

# Wildlife Monitoring System with Image Classification

A Deep Learning Approach to Animal Detection and Classification





## MINI PROJECT

# Project Overview

### Subject

Advance Machine Learning  
(23CSE514)

### Submitted By

SHASHIDHAR B (23BTRCA049)  
Branch & Section: CSE-AI

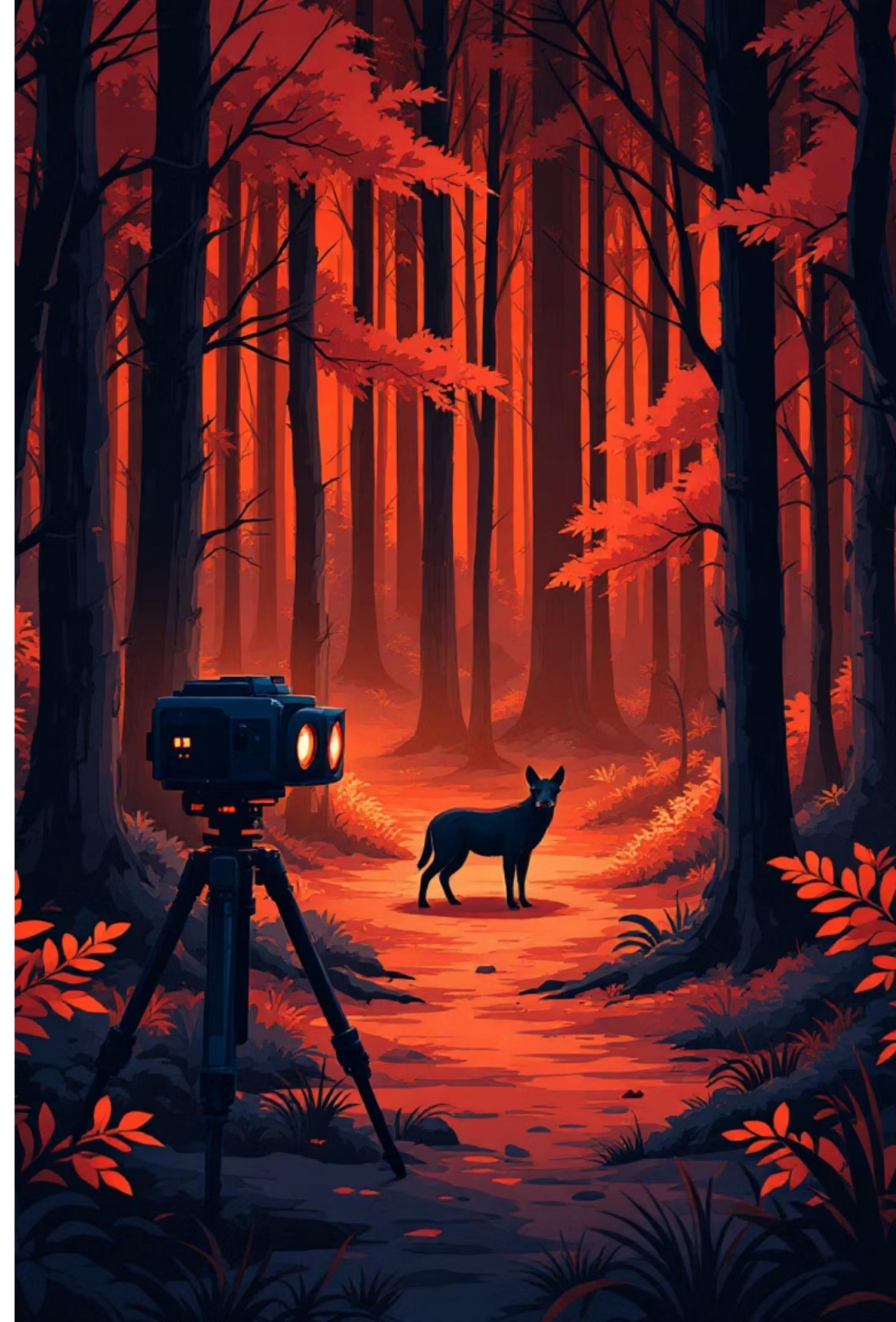
### Submitted To

Prof. Ms. Guruvammal S  
Assistant Professor, Jain (Deemed-  
To-Be) University

# Introduction: Automated Animal Recognition

This mini-project aims to build an Animal Detection and Classification System capable of identifying 90 different animal species using a deep-learning image classification model. Automated animal recognition is crucial for wildlife conservation, biodiversity monitoring, zoological studies, eco-tourism, smart surveillance, and educational platforms.

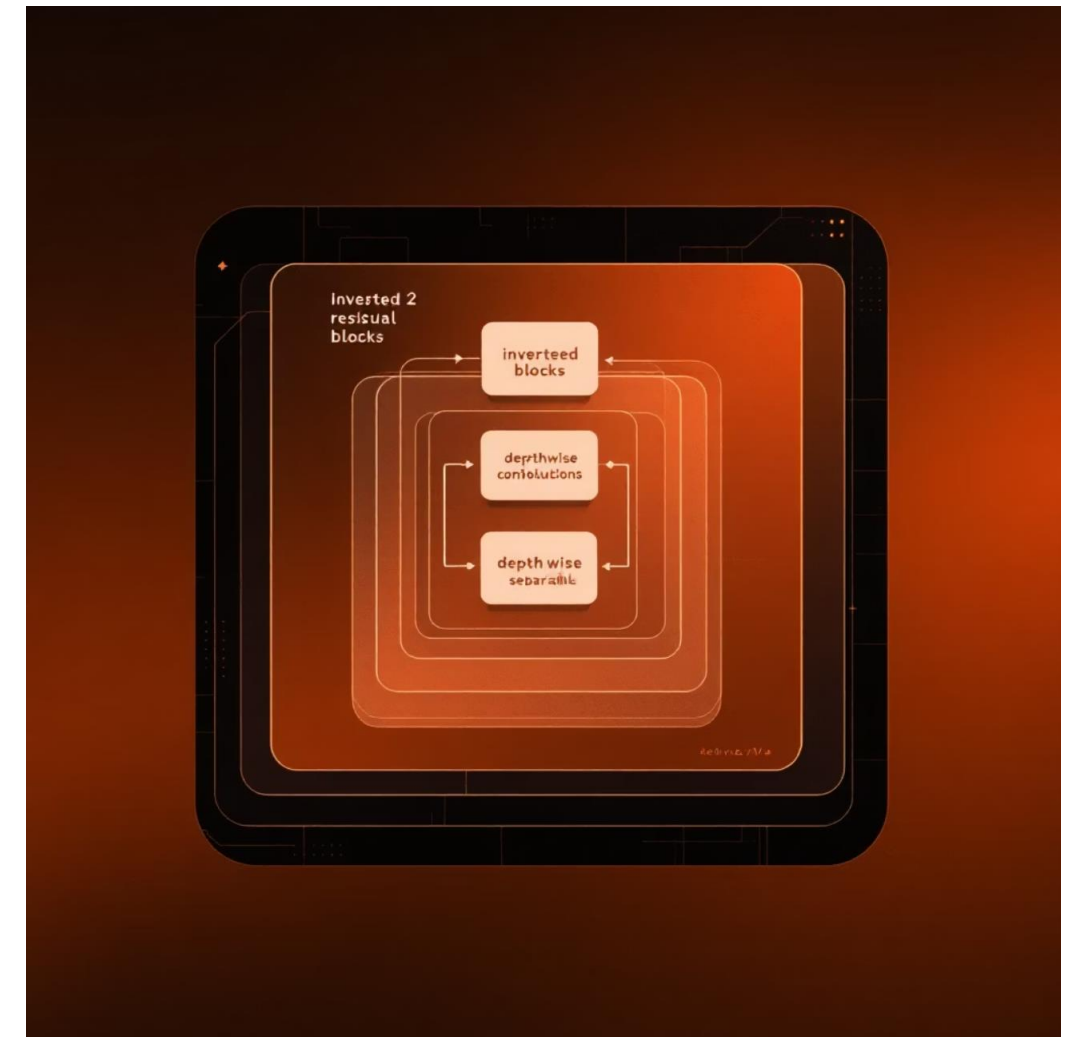
The project leverages MobileNetV2, a lightweight yet powerful Convolutional Neural Network (CNN) architecture. Unlike heavier models, MobileNetV2 offers high accuracy with significantly lower memory usage, making it ideal for deployment on platforms like Google Colab, smartphones, and embedded systems.



# Leveraging Transfer Learning with MobileNetV2

The system utilizes a pretrained MobileNetV2 model, initially trained on ImageNet (1.2 million images across 1,000 classes). Instead of training from scratch, which is computationally expensive, this project employs Transfer Learning.

By freezing the lower layers of MobileNetV2 and fine-tuning higher layers, the model efficiently learns features specific to the 90 animal classes. This technique significantly speeds up training, reduces resource requirements, and improves accuracy, demonstrating the practical use of Deep Learning and CNN feature extraction.



# Addressing Challenges in Animal Classification

Identifying animal species from images is challenging due to variations in lighting, pose, background, and angles, as well as high visual similarity between certain animals. Many datasets are imbalanced, containing inconsistent samples or low-quality images.

Traditional image-processing techniques struggle with these complexities. A deep learning-based approach, utilizing CNNs, automatically learns hierarchical features and adapts to different visual conditions. Transfer Learning further enhances performance by using pretrained ImageNet models, reducing training time and computational cost.



# Project Objectives

## → **Fast & Accurate Model**

Develop a deep learning model that is both quick and precise in its classifications.

## → **90 Animal Categories**

Achieve reliable classification across a broad spectrum of 90 distinct animal species.

## → **Efficient Feature Extraction**

Utilize pretrained MobileNetV2 for highly efficient feature extraction, optimizing performance.

## → **Robustness & Fine-Tuning**

Apply data augmentation and fine-tuning techniques to enhance model robustness and accuracy.

## → **Real-time Prediction**

Provide real-time prediction capabilities with confidence scores for practical applications.

# Dataset Description: Animal Image Dataset

The project uses the "Animal Image Dataset (90 Different Animals)" from Kaggle. This dataset is organized into 90 folders, each representing a distinct animal species like antelope, bear, bat, camel, cobra, deer, fox, kangaroo, and zebra.

The dataset contains images with varying lighting conditions, angles, backgrounds, and resolutions, providing a challenging and realistic benchmark for image classification. These variations ensure the model generalizes well rather than memorizing specific patterns.



1

## Total Classes

90 unique animal species.

2

## Images per Class

Varies from 50 to 400 images.

3

## File Format

JPG / PNG, non-uniform sizes.

4

## Data Split

80% Training, 10% Validation, 10% Testing.

# Methodology: Building the System

01

## Dataset Acquisition

Downloaded via Kaggle API, extracted, and corrected folder paths.

02

## Dataset Splitting

Split into 80% training, 10% validation, and 10% testing using `splitfolders.ratio()`.

03

## Image Preprocessing

Rescaled pixel values, resized images to 160x160, created batches, and shuffled training data. Applied data augmentation (rotation, zoom, flips).

04

## Model Architecture

MobileNetV2 pretrained on ImageNet, with base layers frozen. Added GlobalAveragePooling2D, Dropout (0.3), and a Dense layer (90 neurons, softmax).

05

## Training Phase

Initially trained only the classifier head, then unfroze the last 30 layers for fine-tuning.

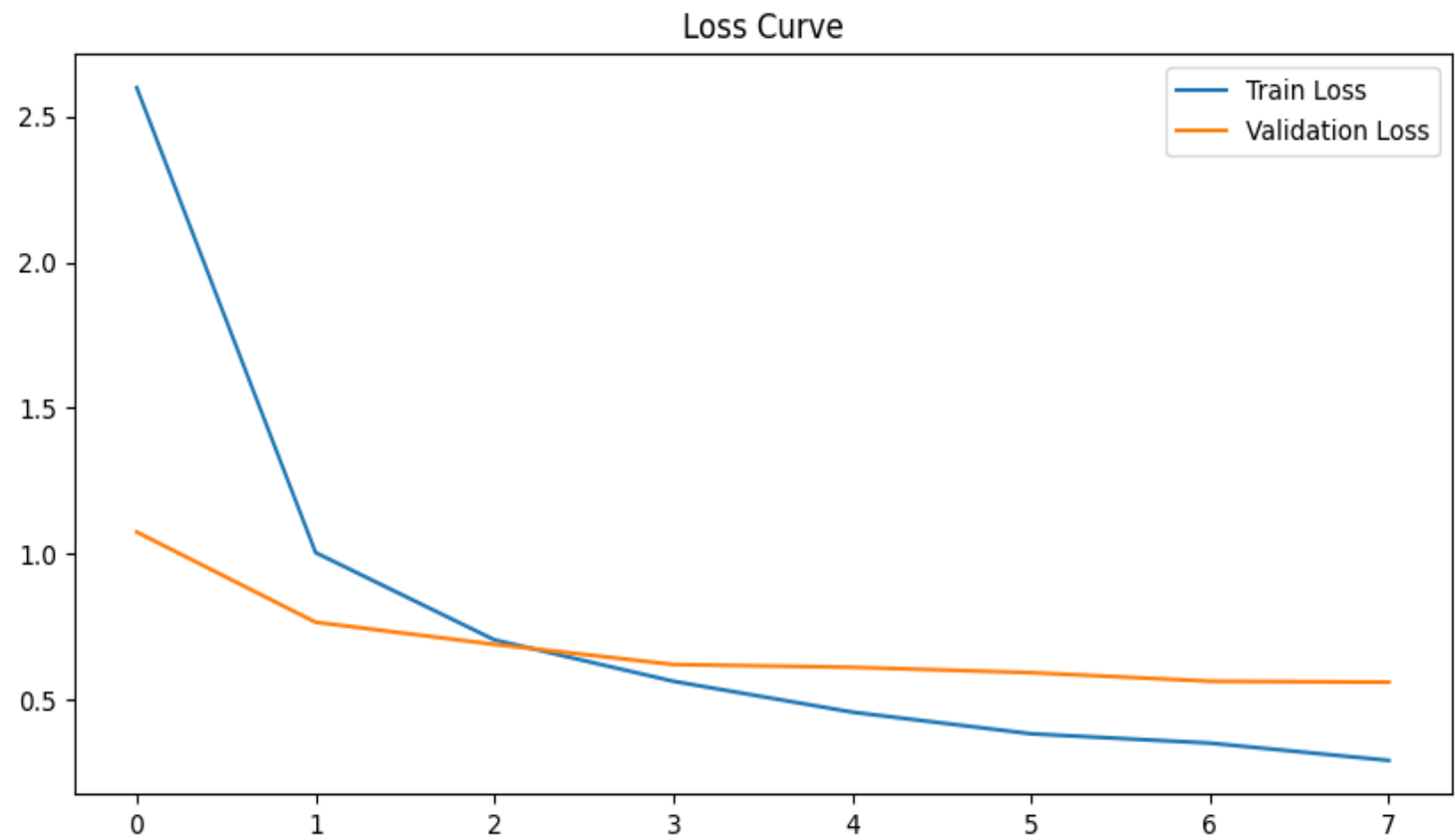
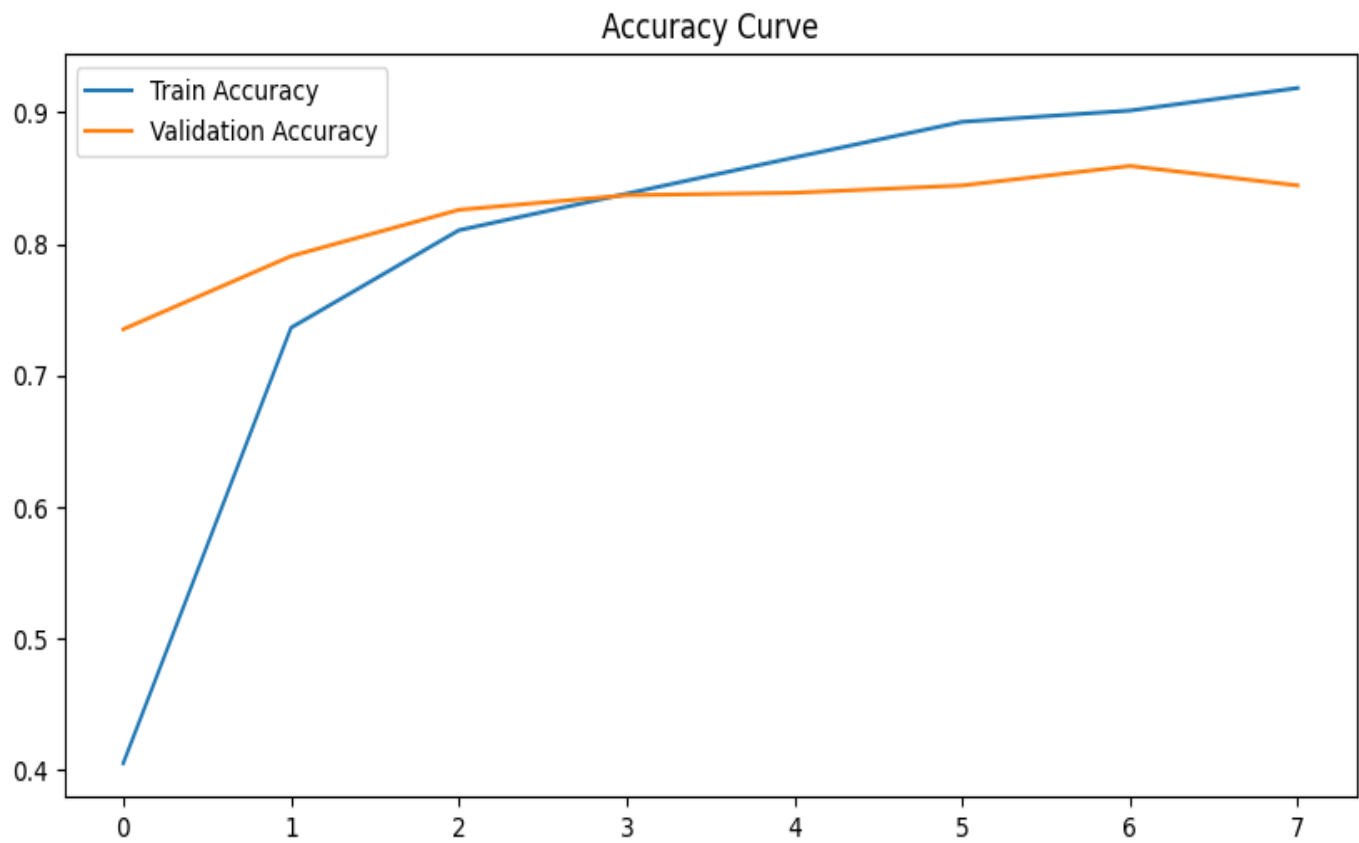
06

## Evaluation & Prediction

Evaluated using Confusion Matrix, Classification Report, Accuracy & Loss Curves. The system predicts species and confidence from uploaded images.

# Results and Performance

The model was trained for 8 epochs, followed by 5 fine-tuning epochs, achieving stable accuracy on both training and validation splits.



## Accuracy Curve

Displayed an upward trend, indicating good generalization.

**Confusion Matrix**  
Visualized correctly classified vs. misclassified samples across all 90 classes.

## Loss Curve

Showed a steady decrease, confirming effective learning.

**Classification Report**  
Provided detailed metrics: Precision, Recall, and F1-Score.

**Misclassification Report**  
Displayed wrongly classified images for further analysis and improvement.

Overall, MobileNetV2 proved effective for large multiclass datasets, though dataset imbalance influenced performance. The model is suitable for real-world deployment after TFLite conversion.



# Conclusion & Future Scope

This mini-project successfully implemented a deep-learning-based animal detection system, identifying 90 species with good accuracy.

## Efficient Transfer Learning

Achieved fast training and inference with MobileNetV2.



## Reliable Predictions

Provided accurate predictions with confidence scores.

## Comprehensive Evaluation

Successfully handled a complex, unbalanced 90-class dataset.

# Future Enhancements

- Implement YOLOv8 for object detection (bounding boxes).
- Improve accuracy by balancing the dataset.
- Deploy as a mobile app using TensorFlow Lite.
- Integrate Grad-CAM for heatmap visualization.
- Develop a Streamlit/Flask web application for broader access.