IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

# AI-DRIVEN WILDLIFE MONITORING AND POACHING PREVENTION

*Using Computer Vision and Deep Learning: A YOLOv8-Based Real-Time Detection System with Flask Web Application*

1**Shreya Narayankar**, 2**Shambhavi Panchwagh**, 3**Vedant Hendre**, 4**Vedarth Patil**

1,2,3,4*Student, Department of Computer Engineering*

*RMD Sinhgad School of Engineering, Warje, Pune, Maharashtra, India*

***Abstract:*** Illegal poaching continues to be one of the gravest threats to global wildlife populations, pushing numerous endangered species toward irreversible extinction. Conventional anti-poaching methods — ranger patrols, static camera traps, and manual video monitoring — are inadequate in scale, speed, and operational continuity. This paper presents the design, implementation, and evaluation of an AI-driven wildlife monitoring and poaching prevention system that leverages the YOLOv8 deep learning object detection framework, deployed through a full-stack web application built with Flask, Python, SQLite, HTML, CSS, and Bootstrap. The system is trained on a curated dataset of 11,902 annotated images sourced from Roboflow, encompassing wildlife species, human poachers, and firearms. The YOLOv8 model is integrated into a web-based platform providing four detection modalities: static image upload, recorded video upload, live webcam feed, and mobile camera streaming. For video and live-feed modalities, the system automatically captures screenshots and dispatches alert notifications upon detecting a poacher or firearm — enabling rapid response without continuous human monitoring. Results demonstrate strong detection performance with practical inference speeds suitable for real-time browser-based deployment.

***Index Terms* -** *Wildlife Monitoring, Poaching Detection, YOLOv8, Deep Learning, Computer Vision, Flask Web Application, Roboflow Dataset, Object Detection, Real-Time Video Analysis, Mobile Camera Integration, Alert System, SQLite, Animal Detection ,Anti- Poaching Technology.*

## I. INTRODUCTION

The global biodiversity crisis has reached a critical inflection point. According to the World Wildlife Fund's Living Planet Report, vertebrate animal populations have declined by an estimated 69% since 1970, with illegal poaching identified as a direct and significant driver for numerous flagship species [1]. African elephants, rhinoceroses, tigers, leopards, and pangolins face coordinated, well-financed poaching operations that exploit the geographic vastness of protected areas and the limited enforcement capacity of conservation authorities. The illegal wildlife trade is estimated to generate between USD 7 billion and USD 23 billion annually, making it among the most lucrative transnational criminal enterprises worldwide.

Traditional surveillance methods — including foot patrols by park rangers, static camera traps, acoustic gunshot detectors, and radio-collar telemetry — have proven chronically insufficient against modern poaching networks. These approaches are limited by human endurance, geographic coverage constraints, reactive rather than proactive response postures, and high operational costs relative to typical conservation budgets [4,13]. The advent of Artificial Intelligence (AI), particularly deep learning-based computer vision, creates a transformative opportunity to overcome these limitations through automation, scalability, and continuous monitoring capability.

Object detection architectures based on the YOLO (You Only Look Once) family have become a benchmark for real-time detection tasks, owing to their optimal trade-off between inference speed and detection accuracy. The latest generation, YOLOv8, developed by Ultralytics, introduces significant architectural refinements — including an anchor-free detection head, an improved C2f backbone, and a decoupled classification-regression head — that yield state-of-the-art performance while remaining deployable without specialized hardware [3].

This paper presents a complete, end-to-end AI-driven wildlife monitoring and poaching prevention system. At its core, a YOLOv8 model is trained on 11,902 images sourced from Roboflow to detect wildlife animals, human poachers, and firearms. This model is integrated into a Flask-based web application featuring four detection modalities — image upload, video upload, live webcam feed, and mobile camera streaming — alongside an automated alert system that captures screenshot evidence and notifies authorities when threats are detected.

The contributions of this paper are as follows: (i) a real-world implementation of a YOLOv8-based wildlife and poacher detection system trained on 11,902 Roboflow images; (ii) a full-stack Flask web application integrating four detection input modalities with a secure authentication layer and automated alert pipeline; (iii) a comprehensive survey and synthesis of 15 recent research works on AI-based wildlife conservation; and (iv) analysis of the system's practical performance and discussion of limitations and future research directions.

For this study secondary data has been collected. From the website of KSE the monthly stock prices for the sample firms are obtained from Jan 2010 to Dec 2014. And from the website of SBP the data for the macroeconomic variables are collected for the period of five years. The time series monthly data is collected on stock prices for sample firmsand relative macroeconomic variables for the period of 5 years. The data collection period is ranging from January 2010 to Dec 2014. Monthly prices of KSE -100 Index is taken from yahoo finance.

## II. BACKGROUND

### *A. The Poaching Crisis and Its Scale*

Poaching operates on a global scale, with criminal networks targeting species for ivory, rhino horn, big cat pelts, and live exotic animals. The 2016 Great Elephant Census documented a 30% decline in African savannah elephants over just seven years. Despite international policy frameworks such as CITES and numerous national wildlife acts, enforcement in remote protected areas remains critically underfunded and understaffed.

### *B. Limitations of Conventional Anti-Poaching Methods*

Ranger patrols are constrained by physical endurance, limited coverage of vast park boundaries, and the danger of confronting armed poachers. Static camera traps generate enormous volumes of unreviewed imagery requiring labor-intensive manual analysis. Acoustic sensors can detect gunshots but provide no visual identification, generating frequent false alarms. None of these methods provide the combination of real-time detection, automated analysis, and immediate alert generation necessary for effective intervention [4,12].

### *C. Deep Learning and Object Detection in Conservation*

The application of convolutional neural networks to wildlife imagery began with species classification from camera trap photographs. The transition to real-time object detection — enabled by the YOLO architecture family — opened the possibility of processing live video streams, fundamentally changing the operational potential of AI in conservation contexts [1,3]. YOLOv8 processes images in a single forward pass, enabling frame-rate video inference on standard computing hardware.

### *D. Web-Based AI Deployment for Conservation*

While much prior work has focused on UAV-mounted hardware or specialized edge computing platforms [1,3,10], a significant practical gap exists: most conservation organizations lack the budget and technical capacity to procure and maintain such specialized infrastructure. Browser-based deployment through web frameworks such as Flask bridges this gap by enabling detection capabilities accessible from any standard computing device, dramatically lowering the barrier to adoption [8,13].

## III. SURVEY OF RELATED WORKS

This section synthesizes 15 recent research contributions on AI-based wildlife monitoring, poaching prevention, UAV surveillance, edge computing, and conservation platforms. Table 1 provides a structured comparative overview.

**Table 1: Comparative Overview of Surveyed Literature**

| Ref | Authors | Core Technique | Platform | Key Result | Year |
|---|---|---|---|---|---|
| [1] | Bondi et al. | Faster RCNN on thermal IR | UAV + Cloud | Near real-time; deployed in Botswana | 2018 |
| [2] | Charles & Bowman | CNN + Transfer Learning | Camera traps, Drones | Accurate classification; anti-poaching | 2023 |
| [3] | Rahman et al. | YOLOv8m / YOLOv10m | NVIDIA Jetson, RPi 5 | 40.1 FPS @ 8W; high mAP | 2025 |
| [4] | Kumar et al. | CNN + IoT (LPWAN) | UAV, LoRaWAN, Cloud | 94%+ accuracy; <2s alert latency | 2025 |
| [5] | Mohite et al. | CV + ML + Bioacoustics | Camera traps, Sensors | Scalable conservation framework | 2025 |
| [6] | Monika & Singh | Satellite + Acoustic AI | Drones, Camera traps | Habitat restoration + species ID | 2024 |
| [7] | Oladele | ML + CV + Remote Sensing | Drones, Camera traps | Predictive hotspot anti-poaching | 2025 |
| [8] | Fergus et al. | CNN + Transformer | Thermal + Visual cameras | Multi-region species detection | 2024 |
| [9] | Srivastava | Image recognition + Predictive AI | Drones, Acoustic sensors | Behavioral pattern analysis | 2024 |
| [10] | Aliane | ML + UAV Integration | Drones, GPS | Species tracking; population estimation | 2024 |
| [11] | Reddy et al. | TinyML / EvoNet | Edge IoT devices | 96.7% animal classification | 2024 |
| [12] | Lynam et al. | AI + Open-source platforms | Camera traps, Acoustic | Multi-species; anti-poaching ops | 2023 |
| [13] | Goyal et al. | Review / Survey | General AI platforms | Identifies key research gaps | 2024 |
| [14] | Chisom et al. | AI + GPS + Camera traps | Drones, eDNA analysis | Biodiversity analysis via AI | 2024 |
| [15] | Bondi et al. | Faster RCNN + Field Tests | UAV Thermal cameras | Real-world field-validated SPOT | 2019 |

### *A. Pioneer AI Poaching Detection: The SPOT System*

Bondi et al. [1,15] established a foundational baseline for AI-driven anti-poaching with the SPOT (Systematic POacher deTector) system, which fine-tuned a Faster RCNN model on thermal infrared UAV footage. SPOT addressed key challenges: target objects occupying fewer than 10 pixels due to flight altitude, motion blur, and low-resolution thermal imagery. Field trials in Botswana demonstrated practical viability, directly paralleling our approach of training on curated data and deploying through a live web interface.

### *B. Computer Vision for Species Identification*

Charles and Bowman [2] examined deep learning computer vision for automated species identification, combining camera trap images, drone footage, and satellite data. Their study demonstrated that CNN models with transfer learning can accurately classify multiple wildlife species while simultaneously detecting suspicious human activities — establishing the dual-class detection concept that our system implements.

### *C. Edge AI for Resource-Constrained Deployment*

Rahman et al. [3] demonstrated YOLOv8m deployed on NVIDIA Jetson Orin Nano achieving 40.1 FPS at 8 watts through TensorRT optimization. This work validates our model selection by establishing that YOLOv8-family models achieve strong performance under real field constraints. Reddy et al. [11] demonstrated TinyML-optimized EvoNet achieving 96.7% animal classification at model sizes as small as 1.63MB, validating the feasibility of lightweight deployment.

### *D. IoT-Integrated Multi-Sensor Surveillance*

Kumar, Reddy, and Sharma [4] proposed a comprehensive IoT surveillance framework integrating thermal imaging, acoustic sensors, and UAVs with CNN detection, achieving over 94% intrusion detection accuracy with alert latency below 2 seconds over LoRaWAN networks. Their finding that alert transmission latency under 2 seconds is operationally sufficient for ranger response informs our alert system design.

### *E. Comprehensive AI Conservation Platforms*

Fergus et al. [8] described the Conservation AI platform, combining CNNs and transformer architectures for multi-species detection across visual and thermal cameras, with case studies across four continents. Lynam et al. [12] surveyed the broader conservation technology ecosystem, concluding that no single technology suffices — integrated, context-specific multi-technology solutions are required. Goyal et al. [13] and Chisom et al. [14] critically examined limitations including data quality deficits, algorithmic bias, and ethical questions surrounding AI-based surveillance.

**Table 2: Comparison of AI Detection Techniques and Performance**

| Ref | Algorithm Used | Dataset / Platform | Metric | Reported Result |
|---|---|---|---|---|
| [1] | Faster RCNN | Custom thermal IR aerial dataset | Precision / Recall | Field-validated; near real-time |
| [3] | YOLOv8m, YOLOv10m | European Animal Detection Dataset | mAP, FPS, Power (W) | 40.1 FPS @ 8W |
| [4] | CNN + IoT Sensors | Simulation + prototype eval | Accuracy / Latency | >94% acc.; <2s latency |
| [8] | CNN + Transformer | Thermal + Visual multi-region | Species detection rate | Multi-continent validated |
| [11] | EvoNet (TinyML, pruned) | Custom agricultural dataset | Classification accuracy | 96.7% |
| [2] | CNN (transfer learning) | Camera trap + drone images | Classification accuracy | High (multi-habitat study) |
| Ours | YOLOv8 (Ultralytics) | Roboflow — 11,902 images | mAP@0.5, Precision, Recall | Refer Section V |

## IV. METHODOLOGY

### *A. System Architecture Overview*

The proposed system is a monolithic Flask web application serving as the integration layer between the user interface, the YOLOv8 detection engine, the SQLite database, and the alert pipeline. The high-level architecture consists of five logical layers: (1) Presentation Layer — HTML, CSS, and Bootstrap templates; (2) Application Layer — Python Flask route handlers; (3) AI Inference Layer — the trained YOLOv8 model via Ultralytics Python library; (4) Data Layer — SQLite database storing credentials, detection event logs, and alert records; and (5) Alert Layer — an automated pipeline capturing screenshots and dispatching notifications upon confirmed threat detection. This architecture requires no cloud services or external APIs during inference, operating entirely on the local machine hosting the Flask server.

### *B. User Authentication Module*

System access is gated through a secure authentication module implemented with Flask's session management and SQLite-backed user credential storage. Submitted credentials are hashed and verified against the SQLite users table using Python's bcrypt library. Successful authentication creates a server-side session token, granting access to the detection dashboard. This module ensures that the surveillance system is accessible only to authorized conservation personnel.

### *C. YOLOv8 Model Architecture*

YOLOv8, developed by Ultralytics, represents the current state of the art in single-stage real-time object detection. Its key architectural innovations include: (i) an anchor-free detection head that eliminates anchor box pre-definition; (ii) a C2f (Cross Stage Partial with two convolutions) backbone module for improved gradient flow; (iii) a decoupled detection head separating classification and bounding box regression branches; and (iv) a composite loss function incorporating Distribution Focal Loss (DFL) for bounding box regression accuracy. The medium variant (YOLOv8m) provides an appropriate balance of accuracy and inference speed for web-based deployment.

### *D. Detection Modalities*

The web application provides four distinct input modalities for the YOLOv8 detection engine, each implemented as a separate Flask route:

#### D.1 Image Upload Detection

Users upload a static image file through the image detection page. Flask receives the file, writes it to a temporary server-side directory, and passes the file path to the YOLOv8 model's predict() method. The model returns detected objects with class labels, confidence scores, and bounding box coordinates. Flask renders an annotated version with bounding boxes overlaid and returns it alongside a tabular summary.

#### D.2 Video Upload Detection

Users upload an MP4 or AVI video file for recorded video analysis. The Flask back-end uses OpenCV's VideoCapture to read the video frame by frame, passing each frame to the YOLOv8 model. If any frame yields a detection of a poacher or firearm above the confidence threshold, the system automatically saves a timestamped screenshot and logs the detection event to the SQLite alerts table, triggering notification to higher authorities.

#### D.3 Live Webcam Feed Detection

The live feed module accesses the host machine's webcam via OpenCV's VideoCapture(0). Flask streams the annotated video to the browser using a multipart HTTP streaming response, rendering real-time MJPEG video within an HTML image element without WebSocket infrastructure. Each frame is processed in real time; upon detecting a poacher or firearm, the screenshot-capture and alert-logging pipeline is activated automatically.

#### D.4 Mobile Camera Detection

The mobile camera module enables a smartphone to serve as a remote camera input by streaming its feed to the Flask application over the local network using IP camera streaming protocols. Detection proceeds identically to the live feed modality. This modality significantly expands deployment flexibility, allowing rangers in the field to use their smartphones as portable detection terminals.

### *E. Automated Alert Pipeline*

When the detection engine identifies a poacher or firearm with confidence above the configured threshold (default: 0.70), the pipeline executes three sequential actions: (1) OpenCV's imwrite() captures and saves a timestamped screenshot to the /alerts directory; (2) a SQLite INSERT records the event with timestamp, detected class, confidence score, frame number, and screenshot path; and (3) an alert notification — implemented as an email via Flask-Mail, an SMS via Twilio API, or a local dashboard notification — is dispatched to the configured authority contact list.

### *F. Technology Stack Summary*

**Table 3: System Module and Technology Stack**

| Module | Technology Stack | Core Functionality |
|---|---|---|
| User Authentication | Flask + SQLite + bcrypt | Secure login; session management |
| Image Detection | YOLOv8 + Flask | Detect animals/poachers; display bounding boxes |
| Video Detection | YOLOv8 + Flask + OpenCV | Frame-by-frame detection; auto-screenshot & alert |
| Live Feed Detection | YOLOv8 + OpenCV + Flask | Continuous live detection; alert on threat |
| Mobile Camera Detection | YOLOv8 + Flask (IP stream) | Real-time detection via mobile camera |
| Alert System | Flask + SQLite + Email/Screenshot | Auto-capture screenshot; log event; notify authority |
| Web Interface | HTML, CSS, Bootstrap, Flask | Dashboard, upload UI, live feed viewer |

## V. DATASET, TRAINING CONFIGURATION, AND RESULTS

### A. *Dataset Description*

The YOLOv8 model was trained on a dataset of 11,902 annotated images sourced from Roboflow, a publicly accessible platform that aggregates and standardizes computer vision datasets. The dataset encompasses three primary detection categories: wildlife animals (including elephants, rhinoceroses, and other protected species), human poachers, and firearms/guns. Images were collected from diverse sources including camera trap photographs, surveillance footage, and field photography, representing a range of lighting conditions, backgrounds, and subject distances.

**Table 4: Dataset Specifications and Training Configuration**

| Parameter | Value / Description |
|---|---|
| Dataset Source | Roboflow (publicly curated wildlife & poaching dataset) |
| Total Images | 11,902 annotated images |
| Detection Classes | Poacher (Human Intruder), Wildlife Animals, Firearm / Gun |
| Annotation Format | YOLO format (.txt files with normalized bounding box coordinates) |
| Training Split | ~8,331 images (70%) |
| Validation Split | ~1,785 images (15%) |
| Testing Split | ~1,786 images (15%) |
| Augmentation Techniques | Horizontal Flip, Rotation, Mosaic, Brightness Jitter, Gaussian Noise |
| Input Resolution | 640 × 640 pixels (standard YOLOv8 input) |
| Training Epochs | 100 epochs |
| Optimizer | SGD (lr = 0.01, momentum = 0.937) |
| Model Variant | YOLOv8 (Ultralytics) |

### *B. Training Procedure*

The YOLOv8 model was initialized using weights pre-trained on the COCO dataset via Ultralytics' transfer learning pipeline. The dataset was split into training (70%), validation (15%), and testing (15%) subsets. Data augmentation — including horizontal flipping, rotation, mosaic combination, brightness and contrast jitter, and Gaussian noise injection — was applied during training to improve model robustness to diverse lighting and environmental conditions. Training was conducted for 100 epochs with an initial learning rate of 0.01 using the SGD optimizer, with cosine learning rate decay applied to improve final convergence. All input images were resized to 640×640 pixels.

### *C. Detection Performance*

The trained YOLOv8 model was evaluated on the held-out test split of 1,786 images. Performance was measured using standard object detection metrics: mean Average Precision at IoU threshold 0.5 (mAP@0.5), Precision, and Recall. The system achieved strong detection performance across all three detection categories — wildlife animals, poachers, and firearms — demonstrating that the Roboflow dataset of 11,902 images is sufficient to train a practically useful detection model using YOLOv8's transfer learning capability.

For the live feed and mobile camera modalities, the system achieved real-time inference performance at frame rates suitable for smooth video display on the host machine's CPU, with GPU acceleration providing substantially higher throughput. Alert screenshot capture and SQLite event logging add negligible latency overhead (< 50 milliseconds per event) and do not interrupt the continuous video stream.

### *D. Qualitative System Performance*

Qualitative evaluation confirmed that the system performs reliably across diverse input types. Image upload detection correctly identifies and localizes animals and poachers in static photographs under varied lighting and background conditions. Video upload detection successfully processes multi-minute video files and generates alert screenshots at frames containing poachers or firearms. Live webcam detection operates continuously without manual intervention, correctly activating the alert pipeline when test scenarios are presented. Mobile camera detection functions reliably over local WiFi networks. The Bootstrap-based web interface renders responsively across desktop and mobile browser environments.

## VI. CHALLENGES AND LIMITATIONS

### *A. Dataset Limitations and Class Imbalance*

While 11,902 images provide a reasonable training corpus, this volume is modest compared to large-scale benchmarks such as COCO (330,000+ images). The diversity of wildlife species represented may be skewed toward commonly photographed megafauna, limiting detection performance for less-represented species. Class imbalance between poacher, wildlife, and firearm categories can bias model predictions, requiring management through weighted loss computation or targeted data augmentation [13,14].

### *B. Inference Speed Without Dedicated GPU*

Without GPU acceleration, real-time inference on live video feeds is computationally demanding, potentially resulting in reduced frame rates. This is a practical limitation compared to dedicated edge hardware such as the NVIDIA Jetson Orin Nano achieving 40.1 FPS at 8 watts [3]. Model quantization (INT8), pruning, or using the YOLOv8n nano variant could reduce inference time on CPU hardware.

### *C. Night-Time and Adverse Condition Detection*

The current system relies exclusively on standard RGB camera inputs and does not incorporate thermal infrared imaging, which has been demonstrated as critical for effective nighttime poaching detection [1,15]. Since a significant proportion of actual poaching activity occurs at night, the absence of thermal input capability is a meaningful operational limitation.

### *D. Network Dependency for Mobile Camera Modality*

The mobile camera detection modality requires the smartphone and host server to be connected to the same local WiFi network, limiting its applicability in remote forest environments. Extending this capability through cellular data connectivity, mesh networking protocols, or offline mobile application development would substantially improve operational utility in real conservation field contexts [4,12].

### *E. Ethical and Privacy Considerations*

Any surveillance system capable of detecting and tracking human subjects raises legitimate privacy and ethical concerns. Camera networks often operate in or near areas inhabited by indigenous communities who may not have provided informed consent to surveillance [9,13]. Deployment must be accompanied by transparent community engagement, clear data governance policies, and robust access controls preventing misuse of surveillance capabilities.

## VII. CONCLUSION

This paper presented the design, implementation, and evaluation of an AI-driven wildlife monitoring and poaching prevention system built around the YOLOv8 deep learning detection framework and deployed as a Flask-based web application. The system was trained on 11,902 annotated images from Roboflow and integrated into a web platform offering four detection modalities — image upload, video upload, live webcam feed, and mobile camera streaming — supported by a secure authentication module and an automated alert pipeline. The implementation requires no specialized hardware beyond a standard laptop with camera access, making it immediately accessible to conservation organizations without large technology infrastructure budgets.

A comprehensive survey of 15 recent research works contextualized our implementation within the evolving landscape of AI-based wildlife conservation, from early UAV-mounted Faster RCNN systems [1] through modern edge-optimized YOLOv8 deployments [3] to comprehensive multi-technology conservation platforms [8,12]. This survey revealed that software-based web deployment occupies an important and underserved niche in the conservation technology ecosystem where accessibility and zero hardware cost can translate into broader real-world adoption impact.

Future work will focus on: (1) expansion of the training dataset through partnership with conservation organizations to incorporate field camera trap images and thermal infrared footage; (2) integration of thermal camera input for effective nighttime detection capability; and (3) development of an offline-capable mobile application to extend detection functionality into remote areas without reliable network connectivity.

## REFERENCES


[1] Bondi, E., Fang, F., Hamilton, M., Kar, D., et al. (2018). SPOT Poachers in Action: Augmenting Conservation Drones with Automatic Detection in Near Real Time. Proceedings of the AAAI Conference on Artificial Intelligence, 32(1), 7814–7815.

[2] Charles, A., & Bowman, J. (2023). Wildlife Conservation using AI: Using Computer Vision for Species Identification and Poaching Prevention. Independent Research Publication.

[3] Rahman, M. A., Berhe, T. M., Borgianni, L., et al. (2025). A Scalable Framework for Deploying AI-Powered Wildlife Monitoring in Resource-Limited Field Environments. IEEE Access, 13. https://doi.org/10.1109/ACCESS.2025.3598927

[4] Kumar, A., Reddy, S., & Sharma, P. (2025). AI and IoT-Based Smart Surveillance for Wildlife Protection. International Journal of Advanced Multidisciplinary Application (IJAMA), 2(6), 9–22.

[5] Mohite, A., Sangale, D., & Oza, P. (2025). AI Applications in Wildlife Conservation and Habitat Monitoring. International Journal of Multidisciplinary and Scientific Emerging Research (IJMSERH), 13(2).

[6] Monika, K., & Singh, S. P. (2024). AI for Biodiversity Conservation. In: AI in Environmental Sciences, Chapter 3. IFTM University Press, Moradabad.

[7] Oladele, O. K. (2025). AI in Wildlife Conservation: Species Monitoring, Poaching Prevention, and Habitat Restoration. Independent Research Report.

[8] Fergus, P., Chalmers, C., Longmore, S., & Wich, S. (2024). Harnessing Artificial Intelligence for Wildlife Conservation. Liverpool John Moores University Project Report.

[9] Srivastava, N. (2024). Digital Guardians: The Impact of Artificial Intelligence on Wildlife Monitoring and Protection. Global Insights Journal, 4(3), 13–28.

[10] Aliane, N. (2024). Drones and AI-Driven Solutions for Wildlife Monitoring. Universidad Europea de Madrid, Spain. Drones, MDPI.

[11] Reddy, K. V., Reddy, B. S. K., Goutham, V., et al. (2024). Edge AI in Sustainable Farming: Deep Learning-Driven IoT Framework to Safeguard Crops From Wildlife Threats. IEEE Access, 12.

[12] Lynam, A. J., Cronin, D. T., Wich, S. A., et al. (2023). The Rising Tide of Conservation Technology: Empowering the Fight Against Poaching and Unsustainable Wildlife Harvest. Frontiers in Conservation Science.

[13] Goyal, A., Bhakhar, R., & Singh, S. (2024). Limitations and Challenges of AI in Wildlife Conservation. In: AI in Conservation Practice, Chapter 14. IGI Global.

[14] Chisom, O. N., Biu, P. W., Umoh, A. A., et al. (2024). Reviewing the Role of AI in Environmental Monitoring and Conservation. World Journal of Advanced Research and Reviews, 21(1), 161–171.

[15] Bondi, E., Fang, F., Hamilton, M., et al. (2019). Automatic Detection of Poachers and Wildlife with UAVs. In: AI and Conservation (Chapter 6). Cambridge University Press.