Enhancing Surveillance Systems leveraging AIoT for Advanced Object Detection in Real-Time Security Applications

Monish Sai Krishna Namana
Department of Electrical, Electronics and Communication Engineering, Gandhi Institute of Technology and Management, Visakhapatnam, India
mnamana@gitam.in (corresponding author)

B. Udaya Kumar
Department of Electrical, Electronics and Communication Engineering, Gandhi Institute of Technology and Management, Visakhapatnam, India
ubudidi@gitam.edu

Received: 12 December 2024 | Revised: 28 January 2025 and 19 February 2025 | Accepted: 21 February 2025

 DOI: https://doi.org/10.48084/etasr.9926

ABSTRACT

Surveillance systems are integral for modern security frameworks, ensuring safety in public areas, traffic monitoring, and infrastructure protection. The rising demand for surveillance necessitates advanced technologies capable of accurately identifying objects such as vehicles and pedestrians in real time. A promising solution is the use of AI technologies that has proven their ability in object detection with exceptional accuracy. This study proposes an optimized object detection model using YOLOv8, incorporating comprehensive hyperparameter tuning through the Optuna framework, significantly enhancing object detection accuracy and response time in dynamic environments. The model was trained on the PASCAL VOC 2012 dataset. Our results highlight a robust model performance, achieving a precision of 86.2%, a recall of 78.9%, a mean Average Precision (mAP) of 84.7%, and an F1-score of 81%, which surpasses previous benchmarks for this dataset, particularly in challenging crowded scenarios. Integrating the Internet of Things (IoT) with AI forms Artificial Intelligence of Things (AIoT), a revolutionary step that leverages interconnected devices for real-time, automated data processing and rapid response in complex scenarios, addressing scalability and responsiveness in security applications. Our model detection results are transmitted to the ThingSpeak IoT platform, enabling efficient real-time monitoring and analysis, demonstrates the feasibility of using AIoT to develop scalable, high-accuracy surveillance systems with potential applications across urban safety, traffic management, and infrastructure security.

Keywords-surveillance; YOLOv8; AI; IoT; object detection

I. INTRODUCTION

Surveillance systems are increasingly used for public safety and for protecting private and governmental properties in various urban areas, infrastructures, and transport networks. Object detection in surveillance systems plays a pivotal role in accurately detecting and identifying objects of all sizes regardless of light, and weather conditions [1]. Object detection enables machines to recognize, track, and detect objects automatically [2]. With the advancements in machine learning and deep learning algorithms, object detection performance has significantly improved under various conditions and scenarios [3]. Two object detection methods are broadly classified, namely two-stage and single-stage detection approaches. The two-stage approach involves generating potential regions of the object's location and then refining these regions, in which classification and bounding box regression are performed. In the single-stage approach, on the other hand, the detection process is streamlined to simultaneously predict the bounding boxes and class probabilities. Examples of the two-stage approach are the Regions with Convolutional Neural Networks (R-CNN), Fast R-CNN, and Faster R-CNN [4], while an example of a single-stage approach is the You Only Look Once (YOLO) Single Shot Detector (SSD) [5].

YOLO has acclaimed prominence in these single-stage detectors due to its speed and accuracy [6], enabling real-time object detection, pivotal for surveillance applications [7]. Several versions of YOLO models have made significant advancements in object detection. Particularly, YOLOv8 was designed on the strengths of the previous generations by

providing more precise predictions, improved efficiency, and accuracy [8, 9].

Authors in [10] introduced a novel deep neural network model, A-YONet, which integrates YOLO and MTCNN to improve object detection and is implemented within an end-edge cloud architecture for optimized data processing, using the PASCAL VOC 2007 and 2012 datasets. It employs a multilevel feature fusion technique alongside a clustering-based pre-adjustment scheme for anchor boxes to enhance accuracy. Authors in [11] presented an improved YOLOv4 algorithm for remote sensing image target detection, improving accuracy and robustness through modifications in no maximum suppression thresholds and anchor frame allocation schemes, validated by experiments on the DOTA dataset. Authors in [12] proposed an enhanced fire detection system for smart cities utilizing the YOLOv8 algorithm, which enhances accuracy, reduces false alarms, and integrates into a multi-layered smart city framework for real-time data processing and various safety applications.

The Internet of Things (IoT) is a network of interconnected devices which collect store and exchange data through embedded devices, sensors, software, and other technologies. Integrating IoT with Artificial Intelligence (AI) leads to the creation of Artificial Intelligence of Things (AIoT) [13, 14]. This combination allows surveillance systems to process, collect, and analyze data in real-time application scenarios. AIoT utilizes the strengths and advantages of AI and learning from vast data from IoT, bringing it together to improve the decision-making process [15, 16]. In surveillance applications, more automated, accurate, and efficient monitoring and response systems can be developed to predict and react to potential security threats with minimal human oversight [17, 18]. This combination of IoT and AI transforms passive monitoring systems into dynamic monitoring systems, which will help enhance the scalability and effectiveness of the surveillance systems [19, 20]. Authors in [21] proposed a lightweight deep learning-based Intelligent Edge Surveillance (INES) method for Industrial Internet of Things (IIoT) applications that combines edge and cloud computing to reduce computational costs and network traffic, validated in a construction site scenario with high detection precision and low operating costs. Authors in [22] proposed PG-YOLO, a lightweight object detection algorithm optimized for edge devices in IIoT environments, achieving significant model compression and speed improvements while maintaining high accuracy. Authors in [23] proposed an AIoT system named Edge YOLO, redesigning the network structure based on YOLOv4, and comparing and evaluating the Edge-Cloud Computing (E-CC) solution with Federated Learning (FL)-based cloud computing, using the COCO2017 and KITTI datasets. Authors in [24] developed and proposed a novel object detection model, FL-YOLO, utilizing depthwise separable convolution and downsampling inverted blocks for real-time edge analysis. Cloud computing facilitates the training and optimization of edge models, whereas edge computing manages real-time video analysis. Authors in [25] presented the lightweight multiscale object detection algorithm YOLO-SK, which enhances the YOLOv5s model by integrating a weighted dense feature fusion network and a Selective Kernel (SK) attention prediction head, thereby improving the model's object detection capacity across various scales.

In this study we propose a new methodology for object detection, using the YOLOv8 algorithm combined with parameter tuning with the Optuna technique, using the PASCAL VOC 2012 for training, and the KITTI dataset for further validation. The performance of the proposed model is compared with the performance of other object detection models in the same dataset. Furthermore, we integrate the proposed model into the ThingSpeak IoT platform for real-time data transmission and monitoring.

II. METHODOLOGY

A. Architecture of YOLOv8

The YOLOv8 architecture is developed using various essential blocks, used for speed, efficiency, and performance in real-time object detection. A representation of the YOLOv8 architecture is shown in Figure 1. The C2f block begins by splitting the input features into parts, processing only one part through bottleneck layers, while the other bypasses, and then concatenating the results before passing them through another convolutional layer, helping capture multiple feature levels [26, 27]. The bottleneck block includes two convolutional layers, optionally with a shortcut connection to mitigate the vanishing gradient problem and enhance feature propagation, making it crucial for capturing fine-grained details [28]. The Spatial Pyramid Pooling Fast (SPPF) block aggregates features at multiple scales through max-pooling layers, enabling the detection of objects at varying sizes by concatenating outputs and refining them with a convolution. The fundamental Conv block consists of a 2D convolution, batch normalization, and Sigmoid Linear Unit (SiLU) activation, extracting spatial features, stabilizing training, and introducing non-linearity for learning complex patterns [29]. The detector handles the object detection by predicting bounding boxes and classification scores. The YOLO architecture is divided into three primary components: backbone, neck, and head, each with specific roles in the object detection process, shown in Figure 1.

1) The Backbone of YOLOv8

The backbone of the YOLO algorithm is responsible for feature extraction from the input image, starting with a 640x640x3 input that undergoes convolution operations, reducing the spatial dimensions while increasing the depth of the feature maps enabling the model to capture low-level features like edges and textures. The C2f layers are used in retaining important spatial information through residual connections by efficiently learning deeper representations. In the backbone, the image is continuously processed until the image reaches 20x20x1024 feature maps which indicate that they are deeper and more abstract.

2) The Neck of YOLOv8

In the neck part, the feature maps are aggregated from different parts of the backbone to handle detection at multiple scales. Feature maps from the deeper layers are upscaled using the upsampling operations and then these upscaled feature maps are concatenated with feature maps of earlier layers. This

fusion of feature maps enables the model to detect objects of various sizes. Features are concatenated from deeper layers (e.g. 80×80×256, 40×40×512) to capture finer details enabling robust object detection at varying sizes. C2f blocks are used to enhance feature fusion while preserving information through residual connections. The neck part ensures that the features from various scales are effectively merged for robust object detection across objects of all different dimensions. The neck part includes additional C2f blocks to refine the feature maps and maintain efficiency while balancing computation costs.

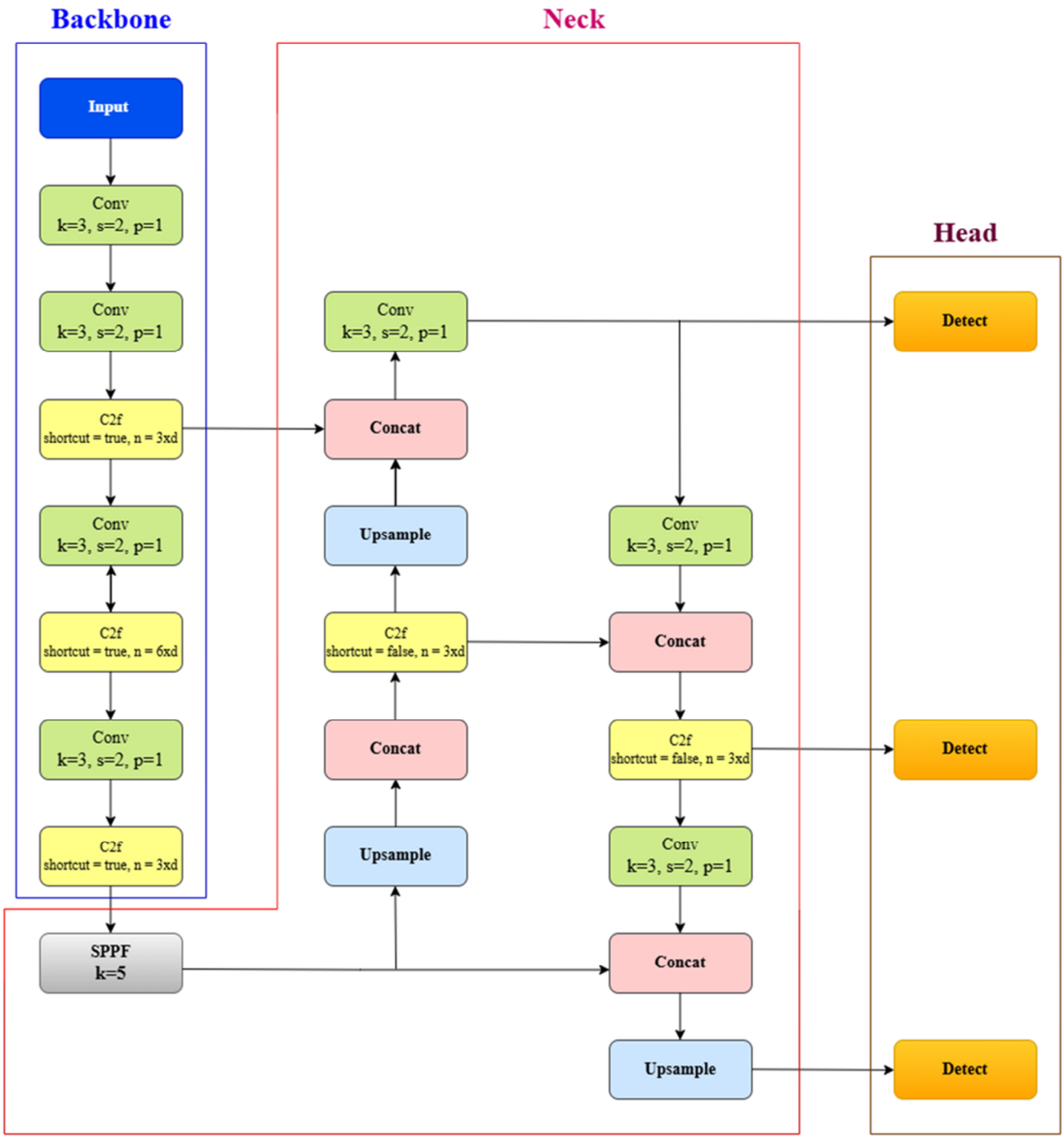

Fig. 1. The Architecture diagram of the YOLOv8 model.

3) The Head of YOLOv8

In the head part, object detection is performed making the predictions. Convolutional layers process the aggregated features before passing them to the detection layers which output the objectness score, bounding boxes, and class probabilities. The three scales of prediction are the 80x80 scale for small objects, the 40x40 scale for medium-sized objects, and the 20x20 scale for large objects. Each detection layer is fine-tuned to detect objects at different resolutions, so the model can accurately identify them regardless of size.

By combining all three processes the YOLO architecture achieves accurate and fast object detection making it suitable for real-time applications.

B. Proposed Methodology

The proposed model is described in the block diagram in Figure 2.

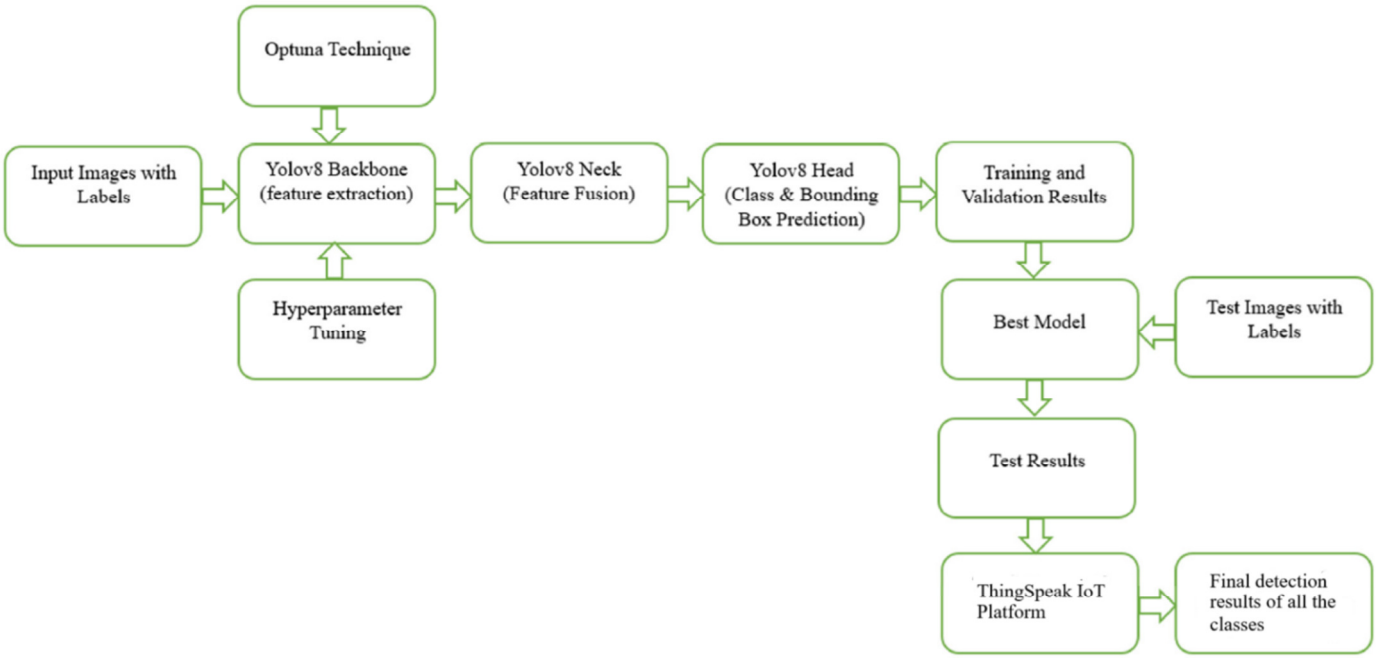

Fig. 2. Block diagram of the proposed model.

The object detection is performed using the YOLOv8 algorithm combined with hyperparameter tuning using the Optuna technique. The labeled annotated images are the first to cross the YOLOv8 backbone architecture for feature extraction. In the backbone, important and relevant features are extracted, which are then processed by the YOLOv8 neck, where feature fusion occurs to combine different levels of feature maps for better contextual understanding. These fused features are passed to the YOLOv8 head, which outputs class predictions and bounding box coordinates for the detected objects. The proposed model performance is assessed through training and validation results by comparing the class and bounding box predictions to the ground truth in the labeled images. Hyperparameters are optimized using Optuna to fine-tune the model, and once the best configuration is identified, the model is tested on a new set of labeled images. The test results evaluate the model's performance and generalization on unseen data. The detection results from the model are sent to the ThingSpeak IoT platform, for real-time monitoring.

C. Dataset and High-Performance Computing (HPC) Configuration

The PASCAL VOC 2012 [30] dataset is a commonly used and recognized benchmark dataset in computer vision applications, particularly for object detection, classification, and recognition tasks. All images in the PASCAL VOC 2012 dataset represent real-world scenarios with different object sizes, variable positions, occlusions, and background complexity variations, making the dataset particularly challenging and ideal for developing robust object detection algorithms. The dataset consists of 16,400 images, of which 13,690 (83.5%) images are used for training, and the rest are equally distributed for validation and testing set. All the images in the dataset are annotated with bounding boxes and class labels for every object of interest, which enables researchers to evaluate models.

The model training is performed using HPC equipped with NVIDIA A100 GPUs of 80 GB RAM and 200 TiB of storage capacity.

D. Data Training Validation Testing and Metrics

In the validation phase, all the important and required model metrics are evaluated on the unused images. In the testing phase, the trained model is checked for its real-world performance by using the developed model on the test dataset and evaluating its performance in various conditions and scenarios. The validation metrics calculated are:

$$Precision = \frac{True\ Positive(TP)}{True\ Positive(TP)+False\ Positive(FP)} \quad (1)$$

$$Recall = \frac{True\ Positives(TP)}{True\ Positives(TP)+False\ Negatives(FN)} \quad (2)$$

$$F1-score = 2 * \frac{Precision}{Recall} \quad (3)$$

$$mAP = \frac{1}{n}\sum_{i=1}^{n} APi \quad (4)$$

The precision measures the proportion of true positive detections relative to all positive detections, while recall represents the proportion of true positive detections relative to all actual positives in the dataset. The F1-score represents the harmonic mean of precision and recall, providing a single metric to evaluate the balance between these two factors.

E. Techniques

This study uses a modified version of the PASCAL VOC 2012 dataset. The original dataset comprises 20 classes, such as airplane, bird, and dining table, which are not relevant for surveillance applications, optimizing the detection process, or enhancing the relevance of object detection to real-time scenarios. Only the most relevant and essential classes are selected, which are bus, car, cat, dog, motorbike, and person. Python 3.10.7 environment is used, and to enable GPU accelerated training in HPC, CUDA 11.8 and cuDNN 11.8 are used. The Optuna framework is utilized to methodically optimize parameters, including learning rate, batch size, momentum, and weight decay. The training is conducted for 150 epochs with an image size of 640. The learning rate was optimized to 0.01 to balance rapid convergence and model stability. The batch size 32 is selected for maximum GPU utilization without causing overfitting and memory overflow. The AdamW optimizer was used to achieve faster convergence, and the momentum was adjusted to 0.937 to reduce oscillations during the training process. The neck of the YOLOv8 architecture was optimized to facilitate multi-scale feature fusion, in order to detect objects of various sizes in dense and dynamic urban environments. Furthermore, various data augmentation techniques were applied to improve generalization, including random scaling, and flipping which helped the model to adapt to diverse lighting conditions. Later, after the detection results are obtained, the ThingSpeak IoT platform is integrated with the YOLOv8 model for real-time data transmission and monitoring.

ThingSpeak, a cloud-based IoT analytics platform, efficiently processes and visualizes IoT data through its RESTful API, which supports safe and secure communication and data exchange using HTTP and HTTPS protocols. The integration process begins with the initialization of the ThingSpeak channel, in which every field is defined to represent every detected object class associated metrics, and bounding box coordinates. The ThingSpeak platform generated unique API Keys and Channel IDs, used to authenticate and secure communication between the YOLOv8 model and the ThingSpeak server. The YOLOv8 model detection results are processed and formatted as a JSON payload, with each consisting of key-value pairs corresponding to the ThingSpeak channel fields. These payloads are transmitted via HTTP POST requests, using Python's requests library in the VS Code, to the ThingSpeak API endpoint. To comply with ThingSpeak's data posting limitations, rate-limiting mechanisms ensure updates occur at 15-second intervals.

III. RESULTS AND DISCUSSION

Figure 3 illustrates the F1-score - confidence curve, highlighting the balance between precision and recall across different confidence thresholds, with a peak F1-score of 0.81 at a confidence threshold of 0.606, indicating the model's effective generalization and strong performance in detecting objects across all classes. Figure 4 illustrates the precision-confidence curve. The precision increases with confidence peaking at 1.0 for all the classes, ensuring high accuracy, but at the cost of decreased recall. Classes like cars, motorbikes, and people maintain lower precision than other classes, such as cats and dogs, while displaying lower precision at low confidence levels, indicating challenges in distinguishing these objects. The fluctuations in the curve at lower confidence are attributed to the inclusion of false positives, which decrease as the threshold increases.

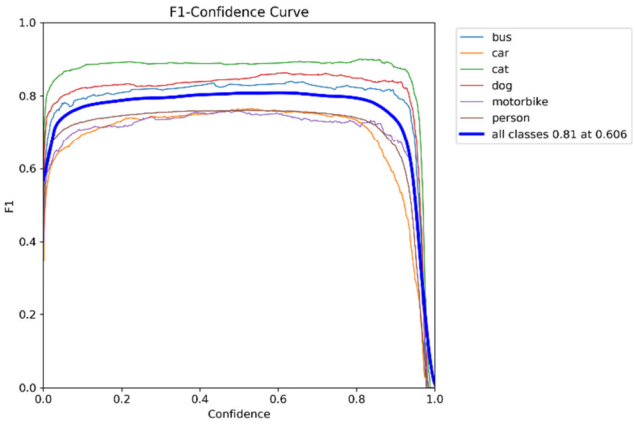

Fig. 3. The confidence curve of the F1-score for the PASCAL VOC-2012 dataset.

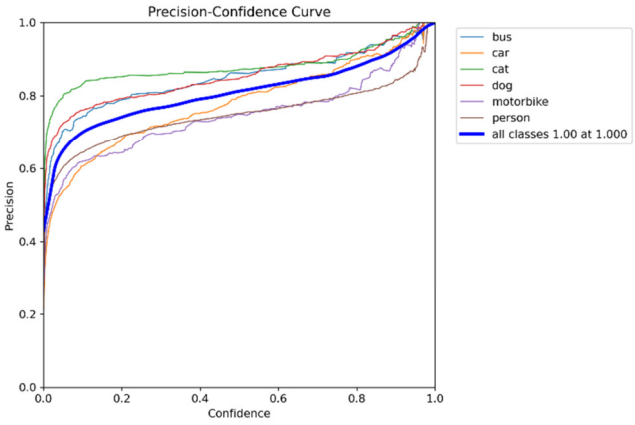

Fig. 4. The confidence curve of precision for the PASCAL VOC-2012 dataset.

Figure 5 presents the recall-confidence curve, showing that recall is highest at low confidence thresholds, with a peak value of 0.91. Based on the graph, as the confidence threshold increases, the recall metric decreases. The cat class consistently maintains higher recall, while categories like cars and motorbikes exhibit sharper declines in recall values.

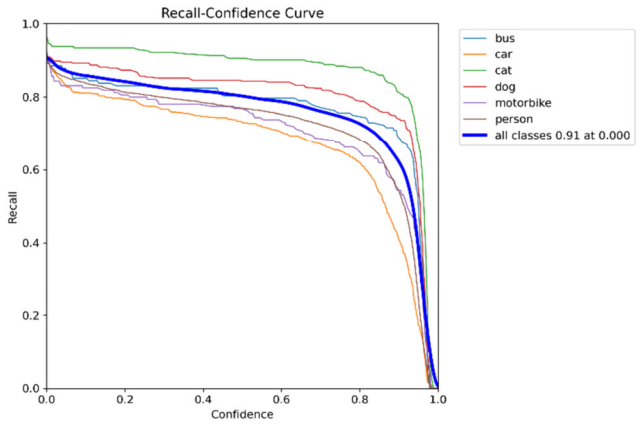

Fig. 5. The confidence curve of recall for the PASCAL VOC-2012 dataset.

Figure 6 depicts the precision-recall curve, demonstrating the model's ability to balance both metrics effectively. The mean Average Precision (mAP) score is derived from this curve, reflecting the model's overall detection accuracy across all classes. The mAP of 0.847 is achieved at an Intersection-over-Union (IoU) at the confidence threshold of 0.5.

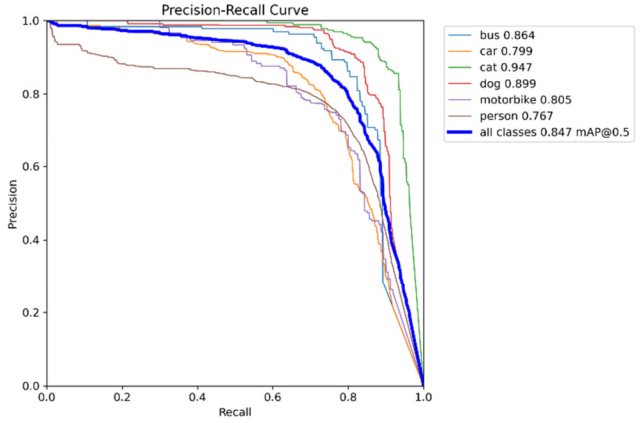

Fig. 6. The precision-recall curve for the PASCAL VOC-2012 dataset.

Figure 7 showcases the confusion matrix, representing the class prediction performance of the YOLOv8m model on the PASCAL VOC 2012 dataset.

Figure 8 showcases, in total, the training and validation losses and all other important performance metrics, reflecting the YOLOv8 model's learning and convergence over training epochs (150). The training losses exhibit a notable sharp drop at the final epochs, which corresponds to the effect of the learning rate scheduler used in YOLOv8, where the learning rate is significantly reduced near the end of the training, allowing the model to converge rapidly with fine-grained parameter updates. The Validation losses (val/box_loss, val/cls_loss, and val/dfl_loss) show a steep decrease during the initial epochs, followed by a slight upward trend after around 50 epochs, indicating potential early signs of overfitting. Despite the overfitting, the performance metrics such as precision, recall, and mAP continue to improve steadily, highlighting the model's robustness and generalization capability. The train/box_loss plot shows a steady decline, before an abrupt decrease at the higher epochs, indicating improved bounding box localization accuracy, while the train/cls_loss and the train/dfl_loss highlight the enhanced object classification and boundary regression for smaller objects. All the validation losses (val/box_loss, val/cls_loss, and val/dfl_loss) also exhibit similar reactions to those training curves with occasional fluctuations that demonstrate the model's robust generalization capabilities. Specifically, the decrease in box loss indicates improved localization accuracy for bounding boxes, while class loss represents better classification accuracy for detected objects. The occasional fluctuations in validation loss are noted as typical when the model is used in unseen data. Moreover, precision, recall, mAP@50, and mAP@50-95 show steady improvements, stabilizing at high levels and indicating balanced detection performance across all classes.

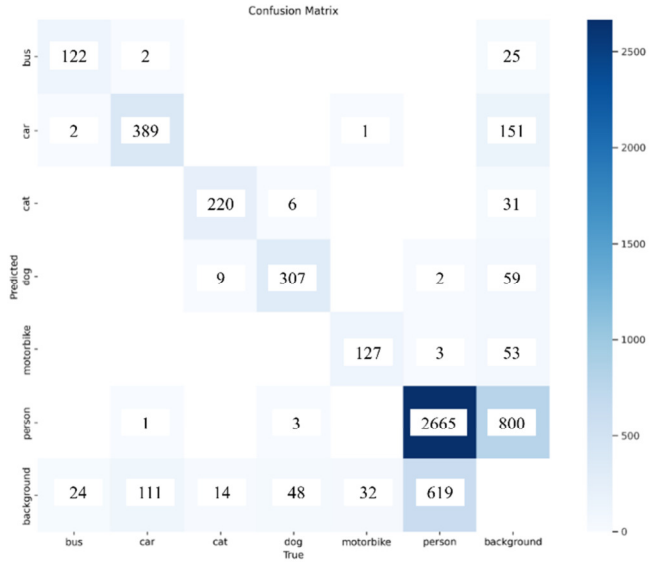

Fig. 7. The confusion matrix for the PASCAL VOC-2012 dataset.

The YOLOv8 framework consists of five variants: nano (n), small (s), medium (m), large (l), and extreme (x), each designed to handle varying dataset complexities and parameter requirements. Investigations are conducted using all YOLOv8 models on the PASCAL VOC 2012 dataset. The results, summarized in Table I, demonstrate that the YOLOv8 medium (m) model outperforms the others, achieving a precision of 0.862, a recall of 0.789, a mAP of 0.847, and an F1-score of 0.81.

TABLE I. TRAINED MODELS METRICS

Technique	Precision	Recall	mAP@50	F1-score	Time
YOLOv8n model	0.820	0.741	0.787	0.76	2.686
YOLOv8s model	0.854	0.767	0.827	0.77	3.292
YOLOv8m model	0.862	0.789	0.847	0.81	4.659
YOLOv8l model	0.837	0.775	0.819	0.79	5.473
YOLOv8x model	0.807	0.727	0.780	0.75	6.508

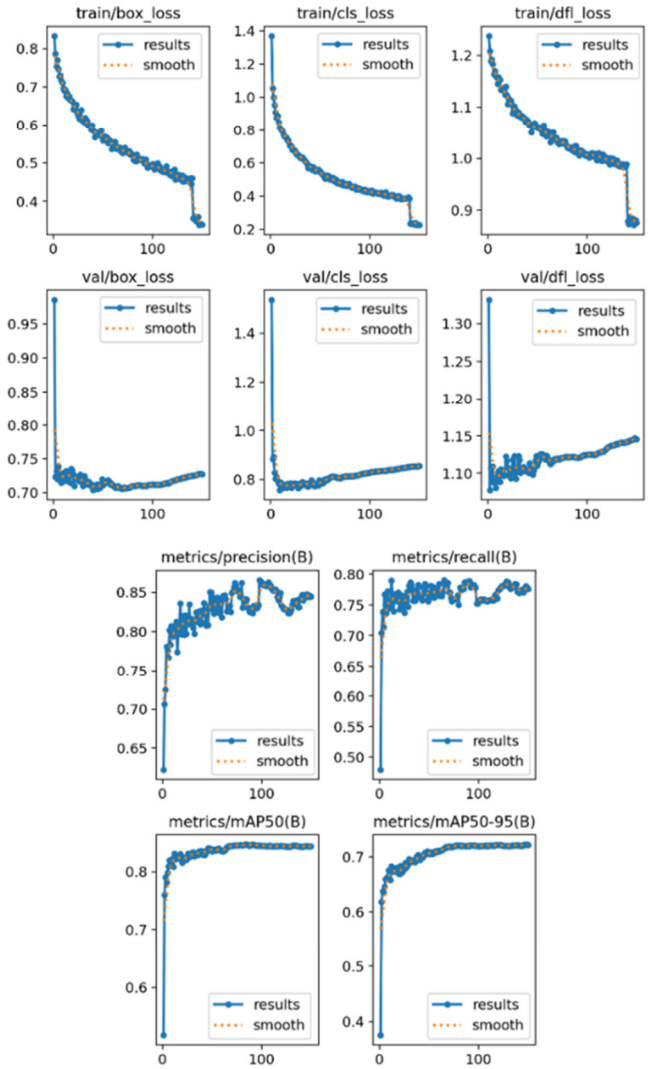

Fig. 8. Performance metrics along with losses.

The YOLOv8m model's training batch results on the PASCAL VOC 2012 dataset are illustrated in Figure 9, which showcases the capability to detect multiple objects across diverse scenarios with accurate localization and classification.

Fig. 9. The PASCAL VOC 2012 dataset training outcomes.

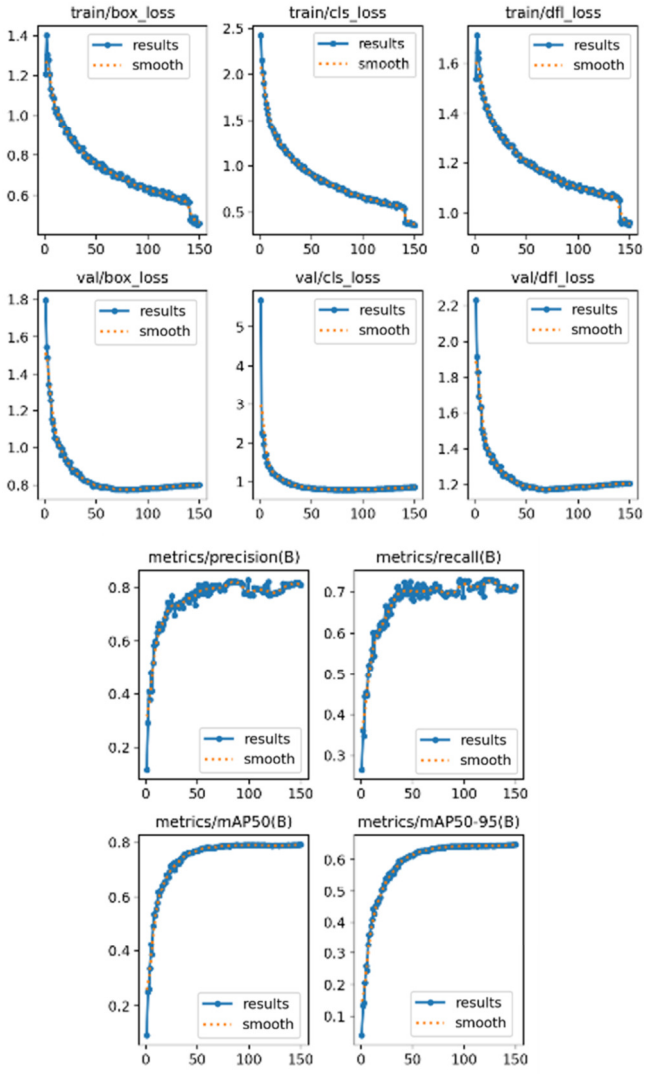

Fig. 10. The metrics chart/graph with 5-fold cross-validation.

In the training batch results, each bounding box is labeled with class names and confidence scores, reflecting the model's

certainty in predictions. The high confidence scores for significant classes like cats, dogs, and buses indicate reliable performance, while slightly lower scores for cars and persons suggest challenges under certain conditions. The training results also showcase that the model adapted to varying object scales, orientations, and occlusions.

To ensure greater robustness and generalizability of the proposed model, a 5-fold cross-validation process was implemented. The results are shown in Figure 10. The cross-validation results include an average mAP@0.5 of 0.792, lower than the originally reported 0.847, demonstrating near-consistent detection capabilities across different splits. Precision and recall values remained stable, with minor decreases. Loss values during training and validation showed smooth convergence across all folds, reaffirming the stability of the model's learning process. The results from the cross-validation process are consistent with the Pascal VOC 2012 results.

Table II compares metrics from prior research studies using various YOLO and R-CNN models on the PASCAL VOC 2012 dataset and highlights the mAP improvements achieved through successive model iterations and optimizations. The traditional models, like Fast R-CNN and Faster R-CNN, demonstrate reliable mAP values of 68.4% and 70.4%, respectively. New models such as SSD and YOLOv3 show improvements, achieving mAP scores of 70.1% and 74.3%, and the YOLOv4-tiny and YOLOv5m models exhibit mAP values of 59.2% and 81.5%, respectively, highlighting the continual advancements in detection accuracy resulting from model modifications. The proposed YOLOv8m model achieves a mAP of 84.7%, exceeding the performance of previous models.

TABLE II. COMPARISON WITH EXISTING LITERATURE

Reference	Techniques	Dataset	mAP Val
[3]	FAST R-CNN	PASCAL VOC-2012	68.4
[3]	FASTER R-CNN	PASCAL VOC-2012	70.4
[3]	SSD	PASCAL VOC-2012	70.1
[10]	YOLOv3	PASCAL VOC-2012	74.3
[25]	YOLOv4-tiny	PASCAL VOC-2012	59.2
[25]	YOLOv5m	PASCAL VOC-2012	81.5
[25]	YOLOv7-Tiny	PASCAL VOC-2012	72.7
Proposed methodology	YOLOv8m with Optuna technique with hyperparameter tuning	PASCAL VOC-2012	84.7

The results of the YOLOv8m model testing batch on the PASCAL VOC 2012 dataset are illustrated in Figure 11, showcasing its performance on new and unseen data and its ability to generalize beyond the training dataset successfully. These test results validate the model's capability for practical use ensuring reliable performance in identifying and classifying objects across different and dynamic environments.

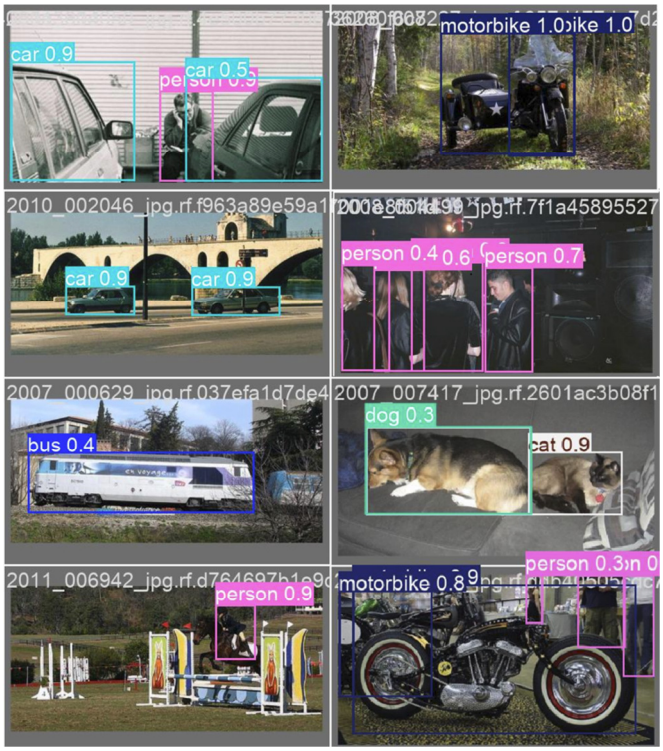

Fig. 11. The PASCAL VOC 2012 dataset test outcomes.

Further evaluations of the proposed model are performed on the KITTI [31] dataset with a 5-fold cross-validation. The KITTI dataset is widely known and recognized for real-world driving scenarios, making it an excellent benchmark for evaluating model performance in challenging situations involving various object categories. A detailed comparison of the metrics achieved on the Pascal VOC 2012 and KITTI datasets is presented in Table III. The results showcase consistent performance, with metrics like precision, recall, and mAP indicating the model's capacity to generalize effectively to new and unseen data.

TABLE III. MODEL TRAINED ON DIFFERENT DATASETS

Datasets	Techniques	Precision	Recall	mAP@50	F1-score
KITTI	Proposed YOLOv8m	0.820	0.741	0.787	0.76
Pascal VOC 2012	Proposed YOLOv8m	0.862	0.789	0.847	0.81

The next step of this study was the integration of the YOLOv8 object detection model with the ThingSpeak IoT platform, enabling the real-time visualization and monitoring of detection results. ThingSpeak's visualization graphs (Figure 12) showcase detected objects from key classes such as buses, cars, cats, dogs, motorbikes, and people.

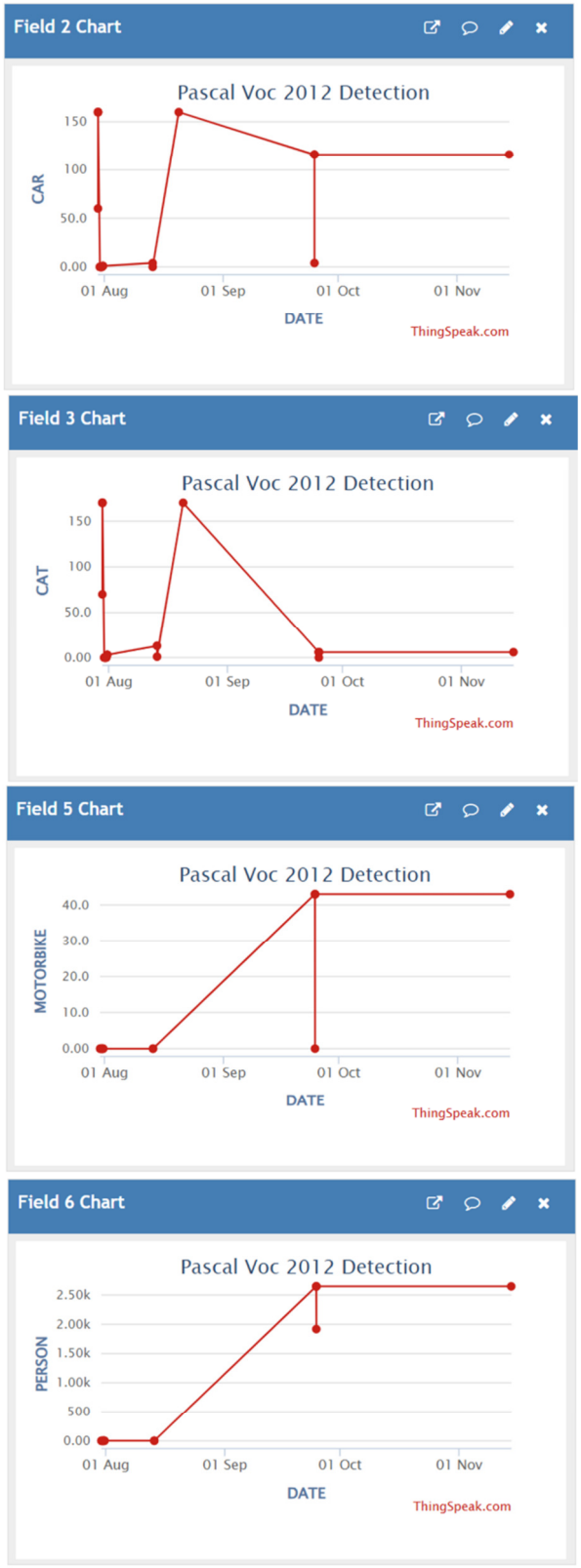

Fig. 12. Real-time object detection in ThingSpeak IoT platform.

The ThingSpeak platform also provides the output results in a tabular summary of detection results, showing each class count, average confidence, and inference time, as illustrated in Figure 13. By integrating the YOLOv8m model with cloud-based IoT infrastructure, the proposed AIoT system enables seamless real-time object detection, transmission, and visualization for monitoring and analysis. Furthermore, it offers a centralized interface for remote access, scalability for multi-site implementations, and improved situational awareness via data visualization. These capabilities, along with high precision, recall, and mAP scores, make the AIoT system an effective instrument for urban planning, traffic control, and security enforcement, regardless of the changes in dynamic environments.

Nevertheless, ThingSpeak IoT platform has several limitations. The limited amount of API requests and the minimum update interval hinder its capacity to handle high-frequency data streams in large-scale applications. Also, the limited data storage capacity per channel presents challenges for applications requiring extensive historical data retention or the management of massive datasets over extended durations. These limitations could mainly be addressed using alternative platforms like AWS IoT Core, Azure IoT Hub, and Google Cloud IoT as they provide scalable solutions for handling large volumes of data and also provide advanced features such as machine learning integration.

Output from last evaluation

```
Detection Results:
------------------------------------------------------------
Class           Count      Avg Confidence  Inference Time (ms)
------------------------------------------------------------
Bus             6          0.95            8.20
Car             116        0.88            8.50
Cat             6          0.90            8.30
Dog             11         0.85            8.10
Motorbike       43         0.92            8.40
Person          2646       0.89            8.30
------------------------------------------------------------
Total Detections 2828
```

Fig. 13. The output detection results table from the ThingSpeak IoT platform.

IV. CONCLUSION

This research presents a methodology that integrates an advanced object detection model YOLOv8 with real-time data transmission over the ThingSpeak Internet of Things (IoT) platform for surveillance applications. The dataset used for object detection training on the YOLOv8 algorithm is PASCAL VOC 2012 which is commonly used for computer vision applications. The proposed YOLOv8 model is optimized through systematic hyperparameter tuning using the Optuna framework which utilizes a systematic strategy to fine-tune the critical hyperparameters like learning rate, batch size, and momentum, rather than using a traditional conventional approach that relies on fixed or manually adjusting hyperparameters. The proposed model achieved a precision of

86.2%, a recall of 78.9%, and a mean Average Precision (mAP) of 84.7%.

The proposed method establishes a new and improved standard for object detection models by achieving better performance compared to traditional models like Faster Regions with Convolutional Neural Networks (R-CNN), Single Shot Detector (SSD) and earlier You Only Look Once (YOLO) versions. These advancements in the model demonstrate the model's robustness and adaptability in various real-world scenarios like crowded and cluttered environments with varying object scales and occlusions. It supports public space monitoring, threat detection, and industrial applications like automated inspection, inventory tracking, and safety monitoring by showcasing its adaptability and scalability for all modern surveillance challenges.

Further research is needed on integrating Artificial Intelligence of Things (AIoT) systems that also explore other datasets with a broader range of objects. Moreover, investigating alternative deep learning models or refining the existing architecture to enhance the system performance and real-time deployment in complex environments is also important for future surveillance application developments.

REFERENCES

[1] Z. Zou, K. Chen, Z. Shi, Y. Guo, and J. Ye, "Object Detection in 20 Years: A Survey," *Proceedings of the IEEE*, vol. 111, no. 3, pp. 257–276, Mar. 2023, https://doi.org/10.1109/JPROC.2023.3238524.

[2] F. M. Talaat and H. ZainEldin, "An improved fire detection approach based on YOLO-v8 for smart cities," *Neural Computing and Applications*, vol. 35, no. 28, pp. 20939–20954, Oct. 2023, https://doi.org/10.1007/s00521-023-08809-1.

[3] A. Gautam and S. Singh, "Deep Learning Based Object Detection Combined with Internet of Things for Remote Surveillance," *Wireless Personal Communications*, vol. 118, no. 4, pp. 2121–2140, Jun. 2021, https://doi.org/10.1007/s11277-021-08071-5.

[4] S. Panigrahi and U. S. N. Raju, "InceptionDepth-wiseYOLOv2: improved implementation of YOLO framework for pedestrian detection," *International Journal of Multimedia Information Retrieval*, vol. 11, no. 3, pp. 409–430, Sep. 2022, https://doi.org/10.1007/s13735-022-00239-4.

[5] S. Bose, M. H. Kolekar, S. Nawale, and D. Khut, "LoLTV: A Low Light Two-Wheeler Violation Dataset With Anomaly Detection Technique," *IEEE Access*, vol. 11, pp. 124951–124961, 2023, https://doi.org/10.1109/ACCESS.2023.3329737.

[6] T. Ahmad, Y. Ma, M. Yahya, B. Ahmad, S. Nazir, and A. U. Haq, "Object Detection through Modified YOLO Neural Network," *Scientific Programming*, vol. 2020, pp. 1–10, Jun. 2020, https://doi.org/10.1155/2020/8403262.

[7] A. A. Alsuwaylimi, R. Alanazi, S. M. Alanazi, S. M. Alenezi, T. Saidani, and R. Ghodhbani, "Improved and Efficient Object Detection Algorithm based on YOLOv5," *Engineering, Technology & Applied Science Research*, vol. 14, no. 3, pp. 14380–14386, Jun. 2024, https://doi.org/10.48084/etasr.7386.

[8] H. Gao, "A Yolo-based Violence Detection Method in IoT Surveillance Systems," *International Journal of Advanced Computer Science and Applications*, vol. 14, no. 8, 2023, https://doi.org/10.14569/IJACSA.2023.0140817.

[9] S.-Y. Jhong, Y.-Y. Chen, C.-H. Hsia, S.-C. Lin, K.-H. Hsu, and C.-F. Lai, "Nighttime object detection system with lightweight deep network for internet of vehicles," *Journal of Real-Time Image Processing*, vol. 18, no. 4, pp. 1141–1155, Aug. 2021, https://doi.org/10.1007/s11554-021-01110-1.

[10] X. Zhou, X. Xu, W. Liang, Z. Zeng, and Z. Yan, "Deep-Learning-Enhanced Multitarget Detection for End–Edge–Cloud Surveillance in Smart IoT," *IEEE Internet of Things Journal*, vol. 8, no. 16, pp. 12588–12596, Aug. 2021, https://doi.org/10.1109/JIOT.2021.3077449.

[11] A. Rehman, T. Saba, M. Z. Khan, R. Damaševičius, and S. A. Bahaj, "Internet-of-Things-Based Suspicious Activity Recognition Using Multimodalities of Computer Vision for Smart City Security," *Security and Communication Networks*, vol. 2022, pp. 1–12, Oct. 2022, https://doi.org/10.1155/2022/8383461.

[12] Z. Wang and Y. Ma, "Detection and recognition of stationary vehicles and seat belts in intelligent Internet of Things traffic management system," *Neural Computing and Applications*, vol. 34, no. 5, pp. 3513–3522, Mar. 2022, https://doi.org/10.1007/s00521-021-05870-6.

[13] C. Li *et al.*, "Fast Forest Fire Detection and Segmentation Application for UAV-Assisted Mobile Edge Computing System," *IEEE Internet of Things Journal*, vol. 11, no. 16, pp. 26690–26699, Aug. 2024, https://doi.org/10.1109/JIOT.2023.3311950.

[14] J. Wu *et al.*, "A Lightweight Small Object Detection Method Based on Multilayer Coordination Federated Intelligence for Coal Mine IoVT," *IEEE Internet of Things Journal*, vol. 11, no. 11, pp. 20072–20087, Jun. 2024, https://doi.org/10.1109/JIOT.2024.3373028.

[15] P. Gao, "Development of YOLO-based Model for Fall Detection in IoT Smart Home Applications," *International Journal of Advanced Computer Science and Applications*, vol. 14, no. 10, 2023, https://doi.org/10.14569/IJACSA.2023.01410117.

[16] Z. Zhi-Xian and F. Zhang, "Image Real-Time Detection Using LSE-Yolo Neural Network in Artificial Intelligence-Based Internet of Things for Smart Cities and Smart Homes," *Wireless Communications and Mobile Computing*, vol. 2022, pp. 1–8, Mar. 2022, https://doi.org/10.1155/2022/2608798.

[17] D. Liu, Z. Wang, and X. Meng, "Fast intensive crowd counting model of Internet of Things based on multi-scale attention mechanism," *IET Image Processing*, vol. 19, no. 1, Jan. 2025, Art. no. e12686, https://doi.org/10.1049/ipr2.12686.

[18] M. Saini, H. Singh, E. Sengupta, A. Aggarwal, H. Singh, and N. Kumar, "An intelligent machine learning-enabled cattle reclining risk mitigation technique using surveillance videos," *Neural Computing and Applications*, vol. 36, no. 4, pp. 2029–2047, Feb. 2024, https://doi.org/10.1007/s00521-023-09143-2.

[19] T. Saidani, "Deep Learning Approach: YOLOv5-based Custom Object Detection," *Engineering, Technology & Applied Science Research*, vol. 13, no. 6, pp. 12158–12163, Dec. 2023, https://doi.org/10.48084/etasr.6397.

[20] S. Dalal *et al.*, "Improving smart home surveillance through YOLO model with transfer learning and quantization for enhanced accuracy and efficiency," *PeerJ Computer Science*, vol. 10, Jun. 2024, Art. no. e1939, https://doi.org/10.7717/peerj-cs.1939.

[21] Y. Zhao, Y. Yin, and G. Gui, "Lightweight Deep Learning Based Intelligent Edge Surveillance Techniques," *IEEE Transactions on Cognitive Communications and Networking*, vol. 6, no. 4, pp. 1146–1154, Dec. 2020, https://doi.org/10.1109/TCCN.2020.2999479.

[22] C. Dong, C. Pang, Z. Li, X. Zeng, and X. Hu, "PG-YOLO: A Novel Lightweight Object Detection Method for Edge Devices in Industrial Internet of Things," *IEEE Access*, vol. 10, pp. 123736–123745, 2022, https://doi.org/10.1109/ACCESS.2022.3223997.

[23] S. Liang *et al.*, "Edge YOLO: Real-Time Intelligent Object Detection System Based on Edge-Cloud Cooperation in Autonomous Vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 12, pp. 25345–25360, Dec. 2022, https://doi.org/10.1109/TITS.2022.3158253.

[24] Z. Xu, J. Li, and M. Zhang, "A Surveillance Video Real-Time Analysis System Based on Edge-Cloud and FL-YOLO Cooperation in Coal Mine," *IEEE Access*, vol. 9, pp. 68482–68497, 2021, https://doi.org/10.1109/ACCESS.2021.3077499.

[25] S. Wang and X. Hao, "YOLO-SK: A lightweight multiscale object detection algorithm," *Heliyon*, vol. 10, no. 2, Jan. 2024, Art. no. e24143, https://doi.org/10.1016/j.heliyon.2024.e24143.

[26] X. Zheng, J. Zou, S. Du, and P. Zhong, "Small Target Detection in Refractive Panorama Surveillance Based on Improved YOLOv8," *Sensors*, vol. 24, no. 3, Jan. 2024, Art. no. 819, https://doi.org/10.3390/s24030819.

[27] Sandhya and A. Kashyap, "Real-time object-removal tampering localization in surveillance videos by employing YOLO-V8," *Journal of Forensic Sciences*, vol. 69, no. 4, pp. 1304–1319, Jul. 2024, https://doi.org/10.1111/1556-4029.15516.

[28] M. Bakirci, "Utilizing YOLOv8 for enhanced traffic monitoring in intelligent transportation systems (ITS) applications," *Digital Signal Processing*, vol. 152, Sep. 2024, Art. no. 104594, https://doi.org/10.1016/j.dsp.2024.104594.

[29] G. Cheng, P. Chao, J. Yang, and H. Ding, "SGST-YOLOv8: An Improved Lightweight YOLOv8 for Real-Time Target Detection for Campus Surveillance," *Applied Sciences*, vol. 14, no. 12, Jun. 2024, Art. no. 5341, https://doi.org/10.3390/app14125341.

[30] Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., and Zisserman, A. (2012). PASCAL Visual Object Classes Challenge 2012 (VOC2012) Dataset. [Online]. Available: http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

[31] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, Sep. 2013, https://doi.org/10.1177/0278364913491297.