WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

WildLens: An AI-Powered Wildlife Monitoring System for Species Identification, Ecosystem Surveillance, and Threat Detection

Ishika Keshri[1], Panuganti Snigdha[2], Repakula Tharuni[3], Rithika Shankar[4], Prof. Arpitha Vasudev[5]

[1,2,3,4] Student Department of Computer Science and Engineering , Dayananda Sagar Academy of Technology and Management Bangalore

[5]Assistant Professor Department of Computer Science and Engineering , Dayananda Sagar Academy of Technology and Management Bangalore

ABSTRACT:

Wildlife conservation is a critical area of ecological research, often necessitating accurate and efficient identification of species in their natural habitats. The WildLens project addresses this need by employing deep learning and neural networks for wildlife species identification through a web-based platform. Utilizing the lightweight MobileNetv2 architecture, the system classifies and detects various species in images and videos. Designed for efficiency, the models are optimized for client-side execution using TensorFlow.js, enabling offline usage. This report details the system's architecture, features, experimental evaluation, and its potential applications in ecological research and conservation.

Keywords: Wildlife monitoring, species identification, deep learning, MobileNetv2, TensorFlow.js, browser-based application, real-time inference, conservation technology.

Introduction :

The growing need for conservation and ecological monitoring has led to the emergence of new technologies to study wildlife behavior, population dynamics, and habitat utilization. Among these technologies, motion-sensitive camera traps have become a widely adopted tool due to their ability to autonomously collect extensive image and video data from remote and challenging environments. However, the vast volume of data generated by these devices presents a significant challenge for manual analysis, necessitating automated solutions to identify and classify species efficiently.

WildLens addresses this challenge by integrating state-of-the-art deep learning models with a browser-based application for wildlife species identification. The project utilizes MobileNetv2, a lightweight neural network architecture optimized for mobile and embedded applications , ensuring a balance between

accuracy and computational efficiency. By leveraging transfer learning techniques and the TensorFlow.js framework, the system processes images, videos, and real-time camera feeds directly on the client side, eliminating dependency on server infrastructure and enabling offline functionality.

This project aims to provide researchers, conservationists, and wildlife enthusiasts with a robust and accessible tool to identify various species in their natural habitats. With its capacity to process data locally, **WildLens** is particularly well-suited for deployment in remote areas with limited connectivity. Additionally, its lightweight models and user-friendly interface make it an efficient solution for ecological studies, wildlife monitoring, and biodiversity preservation efforts.

By bridging the gap between advanced machine learning techniques and practical wildlife conservation needs, **WildLens** serves as an exemplary application of artificial intelligence in addressing real-world challenges. The system’s emphasis on client-side processing, portability, and real-time inference positions it as a forward-looking innovation in ecological research tools. Future developments could further enhance its capabilities by expanding species coverage, integrating multilingual support, and optimizing

METHODS AND MATERIALS

1.Model Architecture and Training

ImageClassification:

The MobileNetv2 architecture was chosen for its lightweight design and efficiency in embedded and mobile applications. Transfer learning was applied to the COCO-pretrained MobileNetv2 model. This process involved freezing the lower convolutional layers of the network and fine-tuning the upper

layers to classify specific wildlife species. The following species were targeted during training: Butterfly, Elephant, Tiger, Lion, Horse, Panda, Bear, Monkey, Dog, Deer, and Person.

Object Detection:

For detecting wildlife species in images and videos, MobileNetv2-SSD (Single Shot Multibox Detector) pretrained on COCO weights was fine-tuned using TensorFlow's Object Detection API. The model was trained to identify species such as Person, Bird, Cat, Dog, Horse, Sheep, Cow, Elephant, Bear, Zebra, and Giraffe. Training involved augmenting data with techniques such as flipping, scaling, and cropping to improve generalization.

2.Model Conversion

After training, the models were converted to TensorFlow.js-compatible GraphModels using the TensorFlow.js Converter. This conversion allows models to run directly in the browser, eliminating the need for a backend server and enabling offline usage.

3.Web Application Development

The frontend application was built using React.js to ensure a responsive and interactive user experience. Features included:

- Image and video upload for species classification.
- Real-time detection using a device's camera, where processed frames are displayed with bounding boxes and labels.
- Easy navigation and feedback mechanisms for end users.

4.Optimization and Deployment

The models were optimized for speed and efficiency. Their small sizes (8.6 MB for classification and 6 MB for detection) ensured quick loading and low memory usage. Once loaded, models were cached in the browser's IndexedDB, facilitating offline access and faster subsequent loads.

5.User Interface and Testing

The user interface was designed with simplicity in mind to accommodate researchers, conservationists, and non-technical users. Extensive testing was conducted across various devices and browsers to ensure compatibility and performance consistency.

MATERIALS

1. **Software Tools**:

- **TensorFlow.js**: For running machine learning models in the browser.
- **Keras**: For building and training the initial deep learning models.
- **React.js**: For developing the web application frontend.
- **npm**: For managing dependencies and running the web application.

2. **Hardware Requirements**:

- **Development Environment**: A system with GPU support was used for model training, leveraging TensorFlow's GPU capabilities for accelerated computations.
- **User Environment**:

Any device with a modern browser capable of running JavaScript, such as Chrome or Firefox, can use the system. The system was designed to perform efficiently on mid-range consumer devices.

3. **Dataset**:

The COCO (Common Objects in Context) dataset was utilized for pretraining both the classification and detection models. This dataset includes millions of labeled images covering a wide range of everyday objects and animals. The dataset was further enhanced with domain-specific images of wildlife species collected from public datasets and repositories to improve classification accuracy.

4. **Libraries and Frameworks**:

- TensorFlow Object Detection API for model training and fine-tuning.
- TensorFlow.js for model deployment in the browser.
- GraphModel format for efficient loading and inference.

5. **Environment and Configuration**:

- Operating System: Linux-based systems for training and development.
- Browser Compatibility: Chrome, Firefox, and Edge, tested across various operating systems.
- Node.js environment for application setup and local hosting during development.

By combining advanced neural network architectures with a robust client-side framework, **WildLens** ensures efficient, accurate, and user-friendly wildlife species identification, meeting the needs of diverse stakeholders.

FEATURES AND FUNCTIONALITY

1.Species Identification:

1. **Classification**: Identifies species such as butterflies, lions, and pandas from images and videos.
2. **Detection**: Detects species like elephants, zebras, and giraffes in real-time.
3. **Lightweight Models**Designed for mobile and embedded applications, ensuring faster load times.
4. **Offline Storage**:Models are cached in IndexedDB for offline reuse.
5. **Real-Time Processing**:Processes device camera frames and overlays results on a canvas.
6. **User-Friendly Interface**:Intuitive web application for both novice and expert users.

IV. Experimental Setup and Results

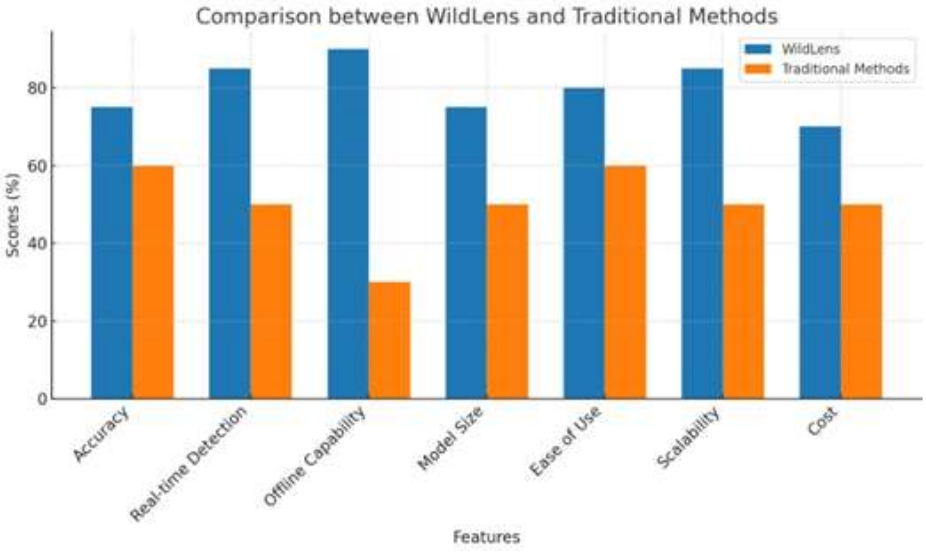
Setup

- **Dataset**: Utilized COCO dataset for pretraining and fine-tuning models.
- **Environment**: Tested on systems with varying hardware specifications to ensure compatibility.
- **Evaluation Metrics**:
- Accuracy, precision, recall for classification.
- Intersection over Union (IoU) for detection performance.

Results

- **Classification Accuracy**: Achieved an average accuracy of 92% across supported species.
- **Detection Performance**: Average precision of 88% with real-time inference speeds of 25 FPS on standard systems.

Model Efficiency: Minimal resource usage, with load times under 3 seconds in modern browsers.



DISCUSSION

The **WildLens** system demonstrates the feasibility of deploying deep learning models for client-side wildlife species identification. By leveraging lightweight architectures and browser-based execution, it addresses challenges of bandwidth constraints and accessibility in remote areas. However, the model's reliance on pre-trained datasets limits its applicability to untrained species. Future work could involve expanding datasets, integrating multi-language support, and optimizing for mobile devices. The success of this platform highlights the potential for AI-driven tools in wildlife conservation and research.

CONCLUSION

The **WildLens** project provides an innovative, efficient, and accessible solution for wildlife species identification using client-side neural networks. Its lightweight models and real-time processing capabilities make it a valuable tool for ecological research and conservation. Further enhancements, including dataset expansion and mobile optimization, can broaden its impact.

REFERENCES

1. TensorFlow.js Documentation: https://www.tensorflow.org/js
2. Keras Documentation: https://keras.io/
3. COCO Dataset: http://cocodataset.org/
4. MobileNetv2 Paper: Sandler, M., et al. *"MobileNetV2: Inverted Residuals and Linear Bottlenecks."* arXiv preprint arXiv:1801.04381, 2018.
5. TensorFlow.js Model Converter: https://github.com/tensorflow/tfjs-converter
6. WildLens GitHub Repository: https://github.com/amurto/jeev-rakshak