International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING

ISSN:2147-6799 www.ijisae.org Original Research Paper

Detection of Unauthorised photography in Prohibited places through CCTV using Machine Learning

Mrs. A. Jeba Sheela[1], Mrs. M. Gowthami[2], Balaji B[3], Balaji S[4], Hemanth Kumar U[5], Akash R[6], Arjun SV[7], Dhanvanth S[8]

Submitted: 29/01/2024 **Revised**: 07/03/2024 **Accepted**: 15/03/2024

Abstract: Unauthorized photography is one of the most common and less accountable crimes. Because more and more people have access to a wider range of photographic technologies, it is becoming more difficult to ensure improved security and privacy protection in a number of contexts. Points of interest include company headquarters, government buildings, research centers, military stations, and other private areas where protecting privacy and avoiding illegal data collection are crucial. A YOLO based model helps to reduce the impact of such crimes and preserve privacy while the current surveillance infrastructure. In order to prevent unwanted data collection, the YOLO classifier in conjunction with the MoveNet for action detection offers a workable solution that promotes a more secure society. The societal impact of this technology is substantial, as it addresses growing concerns related to privacy infringement, industrial espionage, and unauthorized data gathering. With the help of the DarkNet backbone and Adam Optimiser an efficient system with a mean average precision of 89.7% and a mAP50-95 of 80% is achieved swiftly in around 150ms. Additional by using YOLOv8s and YOLOv8m parameter accuracy is improved by 2.1% and 3.3% respectively against the more lightweight and faster YOLOv8n. Objectness scores are penalized and scored based on the Binary Cross-Entropy Loss to improve object localization.

Keywords: *CCTV, CNN, MoveNet, Openpose, Pytorch, RPi, YOLO*

1. Introduction

The widespread use of surveillance systems, especially Closed-Circuit Television (CCTV), in today's networked world has made them indispensable for protecting a variety of settings, from public areas to critical installations. Though surveillance equipment has been widely used, the problem of unauthorized activity still exists and poses serious security hazards, especially in restricted access locations. Unauthorized photography and videography stand out among these activities as possible hazards because they can compromise private information, violate rights to privacy, and jeopardize security procedures.

It will take creative solutions that take advantage of developments in artificial intelligence and surveillance technology to address this serious security issue. Through the integration of CCTV systems and machine learning techniques, the study provides a comprehensive solution to address the identification of unauthorized activities, notably illicit photography, in places that

are prohibited.

Designing and implementing an expert system that can analyze live video streams from CCTV cameras placed in critical areas is the main goal of the project. The technology used in the project will identify and notify authorities of any instances of photographing activity, irrespective of the individual's authorization status, by utilizing machine learning methods, namely deep learning models. The system will rapidly send out notifications for the proper response from security staff by recognizing particular visual clues and patterns linked to photography, like the existence of cameras or smartphone devices held in a capturing position.

The complexities of the system development process is emphasized by carefully examining performance evaluation measures, putting advanced preprocessing techniques into practice, and applying strict model selection criteria. Furthermore, the ethical considerations and privacy issues raised highlight how crucial it is to use surveillance technology responsibly in order to protect responsibility, openness, and individual rights.

2. Literature Survey

Basketball is a sport where only one player is allowed to keep hold of the ball at a time for the play to be considered legal. To track a player holding the ball accurately during the course of the game requires high quality images which is usually the norm for large scale games[21]. The model used is capable of detecting players of different players with the help of their jersey numbers. To separate and find the disparity between the teams the dataset is preprocessed using color contrasting to identify the changes in player jerseys of different teams. The ground truth and the model prediction are compared until a valid Intersection over Union(IoU) has been achieved. With the help of path tracing

Computer Science and Engineering, Easwari Engineering College, Chennai, Tamil Nadu, India.
jebasheela.soft@gmail.com [1], gowthami.m@eec.srmrmp.edu.in [2], bbalaji0702@gmail.com [3], balajidaran@gmail.com [4], u.hemanthkumar4@gmail.com [5], writerakash@gmail.com [6], aarjunbala67@gmail.com [7], dhanvanth23x@gmail.com [8]

and it is possible to gauge the relationship among the players. The output is a tensor of 13 x 13 x (5 x 5 + C) dimensions, where C is the number of classes proposed to be bounding boxed or annotated. The algorithm used is based on YOLO because it is more robust than its contemporaries such as Fast-RCNN and Faster-RCNN. Softmax activation function along with NMS (Non-Maximal Suppression) is used to evaluate the confidence of the classifier. YOLO algorithm is compared with Joy2019 algorithm for further analysis.

Detection of ripe tomatoes through computer vision is a fascinating subject[22]. A custom dataset is used to compare models of YOLOv5 through YOLOv8. The authors have chosen to omit models prior to YOLOv5 because they have been outdated for the presented experiment. According to the results, the performance of YOLOv5 and YOLOv6 series were subpar to YOLOv7, this was even without consideration for the parameters. When comparing with the YOLOv8 series, YOLOv8l and YOLOv8m models managed to achieve better results and brought about higher GFLOPs. The reason for choosing YOLOv7 has to do with its capacity to achieve a brilliant balance between the parameters and the GLOPs. A open source dataset VOC207 is required to solve the generalization problem arising with the model while using OpenCV. SimpleImg is used to annotate the dataset for training with results of multiple deep learning models being compiled for better data visualization.

A similar problem involving analysis of fruit yet again only this time a different use case that focuses on the health of the plant. Another addition to this use case is the utilization of YOLOv8 in the experiment to help identify and detect pests or other potential diseases that harm plant life[23]. A highly compact model of just around 4.5 megabytes is capable of achieving amazing results in about 3.4 milliseconds of time. With the utilization of a dual backbone network the models recognition capability is enhanced. The drawback to the model is its poor efficiency during adverse weather effects that bring in plenty of noise, enough to significantly reduce detection speed with reduced precision. Additionally the model fails at predicting pests of smaller size often leading to no detections which is a huge problem in identification and localization of the targets. In a ideal state with good weather and proper conditions the models accuracy is top notch be it mAP or framerate speed.

In [24] General performance metric comparison, detection techniques, localization methods and other challenges in using the YOLO classifier for object detection are compiled and summarized. Explanations on why foreground and background needs to be diverse are briefed. Correlation between dataset size and accuracy of the model are demonstrated with how the performance is affected. Localisation errors due to discrepancies in the background through approximations in the background are discussed. Miscrosoft’s COCO dataset is used to explain the convolution with the architectural designs of CNNs. The improvements based on increase in network size and parameters have also been given for interpretation. Explains how non relevant and duplicate bounding boxes are omitted using NMS the further they deviate from the ground truth.

A method that uses human poses as an estimate to classify day to day human actions through changes in the keypoints of the perceived human skeleton is employed to aid the growing population and detect events of falls that could lead to grave injuries[25]. A LSTM (Long Short Term Memory) based network is used to perform pose classification. Different outcomes for a variety of networks are performed to estimate and gauge their accuracies. Pose estimation methods such as DCPose, AlphaPose, MoveNet, OpenPose and OpenPifPaf are tabulated with their mAPs and FPS for the analysis of speed and precision. Pose estimation of a human being is a challenging task that involves various factor to be taken into consideration such as camera angle, lighting conditions, occlusions and so on.

Hand gesture recognition in [26] uses the IPN hand dataset and the SHREC’22 dataset which view hand gestures in four different types – static, dynamic coarse, dynamic fine and periodic. Transfer learning is used to train the custom model as per the use case requirements. Levenshtein accuracy is used as a metric for evaluation. Levenshtein accuracy is a metric obtained by the division of the Levenshtein distance by the number of total ground truth gestures in the sequence. Flattened 3D models of hand were used to as inputs for the training set. Optimizer opted by the authors for training the dataset is the Adam optimizer. Learning rate is also constantly changed from time to time in the dataset split into 80:20 for training and testing.

3. Proposed Solution

A delicate combination of cutting-edge approaches was skillfully staged inside the tight framework of developing an advanced machine learning model focused on detecting unauthorised photography within protected locations. The YOLO (You Only Look Once) classifier serves as the foundation of this complex system. The model is designed to run smoothly on resource-constrained systems such as the Raspberry Pi and similar microcontrollers, and it uses CCTV cameras' surveillance capabilities to record and analyse video footage in high-security zones. The YOLO algorithm is the key player capable of detecting over 9000 distinct classes at a time. YOLO was built and based on top of GoogleNet originally. Later iterations of YOLO have used DarkNet and ResNet as a suitable backbone for specific use cases. Modern YOLOv8 models are typically implemented using the CSPDarkNetBackbone architecture. YOLO image segmentation in particular uses the CSPDarkNet53 backbone. A look into the architecture reveals all of YOLOv8's arguments such as stackwise_channel, stackwise_depth, include_rescaling, activation, input_shape and input_tensor. Unlike other CNN algorithms which use multiple passes, YOLO uses a single shot detector (SSD) which requires only a single pass through the input image to get ready for detection.YOLO achieves this by using object localization technique wherein an image is divided into N x N grids. In each of these grids, the unwanted grids are eliminated through means of probabilities. The elimination is done based of a probability function that is calculated on the chance of the object being in the grid. Bounding box algorithms then come into play, enabling precise object detection and the assignment of confidence values based on a pre-trained dataset. This foundational step proves indispensable in identifying a diverse array of objects within the live feed, laying the groundwork for subsequent in-depth analyses.

Understanding YOLO:

The backbone is the core of any CNN architecture. One of the commonly used YOLO architecture utilizes Spatial Pyramid Pooling Fusion. Spatial Pyramid Pooling (SPP) is a technique used to capture information at different scales by dividing the input feature map into regions of different sizes and performing pooling operations on each region independently. SPPF is a popular variation of this technique. BRA or the Bounding-Box Refinement Algorithm is the technique used to refine bounding boxes generated by the object detection model.

The model's architecture has a strong emphasis on computational efficiency, which is crucial for deployment on platforms like as the Raspberry Pi. As the system matures, it remains acutely aware of the multiple obstacles given by different surroundings, always optimising its performance to efficiently protect critical sites. YOLO also has very little preprocessing steps compared to CNN models making detection faster. Because Singe Shot Detection algorithms have lesser layers than traditional models, more frames can be processed at a lesser time interval. Convolution networks need to perform the most number of FLOPs in as little time as possible for rapid detection. FLOPs is a measure of the number of floating point operations performed. This parameter is useful in quantifying the performance with the computing power available.

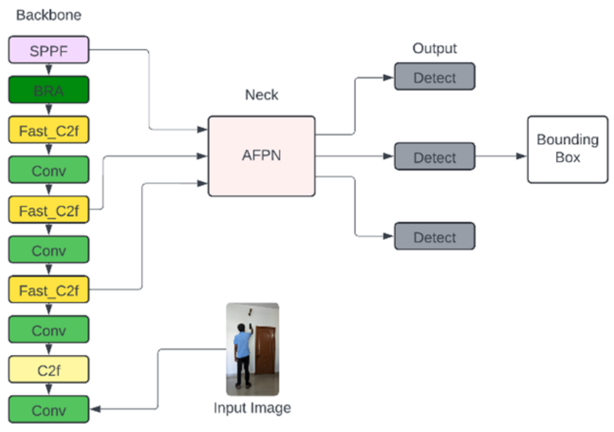

Fig. 1. YOLO architecture

It involves adjusting the coordinates of the bounding boxes to more accurately fit the detected objects. Fast_C2f (Fast Context-to-Focus) is any method or module designed to efficiently incorporate contextual information into the object detection process, helping the model focus on relevant regions of the image for detection. Conv-convolutional layers are fundamental building blocks in convolutional neural networks (CNNs). These layers perform convolution operations to extract features from input images and carry them over to the next stage with changes.

C2f-Context-to-Focus is similar to Fast_C2f and refers to a mechanism within the YOLO architecture that helps the model understand context and focus on relevant regions for object detection. The AFPN (Adaptive Feature Pyramid Network) can dynamically adjust the scale or structure of the feature pyramid based on the input image or task requirements.

YOLO is computationally superior to other CNN models such as RCNN or FRCNN. But since YOLO is only a single shot detector, it might lose on some parameters that CNN models could extract. This means for the purpose of object tracking and feature extraction CNN is still superior. For the use case presented, YOLO is undoubtedly the better algorithm given that the pros outweigh the cons.

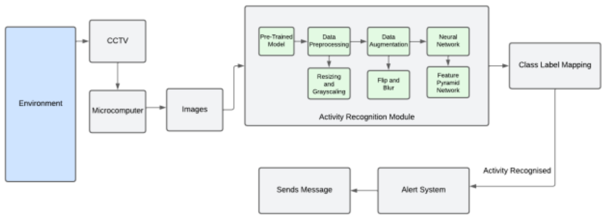

Fig. 2. System Design

To detect events, the system makes use of a 2 megapixel dome style CCTV camera. When compared to cameras with higher resolutions, 2MP dome-type CCTV cameras are typically less expensive. When there are financial constraints, this can be helpful. Dome-type cameras frequently have wide-angle coverage,

which makes it possible to monitor a bigger region with a single camera. With a higher chance of catching pertinent events within the camera's field of view, this broad coverage can help with event identification. Compared to other CCTV camera kinds, these cameras are less noticeable and intrusive due to their dome appearance. When placing the cameras in settings where subtlety or aesthetics are valued, this can be helpful. Dome cameras are often simple to install and need little work to put on walls or ceilings. Both the deployment procedure and installation costs are streamlined as a result.

The amount of detail caught in the footage, however, might be constrained by the camera's 2MP resolution. The precision of event detection may be impacted by this, particularly when identifying small or far-off objects. The clarity and visibility of items in the video may be affected by lower resolution cameras' tendency to produce lower quality images. It might become more difficult to correctly recognise people or detect occurrences as a result. As opposed to other types of cameras, dome-type cameras frequently have restricted zoom capabilities. Zooming in on particular regions of interest for a deeper look may become more difficult as a result. However, the 2MP resolution of the camera may limit the amount of detail captured in the video. This could have an effect on event detection precision, especially when identifying small or distant objects. Because lower resolution cameras tend to create lower quality images, this could have an impact on the clarity and visibility of items in the video. As a result, it could becoming harder to identify individuals or recognise events. Dome-type cameras often have limited zoom capabilities, in contrast to other kinds of cameras. This may make it harder to zoom in on specific areas of interest for a closer examination.

3.1. Visual Object Identification

The system's initial module is dedicated to a complete collection of operations for quickly processing input photos from CCTV cameras in congested areas. The step-by-step processes provide a well-organized workflow, guaranteeing that visual data is properly prepared and analysed for later phases in the system. The module accepts photos taken by CCTV cameras in congested environments[4]. These photos are used as the major data source for the ensuing study. The incoming photographs go through a preprocessing stage where they are scaled to the necessary dimensions. This scaling not only improves image quality, but it also provides the groundwork for precise object recognition in the subsequent phases of the process. TensorFlow, a powerful machine learning framework, is used for the specific purpose of mobile phone categorization. This phase entails recognising and separating mobile phones among the numerous things in the scene. The categorization method is useful in identifying gadgets of interest. Following [8] object detection and classification, the module uses a procedure to create bounding boxes around the detected objects. These bounding boxes spatially separate tagged items from the backdrop, giving a visual representation of the objects' existence in the image. Furthermore, the related labels are presented to improve interpretability. This precise series of processes lays the groundwork for the model's succeeding stages, particularly those centred on more complex analyses such as action

detection and behavioural analysis. The accuracy gained in object detection, particularly the identification and categorization of mobile phones, sets the groundwork for the system's overall effectiveness in detecting unauthorised photography in secure situations. These functionalities are seamlessly integrated, ensuring that the model is well-equipped to deal with the problems given by dense and dynamic visual environments.

YOLO models, as previously indicated, are single-stage object detectors. In a YOLO model, visual frames are enhanced by a backbone. These properties are integrated and blended in the neck and then passed on to the head of the network. YOLO predicts the locations and types of objects around which bounding boxes should be created. Since its inception in 2015, YOLO models have been increasingly popular in the industry. The compact architecture enables novice ML engineers to rapidly learn about YOLO, while the real-time inference performance allows practitioners to devote little hardware computing to power their applications. The effectiveness of the YOLO network's convolutional layers in the backbone is critical for efficient inference speed. Cross Stage Partial Networks marked the beginning of its journey towards maximum layer efficiency. Object detection models are often offered as a series of models that scale up and down in size to meet the needs of various applications. The depth, breadth, and resolution of the network are often taken into account by object detection models. Re-parameterization strategies entail averaging a collection of model weights to construct a model that is more resilient to the general patterns that it is attempting to represent. There has been an emphasis on module level re-parameterization, where each portion of the network has its own re-parameterization strategy. The YOLO network head produces the final predictions for the network, but because it is so far downstream, it may be desirable to add an auxiliary head somewhere in the middle. During the training process, the algorithm supervises both the detecting head and the head that will generate predictions. Python is used to write training scripts, data loaders, and utilities. The yaml file contains a number of critical parameters such as the directory paths for the train and test datasets, object labels, and a list of names for each label.

3.2. Temporal Activity Detection

To perform action detection and a series of crucial functions, enhancing the system's capability to detect and categorize human actions, particularly those indicative of unauthorized photography, within crowded and dynamic environments. The module begins by processing video frames sourced from CCTV cameras. This includes a preprocessing stage where normalization and resizing operations are applied[5]. These preparatory steps optimize the video frames for subsequent feature extraction and analysis within the MoveNet model.

3.2.1 Action Classification

Building upon the spatiotemporal features obtained from MoveNet, the module proceeds to the crucial task of action detection. MobileNetV2 is the backbone of the MoveNet pose estimation model. MoveNet involves identifying and categorizing human actions, specifically targeting instances of individuals taking photographs. Additionally, the system has the capability to implement face detection and behavior analysis, providing a more comprehensive approach to understanding human activities within the surveillance environment. Subsequent to action detection, the module computes confidence scores for the recognized actions.

MoveNet predicts keypoint heatmaps and offset vectors. The heatmap predicts the likelihood of each keypoint being present at each pixel location in the image. The offset vector predicts the displacement needed to move from each pixel location to the exact position of the keypoint.

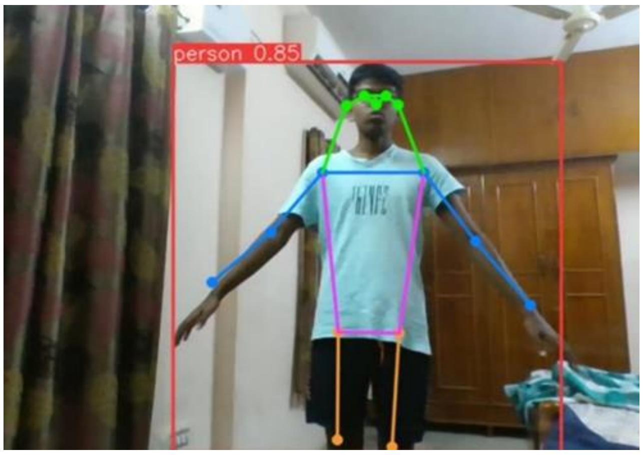

Fig. 3. Pose keypoints on the skeleton of a person

MoveNet is a family of pose detection models developed by Google, specifically designed for detecting human body poses in images and videos. The goal of MoveNet is to accurately and efficiently estimate key points on the human body, such as joints and limbs, in real-time. Google has released different versions of MoveNet, including MoveNet Single Pose and MoveNet Multi Pose. These models are often used in applications related to human-computer interaction, fitness tracking, augmented reality, and other areas where understanding human poses is essential. MoveNet is great with human skeleton identification. This means through the power of MoveNet it is possible to collaborate the obtained behavioural data which later correlates classified data to the user being tracked. By drawing a skeleton on the human in sight, the model learns about positions to take into consideration for future references and there by learns through reinforcement.

One alternative to MoveNet is using PoseNet. The PoseNet tool detects key body points in human figures using the PoseNet model. PoseNet predicts the 2D coordinates of body joints directly from the input image, without relying on intermediate heatmaps[11]. It uses a regression-based approach where the network directly outputs the coordinates of each keypoint. The model runs with either a single-person or multi-person detection algorithm. PoseNet is built using lightweight CNN architectures, making it efficient to deploy on various platforms, including mobile devices and web browsers. Keypoint Detection in PoseNet predicts the 2D positions of human body keypoints, typically including keypoints for the head, neck, shoulders, elbows, wrists, hips, knees, and ankles. These keypoints can be used to reconstruct the pose of a person in the image or video frame. Pose confidence scores along with keypoint positions, provides confidence scores for each detected keypoint, indicating the model's confidence in the accuracy of the prediction. These scores can be used to filter out unreliable keypoints or estimate the overall confidence of the detected pose. Google has also released PoseNet as part of the TensorFlow.js library, allowing developers to use it directly in web applications using JavaScript or integrate it into other TensorFlow-based projects. In contrast, MoveNet[24], a newer model, offers enhanced accuracy and robustness while maintaining efficiency.

Leveraging lightweight deep neural networks (DNNs) optimized for mobile and edge devices, MoveNet strikes a balance between accuracy and computational cost. Its architecture is designed to handle occlusions and crowded scenes better, making it more robust in challenging scenarios. MoveNet is particularly well-suited for applications requiring precise and real-time pose estimation, such as sports analytics, fitness tracking, and human-computer interaction.

When choosing between PoseNet and MoveNet, it is critical to examine the application's unique requirements. PoseNet thrives in circumstances requiring real-time processing and speed, such as augmented reality and gesture detection applications. MoveNet, on the other hand, excels at jobs that need a high level of precision and resilience, such as sports analytics and fitness tracking, where exact pose estimate is critical for accurate performance analysis. Finally, the decision between PoseNet and MoveNet is based on the trade-offs between accuracy, efficiency, and resilience that the application requires.

Pose estimation algorithms can help identify specific body poses associated with taking a photo using a mobile phone. For example, the pose of holding a phone in a horizontal position with both hands extended could indicate that the person is taking a photo. By recognizing these key poses, the model can focus its attention on relevant frames captured by the CCTV. Pose estimation can provide valuable temporal information by analyzing how body poses change over time[29]. The model can assess whether the phone is being held up in a manner compatible with taking a photo or whether it is being used for another function, such as texting or making a call, by analysing the spatial interaction between the person and the mobile phone. Models may be taught to withstand fluctuations in illumination, background clutter, and occlusions. This resilience allows the model to reliably recognise poses associated with snapping a shot even in tough CCTV conditions. Methods usually focus on recognising body positions rather than face characteristics, which might assist protect privacy in surveillance circumstances. This emphasis on body positions enables the model to detect photo-taking behaviour while preserving individuals' privacy by obtaining precise facial shots. Pose estimation can supplement object detection algorithms by giving more information about the person's movements and intentions. By integrating posture estimate and object identification, the model can detect photo-taking behaviour with more accuracy and reliability. Overall, including pose estimate approaches into the model can improve its capacity to detect when a person is taking a photo with their mobile phone in a CCTV scenario, resulting in more effective surveillance and security solutions. PoseNet and MoveNet are both important breakthroughs in posture estimation, with various advantages depending on the application environment. As demand for real-time, accurate pose estimation continues to expand across numerous areas, knowing the strengths and limits of each model is critical for efficiently deploying them in practical scenarios.

3.2.2 Dataset

Dataset: To solve events related to photography events, the MUID-IITR dataset is sufficient enough. Previously available datasets encountered significant limitations, including minimal variations in foreground-background imagery and specialization for specific activities, as well as a lack of high-quality images with proper annotation.

Our dataset addresses these limitations by offering more generic, higher quality images depicting people using phones in diverse environments and scenarios. For annotation, the labelImg tool was employed to meticulously create labels, utilizing "compound-bounding boxes" for both hands and devices to ensure accurate detection of usage. The dataset is broadly divided into two subsets: 540 positive image instances featuring individuals using mobile phones and 348 negative instances depicting no observable phone usage. The primary purpose of the dataset is mobile phone usage detection, with potential applications including the detection and prevention of usage in inappropriate places (e.g., classrooms, confidential meetings), endangering situations (e.g., while driving, on elevated surfaces), unfair practices (e.g., in examination halls, pirating private artistic properties), and incidents of theft or loss.

In addition to the dataset above the training has been further increased by using a custom made dataset which contains over one thousand images of people being idle and carrying a mobile phone. The dataset is pre-processed by resizing it into 640x640 resolution fit and grayscaling. Later the dataset in augmented by flipping the image horizontally to add variations to the datset. In the end, the dataset had nearly tripled from its original volume. The dataset is then split in the ratio of 70:20:10 for training, testing and validation.

3.3 Embedded System Integration

Integration with the Raspberry Pi, a microcontroller device that controls the surveillance equipment. The module performs a number of critical duties, including ensuring compatibility, optimising hardware setup, integrating software components, creating user-friendly interfaces, and enabling real-time communication. The key object is to guarantee that the module seamlessly integrates with the Raspberry Pi while adhering to the microcontroller system's standards. This compatibility is required for the surveillance setup to function well within the Raspberry Pi's resource-constrained environment. Configuration settings are rigorously optimised to meet the Raspberry Pi's particular hardware specs. This ensures that the system functions at peak efficiency, making the most use of existing resources and capabilities.

3.3.1 Software Integration:

The software components are adapted to the Raspberry Pi's operating system environment. This painstaking customisation provides seamless interaction with current systems and applications, improving the overall stability and coherence of the monitoring system. The module prioritises user-friendly interface development, with the goal of making it easier to set up and configure the surveillance system when it is connected to a Raspberry Pi. These interfaces are intended to be simple and user-friendly for system administrators.

The module's primary job is to establish and ease real-time connection between the surveillance system and the Raspberry Pi. This two-way communication channel allows for immediate replies to identified events or abnormalities, guaranteeing that the system can adapt flexibly to changing conditions in the monitoring environment[18].It is important to the successful deployment of the surveillance system on the Raspberry Pi by coordinating these actions. The rigorous attention to compatibility, hardware configuration, software integration, user interfaces, and real-time communication guarantees that the system runs smoothly. As the microcontroller system, the Raspberry Pi becomes a key hub, coordinating the project's operations and offering a user-friendly interface for effective monitoring in constrained locations.

3.3.2 System Evaluation

Last phase in the development lifecycle, focuses on comprehensive testing, validation, and parameter definition to ensure the optimized algorithm's robustness and reliability in real-world applications. The functions of this module are designed to study efficiency, correctness, obtain results, and visualize the acquired data, contributing to a thorough understanding of the algorithm's performance under various conditions.

Subjecting the optimized algorithm to rigorous testing and validation processes through thorough Testing and Validation. These tests assess the algorithm's efficiency, correctness, and performance across diverse scenarios. The objective is to uncover any potential issues, refine the algorithm, and enhance its overall effectiveness before deployment in real-world settings. The testing phase is used to compare the trained batch labels as a reference to ground truth. The allocated dataset for the testing phase is around 20%. 10% of the dataset is exclusively dedicated for validation testing. As part of the evaluation process, the module includes mechanisms for obtaining and visualizing results. Visualization tools contribute to a comprehensive understanding of the algorithm's behavior under varying conditions.

4. Results

This section details all findings from the employed methodology. From calculating mean average precision to recall and other important parameters such as speed and accuracy are visualized. The confusion matrix explains the performance of the model along with its labels. Overlapping cells in the x axis and y axis indicate the number of instances predicted properly. Mean Average Precision (mAP) is defined as the number of true positives divided by the number of true positives plus false positives. The mAP metric takes into account both the precision and recall of the model and is calculated as the mean of the average precision for each class. A higher mAP value indicates a better performance of the model.

Average Precision (AP) is a measure of the precision of the model at different recall levels. The precision is defined as the number of true positives divided by the number of true positives plus false positives. The recall is defined as the number of true positives divided by the number of true positives plus false negatives. AP is calculated as the area under the precision-recall curve. A higher AP value indicates a better performance of the model. Intersection over Union (IoU) is a measure of the overlap between the predicted bounding box and the ground-truth bounding box. It is calculated by taking the intersection areas of the two boxes and dividing it by the area of their union. A higher IoU value indicates a better match between the predicted and ground-truth bounding boxes. There are also several other metrics that evaluate the accuracy too. For example, True Positive Rate (TPR), False Positive Rate (FPR), F1-score, and Log Average Miss Rate (MR). In addition to these metrics, models are also be evaluated based on their computational efficiency.

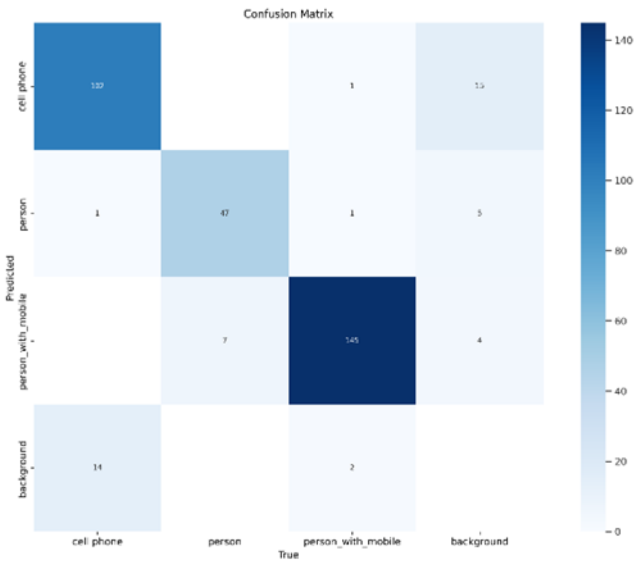

Fig. 4. Confusion Matrix of the trained model

The confusion matrix explains the number of times the trained model predicts the correct label in the duration of time and details all instances where the actual label has been misassigned to another class. The classes given are “cell phone”, “person” and “person_with_mobile”. The additional “background” is a class used to label the area out of the region of interest decided during annotation of the dataset. A normalized confusion matrix can also be used in places where the mean and standard deviation of data appear coherent. The normalization is generally given as:

$$X = x - \mu / S.D \quad (1)$$

Where X is the new layer value,

x is the current value

μ is the mean obtained from the batch

S.D is the standard deviation of the normalized batch.

This way normalization can help stabilise and speedup the training process. It is also pivotal for convergence and improving model performance through the learning rate.

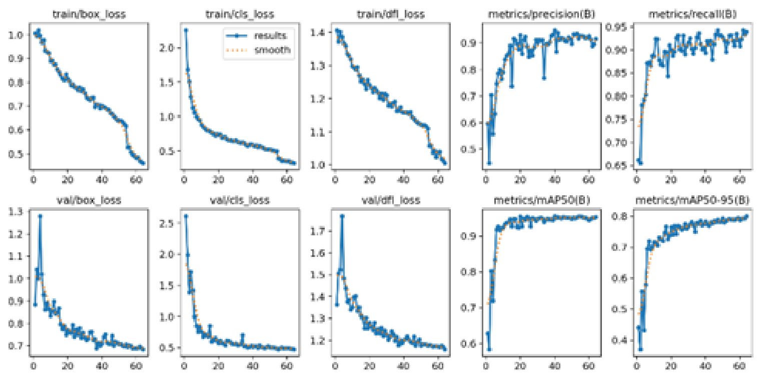

Fig. 5. Training results of YOLOv8n (x-axis = epochs, y-axis= quantity of parameter change per epoch)

Epoch	*Train/box_loss*	*Train/cls_loss*	*Train/dfl_loss*	*Metrics/precision(B)*	*Metrics/recall(B)*	*Metrics/mAP50(B)*	*Metrics/mAP50-95(B)*
1	1.007	2.2601	1.4076	0.5967	0.66233	0.62817	0.44108
10	0.88994	0.87981	1.2984	0.78953	0.88598	0.92345	0.71641
20	0.8139	0.73877	1.2412	0.93018	0.90022	0.95462	0.76404
30	0.75471	0.65654	1.2099	0.90153	0.91518	0.94901	0.76617
40	0.70304	0.58211	1.1577	0.94989	0.89359	0.95577	0.78338
50	0.63768	0.51725	1.122	0.8972	0.92759	0.9478	0.79295
60	0.48479	0.35353	1.0381	0.92657	0.90877	0.94442	0.78685
64	0.46127	0.32372	1.0056	0.91525	0.93935	0.95352	0.80052

Table 1. Performance metrics and loss

The model has run for exactly 64 epochs. These epochs are meant to improve the quality of the ML model as iterations increase. Using a NVIDIA or other GPU(Graphical Processing Unit) is highly recommended while training a custom YOLO model. Even google collaboratory provides GPU support using the Google Compute Engine or similar cloud based GPU environments. The training results showcase the loss curves. The box loss explains how much the model steers away from wrong classification using loss functions as the number of iterations increase. Initially there was high box_loss which has to do with IOU regions not being close to the ground truth. By increasing the iterations the total box_loss of the model had reached a steep drop at around the 57th

epoch. Then the curve stabilizes and reduces the loss in a steady rate meaning further training beyond this is a waste of resources and could lead to overfitting of train data. The class loss (cls_loss) similarly is a measure of how accurately the classes have been predicted. Again there is a steep dip at the 57th epoch. The bottom row has the same parameters as the top row, but the bottom row is only the results of a very small part of the dataset used for validation of the learning model. The performance metrics mAP measures the average precision across all classes at various intersection over union (IoU) thresholds. IoU is the a measure of how well the predicted bounding box overlaps with the ground truth bounding box. mAP50 represents the mean Average Precision calculated at a specific IoU threshold of 0.5. Only the detections with an IoU of 0.5 or higher with the ground truth bounding box are considered correct. On the other hand, mAP50-95 represents the mean Average Precision calculated across a range of IoU thresholds from 0.5 to 0.95, with increments of 0.05. This provides a more comprehensive evaluation of the model's performance across a wider range of IoU thresholds, capturing both strict and lenient criteria for correct detections. The model showcases a respectable mean average precision of 89.7%.

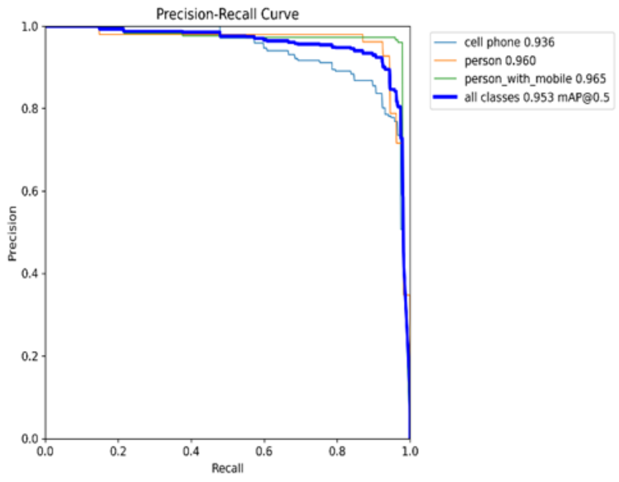

Fig. 6. Precision-Recall curve

The x-axis represents recall, and the y-axis represents precision. Precision is the ratio of correctly predicted positive observations to the total predicted positives. It measures the accuracy of the positive predictions. Mathematically, it is defined as:

Precision= True Positives/(True Positives + False Positives) (2)

Recall, also known as sensitivity or true positive rate, is the ratio of correctly predicted positive observations to all actual positives. It measures the completeness of the positive predictions. Mathematically, it is defined as:

Recall= True Positives/(True Positives + False Negatives) (3)

The curve starts from the point (0,1) where precision is 1 and recall is 0, meaning all instances are classified as negative. It ends at (1,0) where precision is 0 and recall is 1, meaning all instances are classified as positive. A higher area under the curve (AUC) indicates better performance of the model across different threshold settings.

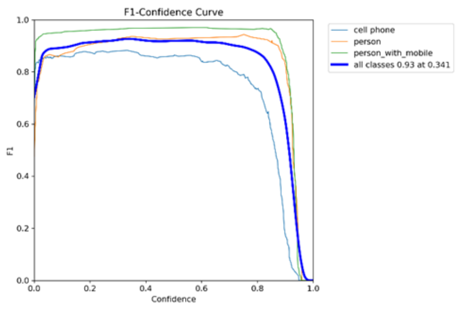

Fig. 7. F1-Confidence Curve

F1 score is a metric commonly used in binary classification tasks. It is the harmonic mean of precision and recall, calculated using:

F1 = 2 × precision × recall / (precision + recall) (4)

Precision measures the accuracy of positive predictions, while recall measures the completeness of positive predictions. F1 score provides a balance between precision and recall. In the graph obtained the mean confidence is about 80% for all classes.

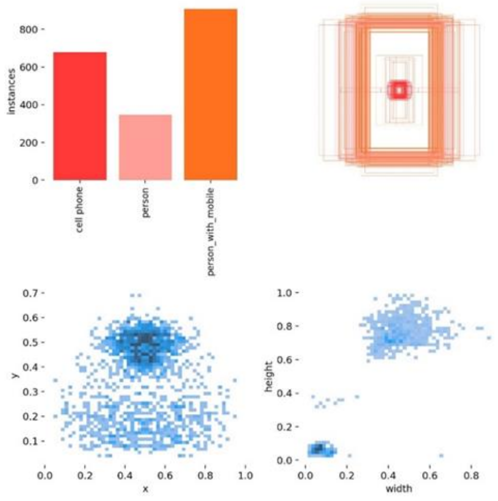

Fig. 8. Labels and Bounding Boxes

Fig.8 illustrates the number of bounding boxes for the training dataset. The reason for the increased number of boxes is due to the augmentation flip of the dataset. This increase makes the bounding boxes appear as mirror images. X and Y axes explain the location of bounding box objectness.

5. Conclusion

The proposed system succeeds in classifying the recognized events properly when provided with ideal conditions such as zero occlusion, high clarity images, proper camera angles with positions and sufficient computational power. The technology efficiently identifies instances of unapproved photography by utilising advanced object recognition algorithms and real-time analysis of CCTV footage. Its capacity to distinguish between certain things and questionable activities improves security protocols all around. Moreover, smooth deployment and integration guarantee effective operation throughout the range of locations that are monitored. All things considered, this system provides a proactive means of protecting security and privacy, which makes it an invaluable tool in monitoring and surveillance settings. YOLO algorithm is extremely proficient when paired with detection tasks. Under the ./keras/utils path lies methods to change optimisers and backbones such as ResNet, EfficientNet, ResNeXt and so on. An average confidence of 80% is a sign of a brilliant model which will further only improve.

Future work should address challenges posed by varying lighting conditions to ensure robust performance across different environments. YOLO struggles with object tracking compared to CNN models and hence there is certainly room for improvement. Additionally, exploring use cases beyond unauthorized photography detection, such as crowd monitoring or object tracking, could broaden the system's applicability. Incorporating algorithms that consider diverse physical attributes of individuals, including weight and height, can improve the accuracy of person identification. Furthermore, enhancing the system's adaptability to dynamic environments and complex scenarios will be crucial for its effectiveness in real-world settings. Finally, conducting extensive field testing and validation across diverse locations and scenarios will be essential for refining and optimizing the system's performances.

References and Footnotes

[1] A. Datta, M. Shah, and N. Da Vitoria Lobo. “Person-on-person violence detection in video data”. In IEEE International Conference on Pattern Recognition, volume 1, pages 433–438, 2002.

[2] O. Deniz, I. Serrano, G. Bueno, and T-K. Kim. “Fast violence detection in video”. International Joint Conference on Computer Vision, Imaging and

Computer Graphics Theory and Applications, pages 478–485, 2014.

[3] Hassner, Y. Itcher, and O. Kliper-Gross. “Violent flows: real-time detection of violent crowd behavior”. In IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, pages 1–6, 2012.

[4] Y. Gao, H. Liu, Xi. Sun, C. Wang, and Y. Liu. “Violence detection using oriented violent flows. image and vision computing”, 48-49(2015):37–41, 2016.

[5] P.C. Ribeiro, R. Audigier, and Q. Cuong. “RIMOC, a feature to discriminate unstructured motions: application to violence detection for video-surveillance”. Computer Vision and Image Understanding, 144:121–143, 2016.

[6] Tao Xiang, Shaogang Gong, “Incremental and adaptive abnormal behaviour detection, computer vision and image understanding”, Volume 111, Issue 1, 2008, Pages 59-73, ISSN 1077-3142.

[7] V. Gajjar, Y. Khandhediya, and A. Gurnani, “Human detection and tracking for video surveillance: a cognitive science approach”, in Proceedings - 2017 IEEE International Conference on Computer Vision Workshops, ICCVW 2017, Institute of Electrical and Electronics Engineers Inc., Jul. 2017, pp. 2805–2809. doi: 10.1109/ICCVW.2017.330.

[8] O. Sang-Hyun, K. Jin-Suk, B. Yung-Cheol, P. Gyung-Leen, B. Sang-Yong, “Intrusion detection based on clustering a data stream”, in: Third ACIS International Conference on Software Engineering Research, Management and Applications, 2005, pp. 220–227.

[9] M.S. Ryoo, J.K. Aggarwal, “Stochastic representation and recognition of high-level group activities”, International Journal of Computer Vision 93, 2011 183–200.

[10] N. Hoose, “Computer vision as a traffic surveillance tool”, IFAC Proceedings Volumes, Volume 23, Issue 2, 1990, Pages 57-64, ISSN 1474-6670,

[11] T. Ellis, “Co-operative computing for a distributed network of security surveillance cameras”, IEE European Workshop Distributed Imaging (Ref. No. 1999/109), London, UK, 1999, pp. 10/1-10/5

[12] P. Wonghabut, J. Kumphong, T. Satiennam, R. Ung-Arunyawee, and W. Leelapatra, “Automatic helmet-wearing detection for law enforcement using cctv cameras”, in IOP Conference Series: Earth and Environmental Science, Institute of Physics Publishing, Apr. 2018.

[13] H. Dammalapati and M. Swamy Das, “An efficient criminal segregation technique using computer vision,” in Proceedings - IEEE 2021 International Conference on Computing, Communication, and Intelligent Systems, ICCCIS 2021, Institute of Electrical and Electronics Engineers Inc., Feb. 2021, pp. 636–641.

[14] K. Lloyd, P. L. Rosin, A. D. Marshall, and S. C. Moore+, “Violent behaviour detection using local trajectory response,” 2016.

[15] A. Wiliem, V. Madasu, W. Boles, and P. Yarlagadda, “A suspicious behaviour detection using a context space model for smart surveillance systems,” Computer Vision and Image Understanding, vol. 116, no. 2, pp. 194–209, Feb. 2012, doi: 10.1016/j.cviu.2011.10.001.

[16] L. M. Fuentes and S. A. Velastin, “Tracking-based event detection for cctv systems,” Pattern Analysis and Applications, vol. 7, no. 4, pp. 356–364, Aug. 2005, doi: 10.1007/s10044-004-0236-z.

[17] S. Tanjila Naurin, A. Saha, K. Akter and S. Ahmed, "A proposed architecture to suspect and trace criminal activity using surveillance cameras," 2020 IEEE Region 10 Symposium (TENSYMP), Dhaka, Bangladesh, 2020, pp. 431-435, doi: 10.1109/TENSYMP50017.2020.9230901.

[18] A. F. D. Marsiano, I. Soesanti and I. Ardiyanto, "Deep learning-based anomaly detection on surveillance videos: recent advances," 2019 International Conference of Advanced Informatics: Concepts, Theory and Applications (ICAICTA), Yogyakarta, Indonesia, 2019, pp. 1-6, doi: 10.1109/ICAICTA.2019.8904395.

[19] S. Shirsat, A. Naik, D. Tamse, J. Yadav, P. Shetgaonkar and S. Aswale, "Proposed system for criminal detection and recognition on cctv data using cloud and machine learning," 2019 International Conference on Vision Towards Emerging Trends in Communication and Networking (ViTECoN), Vellore, India, 2019, pp. 1-6, doi: 10.1109/ViTECoN.2019.8899441.

[20] M. Grega, S. Łach and R. Sieradzki, "Automated recognition of firearms in surveillance video," 2013 IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support (CogSIMA), San Diego, CA, USA, 2013, pp. 45-50, doi: 10.1109/CogSIMA.2013.6523822.

[21] Y. Yoon et al., “Analyzing basketball movements and pass relationships using realtime object tracking techniques based on deep learning,” IEEE Access, vol. 7, pp. 56564–56576, 2019.

[22] J. Guo et al., “Revolutionizing agriculture: real-time ripe tomato detection with the enhanced tomato-yolov7 system,” IEEE Access, vol. 11, pp. 133086–133098, 2023.

[23] D. Luo, Y. Xue, X. Deng, B. Yang, H. Chen, and Z. Mo, “Citrus diseases and pests detection model based on self-attention yolov8,” IEEE Access, vol. 11, pp. 139872–139881, 2023.

[24] T. Diwan, G. Anirudh, and J. V. Tembhurne, “Object detection using yolo: challenges, architectural successors, datasets and applications,” Multimed Tools Appl, vol. 82, no. 6, pp. 9243–9275, Mar. 2023.

[25] Z. Dozdor, Z. Kalafatic, Z. Ban, and T. Hrkac, “TY-Net: transforming yolo for hand gesture recognition,” IEEE Access, vol. 11, pp. 140382–140394, 2023.

[26] S. Juraev, A. Ghimire, J. Alikhanov, V. Kakani, and H. Kim, “Exploring human pose estimation and the usage of synthetic data for elderly fall detection in real-world surveillance,” IEEE Access, vol. 10, pp. 94249–94261, 2022.

[27] X. Zhao, Z. Li, Y. Liu, and Y. Xia, “A progressive decoding strategy for action recognition,” IEEE Access, vol. 11, pp. 92424–92432, 2023.

[28] F. Sajid, A. R. Javed, A. Basharat, N. Kryvinska, A. Afzal, and M. Rizwan, “An efficient deep learning framework for distracted driver detection,” IEEE Access, vol. 9, pp. 169270–169280, 2021.

[29] Z. Zhou, F. Shi, and W. Wu, “Learning spatial and temporal extents of human actions for action detection,” IEEE Trans Multimedia, vol. 17, no. 4, pp. 512–525, Apr. 2015.

[30] M. Mudgal, D. Punj, and A. Pillai, “Suspicious action detection in intelligent surveillance system using action attribute modelling,” Journal of Web Engineering, vol. 20, no. 1, pp. 129–145, Feb. 2021