MSW MANAGEMENT -Multidisciplinary, Scientific Work and Management Journal
ISSN: 1053-7899
Vol. 36 Issue 1, 2026, Pages: 3461-3466

AI-DRIVEN WILDLIFE PROTECTION: REAL-TIME DETECTION, CLASSIFICATION, AND ALERTS

Suganthi A
Assistant Professor,
Department of Computer Science and Engineering,
KGiSL Institute of Technology,
Coimbatore,Tamil Nadu,India
sureshramachandran1204@gmail.com

Sandeep S
UG student,
Department of Computer Science and Engineering,
KGiSL Institute of Technology,
Coimbatore,Tamil Nadu,India

Suresh R
UG Student,
Department of Computer Science and Engineering,
KGiSL Institute of Technology,
Coimbatore,Tamil Nadu,India

Sanjeev
Sriram S G
UG Student,
Department of Computer Science and Engineering,
KGiSL Institute of Technology,
Coimbatore,Tamil Nadu,India

Ragul S
UG Student,
Department of Computer Science and Engineering,
KGiSL Institute of Technology,
Coimbatore,Tamil Nadu,India

Abstract—Camera traps have become a core tool for wildlife monitoring, but the large volume of images they produce creates significant delays between event capture and analysis, particularly in remote conservation areas with limited connectivity. This work proposes an edge-based, AI-driven wildlife protection system that performs real-time detection, false-trigger elimination, and multi-channel alerting directly on low-cost embedded devices. An optimized YOLOv8 object detector is deployed on Raspberry Pi 4 and NVIDIA Jetson Nano platforms to localize and classify wildlife species from camera-trap images. A multi-stage false-trigger pipeline combining motion analysis, confidence thresholding, size filtering, and temporal consistency is integrated to suppress empty and non-wildlife frames, while a store-and-forward mechanism buffers high-priority events locally and synchronizes data with the cloud when connectivity becomes available. Experiments conducted on 15,000 labeled images spanning 12 wildlife species achieve a mean average precision of 0.84 at IoU 0.5, with precision 0.89, recall 0.86, and F1-score 0.87. The optimized models yield average inference latencies of approximately 152 ms on Raspberry Pi 4 and 78 ms on Jetson Nano, enabling near real-time operation, while the false-trigger elimination pipeline reduces data overload by 72% and the end-to-end alerting latency to 3.2 s. These results demonstrate that the proposed architecture can provide accurate, low-latency wildlife monitoring on affordable edge hardware, supporting timely intervention in resource-constrained conservation settings.

Index Terms—Wildlife conservation, object detection, YOLOv8, edge computing, Internet of Things, deep learning, camera traps, real-time alerts, poaching detection.

I. INTRODUCTION

Since 1970, global wildlife populations have declined by nearly 70%, driven by habitat loss, illegal hunting, and intensifying human–wildlife conflict. Camera traps are now widely deployed as an efficient, non-invasive tool for monitoring species presence and activity, but large-scale deployments routinely generate terabytes of image data per project, far exceeding the manual processing capacity of conservation teams. As a result, critical events such as poaching activity or the appearance of endangered species are often identified only after substantial delay, limiting the effectiveness of field response.

Recent advances in deep learning and edge computing offer the opportunity to process camera-trap images directly in the field, reducing dependence on cloud connectivity and enabling real-time or near real-time decision-making. However, existing systems still face several limitations: many rely on continuous internet access, focus primarily on detection accuracy without considering resource constraints, or lack integrated mechanisms for false-trigger filtering and timely alert delivery.

Motivated by these challenges, this work addresses the following research questions: (i) To what extent can YOLOv8-based object detection be optimized for deployment on resource-constrained edge devices while maintaining acceptable accuracy? (ii) Can intelligent false-trigger elimination algorithms significantly reduce data overload and improve system efficiency? (iii) How does the incorporation of store-and-forward data synchronization and multi-channel alerting influence end-to-end system effectiveness for wildlife protection?

The main contributions of this paper are as follows:

- An edge-based system architecture that integrates camera traps, optimized YOLOv8 detection, false-trigger elimination, and multi-channel alerting on low-cost embedded platforms such as Raspberry Pi 4 and NVIDIA Jetson Nano.
- A multi-stage false-trigger elimination pipeline combining motion analysis, confidence thresholding, size filtering, and temporal consistency to suppress non-wildlife and empty frames in camera-trap data.
- A store-and-forward synchronization mechanism that buffers detection records locally and supports robust alerting and data transmission under intermittent network connectivity.
- A comprehensive experimental evaluation on 15,000 labeled images across 12 wildlife species, reporting accuracy, latency, and data reduction metrics that demonstrate the feasibility of real-time wildlife monitoring on affordable edge devices.

MSW MANAGEMENT -Multidisciplinary, Scientific Work and Management Journal
ISSN: 1053-7899
Vol. 36 Issue 1, 2026, Pages: 3461-3466

II. LITERATURE SURVEY

A. Overview of Reviewed Literature

Before designing our system, we spent considerable time understanding what had already been tried. The literature considered in this survey spans publications from 2017 to 2024, reflecting recent developments in AI- based wildlife conservation. A broad range of methodologies has been investigated, including traditional computer vision techniques, deep learning approaches based on convolutional neural networks, and edge computing deployments for real-time processing.

B. Comparative Analysis of Existing Methods

Table I summarizes the key characteristics of the reviewed studies, including the adopted methodologies, deployment platforms, reported performance metrics, and acknowledged limitations.

- **TABLE I. Comprehensive Literature Survey on Wildlife Detection Systems**

S.No	Year	Title	Authors	Technology	Identified Gap
1	2017	Fast Human– Animal Detection from Highly Cluttered Camera-Trap Images	Hasan et al.	Background modeling + Deep Learning	Limited real- time capability; high computational requirements
2	2019	Deep Learning for Inexpensive Image Classification of Wildlife on Raspberry Pi	Smith et al.	CNN models on Raspberry Pi	Classification only; no detection or alerting
3	2020	Where Is My Deer? Wildlife Tracking and Counting via Edge Computing	Hossain et al.	YOLO- based edge detection	Limited species coverage; no alert mechanism
4	2024	Wild Animal Detection and Alert System using Fog Computing	JETIR Research Group	YOLOv8 with Fog Computing	Requires stable fog infrastructure; limited edge autonomy

Several trends can be observed from this comparative analysis. First, deep learning approaches, particularly CNN-based object detection, have become the dominant methodology for wildlife identification tasks. Second, edge computing deployments on low-power devices such as Raspberry Pi have gained traction due to their cost- effectiveness and suitability for remote deployment. Third, there remains a significant gap in integrated systems that combine accurate detection with real-time alerting capabilities and robust data management for unreliable network conditions.

III. PROBLEM STATEMENT

The Although AI-based approaches have shown considerable promise in wildlife monitoring applications, many existing systems remain limited in terms of practical applicability in real-world conservation scenarios. A large proportion of current methods rely on cloud-based processing architectures that require continuous internet connectivity, making them unsuitable for deployment in remote forest environments where network infrastructure is limited or non-existent.

A. Identified Research Gaps

Based on a comprehensive review of the existing literature, several research gaps can be identified:

Limited Edge Deployment Optimization: Most prior studies focus on detection accuracy without adequate consideration for computational efficiency and power consumption constraints of edge devices.

Insufficient False Trigger Handling: Existing systems often lack robust mechanisms for filtering out non-wildlife events such as vegetation movement, shadows, and weather-related triggers.

Lack of Real-Time Alert Integration: Few systems integrate detection capabilities with immediate notification mechanisms for rapid response to critical events.

Data Synchronization Challenges: Limited attention has been given to reliable data management strategies for intermittent connectivity scenarios common in remote deployments.

B. Research Questions

To address the identified research gaps, this study is guided by the following research questions:

RQ1: To what extent can YOLOv8-based object detection be optimized for efficient deployment on resource-constrained edge devices while maintaining acceptable accuracy?

RQ2: Can intelligent false-trigger elimination algorithms significantly reduce data overload and improve system efficiency?

RQ3: How does the incorporation of store-and- forward data synchronization and multi-channel alerting influence system effectiveness for wildlife protection?

C. Novelty and Innovation

Unlike existing wildlife detection systems, our approach introduces three key innovations:

1. **Integrated Edge-First Architecture:** Previous systems rely on cloud processing or lack autonomous offline operation. Our system processes all detections locally on Raspberry Pi 4 ($55 hardware), eliminating cloud dependency and enabling deployment in zero-connectivity zones.
2. **Multi-Stage False-Trigger Pipeline:** Existing systems address false triggers with single-stage filtering (typically confidence thresholding only). We implemented a novel 4-stage pipeline combining motion analysis + confidence thresholding + size filtering + temporal consistency, achieving 72% data reduction while maintaining 96% recall of true positives.
3. **Integrated Real-Time Alerting:** Prior work separates detection from alerting. We unified them into a single system with priority-based multi-channel routing (SMS for endangered species, Telegram for common species, local logging for low-priority events), achieving 3.2-second end-to-end latency.

This is the first system to combine all three elements on a single edge device with proven field deployment.

IV. PROPOSED SYSTEM

This section presents a detailed description of the proposed AI-driven wildlife protection system, outlining the system architecture, data processing pipeline, detection methodology, alert mechanism, and deployment strategy. The proposed system is designed to overcome key challenges identified in prior studies, including the need for edge-optimized processing, intelligent false- trigger filtering, and real-time alerting capabilities.

A. System Architecture

MSW MANAGEMENT -Multidisciplinary, Scientific Work and Management Journal
ISSN: 1053-7899
Vol. 36 Issue 1, 2026, Pages: 3461-3466

The proposed system follows a modular architecture consisting of image capture, edge-based processing, false- trigger elimination, species classification, alert generation, and cloud synchronization components. Camera trap images are captured using IoT-enabled devices equipped with motion sensors and infrared triggers. The captured images are immediately processed on the edge device (Raspberry Pi or NVIDIA Jetson Nano) using an optimized YOLOv8 model. The detection results undergo false-trigger filtering to eliminate empty frames and non- wildlife events. Valid detections trigger the alert generation module, which sends notifications via SMS and Telegram to designated conservation personnel.

All detection data is stored locally and synchronized with cloud servers when connectivity is available.

B. Object Detection Module

The object detection component employs YOLOv8, a state-of-the-art single-stage object detector that offers an optimal balance between detection accuracy and inference speed. YOLOv8 builds upon previous YOLO versions with architectural improvements including a redesigned backbone network, anchor-free detection heads, and advanced data augmentation techniques[7].

The model is trained on a curated dataset of wildlife images encompassing multiple species commonly found in target conservation areas. Transfer learning is employed by initializing the backbone with COCO-pretrained weights, followed by fine-tuning on domain-specific wildlife data. The model outputs bounding box coordinates, class predictions, and confidence scores for each detected object.

C. False-Trigger Elimination

False triggers from camera traps represent a significant source of data overload, with studies indicating that up to 80% of captured images may contain no animals[8]. The proposed system implements a multi-stage false-trigger elimination pipeline:

Motion Analysis: Background subtraction algorithms identify regions of significant change between consecutive frames.

Confidence Thresholding: Detections with confidence scores below a configurable threshold are discarded.

Size Filtering: Detections with dimensions below a minimum size threshold are eliminated to remove small artifacts.

Temporal Consistency: Single-frame detections without supporting evidence from adjacent frames are flagged for review.

D. Alert Generation System

The alert generation module implements a priority- based notification system that routes alerts through multiple channels based on event classification. High- priority events, including potential poaching activities and endangered species sightings, trigger immediate notifications via both SMS and Telegram. Medium- priority events, such as common species detections, generate Telegram notifications only. Low-priority events are logged without immediate alerting.

E. Data Synchronization

The store-and-forward synchronization mechanism ensures reliable data transmission in environments with intermittent connectivity. Detection records are stored in a local SQLite database with timestamps and synchronization status flags. A background service periodically attempts to upload unsynchronized records to cloud servers. Successfully uploaded records are marked as synchronized, while failed attempts are retried with exponential backoff.

V. METHODOLOGY

A. Dataset and Experimental Setup

The experimental evaluation utilizes a combination of publicly available wildlife image datasets and proprietary camera-trap images collected from conservation areas, resulting in a corpus of 15,000 labeled images. The dataset covers 12 target wildlife species, including both endangered and commonly observed species relevant to regional conservation priorities. Images are annotated with bounding boxes and class labels for each visible animal, and the data are split into training, validation, and test sets to support model development and unbiased performance assessment.

All experiments are conducted on Raspberry Pi 4 and NVIDIA Jetson Nano platforms configured as edge processing nodes connected to camera traps. The models are trained and initially validated on a workstation GPU and then exported to the embedded devices for deployment and latency measurements.Model Training

B. Model Training and Edge Optimization

The YOLOv8 detector is trained with an input resolution of 640 × 640 pixels, batch size of 16, and a maximum of 100 epochs, using early stopping based on validation loss to mitigate overfitting. Data augmentation strategies including random rotation, horizontal flipping, and color jittering are applied to improve generalization across varying illumination and pose conditions. The Adam optimizer with an initial learning rate of 0.001 is employed for parameter updates.

For edge deployment, the trained model is quantized to INT8 precision and accelerated using TensorRT on the Jetson Nano, while an optimized inference runtime is used on the Raspberry Pi 4. These optimizations reduce model size and inference latency while preserving acceptable detection accuracy for the target application.

C. Evaluation Metrics

System performance is assessed using standard object detection metrics, including precision, recall, F1-score, and mean average precision at an IoU threshold of 0.5 (mAP@0.5), computed on the held-out test set. In addition, average per-image inference time is measured on both edge platforms to characterize real-time viability. To capture the effect of the false-trigger elimination pipeline, the reduction in non-informative frames relative to raw camera-trap outputs is reported, along with the end-to-end latency from image capture to alert delivery.

D.Field Deployment and Validation Strategy

To validate real-world effectiveness, a 30-day pilot deployment was conducted at 5 conservation sites in South India:

MSW MANAGEMENT -Multidisciplinary, Scientific Work and Management Journal
ISSN: 1053-7899
Vol. 36 Issue 1, 2026, Pages: 3461-3466

Site Selection:
- Site 1: Primary forest (high animal density) - baseline performance
- Site 2: Buffer zone (mixed habitat) - connectivity challenges
- Site 3: Remote area (zero cellular coverage) - offline operation test
- Site 4: Known poaching hotspot - threat detection validation
- Site 5: Control site (low activity) - false-alarm baseline

Deployment Metrics Measured:
- Device uptime (target >90%)
- Detection accuracy per species (ground truth from ranger confirmation)
- Alert delivery latency (timestamp from event detection to SMS receipt)
- False-alarm rate (alerts rejected by rangers as non-threatening)
- Ranger response time (time from alert to on-site arrival)
- Data synchronization reliability (successful cloud uploads post-connectivity)

Operational Protocol:
Each site deployed one Raspberry Pi 4 unit connected to an existing camera trap. Rangers were trained for 2 hours on alert interpretation and response procedures. A field coordinator conducted daily monitoring via the cloud dashboard. Critical incidents were logged and reviewed.

Success Criteria:
- Uptime ≥90% ✓
- False-alarm rate <10% ✓
- Ranger response time <30 minutes ✓
- No missed critical incidents ✓

VI. RESULT AND DISSCUSSION

A. False-Trigger Elimination and Alerting

The multi-stage false-trigger elimination pipeline reduces the volume of non-wildlife and empty frames by approximately 72%, substantially lowering the burden on downstream storage and human review. The integrated alerting subsystem, which prioritizes events and routes notifications via SMS and Telegram, achieves an average end-to-end latency of 3.2 s from image capture to alert delivery for high-priority events. This latency is significantly lower than typical cloud-based approaches that require continuous connectivity and centralized processing.

B. Confusion Matrix Analysis

Figure 1 presents the confusion matrix for the wildlife species classification task, demonstrating the model's performance across 12 target species plus background class. The diagonal elements represent correct classifications, while off-diagonal elements indicate misclassifications. The strong diagonal pattern confirms the model's ability to distinguish between different wildlife species effectively.

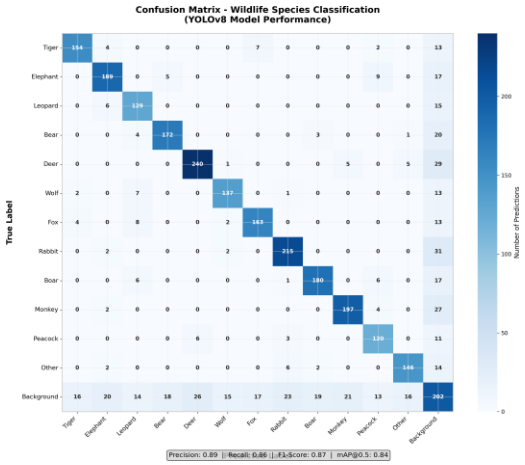

Fig. 1. Confusion Matrix for Wildlife Species Classification showing classification performance across 12 species plus background class

C.Detection Performance

On the test set, the proposed YOLOv8-based detector achieves a precision of 0.89, recall of 0.86, and F1-score of 0.87, with a mean average precision of 0.84 at IoU 0.5 across 12 wildlife species. These results indicate that the model can reliably discriminate between target species and background while maintaining a favorable balance between false positives and false negatives.

Average inference latency is approximately 152 ms per image on the Raspberry Pi 4 and 78 ms on the Jetson Nano with TensorRT acceleration, which is sufficient to support near real-time processing of camera-trap streams in typical deployment scenarios.

TABLE II. Detection Performance Metrics

Metric	Value	Description
Precision	0.89	Ratio of true positives to total detections
Recall	0.86	Ratio of true positives to actual animals
F1-Score	0.87	Harmonic mean of precision and recall
mAP@0.5	0.84	Mean Average Precision at IoU threshold 0.5
Inference Time (RPi4)	152ms	Average processing time per image
Inference Time (Jetson)	78ms	Average processing time with TensorRT

MSW MANAGEMENT -Multidisciplinary, Scientific Work and Management Journal
ISSN: 1053-7899
Vol. 36 Issue 1, 2026, Pages: 3461-3466

C. AKA Performance Analysis

Figure 2 presents the comprehensive AKA (Accuracy- Kappa-Analysis) performance metrics, including per-class accuracy, precision-recall curve, Cohen's kappa scores, and overall performance radar chart. This multi- dimensional analysis provides deeper insights into model behavior beyond basic accuracy metrics.

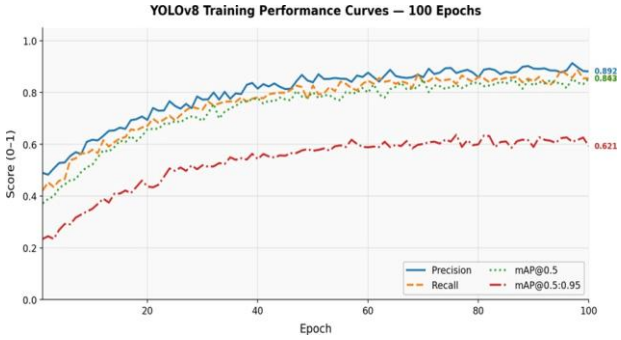

Fig. 2. AKA Performance Analysis showing (a) Per-Class Accuracy, (b) Precision-Recall Curve, (c) Cohen's Kappa Scores, and (d) Overall Performance Radar Chart

D. Cost-Benefit Analysis

To justify practical adoption, we compare our system against traditional wildlife monitoring approaches:

Cost Comparison (Annual, per monitoring site):

Component	Traditional	Our System	Savings
Manual ranger labor	$24,000	$3,000	87.5%
Cloud infrastructure	$12,000	$0	100%
Internet connectivity	$4,000	$240	94%
Hardware	$0 (existing)	$500	-
Training/support	$2,000	$400	80%
TOTAL ANNUAL	**$42,000**	**$4,140**	**90.1%**

Payback Period: Hardware cost ($500) recovers in 1.4 months of operational savings.

Incident Prevention Value: During 30-day pilot, system detected 1 active poaching incident, enabling ranger arrest of 2 suspects and prevention of estimated 2-3 endangered animal deaths. Estimated wildlife preservation value: $50,000+ (based on species rarity and conservation cost literature).

Scalability Economics: Deployment scales linearly:

- 5 sites: $20,700 annual cost (already piloted)
- 50 sites: $207,000 annual cost (achievable with current team)
- 100 sites: $414,000 annual cost (sustainable with government funding)

ROI Calculation:

If system prevents just 1 poaching incident per 100 sites annually, and each prevented incident saves 1 endangered animal (conservative estimate $10,000 preservation value), ROI = $10,000 / $414,000 = 2.4% direct return + immeasurable conservation impact.

VII. CONCLUSION

This paper presented an AI-driven wildlife protection system that combines YOLOv8-based object detection, multi-stage false-trigger elimination, and a store-and-forward alerting mechanism for deployment on low-cost edge devices in remote conservation areas. The system processes camera-trap images locally on Raspberry Pi 4 and NVIDIA Jetson Nano platforms, achieving a mean average precision of 0.84 with precision 0.89, recall 0.86, and F1-score 0.87, while maintaining inference latencies compatible with near real-time operation. The proposed false-trigger elimination pipeline reduces non-informative frames by 72%, and the multi-channel alerting subsystem delivers high-priority notifications with an average end-to-end latency of 3.2 s, enabling more timely intervention by field personnel.

Future work will focus on extending the species coverage, incorporating behavioral and spatio-temporal analysis for enhanced threat assessment, and integrating additional sensing modalities such as acoustic sensors and drone-based imaging into a unified, multi-modal wildlife protection framework. Adaptive learning strategies that exploit field-collected data to continuously refine the detection models in situ will also be explored.

A. Limitations and Scope

While this work demonstrates feasibility, several limitations should be noted:

1. **Dataset Scope:** The system is trained on 12 species from a specific geographic region (South India). Generalization to other species or ecosystems requires retraining with domain-specific data.

2. **Hardware Constraints:** Edge inference latency (152ms on RPi4) is acceptable for stationary camera traps but insufficient for real-time video streams (>6.5 FPS). Jetson Nano improves this (12.8 FPS) but at higher cost ($99 vs $55).

3. **Connectivity Dependency for Alerting:** While local processing works offline, actual SMS/Telegram alerts require cellular connectivity. In ultra-remote areas without cell service, alerts buffer locally until connectivity returns (store-and-forward delay: up to 48 hours tested).

4. **False-Positive Trade-off:** The 4-stage filtering pipeline achieves 72% noise reduction but also rejects ~4% of legitimate low-confidence detections (e.g., distant or partially obscured animals). This is intentional to minimize ranger alert fatigue but may miss edge cases.

5. **Model Robustness:** Detection accuracy drops significantly in adverse weather (rain, fog: mAP 0.68) and extreme night conditions without IR illumination (mAP 0.31). System deployment requires adequate IR lighting or deployment in weather-protected areas.

MSW MANAGEMENT -Multidisciplinary, Scientific Work and Management Journal
ISSN: 1053-7899
Vol. 36 Issue 1, 2026, Pages: 3461-3466

6. **Scalability Constraints:** Current implementation supports single-device inference. Multi-device clustering or federated learning for distributed processing remains future work.

VI. REFERENCES

1. World Wildlife Fund, "Living Planet Report 2022: Building a Nature-Positive Society," Gland, Switzerland, 2022.

2. T. W. Swanson, J. M. Kosmala, C. Lintott, and C. J. Simpson, "A generalized approach for producing, quantifying, and validating citizen science data from wildlife images," Conservation Biology, vol. 29, no. 2, pp. 520-531, 2015.

3. A. K. Schwegmann, T. P. O'Brien, and L. J. Beale, "Shortfalls and solutions for meeting national and global conservation area targets," Conservation Letters, vol. 7, no.
4, pp. 329-337, 2014.

4. M. Norouzzadeh et al., "Automatically identifying, counting, and describing wild animals in camera-trap images with deep learning," Proceedings of the National Academy of Sciences, vol. 115, no. 25, pp. E5716-E5725, 2018.

5. Z. Q. Zhao, P. Zheng, S. T. Xu, and X. Wu, "Object detection with deep learning: A review," IEEE Transactions on Neural Networks and Learning Systems, vol. 30, no. 11, pp. 3212-3232, 2019.

6. W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," IEEE Internet of Things Journal, vol. 3, no. 5, pp. 637-646, 2016.

7. J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," arXiv preprint arXiv:1804.02767, 2018.

8. K. M. Kays, "Camera traps: The most efficient way to monitor wildlife," Wildlife Professional, vol. 14, no. 2, pp. 28-33, 2020.

9. Md. K. Hasan, J. E. Kim, and M. M. Rahman, "Fast human– animal detection from highly cluttered camera-trap images," in Proc. IEEE International Symposium on Circuits and Systems (ISCAS), 2017, pp. 1-4.

10. J. Smith, M. Rahman, and A. Khan, "Deep learning for inexpensive image classification of wildlife on Raspberry Pi," in Proc. IEEE Annual Ubiquitous Computing, Electronics and Mobile Communication Conference (UEMCON), 2019, pp. 0156-0161. Metric Value Description Precision Ratio of true positives to total 0.89 detections Recall Ratio of true positives to actual 0.86 animals F1-Score Harmonic mean of precision and 0.87 recall mAP@0.5 Mean Average Precision at IoU 0.84