sensors are i) they are privacy-preserving because of their low-resolution thermal images and ii) they are non-intrusive as the sensors can be placed at a particular place to capture data. This makes it ideal for usage in systems involving people counting and activity recognition.

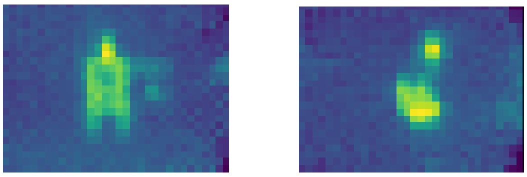
2.1 Limitations of existing solutions and challenges

Placement of sensor

In the work done in [49, 47] the sensors are placed in the horizontal plane (wall mounted sensor). This provides a better profile of the posture of a person which can be used for recognition. This can be seen in Figure 2.1a. However, in a typical office setting if a sensor is placed on the wall most of its field of view will be blocked. Hence mounting the sensor on the ceiling is ideal to get the complete view of the area.
Challenge: Placing the sensor on the ceiling poses a challenge in recognizing the posture of a person as compared to wall mounted sensor as there are less details available. In such cases, the shape information related to postures is less pronounce which can be observed in Figure 2.1.

Postures of multiple people

Previous works [30, 6] are mostly concerned with single-subject posture recognition using thermopile array sensors. Extending it to multiple people is an important step considering the real settings of office spaces.
Challenge: The main challenge include simultaneous detection of different type of postures of

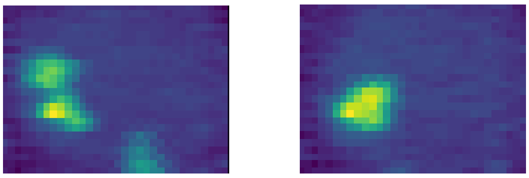


(a) Single person sitting (Sensor on table in front of person)

(b) Single person sitting (Ceiling mounted sensor)

Figure 2.1: Sitting postures from sensor placed on the table and ceiling mounted sensor

multiple people. The scenarios may include, for example, one person sitting and other person standing when there are two people under the sensor and also these combinations might change with varying number of people. Hence it is also important to identify the posture of each individual when there are two or more people. Identifying each individual's posture will also pave way for understanding group behavior which is out of scope for this research. In addition to this, when people are close to each other low resolution of the thermal image contributes to the merging of heat profiles making it, even more, difficult to recognize the postures.



(a) People Separated

(b) People Close by

Low-resolution and range

Challenge: The thermopile array sensor being used in this research produces a low-resolution image of size 24x32 pixels. Recognizing if a human is present or not in the image is not so complicated task as radiations received by the sensor can be analyzed to detect the presence of a human. Also, when there are different objects present humans can be detected by their motion pattern. However, recognizing the posture of a person becomes a challenging task with a low-resolution image as the details are not enough. This can be understood from Figure 2.2. In addition, the sensor mounted on the ceiling will be at varying heights which further makes it difficult to recognize different postures as the details keep on decreasing with increase in the ceiling height. This can be inferred from Figure 2.3.

Figures 2.4a and 2.4b show sitting and standing postures of people where the images are captured using a ceiling-mounted sensor. Looking at the images it is difficult for one to recognize if a person is sitting/standing.

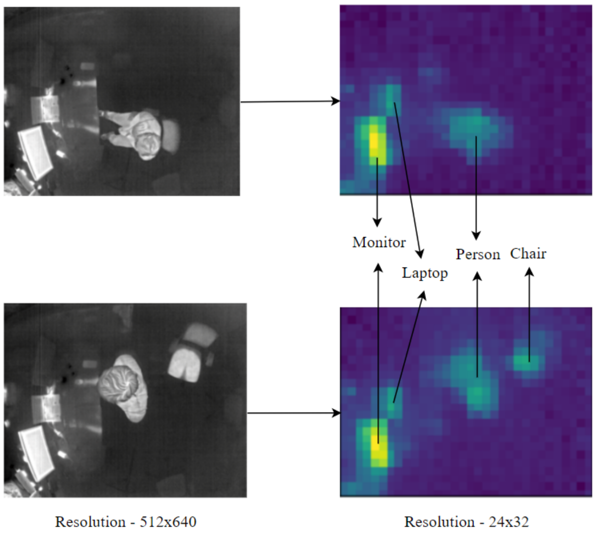


Figure 2.2: Comparison of high resolution images with low resolution thermopile images when the person is sitting and standing

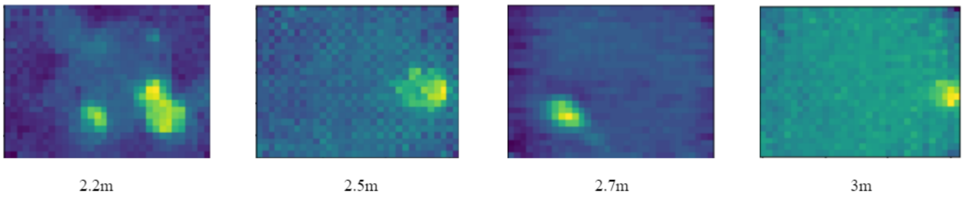
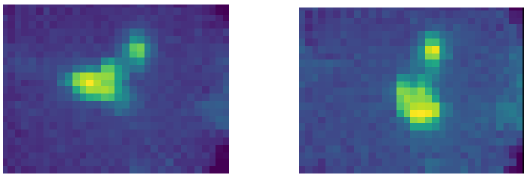


Figure 2.3: Images captured at different ceiling heights



(a) One person standing (b) One person sitting

Figure 2.4: Sitting and Standing postures

2.2 Goal of the research

In this research, the main goal is to achieve activity recognition of multiple people in office spaces using thermopile array sensors. Here the stress is on multiple people as office space is considered as a use case. In such spaces, one can easily imagine multiple people being involved in different activities coming in the field-of-view of the sensor. We consider recognizing two simple activities, sitting and standing which are most commonly occurred activities in an office space. These activities can further help in applications like understanding social interactions. In addition, low-resolution of thermopile sensor will itself not allow for very detailed activities to be recognized as opposed to high resolution images. In literature, most of the works on activity/posture recognition is based on single-subject [47, 24, 30]. In most cases, solutions based on single-subject may not work with multiple subjects. For example, a model trained with single-subject may require changes in architecture and different training data to handle multiple people. Hence, as said in Chapter 1, to be practical the provided solution has to handle multiple people scenarios. With this, the main research question can be formulated as

MRQ: *How accurately can multiple people activities be recognized using a thermopile array sensor mounted on the ceiling?*

To further specify, the first step in solving the problem would be to detect multiple people in a given image. This can be partly attributed to people counting problem. In case of RGB images, the differentiation is easier because of the available details. But in low-resolution thermal image it is hard to differentiate between static humans and other objects as the images are fuzzy in nature. Present literature provide solutions to this problem using thermopile array sensors which include extracting statistical features from thermal data [10] or using deep learning based approach [31, 17]. However, their test setup does not include other objects which emit thermal radiations like radiators, monitors etc. This leads to next research question.

RQ1: *How to detect multiple people using thermopile array sensor?*

Once the system is able to detect multiple people, the next step would be to recognize the posture/activity of each human in the image. For this purpose, single-subject posture recognition can be employed. But the catch here is to simultaneously detect multiple postures/activities in an image. This opens up a research question.

RQ2: *How to recognize a posture/activity of each individual person when there are two or more people using thermopile array sensor?*

Chapter 3

Related Work

This chapter presents the previous work done in HAR. Here the categorization is based on different kind approaches followed in HAR namely camera-based, radio frequency based and sensor-based.

3.1 Camera-based techniques

Many surveys have been undertaken to provide an outline on different kinds of approaches taken for vision-based activity recognition. Vrigkas et al. [53] presented a survey on computer vision techniques used in activity recognition. In their literature they classified the approaches into two main categories: unimodal - input data from a single modality and multimodal - input data from multiple modalities. Later these categories were divided into sub-categories based on the type of representations and interactions corresponding to the states of a person. Herath et al. [18] categorized the approaches based on feature representations and deep networks. Generally, the input data in vision-based techniques consist of RGB data and RGB-D data(RGB including depth). Due to a large amounts of publicly available datasets most of the solutions using them for bench marking can be observed.

3.1.1 Action recognition based on RGB data

Wang et al. [55] proposed a video representation method called trajectory-pooled deep-convolutional descriptor (TDD) for activity recognition. This method combined the advantages of both hand-crafted based feature extraction and deep-learning-based feature extraction. Hand-crafted features were based on SURF and RANSAC and deep-learning features were extracted from spatial and temporal networks. Finally, SVM is used for action classification. Experiments were conducted on two public large datasets, namely HMDB51 and UCF101 which provided an accuracy of 65.9% and 91.5% respectively.

Zhen et al. [60] evaluated different representation methods for action recognition. The representation methods included Bag of Words, Sparse Coding, the improved Fisher kernel, Vector of locally aggregated descriptors and the match kernels. The idea behind this evaluation was to look into the effectiveness of transferring the image domain knowledge to video domain. Experiments were conducted using each of these methods on KTH, UCF-Youtube and HMDB51 datasets and results are presented.

Paul E. Rybski and Manuela M. Veloso [44] worked on the human activity detection algorithm within the CAMEO (Camera Assisted Meeting Event Observer) system. The system consisted of four FireWire cameras to capture 360° data. The captured images were merged to form a single panoramic image. As the first step they recognize the faces of each person and then the data is fed into Dynamic Bayesian Networks to recognize the context. Activities in meeting scenarios were considered which included “Standing”, “Walking”, “Fidgeting” and “Sitting”. The model was able recognize these activities with up to 90% accuracy.

K. Sarker et al. [46] proposed an architecture to classify human actions from RGB-only streams. The pipeline included extracting pose key-points from RGB videos using OpenPose API. Then data augmentation was performed on the extracted key-points to mitigate data scarcity. Binary LSTM coupled with dense layers was used as classifier. Their method achieved up to 96% accuracy on KTH dataset.

3.1.2 Action recognition based on RGB-D data

Action recognition systems which use RGB data has shown significant progress in terms of accuracy. However, the input data may suffer form external interference like lighting, shadow, etc. In these scenarios, depth cameras can provide skeletal information irrespective of lighting conditions. Due their low-cost, Kinect depth cameras are used in many of the solutions and public datasets [59].

J. Imran and P. Kumar [22] presented a 4-channel convolutional network for human action recognition evaluated on UTD-MHAD dataset containing RGB-D data. Depth Motion Maps were created by projecting the depth images to three orthogonal planes. Each of these was taken as a channel for the convolutional network; the fourth channel consisted of temporal data of the motion of the image plane. A Pre-trained VGG-16 model was used as the base model and final layers were trained for action classification. The model was able to achieve an accuracy of 91.2%.

Yansong Tang et al. [50] presented a multi-stream deep neural networks (MDNN) method for RGB-D egocentric action recognition. Unlike regular action recognition datasets, egocentric dataset consists of actions performed by a user wearing the camera. A Primesense Carmine camera was mounted on the helmet to collect RGB and depth videos. The MDNN consisted of three inputs in the feature extraction stage namely RGB frames, optical flows and depth frames. After feature extraction, three separate deep neural networks are used to learn features for each modality. Then distinctive and shareable features are separated. These are combined with different weights and then fed into the classifier layer for action recognition. The proposed work was also evaluated on other datasets and superior results are presented in comparison with other methods.

Time-of-Flight (TOF) based solutions

The depth maps provided by the devices such as Kinect can provide enough information about the physical characteristics of a person which can breach privacy. Hence as a privacy-preserving mechanism TOF based solutions are explored.

Ikechukwu Ofodile et al. [37] proposed a concept of detecting actions using Single-Pixel Time-of-Flight Detection. The researchers used humanoids robots with pre-defined actions as test subjects. To collect the data a laser source was used to illuminate the scene and the reflected light was collected using photodetector. Up to two robots performing five actions such as walking forward, walking reverse, sitting down, standing up, and waving hand were included. Machine learning techniques such as Gated Recurrent Unit (GRU) and LSTM were adopted to perform action recognition and obtained an average recognition rate of 96.47%.

I. Bhattacharya and R. J. Radke [11] described a method for pose estimation using a sparse array of ceiling-mounted single-pixel ToF sensors. Measurements were collected for a person sitting, standing and walking. To estimate the coarse posture of a person maximum likelihood classifier was used. Experiments were conducted to see the trade-offs between frame rate and accuracy and concluded that low frame rate reduced accuracy.

3.2 Radio frequency based techniques

Traditional activity recognition systems use specific sensing elements to capture data for the recognition. The advantage in radio-based activity recognition is that they exploit the wireless communication features hence mitigating the need for physical sensing device [56].

Scholz et al. [48] investigated both device-bound and device-free activity recognition methods using IEEE 802.15.4 RSSI values. They deployed 8 IEEE 802.15.4 transceiver nodes in an office room. All nodes were synchronized by adjusting to the clock of a pre-determined master node. Walking, sitting, standing, sitting and typing, lie, lying and waving and being outside room were selected as the activities to be recognized. For device-bound method data was captured from the mobile node attached to the subject. Three types of machine learning classifiers namely k-Nearest Neighbours, naive Bayes and C4.5 decision tree were compared. k-Nearest Neighbours provided good accuracy with 89.6% and 89.2% f1-score for device-bound and device-free methods respectively.

H. Yan et al. [58] proposed a system called WiAct to recognize human activities using WiFi signals. The process of activity recognition included three steps namely data pre-processing, activity data cutting and activity classification. In the pre-processing step Channel State Information (CSI) signals were extracted from the transmitting end to the receiving end of wireless signal. These signals were passed through a butterworth filter to remove the noise. In the next step, an algorithm was employed to extract activity parts and static parts of the signal path. Later, a machine learning technique was used for classifying ten different actions. The authors compare five different machine learning techniques which included Hidden Markov Models, Sparse Autoencoder, Back Propagation neural network, LSTM and Extreme Learning Machine. The Extreme Learning Machine algorithm provided better average accuracy of 94.2% compared to other algorithms.

3.3 Sensor-based techniques

Different types of sensors are used to capture data for activity recognition. Based on sensor modality Wang et al. [54] classified them into three categories namely body-worn sensors - Smartphone, watch, or band's accelerometer, gyroscope etc., object sensors - RFID, accelerometer on cup etc. and ambient sensors - door sensor, thermopile etc.

A. Nandy et al.[33] used an ensemble classifier to classify static and dynamic activities which included sitting, sitting with weight, standing, standing with weight, lying down, lying down with weight and walking, walking with weight, climbing stairs, climbing stairs with weight respectively. The authors used accelerometer readings and heart rate readings as input data. The work flow consisted of four phases namely Data collection and pre-processing, Feature extraction, Feature selection and Learning and sensor fusion. Experiments were conducted with different classifiers and ensemble classifier provided a better accuracy of 94.61%.

Saeed et al. [45] presented a self-supervised learning method for human activity recognition. The authors worked on six publicly available datasets which included accelerometer and gyroscope readings taken from smartphones and smart watches. The self-supervised approach included two phases. In the first phase, a temporal convolutional network was trained to learn the transformations applied to the unlabelled data. Authors used eight different transformations for this purpose. Once the transformations were learned by the network, the convolutional layers were transferred to another model. In the next step, those convolutional layers were attached with a different head to make the activity recognition model. This model was trained for the head with labeled data. The results obtained was superior or comparable with fully-supervised method. Through this approach the authors leveraged the large amount of unlabelled data produced by smart devices and hence mitigating the annotation costs.

3.3.1 Thermopile-based techniques

Thermopile array sensors have been explored for activity recognition mainly because of their privacy-preserving properties. Takayuki Kawashima et al. [24] presented a single subject action recognition using a 16x16 far-infrared sensor array. The sensor was mounted on the ceiling at a height of 2.2m. Dataset collected consisted of 2520 sequences which included actions like walking, sitting down, and standing up as daily actions, and falling down as an abnormal action. From the collected images a 10x10 image was cropped around the gravity center of human using a Gaussian mixture model and then these images along with frame difference thermal image was fed into a machine learning model. The classifier architecture consisted of both CNN and LSTM. LSTM was included after the dense layers of the CNN to take into account the temporal variations of all frames of a sequence. The authors conducted four experiments with different input data. Their proposed method which included input data consisting of thermal image and frame difference image provided a higher accuracy with an average of 91.07%.

Jindrich Adolf et al. [6] used 8x8 Grid_EYE thermopile array sensor to develop a fall detection system for elderly people. The sensor was mounted on the ceiling at a height of 2.85m. A total of 4950 frames with single-subject were collected which were labeled into five classes namely no person and no object, only an object (chair or table), standing person, sitting person and laying person. The authors used Inception v3 model for classification by retraining the final layer. Experiments were conducted for different setting which included classifying each image, averaging 10 consecutive images and classifying them and classifying for 3 classes and 5 classes. Furthermore, the experiments revealed that the system was not able to classify between no person-no object and no person-single object. Apart from them, for remaining postures the system provided an accuracy of approximately 90%.

Jeroen Schipper [47] worked on a room-wide posture recognition system using thermopile array sensors. The system was made adaptable by training models based on the location of a person. Three sensor nodes were used in the system. If a person was closer to a particular sensor node then that sensor was responsible for the recognition and if the person was far enough from all the sensors then data collected from all the sensors were communicated to a central computing unit for recognition. The system was able to recognize eight different postures with an overall accuracy of 93%.

3.4 Multiple people activity recognition

Activities involving multiple people depend on the scenario they are in. There can be multiple people performing different activities in the same place which is usually termed as multi-user activity and a group of people can also be involved in performing different/same activities to achieve a common goal which is usually termed as group activity [13].

Noor Almaadeed et al.[7] proposed an approach for multi-human action recognition based on convolutional networks. To test the proposed method, the authors created their own dataset consisting of 3-5 persons performing multiple activities like boxing, walking, running, hand waving, hand clapping, jogging, carrying, standing, backpack carrying, and two persons fighting. In the pre-processing step, a block-based background initialization algorithm was employed to extract the sequence of body motions of each person in the scene. This served as an input to the neural network. The network architecture consisted of both 2-dimensional network to recognize the actions based on the Motion History Images and 3-dimensional neural network to recognize action from the generated sequences of each person. Experiments conducted on the created dataset provided an average accuracy of 95.31% with background-subtracted image sequence. To further test the region of interest extraction method, they also performed experiments on public datasets which gave comparable results.

S. Arshad et al. [9] presented a framework to recognize multiple human activities by exploiting CSI signals of IEEE 802.11n based WiFi. The captured CSI signals were passed through an Abnormal Environment Detection algorithm to remove noise and also extract parts of the signal where activities are performed. Then a CSI-To-Image Transformation Module was used to convert the acquired signals to images so that they can be fed into CNN. In the deep learning module, the authors focused on using transfer learning techniques to learn features from a pre-trained model. Inception V3 was used as base model and only final layers were trained for activity recognition. The dataset consisted of activities like walk, run and hands-move performed by 10 different subjects. Different combinations of activities were also performed to make the dataset diverse. Finally experiments were conducted using different machine learning algorithms apart from CNN used in the transfer learning setting. The presented results show that CNN (used in transfer learning setting) provided good accuracy (approximately 99%) compared to other algorithms.

Most of the multiple human activity recognition methods use RGB data as their input. Very less research has been done to recognize the activities of multiple humans using different sensors. Saipriyati Singh and Baris Aksanli [49] worked on presence detection and static activity recognition of multiple people using thermopile array sensors. For presence detection and counting part, the authors experimented with three sensors placed at three locations each on x-axis, y-axis and z-axis. Two algorithms were compared for this purpose namely window size and connected component algorithms. For activity recognition, the authors only considered static activities like standing and sitting. For this purpose, two sensors was placed along x-axis (top and bottom) where the complete posture of a person was visible. Experiments were conducted with two people and three people performing different activities. In addition, different algorithms were compared for the collected dataset. The results show that random forest classification algorithm provided better accuracy compared to others.

From the previous works done in HAR, we could see that majority of them are focused on input data from camera. The high resolution of the images provide those additional details which help to improve accuracy. Deep learning has been a go to solution for HAR because of its ability to learn subtle patterns which otherwise could not be learned by conventional approaches [54]. Posture recognition performed using thermopiles provide good accuracy with single subject. Most of the works miss multiple subjects which adds complexity to the recognition system. The work of Saipriyati Singh and Baris Aksanli consider multiple subjects but what their work miss is the ability to recognize each individual's activity.

Chapter 4

Theoretical background on CNN, LSTM and Object Detection

In this chapter, firstly, a motivation is provided for using deep learning. Next sections give brief introduction to convolutional and recurrent neural networks. Later in this chapter, another section is included to give an overview of state-of-the-art object detection algorithms based on deep learning.

4.1 Motivation behind using Deep Learning

In the last decade deep learning has shown tremendous growth because of abundant data, breakthroughs of algorithms and the advancements in hardware. Deep learning has wide range of applications which include self-driving cars, natural language processing etc. When it comes to image processing tasks deep learning has shown lot of promise and has outperformed traditional computer vision techniques [15]. In case of low-resolution thermal images this is no different. Aly Metwaly et al. [31] showed that deep learning based solution outperformed other techniques for a people counting problem. Previous chapter provided details of the related work performed to recognize human postures using thermopile array sensors. Most of the works which produce good results [16, 47] use deep learning as a technique to learn features and recognize postures. The ability to recognize patterns beyond traditional approaches has made deep learning a state-of-the-art technique. This motivated us to use deep learning for activity recognition purposes.

4.2 Convolutional and Recurrent Neural Networks

4.2.1 CNN

Images are high dimensional vectors. Using traditional Artificial Neural Networks for computation of image data is too complex as the weights on each neuron increases with the increases in the size of the image. To tackle this problem, CNNs were introduced. A CNN usually consists of three types of layers namely convolution layers, pooling layers and fully connected layers. Convolution and fully connected layers are usually followed by non-linear activation functions.

The convolution layer consists of a series of filters known as kernels. These kernels act as feature detectors and are learnable parameters in the network. Each kernel is used like a sliding window across the size of the input image. Once the convolution is done, each kernel produces a feature map. The depth of the feature map depends on the number of filters used. Using of such different types of filters helps the network to learn different features of the input image. This operation is usually followed by activation layer where a non-linear activation function helps

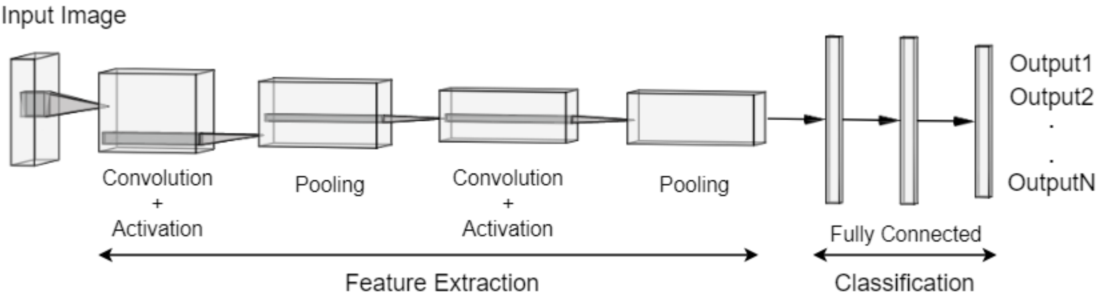


Figure 4.1: Sample CNN Architecture

network to activate kernels when a specific feature is observed at a specific location.

Pooling layers help to reduce the dimensions of the feature maps produced by the convolution layer and hence reducing the computational complexity of the model. Different types of pooling strategies are applied like max-pooling, average-pooling etc,. Finally, the fully connected layers help in the classification process. The features learnt in the convolution layers are reshaped to a single dimension vector. Each neuron in this layer is connected to every other neuron in the next layer. The final layer consists of neurons equal to the number of classes in the dataset. The output of these neurons are passed through an activation function like softmax, to extract the output or the class the input image represents.

4.2.2 RNN

Generally, sequence processing like text or video processing needs information from history for accurate predictions. A traditional feed-forward neural networks cannot perform this as they do not possess any memory units to remember information from the past. This is where RNNs are useful as they specifically contain memory units. Unlike traditional methods, the input to the RNNs is provided in a sequential way. To make decision on the current input the RNN consider current input and also the output that it has learnt from the previous output. A general structure of RNN is shown in Figure 4.2. Here X is the input at each time step, S represents the hidden state/memory of the RNN unit at that time step and o is the output for each time step. At each time step the state of each unit is calculated based on the previous state and the current input. Training RNNs is similar to training other networks. Backpropagation is used to train the network. Since the parameters are shared through all time steps, calculation of gradients is not only based on the current step but also on previous time steps. This is formally called as

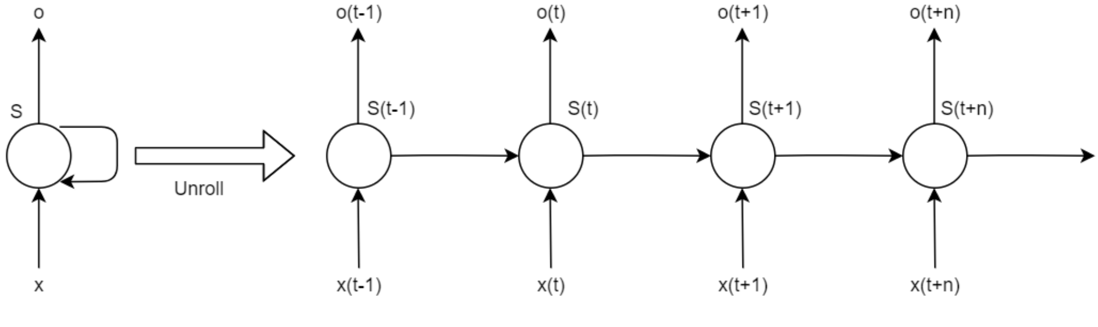


Figure 4.2: RNN and its unrolled version

Backpropagation Through Time.

4.2.3 LSTM

In a standard RNN during training, for each time step backpropagation is performed and the gradients are multiplied with weight matrix. This multiplication operation increases with increase in time steps. Hence the updated weights tend to shrink if the gradient is a low value or grow if it is a large value. Over the time steps the weights typically vanish causing vanishing gradients problem or explode causing exploding gradients problem. The vanishing gradients make the network take longer time to learn or not learn at all and exploding gradients produces oscillating weights making learning to diverge. Hence RNNs cannot remember longer time steps. To address this issue, LSTMs [19] are used which is also a type of RNN. A repeating module of LSTM units is shown in Figure 4.3 where x_t is the input, C_t is the cell/unit state and h_t is the cell/unit output.

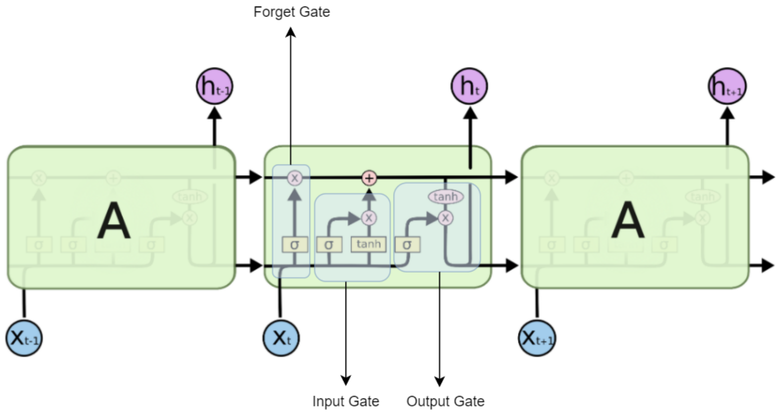


Figure 4.3: Repeating LSTM units [38]

As shown in the figure a single LSTM unit consists of forget gate, input gate and output gate. A forget gate is characterized by a sigmoid function which takes previous unit output and current input and outputs a value between 0 and 1 for each value of cell state. This gate helps the unit to remember previously learnt data or completely forget it.

Next is the input gate which adds new information to the cell state. The outputs of sigmoid and tanh layers [36] are combined to produce new candidate values. These values are then updated to the cell state. In the output gate, the cell state is passed through tanh layer to keep the values between -1 and 1. The input is passed through sigmoid layer and the output multiplied with output of tanh layer to produce the final output. This makes sure that only required parts of cell state are produced as output.

4.3 Object detection - A Deep Learning approach

Object detection is a classical problem in computer vision. Advancements being done in deep learning are providing continuous breakthroughs in this domain. The primary goal of a neural network used for object detection is to detect the object in the image and classify it. This attributes to two problems which are classification and localization. Classification problems are usually solved using classifiers like Support Vector Machines (in case of RCNN) or using the elements of the final layers of the network. The localization is characterized by bounding boxes of the object of interest. A selected network is fed with training images and corresponding bounding boxes and class labels for those images. The bounding boxes and class labels form ground truth. For each of the boxes predicted IoU between ground truth and predicted boxes helps in verification. The present literature classifies the object detection frameworks into two categories namely multi-stage detectors and single stage detectors. An overview of each of these categories is provided below.

4.3.1 Multi-stage detectors - The RCNN Family

Traditional neural networks were used for classification problems. However they could not address object detection purely based on the fact that an image can contain many objects and traditional neural networks cannot be scaled based on the number of objects. To address this issue, Ross B. Girshick et al.[15] proposed RCNN which produced a dramatically higher object detection results compared to other approaches which were not based on neural networks. The architecture consists of two stages i.e region proposals and feature extraction. The region proposal is done using selective search algorithm which proposes around 2000 regions for each image. Since the proposed regions are of different sizes, the authors use a warping region to convert to a square region. These regions are fed into next stage which is feature extraction. 4096 feature are extracted using CNN and in the final layer Support Vector Machine is employed to detect the class of each proposed region. Visual representation of the architecture can be seen in the Figure 4.4.

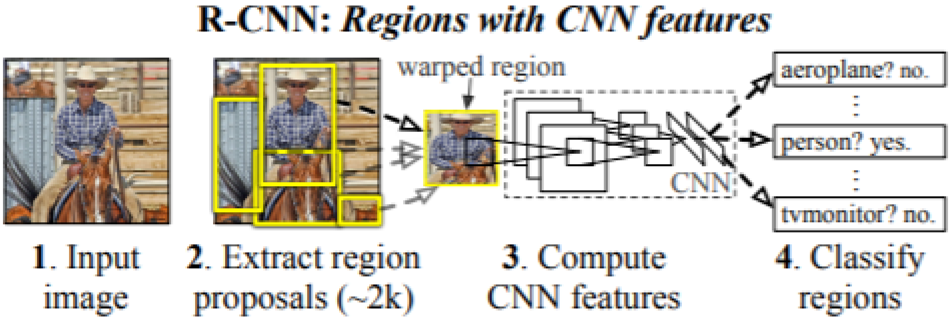


Figure 4.4: RCNN Architecture [15]

Although RCNN provided good accuracy it suffered from slowness. The algorithm took around 47sec/image (image size - 500x375) at test time for detection as it performed feed forward computations for each proposal without sharing. Due to its multi-stage design re-training for new datasets is difficult and time consuming. To mitigate these drawbacks, Ross Girshick [14], came up with Fast - RCNN. The architecture is shown in figure 4.5. Instead of passing each region proposal through a series of convolutions, a single CNN is used to generate a feature map. Then the proposed regions are mapped on to the feature map using the Region-of-Interest projection layer. In the next stage the RoI pooling layer uses max pooling to convert the feature maps lying inside the region of interest to fixed size feature map. Then a sequence of full connected layers are used to generate feature vectors. Finally, two output layers are defined to perform classification and regression. Due to its output structure, the network is trained with multitask loss and SVM is replaced with softmax layer for classification. In this method, all the layers make one network as opposed to multi-stage pipeline in RCNN which reduces computation time.

Although fast-RCNN decreased the computation time, a considerable amount of time is taken by the selective search algorithm to propose the regions. Faster-RCNN [43] introduces a region proposal network to propose regions instead of using dedicated algorithm. This region proposal network share convolutions with the detector network which greatly reduces the cost of computing proposals. The architecture is shown in Figure 4.6. Region proposals are done by sliding a n x n convolutional window over the last layer of shared convolution network. Then two fully connected sibling layers are used get confidence score and bounding box values. 9 anchor boxes are used

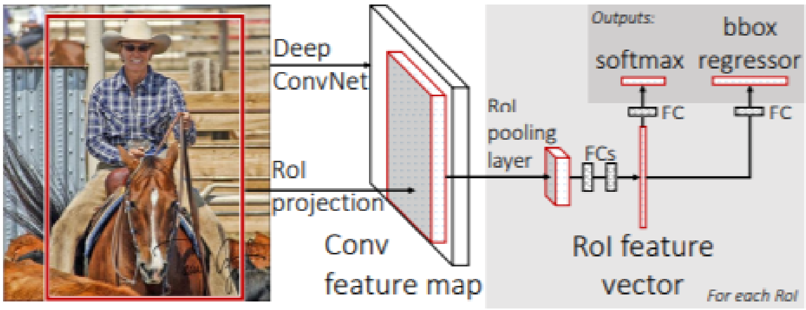


Figure 4.5: Fast-RCNN Architecture
[14]

to map the size of the object. Now the proposed regions are fed to the final layers of detector network to predict the objects. Here both region proposal network and detector network are trained independently. Since both have two outputs four loss functions are used for training namely region proposal classification loss and bounding box regression loss and detection network classification and detection network bounding box regression loss.

Algorithm	**Inference time per image**	**Speed up**	**mAP on VOC 2012 Dataset**
RCNN	∼49 sec	1x	53.3
Fast-RCNN	2.32 sec	25x	68.4
Faster-RCNN	0.2 sec	250x	75.9

Table 4.1: Comparison of multi-stage Detectors

Table 4.1 gives a comparison of all three detectors. RCNN provided a foundation for object detection using deep learning but suffered from computation time. Further versions were implemented with better accuracy and speed. However these architectures can hardly be used in a real-time application. The multi-stage design makes training time consuming and inference non-real time. In addition, running these networks on low-powered embedded devices is difficult as they require huge computational resources.

4.3.2 Single-stage detectors

The main idea behind single-stage detectors is to combine the two stages (region proposal and detection) of multi-stage networks into a single network and hence achieve higher inference speed. These networks make use of predefined bounding boxes called anchors/priors as used in faster-RCNN and the convolutional feature map from last layer to determine the confidence scores and bounding box offsets.

YOLO is one of the earliest single-stage detectors. Joseph Redmon et al. [41] implemented the algorithm as a unified detection system meaning it predicts all the bounding boxes across all the classes in an image simultaneously. The feature map from a single network is used make all the predictions. The idea behind the algorithm is shown in figure 4.7. The given input image is split into SxS grid. Each grid is responsible for predicting the conditional class probability and B bounding boxes. Along with class probability and bounding box values, each grid cell also predicts the confidence score. This score tells how confident the model is regarding the particular grid has an object and how accurate it is. The authors define it formally as $Pr(object)$ x IOU^{truth}_{pred}. Therefore for each bounding box a total of 4+1 values are predicted. First four values are the

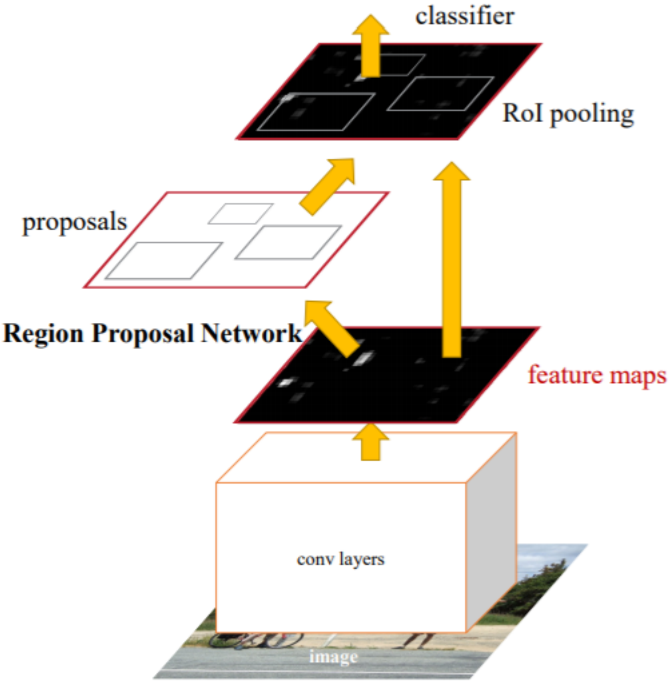


Figure 4.6: Faster-RCNN Architecture
[43]

bounding box values i.e. x, y, w, h where x, y denote the centre and w, h denote width and height. The x, y values are between 0 and 1 and w, h are predicted as fraction of width and height of whole image. The remaining one is the confidence score, $Pr(object)$ indicating whether the box has an object or not. Hence each grid predicts B x 5 parameters. To determine which class does the predicted box belong to YOLO predicts a set of class probabilities C, $Pr(Class_i|Object)$ per grid. During inference the class probabilities are multiplied with confidence score which gives class specific confidence score. Accumulating all these the final output of YOLO will be (S x S x (B x 5 + C). As an example, the architecture of YOLO trained on the PASCAL VOC dataset as shown in Figure 4.8. This dataset has 20 classes. The model divides each image into 7 x 7 grids in the final layer with 2 bounding boxes for each grid. Hence the size of the final layer is 7 x 7 x 30 i.e. 7 x 7 x (20 + 2 x 5).

During training, separate sum-squared error loss functions are used for bounding box, confidence score and class probability predictions. Once trained, thresholding and Non-maximum Suppression (NMS) is used to give out the final result. This model imposes strong spatial constraint and hence struggles at detecting group of objects. It also struggles to predict objects with unusual aspect ratios. The speed of the model is significantly higher (45 FPS) compared to Faster-RCNN (7 FPS), the mAP of YOLO (63 %)is lower compared to Faster-RCNN (75 %).

YOLOv1 suffered from significant localization errors and had a low recall value. To overcome this and to improve the accuracy Joseph Redmon et la. [42] came up with second version of YOLO. Architecturally the network was similar to VGG model used in the first version but with some changes like addition of BatchNormalization layers and removal of fully-connected layers. The network consisted of 19 convolution layer and 5 maxpooling layers. This version makes use of anchor boxes whose size is determined using K-means algorithm. The custom implementation of the network makes it faster compared to other methods and also the network is trained with different input dimensions varying by a factor of 32. Hence the same network is used to make

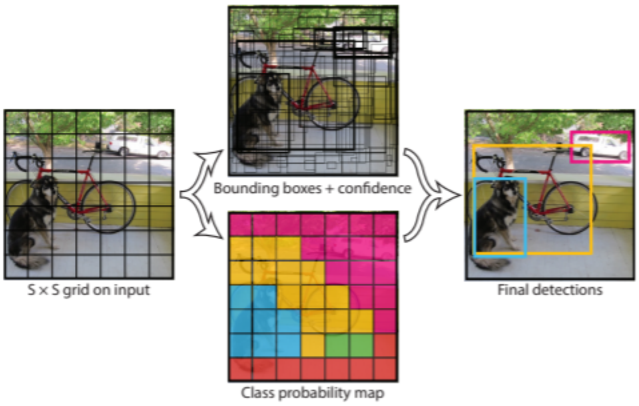


Figure 4.7: YOLO Model
[41]

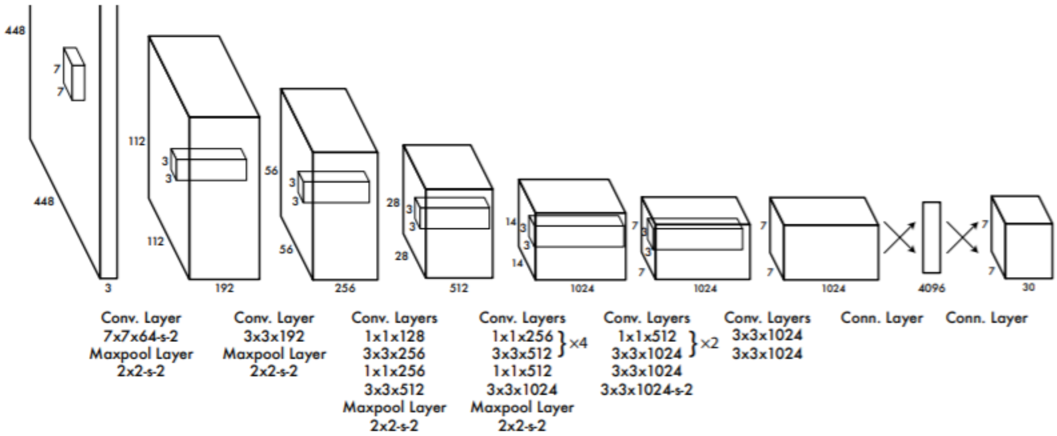


Figure 4.8: YOLO Architecture
[41]

detections at different resolutions. All details regarding implementation of this version is provided in next chapter.

Another state-of-the-art single-stage detector is the SSD proposed by Wei Liu et al [28]. This model makes use of default boxes with different aspect ratios at each location of different feature maps to predict the shape offsets in addition to class probabilities. For example, in Figure 4.9 the cat and dog in the image have different aspect ratios. The default boxes are chosen in such a way that they should have an overlap greater than 0.5 (IoU>0.5) with the ground truth. In the Figure it can be seen that the cat image is mapped with 4 x 4 feature map and dog image is mapped with 8 x 8 feature map. These feature maps are obtained after a series of convolutions and a 3 x 3 convolution is obtained on the obtained feature maps. For each location in this feature map k bounding boxes are predicted along with class scores. Hence for each feature map, SSD produces $((c + 4)$ x k x m x $n)$ outputs where m x n is the size of the feature map, k is number of bounding boxes, c is class probabilities. Number 4 represents the offsets relative to the original shape. To make predictions of objects with different aspect ratios SSD makes use of multi-scale feature maps. In comparison with YOLOv1, SSD uses only convolution layers to get the outputs. An example where VGG16 is used as a base network is shown in Figure 4.10. This feature helps SSD to make

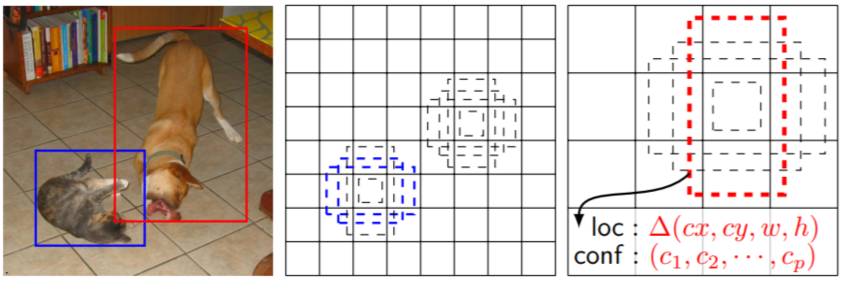

loc : $\Delta(cx, cy, w, h)$
conf : $(c_1, c_2, \cdots, c_p)$

Figure 4.9: SSD Framework
[28]

better predictions of smaller objects compared to YOLO. During training localization loss (smooth L1 loss) and confidence loss (softmax loss of class probabilities) is used.

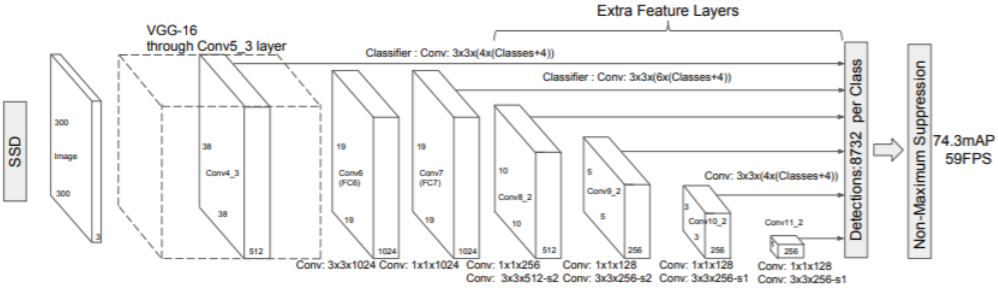


Figure 4.10: SSD Architecture
[28]

Table 4.2 shows the comparison of YOLOv1, YOLOv2 and SSD. It can be seen that SSD produces better results and also has real-time detection capability. The numbers 300, 512, 288 and 544 in the table indicate input image size. It can be seen that both SSD and YOLOv2 produce better accuracy on increasing the size of the input size.

Algorithm	**Frames per second**	**mAP on VOC 2007 Dataset**
Fast YOLOv1	155	52.7
YOLOv1 VGG-16	21	66.4
SSD300	59	74.3
SSD512	22	76.8
YOLOv2 288	91	69.0
YOLOv2 544	40	78.6

Table 4.2: Comparison of single-stage Detectors

Chapter 5

Activity Recognition System and Implementation

In this chapter, the activity recognition system is described in the first section. Later sections focus on the implementation. In the previous works [24, 6] the thermopile sensors were placed on the ceiling to recognize the activities of a single person. Although these works provide good accuracy, the authors worked only on single subject not multiple subjects. In this direction, this chapter provides a baseline implementation using a CNN model for a single person activity recognition and then we extend to multiple people activity recognition.

5.1 Activity recognition system

Activity recognition system is an amalgamation of hardware and software components. A typical activity recognition system consists of four stages shown in Figure 5.1. The first stage is sensor deployment and data collection. In this stage sensors of interest of deployed in the environment or a user is asked to wear sensors (in case of wearable sensors) to collect the data. Next step is to pre-process the collected data if required and extract features to create a feature set. Feature extraction is usually done through hand crafted algorithms like background subtraction, connected component analysis [10]. Further these extracted features are used by machine learning model, such as Support Vector Machine (SVM), Neural network (NN), K-Nearest Neighbours (KNN) for training. Finally, last step include inference of the activities.

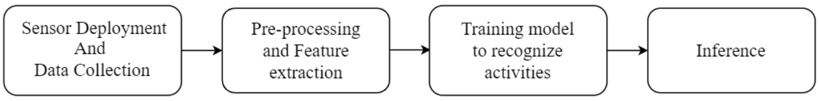


Figure 5.1: Stages of Activity recognition

5.1.1 Data collection and pre-processing

The first step in activity recognition is to deploy a sensor of interest and collect data. In this work Melexis MLX90640 thermopile sensor array is used for activity recognition purpose. The sensor is mounted on the ceiling at varying heights which include 2.2m, 2.5, 2.7m and 3m. This is done to introduce diversity in the data. The whole dataset collected contained up to 3 people either ”sitting” or ”standing”.

The thermopile consists of 24x32 sensing elements where each of them are responsible for capturing the IR radiations emitted in their field-of-view. The field of view of each element is combined

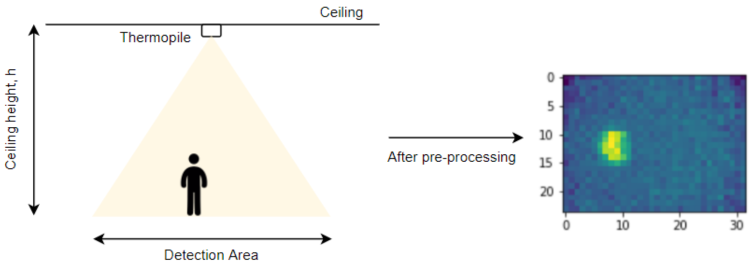


Figure 5.2: Example of sensor set up

to form the field of view of whole sensor. The sensor produces a total of 768 temperature values which are then mapped to pixel values to create a heat map. For each 768 temperature values captured, minimum and maximum temperature values are extracted. These values are then used for translating temperature values to pixel values. Since minimum and maximum values change for each image, keeping minimum and maximum values static make images to change significantly if the conditions in the room/area change. Hence to be more consistent irrespective of changes in the surroundings for each image minimum and maximum values are obtained. An example of the set up followed to collect the data from sensor is shown in Figure 5.2.

Next step is to use the obtained heat maps to recognize activities of each person in the image. For this purpose, a CNN model is built whose architectural details are explained in further sections.

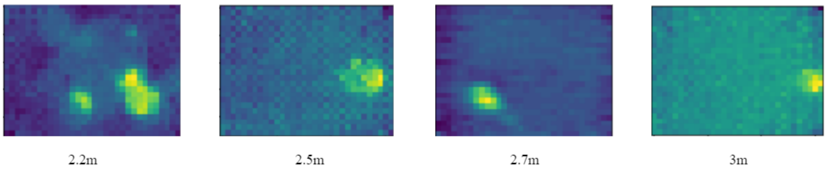


Figure 5.3: Images captured at different ceiling heights

5.2 Overview of the system

In Section 5.1, a typical activity recognition system is discussed. Similar stages are followed in this work. Figure 5.4 gives an overview of the components and the process followed in the proposed system. The deployment, data collection and pre-processing steps are explained in the Section 5.1.1. Once images are obtained, a training dataset is created by annotating the images. This step is required as we follow supervised training. Once the model is trained, predictions are made on the test set. Finally, the predictions are evaluated to record the performance.

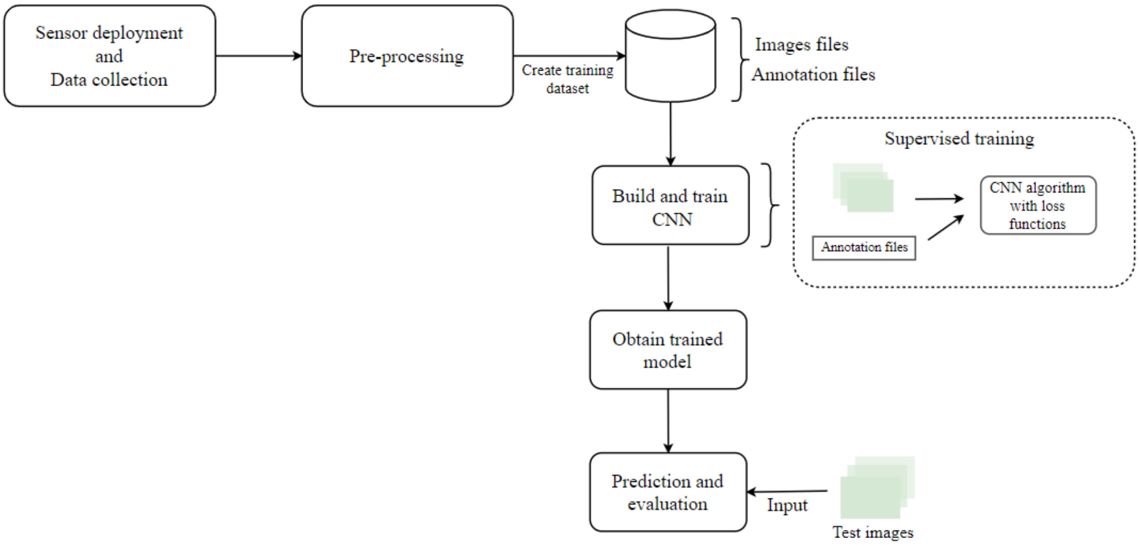


Figure 5.4: Overview of the system

5.3 Requirements of the system

Before getting into the details of CNN architecture, we identify requirements which are to be satisfied by the system. Recognizing multiple people activities can be split into two tasks. First task is to detect multiple people itself and second is to recognize activities of each person. The task of detecting multiple people naturally defines people counting problem. Hence the requirement is that

- The system must be able to identify the number of people present in a given 24x32 thermal image.

The second task of recognizing activities involves localization of each person and identifying each person's activity. The requirements for this purpose are

- The system must be able to localize each person in a given 24x32 thermal image.
- The system must be able to identify each person's activity in a given 24x32 thermal image.

In Chapter 2, a challenge was outlined for a specific case when people come close to each other. In this case, the heat profiles of people merge and it is difficult to recognize activity. Hence another requirement is

- The system must be able to identify each person's activity when two or more people come close to each other.

5.4 CNN Architecture

Information regarding type of layers used in a typical CNN is given in Section 4.2.1. The type of architecture chosen depends on the output desired from the network. In this work the focus is on multiple people activity recognition. As explained in Section 5.3, this can be partly attributed to the people counting problem as well. To take it one step at a time, firstly we define architecture for the people counting problem and then move on to single person and multiple people activity recognition. All the architectures were built using Keras API [3] with Tensorflow [5] backend in Python. As the algorithm explained in Section 5.6 was built from scratch, the loss functions are programmed using TensorFlow API. The types of architecture for each purpose are explained below.

5.4.1 Architecture for people counting

Figure 5.5 shows the architecture of the network used for people counting. The architecture is based on the work done in [47, 16] as their system also uses low-resolution thermal images. Similar to theirs, three convolutional layers are used but the number of kernels used are changed as our input image resolution is different. The architecture is designed to take an input image of size 24x32x1. For this experiment number of people involved range from 0 to 4. Hence the final layer consists of 5 neurons used for classification of the input image into categories namely "No human", "1 Person", "2 People", "3 People" and "4 People". In the figure, the values above each block represent the output shape after convolution and max pooling operations and specifications of each operation is present in between two blocks below. All the convolution layers used have 3x3 kernel size and convolution and max pool layers have a stride of 1. After each convolution and max pooling layer, ReLu activation [36] function is used. For classification in the final layer softmax activation function is used.

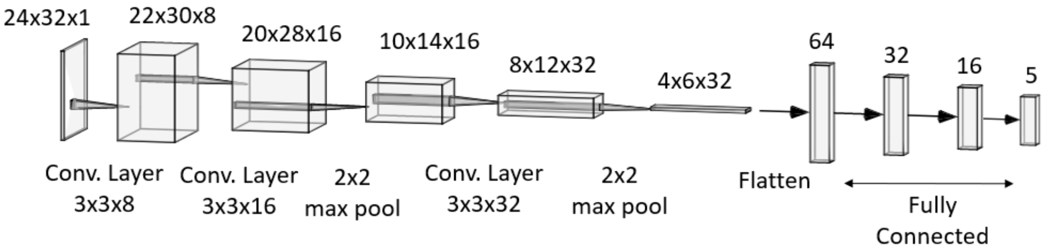


Figure 5.5: Architecture used for people counting

5.4.2 Activity Recognition - Single person

Figure 5.6 shows the architecture of the network used for single person activity recognition. The architecture is similar to the one explained in previous section as input image is same and similar layers can be used to learn features. Only changes is made in the final layer to accommodate localization task as well. This network takes an input image of size 24x32x1. The final layer consists of 7 neurons where 4 neurons are used of localization (bounding box) of the person and 3 neurons are used for classification which included "No human", "Person Sitting" and "Person Standing". In the figure, the values above each block represent the output shape after convolution and max pooling operations and specifications of each operation is present in between two blocks below. All the convolution layers used have 3x3 kernel size and convolution and max pool layers have a stride of 1. After each convolution and max pooling layer, ReLu activation function is used. For bounding box prediction sigmoid activation function is used and for classification softmax is used.

5.4.3 Activity Recognition - Upto two people

Single person activity recognition is extended to two people with a very similar architecture. The only change made is in the final layer which consists of 14 neurons. This is done because if there are two people in an image one can be sitting and other can be standing or both can be sitting or standing. Hence for classification all the combinations are considered which include "No human", "Person Sitting", "Person Standing", "1 Sitting and 1 Standing", "Two Sitting" and "Two Standing". Since there are two people involved, for localization 8 neurons are used.

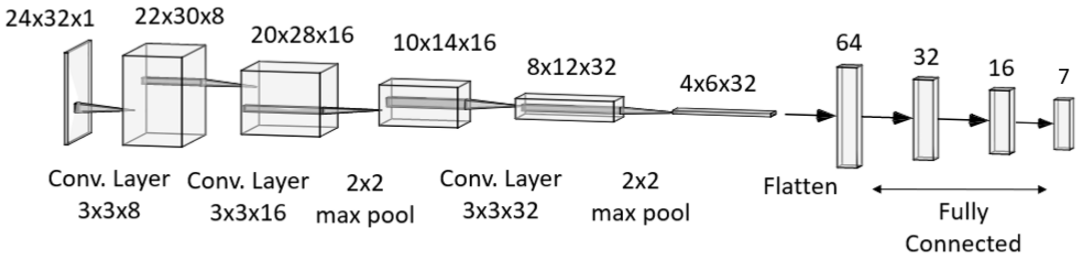


Figure 5.6: Architecture used for single person activity recognition

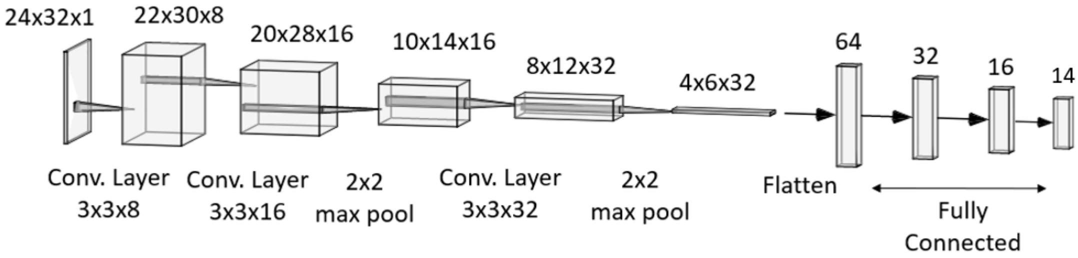


Figure 5.7: Architecture used for two people activity recognition

5.5 Need for scalable framework

Previous two sections provided architectural details for single person and two people activity recognition. Both are similar architectures with only changes in the final layer as two people detection required more number of neurons for classification and localization. Continuing with this kind of architecture does not scale with increasing number of people. For example, consider images in which number of people can vary between zero and three and people are either "Sitting" or "Standing". Now there will be 10 classes for an image to be classified into. In addition, 4 more neurons are required in the final layer to localize another person. This keeps on growing with number of people and postures. Moreover this kind of architecture can predict the bounding box values of each person in the image but it is not able to identify each individual's activity.

As we consider office space as the use case scenario, recognizing each individual's activity is important. This can provide a foundation for understanding the behavior of a group. As said in Chapter 1, these insights are important for facility managers to smartly manage office spaces. Hence as per the requirements, the chosen architecture must be able to

- Localize each person in the image
- Identify activity/posture of each person in the image

For this purpose, this problem is seen as an object detection problem where the goal is to identify each object in the image. Chapter 4 provides details regarding object detection algorithms. These algorithms mainly deal with RGB images. The size of the input images vary between 224 x 224 and 608 x 608. However, all these algorithms can be re-purposed even for lower resolution of the input images. In this work, a YOLOv2 based framework is developed to realize the defined goals due to the following reasons.

1. Unified architecture of YOLO helps to achieve real-time or near real-time performance when compared to multi-stage detectors.
2. The size and the computations required for the algorithm depends on the number of layers and filters being used. For this research, the number of layers and filters can be greatly

reduced compared to the original YOLO algorithm since low-resolution heat maps are input images. This helps to port the algorithm to embedded devices with limited memory and computational resources. Again with region based networks this would be difficult because of their multi-stage design. This also applies for SSD as it uses multi-scale feature maps. Convolutions are again performed on these feature maps which increases the number of floating point operations.

5.6 Re-purposing YOLO framework

The thermopile array sensor used in this work consists of 24x32 array of single pixel thermopile sensors. Hence its output is a low-resolution 24x32 thermal image. This acts as an input to the neural network. The network divides the input image of size 24x32 into s_1 x s_2 grids. This size of the grids is the size of the final feature map layer of the network shown in Figure 5.11. Each grid cell predicts only one object and a particular grid cell is responsible to predict a particular object if the centre of the object falls in that grid. For example, in Figure 5.8, the fifth grid in the first row is responsible for predicting that particular object which is a person. Further, in Figure 5.9, feature map for the input image is shown where the same fifth grid can be seen activated.

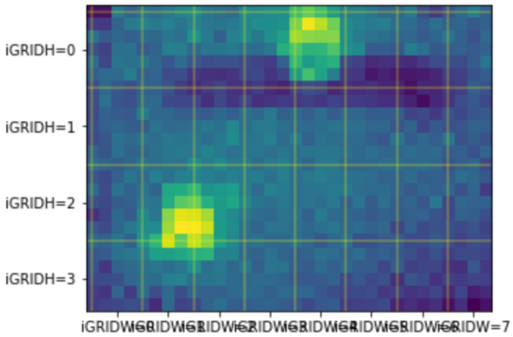


Figure 5.8: 4x8 grids on input image

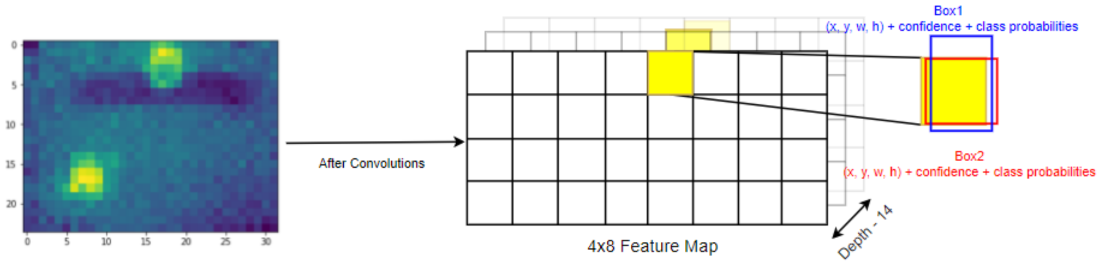

0
5
10
15
20
0 5 10 15 20 25 30
After Convolutions
4x8 Feature Map
Depth - 14
Box1
(x, y, w, h) + confidence + class probabilities
Box2
(x, y, w, h) + confidence + class probabilities

Figure 5.9: Input image to feature map

The values of s_1 and s_2 depend on the architecture of the network. Here the architecture is chosen such that the feature map size is 4x8 which is s_1xs_2. These values are chosen so that the major part of a person in the image falls into a single grid.

To predict an object, each grid cell

1. predicts B bounding boxes. In this case, each grid predict two bounding boxes.
2. predicts the confidence scores for each box which tells how confident the model is about the presence of an object in a box.

3. predicts class probabilities for each box. In this case, two classes, ”person sitting” and ”person standing” are considered.

To get into more details, the final output shape will be 4x8x2x(4+1+2). Here, 4x8 is the size of the feature map. 2 is the number of bounding boxes, *B*, for each grid. Each box has (4+1+2) values. First four are bounding box values (*x, y, w, h*) where *x, y* is the centre of the object and *w, h* is the width and height of the object, fifth is the confidence score and last two values are the class probabilities. In total, the model predicts 4x8x2 = 64 boxes with each box representing 7 values. A single grid with 2 boxes can be visualized as shown in Figure 5.10. First four values of each box are bounding box values, O is the Confidence score/Objectness score and last two are the class probabilities for each class.

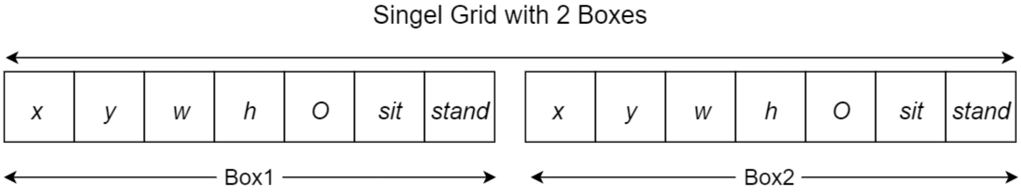


Figure 5.10: Single grid

5.6.1 Network Architecture

Figure 5.11 shows the architecture of the network used in this work. The architecture is designed such that the output produced should correspond to the shape discussed in the previous section. The number of layers used in the original YOLO [42] is 24 and their input is large RGB images. To make it more suitable for our low-resolution images the number of layers are decreased. This network takes an input image of size 24x32x1 and predicts an output with shape 4x8x2x7. In Figure 5.11, the values above each block represent the output shape after convolution and max pooling operations and specifications of each operation is present in between two blocks. All the convolution layers used have 3x3 kernel size and convolution and max pool layers have a stride of 1. After each convolution layer, batchNormalization is used to normalize the outputs of hidden layers. LeakyReLu [57] activation function is used after each convolution and in the final layer ReLu activation is used.

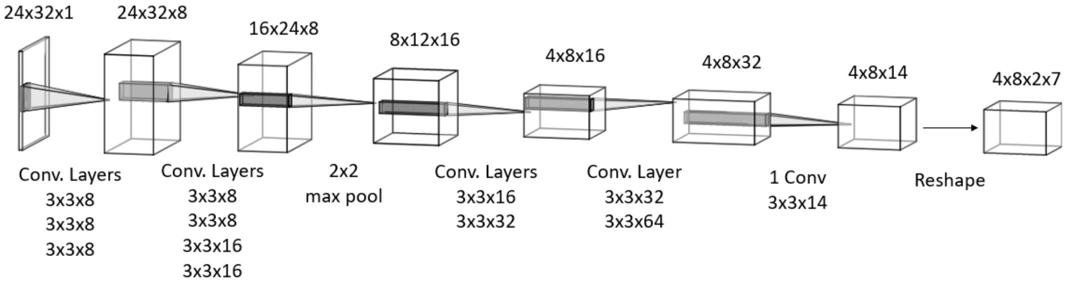


Figure 5.11: YOLO Network

5.6.2 Ground Truth and Output

To train the network in a supervised way, providing the labels is necessary. Since the output of the network is a 3-dimensional tensor, the ground truth should be provided in a similar way to calculate the loss.

Anchor boxes

One way to compute bounding boxes is to directly predict the values of the bounding boxes from the network. This approach can be error prone because during training the network can be biased towards larger bounding boxes. To increase the accuracy of bounding box predictions YOLO makes use of something called as anchor boxes. These are pre-defined set of bounding boxes with a certain height and width used to capture the scale and aspect ratio of specific object and are typically chosen based on object sizes in the training datasets. The addition of these anchor boxes makes YOLO learn better. The dimensions of these boxes are decided by running k-means clustering algorithm over the complete dataset as used in the original YOLO paper [42]. This is simple to implement and also produce efficient results. If the standard Euclidean distance metric is used, larger boxes produce more error compared to smaller boxes. Hence another distance metric is used to concentrate more on IoU independent of size of boxes. The distance metric used in the algorithm is given by

$$d(box, centroid) = 1 - IOU(box, centroid) \tag{5.1}$$

Here *box* represents the width and height of actual boxes and *centroid* is the centroids chosen by the k-means clustering algorithm.

The metric IoU used here tells how much the chosen centroid overlaps with the actual box. A visual representation of IoU can be seen in Figure 5.12. Consider Box1 as ground truth and Box2 as anchor box. To obtain IoU score, the area of overlap between the Box1 and Box2 is divided by area encompassed by both Box1 and Box2.

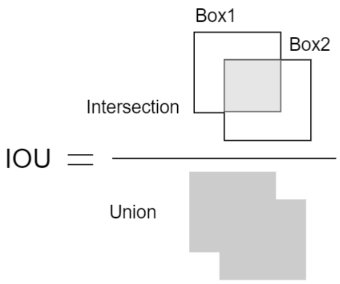
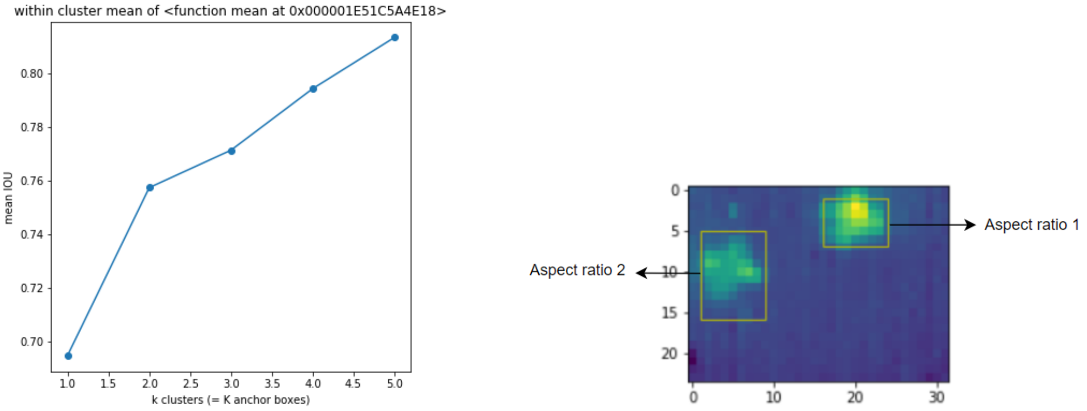


Figure 5.12: IoU

In this work, the k-means algorithm was computed with different values of k for the complete dataset. The details of the dataset is given in Table 6.7. The average IoU against the number of clusters is plotted which is shown in Figure **??**. Higher the values of k, higher the number of boxes predicted. For every increment in the value of k, number of boxes increase by 32xk. Hence the value of k = 2 is chosen to decrease the model complexity while still maintaining an overlap of more than 75% with the actual boxes. These two cluster centroids provide the width and height of the anchor boxes. Moreover, using a single anchor box, only one aspect ratio can be captured. Since we are working with different ceiling heights and different postures, the aspect ratios of boxes can be different and hence to accommodate this 2 anchor boxes are used. In the plot shown in Figure 5.13a it can be seen that 2 and 3 clusters do not differ much in average IoU and so using 2 anchor boxes reduces total number of boxes and further post processing computations. Figure 5.13b provides as example to show the difference in the aspect ratios of the bounding boxes.

Once the dimensions of anchor boxes are known, the ground truth is reshaped to match the output shape of network i.e. the ground truth shape will be 4x8x2x(4+1+2). *x, y, w, h* of the boxes are rescaled relative to the locations of the grids. This puts a constraint on the ground truth values (*x, y*) to be between 0 and 1. Since there are two anchor boxes, only one should be assigned for an object in that grid. Hence the anchor box having the highest IoU with the actual



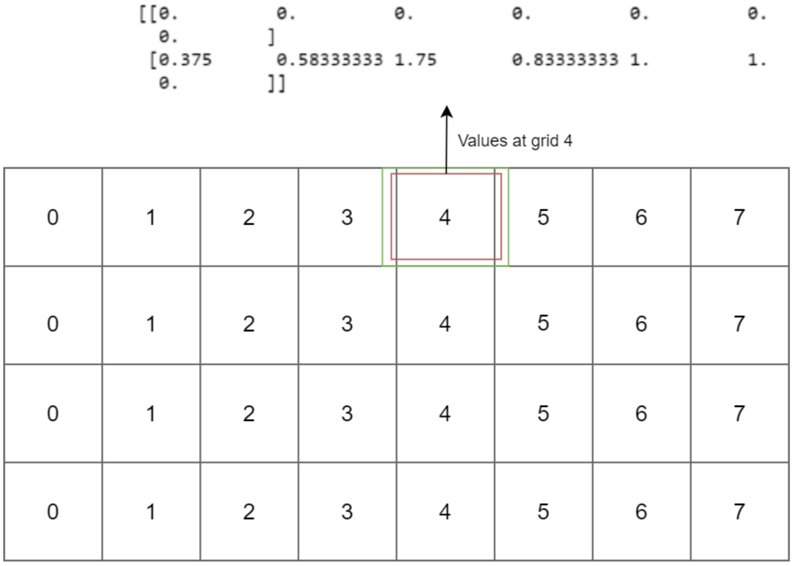
(a) Avg IOU vs No. of clusters

(b) Example image with 2 aspect ratios

Figure 5.13: Plot and an example image

box is assigned for that object. The objectness/confidence score is 1 if that box is assigned for an object or else it is 0. For class categories their respective class index is 1 and other indices are 0.

For example, consider the image shown in Figure 5.8. The person at the top belongs to the fifth grid from left. Hence at the fifth grid the values of boxes and scores are assigned as shown in Figure 5.14. The red and green boxes are the two anchor boxes. At the top, two rows represent two anchor box values and only second row is containing values as that box has higher IoU value with the actual box compared to the other. The sixth value in that row is 1 as that person in the image is sitting. All the other boxes in other grids where there is no object of interest are 0. For simplification, in the figure, assume green box has higher IoU with actual box than red box. Hence only the green box is considered and the values of the red box is 0.



```
[[0.         0.         0.         0.         0.         0.
  0.        ]
 [0.375      0.58333333 1.75       0.83333333 1.         1.
  0.        ]]
```

0	1	2	3	4	5	6	7
0	1	2	3	4	5	6	7
0	1	2	3	4	5	6	7
0	1	2	3	4	5	6	7

Figure 5.14: Representation of ground truth in the form of 4x8 grid

Extracting output

Since the ground truth is encoded as relative values with respect to grids, the output produced by YOLO is also relative to the location of the grids. To extract the exact values some computations are to be done. Let the first five values of a predicted box be t_x, t_y, t_w, t_h, t_o. A sigmoid function is applied to t_x, t_y, t_o to limit the offset range between 0 and 1. Let c_x, c_y be the grid offset calculated from the top-left corner of the image and p_w, p_h be the dimensions of the anchor boxes. Now to calculate the actual predicted values the following equations are used.

$$b_x = \sigma(t_x) + c_x \tag{5.2}$$

$$b_y = \sigma(t_y) + c_y \tag{5.3}$$

$$b_w = p_w \exp(t_w) \tag{5.4}$$

$$b_h = p_h \exp(t_h) \tag{5.5}$$

$$b_0 = \sigma(t_o) \tag{5.6}$$

Here b_x, b_y, b_w, b_h, b_o are the final predicted values.

To predict the class probabilities, the last two values are passed to a softmax function. During inference, these probabilities are multiplied with confidence score to obtain class specific confidence values. A visualization of this step is shown in Figure 5.15. The blue box is the predicted one and the dotted box represents the anchor box.

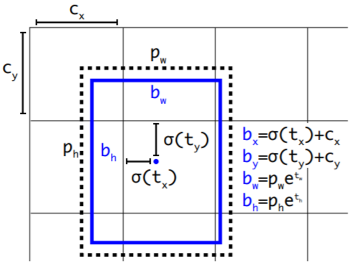


Figure 5.15: Bounding box calculations [42]

Inference

Each grid predicts two bounding boxes. In this case, a total of 64 boxes are predicted for each image. But not all the grids contain objects of interest. The steps followed for getting the final output is shown in Figure 5.16 where the red boxes are predicted boxes along with confidence scores and blue box is the ground truth for reference. As a first step, all the boxes having confidence scores less than a provided threshold are discarded. This can be seen in the Figure 5.16 where in step 1 there are boxes with low confidence values and in unwanted positions. Those unwanted boxes are removed using confidence threshold. After this, there can be more than one box for the same object with confidence score greater than the threshold (see Figure 5.16). To remove those unwanted boxes, non-maximum suppression is employed.

The pseudo code of non-maximum suppression is provided in Algorithm 1. First step is to select box containing the highest confidence score. Next for each box we check if the box and the selected box belong to the same class. If both belong to the same class, then IoU is calculated between the box and the selected box. If IoU is greater than the NMS threshold then that box is considered as unwanted and removed. This is repeated for each class to obtain the final result.

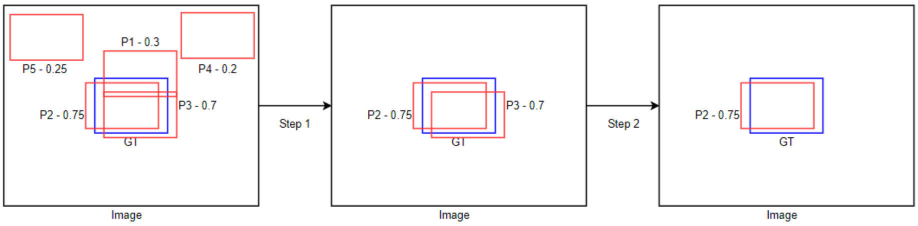


Figure 5.16: Inference steps

Algorithm 1: Non-Maximum Suppression

```
Input: B = [b_0, b_1, ..., b_n], S = [s_0, s_1, ..., s_n], N_t
       B is a list containing all the boxes
       S is a list containing all the respective confidence scores
       N_t is the non-maximum threshold
Output: Bbox = [b_0, .., b_n], Scores = [s_0, ..., s_n]
        Bbox is a list of final boxes
        Scores is a list of final confidence scores
begin
Bbox = [ ]
while $B \neq \emptyset$ do
    $j \leftarrow argmax(S)$;
    $v \leftarrow b_j$;
    $Bbox \leftarrow Bbox \cup v; B \leftarrow B - v; S \leftarrow S - s_j$;
    for $box$ in $B$ do
        if $class(box, v)$ is same then
            if $iou(box, v) \geq N_t$ then
                $B \leftarrow B - box; S \leftarrow S - box$;
            else
                do nothing
            end
        else
            do nothing
        end
    end
end
```

5.6.3 Loss functions

Since this system predicts three different categories of output, three different loss functions are used. All three losses are added to get the final loss value.

Localization loss

This loss is split into *x, y* coordinate loss and *w, h* loss for width and height. Both these losses are defined as sum-squared error loss as shown in the equation below.

$$L_{loc} = \lambda_{coord} \sum_{i=0}^{s_1 * s_2} \sum_{j=0}^{2} 1_{i,j}^{obj} [(x_i - \hat{x_i})^2 + (y_i - \hat{y_i})^2] + \lambda_{coord} \sum_{i=0}^{s_1 * s_2} \sum_{j=0}^{2} 1_{i,j}^{obj} [(\sqrt{w_i} - \sqrt{\hat{w_i}})^2 + (\sqrt{h_i} - \sqrt{\hat{h_i}})^2] \tag{5.7}$$

where,
x_i, y_i - predicted *x, y* values
$\hat{x_i}$, $\hat{y_i}$ - ground truth *x, y* values
w_i, h_i - predicted *w, h* values
$\hat{w_i}$, $\hat{h_i}$ - ground truth *w, h* values

The first summation is done from 0 to s_1 x s_2 i.e.32 as there are 4 x 8 grids and second summation is done from 0 to 2 as there are 2 boxes for each grid.

The term $1_{i,j}^{obj}$ is 1 if that box is responsible for predicting the object otherwise it is 0. While using sum squared error, small deviations become prominent in small boxes than in large boxes. To tackle this, the authors of YOLO use square root of width and height instead of their direct values. To focus more on detection, a constant $\lambda_{coord} = 5$ [41] is multiplied with loss value. By doing this, the loss increases for the boxes that don't contain any object. Since many grids do not contain any object, this trick helps in unbiased weights update.

Confidence loss

This loss is split into object loss and no object loss. Again, here sum squared error loss is used. The equation is given by,

$$L_{obj} = \sum_{i=0}^{s_1 * s_2} \sum_{j=0}^{2} 1_{i,j}^{obj} [(C_i - \hat{C_i})^2] + \lambda_{noobj} \sum_{i=0}^{s_1 * s_2} \sum_{j=0}^{2} 1_{i,j}^{noobj} [(C_i - \hat{C_i}] \tag{5.8}$$

where,
C_i - predicted confidence value
$\hat{C_i}$ - ground truth confidence value

Since all the grids generate boxes, most of the boxes in an image do not contain any object whose confidence is assigned as 0. The loss of these boxes are overpowered the ones which contain objects. Hence to give direction to the network while updating the weights, λ_{noobj} is set to 0.5 [41] so that loss decreases for the boxes which do not contain any object. $1_{i,j}^{obj}$ is 1 if that box is responsible for predicting the object otherwise it is 0. Similarly, $1_{i,j}^{noobj}$ is 1 if there is no object in that box otherwise 0.

Classification loss

The classification loss is a standard cross-entropy loss given by the below equation. Again here $1_{i,j}^{obj}$ is used to penalize the error only if an object is present in that particular box.

$$L_{class} = 1_{i,j}^{obj} \sum_{i=0}^{s_1 * s_2} \sum_{j=0}^{2} \sum_{c \in classes} \hat{p_c} log(p_c) \tag{5.9}$$

where,
p_c is class probabilities of predicted boxes
$\hat{p_c}$ is ground truth class probabilities

The total loss is computed by summing up all the losses i.e.

$$Loss = L_{loc} + L_{obj} + L_{class} \tag{5.10}$$

5.7 Temporal Model

The images obtained by thermopile are low-resolution. When people get close to each other, the heat profiles of people merge making it difficult to identify postures. This is shown in Figure 5.17. In the first image, two people are more than 0.5m apart and when they start coming close to each other the heat profiles start to merge.

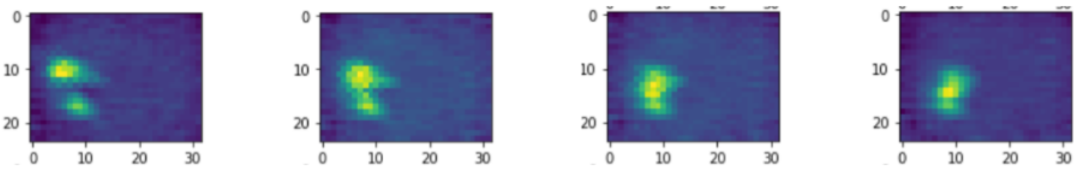


Figure 5.17: Two people coming close in time

The problem with analyzing single image using a neural network would be the inability of the network to tell if there is one person or more than one person in these kind of scenarios. This can be seen in Figure 6.12 which was resulted from analyzing single image. For this purpose, it is advantageous to analyze sequences of images.

5.7.1 Network Architecture

The main difference with the architecture described in section 5.6.1 is the addition of convolutional LSTM layers as shown in Figure 5.18. In this figure, t represents sequence length. A normal LSTM cell accepts one dimensional data as input. Since 2D spatial features are generated at each layer, convolutional LSTM is used to accept input of higher dimensions. Two convolutional LSTM layers are used with tanh as activation function. The convolution layers are made time distributed [4] which is a wrapper provided by Keras to allow apply all intermediate convolutional layers to every temporal slice of an input. No changes are made to the input encoding and output extraction. The loss functions used are also same as described in the previous section.

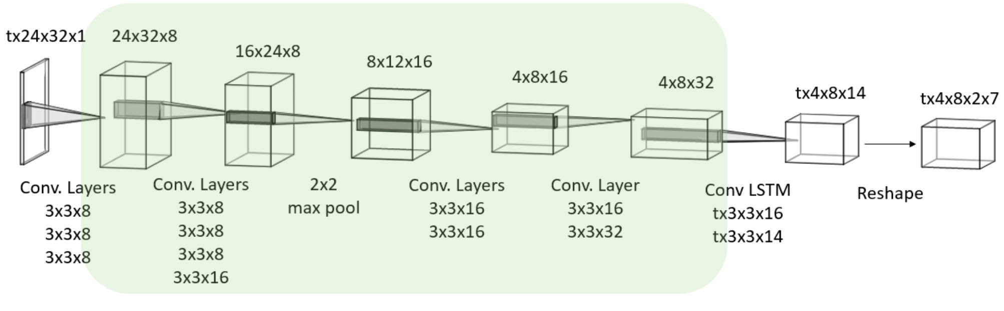


Figure 5.18: Temporal Model Architecture

Chapter 6

Experiments and Results

In this chapter, first section describes the system setup including sensor selection and software part. Next, evaluation metrics used are defined. Then experiments carried out for each of the architectures described in the previous chapter are presented along with their results and finally these results are also compared with the literature.

6.1 System setup

Sensor selection

Previous works use Panasonic GRID EYE sensor with pixel resolution of 8x8 for posture recognition [16, 47]. Using such a low resolution sensor for recognizing activities of multiple people would be difficult as the low spatial resolution will not allow multiple people to fit in the thermal image and would be hard to discern posture from a ceiling mounted sensor. High resolution sensors like FLIR Lepton [1] (pixel resolution of 120x160) are also available but they cost more and also consumes more power. We chose to use Melexis MLX90640 as it was already used used by Aly Metwaly et al. [31] for people counting task. In this research, along with people counting we aim to recognize multiple people activities as this area has not been explored using a thermopile array sensor. The specifications of Melexis MLX90640 thermopile array sensor is given in Table 6.1. The sensor bundle is equipped with STM32F4 series microcontroller for processing raw data.

Manufacturer	Resolution (pixels)	Field-of-view (HxV)	Current consumption	Price
Melexis	24x32	55°x35° and 110°x75°	<23mA	$70

Table 6.1: MLX90640 specifications

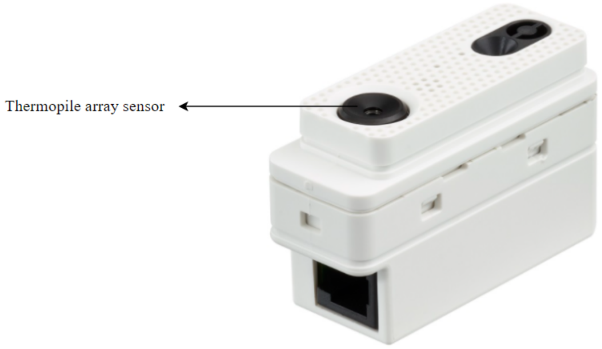


Figure 6.1: Advanced Sensor Bundle from Signify used in this work

Software setup

This proposed system performs two tasks. For people counting task the number of people is limited to 4 and for action recognition task 3 people with two activities (sitting and standing) is considered. For these tasks CNNs are used which are built using Keras API [3] with Tensorflow [5] backend in Python. To train the network AWS Sagemaker [8] is used. Once the trained model is available, inference is made using a personal computer equipped with Intel Core i7 and 8 GB RAM. While this is a general setup used for all the experiments, CNN related parameters change for experiments which is explained in their respective sections. A general description of parameters used is given in Table 6.2.

Parameter	Description
Learning rate	This parameter scales the magnitude of weight updates in order to minimize the network's loss function
Batch size	Number of training samples to work through before performing a gradient update
Optimizer	A method used to change the attributes of neural networks in order to reduce loss
Epochs	A single epoch is completed when the entire dataset is fed into the network. The number of epochs refers to the number of times a network works through the entire dataset

Table 6.2: Description of general parameters

6.2 Evaluation metrics

6.2.1 Evaluation metrics for people counting, single person and up to two people activity recognition

Accuracy

The experiments performed in Section 6.3 and 6.4 are classification tasks. Hence we use accuracy as a metric for evaluation. It is obtained by computing set of predicted labels for a sample that exactly match the corresponding set of labels in ground truth. The equation to calculate accuracy is given by

$$Accuracy(y, \hat{y}) = \frac{1}{n_{samples}} \sum_{i=0}^{n_{samples}-1} 1(y_i = \hat{y}_i) \tag{6.1}$$

where $n_{samples}$ is the total number of samples, y is the ground truth and $\hat{y}$ is predicted value

IoU

For localization evaluation in Experiment 6.4, IoU or Jaccard index [2] is used as a metric. This is explained in Section 5.6.2. A visual representation is provided in Figure 5.12. The equation for computing IoU is given by

$$\text{IoU} = \frac{\text{Area of overlap between two boxes}}{\text{Area of union between two boxes}} \tag{6.2}$$

The two boxes here represent the predicted box and the ground truth box.

6.2.2 Evaluation metric for YOLO based models

The YOLO based models perform both classification and localization tasks. Each box predicted should be classified correctly and also localized correctly. Hence to evaluate both a standard evaluation metric called Average Precision is used. This average precision is calculated for each class and mean of average precision of all the classes is taken to get mAP. To calculate the Average

Precision two commonly used metrics in classification tasks, precision and recall, are used.

Precision is the ratio of true positives to the total number of predicted positives. This ratio helps in understanding what percentage of predictions are correct. Formally it is written as,

$$Precision = \frac{TP}{TP + FP} \tag{6.3}$$

where TP - True Positives, FP - False Positives.
Recall is the ratio of true positives to the total ground truth positives. Formally it is written as,

$$Recall = \frac{TP}{TP + FN} \tag{6.4}$$

where FN - False Negatives. Here it can be seen that TP + FN gives total ground truth positives.

To determine TP and FP values, predicted class labels and IoU are used. Predicted class labels are used to check if classification task is performed correctly and IoU (5.12) determines how much the predicted box is overlapping with the ground truth. Hence if the IoU between the predicted box and ground truth box is more than a pre-defined threshold value and both belong to the same class then that box is considered as True Positive. The visualization of the TP and FP is shown in Figure 6.2. Here green boxes indicate ground truth values and red boxes indicate predicted ones. The predicted boxes are provided with their confidence values. Assuming IoU threshold for considering a predicted class to be 0.5 and all classes are predicted correctly, P1, P3 and P5 are True Positives in the figure 6.2.

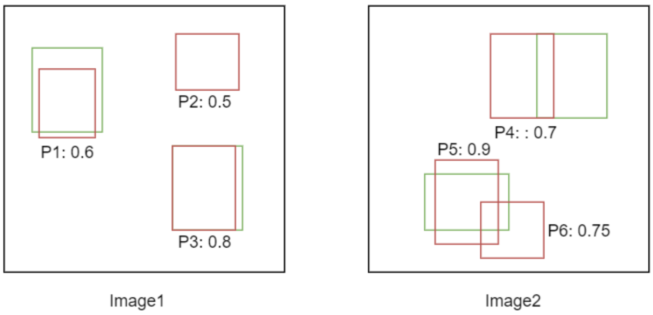


Figure 6.2: True Positives and False Positives

To calculate precision and recall values, firstly the predicted boxes are ordered by their confidence values. Calculated precision and recall values for the example shown in Figure 6.2 is as follows.

Image	Predicted Box	Confidence Value	TP or FP	Precision	Recall
Image2	P5	0.9	TP	1	0.25
Image1	P3	0.8	TP	1	0.5
Image2	P6	0.75	FP	0.66	0.5
Image2	P4	0.7	FP	0.5	0.5
Image1	P1	0.6	TP	0.6	0.75
Image1	P2	0.5	FP	0.5	0.75

Table 6.3: Precision and Recall values

After obtaining precision and recall values, precision vs recall curve is plotted to get the average precision. The precision vs recall curve for the values given in Table 6.3 is shown in Figure 6.3.

Now to obtain Average Precision, the areas are sampled for unique recall values where precision values drop. In Figure 6.3, two areas, A1 and A2, are sampled at recall values 0.5 and 0.75. where precision values drop from 1 to 0.5 and 0.6 to 0.5 respectively. Now the Area under the Curve (AUC) is calculated by numerically integrating all the sampled areas which gives the Average Precision. Mathematically, Average Precision can be written as,

$$AP = \sum_{i=0} (r_{i+1} - r_i) * p_{interp}(r_{i+1}) \tag{6.5}$$

where $p_{interp}(r_{i+1})$ is given by,

$$p_{interp}(r_{i+1}) = \max_{\hat{r}:\hat{r} \geq (r_i+1)} p(\hat{r}) \tag{6.6}$$

where $p_{interp}(r_i)$ is the precision value at recall value r_i.
Hence using the above equation, the Average Precision for the example plot shown in Figure 6.3 can be calculated as,

$$AP = A1 + A2 = 1 * 0.5 + 0.25 * 0.6 = 0.65 \tag{6.7}$$

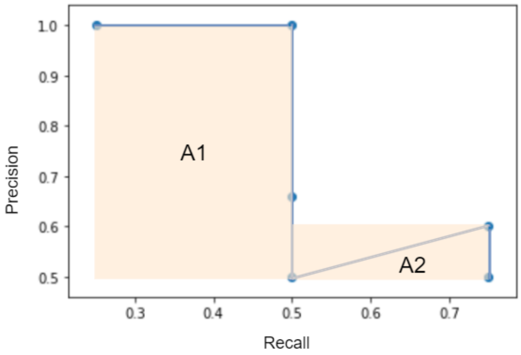


Figure 6.3: Precision vs Recall curve

This calculation of Average Precision is done for each class and mean of all the Average Precision for each class is taken to determine mAP. It can be formally written as,

$$mAP = \frac{1}{N_{classes}} \sum_{c \in classes} AP[c] \tag{6.8}$$

where $N_{classes}$ is total number of classes.

6.3 Experiment on people counting

Before recognizing people activities, first step is to make sure that we can detect multiple people using thermopile array sensor. As explained earlier, this can also be attributed to people counting as well. The details of the CNN used for this experiment is given in section 5.4.1. The collected dataset consisted of images with maximum of four people. To add diversity into the dataset and make model independent of ceiling height data was collected at different locations and different ceiling heights. Examples of collected images can be seen in Figure 5.3. To balance the dataset augmenting strategies like horizontal flipping and vertical flipping are used. The distribution of the dataset can be seen in Table 6.4.

While training the CNN, this dataset was split into training and test set with a ratio of 80:20. Further training set is divided into training and validation set. The CNN was trained with a

Total images	No Human	1 Person	2 People	3 People	4 People
12079	2676	3288	3041	1100	1974

Table 6.4: Dataset distribution for people count

learning rate of 1e-4. Learning rate was reduced during training if loss did not decrease for more than 5 epochs. Batch size was set to 32 and Adam optimizer [25] was used as a gradient descent optimization algorithm. To avoid overfitting, l2 regularization [26] is used in the dense layers. Since there are 5 classes, categorical cross entropy [34] loss function is used. Early stopping was used to stop training when the validation error did not decrease for more than 10 epochs.

Results

Once the model was trained, it was tested on 2416 images. the accuracy achieved for the test is 98.5%. This accuracy is comparable with the results shown in the work of Aly Metwaly et al. [31]. Their dataset consisted of captured in an empty room with people coming into the field-of-view of the sensor. However, they do not mention any details regarding the height of the ceiling. The dataset used in this experiment had images captured at different ceiling heights. Figure 6.4 shows the confusion matrix of the test set where the performance of the model for each category can be seen.

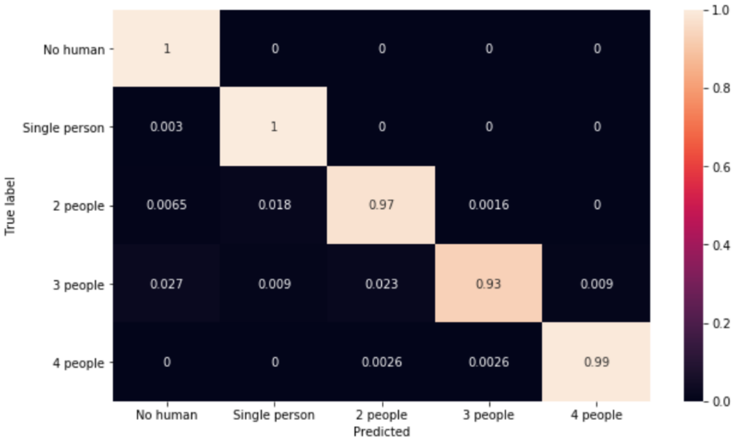


Figure 6.4: Confusion matrix of the test set

6.4 Experiment on single person and upto two people activity recognition

In the previous experiment it is successfully shown the people detection is possible using thermopile array sensor. Good accuracy is achieved in the previous experiment which showed the system is able to predict the number of people. Before moving on to activity recognition with multiple people, we start with single person activity recognition and move on to up to two people activity recognition. For this purpose two activities, sitting and standing, are considered.

The architecture used of single person and two people activity recognition is shown in Section 5.4.2 and 5.4.3 respectively. As explained in earlier chapters, it is also intended to localize each person. Hence for this, the dataset is prepared in a different way compared to the previous experiment. To annotate the locations of people in the image, an object detection annotation tool called LabelImg [52] is used. The details regarding this tool is provided in Appendix A. Once the images are annotated, they can be visualized as shown in Figure 6.5 and 6.6.

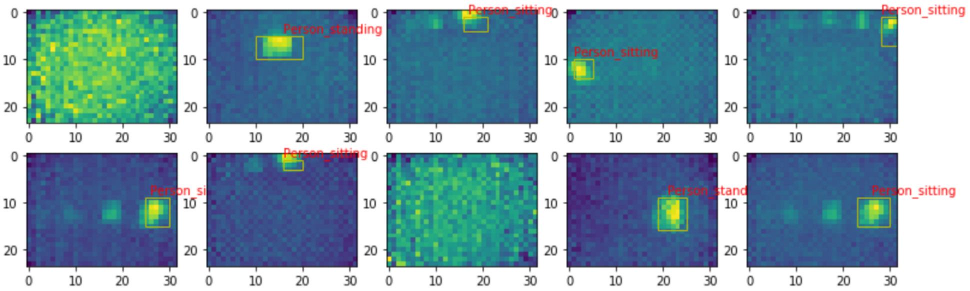


Figure 6.5: Single person images annotation

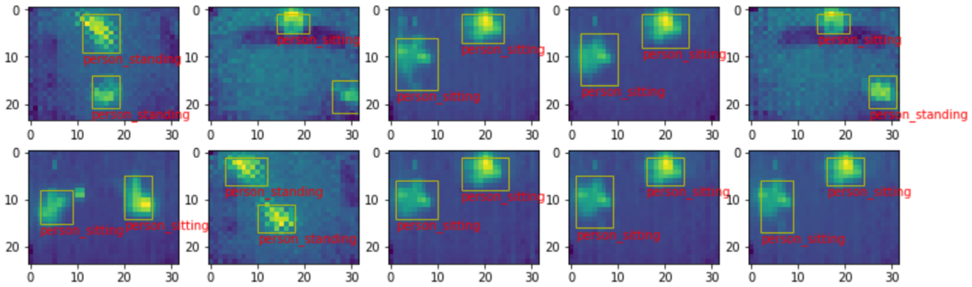


Figure 6.6: Two people images with annotation

For single person experiment, the images collected consisted of a person sitting and standing and for experiment with upto two people, the images collected consisted of one person sitting and standing and two people sitting and standing. The dataset distribution for each of the experiments is shown in Table 6.5.

Single person samples	
No human	- 407
Person sitting	- 935
Person standing	- 893
Total	- 2235

Upto two people samples	
No human	- 2676
Single person sitting	- 935
Single person standing	- 388
Two people sitting	- 1036
Two people standing	- 401
One sitting and one standing	- 577
Total	- 6013

Table 6.5: Dataset distribution

Similar to previous experiment the dataset was split between test and training sets with a ratio of 20:80. Both models were trained with a learning rate of 1e-4. Learning rate was reduced during training if loss did not decrease for more than 5 epochs. Batch size was set to 32 and Adam optimizer was used as a gradient descent optimization algorithm. For classification task, categorical cross entropy was used as loss function and for bounding box regression mean squared error was used as loss function. Early stopping was used to stop training when the validation error did not decrease for more than 10 epochs.

Results

Test set for single person experiment consisted of 447 images and for experiment with upto two people consisted of 1203 images. For classification, accuracy is used as an evaluation metric for both the experiments. For bounding box regression, IoU is calculated for each predicted box. Mean of all the IoUs is taken for the whole test set to get mean IoU.

Single person experiment	
Classification accuracy	- 99.77%
Mean IoU	- 0.66

Upto two people experiment	
Classification accuracy	- 99.91%
Mean IoU	- 0.58

Table 6.6: Results of single person and upto two people experiment

Table 6.6 shows results of both experiments. In both the experiments classification accuracy is good. Comparing them for mean IoU, it can be seen that by increasing number of people to two there is slight decrease in mean IoU. To visualize the results, some of the images with predictions and ground truth are shown in Figure 6.7 for single person experiment. The boxes plotted in red are the predicted ones and the ones in yellow are ground truth.

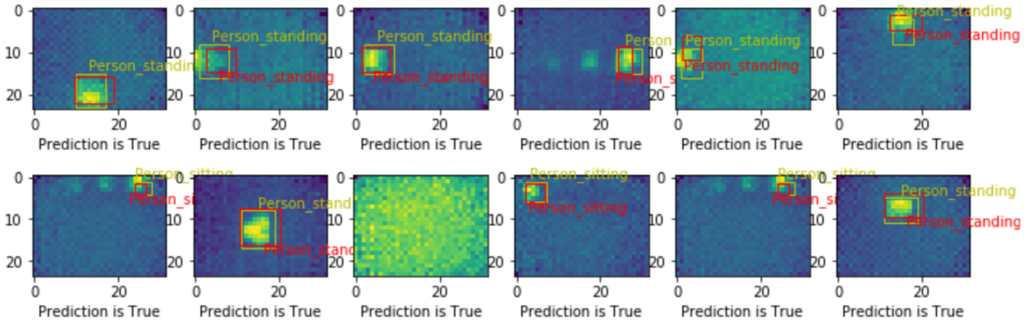


Figure 6.7: Single person results visualization

Similarly, for experiment up to two people the visualization is shown in Figure 6.8. The boxes plotted in red are the predicted ones and the ones in yellow are ground truth. The test set also consisted of images where there were laptops on a table (second image in the first row) which could also emit radiations and were captured by thermopile sensor. The trained neural network was able to classify those objects correctly as non-human. The difference between Figures 6.7 and 6.8 is that in the former one what person is doing can be recognized as the architecture classifies each image into an activity and produces bounding box values. The bounding box can be associated with the activity classified to tell what the person is doing. But in the latter one the architecture does not allow this as the classification is based on combined activity of two people. Hence as explained in Section 5.5, this kind of architecture is not able to identify each individual's activity.

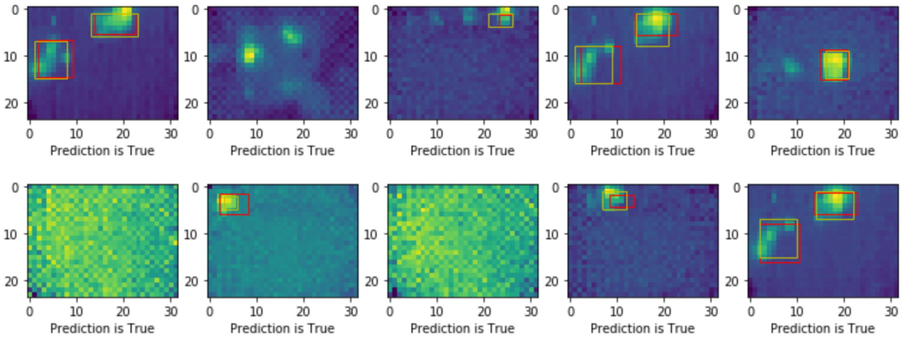


Figure 6.8: Upto two people results visualization

6.5 Experiment with YOLO based model

The network architecture described in Section 5.6.1 was trained on the collected images from the thermopile array sensor. As described in the previous experiment, the images were annotated manually with LabelImg tool. The distribution of the collected dataset is shown in the Table 6.7. Two static activities, "person_sitting" and "person_standing" are considered in this work and their distribution are shown in Table 6.7. We mainly concentrate on these two activities as our use case is office space and these two activities are most common form of activities in office areas. Recognizing such activities of each person can be further developed to recognize group behavior.

Total images	**- 5484**
Single person images	- 2104
Two people images	- 3057
Three people images	- 323

Total postures	**- 9187**
person_sitting	- 5813
person_standing	- 3374

Table 6.7: Dataset distribution

The dataset shown in the above table is collected from three different locations. Example of images for each of the location is shown in Figure 6.9. It can be seen that the images appear different with respect to background. This happens because during pre-processing the maximum and minimum temperature values are obtained for each image and the values change for every image. Hence pixel values also change. In addition, different locations have different temperature values as in the center image of Figure 6.9 where the background appears to be more cooler compared to other two. Training neural network with such diverse data always helps network to generalize better.

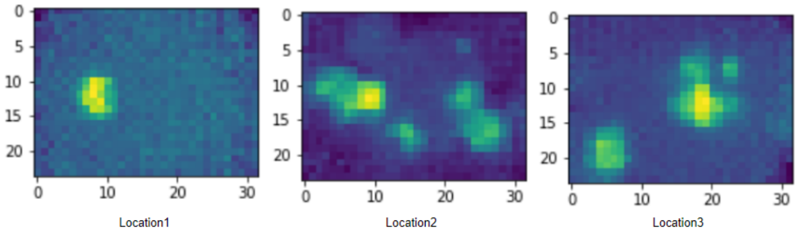


Figure 6.9: Images from different locations

For training the network, batch size was set to 128 and was trained for 80000 epochs. Since training for such long epochs would take time in CPU, AWS Sagemaker [8] was used to reduce it. AWS sagemaker provides GPU based compute instances specifically for training. A docker image with required libraries was set up and pushed to amazon compute resources. During training Adam optimizer was used as a gradient descent optimization algorithm with learning rate of 1e-5.

Results

A total 361 images were tested on the trained model. To determine if a predicted box is true positive, IoU threshold of 0.3 is set. mAP for different values of NMS threshold and confidence threshold is shown below. As per the Table 6.8, confidence threshold of 0.5 and NMS threshold of 0.4 provided good mAP. Hence these two values are selected and mAP is calculated for different IoU thresholds. This is shown in the Table 6.9.

NMS Threshold	**Confidence Threshold**	**AP-Person_sitting**	**AP-Person_standing**	**mAP**
0	0.4	0.54	0.31	0.43
0	0.5	0.62	0.65	0.63
0	0.6	0.63	0.66	0.64
0	0.7	0.63	0.64	0.64
0	0.8	0.66	0.62	0.64
0	0.9	0.66	0.61	0.64
0.1	0.4	0.66	0.44	0.55
0.1	0.5	0.74	0.65	0.69
0.1	0.6	0.73	0.66	0.69
0.1	0.7	0.74	0.64	0.69
0.1	0.8	0.74	0.62	0.68
0.1	0.9	0.74	0.61	0.67
0.2	0.4	0.68	0.49	0.58
0.2	0.5	0.74	0.65	0.69
0.2	0.6	0.73	0.66	0.69
0.2	0.7	0.74	0.64	0.69
0.2	0.8	0.74	0.62	0.68
0.2	0.9	0.74	0.61	0.67
0.3	0.4	0.67	0.47	0.57
0.3	0.5	0.74	0.66	0.70
0.3	0.6	0.73	0.66	0.69
0.3	0.7	0.74	0.64	0.69
0.3	0.8	0.74	0.62	0.68
0.3	0.9	0.74	0.61	0.67
0.4	0.4	0.74	0.43	0.57
0.4	**0.5**	**0.79**	**0.61**	**0.70**
0.4	0.6	0.78	0.61	0.69
0.4	0.7	0.78	0.60	0.69
0.4	0.8	0.78	0.59	0.69
0.4	0.9	0.77	0.59	0.68

Table 6.8: Results with different NMS and confidence values

In the previous experiment which included up to two people, the results showed that it is possible to classify an image into combined activity like 2 standing, 1 sitting and 1 standing, etc. But it was not able to recognize each individual's activity. This experiment performed using YOLO based model showed that recognition of each individual's activity is possible with mAP of 0.70. In previous experiment localization and classification were taken as separate tasks and hence the model could achieve good accuracy. In this experiment, the YOLO model combines both tasks as it is necessary to predict each individual's activity and therefore compromising on mAP.

mAP(0.2)	**mAP(0.3)**	**mAP(0.4)**	**mAP(0.5)**	**mAP(0.6)**	**mAP(0.7)**
0.71	0.70	0.63	0.49	0.27	0.07

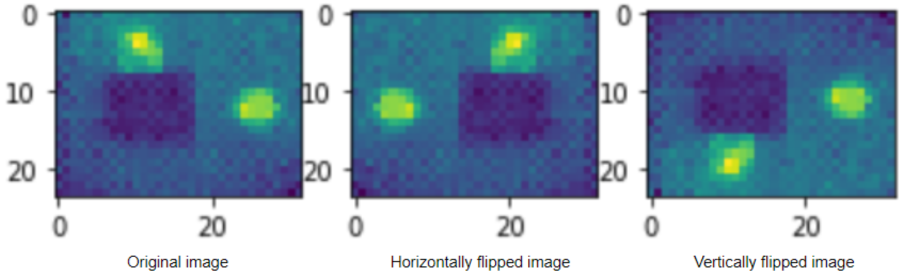
Table 6.9: mAP values for different IoU thresholds. The values in parenthesis indicate IoU threshold.

6.5.1 Augmenting dataset

The dataset shown in Table 6.7 is augmented to make data more diverse and also help in regularizing. Augmentation strategies used are i)Vertical flipping and ii)Horizontal flipping. These two strategies mainly help network to generalize on location information. Since each image was horizontally and vertically flipped the whole dataset size was tripled. The total number of images included for training was around 16452. Same network was trained with same hyperparameter settings as described in previous section.

Original images	5484
Horizontally flipped images	5484
Vertically flipped images	5484
Total	**16452**

Table 6.10: Augmented dataset distribution



Original image

Horizontally flipped image

Vertically flipped image

Figure 6.10: Original image and augmented images

Results after Augmenting dataset

Same 361 images were tested on the trained model. IoU threshold is set to 0.3 and mAP for different values of NMS threshold and confidence threshold is shown in Table 6.11.

As per the Table 6.11, confidence threshold of 0.5 and NMS threshold of 0.4 provided good mAP. Hence these two values are selected and mAP is calculated for different IoU thresholds. This is shown in the Table 6.12. In comparison with previous experiment, YOLO based model is able to identify each individual's activity which can be seen in the Figure 6.11. The Tables 6.9 and 6.12 also show that augmenting the dataset improved mAP by 5% for an IoU threshold of 0.2. In the Figure 6.11, the red boxes are the predicted boxes along with confidence scores and yellow boxes are the ground truth. Some of the unsuccessful predictions where the model was unable to identify and localize is shown in Figure 6.12. This was mainly because the people in the images are very close to each other (less than 0.5m) which makes the heat profiles to merge. Hence the model thinks that only one person is present and identifies the activity of that single person.

NMS Threshold	Confidence Threshold	AP-Person_sitting	AP-Person_standing	mAP
0	0.4	0.57	0.47	0.53
0	0.5	0.65	0.67	0.66
0	0.6	0.66	0.66	0.66
0	0.7	0.66	0.66	0.66
0	0.8	0.65	0.65	0.65
0	0.9	0.66	0.64	0.65
0.1	0.4	0.69	0.58	0.63
0.1	0.5	0.73	0.69	0.71
0.1	0.6	0.74	0.68	0.71
0.1	0.7	0.74	0.68	0.71
0.1	0.8	0.74	0.66	0.70
0.1	0.9	0.76	0.64	0.70
0.2	0.4	0.69	0.59	0.64
0.2	0.5	0.74	0.69	0.71
0.2	0.6	0.74	0.68	0.71
0.2	0.7	0.74	0.67	0.71
0.2	0.8	0.74	0.65	0.70
0.2	0.9	0.87	0.51	0.69
0.3	0.4	0.69	0.59	0.64
0.3	0.5	0.74	0.69	0.71
0.3	0.6	0.74	0.68	0.71
0.3	0.7	0.74	0.67	0.71
0.3	0.8	0.74	0.65	0.70
0.3	0.9	0.76	0.64	0.70
0.4	0.4	0.75	0.55	0.65
0.4	**0.5**	**0.78**	**0.66**	**0.72**
0.4	0.6	0.78	0.64	0.71
0.4	0.7	0.77	0.63	0.70
0.4	0.8	0.78	0.62	0.70
0.4	0.9	0.78	0.62	0.70

Table 6.11: Results with different NMS and confidence values for augmented data

mAP(0.2)	mAP(0.3)	mAP(0.4)	mAP(0.5)	mAP(0.6)	mAP(0.7)
0.76	0.72	0.64	0.49	0.29	0.07

Table 6.12: mAP values for different IoU thresholds

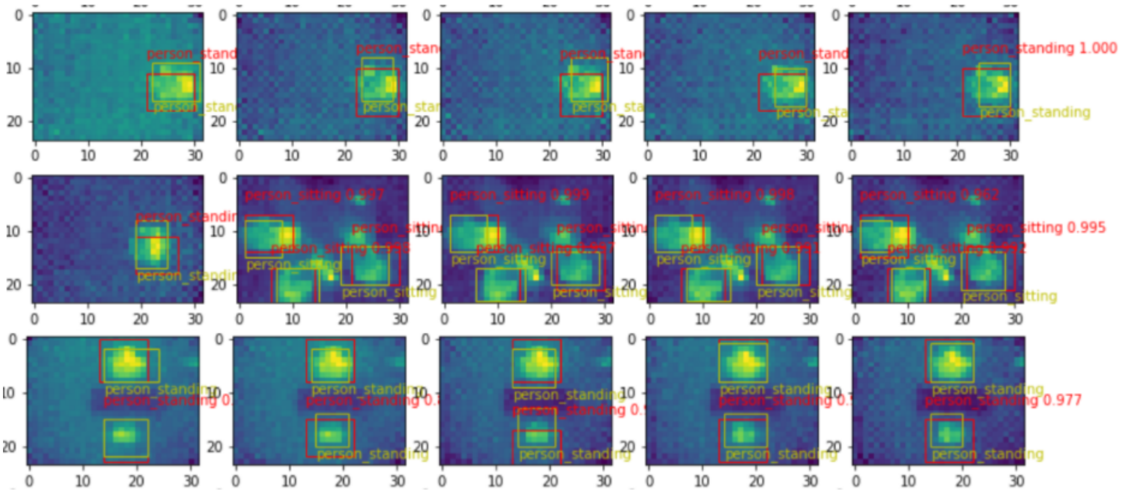


Figure 6.11: Successful predictions of YOLO based model

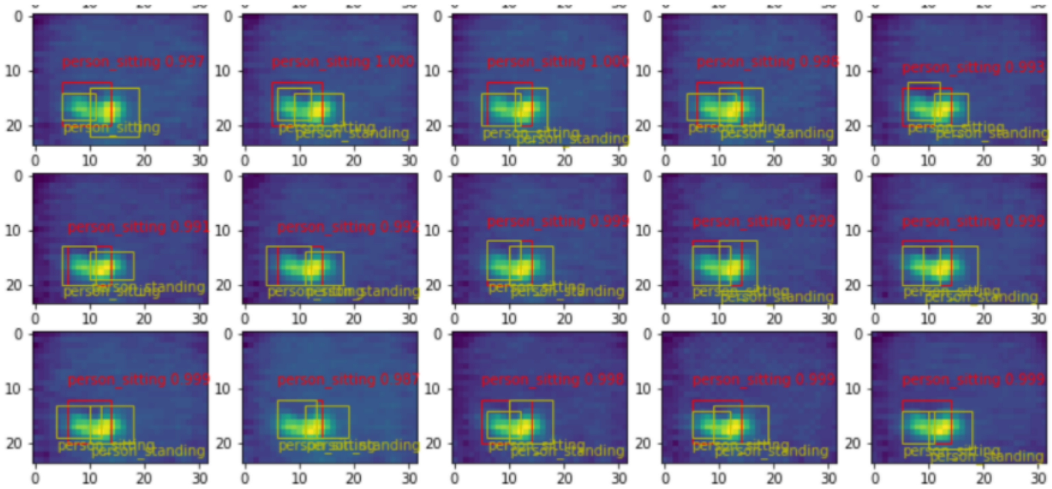


Figure 6.12: Unsuccessful predictions of YOLO based model

6.6 Experiment with temporal based YOLO model

The network architecture used for this experiment is described in section 5.7.1. A total of 1300 images were collected which consisted of people close by scenarios. This dataset consisted of two large sequences of 560 and 740 frames. The sequences consisted of one person sitting and other person coming close to the person sitting. Training epochs was set to 80000 with learning rate of 1e-5 and Adam optimizer was used as a gradient descent optimization algorithm. While training, Keras internally resets the states of the LSTM units between batches. This is not useful if information between the batches need to be shared. Hence all the frames are considered in a single batch so that states are maintained.

Results

To decide on the number of timesteps the model was trained with different values of timesteps which included 5, 10, 20, 40 and final one with length of each sequence as the value of timestep. Test set consisted of 400 images collected in sequence. For each trained model the test results are shown in Table 6.13. Based on the previous experiments, confidence threshold was set to 0.5, IoU threshold was set to 0.2 and NMS threshold was set to 0 as it was already known that there were two people in the test images and one was sitting and other was standing. Hence to avoid duplicates it was set to 0.

Timesteps	**AP-Person_sitting**	**AP-Person_standing**	**mAP**
5	0.89	0.33	0.6
10	0.92	0.50	0.71
20	0.88	0.27	0.57
40	0.92	0.35	0.62
Variable length	0.64	0.21	0.43

Table 6.13: Results for different timesteps

While model trained with 10 as timesteps provided good results, the model with variable length of timesteps did not have good results. This is due to the inability of LSTM units to remember very long sequences. To further varify, the model trained with 10 timesteps was tested with different timesteps again on the test set. It can be seen from the Table 6.14 that as the timesteps increased mAP kept on decreasing because of the difficulty in remembering long sequences by the LSTM units.

Timesteps	AP-Person_sitting	AP-Person_standing	mAP
5	0.89	0.38	0.64
10	0.92	0.50	0.71
20	0.92	0.52	0.72
40	0.92	0.28	0.60
50	0.94	0.43	0.69
100	0.94	0.34	0.64
200	0.91	0.24	0.58
400	0.93	0.20	0.57

Table 6.14: Results for different timesteps of the model trained with 10 timesteps

For comparison with non-temporal version, same test set was used to test the non-temporal model. The non-temporal model, explained in section 5.6.1, was also trained with same training set and the results of both temporal and non-temporal versions are shown in Table 6.15. As expected, temporal model provides good mAP compared to non-temporal version. This shown that time information helps in determining the number of people and their activity when they are close by and their heat profile merge.

Model	AP-Person_sitting	AP-Person_standing	mAP
Non-temporal	0.68	0.08	0.38
Temporal (10 timesteps)	0.92	0.50	0.71

Table 6.15: Comparison between Non-temporal and Temporal model

To visually compare both the models the sequence of images are plotted as shown in Figures 6.13 and 6.14. The difference can be clearly seen between both the versions.

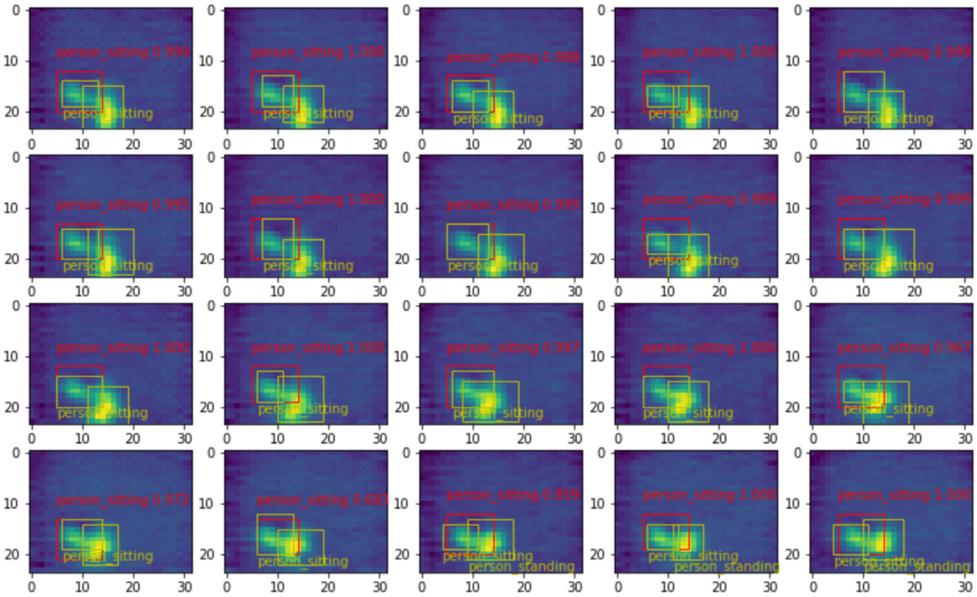


Figure 6.13: Predictions of non-temporal model

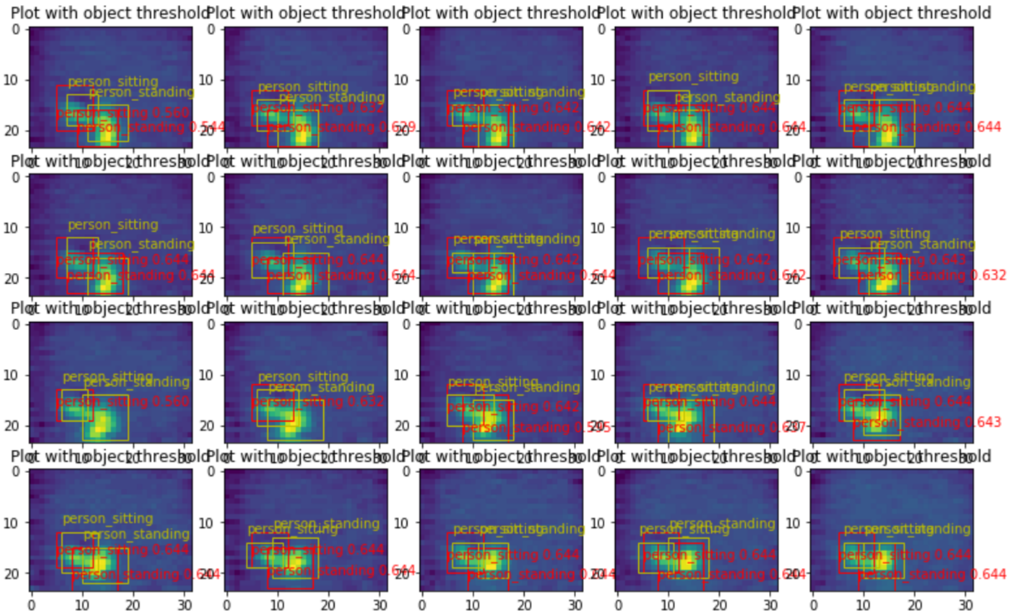


Figure 6.14: Predictions of temporal model

6.7 Comparison of obtained results with literature

6.7.1 People Counting

Table 6.16 is provided to summarize the previous works and our work related to people counting. From the results shown in the table it can be seen that our work is comparable to previous work done in people counting. The difference is that, compared to previous work, we have considered different heights of ceiling in our dataset. Additionally, the previous works do not mention about other objects being involved while capturing the data. Objects like laptops and monitors can also have same heat profile as a human which can affect the accuracy. We considered this while capturing data by making sure that there were objects like laptops around humans.

Work	Sensor	Placement	Technique	Accuracy
Tyndall et al. [51]	PIR + Thermopile	Ceiling Height of ceiling: 2.6m	K* algorithm	82.56%
Metwaly et al. [31]	Thermopile	Ceiling Height of ceiling: not available	Deep learning	98.9%
Ours	**Thermopile**	**Ceiling** **Height of ceiling: 2.2m - 3m**	**Deep learning**	**98.5%**

Table 6.16: Comparison of related work with ours for people counting

6.7.2 Action Recognition

Some of works for action/posture recognition consider placing thermopile sensors on wall [47], [16]. Such a system cannot be compared with ours as we use ceiling mounted sensor. There are also previous works where ceiling mounted sensor is used for single subject action recognition [6]. The comparison with that work is provided in the Table 6.17. However, the work of Jindrich et al. [6] used 8x8 GRID EYE thermopile array with pre-trained inception v3 model for classification where as we use 24x32 Melexis thermopile array and train the convolutional neural network from scratch.

Work	Ceiling height	Number of postures	Output classes	Precision
Jindrich et al. [6]	2.85m	2	3	0.85
Ours	**Different heights: 2.2m - 3m**	**2**	**3**	**0.99**

Table 6.17: Single person activity recognition comparison

Shota Mashiyama et al. [30] worked with sensor of resolution 32x31. They had no event, stopping, walking, sitting and falling as classification categories. Stopping event considered both sitting and standing postures. For no event and stopping categories their system achieved 100% and 94.8% accuracy giving an average of 97.4%. Our system provides slightly better accuracy. However, they consider five categories where as we consider three. But their system is not able to localize the person in the image as we do.

For multiple people action recognition only one work is available to the best of our knowledge. The work done by Saipriyati Singh et al. [49] consider three people and sitting and standing postures. Their system uses two 4x16 thermopile sensor mounted on wall as opposed to ours where a single sensor is mounted on the ceiling. Table 6.18 shows the comparison of their system with ours. The results of experiment 6.4 is compared with their work as their system is not able identify each individual's activity. For example, if there are three people sitting, their system classifies it is as 'three sitting' which is similar to what we have done in Section 6.4.

Work	Sensor placement	Number of sensors	Number of people	Localization	Accuracy
Saipriyati Singh et al. [49]	Wall	2	3	No	97.5%
Ours (as per experiment 6.4)	**Ceiling**	**1**	**up to 2**	**Yes**	**99.9%**

Table 6.18: Comparison of multiple people activity recognition

Our proposed solution using YOLO framework accomplishes two tasks i.e. localizing each person and identifying each person's activity. To the best of our knowledge, this kind of system is not available in literature which works with low-resolution thermal images to identify activities of people in a multi-human setting.

Chapter 7

Conclusions and Future Work

This thesis proposes a solution for recognizing multiple people activities using a thermopile array sensor. We started with a literature review on posture recognition using thermopile array sensors. The study revealed that there is no such system which is capable of recognizing each individual's activity when there are two or more people and when the sensor is mounted on the ceiling. We consider office space as a use case and hence for practical applications we focus on recognizing multiple people activities. In Chapter 2, we presented the limitations in existing solutions and challenges that are needed to be addressed in this work. For proposing an effective solution we use deep learning techniques because of their ability to learn deep patterns and also based on the studies performed on single subject posture recognition which show good results.

As a first step we detect multiple people which is attributed to people counting problem. We achieved high accuracy for this task. Next we built deep learning models to recognize single person activity and up to two people activity. These models gave good accuracy but was not scalable with the increase in number of people and postures. For this purpose, we re-purposed YOLO framework. Using this method we could successfully localize and identify each individual's activity in a multi-human setting. Further, we extended this to another problem where heat profiles of humans merge in a thermal image when they come close to each other and it gets difficult for recognition. For this purpose, we analyzed sequence of images in time and then predicted the activities of each human.

7.1 Contributions

The contributions of this thesis are as follows:

1. Provides evidence that multiple people can be detected using a thermopile array sensor by providing a solution to people counting problem using a deep learning model.

2. Provides evidence that activity recognition is possible using a thermopile sensor when it is mounted on ceiling

3. Provides a scalable framework based on deep learning for localizing and identifying each individual's activity when there are two or more people.

4. Show that using temporal data improves the performance in a specific case when there are people coming close to each other in time.

5. Provides analysis on timesteps that can be used for temporal model for this particular application.

7.2 Future work

1. This work provides a solution to identify each individual's activity. Another avenue for research is to extend this work to identify patterns in group behavior in office spaces.

2. Another research direction would be to recognize the direction of each person. This can be coupled with posture recognition to tell, for example, if two people are standing facing each other or not. Further, this can also be extended to recognize interactions between people.

3. During this work, one of the time consuming task was manual annotation of images. Saeed et al. [45] had worked on self-supervised learning method for activity recognition using wearable sensor datasets which were not images. Since capturing thermal images is easier and can be vastly available, self-supervised learning technique with proposed framework using images can be investigated.

4. In this work inference is performed using a personal computer with enough computational resources. As a future work, edge devices can be considered for inference which present challenges related to computation, energy and memory.

Bibliography

[1] Flir. URL: https://www.flir.com/products/lepton/. 34

[2] Jaccard index. URL: https://en.wikipedia.org/wiki/Jaccard_index. 35

[3] Keras. URL: https://keras.io/. 23, 35

[4] Keras. URL: https://keras.io/api/layers/recurrent_layers/time_distributed/. 33

[5] Tensorflow. URL: https://www.tensorflow.org/. 23, 35

[6] Jindrich Adolf, Martin Macas, Lenka Lhotska, and Jaromir Dolezal. Deep neural network based body posture recognitions and fall detection from low resolution infrared array sensor. *Proceedings - 2018 IEEE International Conference on Bioinformatics and Biomedicine, BIBM 2018*, pages 2394–2399, 2019. 4, 11, 21, 47, 48

[7] Noor Almaadeed, Omar Elharrouss, Somaya Al-Maadeed, Ahmed Bouridane, and Azeddine Beghdadi. A Novel Approach for Robust Multi Human Action Detection and Recognition based on 3-Dimentional Convolutional Neural Networks. 2019. 11

[8] Amazon. Aws sagemaker. URL: https://aws.amazon.com/sagemaker/. 35, 41

[9] S. Arshad, C. Feng, R. Yu, and Y. Liu. Leveraging transfer learning in multiple human activity recognition using wifi signal. In *2019 IEEE 20th International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM)*, 2019. 12

[10] Alex Beltran, Elect Eng, Comp Science, and Alberto E Cerpa. ThermoSense : Occupancy Thermal Based Sensing for HVAC Control. 7, 21

[11] I. Bhattacharya and R. J. Radke. Arrays of single pixel time-of-flight sensors for privacy preserving tracking and coarse pose estimation. In *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2016. 9

[12] A.I. Dounis and C. Caraiscos. Advanced control systems engineering for energy and comfort management in a building environment—a review. *Renewable and Sustainable Energy Reviews*, 13(6):1246 – 1261, 2009. 1

[13] Chairani Fauzi, Selo Sulistyo, and Widyawan. A survey of group activity recognition in smart building. *2018 International Conference on Signals and Systems, ICSigSys 2018 - Proceedings*, 2018. 11

[14] Ross Girshick. Fast r-cnn, 2015. 16, 17

[15] Ross B. Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *CoRR*, 2013. viii, 13, 16

[16] Munkhjargal Gochoo, Tan-Hsu Tan, Tsedevdorj Batjargal, Oleg Seredin, and wei-chih Yeh. Device-free non-privacy invasive indoor human posture recognition using low-resolution infrared sensor-based wireless sensor networks and dcnn. 2018. 13, 24, 34, 47

[17] Andres Gomez, Francesco Conti, and Luca Benini. Thermal image-based cnn's for ultra-low power people recognition. 2018. 7

[18] Samitha Herath, Mehrtash Harandi, and Fatih Porikli. Going deeper into action recognition: A survey. 2016. 8

[19] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 1997. 15

[20] Zawar Hussain, Michael Sheng, and Wei Emma Zhang. Different approaches for human activity recognition– a survey. 2019. 2

[21] Mostafa S. Ibrahim, Srikanth Muralidharan, Zhiwei Deng, Arash Vahdat, and Greg Mori. A Hierarchical Deep Temporal Model for Group Activity Recognition. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016-December, 2016. 3

[22] J. Imran and P. Kumar. Human action recognition using rgb-d sensor and deep convolutional neural networks. In *2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, 2016. 9

[23] JLL. A surprising way to cut real estate costs. URL: `https://www.us.jll.com/en/trends-and-insights/workplace/a-surprising-way-to-cut-real-estate-costs`. 1

[24] Takayuki Kawashima, Yasutomo Kawanishi, Ichiro Ide, Hiroshi Murase, Daisuke Deguchi, Tomoyoshi Aizawa, and Masato Kawade. Action recognition from extremely low-resolution thermal image sequence. *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance, AVSS 2017*, 2017. 7, 11, 21

[25] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. 38

[26] Anders Krogh and John A. Hertz. A simple weight decay can improve generalization. In *Proceedings of the 4th International Conference on Neural Information Processing Systems*, NIPS'91, page 950–957, San Francisco, CA, USA, 1991. Morgan Kaufmann Publishers Inc. 38

[27] Xinyu Li, Yanyi Zhang, Ivan Marsic, Aleksandra Sarcevic, and Randall S. Burd. Deep learning for rfid-based activity recognition. In *Proceedings of the 14th ACM Conference on Embedded Network Sensor Systems CD-ROM*, SenSys '16. Association for Computing Machinery, 2016. 2

[28] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. Ssd: Single shot multibox detector. *Lecture Notes in Computer Science*, 2016. 19, 20

[29] Ilde Lorato, Tom Bakkes, Sander Stuijk, Mohammed Meftah, and Gerard de Haan. Unobtrusive respiratory flow monitoring using a thermopile array: A feasibility study. *Applied Sciences*, 2019. 3

[30] Shota Mashiyama, Jihoon Hong, and Tomoaki Ohtsuki. Activity recognition using low resolution infrared array sensor. *IEEE International Conference on Communications*, 2015. 4, 7, 48

[31] Aly Metwaly, Jorge Peña Queralta, Victor Kathan Sarker, Omar Nasir, and Tomi Westerlund. Edge Computing with Embedded AI : Thermal Image Analysis for Occupancy Estimation in Intelligent Buildings Edge Computing with Embedded AI : Thermal Image Analysis for Occupancy Estimation in Intelligent Buildings. 2019. 7, 13, 34, 38, 47

[32] M. Milenkovic and O.D. Amft. An opportunistic activity-sensing approach to save energy in office buildings. In *Proceedings of the Fourth International Conference on Future Energy Systems (e-Energy '13), 22-24 May 2013, Berkeley, California.* Association for Computing Machinery, Inc, 2013. 1

[33] A. Nandy, J. Saha, C. Chowdhury, and K. P. D. Singh. Detailed human activity recognition using wearable sensor and smartphones. In *2019 International Conference on Opto-Electronics and Applied Optics (Optronix)*, 2019. 10

[34] G.E. Nasr, E. Badr, and C. Joun. Cross entropy error function in neural networks: Forecasting gasoline demand. 2002. 38

[35] Nguyen, Tuan Anh, and Marco Aiello. Energy intelligent buildings based on user activity. 2013. 1

[36] Chigozie Nwankpa, Winifred Ijomah, Anthony Gachagan, and Stephen Marshall. Activation functions: Comparison of trends in practice and research for deep learning, 2018. 15, 24

[37] Ikechukwu Ofodile, Ahmed Helmi, Albert Clapés, Egils Avots, Kerttu Maria Peensoo, Sandhra-Mirella Valdma, Andreas Valdmann, Heli Valtna-Lukner, Sergey Omelkov, Sergio Escalera, et al. Action recognition using single-pixel time-of-flight detection. *Entropy*, 2019. 9

[38] Christopher Olah. Understanding lstm networks. URL: `http://colah.github.io/posts/2015-08-Understanding-LSTMs`, 2015. viii, 15

[39] Panasonic. Thermopile arrays open up a new world of automation applications. 3

[40] S Parnin and M M Rahman. Human location estimation using thermopile array sensor. *IOP Conference Series: Materials Science and Engineering*, 260:012007, nov 2017. 3

[41] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection, 2015. 17, 19, 32

[42] Joseph Redmon and Ali Farhadi. Yolo9000: Better, faster, stronger, 2016. viii, 18, 27, 28, 30

[43] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks, 2015. 16, 18

[44] Paul E. Rybski and Manuela M. Veloso. Robust real-time human activity recognition from tracked face displacements. 8

[45] Aaqib Saeed, Tanir Ozcelebi, and Johan Lukkien. Multi-task Self-Supervised Learning for Human Activity Detection. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 2019. 10, 50

[46] K. Sarker, M. Masoud, S. Belkasim, and S. Ji. Towards robust human activity recognition from rgb video stream with limited labeled data. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 145–151, 2018. 9

[47] Jeroen Schipper. Distributed Human Posture Recognition using Thermopile Array Sensors. 2019. 4, 7, 11, 13, 24, 34, 47

[48] Markus Scholz, Till Riedel, Mario Hock, and Michael Beigl. Device-free and device-bound activity recognition using radio signal strength. In *Proceedings of the 4th augmented human international conference*, 2013. 10

[49] Saipriyati Singh and Baris Aksanli. Non-intrusive presence detection and position tracking for multiple people using low-resolution thermal sensors. *Journal of Sensor and Actuator Networks*, 2019. 4, 12, 48

[50] Y. Tang, Z. Wang, J. Lu, J. Feng, and J. Zhou. Multi-stream deep neural networks for rgb-d egocentric action recognition. *IEEE Transactions on Circuits and Systems for Video Technology*, 29(10):3001–3015, 2019. 9

[51] Ash Tyndall, R. Cardell-Oliver, and A. Keating. Occupancy estimation using a low-pixel count thermal imager. *IEEE Sensors Journal*, 16:3784–3791, 2016. 47

[52] Tzutalin. Labelimg. URL: `https://github.com/tzutalin/labelImg`. 38, 55

[53] Michalis Vrigkas, Christophoros Nikou, and Ioannis A. Kakadiaris. A review of human activity recognition methods. *Frontiers in Robotics and AI*, 2:28, 2015. 8

[54] Jindong Wang, Yiqiang Chen, Shuji Hao, Xiaohui Peng, and Lisha Hu. Deep learning for sensor-based activity recognition: A survey. *Pattern Recognition Letters*, 2019. 10, 12

[55] Limin Wang, Yu Qiao, and Xiaoou Tang. Action recognition with trajectory-pooled deep-convolutional descriptors. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015. 8

[56] Shuangquan Wang and Gang Zhou. A review on radio based activity recognition. *Digital Communications and Networks*. 10

[57] Bing Xu, Naiyan Wang, Tianqi Chen, and Mu Li. Empirical evaluation of rectified activations in convolutional network, 2015. 27

[58] H. Yan, Y. Zhang, Y. Wang, and K. Xu. Wiact: A passive wifi-based human activity recognition system. *IEEE Sensors Journal*, pages 296–305, 2020. 10

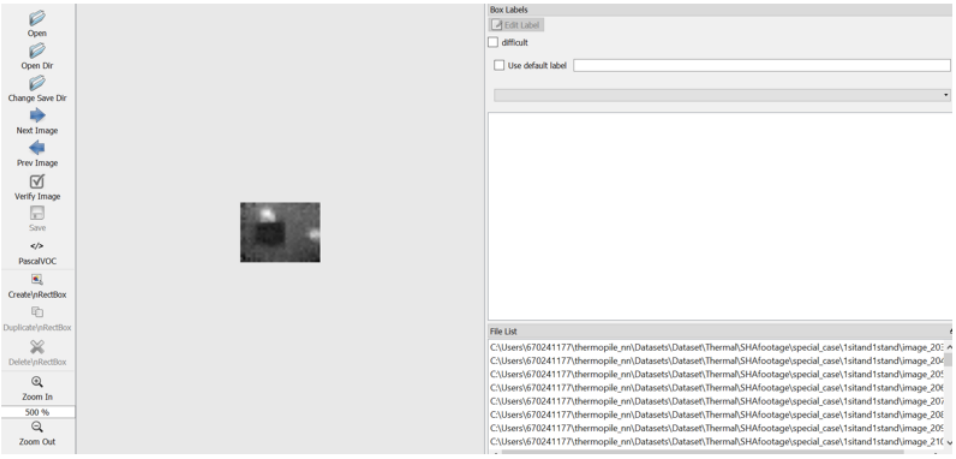
[59] Shugang Zhang, Zhiqiang Wei, Jie Nie, Lei Huang, Shuang Wang, and Zhen Li. A review on human activity recognition using vision-based method. *Journal of Healthcare Engineering*, 2017. 9

[60] Xiantong Zhen and Ling Shao. Action recognition via spatio-temporal local features: A comprehensive study. *Image and Vision Computing*, 50:1–13, 2016. 8

Appendix A

Annotation of images

To annotate the thermal images an open-source tool called labelImg [52] is used. This tool provides features to draw a bounding box around the object of interest and label them. Later, the required details can be extracted as PASCAL VOC (.xml) format or YOLO format (.yaml). A screen shot of the tool is shown in Figure A.1.



□ difficult

□ Use default label

```
C:\Users\670241177\thermopile_nn\Datasets\Dataset\Thermal\SHAfootage\special_case\1sitand1stand\image_203
C:\Users\670241177\thermopile_nn\Datasets\Dataset\Thermal\SHAfootage\special_case\1sitand1stand\image_204
C:\Users\670241177\thermopile_nn\Datasets\Dataset\Thermal\SHAfootage\special_case\1sitand1stand\image_205
C:\Users\670241177\thermopile_nn\Datasets\Dataset\Thermal\SHAfootage\special_case\1sitand1stand\image_206
C:\Users\670241177\thermopile_nn\Datasets\Dataset\Thermal\SHAfootage\special_case\1sitand1stand\image_207
C:\Users\670241177\thermopile_nn\Datasets\Dataset\Thermal\SHAfootage\special_case\1sitand1stand\image_208
C:\Users\670241177\thermopile_nn\Datasets\Dataset\Thermal\SHAfootage\special_case\1sitand1stand\image_209
C:\Users\670241177\thermopile_nn\Datasets\Dataset\Thermal\SHAfootage\special_case\1sitand1stand\image_210
```

Figure A.1: Screenshot of LabelImg tool

Once the images are imported into the tool, the boxes are drawn manually around the objects and the contents are saved in a .xml file. Consider the image in Figure A.2. The xml code obtained after annotating this image is shown in Figure A.3. Later this code is parsed to required format which is used for training.

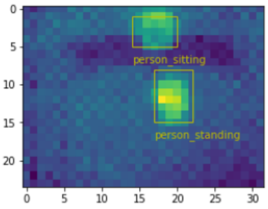
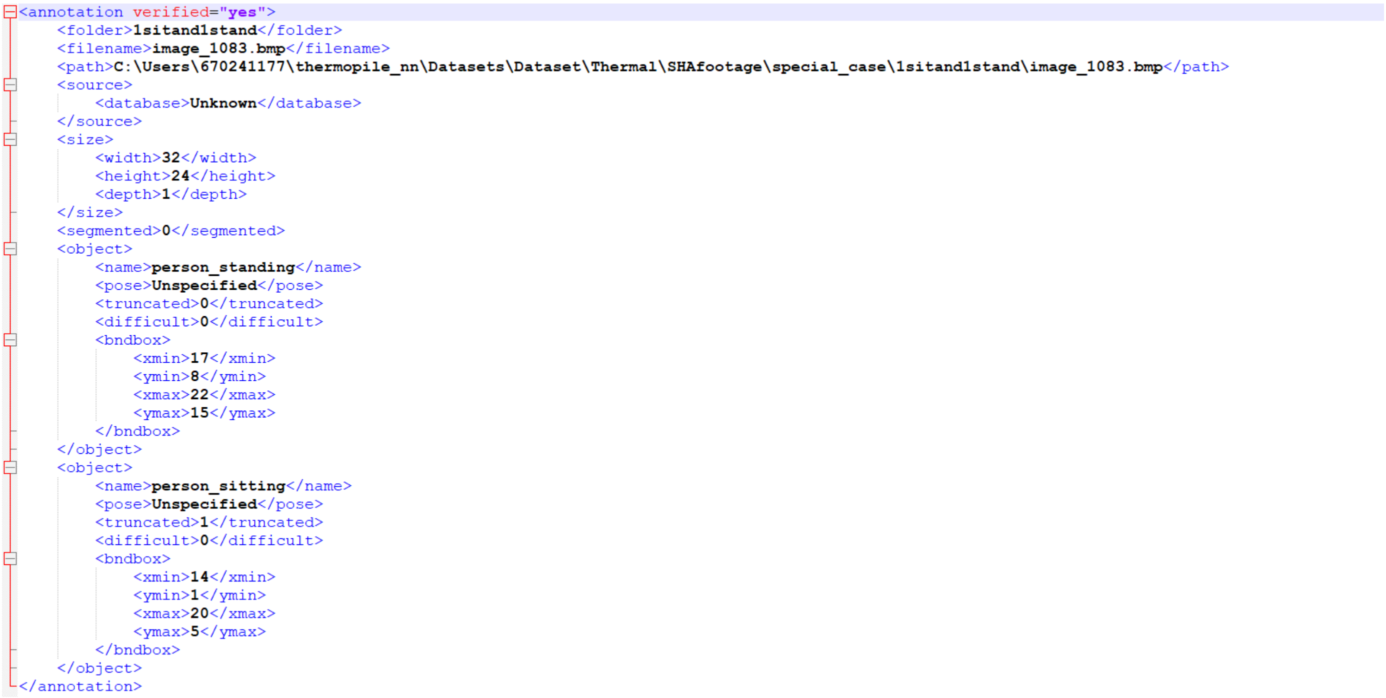


Figure A.2: Example image with annotation



```
<annotation verified="yes">
    <folder>1sitand1stand</folder>
    <filename>image_1083.bmp</filename>
    <path>C:\Users\670241177\thermopile_nn\Datasets\Dataset\Thermal\SHAfootage\special_case\1sitand1stand\image_1083.bmp</path>
    <source>
        <database>Unknown</database>
    </source>
    <size>
        <width>32</width>
        <height>24</height>
        <depth>1</depth>
    </size>
    <segmented>0</segmented>
    <object>
        <name>person_standing</name>
        <pose>Unspecified</pose>
        <truncated>0</truncated>
        <difficult>0</difficult>
        <bndbox>
            <xmin>17</xmin>
            <ymin>8</ymin>
            <xmax>22</xmax>
            <ymax>15</ymax>
        </bndbox>
    </object>
    <object>
        <name>person_sitting</name>
        <pose>Unspecified</pose>
        <truncated>1</truncated>
        <difficult>0</difficult>
        <bndbox>
            <xmin>14</xmin>
            <ymin>1</ymin>
            <xmax>20</xmax>
            <ymax>5</ymax>
        </bndbox>
    </object>
</annotation>
```

Figure A.3: XML code