IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

CNN-BASED REAL-TIME OBJECT DETECTION FOR SMART FARMING

[1]Kapil Gurunath Dhappadhule, [2]Dr. Sushil V. Kulkarni

[1]M.Tech Student Department of Computer Science & Information Technology,
MBE Society's College of Engineering, Ambajogai, Maharashtra, India

[2]Professor & Head, Department of Computer Science & Information Technology,
MBE Society's College of Engineering, Ambajogai, Maharashtra, India

Abstract:

Recent advances in deep learning—particularly Convolution Neural Networks (CNNs)—have opened transformative opportunities for automated object detection and classification in complex outdoor environments. Models from the YOLO (You Only Look Once) family, especially YOLOv8, have emerged as highly effective tools for real-time multi-class detection, offering an optimal balance between inference speed and detection accuracy. Unlike rule-based or template-matching methods, CNN-based architectures learn intricate visual features directly from data, enabling them to adapt to varying lighting conditions, occlusion from vegetation, and complex natural backgrounds. This paper proposes a complete smart farm surveillance framework that combines YOLOv8-based object detection with automated Telegram Bot API notifications. The system processes live video feeds from farm cameras, identifies threats as they occur, captures annotated snapshots, and delivers instant alerts to farmers' smartphones. Key contributions include: (1) A fine-tuned YOLOv8 detection model trained on a diverse agricultural dataset covering humans, birds, and animals; (2) An integrated Telegram Bot alerting module providing real-time notifications with timestamped annotated images; (3) Deployment on resource-constrained edge devices (Raspberry Pi / NVIDIA Jetson Nano) demonstrating practical feasibility; (4) Comprehensive experimental evaluation under varied farm conditions confirming 94.7% precision and 93.1% recall.

Keywords: CNN, YOLOv8, Smart Farming, Object Detection, Telegram Alerts, Agricultural Surveillance, Real-Time Monitoring, Edge Computing, IoT, Deep Learning

I. INTRODUCTION

The evolution of automated agricultural surveillance systems has been shaped by advances in deep learning, computer vision, and IoT integration. A comprehensive review of relevant prior work is presented below, highlighting the critical need for real-time, accurate, and accessible farm monitoring solutions.

II. LITERATURE REVIEW

Redmon and Farhadi introduced YOLOv3 with the Darknet-53 backbone and multi-scale predictions, making it one of the earliest real-time detectors applicable to farm surveillance. However, performance degrades for small, fast-moving objects such as birds at distance. Bochkovskiy et al. proposed YOLOv4, incorporating CSPDarknet53, Mosaic data augmentation, PANet path aggregation, and CIOU loss, achieving superior accuracy-speed trade-offs at the cost of high computational resource requirements. The Ultralytics team further advanced the field with YOLOv8, introducing anchor-free detection, improved modularity, and native support for edge deployment—making it the most suitable candidate for real-time smart farm applications.

In the agricultural IoT domain, Rahul and Sharma demonstrated an AI-and-IoT intrusion detection system for farmland monitoring in real time, though scalability remained limited for large farm areas. Singh and Kumar proposed a smart farming surveillance system combining IoT cameras, environmental sensors, and AI algorithms, highlighting continuous automated monitoring but noting high network dependency and maintenance costs as barriers to rural adoption.

The literature reveals a consistent research gap: existing systems either achieve high accuracy but lack real-time responsiveness, or operate in real-time but without robust multi-class detection or instant alerting. The proposed YOLOv8-plus-Telegram framework directly addresses this gap.

Table 1: Comparative Study of Existing Approaches

Ref.	Author(s)	Approach	Methods Used	Limitations
[1]	Redmon & Farhadi (2018)	YOLOv3 Detection	Darknet-53, logistic classifiers	Struggles with small objects
[2]	Bochkovskiy et al. (2020)	YOLOv4 Detection	CSPDarknet53, data augmentation	High computational demand
[3]	Jocher et al. (2023)	YOLOv8 Framework	Ultralytics, Python API	Mostly documentation
[4]	Rahul & Sharma (2022)	Intrusion Detection	DL + IoT sensors	Limited scalability
[5]	Singh & Kumar (2021)	Smart Farm IoT	IoT + AI monitoring	Network dependency

III. EXISTING SYSTEM

Current smart farm surveillance solutions typically follow a four-stage pipeline: (1) Data Collection via IP cameras, CCTV, or IoT motion sensors; (2) Object Detection using pre-trained CNN models such as YOLOv3, SSD, or Faster R-CNN hosted on cloud servers; (3) Threat Inference using confidence thresholds and predefined classification rules; and (4) Alert Generation through SMS, email, or dedicated mobile applications.

While these systems have contributed meaningfully to agricultural automation, they exhibit critical limitations. Cloud dependency requires high-speed internet connectivity unavailable in many rural farming areas. High hardware costs restrict adoption by small-scale farmers. Detection accuracy degrades under outdoor conditions caused by dust, shadows, rain, and low light, generating false alarms and missed detections. Additionally, existing systems rely on complex mobile applications that pose usability barriers for farmers with limited digital literacy, ultimately limiting their real-world impact.

IV. PROPOSED METHODOLOGY

A. System Architecture

The proposed system is built as a modular, end-to-end surveillance pipeline comprising six interconnected components: (1) Camera Module: Live video streams captured via USB, IP, or CCTV cameras positioned at strategic farm locations; (2) Frame Processing Unit: Frames preprocessed using OpenCV with resizing to 640x640, pixel normalization, and RGB conversion; (3) YOLOv8 Detection Engine: Core detection module performing real-time multi-class object localization and classification with bounding box overlays; (4) Threat Classification Module: Applies confidence-threshold rules to classify detected objects as humans, animals (cow, dog, boar, monkey), or birds; (5) Telegram Alert System: Transmits annotated frame snapshots with detection metadata (type, confidence, timestamp, farm zone) to the farmer's registered smartphone; (6) Data Logging Module: Archives detected threat frames and metadata locally or on Firebase for audit trails and security analytics.

B. Workflow of the System

The operational workflow follows a sequential, continuous monitoring loop ensuring 24×7 autonomous surveillance. Step 1 involves camera initialization: activate farm camera and begin live frame capture. Step 2 involves frame preprocessing: resize, normalize, and convert frames for YOLOv8 input

compatibility. Step 3 involves YOLOv8 inference: detect objects and output bounding boxes, class labels, and confidence scores. Step 4 involves threat analysis: match detected classes against predefined threat categories with confidence thresholds. Step 5 involves decision logic: if no threat is detected, continue scanning; if a threat is detected, trigger the alert mechanism. Step 6 involves Telegram notification: send annotated image with object type, timestamp, and farm zone identifier to farmer's Telegram account. Step 7 involves continuous monitoring loop: repeat steps 1–6 indefinitely for uninterrupted surveillance.

C. Algorithmic Flow

The detection algorithm operates as follows: (1) Initialize YOLOv8 model and farm camera; (2) Capture and preprocess each video frame; (3) Run YOLOv8 inference to extract class labels and confidence values; (4) For each detected object, if label belongs to the threat set {human, cow, dog, boar, bird, monkey} AND confidence exceeds the defined threshold, classify as a Threat; (5) If a threat is detected, save the annotated frame and transmit a Telegram alert containing the image, label, confidence score, timestamp, and location identifier; (6) Continue to the next frame in a continuous processing loop.

D. Advantages of the Proposed System

The proposed system offers multiple significant advantages. Real-Time Processing: YOLOv8 achieves approximately 30 FPS on GPU hardware, ensuring immediate threat detection and response. High Detection Accuracy: CNN-based multi-class detection achieves 94.7% precision with low false alarm rate across diverse farm conditions. Instant Telegram Alerts: Farmers receive annotated snapshots within 1-2 seconds of intrusion detection, enabling prompt intervention. Edge Deployment Capability: System operates on Raspberry Pi and NVIDIA Jetson Nano, eliminating cloud dependency and internet requirements. 24×7 Autonomous Operation: Fully automated continuous surveillance without manual intervention, reducing labour requirements significantly. Cost-Effective and Scalable: Built with open-source tools and affordable hardware, making it accessible for small and medium-scale farmers.

V. IMPLEMENTATION

A. Development Environment

Hardware components: Raspberry Pi 4 / NVIDIA Jetson Nano (edge inference), HD/IP CCTV Camera (live video feed), and optional GPU-enabled server for model training. Software stack: Python 3.10, PyTorch, Ultralytics YOLOv8 framework, OpenCV for video processing and frame preprocessing, Telegram Bot API via Python requests library, Firebase for optional cloud storage, NumPy and Pandas for data management, and GitHub for version control.

B. Dataset and Model Training

The YOLOv8 model was fine-tuned on a custom agricultural surveillance dataset comprising 8,500 annotated images spanning six detection classes: Human, Cow, Dog, Bird, Boar, and Monkey. Images were collected from diverse farm environments under varying lighting conditions, seasonal changes, camera angles, and weather scenarios. Data augmentation techniques including horizontal flipping, mosaic composition, brightness adjustment (±30%), random cropping, and rotation were applied to improve model robustness. Training was performed for 120 epochs using the AdamW optimizer with an initial learning rate of 0.001 and cosine decay scheduling, achieving convergence with stable training and validation loss curves indicating strong generalization.

C. Module-Wise Implementation

The Camera Interface Module captures live frames using OpenCV at a fixed rate of 30 FPS with brightness and contrast normalization applied to handle outdoor illumination variability. The Detection Module loads the fine-tuned YOLOv8 model onto the target hardware and processes each frame in real time, generating annotated bounding boxes with class labels and confidence scores. The Alert Module, implemented using the Telegram Bot API through Python's requests library, captures annotated frames upon threat detection, generates timestamped alert messages specifying the intruder type, confidence score, and farm zone, and delivers them to the farmer's registered Telegram account within 1-2 seconds. The Storage Module archives all detection events including frame images and metadata in structured local directories, with optional Firebase integration for long-term security analytics and audit trail maintenance.

VI. RESULTS AND DISCUSSION

A. Detection Performance

The proposed YOLOv8-based system was evaluated on a test set of 1,200 images across six object classes under diverse farm conditions including daylight, low-light, cloudy weather, and partial occlusion scenarios. Table 2 compares the detection performance of the proposed model against established baseline architectures.

Table 2: Performance Comparison of Detection Models

Model	Precision (%)	Recall (%)	F1-Score (%)
YOLOv3	82.4	79.1	80.7
YOLOv5	88.6	85.3	86.9
Faster R-CNN	91.2	88.7	89.9
YOLOv8 (Proposed)	94.7	93.1	93.9

The proposed YOLOv8 model achieves the highest precision (94.7%), recall (93.1%), and F1-Score (93.9%) among all compared models while maintaining near real-time inference at 30 FPS on GPU-enabled hardware. Faster R-CNN demonstrates superior precision in cluttered environments but operates at significantly slower inference speeds (~11 FPS), making it unsuitable for real-time farm deployment. Training and validation accuracy curves showed steady improvement and converged by epoch 90, with training and validation loss declining smoothly throughout, indicating effective learning without overfitting. The final model accuracy stabilized at approximately 95%, reflecting strong performance in multi-class object recognition under real-world farm conditions.

B. Alert Efficiency

Intrusion snapshots were successfully transmitted to the farmer's Telegram account within 1-2 seconds of threat detection, representing a significant improvement over traditional CCTV systems that require manual checking. Each alert contains the detected object type, confidence score, timestamped annotated image, and farm zone identifier. This near-instantaneous notification enables prompt intervention even when the farmer is physically distant from the farm, substantially reducing potential crop damage and livestock harm.

C. Edge Deployment Performance

Deployment on Raspberry Pi 4 achieved 6-8 FPS using the lightweight YOLOv8n model variant—sufficient for practical farm surveillance with minor responsiveness reduction. NVIDIA Jetson Nano achieved 18-22 FPS, providing an excellent balance between performance and cost for rural deployment. GPU server deployment maintained full 30 FPS performance with simultaneous multi-camera stream support, confirming system scalability for large agricultural operations. These results validate the framework's adaptability across both resource-constrained and high-performance hardware configurations.

D. Limitations

False detections occasionally occurred under challenging conditions including swaying vegetation in strong wind, cast shadows misclassified as animals, and very small birds at significant distance from the camera. Continuous real-time monitoring on Raspberry Pi introduces computational overhead that limits sustained operation without dedicated power management. Future work will address these issues through model compression techniques including pruning and quantization, integration of complementary motion sensors for false alarm reduction, and incorporation of thermal cameras for improved night-time detection.

VII. CONCLUSION

This paper presented a CNN-based real-time object detection and intelligent alerting framework for smart farm surveillance. By combining the state-of-the-art YOLOv8 deep learning model with the Telegram Bot API communication layer, the system provides farmers with automated, instant, and accurate intrusion alerts for humans, animals, and birds across 24 hours a day, 7 days a week. Experimental results confirm that the framework achieves 94.7% precision and 93.1% recall with near

real-time inference at 30 FPS, significantly outperforming existing baseline methods in both accuracy and response speed.

The system's modular architecture supports practical deployment on affordable edge devices, making it directly accessible to small and medium-scale farmers without requiring cloud infrastructure, high-speed internet connectivity, or specialized technical expertise. By eliminating dependency on continuous human supervision and enabling proactive, data-driven threat response, the proposed framework represents a meaningful and practical contribution toward intelligent, sustainable, and efficient agricultural practices in India and beyond.

VIII. FUTURE SCOPE

Future work will explore: Night Vision and Thermal Camera Integration: Extending surveillance to low-light and nighttime conditions using infrared and thermal imaging hardware. Drone-Based Aerial Monitoring: Equipping autonomous drones with lightweight YOLO models for large-area farm coverage impractical with fixed cameras. Automated Response Actions: Integration with actuators such as automated alarms, strobe lights, and irrigation sprinklers for active intrusion deterrence. Multi-Camera Scalability: Extending the framework to support simultaneous multi-camera streams with a unified monitoring dashboard. Cloud Analytics Dashboard: Developing a long-term intrusion pattern analysis portal providing actionable farm security insights. WhatsApp and SMS Integration: Expanding alert delivery to WhatsApp and SMS for farmers without smartphone data access. Model Compression: Applying pruning, quantization, and knowledge distillation to further optimize inference speed on resource-constrained edge devices. Crop Disease and Pest Detection Fusion: Extending detection categories to include crop diseases and pest infestations for a comprehensive smart farming assistant.

REFERENCES

[1] J. Redmon and A. Farhadi, 'YOLOv3: An Incremental Improvement,' arXiv preprint arXiv:1804.02767, 2018.

[2] A. Bochkovskiy, C. Y. Wang, and H. Y. M. Liao, 'YOLOv4: Optimal Speed and Accuracy of Object Detection,' arXiv:2004.10934, 2020.

[3] G. Jocher, A. Chaurasia, J. Qiu, and A. Stoken, 'YOLOv8 Ultralytics Documentation,' 2023. [Online]. Available: https://docs.ultralytics.com

[4] A. Rahul and P. Sharma, 'Real-Time Intrusion Detection in Agricultural Fields Using Deep Learning and IoT Devices,' International Journal of Computer Applications, vol. 184, no. 20, pp. 10–16, 2022.

[5] R. Singh and S. Kumar, 'Smart Farming Using IoT and AI: A Real-Time Surveillance System,' Journal of Agricultural Informatics, vol. 12, no. 3, pp. 45–52, 2021.

[6] A. Sharma and N. Jain, 'Automated Animal Detection System for Farm Monitoring Using CNNs,' Procedia Computer Science, vol. 167, pp. 2420–2429, 2020.

[7] V. Kumar and D. Patel, 'Edge-Based Object Detection Using YOLO and Raspberry Pi for Agricultural Monitoring,' IEEE Access, vol. 10, pp. 65129–65138, 2022.

[8] I. Krishnateja et al., 'Agricultural Pest Detection Using Convolutional Neural Networks,' International Journal on Advanced Computer Engineering and Communication Technology, 2025.

[9] OpenCV Documentation. [Online]. Available: https://docs.opencv.org/

[10] Telegram Bot API Documentation. [Online]. Available: https://core.telegram.org/bots/api

[11] S. Ren, K. He, R. Girshick, and J. Sun, 'Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,' Advances in Neural Information Processing Systems, 2015.

[12] A. Howard et al., 'MobileNets: Efficient CNNs for Mobile Vision Applications,' arXiv:1704.04861, 2017.