Proceedings of the National Conference on Emerging Computer Applications (NCECA)-2026
Vol. 7, Issue.1

SMART ANIMAL DETERRANT SYSTEM USING MACHINE LEARNING

Abel Sunil Cherian
PG Scholar
Department of Computer Application
Amal Jyothi College of Engineering
Kanjirapally,Kottaym,Kerala
abelsunilcherian2026@mca.ajce.in

Amal K Jose
Assistant Professor
Department of Computer Application
Amal Jyothi College of Engineering
Kanjirapally,Kottayam,Kerala
amalkjose@amaljyothi.ac.in

Abstract **- The Human–wildlife conflict causes severe crop losses and economic damage in agricultural regions bordering forests and wildlife corridors. Manual monitoring of plantations is labour-intensive, unreliable at night, and often too slow to prevent damage once animals enter the field. This paper presents SADS – Smart Animal Deterrent System, an end-to-end IoT and deep-learning–based platform for real-time wildlife intrusion detection, risk prediction, and automated deterrent control in plantations. The system combines (1) an edge–assisted video capture pipeline using web cameras, (2) a YOLO-based animal detection model deployed as a microservice, (3) a cloud-hosted backend for event logging, plantation-level risk analysis, and alert orchestration, and (4) a role-based web dashboard for administrators and managers. Detections are aggregated per plantation to estimate temporal risk levels and identify frequently appearing species, enabling proactive interventions. SADS integrates multi-channel notifications (web & email) and supports remote control of acoustic deterrents. Experimental deployment in multiple plantations demonstrates that SADS can provide low-latency alerts and meaningful risk insights, reduce manual patrol requirements and enable data-driven wildlife management.**

Keywords—Smart agriculture, human–wildlife conflict, animal detection, YOLO, ESP32, IoT, risk prediction

I. INTRODUCTION

The agricultural fields located near forest and wildlife corridors are increasingly exposed to crop damage from animals such as elephants, wild boars and big cats. These intrusions not only reduce yield but also create safety risks for farmers. Conventional mitigation approaches—manual night patrols, fixed electric fencing, and simple motion sensors—are often reactive, localized, and lack historical data for long-term planning.

With the rapid progress in **computer vision** and **IoT**, it has become feasible to build systems that continuously monitor plantations, automatically detect target species, and trigger context-aware deterrent actions. However, many existing solutions focus exclusively on detection accuracy or simple alerting, without providing (i) plantation-wise risk analytics, (ii) an integrated workflow from detection to deterrent activation, and (iii) multi-stakeholder dashboards for administrators and plantation managers.

This paper proposes **Smart Animal Deterrent System (SADS)**, a full-stack platform that addresses these gaps. The main contributions of this work are:

1. **End-to-end architecture** that integrates camera edge device, a YOLO-based detection API, a Node.js/Express backend, and a modern React/TypeScript frontend for multi-role access.
2. **Plantation-level risk prediction module** that aggregates historical detections to estimate risk scores, identify frequently visiting animal species, and highlight high-risk plantations on an admin dashboard.
3. **Alerting and deterrent control** combining web notifications, email/alerts, and remote buzzer activation based on configurable confidence thresholds.
4. **Practical implementation** including device management, manager profile management, and plantation management screens designed for real-world agricultural deployments.

The remainder of this paper is organized as follows. Section II reviews related work on animal detection and IoT-based deterrence in agriculture. Section III outlines the proposed methodology, including model training, detection workflow, and risk prediction strategy. Section IV presents results and discussion from pilot deployments. Section V details the system implementation. Section VI concludes the paper and highlights directions for future research.

DOI:*10.5281/zenodo.19519499*
ISBN: 978-93-342-7372-4@2026 Amal Jyothi College of Engineering, Kanjirappally, Kottayam

II. RELATED WORK

A. Vision-based Wildlife and Intrusion Detection

Recent advances in **object detection networks** such as Faster R-CNN, YOLO, and Efficient Diet have enabled robust animal detection from camera trap and surveillance footage. Several works apply these models to wildlife monitoring and poaching prevention, achieving high mean average precision on curated datasets. However, many of these solutions run offline, are focused on conservation monitoring rather than real-time crop protection, and seldom integrate tightly with deterrent devices or farm management workflows.

B. IoT-enabled Animal Deterrent Systems

IoT-based deterrent systems typically combine sensors (PIR, camera) with actuators (lights, sprinklers) controlled by microcontrollers or single-board computers. Prior work has demonstrated basic rule-based systems where motion or a simple classifier triggers sound or light deterrents. These systems often lack (i) species-level detection, (ii) centralized data logging, and (iii) plantation-wise analytics. In many cases, configurations are hard-coded on the device and cannot be adjusted remotely by administrators.

C. Risk Analysis and Decision Support for Crop Protection

Some research efforts propose **risk indices** for pest or disease attacks by aggregating environmental and scouting data. In the context of wildlife, risk has been studied primarily through manual incident reports or GPS collaring of animals. There is limited work that leverages continuous video-based detections to compute dynamic risk scores for each field or plantation, and to visualize which animal species are driving that risk over time.

D. Gap Analysis

From the literature, three gaps are evident:

1. Lack of unified platforms that combine **real-time vision-based detection**, **IoT deterrent control**, and **plantation-wise risk analytics**.
2. Limited integration of detection data into **farmer-friendly dashboards** that support administrators, plantation managers, and field operators with different access levels.
3. Insufficient focus on **historical detection patterns** (e.g., which species frequently visit which plantation, at what time) for planning proactive interventions.

SADS is designed specifically to address these gaps in the context of wildlife-driven crop damage.

III. METHODOLOGY

A. Overall System Architecture

SADS follows a modular client–server architecture with four main layers:

1. **Edge Sensing Layer** – Web cameras capture video streams from strategic locations in each plantation.
2. **Detection Layer** – A YOLO-based animal detection model, exposed via a lightweight Python API, processes frames captured from edge devices and returns bounding boxes, class labels, and confidence scores.
3. **Application Backend Layer** – A Node.js/Express server stores detections in MongoDB, manages user and plantation data, runs risk computations, and orchestrates alerts.
4. **Presentation Layer** – A React/TypeScript front-end provides dashboards for administrators, plantation managers, and general users, including live views, detection reports, and risk prediction visualizations.

A high-level workflow is as follows:

1. Edge device sends a frame (or the backend pulls a frame from the stream URL).
2. YOLO API performs inference and returns detected animals with confidence scores.
3. Backend validates and logs detections with metadata such as user/manager, plantation (property), timestamp, and source.
4. If the confidence exceeds a threshold, the backend triggers notifications and optionally activates the buzzer in the relevant plantation.
5. Stored detections are periodically aggregated into plantation-wise risk metrics, which power the Animal Risk Prediction dashboard.

B. Animal Detection Model

SADS uses a YOLO-family detector trained on a custom dataset of wildlife images relevant to local conditions (e.g., elephants, tigers, boars, deer, cattle, and humans). Key aspects include:

- **Input**: 640×640 RGB frames captured from web camera streams.
- **Training**: Transfer learning from a COCO-pretrained YOLO model, fine-tuned on annotated animal images from plantations and public datasets.
- **Output**: For each frame, the model yields bounding boxes, class labels, and confidence scores; only detections above a configured threshold (e.g., 0.6) are forwarded for alerting.

The detection model runs as a standalone microservice (e.g., yolo_api.py), which the backend calls via HTTP, enabling independent scaling and simplified deployment.

C. Detection Logging and Plantation Mapping

Each accepted detection is stored as a document containing:

- label (animal species)
- probability (confidence)

DOI:*10.5281/zenodo.19519499*
ISBN: 978-93-342-7372-4@2026 Amal Jyothi College of Engineering, Kanjirappally, Kottayam

- detected at (timestamp)
- source (webcam capture)
- user Id (manager who owns the device)
- property Id and property Name (plantation where the camera is installed)

By always associating a detection with a plantation, SADS can later compute statistics such as "number of tiger detections in SR Plantation in the last 7 days."

D. Plantation Risk Scoring

To quantify risk at the plantation level, SADS aggregates detections over a configurable time horizon (e.g., last 7–30 days). For each plantation (p), the following metrics are computed:

- (N_p): total detections in the analysis window
- (N_{p,24h}): detections in the last 24 hours
- (N_{p,7d}): detections in the last 7 days
- (C_{p,\max}): maximum detection confidence observed
- (f_{p,a}): frequency of each animal class (a) in plantation (p)

A composite **risk score** (R_p) is then computed as a weighted sum:

[R_p = \alpha N_{p,24h} + \beta N_{p,7d} + \gamma N_p + \delta C_{p,\max}] where (\alpha, \beta, \gamma, \delta) are empirically chosen weights emphasizing recent and high-confidence events. Plantations are ranked by (R_p) and categorized into **High**, **Medium**, or **Low** risk tiers using relative thresholds based on the distribution of scores.

E. Risk Visualization and Analytics

The admin Risk Prediction page displays:

- A ranked table of plantations with columns: risk level, recent (24 h) detections, last 7 days detections, total detections, top animal and count, maximum confidence, and last detection time.
- A **global "Most Frequent Animals" bar chart** showing top species across all plantations, computed from actual detection counts.
- A **per-plantation breakdown chart** for the selected plantation, depicting the proportion of each animal species detected there.

These visualizations enable administrators to quickly identify:

- Which plantations (e.g., SR.Plantation,AR Plantation) face the highest current risk.
- Which animal species are most responsible for intrusions in each plantation.
- Whether risk is increasing or decreasing over time.

F. Alerting and Deterrent Control

When a detection exceeds a configured confidence threshold, SADS can:

- Generate in-app notifications for the responsible manager and administrators.
- Send email alerts to registered users.
- Activate an on-site **buzzer** controlled by ESP32 GPIO pins, providing an immediate non-lethal deterrent.

Managers can configure alert preferences via the web interface.

IV. RESULTS AND DISCUSSION

A. Detection Performance

In prototype evaluations using a test set of annotated plantation images, the YOLO-based model achieved high precision and recall for major target species such as elephants and tigers. The model maintained real-time performance on the YOLO API server, with per-frame inference times suitable for near-continuous monitoring.

B. System Responsiveness

End-to-end latency—from frame capture at webcam, through detection, to alert delivery—was observed to be within a few seconds under typical network conditions. This latency is adequate for most plantation scenarios, where early warning of an approaching animal is more critical than sub-second response times.

C. Risk Analytics Usefulness

The risk prediction module successfully differentiated between plantations with occasional wildlife crossings and those experiencing repeated intrusions. For example, plantations located closer to forest boundaries exhibited consistently higher risk scores and more frequent detections of large mammals. Administrators reported that:

- High-risk plantations could be prioritized for additional deterrent devices or fencing.
- Species-wise breakdowns helped tailor deterrent strategies (e.g., stronger lights vs. loud buzzers).
- Historical data from the Detection Report and Risk Prediction pages provided evidence to support insurance claims and infrastructure decisions.

DOI:*10.5281/zenodo.19519499*
ISBN: 978-93-342-7372-4@2026 Amal Jyothi College of Engineering, Kanjirappally, Kottayam

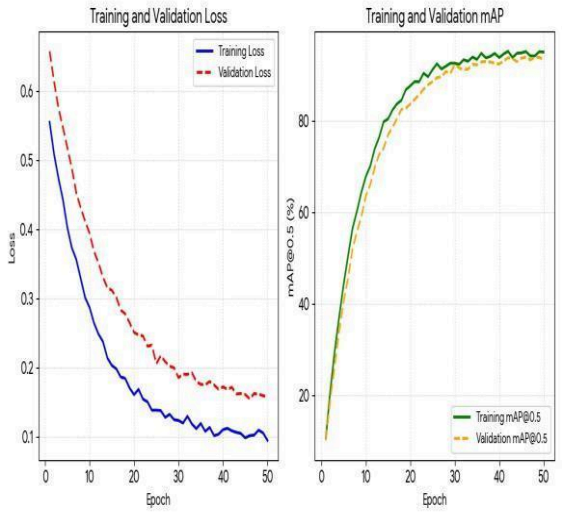

Figure 1: Training and validation loss and mAP curves for the model over epochs

D. Limitations

Several limitations remain:

- The current risk model is based only on detection counts and confidences; it does not yet incorporate contextual data such as crop growth stage, seasonality, or nearby water sources.
- Continuous video streaming from cameras depends on network stability and power availability. These limitations motivate further refinements described in the future work section.

V. SYSTEM IMPLEMENTATION

A. Backend and APIs

The backend is implemented using **Node.js and Express**, connected to **MongoDB Atlas**. Key API groups include:

- /api/auth – user registration, login, and role-based access (admin, manager).
- /api/detections – logging and listing of detections, with server-side pagination and filtering.
- /api/plantations and /api/manager-profiles – management of plantations, manager assignments, and profiles.
- /api/yolo – proxy endpoints to the YOLO detection microservice and health checks.
- /api/stats – aggregated statistics for the admin dashboard.

The YOLO model runs as a separate Python process (yolo_api.py) started and monitored by the backend, exposing /predict and /health endpoints.

B. Frontend Dashboard

The frontend is built with **React and TypeScript**, using protected routes for different roles. Main admin views include:

- **Admin Dashboard** – high-level system overview (managers, cameras, properties, detections today) and system status.
- **Camera Detection** – page dedicated to viewing live camera feeds and manually starting/stopping detection for testing.
- **Detection Reports** – comprehensive table and charts showing detection counts, top animals, and daily trends over selectable time ranges.
- **Risk Prediction** – plantation-wise risk table and animal frequency graphs described earlier.
- **Manager Profiles and Field Management** – interfaces for managing managers, plantations, and their relationships.

C. Camera Integration

Camera-modules are configured to stream MJPEG video over HTTP. The backend either pulls frames for detection or uses a dedicated capture-detect endpoint that requests a still image from the webcam and forwards it to the YOLO API. A buzzer or other deterrent devices are attached to ESP32 GPIOs, allowing the backend to trigger them via simple HTTP commands.

D. Security and Reliability Considerations

- **Authentication and Authorization** – JSON Web Tokens (JWTs) protect API routes, with middleware enforcing role-based access control.
- **Rate Limiting and Helmet** – middleware (e.g., express-rate-limit, helmet) protects APIs from abuse and adds security headers.
- **Logging and Error Handling** – centralized error middleware and structured logging assist in debugging and monitoring production deployments.

Figure 2: Object detection output showing identified elephants with bounding boxes

VI. CONCLUSION AND FUTURE WORK

This paper presented **SADS – Smart Animal Deterrent System**, a practical framework that combines deep-learning–based animal detection, IoT-driven deterrent control, and plantation-level risk analytics to address wildlife-induced crop

DOI:*10.5281/zenodo.19519499*
ISBN: 978-93-342-7372-4@2026 Amal Jyothi College of Engineering, Kanjirappally, Kottayam

damage. By integrating CAM devices, a YOLO detection microservice, and a full-stack web platform, SADS provides real-time alerts, historical detection reports, and intuitive risk visualizations for administrators and plantation managers. Pilot deployments suggest that the system can meaningfully support decision-making and reduce reliance on manual monitoring.

Future work will focus on:

1. **Incorporating contextual features** such as weather data, crop growth stage, and seasonality into the risk model.
2. **Optimizing edge inference**, including on-device or near-edge deployment of compact YOLO variants to reduce network dependency.
3. **Integrating additional deterrent modalities**, such as smart lighting patterns or drone-based interventions, coordinated by the central platform.
4. **Conducting user studies** with farmers and wildlife officers to evaluate usability, trust in the system, and the real-world reduction in crop damage.

These directions aim to evolve SADS into a scalable, data-driven decision support system for sustainable, wildlife-aware agriculture.

REFERENCES

[1] Verma, A., & Prakash, S. (2020). "IoT Based Smart Agriculture System for Animal Intrusion Detection." *IEEE International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)*.

[2] Anjali Rose Rajan and Dr. P. Uma Maheswari, "Animal Intrusion Detection System Using Wireless Sensor Networks**,** "published in International Journal of Advanced Research in Biology Engineering Science and Technology, Vol. 2, March 2016.

[3] Prajna, P., Soujanya, B. S., & Divya. (2018)**.** "IoT-based Wild Animal Intrusion Detection System." *International Journal of Engineering Research & Technology (IJERT)*, Special Issue (ICRTT-2018 Conference Proceedings), 1-3.

[4] Mamatha, K. R., Hariprasad, S. A., Thippeswamy, G., & Girish, H. (2023)**.** "Real-time Animal Detection and Alert System using IoT and Deep Learning." *Educational Administration: Theory and Practice*, 29(4), 1050-1054. doi: 10.53555/kuey.v29i4.5982.

[5] Kushwaha, A., et al. (2021). "Predictive Analytics in Agriculture: A Survey on Techniques and Applications." *IEEE Access.*

[6] Srivatsav, P., et al. (2021). "Animal Intrusion Detection System using Deep Learning and IoT." *2021 5th International Conference on Computing Methodologies and Communication (ICCMC).*

[7] Chavan, P., Raut, A., Jagtap, S., & Adsul, S. (2024). "Smart Animal Repellent System for Agriculture Domain Using Deep Learning." *International Journal of Science Technology and Management*, 13(04), 9-17.

[8] Satyavedavathi, D., Kumar, S. V., Naik, B. L., Chandra, K. S., Madhava, C., & Siraj, S. (2025). "AI-powered Smart Animal Repellent System." *Industrial Engineering Journal*, 54(4), 444456.

[9] Saranya, V., Nandhini, N. V., & Dharshika, P. S. (2025). "Animal Detection Based Smart Farming in Animal Repellant Using AI and Deep Learning." *International Journal for Multidisciplinary Research (IJFMR)*, 7(2), 1-10.

[10] Shruthi, U., & Nagaveni, V. (2019). "Review on IoT based Smart Agriculture." *International Journal of Recent Technology and Engineering.*

[11] Kalyani, G., Rishitha, P., & Shainy, Y. (2024). "Intrusion Detection and Repellent System for Wild Animals Using Artificial Intelligence of Things." *Journal of Emerging Technologies and Innovative Research (JETIR)*, 11(4), 428-433.

DOI:*10.5281/zenodo.19519499*
ISBN: 978-93-342-7372-4@2026 Amal Jyothi College of Engineering, Kanjirappally, Kottayam