International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 13 Issue: 02 | Feb 2026 www.irjet.net p-ISSN: 2395-0072

Daroda Pratibandhak-An Anti Robbery System to protect shops and Banks from Robbery using Deep Learning and IoT

R. S. Kakade[1], Adak Rupesh Hanumant[2], Labade Vivek Anil[3], Shejul Devendra Shirish[4]

[1]*HOD, Dept. of Computer Technology, P.Dr.V.V.P. Institute of Technology and Engineering (Polytechnic), Loni, Maharashtra, India*

[2,3,4] *Final year Diploma Student, P.Dr.V.V.P. Institute of Technology and Engineering (Polytechnic), Loni, Maharashtra, India*

Abstract - *There is a direct correlation between the increase in youth unemployment and the exponential growth in the number of young people without the necessary skills. A rise in criminal activity, including robbery and burglary, is generally accompanied by these variables. Retailers, including jewelers and others, bear a disproportionate share of the crime rate associated with armed robberies. Offenders frequently hide their identities after an incident because they cover their faces. auther can successfully employ AI to prevent these outcomes and promptly notify the authorities in order to curb this threat. For improved accuracy, auther train the weapon dataset—which includes knives and revolvers—using a deep learning model called EfficientDet D4. The system starts receiving live feeds from the CCTV cameras after the data has been trained. The proposed model promptly activates the IOT model and issues a loud audio alert whenever a shoplifter brandishes a weapon against the store owner. When the IoT model is turned on, an Arduino-driven motor is instructed to automatically close the store's door. A loud audio alert is then raised, and a WhatsApp alert is sent to the closest police station, along with a photo of the store and its location on a map.*

***Key Words*: AI, EfficientDet D4, closed-circuit television, internet of things (IoT) paradigm**

1. INTRODUCTION

Traditional surveillance isn't always adequate to protect financial institutions and retail locations from high-stakes crimes like armed robberies in today's technologically advanced world. Traditional CCTV systems are better for forensic investigation than for protecting people or property. To overcome this issue, financial and retail institutions need an automated anti-robbery system. This digital sentry might employ AI's lightning-fast decision-making to anticipate and eliminate threats. This technology's debut is crucial since crimes can happen faster than guards or staff can activate alarms. These technologies inform authorities quickly if they detect suspicious activities or risks. This creates a constant line of defense without human fatigue or distraction, dramatically reducing response times, and ending bloodshed. The research relies on Google AI's net 4 deep learning model's robust convolutional neural network design. Its unique blend of processing speed and precision makes it famous. The efficient b4 model of net 4 scales depth, width, and input resolution with its revolutionary compound scaling technique. It was chosen for this reason. This balanced scaling improves feature extraction with fewer parameters than conventional models, which solely increase layer count to improve performance. Retail and banking applications require low-latency HD video processing on commodity hardware or edge devices. To simulate a well-trained computer brain, the model meticulously evaluates each frame of the video for intricate visual patterns that indicate danger, such as the precise outlines of a weapon or the feel of a mask, while ignoring background features.

The project carefully blends several cutting-edge approaches to ensure the identification process works in varied situations. After pre-training on ImageNet, a successful net 4 model is fine-tuned using hundreds of weapons, masks, and suspicious stance photographs. This method focuses on transfer learning. This allows the system to focus on a problem while using its enormous library of visual attributes. Using backdrop matting and anomaly detection is another key method. Equipment can detect non-threatening "abandoned objects" from ordinary consumer flow. Real-time object identification and multi-object tracking monitor visitors' movements throughout the facilities for hostility or excessive time in prohibited areas. The system provides comprehensive and scalable security by combining optical identification with automated alert protocols like the simple mail transmission protocol for timely notifications. It transforms static surveillance into smart, dynamic defense.

[1] Adding class and location labels to each training image is proposed by Keong-hun choi et al. to improve supervised learning for object detection. The ability of the new environment to identify things outside of the training context depends on the presence of a matching label. Our research presents an object detection system that utilizes reinforcement learning. A few photos and an inventory of the contents will do the trick. Three models are proposed: one for area-based evaluation, one for incentive configuration, and one for transformer-based item proposal. [2] Khalid Elgazzar and colleagues investigated the application of deep learning to object detection. Using the training capabilities of Convolutional Neural Networks (CNNs), thousands of objects can be reliably detected even in tough lighting and occlusion settings. The availability of a large number of training images

allows CNNs to achieve remarkable accuracy across several classes. Finding and classifying items in photos is a breeze with modern CNN-based object detectors. [3] The work of Ramesh Chandra Poonia et al. An enormous stride forward in the fight against crime and for public safety has been the development of live CCTV object detection. In state-of-the-art video surveillance systems, the YOLO models accurately identify and categorize objects.

An examination of earlier research that was deemed a Litereature Survey is presented in the second part of this publication. Section 3 provides a comprehensive description of the proposed methodology, outlining the path of action. The experimental evaluation is covered in Part 4, possible modifications are discussed in Section 5, and the essay concludes with a conclusion on the existing plan.

2. LITERATURE SURVEY

[4] Deep convolutional neural networks (CNNs) can identify items in complex landscapes collected by high-resolution remote sensing photos, regardless of their size, according to a two-stage process developed by Qiyang Xie et al. Part of the method are three smaller networks. Feature extraction from deep pictures can be started by utilizing a feature extraction backbone network. Two others are a multi-scale region proposal generator (MS-RGN) and a fine-grained category and object position FC-RCN. Through comparison trials, the authors show that the method outperforms state-of-the-art algorithms on the FAIR1M aircraft category dataset. No matter the size of the object or the degree of fine-grained type coverage, this remains true. In order to greatly enhance the network's detection accuracy, future research will center on comprehending the rotational characteristics of objects in remote sensing images.

[5] To tackle the sparsity of 3D LiDAR data and the inherent occlusion issues, Minho Cho et al. presented a new method called 3D LiDAR-based Multi-Object Tracking (MOT). This method enhances tracking accuracy across a variety of object morphologies by using multi-positive contrastive learning to increase object similarity at different distances. In addition to reducing ID-switching mistakes and providing strong tracking, deep reinforcement learning is utilized for effective handling of obstructed targets. Compared to previous 3D LiDAR and camera-LiDAR fusion methods, the suggested approach improves Order Tracking Accuracy (HOTA) scores while reducing ID-switching on the KITTI MOT dataset.

[6] The early detection of classified gunshot noises helps with both incident management and crime prevention (Ali raza et al., 2015), which is an important part of criminal investigations. The main objective of this research is to find gunshots by employing a meta-learning approach. For our experiments, we used a benchmark dataset for gunshot noises that included 851 audio samples taken from freely available videos on YouTube.

[7] That was made by Peng Wang and colleagues. A considerable portion of RS research is devoted to developing methods for object detection in RS photos. Contemporary methods often increase the number of network layers to extract more granular data from the initial attributes. To achieve their impressive performance improvement, deep neural networks really need a big amount of labelled data training samples.

[8] Going beyond their object detection score when it came to identifying the hand wrist crease, Gokulakrishnan Elumalai et al. proved that innovative method of object detection by combining HWC-Yolov8x with the Yolov5 and Yolov8 models. We could tweak the HWC-Yolov8x configuration's hyperparameters to get an even better object detection score. The results of our study indicate that biometric identification using the hands and wrists could replace existing biometric techniques in the near future.

[9] Safa Riyadh Waheed et al. presented a state-of-the-art object detection system that combines TL with SSMD (Semantic Segmentation and Object Detection). The first step was to prepare a model for transfer learning by pre-training it. Following this, the KITTI dataset was used to pre-train the SSMD-512 and SSMD-300 models. Using a set of strict performance metrics, the model's performance was evaluated thoroughly.

[10] Using a combination of Dense ASSP and backdrop matting, Mingu Jeong et al. demonstrated a way to locate abandoned objects. The proposed method minimized the occurrence of false positives by utilizing pre-processing, abandoned object recognition (AOR), and abandoned object decision by feature correlation (AODFC). This effectively addressed the shortcomings of current systems for abandoned object detection.

[11] A wireless object recognition system that makes use of the Channel State Information (CSI) of WiFi channels has been studied for deployment, according to Igor Bisio et al. The main goal was to thoroughly assess the system's performance using cutting-edge CSI-based target identification algorithms. Author accomplished this by creating a system that can gather CSI data, clean up the raw data, extract valuable information from the amplitude component, and categorize commonplace things.

[12] N. Harihara Valliappan et al. evaluated YAMNet's ability to identify gun models by implementing transfer learning and making selective hyperparameter adjustments. Through rigorous testing of the proposed model with varying parameter values during training, the authors demonstrate that YAMNet with fine-tuned hyperparameters surpasses state-of-the-art deep learning and hybrid methods. Our research focuses on investigating

and implementing additional Keras layers to enhance the model's training efficiency.

[13] A technique for estimating pitch location using a single camera and a radar gun was introduced by Daishi Kitano et al. By doing away with complicated setups and costly equipment, the approach aspires to offer an accessible and accurate answer. Because of this, it is ideal for use in real-life field situations, especially by amateur baseball teams.

[14] This was proposed by Sara Azizian Amiri and colleagues. Temperatures above 400 degrees Celsius are possible with certain electrosurgery conditions. The purpose of this research was to improve the design of the smart electrosurgical knife in order to solve important problems including heat-induced optical alterations and debris attachment. When tested in real-world scenarios, including interrupted electrosurgery modes, the second version—which used quartz and PTFE tubes—showed the most promise.

[15] The difficulty of continuously cutting a large number of complicated sponge samples is addressed in this study by Kun Xie et al., which suggests an S-shaped sorting method based on region segmentation. Effective continuous cutting of sponge samples is made possible by this procedure, which specifies the sample's entrance and exit positions.
The first step is to collect samples at the molecular level.

3. PROPOSED METHODOLOGY

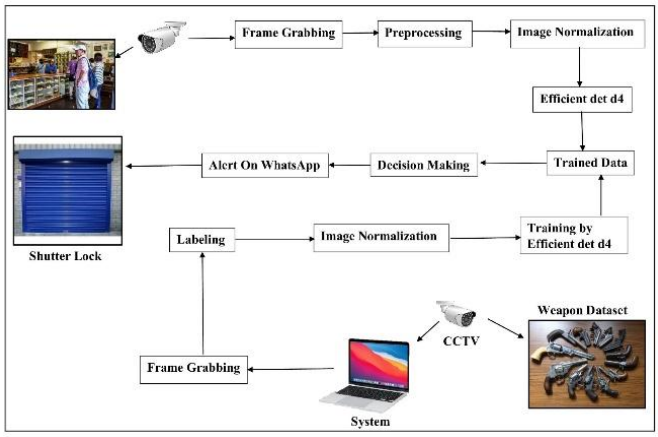


Figure 1: System overview

This step follows a description of the steps used in the overview diagram to drive the anti-robbing from the shops and Banks.

3.1 Step 1: Dataset Generator

Before anything else, you'll need to use OpenCV to capture pictures of the weapons that are breaking into the stores. In order to verify the model's detection capabilities, the Video Capture method in the CV2 package takes pictures of the target weapon. All of the pictures of weapons are kept in the dataset folder. Some models, like the gun, are currently under development by the model. A folder is created to contain the acquired photos, which will be utilized in the subsequent model process.

3.2 Step 2: Data Labelling

Here, we use the labeling software to annotate the previously taken photographs. In order to label a rectangle, the user must input the image into the labeling software and mark the coordinates of its top right corner (x1, y1) and bottom right (x2, y2). The acquired coordinates are saved in an.xml file, which is subsequently utilized to train the model with the help of an effective det d deep learning network.4.

3.3 Step 3: Installation of APIs

The TensorFlow Object Detection API needs to be installed in this Google Colab instance first. Executing a few installation scripts and cloning the (https://github.com/tensorflow/models)are necessary for this. The following code portions can be run by clicking the play button. The next step in downloading and extracting cuDNN files is to install the conda environment. The next thing to do is to copy the tensorflow model's repository off of GitHub. And then the Object Detection API has to be installed.

3.4 Step 4: Upload Image Dataset and Prepare Training Data

This is where we get ready to prepare the TensorFlow training data by uploading our training photos and using the TF Record generating routines. In order to create TF Records from our data, we need to upload our photos, divide them into a train, validation, and test folder, and then execute scripts. To begin, on our local PC, create a single folder named "images.zip" and bundle all of our training images and XML files into it. The files must to be contained within the zip folder. After it's uploaded, we need to extract its contents and configure our picture folders using a few commands. In this particular instance, the file system's /content subdirectory is where these directories are generated. To access the file system, we can use the "Files" icon located on the left side of the screen.

3.5 Step 5: Split images into train, validation, and test folders

Locate our "images.zip" file among the listed ones; it's the folder icon on the left side of the screen. The next step, after the dataset has been uploaded, is to extract its contents and create folders to store the photographs. In this particular instance, the file system's /content subdirectory is where these directories are generated. By utilizing the "Files" symbol situated on the left, we are able to peruse the

file system. The following step is to divide the photos into three sets: train, validation, and test. The purpose of each set is as follows:

3.5.1 Train: To train the model, these are the real photographs. At each stage of training, the neural network is fed a new batch of photos from the "train" set. In these pictures, the network identifies what things are and where they are located. The loss is computed by the training method, which then uses back propagation to modify the network weights.

3.5.2 Validation: The training algorithm can utilize the images from the "validation" set to evaluate the training progress and make hyperparameter adjustments, such as the learning rate. These photos, in contrast to the "train" images, are only utilized on a periodic basis throughout the training process, specifically, once every specific number of steps.

3.5.3 Test: Throughout training, the neural network does not see these images. A human being should utilize them to do the last tests on the model to see how accurate it is.

3.6 Step 6: Create TF Records: Last but not least, we must transform the photos into TF Records, a data file format utilized by TensorFlow for training purposes. The data is automatically converted into TF Record format using Python programs. We must first establish a class label map before we can run them. A "labelmap.txt" file with a list of classes can be generated by executing the code section below. Make a new line for each of our classes (e.g., "gun" and "knives") and replace the words "class1" and "class2" with them. After that, press play to run the code. As a result, the object detection model's detectable classes are recorded in a "labelmap.txt" file.

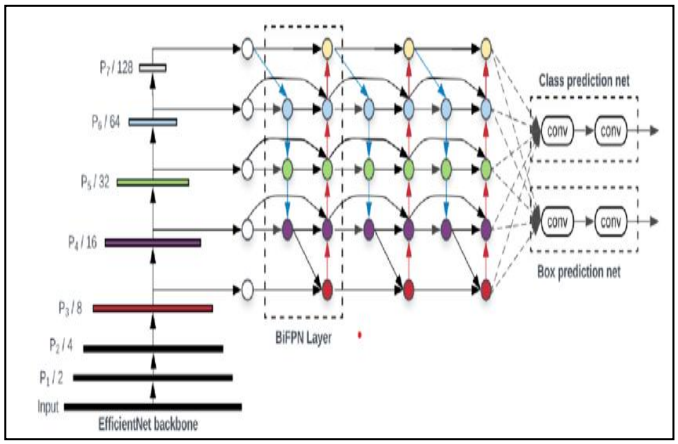
3.7 Step 7: Set Up Training Configuration

An effective configuration for training the det d4 model is being built up in this section. Here, we're naming the TensorFlow 1 Object Detection Model that we'd like to employ as our starting point. A configuration file is included with each model. This file does a lot of things, such setting training parameters (like learning rate and total number of training steps) and pointing to file locations. We are making changes to the custom training job's configuration file. The first part of the code establishes many filenames that will be utilized to download the model and configuration file later on and lays out various models that are accessible in the TF1 Model Zoo. This makes it simple to keep track of the models we're using and to add more models as needed.To train with a specific model, set the "chosen_model" variable to its name. The "efficient det d4 -quantized" model is presently selected. After that, specify the parameters for the pre-trained model and the configuration file, and then press play in the following three steps to download them.After getting the model and configuration files downloaded, the next step is to add some general training parameters to the configuration file. Training phases are controlled by the following variables:

3.7.1 num_steps: The sum of all the steps that will be used to train the model. Forty thousand steps is a decent starting point. If we find that the loss measures are still going down when training is over, we can add extra steps. It takes more time to train if there are more steps. If the loss level reaches the specified point before the training period ends, training can be terminated early.

3.7.2 batch_size: The number of photos to utilize for each training stage. Although training a model with a higher batch size requires fewer steps, the size can only be as big as the GPU RAM that is available for training. In most cases, 16 GPUs is an adequate amount for a Colab instance.

3.7.3 quant_delay_steps: (Intended solely for training with quantization in mind) There are a lot of processes involved before the training algorithm simulates quantization by adding "fake" quantization nodes to the network. Halving the total number of training steps is a decent place to start. At this stage, you also provide additional training data, such as the total number of classes, the location of the configuration file, and the file containing the pre-trained model. In order to apply the training parameters that we have just defined; we will need to edit the configuration file. This code will take the.config file you downloaded and save it as our own "pipeline_file.config" with all of the required parameters already filled in. Add our dataset, model checkpoint, and training parameters to the main pipeline file to create a custom configuration file. The subsequent block updates the training script such that checkpoints are saved every 1000 steps. To adjust the frequency of saving checkpoints, change 'num_eval_steps'. Table 1 below shows the design of the efficient det d4.



3.8 Step 8: Train Custom TF1 Object Detector Our object identification model training has begun! The TF Object Detection API's "model_main_tf2.py" script is used to train the model. In earlier sections of this Colab, we have defined

all the arguments and parameters utilized by'model_main_tf2.py'. Model, batch size, and total number of training steps determine how long the training process takes, which can range from two to six hours. The result is a. tflite file that we can use to put the model through its paces.

3.9 Step 9: Testing the model for weapon detection

Here, the Python software uses the mobile phone's camera to record video and, by extension, the frames. It does this by utilizing the Droid Cam app, which is compatible with both laptops and mobile phones. In order to identify the fire engine and ambulance in the live-streaming frames, the trained model file. tflite is utilized. The location of the stores is communicated to the appropriate authorities as soon as firearms are identified, and neighbors are also alerted through voice alert. The last step in preventing shoplifting is to utilize the ESP 32 model to lock the doors.

4. RESULTS AND DISUSSIONS

The Windows-based system, which has 16 GB of main memory and an Intel Core i7 processor, is used to implement the proposed model. The Anaconda IDE repository for Spyder and Jupyter IDEs is used by the model for the experiment. The created model is rigorously evaluated using the parameters of the confusion matrix. You may understand the parameters of the confusion matrix using the following equations: accuracy, precision, recall, and macro F1.

$$\text{Accuracy}=(TP+TN)/(TP+FP+TN+FN) \quad -(1)$$

$$\text{Precision}(P)=TP/(TP+FN) \quad -(2)$$

$$\text{Recall}(R)=TP/(TP+FP) \quad -(3)$$

$$\text{Macro-F1}=(2*P*R)/(P+R) \quad -(4)$$

At this point, we have TP for true positives, TN for true negatives, FP for false positives, and FN for false negatives.

The obtained results are shown below,

1.Confusion Matrix

2.F1-Confidence Curve

3. Precision Confidence Curve

4. Precision- recall Curve

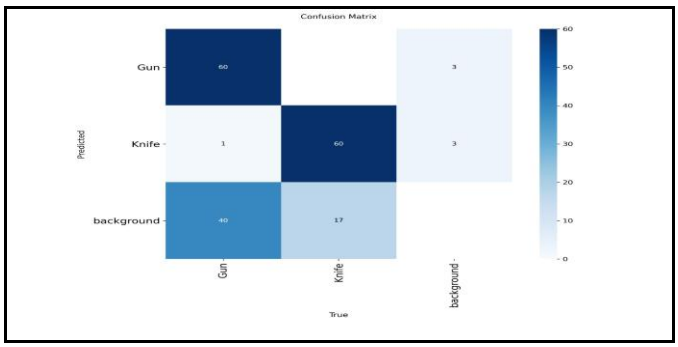


Figure 2: Confusion Matrix

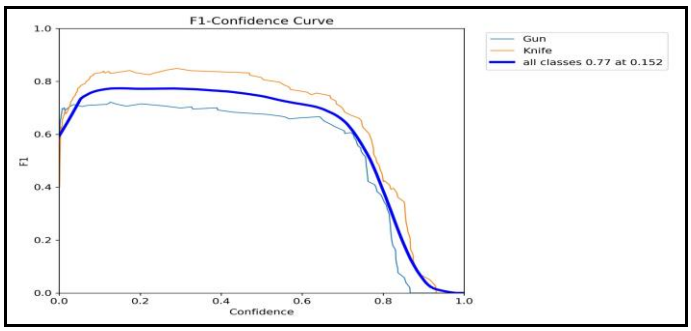


Figure 3: F1-Confidence Curve

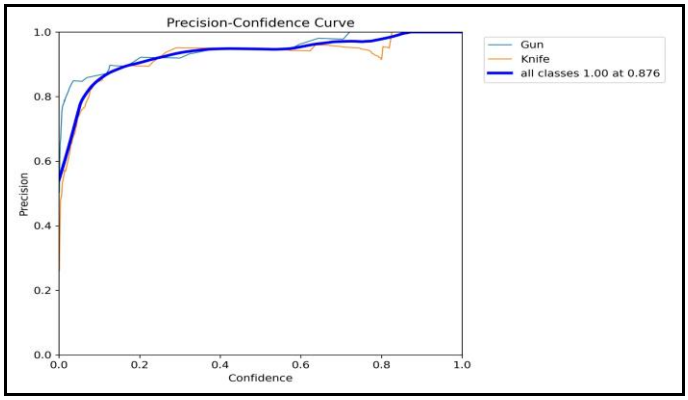


Figure 4: Precision-Confidence Curve

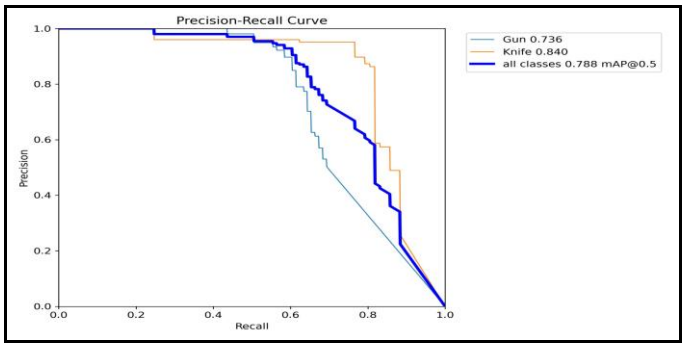


Figure 5: Precision-Recall Curve

Figure 6: Result

5. CONCLUSION and Future Scope

One major departure from the old model of passive surveillance is the creation of automated anti-robbery systems for banks and stores. This research achieves the difficult balancing act of meeting the rigorous demands of fast processing and cutting-edge detection accuracy with the help of the EfficientNet-D4 deep learning model. Even in the busy, often dimly lit environments of banks and shopping malls, the EfficientNet architecture's built-in compound scaling mechanism guarantees accurate feature extraction, whether it be identifying a person wearing a mask or a weapon. Incorporating multi-modal techniques such as YAMNet for auditory gunshot recognition and backdrop matting for abandoned object detection, in addition to visual analysis, provides a thorough security framework. The system's accuracy and its ability to adapt to new situations with little retraining are both enhanced by the incorporation of transfer learning and meta-learning. In the end, this system offers a constant and scalable solution that improves the safety of both public and private areas by decreasing the likelihood of human mistake, increasing the speed of emergency response times, and deterring criminal activities.

Several exciting new directions for development and application lie ahead for this technology. Improving the model for tinyml and edge computing is a top priority. This will make high-level security available to smaller retail enterprises by enabling the efficient net 4 architecture to run directly on low-cost cameras and smart devices.

REFERENCES

[1] K.-H. Choi and J.-E. Ha, “Object Detection Method Using Image and Number of Objects on Image as Label,” IEEE Access, vol. 12, pp. 121915–121931, 2024, doi: 10.1109/ACCESS.2024.3452728.

[2] K. Elgazzar, S. Mostafi, R. Dennis, and Y. Osman, “Quantitative analysis of deep learning-based object detection models,” IEEE Access, vol. 12, pp. 70025–70034, May 2024, doi: 10.1109/ACCESS.2024.3401610.

[3] R. C. Poonia, K. Upreti, N. Singh, J. Kesarwani, and M. S. Alam, “An efficient detection of suspicious objects from dynamic video surveillance by fusion-based multiview deep learning techniques,” Journal of Mobile Multimedia, vol. 21, no. 1, pp. 1–26, 2025, doi: 10.13052/jmm1550-4646.2111.

[4] Q. Xie, D. Zhou, R. Tang, and H. Feng, “A deep CNN-based detection method for multi-scale fine-grained objects in remote sensing images,” IEEE Access, vol. 12, pp. 15622–15630, Jan. 2024, doi: 10.1109/ACCESS.2024.3356716.

[5] M. Cho and E. Kim, "3D LIDAR Multi-Object Tracking Using Multi Positive Contrastive Learning and Deep Reinforcement Learning," IEEE Access, vol. 13, pp. 12447-12461, 2025, doi: 10.1109/ACCESS.2024.3521334.

[6] A. Raza, F. Rustam, B. Mallampati, P. Gali, and I. Ashraf, "Preventing Crimes Through Gunshots Recognition Using Novel Feature Engineering and Meta-Learning Approach," IEEE Access, vol. 11, pp. 103115-103131, 2023, doi: 10.1109/ACCESS.2023.3316695.

[7] P. Wang, X. Wang, E. Ji, Y. Zhang, J. Du, X. Wang, and X. Zhang, "Multilayer Feature Extraction Object Detection Based on Deep Forest," IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 18, pp. 27224–27241, 2025, doi: 10.1109/JSTARS.2025.3622047.

[8] G. Elumalai and G. Malathi, "Deep Learning Based Hand Wrist Crease Object Detection Models: An In-Depth Analysis," IEEE Access, vol. 12, pp. 83125–83139, 2024, doi: 10.1109/ACCESS.2024.3409087.

[9] S. R. Waheed, N. M. Suaib, M. S. M. Rahim, A. R. Khan, S. A. Bahaj, and T. Saba, "Synergistic Integration of Transfer Learning and Deep Learning for Enhanced Object Detection in Digital Images," IEEE Access, vol. 12, pp. 13525–13536, 2024, doi: 10.1109/ACCESS.2024.3354706.

[10] M. Jeong, D. Kim, and J. Paik, "Practical Abandoned Object Detection in Real-World Scenarios: Enhancements Using Background Matting With Dense ASPP," IEEE Access, vol.12,pp.60808–60825,2024,doi: 10.1109/ACCESS.2024.3395172.

[11] I. Bisio, C. Fallani, C. Garibotto, A. Grattarola, F. Lavagetto, A. Sciarrone, S. Zappatore, and M. Zerbino, "Performance Evaluation of Machine/Deep Learning-Based Object Recognition Techniques Leveraging Channel State Information Using a Real Testbed," IEEE Access, vol. 12, pp. 98680–98692, 2024, doi: 10.1109/ACCESS.2024.3428612.

[12] N. Harihara Valliappan, S. Dhanraj Pande, and S. Reddy Vinta, "Enhancing Gun Detection With Transfer Learning and

YAMNet Audio Classification," IEEE Access, vol. 12, pp. 58940–58949, 2024, doi: 10.1109/ACCESS.2024.3392649.

[13] D. Kitano and D. Mikami, "Pitch Location Measurement Using a Single Camera and Radar Gun," IEEE Access, vol. 13, pp.175336–175340,2025,doi: 10.1109/ACCESS.2025.3618720.

[14] S. Azizian Amiri, J. Dankelman, and B. H. W. Hendriks, "Enhancing Intraoperative Tissue Identification: Investigating a Smart Electrosurgical Knife's Functionality During Electrosurgery," IEEE Transactions on Biomedical Engineering, vol. 71, no. 7, pp. 2119–2134, July 2024, doi: 10.1109/TBME.2024.3362235.

[15] K. Xie and Z. Zhu, "Optimization Cutting Trajectory for Ring Knife Sponge Cutting via S-Shaped Sequencing Algorithm Based on Region Segmentation," IEEE Access, vol. 11,pp.99319–99330,2023,doi: 10.1109/ACCESS.2023.3313636.