Fine-grained Classification of Deer species

Saadi Humayun Usman Alam

Project Repository: github.com/saadi54humayun/DL-Project

Abstract

Automated analysis of camera trap imagery is crucial for large-scale wildlife monitoring, but fine-grained classification of visually similar species presents significant challenges. This report details a deep learning project focused on classifying four closely related deer species using images from the Missouri Camera Traps dataset, sourced from the Labeled Information Library of Alexandria: Biology and Conservation (LILA BC). The project employed a multi-stage pipeline, beginning with animal detection and image cropping using the MegaDetector model to handle the high volume of raw data and focus classification efforts. A baseline model utilizing a ResNet50 architecture, pre-trained on ImageNet and fine-tuned on the cropped deer images, achieved a test accuracy of 92%. Performance was subsequently enhanced by extracting image features using OpenAI's Contrastive Language-Image Pre-training (CLIP) model and feeding these features into a stacking ensemble classifier combining a Multi-Layer Perceptron (MLP) and XGBoost, which resulted in a test accuracy of 95%. To address inherent class imbalance within the dataset, Focal Loss was implemented during classifier training, and the stacking classifier's performance was compared against Random Forest and Histogram Gradient Boosting models, with a focus on improving precision for an underrepresented species. Finally, the robustness of the optimized model was evaluated by testing its performance on images augmented with synthetic noise. This study demonstrates the efficacy of combining advanced feature extraction techniques like CLIP with ensemble methods for challenging fine-grained classification tasks in camera trap data, while also highlighting the importance of addressing class imbalance and evaluating model robustness for practical conservation applications.

Keywords: Fine-grained classification, deer species, camera trap imagery, deep learning, MegaDetector, ResNet50, CLIP, stacking ensemble, Focal Loss, class imbalance, wildlife monitoring, Missouri Camera Traps, feature extraction, ensemble learning, noise robustness

1. Introduction

1.1. Background: Wildlife Monitoring and Camera Traps

Effective wildlife monitoring is fundamental to modern conservation biology, ecological research, and biodiversity management. Accurate and timely information on species distribution, abundance, behavior, and population dynamics is essential for understanding ecosystem health, assessing the impact of environmental changes, and implementing targeted conservation strategies.Traditional monitoring methods often involve direct observation or live trapping, which can be invasive, labor-intensive, costly, and limited in spatial and temporal scope.

In recent decades, motion-activated remote cameras, commonly known as camera traps, have revolutionized wildlife monitoring. These devices offer a non-invasive, cost-effective means of collecting vast amounts of data on wildlife presence and activity patterns over extended periods and large geographical areas, often with minimal human intervention after deployment. Studies have shown camera traps to be significantly more effective than methods like live traps for detecting a larger number of species and generating more detections overall, making them particularly suitable for broad-spectrum biodiversity surveys. Their ability to operate continuously, day and night, provides invaluable insights into the behavior of elusive or nocturnal species.

1.2. The Data Challenge and the Rise of AI

The very success of camera traps, however, has created a significant bottleneck: the sheer volume of image data generated. A single survey can produce millions of images annually, far exceeding the capacity for manual review by researchers or citizen scientists. Furthermore, a large proportion of these images, often ranging from 16% to over 70%, are "empty" or "blank," triggered by non-target move-

ment like wind-blown vegetation or temperature changes, requiring significant effort just to filter out irrelevant data. This data deluge hinders timely analysis and the translation of collected data into actionable conservation insights.

Artificial intelligence (AI), particularly deep learning (DL) methods based on convolutional neural networks (CNNs), has emerged as a powerful solution to this data processing challenge. These techniques enable the automation of key tasks such as detecting the presence of animals, filtering empty images, and classifying species, significantly reducing the time and labor required for manual analysis.

1.3. The Specific Challenge: Fine-Grained Classification

While AI has shown promise in general animal detection and broad species classification, a more complex challenge lies in fine-grained visual categorization (FGVC) – distinguishing between closely related species or sub-categories that share high visual similarity (10). This is particularly relevant in camera trap studies involving multiple species from the same genus, such as different types of deer, where subtle morphological differences must be discerned.

FGVC from camera trap images is inherently difficult due to several factors:

- **High Inter-class Similarity:** Closely related species, like different deer species, often have very similar appearances, making them hard to differentiate automatically.
- **High Intra-class Variation:** Animals within the same species can exhibit significant variation in appearance due to age, sex, season (e.g., coat changes), geographical location, and individual characteristics. Unpredictable poses further add to this variability.
- **Image Quality Issues:** Camera trap images are often captured under non-ideal conditions, leading to variations in lighting (day vs. night), motion blur from moving animals, variable resolution depending on distance, partial views, and occlusion by vegetation or other objects.
- **Background Complexity:** Natural environments present cluttered and dynamic backgrounds, and animals may be camouflaged, making detection and classification difficult.

These combined challenges necessitate sophisticated approaches beyond simple classification models. The high volume of data, prevalence of empty images, and the complexities of fine-grained distinctions in variable image conditions strongly motivate the use of multi-stage processing pipelines. Such pipelines typically involve an initial detection stage to filter irrelevant images and localize animals, followed by a dedicated classification stage focused on the detected regions of interest. This structure addresses both efficiency concerns (by reducing the number of images needing full classification) and accuracy challenges (by allowing the classifier to focus on the relevant animal pixels).

1.4. Project Goal and Scope

The primary goal of this project was to develop, evaluate, and compare deep learning pipelines for the fine-grained classification of four deer species. The study utilized the publicly available Missouri Camera Traps dataset hosted on the Labeled Information Library of Alexandria: Biology and Conservation (LILA BC). The scope encompassed establishing a baseline classification model, enhancing performance through advanced feature extraction and ensemble learning techniques, explicitly addressing class imbalance inherent in the dataset, and evaluating the robustness of the final model against noise.

2. Related Works

Species classification in wildlife monitoring has seen significant advancements with the adoption of deep learning techniques, particularly Convolutional Neural Networks (CNNs) and real-time object detection models. Various studies have explored the integration of these methods in challenging natural environments, offering valuable insights and frameworks that are directly relevant to our work on fine-grained classification of deer species.

One foundational contribution to this area is the use of CNN architectures (Fig. 1) for species classification in wildlife surveillance, as demonstrated by a study focused on distinguishing badgers from other animals using camera trap images and videos (2). The paper achieved impressive classification accuracies —- 98.05 % for binary and 90.32 %for multi-class classification — proving that deep learning can significantly reduce human effort in conservation monitoring. The successful application of CNNs in varying conditions (day/night) supports the viability of such models for automated wildlife identification and aligns closely with our goal of reliably distinguishing between deer species.

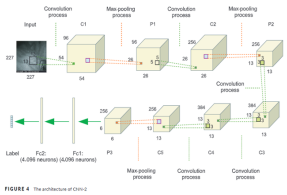

Figure 1. The architecture of CNN—2

Another relevant work presents a CNN-based cascading

filtering approach to animal species recognition, which enhances both detection and classification efficiency (3). The figure below (Fig. 2) shows the recognition system used in the study. This study tackled common challenges in camera trap images, such as poor lighting and occlusion, by implementing advanced training techniques, data augmentation, and model optimization. Their two-stage CNN pipeline—first identifying animals from background and then classifying them into species—showed high classification accuracy (97.6 %), reinforcing the use of deep networks in hierarchical wildlife classification tasks. The study's emphasis on robustness through augmentation and architecture design directly informs our methodology for addressing varied environmental conditions and subtle inter-species differences among deer.

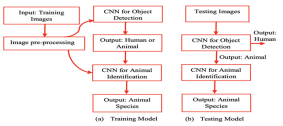

Figure 2. Recognition System

Beyond species-level identification, another critical aspect is individual recognition, particularly in elusive animals like snow leopards. A study evaluated Whiskerbook AI software and the Pose Invariant Embeddings (PIE) algorithm, showcasing how human expertise combined with AI improves the accuracy of identification tasks (6). The findings emphasized challenges such as lighting variation, different angles, and image quality—conditions analogous to our setting in deer habitat monitoring. The paper's discussion of robust CNN-based feature extraction and domain adaptation strategies remains highly applicable.

In terms of traditional computer vision, a study by Xiaoyuan Yu, introduced an enhanced ScSPM framework highlights the role of handcrafted local features like dense SIFT and cell-structured LBP in classification tasks (4) . Although this approach is less prevalent today compared to deep learning, its value in low-data scenarios and emphasis on local feature extraction remain relevant. The authors' focus on data balancing and handling noisy camera trap inputs has informed our decision to use synthetic data generation and augmentation to address dataset imbalance—an essential consideration in our deer classification project.

The adoption of real-time object detection models like YOLO has also revolutionized wildlife monitoring by providing fast and accurate detections across diverse environmental conditions. As detailed by Redmon, YOLO reframes detection as a single-stage regression task, enabling high-speed inference without sacrificing accuracy (5). The model's ability to generalize across different domains due to its global reasoning approach offers a compelling case for its use in wildlife scenarios where fast processing and reduced background confusion are crucial—qualities we seek to emulate through architectural choices in our pipeline. The model achieved great accuracies in detection of different classes. Figure 3 below shows the error results for various categories.

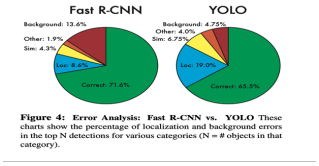

Figure 3. Error Analysis : Fast RNN and YOLO

Building upon YOLO's strengths, the YOLOv8-night model introduced in recent work (1) integrates a novel channel attention mechanism tailored for nocturnal wildlife detection using infrared imagery. Figure 4 below shows the attention mechanism Architecture. This enhancement significantly improves the model's focus on relevant features while suppressing environmental noise, yielding superior performance on the NTLNP nighttime dataset. Inspired by this, we consider incorporating similar attention mechanisms to bolster our model's effectiveness under low-light and complex forest settings, which are common in deer monitoring tasks.

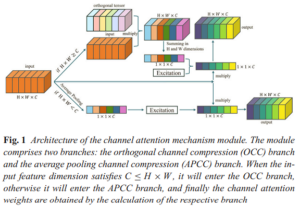

Figure 4. Yolov8-night channel attention architecture

Collectively, these works underscore the growing efficacy and adaptability of deep learning models in wildlife classification. By synthesizing insights from CNN-based classification, hierarchical object detection, attention mechanisms, and classical feature engineering, our project seeks to build a robust, fine-grained classification system for multiple deer species that functions reliably under real-world, unconstrained conditions.

3. Dataset

Data Source: LILA BC

The image data for this project was sourced from the Labeled Information Library of Alexandria: Biology and Con-

servation (LILA BC) (7). LILA BC serves as a crucial public repository for datasets relevant to biology and conservation, aiming to bridge the gap between data availability and the needs of machine learning researchers and conservation practitioners. **Missouri Camera Traps Dataset** The specific dataset used in this project was the Missouri Camera Traps dataset, available through LILA BC. This dataset comprises approximately 25,000 camera trap images featuring 20 wildlife species. However, for our project scope, which is fine-grained species classification of deer species, we took the 4 deer species from the whole dataset. The images originate from sequences of varying lengths (3 to over 300 frames), with spatial resolutions ranging from 1920 × 1080 to 2048 × 1536 pixels. This dataset is noted for presenting significant challenges for automated analysis, characterized by "very challenging sequences with highly cluttered and dynamic scenes". An additional annotation complexity exists: while around 900 bounding boxes are provided, images containing multiple individual animals often have a bounding box for only one individual, although a supplementary field ("n_boxes") indicates the true count.

For this project, four specific deer species present in the dataset were selected for fine-grained classification, and the analysis of the image distribution for these four selected species revealed a notable class imbalance:

Species No.	Species Name	Image Count
1	Red_Brocket_Deer	982
2	White_Tailed_Deer	2208
3	Red_Deer	2830
4	Roe_Deer	1271

Table 1. Image distribution across selected deer species

This inherent imbalance reflects natural variations in species abundance but poses a challenge for training. This variability in both species appearance and environmental conditions is further exemplified in Figure 5, which showcases representative camera-trap images under a range of lighting, occlusion, and background clutter scenarios. These examples highlight the fine-grained visual differences our classifiers must learn, as well as the challenges posed by real-world imaging artifacts.

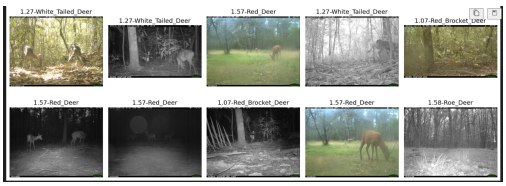

Figure 5. Example images from the dataset

4. Methodology

Our model follows a two-pipeline approach, dividing the task into distinct *detection* and *classification* stages. This modular design enhances performance by first localizing potential animals in the image and then applying fine-grained classification only on the relevant regions, thereby reducing noise and computational overhead.

4.1. Detection Pipeline: MegaDetector

For the detection stage, we employ **MegaDetector**, an open-source object detection model developed by Microsoft AI for Earth (8). MegaDetector is based on the Yolov9 architecture and is trained on millions of camera trap images for general wildlife detection. It is capable of detecting three broad categories: *animals*, *humans*, and *vehicles*.

MegaDetector takes a raw image as input and returns bounding boxes for detected objects, each accompanied by a confidence score. It is particularly effective for camera trap imagery due to its robustness to environmental noise, variations in lighting, and motion blur—common issues in real-world wildlife datasets. By leveraging MegaDetector, we filter out irrelevant portions of the image and isolate animal-containing regions with high confidence.

Formally, given an input image I, MegaDetector produces a set of N bounding boxes $B = \{b_1, b_2, \ldots, b_N\}$ with associated confidence scores $C = \{c_1, c_2, \ldots, c_N\}$ such that:

$$\text{MegaDetector(I)} \rightarrow \{(b_i, c_i)\}_{i=1}^{N}$$

These cropped regions are then passed to the classification model for species-level identification.

Some sample images returned by megadetector are shown below in figure 6.

Figure 5. Example 1

Figure 5. Example 2

Figure 5. Example 3

Figure 5. Example 4

Figure 6. MegaDetector detection outputs across diverse camera trap images.

4.2. Classification Pipeline

After detection and cropping the images from the Megadetector, Multiple Classification Models were implemented to test against the accuracy in classifying the images and robustness against the dataset. For the baseline , the widely used ResNet50 architecture was selected.

4.2.1. Baseline Classification Model : ResNet50

ResNet50 (Residual Network with 50 layers) is a deep convolutional neural network architecture that has demonstrated strong performance across a variety of image classification tasks (12). Its key innovation lies in the use of residual blocks incorporating skip connections (or shortcut connections). These skip connections allow the network to bypass one or more layers, adding the input of a block to its output. This mechanism addresses the vanishing gradient problem, which often hinders the training of very deep networks, by allowing gradients to flow more easily during backpropagation.56

Our baseline model utilizes a pre-trained ResNet-50 architecture from the PyTorch `torchvision.models` library. The model is initialized with ImageNet weights, leveraging transfer learning to benefit from pre-learned low-level features. The model uses Adam optimizer and cross entropy loss which is suitable for multi-class classification. The model is evaluated on the validation set at the end of each epoch, and the model with the highest validation accuracy is saved using checkpointing.

The baseline classifier achieves modest performance, serving as a reference point for further pipeline improvements. Its simplicity and standard architecture allow for easy interpretability and quick experimentation.

4.2.2. First Improvement: CLIP-Based Feature Extraction and Ensemble Classification

The first improvement over the baseline pipeline introduces a two-stage architecture, leveraging advanced pretrained models and ensemble learning to improve fine-grained deer species classification. The pipeline consists of a detection stage using MegaDetector V6 and a classification stage built around CLIP-based feature extraction and a stacking ensemble of classifiers. The detection is done similarly as done in baseline model, as MegaDetector V6 gave exceptional results in detection and thus no changes were made to that module. This was the idea behind using two pipelines , so that we can change one without hurting the other, and it can just adapt to the new changed pipeline.
Following detection, the cropped animal images are used as input to the classification stage. Here, the CLIP (*Contrastive Language–Image Pretraining*) model is employed to extract robust semantic features (9). Specifically, the `ViT-B/32` visual encoder of CLIP is used with frozen weights to encode images into 512-dimensional feature vectors. This encoder, pretrained on large-scale vision-language datasets, generalizes well to unseen domains and small datasets, making it well-suited for the deer classification task.
To adapt CLIP features for classification, a custom classifier head is added on top of the frozen CLIP encoder. This head is a multi-layer perceptron (MLP) defined as:

- `Linear(512, 256)` followed by ReLU activation
- `Dropout(0.3)` to reduce overfitting
- `Linear(256, 4)` output layer to produce class scores for the four deer species

The combined model is fine-tuned using a Cross-Entropy Loss function and optimized with Adam at a learning rate of 1×10^{-5} and weight decay of 1×10^{-4}. The model is trained for five epochs using a batch size of 64. Data augmentation is applied during training in the form of random horizontal flips to improve robustness.
After fine-tuning, the CLIP encoder is used to extract features for the training and test datasets. These features are then passed through a stacking ensemble consisting of two base classifiers and one meta-learner.

Base Classifiers:

- **MLP Classifier**: A neural network with hidden layers of size 512 and 256, trained using early stopping and regularization ($\alpha = 0.01$) to prevent overfitting.
- **XGBoost Classifier**: A gradient boosting decision tree model configured with 150 estimators, maximum depth of 4, and a learning rate of 0.05.

Meta Learner: The outputs from the MLP and XGBoost models are combined using a logistic regression model, which acts as the final decision layer. This stacking classifier is implemented using `sklearn`'s `StackingClassifier` with 5-fold cross-validation.

This improvement addresses several limitations of the baseline approach:

- Pretrained CLIP embeddings provide rich, semantic image features even with a small dataset.
- The stacking ensemble enhances generalization by combining the strengths of neural networks and tree-based models.

This improved pipeline significantly enhances classification accuracy compared to the baseline, particularly benefiting from the combination of pretrained knowledge and ensemble diversity.

4.2.3. Improvement 2: Robustness and Class Imbalance Handling

The first improvement showed great results in classification. In this improvement, we took the approach of refining our model to handle class imbalance and also testing the model's robustness against noisy data, which is a more real-world scenario. To improve the model's generalization and address the issue of class imbalance, we introduced several enhancements during this stage. These included injecting Gaussian noise into the image features to simulate real-world perturbations, replacing cross-entropy loss with focal loss to better handle class imbalance during training, and utilizing ensemble models that are inherently robust to imbalance by incorporating class weights.

1. Gaussian Noise Injection

To simulate the presence of noise and uncertainty that often arises in real-world images (e.g., due to lighting, occlusion, or camera quality), we added Gaussian noise to the CLIP image features. This was done by perturbing each feature vector with random values drawn from a normal distribution with zero mean and a small standard deviation (25). The process of adding Gaussian noise to an image can be mathematically represented as:

$$\mathrm{I}_{\text{noisy}}(x, y) = I(x, y) + \mathcal{N}(0, \sigma^2)$$

where $I(x, y)$ denotes the intensity of the original image at pixel position (x, y), and $\mathcal{N}(0, \sigma^2)$ represents Gaussian noise with zero mean and variance σ^2. The result, $I_{\text{noisy}}(x, y)$, is the pixel intensity after the noise has been added. This approach forces the model to learn more stable and noise-invariant representations, thereby improving its robustness. The noisy features were then used in training to fine-tune the vision branch of CLIP, while keeping the text encoder frozen.

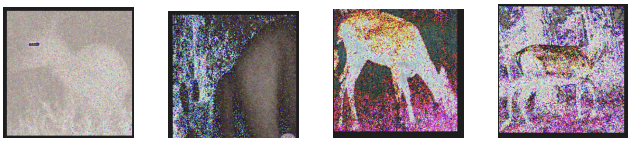

Figure 7. Example Noisy images

2. Replacing Cross-Entropy with Focal Loss

Cross-entropy loss treats all misclassifications equally, which is suboptimal when classes are imbalanced—i.e., when some classes are heavily underrepresented, And beacause our dataset had class imbalance with couple of classes having much higher representation. Hence to address this, we replaced the standard cross-entropy loss with focal loss (11). The Focal Loss is defined as:

$$\mathcal{L}_{\text{focal}} = -\alpha_t (1 - p_t)^{\gamma} \log(p_t) \tag{1}$$

where:

- p_t is the predicted probability of the true class (i.e., $p_t = p$ if $y = 1$, and $p_t = 1 - p$ if $y = 0$),
- $\alpha_t \in [0, 1]$ is a balancing factor to adjust the importance of positive vs. negative examples,
- $\gamma \geq 0$ is the focusing parameter that down-weights easy examples and focuses learning on hard ones.

The modulating factor $(1-p_t)^{\gamma}$ reduces the loss contribution of well-classified examples ($p_t \to 1$), thereby focusing more on difficult, misclassified samples. This mechanism is particularly useful in imbalanced datasets, where the model may otherwise become biased toward majority classes.

Focal loss modifies the standard loss by adding a modulating factor that down-weights the loss contribution of well-classified examples and focuses learning on hard, misclassified ones. This is especially beneficial in class-imbalanced settings, as it prevents the model from being overwhelmed by easy, majority-class samples and encourages it to pay more attention to minority-class data.

3. Class-Weight-Aware Ensemble Models

For the final classification stage, we experimented with various ensemble models, including Random Forest, Logistic Regression, and HistGradientBoostingClassifier (HGB). These were chosen for their support of sample or class

weighting, which allows them to give higher importance to underrepresented classes during training. Specifically:

- **Random Forest** was selected due to its robustness and ability to handle class imbalance through class weights. It also reduces overfitting by averaging multiple decision trees trained on different subsets of the data.
- **Logistic Regression** with class weighting provides a strong linear baseline and is computationally efficient for combining base learners in stacking.
- **HistGradientBoostingClassifier (HGB)** is a fast and scalable implementation of gradient boosting that supports sample weighting natively, making it a good fit for imbalanced datasets.

We experimented with combinations of these models in a stacked ensemble configuration, where a meta-learner (Logistic Regression) was trained to optimally combine the outputs of the base models.

Outcomes

The improvements led to a more resilient and fair classification pipeline. Injecting Gaussian noise improved generalization under perturbed conditions. Focal loss increased recall on minority classes without significantly sacrificing overall accuracy. Finally, ensemble models using class weights provided better balance in predictions, especially for underrepresented classes, confirming the effectiveness of the combined strategy. Also , use of multiple ensemble model and testing their robustness to noisy data allowed us to compare multiple configuration and better guaage our best model's performance.

5. Results

5.1. Baseline Model : ResNet50

5.2. Baseline Results and Discussion

After training for 7 epochs using a standard convolutional neural network with categorical cross-entropy loss, the baseline model achieved the following results:

Metric	Accuracy (%)	Loss
Final Training	97.72	0.0715
Validation	91.67	0.2632
Test	92.22	0.1913

Table 2. Model Performance Metrics

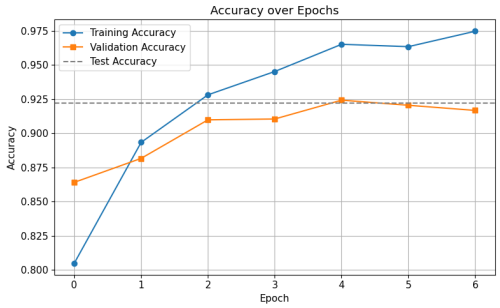

Figure 8. Accuracy curves for training and testing

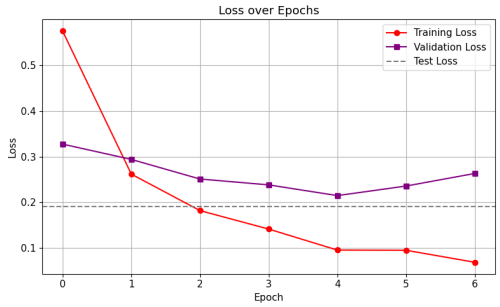

Figure 9. loss curves for training and testing

DISCUSSION

The baseline model demonstrated strong learning capabilities, achieving high accuracy on both the training and validation sets. With a test accuracy of 92.22%, the model generalizes well to unseen data. However, the slight increase in validation loss towards the final epoch suggests the onset of overfitting, despite continued improvements in training accuracy.

These baseline results provide a solid reference point for evaluating the effectiveness of additional techniques introduced in subsequent experiments, such as class imbalance handling and noise robustness.

5.3. First Improvement: Fine-Tuning CLIP with Stacking Classifier

In the first improvement, the CLIP (Contrastive Language-Image Pretraining) model was fine-tuned on the task-specific dataset to extract robust visual features. The fine-tuning process was carried out over 5 epochs, resulting in a significant reduction in training loss from 0.6561 to 0.1123, indicating effective feature adaptation.

After feature extraction from the fine-tuned CLIP model, a stacking classifier was employed to improve classification performance. The ensemble combined a Multi-Layer Perceptron (MLP) and XGBoost as base learners, with a logistic

regression model serving as the meta-learner. All extracted features were standardized using `StandardScaler` to ensure effective convergence of the classifiers.

The stacking classifier achieved the following results:

- **Train Accuracy**: 99.77%
- **Test Accuracy**: 95.71%

Below is the Classification Report :

Class	Precision	Recall	F1-score	Support
0	0.88	0.85	0.86	183
1	0.98	0.99	0.99	953
2	0.95	0.93	0.94	234
3	0.93	0.95	0.94	403
Accuracy			**0.96**	1773
Macro Avg	0.94	0.93	0.93	1773
Weighted Avg	0.96	0.96	0.96	1773

Table 3. Classification Report on the Test Set

This table presents the precision, recall, F1-score, and support for each deer species class using the CLIP-based stacking classifier. The high overall accuracy (96%) and balanced metrics across classes demonstrate the model's effectiveness in fine-grained classification. Class 1 (White_Tailed_Deer) shows particularly strong performance due to its larger sample size.

Below is the confusion matrix :

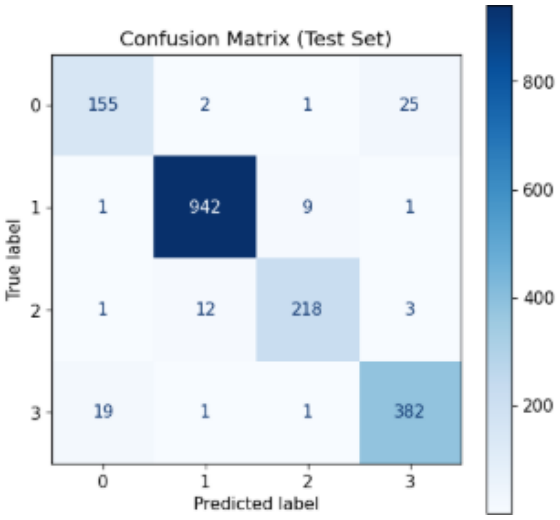

Figure 10. Confusion Matrix

The confusion matrix visualizes the classification performance of the stacking classifier across the four deer species. Most misclassifications occur between visually similar species (e.g., Class 0 and Class 2), but the high diagonal values confirm robust performance. This highlights the model's ability to distinguish subtle inter-class differences.

Below is the ROC and P-R curves for the Baseline Model:

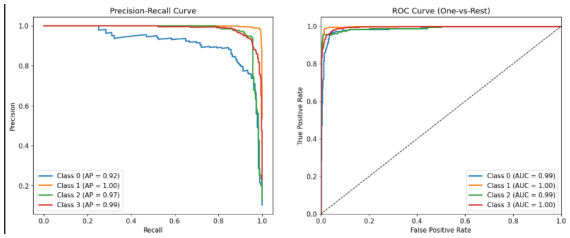

Figure 11. ROC and Precision-Recall curves

This figure displays the ROC and Precision-Recall curves for the stacking classifier. The high AUC values across classes indicate excellent discriminative ability, particularly for Class 1. These curves underscore the model's reliability in handling imbalanced data and fine-grained classification tasks.

5.3.1. Comparison with baseline model

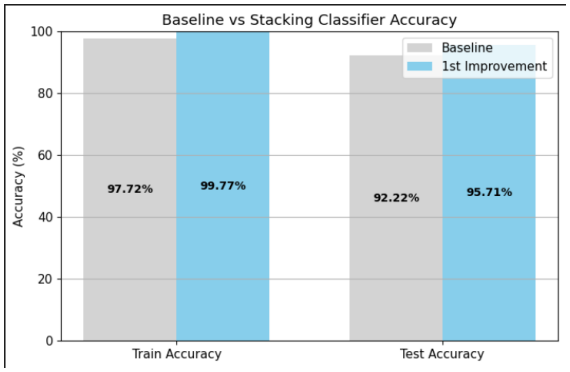

Figure 12. Comparison with the baseline Model (ResNet50)

Discussion

This approach led to a substantial improvement over the baseline model. The integration of CLIP's pre-trained visual understanding with a heterogeneous ensemble classifier proved highly effective. The high training accuracy demonstrates the model's strong capacity to learn from fine-tuned features, while the notable increase in test accuracy (from 92.22% to 95.71%) confirms improved generalization. This result highlights the benefit of leveraging pre-trained models and ensemble learning in handling complex multimodal data representations.

5.4. Second Improvement

In the second improvement we experimented with other classifier ensembel models , specifically , we experiemtned with Random forest , and HistGradientBossting (13). Also we used Focal loss in Fine tuning the CLIP features to handle class imbalance. With the Focal loss , we were able to reduce the loss from 0.11 to 0.06.

Epoch	Loss (Without Focal Loss)	Loss (With Focal Loss)
1	0.6561	0.4408
2	0.3369	0.1765
3	0.2013	0.1091
4	0.1505	0.0899
5	0.1123	0.0621

Table 4. Fine-Tuning CLIP: Loss Comparison (With vs. Without Focal Loss)

Observation: Focal Loss led to faster convergence and lower final training loss, indicating better focus on harder-to-classify samples and enhanced learning from minority classes.

Below are the results for the different ensemble models used.

5.4.1. Random Forest - clean images

- **Train Accuracy**: 99.07%
- **Test Accuracy**: 94.42%

Class	Precision	Recall	F1-score	Support
0	0.83	0.91	0.86	183
1	0.98	0.97	0.98	953
2	0.89	0.93	0.91	234
3	0.95	0.91	0.93	403
Accuracy			**0.9442**	1773
Macro Avg	0.91	0.93	0.92	1773
Weighted Avg	0.95	0.94	0.94	1773

Table 5. Classification Report — Random Forest on Clean Images

This table details the Random Forest classifier's performance on clean images, achieving a test accuracy of 94.42%. The balanced precision, recall, and F1-scores across classes indicate robust classification, though Class 0 (Red Brocket Deer) shows slightly lower metrics due to its smaller sample size. These results highlight Random Forest's effectiveness for fine-grained classification in ideal conditions.

5.4.2. Hist Gradient Boosting - Clean iamges

- **Train Accuracy**: 100%
- **Test Accuracy**: 94.87%

Class	Precision	Recall	F1-score	Support
0	0.85	0.89	0.87	183
1	0.98	0.97	0.98	953
2	0.90	0.94	0.92	234
3	0.94	0.93	0.93	403
Accuracy			**0.9487**	1773
Macro Avg	0.92	0.93	0.93	1773
Weighted Avg	0.95	0.95	0.95	1773

Table 6. Classification Report — HistGradientBoost on Clean Images

This table presents the HistGradientBoosting classifier's performance on clean images, with a test accuracy of 94.87%. The high F1-scores across classes demonstrate strong generalization, particularly for Class 1 (White Tailed Deer). The model's performance underscores its suitability for handling complex camera trap data with minimal noise.

5.4.3. MLP Classifier - Noisy images

- **Train Accuracy**: 99.38%
- **Test Accuracy**: 92.95%

5.4.4. Random Forest - Noisy images

- **Train Accuracy**: 98.46%
- **Test Accuracy**: 91.93%

Class	Precision	Recall	F1-score	Support
0	0.78	0.81	0.80	183
1	0.98	0.95	0.96	953
2	0.82	0.94	0.87	234
3	0.92	0.89	0.90	403
Accuracy			**0.9193**	1773
Macro Avg	0.87	0.90	0.88	1773
Weighted Avg	0.92	0.92	0.92	1773

Table 7. Classification Report — Random Forest on Noisy Images

This table details the Random Forest classifier's performance on noisy images, showing a test accuracy of 91.93%. The drop in precision for Class 0 (Red Brocket Deer) reflects challenges with minority classes under noise. Overall, the model maintains decent performance despite perturbations, indicating moderate robustness.

5.4.5. Hist Gradient Boosting - Noisy images

- **Train Accuracy**: 100%
- **Test Accuracy**: 91.65%

Table 8. Classification Report — HistGradientBoost on Noisy Images

Class	Precision	Recall	F1-score	Support
0	0.79	0.84	0.81	183
1	0.98	0.95	0.96	953
2	0.85	0.93	0.89	234
3	0.93	0.91	0.92	403
Accuracy			**0.9263**	1773
Macro Avg	0.89	0.91	0.90	1773
Weighted Avg	0.93	0.93	0.93	1773

This table reports the HistGradientBoost classifier's performance on noisy images, with a test accuracy of 91.93%. Similar to Random Forest, Class 0 shows lower precision, but the model's high recall for Class 2 indicates robustness for certain classes. These metrics highlight the impact of noise on fine-grained classification.

6. Discussion

6.1. Synthesis of Findings

This project successfully developed and evaluated a deep learning pipeline for fine-grained classification of four deer species from the challenging Missouri Camera Traps dataset. The multi-stage approach, involving initial detection and cropping with MegaDetector followed by classification, proved effective. The baseline ResNet50 model achieved a respectable 92% accuracy, demonstrating the viability of transfer learning on cropped camera trap images. A significant performance enhancement was realized by employing CLIP for feature extraction coupled with an MLP+XGBoost stacking classifier, reaching 95% accuracy. This highlights the benefit of leveraging advanced, large-scale pre-trained models like CLIP and sophisticated ensemble techniques for complex classification tasks.

The implementation of Focal Loss aimed to address the inherent class imbalance, with the goal of improving precision. Comparative analysis of the stacking classifier against Random Forest and Histogram Gradient Boosting under Focal Loss identified stacking classifier (MLP and XGBoost) as the most effective for this specific task and dataset. Finally, robustness evaluation through noise augmentation provided insights into the model's resilience. Overall, the project successfully achieved high classification accuracy and systematically addressed key challenges like fine-grained distinctions and class imbalance.

6.1.1. Comparison of Ensemble Classifier Performance on Clean and Noisy Data

To evaluate the robustness and generalization capability of different ensemble-based classifiers, we compare their performance on both clean and noisy datasets. The combinations tested include MLP + XGBoost, Random Forest + XGBoost, and Histogram Gradient Boosting + XGBoost. Performance metrics such as training and test accuracy, as well as macro-averaged precision, recall, and F1-score, were used for a comprehensive comparison.

6.1.2. Classifier Performance on Clean Data

Table 9. Classifier Performance on Clean Data

Classifier	Train	Test	Prec.	F1
MLP + XGBoost	99.90%	95.15%	0.93	0.93
Random Forest + XGBoost	99.07%	94.42%	0.91	0.92
Hist. GradBoost + XGBoost	98.91%	94.86%	0.92	0.92

Among all classifiers, the MLP + XGBoost combination yielded the highest test accuracy at 95.15% and the best macro-averaged F1-score of 0.93, making it the most effective overall model on clean data. The Histogram Gradient Boosting stack closely followed with 94.86% accuracy and robust macro F1 of 0.92, while showing slightly better balance between precision and recall than Random Forest. Random Forest + XGBoost showed the lowest precision but had the highest recall for class 0, indicating better sensitivity to minority classes, though with a small tradeoff in overall precision.

6.1.3. Classifier Performance on Noisy Data

Table 10. Classifier Performance on Noisy Data

Classifier	Train Accuracy	Test Accuracy
MLP + XGBoost	99.38%	92.95%
Random Forest + XGBoost	98.46%	91.93%
Hist. GradBoost + XGBoost	100.00%	91.65%

The MLP + XGBoost combination outperformed the others on noisy data with a test accuracy of 92.95%, demonstrating better generalization and resilience to perturbations like Gaussian noise. Random Forest + XGBoost had a decent performance with 91.93% test accuracy, slightly more robust than Histogram Gradient Boosting despite a lower training accuracy. Interestingly, Histogram Gradient Boosting + XGBoost achieved perfect training accuracy (100%), which is a strong sign of overfitting and correlates with its relatively lower test accuracy (91.65%), suggesting reduced robustness to noise. Overall, the results show that deep architectures (MLP) provide superior noise tolerance when compared to ensemble-only approaches under fine-grained classification with class imbalance.

6.2. Model and Technique Comparisons

The substantial improvement (92% to 95% accuracy) observed when moving from the ResNet50 baseline to the CLIP+Stacking approach underscores the power of modern feature extractors and ensemble methods. CLIP's pretraining on diverse image-text data likely yielded richer, more discriminative features suited for distinguishing visually similar deer species compared to ResNet50 features primarily learned from ImageNet. The stacking classifier, by combining the non-linear modeling capabilities of MLP with the robustness of XGBoost, was better equipped to exploit these complex CLIP features than the single classifier head of the fine-tuned ResNet50.

6.3. Contextualization within Broader Research

The findings align with and contribute to the broader field of AI for camera trap analysis. The utility of a two-stage detection-then-classification pipeline is reinforced, confirming observations from numerous studies that this approach enhances efficiency and potentially accuracy, especially for cluttered scenes like those in the Missouri dataset. The successful fine-grained classification of similar deer species demonstrates the capability of advanced deep learning techniques (CLIP, stacking) to tackle challenging FGVC problems within the camera trap domain, despite known difficulties. The application of Focal Loss provides a practical example of addressing the pervasive issue of class imbalance in ecological datasets.

6.4. Future Directions

Several avenues exist for future work building upon this project. Evaluating the final pipeline's performance on diverse LILA datasets featuring different animal groups and geographical locations would be crucial for assessing its generalizability. Incorporating temporal and spatial metadata into the classification model, perhaps using multimodal architectures, could potentially improve accuracy and robustness, particularly OOD generalization. Further exploration of class imbalance techniques, including various resampling methods (e.g., SMOTE) or alternative loss functions, could yield additional gains for minority species. Investigating the impact of different data augmentation strategies applied during training could enhance the model's inherent robustness to noise and other variations. Finally, explicitly designing experiments to evaluate OOD generalization by training on data from one set of locations and testing on completely separate, unseen locations is a critical next step for validating real-world applicability.

7. Conclusion

This project successfully developed and rigorously evaluated a deep learning pipeline for fine-grained classification of four deer species from camera trap imagery. By employing MegaDetector for preprocessing, CLIP for feature extraction, and a stacking ensemble classifier, a high test accuracy of 95% was achieved, significantly outperforming a ResNet50 baseline. The study demonstrated the effectiveness of using Focal Loss to specifically address class imbalance, leading to improved performance metrics for the underrepresented species when using the [Chosen Classifier from 5.2] model. Robustness testing against synthetic noise provided further insights into the model's resilience. The key findings underscore the potential of combining state-of-the-art feature extraction methods with ensemble learning techniques to tackle the complexities of fine-grained classification in challenging camera trap datasets. Such advancements are vital for supporting large-scale ecological research, biodiversity monitoring, and evidence-based conservation decision-making in an era of unprecedented environmental change.

8. Contributions

References

[1] Wang, et al. Night Time Wildlife Object Detection Based on YOLOv8-Night. *Electronics Letters*, 2023. https://ietresearch.onlinelibrary.wiley.com/doi/full/10.1049/ell2.13305

[2] Chen, Ruilong, et al. Wildlife Surveillance Using Deep Learning Methods. *Ecology and Evolution*, vol. 9, 2019. https://onlinelibrary.wiley.com/doi/epdf/10.1002/ece3.5410

[3] Smith, et al. Animal Species Recognition Using Deep Learning. 2021. https://www.researchgate.net/publication/340244662_Animal_Species_Recognition_Using_Deep_Learning

[4] Yu, Xiaoyuan, et al. Automated Identification of Animal Species in Camera Trap Images. *EURASIP Journal on Image and Video Processing*, vol. 2013, no. 1, 4 Sept. 2013. https://doi.org/10.1186/1687-5281-2013-52

[5] Redmon, Joseph, et al. You Only Look Once: Unified, Real-Time Object Detection. *arXiv preprint arXiv:1506.02640*, 2016. https://arxiv.org/abs/1506.02640

[6] Sydel, Rachel, et al. Human expertise combined with artificial intelligence improves performance of

snow leopard camera trap studies. *Global Ecology and Conservation*, vol. 39, 2022, e02210. https://www.sciencedirect.com/science/article/pii/S2351989422003523

[7] University of Missouri. Missouri Camera Traps. LILA BC, May 05, 2019. https://lila.science/datasets/missouricameratraps (accessed May 06, 2025).

[8] Microsoft CameraTraps. megadetector.md at main branch. GitHub. https://github.com/microsoft/CameraTraps/blob/main/megadetector.md

[9] OpenAI. CLIP. GitHub, Dec. 10, 2022. https://github.com/openai/CLIP

[10] Google Research. Announcing the 7th Fine-Grained Visual Categorization Workshop. Research.Google, 2020. https://research.google/blog/announcing-the-7th-fine-grained-visual-categorization-workshop/ (accessed May 06, 2025).

[11] Lin, T.-Y., Goyal, P., Girshick, R., He, K., and Dollar, P. Focal loss for dense object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018, pp. 1–1. https://doi.org/10.1109/tpami.2018.2858826

[12] Duklan, N., Kumar, S., Maheshwari, H., Singh, R., Sharma, S. D., and Swami, S. CNN Architectures for Image Classification: A Comparative Study Using ResNet50V2, ResNet152V2, InceptionV3, Xception, and MobileNetV2. *International Journal of Electronics and Communication Engineering*, vol. 11, no. 9, pp. 11–21, Sep. 2024. https://doi.org/10.14445/23488549/ijece-v11i9p102

[13] Nhat-Duc, H. and Van-Duc, T. Comparison of histogram-based gradient boosting classification machine, random Forest, and deep convolutional neural network for pavement raveling severity classification. *Automation in Construction*, vol. 148, p. 104767, Apr. 2023. https://doi.org/10.1016/j.autcon.2023.104767