Colombian Wildlife Classification

Henry Gardner
henrygardner@berkeley.edu

Juliana Gómez Consuegra
julianagc@berkeley.edu

Jacqueline Lam
jacq@berkeley.edu

Eunice Ngai
eunicen@berkeley.edu

Abstract

This paper presents a computer vision approach to classifying Colombian wildlife through camera traps in the Orinoquia region. Utilizing a dataset originally consisting of over 100,000 images from 50 cameras, we aim to classify the following six animal species or genus: Peccary genus (collared and white lipped), Black Agouti, Spotted Paca, Dasypus Species, South American Coati, and Bos Species. Significant variation in class sizes, colored and monochrome images, animal movement, and depth variation presented difficult challenges that were addressed through various preprocessing and feature extraction techniques. We developed models using Linear Discriminant Analysis (LDA), Support Vector Machines (SVM), and Neural Networks (NN) to classify data into these six classes. The SVM model on the HOG feature vector demonstrated the highest performance, achieving 77% accuracy on the testing set. While there is potential for further improvement, these results represent a significant step towards automating a traditionally manual task contributing to wildlife preservation.

1. Introduction

Colombia's Orinoquia region (figure 1) is a biologically diverse area, home to a variety of wildlife species. Effective monitoring and classifying of these species is essential for conservation efforts, ecological studies, and biodiversity management. Traditional manual labeling and physical tracking can be labor/time intensive with inevitable errors. Advances in computer vision have provided new opportunities to automate these intensive tasks through the use of camera traps and machine learning. Our research presents a modern approach to this question with a series of implemented models that accurately classify six species present in this biologically diverse region. Using a variety of features and image processing techniques, we were able to leverage information not available through the human eye - providing us with an advantaged machine-built taxonomist.

Figure 1. Map of the Orinoquia region in Colombia. Source: (1)

2. Data

The data was sourced from camera traps and contains images from January to July 2020. The dataset contains more than 100,000 images collected from 50 camera traps, located in two natural reserves and labelled by wildlife experts (6). A total of 51 animal species are identified from the images, with approximately 20% of the images being empty, resulting in 52 classes. The class sizes vary significantly, ranging from as few as 1 to over 20,000 images. Initial Exploratory Data Analysis showed that 61 images

contain multiple animal species; these images were discarded for simplicity. The cameras were motion-activated, typically capturing snapshots of animal movement, which meant that our feature engineering task required handling motion blur, as well as sequences of images with the same animal. The images all had a resolution of 1440 by 1920, and where either RGB or infrared, depending on the time of day when the image was taken. They also included a banner with metadata at the bottom of the image. Due to the large volume of high-resolution images, the dataset was hosted on a GCS instance to facilitate easy retrieval and modification.

Out of the 52 classes, we decided to focus on classifying the following individual species: Black Agouti (*Dasyprocta fuliginosa*), Spotted Paca (*Cuniculus paca*), South American Coati (*Nasua nasua*), the two genuses: Dasypus Species (*Dasypus sp.*) and Bos Species (*Bos sp.*), and an artificial group, the peccaries (*Tayassu pecari* and *Pecari tajacu*) which we grouped under a single class - Peccaries. We decided to group the peccaries under a single class as they are very similar, visually, so we imagined it would be hard for the classifier to tell them apart.

3. Data Preprocessing

We divided the data into train (70%), validation (10%), and test (20%) sets. Images from the same sequence of movement were kept in the same split to prevent data leakage. To address the high variability in class sizes, we downsampled the dataset for greater uniformity. The resulting distribution sets are shown in figure 2.

Subsequent preprocessing steps included removing the metadata, leaving only the red color channel (in order to standardize daytime and nighttime images) by turning the blue and green channels to zeros, reducing the resolution to 0.1x, and applying Contrast Limited Adaptive Histogram Equalization (CLAHE, (7)) of grid size 8x8 to adjust the pixel distribution of the images. CLAHE is a variant of Adaptive Histogram Equalization (AHE) which provides better contrast in local areas than that achievable utilizing traditional histogram equalization methods as it utilizes a local contextual region instead of the entire image. Examples of each image class, before and after preprocessing are shown in figure 3.

4. Feature Extraction

Due to our high-dimensional image data, feature extraction is a crucial step to enhance our interpretability of each class required for an effective computer vision model. Given that our classes had complex shapes, we focused on complex features, but also included some simple features, as a baseline.

```
no. of images in train set:

common_name
Black Agouti            1074
Bos Species              976
Dasypus Species         1074
Peccary                 1075
South American Coati     981
Spotted Paca            1074
Name: image_id, dtype: int64
------------------------
no. of images in validation set:

common_name
Black Agouti             156
Bos Species              134
Dasypus Species          155
Peccary                  156
South American Coati     136
Spotted Paca             154
Name: image_id, dtype: int64
------------------------
no. of images in test set:

common_name
Black Agouti             307
Bos Species              282
Dasypus Species          309
Peccary                  307
South American Coati     281
Spotted Paca             308
Name: image_id, dtype: int64
------------------------
```

Figure 2. Distribution of images per class in each set.

4.1. Histogram of Oriented Gradients and Edge Detection

Our first complex feature was Histogram of Oriented Gradients (HOG, (2)). This method is essentially a descriptor for object detection that captures the edge orientation in localized portions of an image. Briefly, deduction of HOG contains the following steps:

- **Gradient Computation:** compute the gradient of the images using a filter - resulting in gradient magnitude and direction for each pixel.
- **Orientation Binning:** divide the image into small regions and compute the histogram of gradient directions for each region. The gradient magnitude is then used to weight the contributions to the overall histogram.
- **Normalization:** group adjacent cells and normalize the histograms within these blocks to account for any variation.
- **Reconstruction:** concatenate all normalized histogram groups to form a single feature vector representing the entire image.

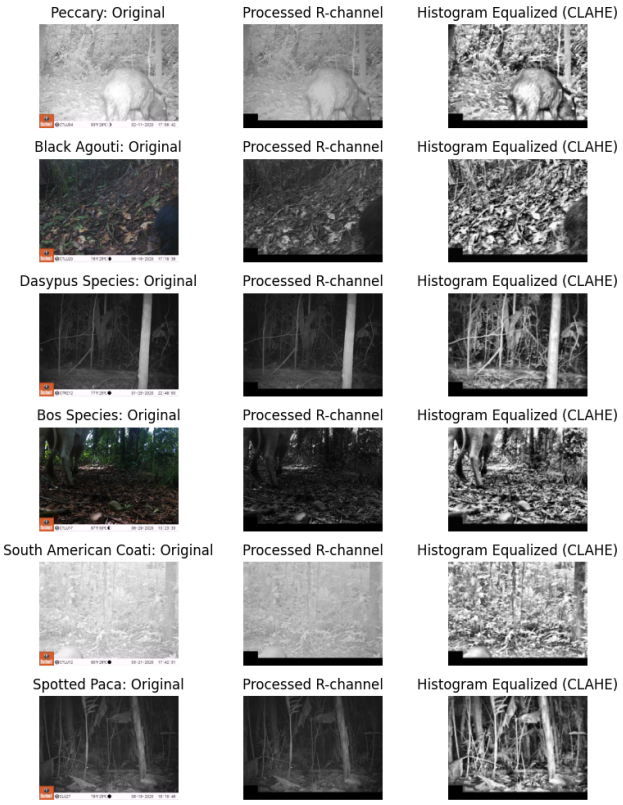

Figure 3. Image before and after preprocessing per class.

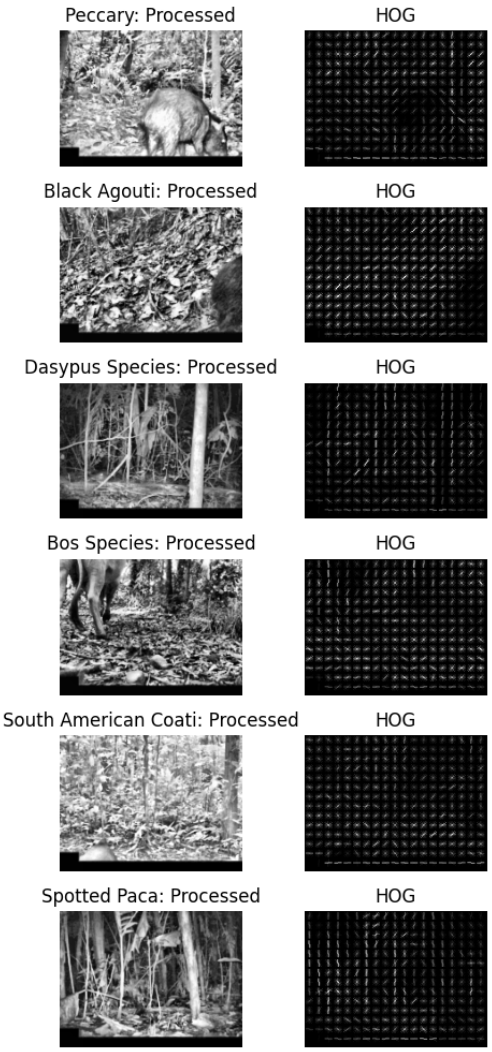

Figure 4. HOG feature visualization for each animal class.

We selected this feature due to its benefits in providing information on shape of objects in the image. Since the animals appear at varying distances from the camera, the sensitivity of edge and shape structure is crucial, and can be effectively preserved with this approach. We used 8 orientations and 10 by 10 pixel blocks as the parameter. Figure 4, shows an example visualization of the HOG feature for each class.

As noted, edge and shape detection are crucial for this problem due to the variability in depth, movement, and positioning of each animal in the dataset. Therefore, we additionally used the Sobel and Canny filtering techniques to extract edges of the image as simple features. The Sobel operator detects gradient intensity changes in both horizontal and vertical directions with two 3×3 convolutions after applying a Gaussian blur for smoothing (weights of 0.5 were used for both x and y directions). Canny is a more precise operator involving addition steps of non-maximum suppression and hysteresis thresholding (85 and 255 were used as the lower and upper threshold) to connect the edges properly and eliminate the weak ones. Figure 5, shows example visualizations of the edge detection features derived from Sobel and Canny for each class.

The two operators were opted for their readily availability and computational efficiencies. In the final classification models, the feature extracted with the Sobel operator was adopted, given its higher performance in individual feature tests over Canny in all models (detail accuracy scores illustrated in a later section, along with model evaluation). Despite the edge feature vectors being encoded in two dimensions, we considered it an essential piece of information to use alongside other more complex features in our models. Additionally, since we opted for a pre-trained CNN feature (discussed later), we believe this edge detection approach would provide efficient computation when combining features. However, some of our models underperformed with this feature due to the presence of numerous trees in the images, which complicated edge detection.

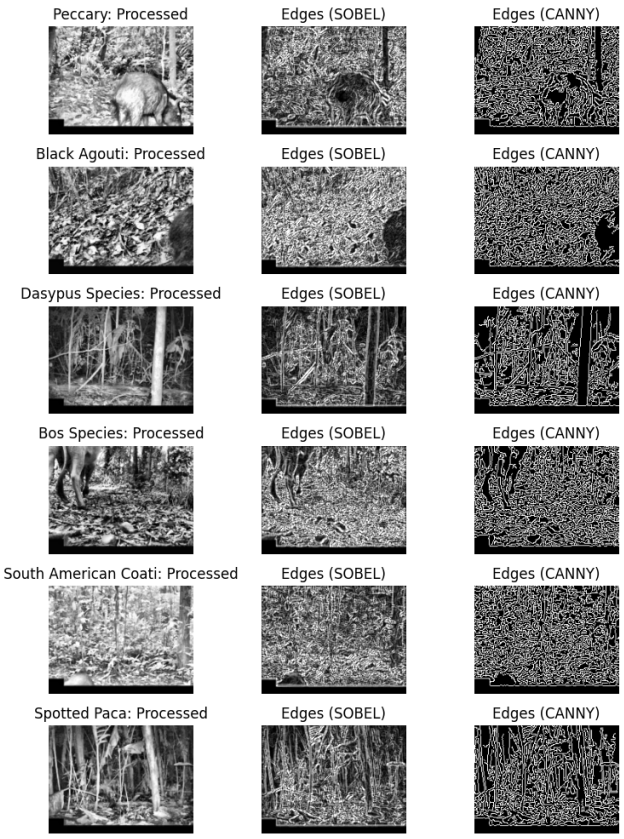

Figure 5. Edge detection feature visualization for each animal class.

4.2. Scale-Invariant Feature Transform, Bag of Visual Words, and Term Frequency{inverse Focument Frequency

The Scale-Invariant Feature Transform (SIFT, (3)) algorithm is used to describe local features in an image. Its primary objective is to ensure invariance in both scale and rotation, which makes it highly beneficial for classifying our multi-angled classes. Following the proceedings mentioned by Tharwart et. al. (2017) (5), we decided to include it as one of our complex feature vectors given that our subjects could be found either close to the camera, or far away, thus changing scale, and at somewhat different angles. Here are the key steps involvedin SIFT:

- **Extrema Detection:** use Difference of Gaussians (DoG) function to search for local extrema to identify keypoints.
- **Localization:** search through keypoints previously identified and remove points with poorly localized edges.
- **Describe Orientation:** using the gradients, assign a orientation to each keypoint.
- **Descriptor Generation:** for each keypoint, sample around it and gather the gradient information to generate a vector characterizing the region.

Once these descriptor vectors were generated, we employed a Bag of Visual Words (BoVW) model to cluster the features and key points into a set of visual words, using k-means clustering. Essentially, the key points identified in SIFT are assigned visual word descriptors to represent parts of the image. Thus, the entire image is characterized in a histogram of these words. This approach is significantly more cost-effective as the feature vectors are much more compact than using raw SIFT descriptors. We experimented with multiple cluster sizes for our k-means, but found that $\sqrt{numfiles}$ worked best. Additionally, we experimented with the following hyperparameters (table 1), to determine whether keypoint accuracy increased.

Hyperparameter	Tested	Selected
sigma	[1.6, 5, 10]	5
n octave layers	[1,4,5]	5
contrastThreshold	[0.03, 0.01]	0.01
edgeThreshold	[10,20]	20

Table 1. SIFT hyperparameter testing.

The selected hyperparameters mentioned above were chosen for the following reasons: the sigma was increased to be able to detect keypoints in blurry images, by increasing the Gaussian blur. The number of octave layers was increased to account for different distances between the camera and the animal. The contrast threshold was decreased to increase more low-contrast keypoints, since, even after histogram equalization, some of our images had low contrast. The edge threshold was increased to focus on keypoints from rounder images (animals) as opposed to more keypoints in more jagged areas (like grass).

Given that many of the keypoints were found in background images (trees or grass), we opted a Term Frequency–inverse Focument Frequency (TF-IDF) weighting approach. This method re-weights the bag of words histogram to emphasize the influence of unique (less frequent) words, thereby improving the discriminative power of the feature vectors. In our case, this meant excluding background descriptors, which were more common, and highlighting unique descriptors, our animals, since each image only included a couple of animals, at most five. Refer to figure 6 for a visual representation of these steps.

We chose SIFT because of its robustness to illumination changes and its scale invariance. Given that the camera traps captured images during both day/night and accounted for variations in animal size, we needed a feature that could

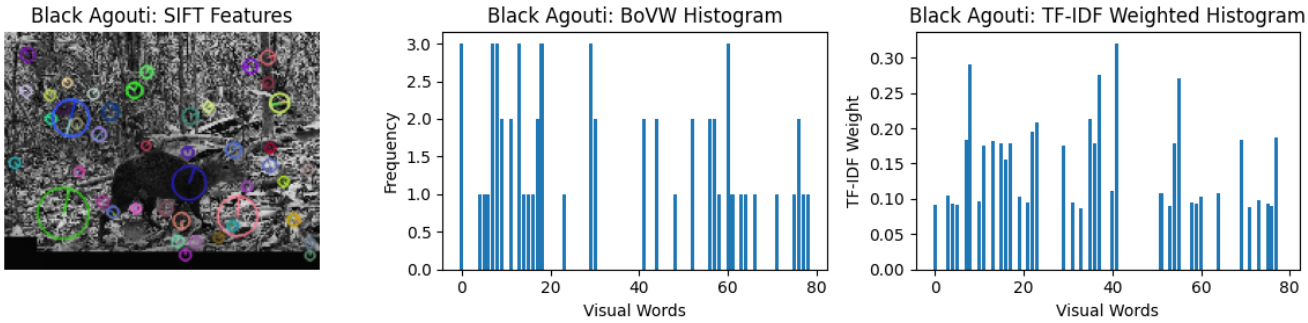

Figure 6. Example of SIFT features extracted from a Black Agouti class image. Note the differences in the histograms after the BoVW and TF-IDF weighting.

effectively manage the diverse lighting and orientation conditions, even after our preprocessing steps (such as handling infrared camera images and capturing only portions of the animals).

4.3. Weights from Pre-trained Convolutional Neural Network

We also selected a complex feature derived from a pre-trained Convolutional Neural Network (CNN) model. Specifically, we utilized Keras' InceptionV3 model (4), which is based on Google's Inception CNN, originally developed for the ImageNet Recognition Challenge. The preprocessed camera trap images were sent to the CNN model, then the weights from the third layer from the last were extracted as a feature vector to be used in our models. This layer was chosen since the last 2 layers of the original model was designed for pooling and classification purposes.

The inclusion of the pre-trained weights from a complex model like InceptionV3 allow us to leverage the robust representations for general image classification the model already learned and improves overall analysis efficiency.

4.4. Principal Component Analysis and t-SNE

Principal Component Analysis (PCA) is a linear dimensionality reduction technique used to transform our data into a new coordinate system. This transformation produces new principal components that are arranged according to the amount of variance they explain in the data. We can retain the principal components that account for a significant portion of the variance (in our case 95%, figure 7), while discarding the others. This approach reduces dimensionality while preserving the most important information. We selected this operation to mainly improve efficiency. This helped us reduce noise and dimensions in our feature, allowing for faster and more centralized computations.

Additionally, t-Distributed Stochastic Neighbor Embedding converts similarities in data points into joint probabilities, further reducing dimensionality. It allows for effective visualization of high-dimensional data into a lower-dimensional space. We selected this approach due to the visual benefits. As seen in figure 8, we can visualize the groupings of these joint probabilities across each of our feature vectors and tune accordingly. The example shown in figure 8 is after several t-SNE visualization iterations after tuning feature vectors accordingly.

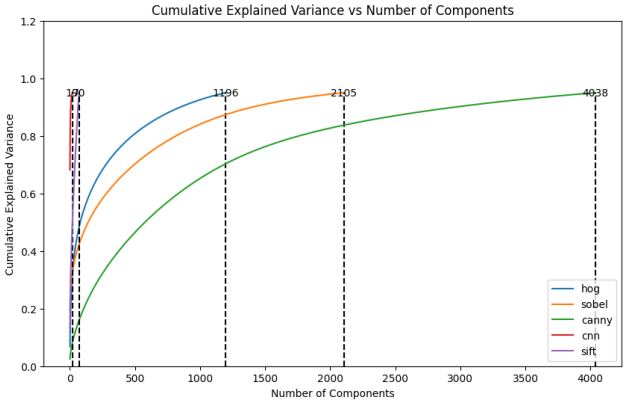

Figure 7. Number of components that explain 95% of the variance for each of our features.

5. Classification

5.1. Linear Discriminant Analysis

Our baseline model was a Linear Discriminant Analysis (LDA). It identifies a linear combination of features that optimally separates two or more classes. It operates on the premise that each class is represented by a distinct Gaussian distribution. The goal of LDA is to determine the decision boundary that maximizes the distance between the means of different classes while minimizing the variance within each class. LDA models are based on several fundamental assumptions:

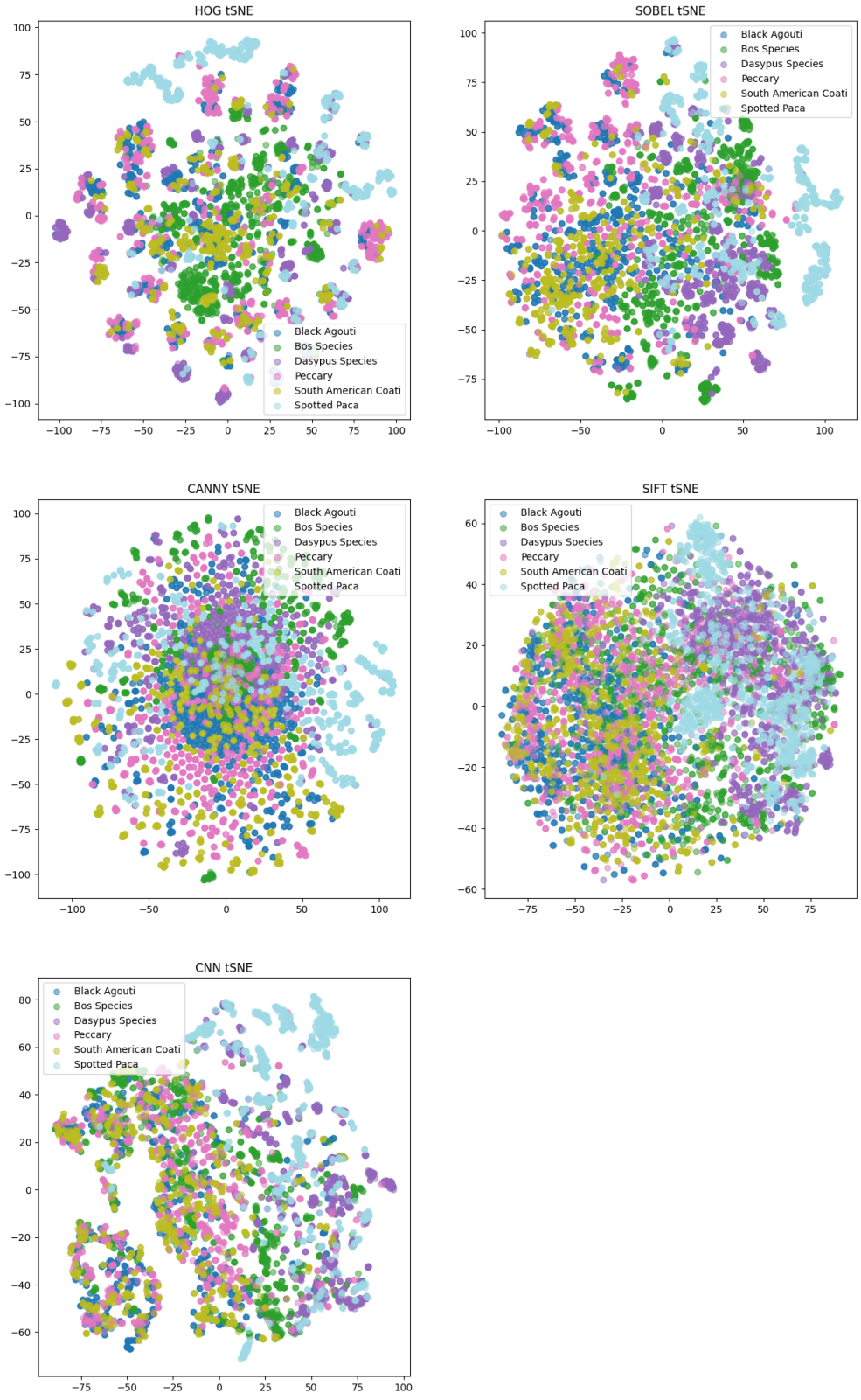

Figure 8. Dimensionality reduction applied to each of our features with a t-SNE visual.

- **Normal Distribution:** assumes all data points for each class are normally distributed.
- **Equal Covariance:** assumes all classes share the same covariance matrix.
- **Linearity:** assumes the decision boundaries between classes are linear.

We trained our LDA model using all features across 5 folds in Group Cross Validation Search. To further customize our training process, we incorporated hyperparameters that cannot be learned from the data itself. The proper selection of these parameters is essential, as it can significantly enhance the results. The following hyperparameters that affect the training process:

- **solver:** the algorithm used to compute the LDA.
  - svd: Singular Value Decomposition does not require the computation of the covariance matrix.
  - lsqr: Least Squares Solution, works best when the large datasets are expected to have well separated class distributions.
  - eigen: Eigenvalue Decomposition, generally used when the number of features is small.
- **shrinkage:** used to regularize the covariance matrix in the lsqr or eigen solvers.
- **priors:** class prior probabilities, default to the training data class proportions.
- **n_components:** number of linear discriminants to retain.
- **store_covariance:** boolean to determine if the class covariance matrix is computed and stored.
- **tol:** threshold for rank estimation when using the svd solver.
- **covariance_estimator:** custom covariance estimator.

To explore different options, we employed a grid search to identify the best model with an lsqr solver and custom shrinkage of 0.4. A grid search tests all possible combinations of hyperparameters and compares the outcomes, allowing us to identify the best possible model within our original constraints. This achieved a train accuracy of 0.7600 and a test accuracy of 0.7748 with a model training time of 41.5984 seconds. This is very generalizable, as the train and test accuracy's are very similar, meaning the model did not overfit the data and is expandable to unseen images. This can further be seen in the ROC curve in figure 10 , indicating that for each class the area is > 0.91 indicating great discrimination.

As illustrated in the train/test set confusion matrices and the ROC curve (figures 9 and 10), our model effectively fit the classes. These results indicate that the model excelled in classifying the Bos Species, but frequently confused the others. Although this train/test set accuracy is higher than SVM (discussed later) the computational complexity is higher as well. Refer to tables 2 and 5 for more information on the model features and grid search values.

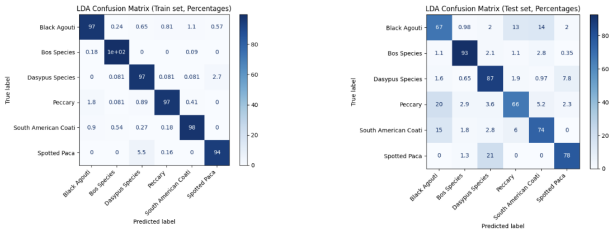

Figure 9. LDA confusion matrices for the train and test sets with percentage bounds.

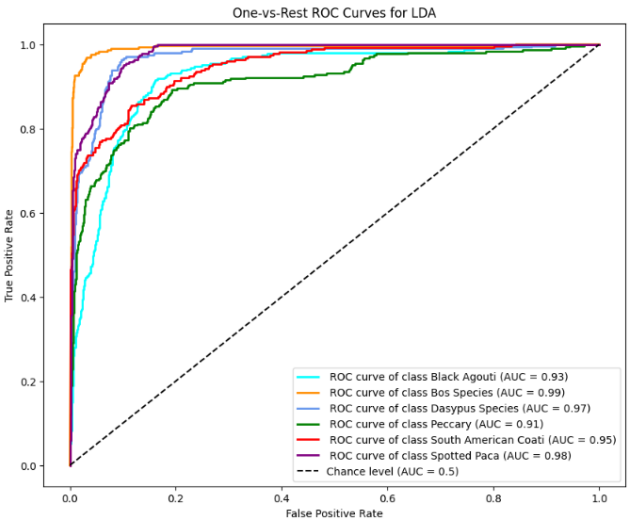

Figure 10. ROC curves for LDA split from chance level to individual class levels.

5.2. Support Vector Machines

Support Vector Machines (SVM) are supervised learning models particularly effective at classifying high-dimensional spaces. The two main features of a SVM are the following:

- **Hyperplane:** SVMs find the boundary that best separates the data into classes.
- **Support Vectors:** the data points that are closest to the hyperplane and influence its positioning/orientation.

Feature	Dimension	Accuracy (train)	Accuracy (test)	Training time (s)
HOG, SIFT, CNN, SOBEL	3390	0.7600	0.7748	121.7246
HOG	1196	0.8783	0.7291	75.7028
SIFT	70	0.4960	0.4370	1.5478
CNN	19	0.4382	0.4365	1.1922
CANNY	4038	0.9790	0.7391	2263.5867
SOBEL	2105	0.9456	0.7586	342.1045

Table 2. Grid search and performance of all LDA models.

Feature	Dimension	C	Accuracy (train)	Accuracy (test)	Training time (s)
HOG, SIFT, CNN, SOBEL	3390	1	0.7624	0.7586	102.2918
HOG	1196	1	0.766	0.7715	19.7424
SIFT	70	1	0.5282	0.5123	1.7843
CNN	19	100	0.6085	0.5892	1.1735
CANNY	4038	1	0.7233	0.7068	97.0231
SOBEL	2105	10	0.7411	0.7285	27.29

Table 3. Grid search and performance of all SVM models.

This ideal orientation maximizes the margin between support vectors of all classes.

We performed a Grid search for all of our SVM models across 3 folds in Group Cross Validation Search. In all cases, the best kernel was a radial basis function (rbf), but the best regularization parameter varied depending on the feature vector(s) used (see table 3)

The following hyperparameters were selected as part of the grid search. In all cases, the selected decision function was one-vs-rest, to improve computational efficiency:

- **kernel:** transforms the input into a different dimensional space where it might be more separable.
  - lin: Linear, has no transformation, used when linearly separable.
  - rbf: Radial Basis Function, using Gaussian function, maps to an infinite-dimensional space.
- **C:** this is the regularization parameter that controls the trade-off between low training error and low testing error. A smaller value of C represents a larger margin.

The best performing SVM model appeared to be the HOG-only SVM model, containing the following hyperparameters: kernel=rbf and C=1. This model obtained a train accuracy of 0.766 and a test accuracy of 0.7715 with model training times of 19.7424 seconds and 6.4477 seconds, respectively. Since our train and test accuracies are very similar, this is a good indication that our model is generalizable and not overfitted. This is further validated in the ROC curve seen in figure 12. This curve shows each class has an area > 0.89, indicating great discrimination. While not as good as the ROC curves for our LDA model, the computational efficiency makes the trade-off worth it.

This model was able to classify the Bos Species the best, consistent with the results from our LDA model. However, it struggled with differentiating the black agoutis and peccaries (both are round and black), and spotted pacas and dasypus (both are nocturnal), as seen in the SVM test confusion matrix in figure 11. We also noticed that it struggled with animals in movement, due to the histogram of gradients essentially being flat due to the motion (see figure 13)

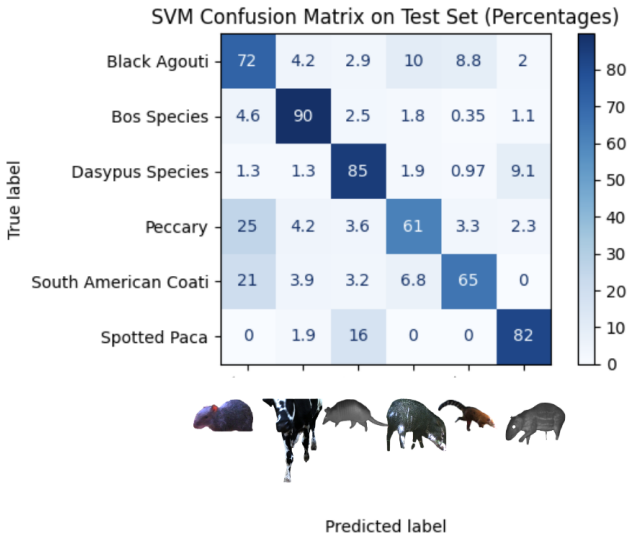

Figure 11. SVM test set confusion matrix with percentage bounds.

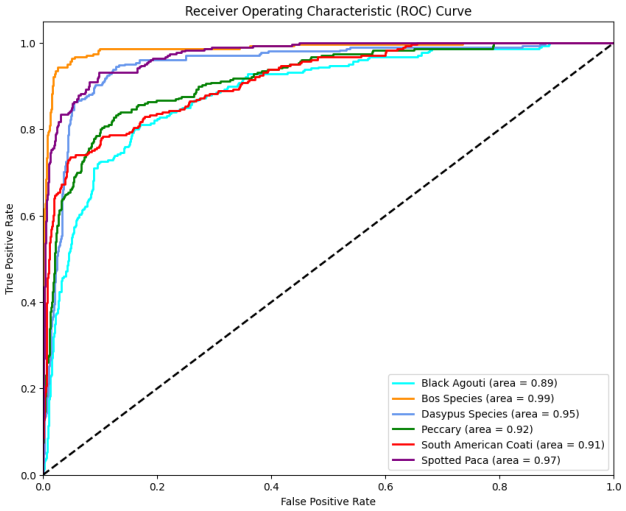

Figure 12. Test ROC curve of best-performing HOG-only SVM.

5.3. Neural Networks

Neural Networks (NN) are computational models inspired off the biological neural networks in the human brain. They consist of layers of interconnected neurons (or nodes), where each connection has an assigned weight. These artificial networks can learn complex relationships in data through a training process that adjusts the weights based on predicted and expected outputs. The following are fundamental concepts found in all artificial networks:

- **Input Layer:** receives the data
- **Hidden Layers:** optional intermediate layers to process the inputs and pass them forward
- **Output Layer:** uses the processed data through the previous layers to select an output of the network
- **Activation Functions:** applied to node outputs to introduce non-linearity - enabling complex learning patterns
- **Weights/Biases:** adjusted parameters on the nodes to minimize the error between the networks predictions and the actual values

We have decided to explore the use of NN in this problem with a simple architecture of only one hidden layer and one dropout layer positioned between the input and output layers. Similar to the handling of other models, grid search was applied in order to identify the best model settings based on validation accuracy. The best model had the following hyperparameters: activation function=relu, 300 neurons, weight_constraints=2.0, and dropout_rate=0.2. This resulted in training accuracy of 0.7453 and testing accuracy of 0.7653 with model training time of 10.7483 seconds (table 4). The similar train and test accuracy performance and AUC over 0.9 for all classes (see figure 15) supported that the model is generalizable and not overfitted.

The test confusion matrix 14 reveals that the model was able to identify the Bos Species most accurately, as expected based on the outcome of our previous model. In this case, the NN probably excelled at identifying them due to their distinctive appearance of long, straight legs. For other classes, when the model failed to predict the correct class, it was usually because it predicted another animal class of the same habit. In other words, the model struggled to differentiate within the group of diurnal animals (black agouti, peccaries, coati) and within the group of nocturnal animals (dasypus, spotted paca) in negative cases.

6. Generalizability

Generalizability refers to the ability of a model to perform well on unseen data that was not part of any training or validation sets. It is crucial as it determines how well a model can make classifications in real-world scenarios. As previously referenced, our data was incredibly diverse in animal orientation, lighting, movement, number of cameras, among others, making animal classifications a difficult task.

As discussed all 3 models in evaluation achieved training and testing accuracies close to or over 75% with ROC curves indicating a high level of distinction. These models are generalizable because they performed significantly better than chance and were not overfit.

7. Efficiency vs. Accuracy

The trade-off between efficiency and accuracy depends on the objectives of the model. Efficiency refers to how well a model utilizes computational resources and time to yield results, while accuracy refers to a model's ability to correctly classify instances within a dataset.

For this project, we aimed to optimize two solutions that address both domains due to the practical applications of our findings. If the model is intended for real-time classification, we would opt for the more efficient approach. Conversely, for descriptive outputs that are not time-sensitive, we would choose the more accurate option.

Our most efficient solution was the SVM model with the HOG feature vector, which trained in a fraction of the time required by the other approaches. This advantage is particularly beneficial as the dataset scales to include more classes and images. The most accurate solution was the LDA model achieving nearly 80% testing accuracy. With all things considered, our best model was our SVM approach as it yielded the high testing accuracy compared the training time. From all our approaches, this is the obvious well-rounded solu-

Figure 13. Misclassified peccary under HOG-only SVM model.

Feature	Dimension	Accuracy (train)	Accuracy (test)	Training time (s)
HOG, SIFT, CNN, SOBEL	3392	0.7452	0.7653	10.7483
HOG	1196	0.7229	0.7235	16.3108
SIFT	70	0.4964	0.4989	9.4175
CNN	19	0.5569	0.5229	10.2079
CANNY	4038	0.6781	0.7313	13.1783
SOBEL	2105	0.7124	0.7581	10.7483

Table 4. Grid search and performance of all NN models.

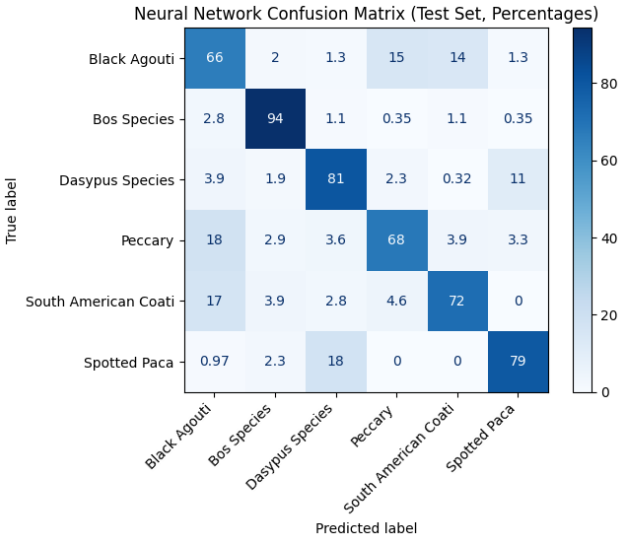

Figure 14. NN confusion matrix for the test sets with percentage bounds.

tion. More detail into the efficiency and accuracy of each model can be found in table 5.

8. Limitations

Although initially thought to have been clean dataset, given that the labels had been reviewed by scientists, a subsequent review revealed six mislabeled images out of the 6254 training images. We did not review the validation nor the test data, but decided that the percentage of mislabeled images would not be likely to have a significant effect in our classification. Moreover, the mislabeled images were not from all classes, just 3 from the Peccary and the 3 from the Black Agouti classes totalling 6 training images. We also identified some images in the training set which were over-exposed, and appeared mostly white, even after histogram equalization.

Additionally, the edge detection feature (either from Sobel or Canny) on its own offered some limitation. This is because the feature resulted in a similarly dense image with less information as the original. The goal was to use this feature in conjunction with the others, but on its own, it was not a strong performer. This was clearly seen in images involving a lot of noise. We also noticed, when performing the gap analysis of our models, that Sobel recognized our ”blacked-out” metadata as an edge (see figure 13), which is not ideal. When performing gap analysis on the Sobel-only

Model	Parameter search values	Best value	Best accuracy	Model time
LDA	solver: svd, lsqr, eigen	lsqr	Train: 0.75997 Test: 0.7748	Training: 121.7246 s Inference: 0.0152 s
	shrinkage: auto, 0.2, 0.4, 0.6, 0.8	0.4		
	n_components: None, 1, 2, 3, 4, 5	None		
	tol: 1e-4, 1e-5, 1e-6	-		
SVM	kernel: linear, rbf	rbf	Train: 0.7624 Test: 0.7586	Training: 102.2918 s Inference: 29.8939 s
	C: 0.001, 0.5, 1, 10, 50, 100	1		
NN	activation: relu, tanh, linear	relu	Training: 0.7453 Test: 0.7653	Train: 10.7483 s Inference: 0.1899 s
	neurons: 100, 200, 300	300		
	weight_constraints: 2.0, 3.0, 4.0	2.0		
	dropout_rate: 0.2, 0.4, 0.6	0.2		

Table 5. Grid search for our 3 model approaches with all feature vectors.

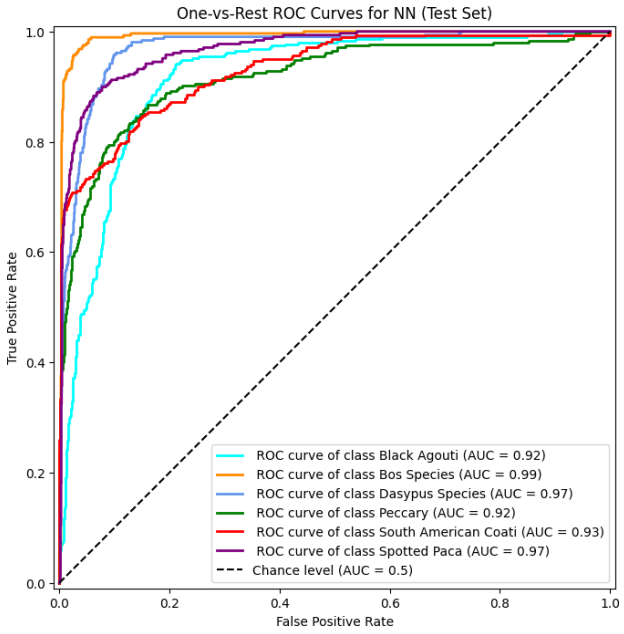

Figure 15. ROC curves for NN split from chance level to individual class levels.

SVM model, we noticed that it struggled when there were many trees in the background, likely due to edge detection prioritizing the tree leaves and shape.

The use of SIFT as a feature vector, while commonly used in animal classification (2), was hard to grasp, in terms of interpretation, given that the visualization of SIFT shows keypoints, but it is hard to discriminate which keypoints will be useful and which won't, given that the animals varied in position and location. We focused on selecting keypoints in areas that made sense to us in terms of discrimination, such as the banded tail of the coati, the legs and udder of bos species, the lined armor of the dasypus species, the spots of the spotted paca, the rounded ears of the coati, and the tusks of the peccary. However, while SIFT is a scale-invariant feature transform, when the animal is blocking most of the camera, or facing away from it, many of its features won't be easily identifiable. Additionally, while we implemented the TF-IDF weighing, it is hard to visually compare the unweighted vs. the unweighted histograms, in order to know which words in the visual vocabulary are more common than others (see figure 6); we had assumed that the background descriptors such as trees or grass would be more common, but we noticed that our model underperformed in classifying bos species, a class in which most models excelled, and we surmise that it may be due to the frequency in which legs of this species appear in the images. We also noticed that some of the keypoints included the blocked-out metadata, which we did not want included in our classification. When performing gap analysis on the SIFT only SVM model, we notices that it struggled in classifying the Bos Species, most likely due to their legs being a frequent descriptor which was discarded from the TF-IDF based on its frequency.

When performing gap analysis on the The CNN-only SVM model, we notices that it struggled with images from two of the cameras (A01 and N24), due to the setting involving a lot of undergrowth - resulting in the pre-trained model confused by what to extract. The fact that most of our classifiers struggled in telling black agoutis and peccaries apart may be due to them both being black. Despite using strategies to deal with different scales (i.e, distance between the animal and the camera), since both animals are black, when the small agouti is close to the camera, it may appear to the classifier as a peccary, and vice-versa. To solve this problem, we would need to triangulate with multiple cameras, which was not an option in this data set. However, we plan to use the sequence of images to combat this issue in future versions of our project.

9. Next Steps

Addressing the mislabeled data is the immediate priority, as well as eliminating saturated images from the dataset. We

also want to take another approach to eliminating the metadata at the bottom of the image, as some of the features still included it, such as Sobel or SIFT. Following this, we will further investigate the impact of various features on model performance. Incorporating additional feature vectors, such as image differencing, might also help classify images depicting animal movement, as well as combining simple feature vectors.

10. Conclusion

In this study, we explored the classification of six animal species form the biologically diverse Orinoquia region in Colombia using a variety of computer vision models. Amongst these, the Support Vector Machine (SVM) demonstrated the highest performance with a testing accuracy of 0.7715, when using the HOG feature vectors.

While the Linear Discriminant Analysis (LDA) and Neural Network (NN) models also showed promising results with high accuracy, the SVM's balance of efficiency and accuracy makes it the optimal choice for practical applications.

Future work will focus on addressing the identified limitations, including mislabeled data and feature extraction uncertainties. Further exploring the impact of various feature vectors and incorporating more advanced techniques like image differencing to create image mask for region of interest (ROI) and alternative edge detection methods will be key to enhance our models.

Our findings represent a significant step towards automating wildlife monitoring and classification, offering a foundation for future research and practical implementation in ecological studies and conservation efforts. All of the code for the project can be found in our repository.

References

[1] W. Commons. Mapa de colombia (región de la orinoquía).svg, 2014.

[2] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 886–893 vol. 1, 2005.

[3] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60:91–110, 2004.

[4] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[5] A. Tharwat, T. Gaber, A. E. Hassanien, G. Schaefer, and J.-S. Pan. A fully-automated zebra animal identification approach based on sift features. In *Genetic and Evolutionary Computing: Proceedings of the Tenth International Conference on Genetic and Evolutionary Computing, November 7-9, 2016 Fuzhou City, Fujian Province, China 10*, pages 289–297. Springer, 2017.

[6] J. Vélez, W. McShea, H. Shamon, P. J. Castiblanco-Camacho, M. A. Tabak, C. Chalmers, P. Fergus, and J. Fieberg. An evaluation of platforms for processing camera-trap data using artificial intelligence. *Methods in Ecology and Evolution*, 14(2):459–477, 2023.

[7] K. Zuiderveld. *Contrast limited adaptive histogram equalization*, page 474–485. Academic Press Professional, Inc., USA, 1994.