Animal Detection in Camera Trap Images and Videos

Jiayue(JY) Xu, Yingyu Fu, Xinwen(Melody) Li

Abstract

The project deals with building up an effective image/video animal detection model for Saving Nature. We train deep learning models to identify classes of animals in the Saving Nature dataset. Our current image classification model correctly identifies animals with an accuracy > 90% and our current image detection model correctly identifies class of animals with an accuracy > 87%. We expect the accuracy to improve as we are gaining access to more raw data in the near future.

1 Introduction

The ability to access and extract accurate real-time data about the occurrence of species in the wild could help scientists to study more about the ecosystem and evaluate the effectiveness of strategies for helping endangered species. In this project, we build a product that has the ability to collect essential information from a data source to best serve our clients in need. Our client, Saving Nature, collects their raw data from installing motion sensor camera traps across different rainforests around the world and these camera traps enable Saving Nature to continuously get both image and video data in a real-time manner. However, extracting and recording essential information from a fairly large volume of data is very time-consuming for scientists when they rely on human labor to detect animals. This repetitive process for Saving Nature can be very difficult at times especially when the animal is very good at camouflage, or because of poor image quality captured by camera trap due to different weather conditions and change of light, or the animal is just relatively small in size. All these situations make the animal difficult to detect by human eyes. As a result, manual detection not only expends more time, but also causes a more unavoidable human error that could lose a part of essential information wanted by Saving Nature. Therefore, our main goal is to help Saving Nature eliminate the downsides of the current process by using computer vision to build deep learning models to detect animals in the data.

2 Product Roadmap

Embedded in the main goal, we have a series of sub-goals to achieve to get to the final product.

Stage 1

The first sub-goal is data cleaning. This stage involves both the Saving Nature team and us. Saving Nature is in charge of providing the raw data together with the correct label to us. We are responsible for cleaning and organizing the data into the exact format we want for our next stage. This stage sets up the foundation as we develop our product.

Stage 2

The second sub-goal is building classification deep learning models that process image data only. The models classify the data into different categories and the target categories can be either binary or multi-class. When the model is doing binary classification, it helps to filter out almost half the amount of non-animal images we are not interested in. When the model is doing multi-class classification, we build our model based on the most general class of the animals first. According to the biological taxonomy, there are ranks of groups of categories that can define an animal. Higher rank means that each of the categories in that rank contains a broader range of animals and using the higher rank means the animal defined will not be as specific as to what exact species it is. As we continue to get more cleaned data from the first stage, we build a model that categorizes the animal into the lower ranks, such as order and species.

Stage 3

Having the ability to classify images motivates us to the next sub-goal which involves building a detection model to detect not only the type of animal that appears but also the location of the animal on the image. This model will help the scientists to locate the animal even if the human is unable to detect it easily when difficult conditions appear as discussed earlier.

Stage 4

At this point, we have not touched anything on the video data and the fourth sub-goal is building a model handling video data. Video data can be seen as sequential frames of images and we are going to use the previous image-handling models to build a new detection model with the ability to process video data.

Stage 5

At the last stage, as we finish all the model building, we will start building a user-friendly product together with a detailed user manual for the scientists to use. The user-friendly product will involve the least amount of programming intervention and should connect to the raw data source location as its input and perform the detection internally to produce useful outputs. It should also include the functionalities of change parameters such as the model type (classification vs detection), the output type (binary vs multiclass), and adding a new animal type to retrain the model. Completing all the sub-goals would greatly reduce the need for manual work and human error. It would help Saving Nature get access to essential data more quickly leaving the scientists more time to study animal species occurrence, populations, and behavior, and evaluate the effectiveness of restoring nature and take actions to protect them.

3 Data

Our data are multimedia files of images and videos which are produced by camera traps placed at various locations within the rainforests in the South American and Southeast Asian regions. These camera traps are motion-censored and are triggered by any movements within the capturing frame, including the passing of any animal species and any natural motions in the environment. When an animal comes into frame, the camera trap captures a still image as well as a video recording of 15 or 30 seconds. This produces captures with animal species, and are referred to as “Animal” images and videos. However, it is possible that the camera traps are triggered by natural movements, such as raindrops, falling leaves, moving branches. This produces images and videos without any animal species captured, hence are labeled as “Ghost”. Our goal in this capstone project is to differentiate the “Animal” captures from the “Ghost” captures, and also to identify the animal species in the “Animal” captures.

3.1 Data Preprocessing Pipeline

Our data preprocessing pipeline consists of four stages as illustrated in Figure 1. This pipeline is applied to every location where the camera trap data is coming from, on a different timeline according to the availability of the data.

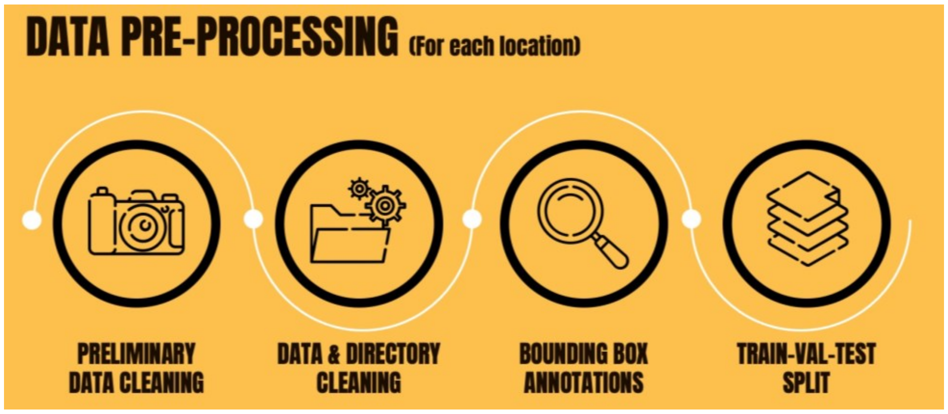

DATA PRE-PROCESSING (For each location)
PRELIMINARY DATA CLEANING
DATA & DIRECTORY CLEANING
BOUNDING BOX ANNOTATIONS
TRAIN-VAL-TEST SPLIT

Figure 1: Data Preprocessing Pipeline

Stage 1: Preliminary Data Cleaning

The pipeline begins with the preliminary data cleaning is done by our clients, who manually classifies the raw camera trap images and videos by the animal species that is being captured, and separates them into folders by the camera trap identifiers and the species scientific names. The Saving Nature team also updates the metadata of the multimedia files based on the settings of each camera trap and the file names with the date and time of the capture. This is the existing data cleaning processing of our clients, which is extremely time and labor intensive, that we hope to replace with our machine learning algorithm.

Stage 2: Data and Directory Cleaning

As machine learning algorithms are very memory-consuming, especially neural networks with multimedia files, it is common practice to not load all the data into memory, but to load as much data the algorithm can process in each batch, which requires the file directory and file names to be set up in a specific way. For stage two, we proceed with the reorganization of the file directory in a way that is suitable for the data loaders for our models. In order not to lose any relevant information, the original file directory and file names are extracted, parsed and embedded into the file names of the images and videos.

Stage 3: Bounding Box Annotations

In addition to classifying the camera trap data by the animal species captured, we hope to locate the specific animal instances within the images or videos. For object detection, in either images or videos, bounding boxes are often used to describe the locations of the subject matter. A bounding box is an upright rectangle that encloses an object of interest. After we pass the cleaned data directory, our clients will then engage student volunteers to annotate the images and videos with bounding boxes. Using publicly available annotation platforms, we designed the workflow for the annotation process for images and videos separately.

Images

For image annotations, we utilize the MakeSense online platform, which provides an interactive user interface to draw bounding boxes on images. The student volunteers are instructed to upload a whole directory of images, such as a single animal species of a particular camera trap, for annotations and export the bounding boxes in three file formats, including CSV, TXT and XML, to use for different image detection algorithms.

Videos

For video annotations, we utilize the Supervisely online platform, which effectively splits each video into a series of still images according to its frame rate. Similarly, the student volunteers are instructed a whole directory of videos and export annotations in various formats. Additionally, the platform provides a “tracking” functionality which allows auto-prediction of bounding boxes once an animal instance is annotated. This significantly reduces the amount of time required to annotate each video.

For the first semester of the capstone project, we prioritized the image annotations, hence the last two stages of the current data preprocessing pipeline only involve images. The video preprocessing stages are expected to begin during the winter break.

Stage 4: Train-Val-Test Split

When building machine learning algorithms, it is common practice to split the whole dataset into three subsets for training, validation and testing purposes, where the training set is used to build the model, the validation set is used to fine-tune hyperparameters of the model and the testing set is used to estimate model’s performance on unseen data. As a final step, we perform the splitting the image dataset into the three subsets which consist of 80%, 10%, 10% of the images respectively. We implement the split by stratifying on the animal species and the camera traps, so as to ensure the proportion of images in different categories are equal across the subsets. For example, using binary categories, this ensures the proportion of “Animal” and “Ghost” images are the same across the subsets. This is important for model building as the subsets have to be similar in distribution for the model performance to be consistent and generalizable to unseen data.

3.2 Exploration Data Analysis

The camera trap data comes from four different geographical locations, namely Brazil, Ecuador, Colombia, and Sumatra. Across different locations, the distribution of the data varies significantly, with respect to the number of camera traps and the proportion of images and videos. While there are predominately images in the data from Brazil and Ecuador, the data from Colombia and Sumatra are nearly all videos.

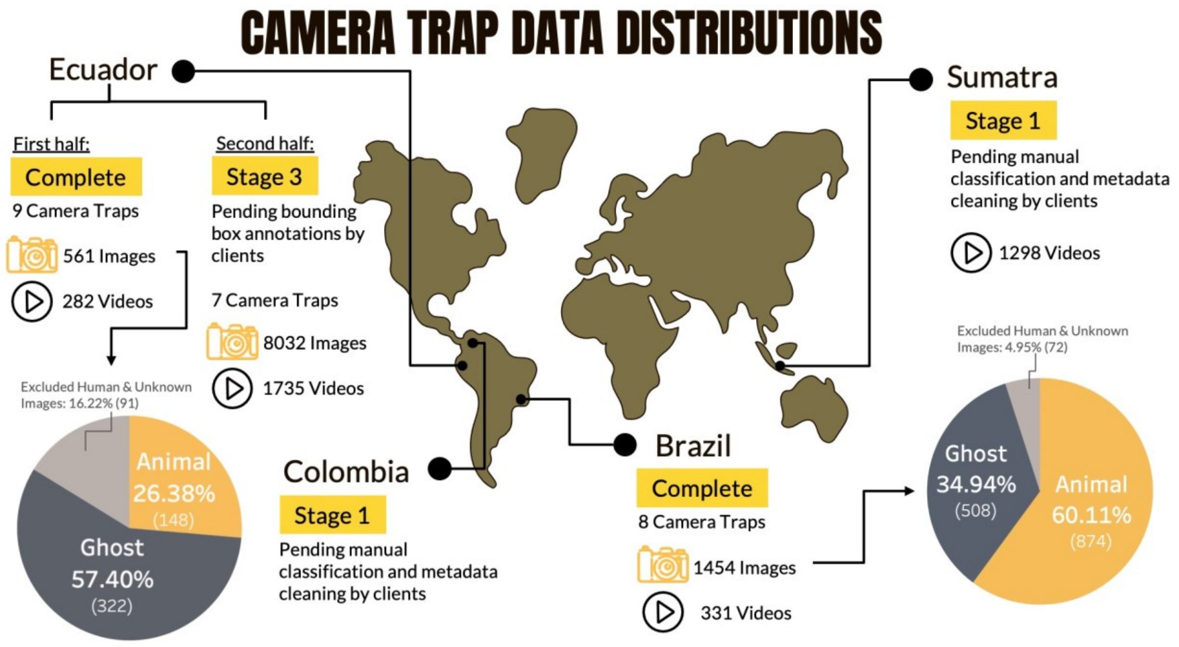


Figure 2: Camera Trap Data Distributions

As shown in Figure 2, data from each of the four locations is at a different stage along the data preprocessing pipeline. All the images from Brazil and the first half of images from Ecuador are completely preprocessed. The other half of images from Ecuador, together with data from Colombia and Sumatra, is pending bounding box annotations and preliminary data cleaning respectively by the clients. For the preprocessed images from Brazil and Ecuador, the proportion of "Animal" and "Ghost" images are almost opposite, with nearly two-thirds of "Animal" images in Brazil data and "Ghost" images in Ecuador data. This allows for a relatively equal proportion of images in these binary categories, which is ideal for training algorithms to identify if the image contains an animal or not. There are also a small percentage of the images that contain humans or are unknown, which we will exclude from our analysis.

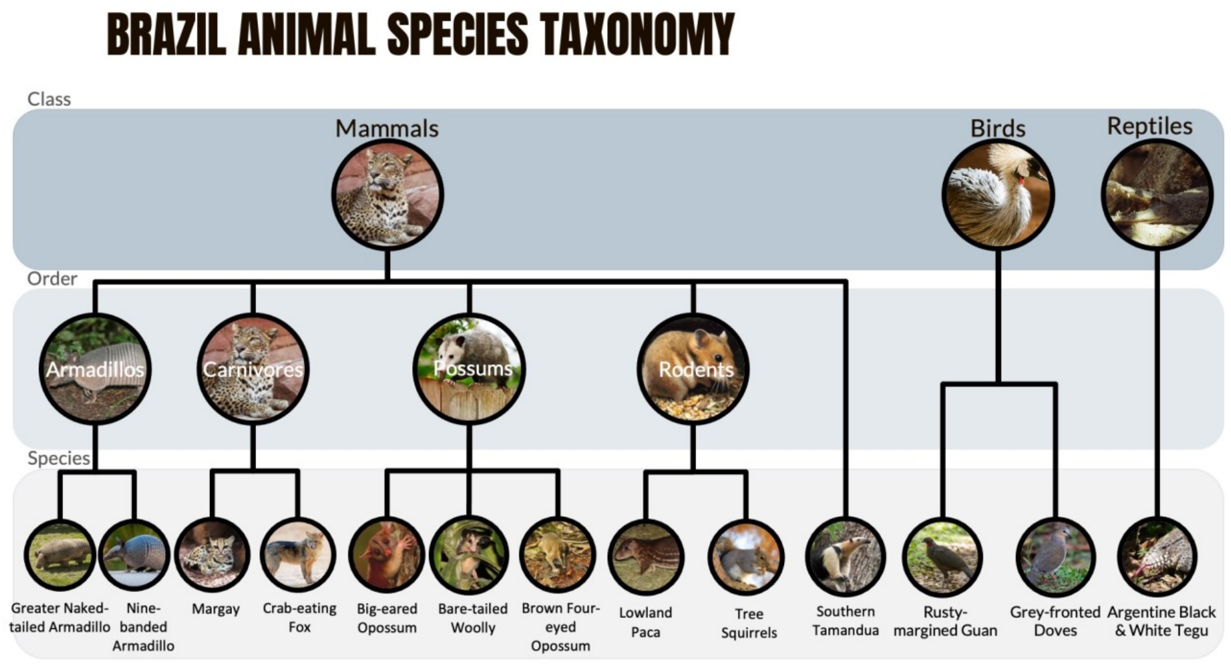


Figure 3: Animal Species Taxonomy with Brazil Camera Trap Data

To further classify the images beyond binary categories of “Animal” and “Ghosts”, we need to understand the animal species present in the images. Using the preprocessed images from Brazil and Ecuador, we explored the taxonomy tree of the animal species captured, with those included in the Brazil images shown in Figure 3. The animals belong to three animal classes, namely mammals, birds and reptiles. There are only a few different species of birds and reptiles class captured, but there are various kinds of mammals present in the data, which can be further grouped into different animal orders, as shown as the second level in Figure 3. As not all animal species are equally represented in the data, this taxonomy chart shows two different levels of hierarchy that can be used to group the images into categories to achieve more balanced distributions.

However, Figure 4 shows the data distributions after grouping at animal class and order level which are still largely unbalanced. Besides the “Ghost” class, which accounts for roughly 45% of the images, the mammal’s class, “Mammalia” is the next most represented in the data, with 35% of the images. The bird’s class, “Aves”, which is less than half the size of the “Ghost” class, has about 20% of the image. The reptile’s class, “Reptila”, is significantly under-represented in the preprocessed data with less than 2% of images. Looking at the level of animal orders, excluding the “Ghost” images, there are only three animal orders that are relatively represented in the data with more than 100 images. These include the generic birds order, “Aves”, which are birds that cannot be recognised at the species level. The other two animal orders are “Rodentia”, which are rodents, and “Didelphimorphia”, which are opossums.

By including more locations, we not only hope to balance the proportion of “Animal” and “Ghost” images, but also the distribution of images across different animal classes and orders. However, besides increasing the proportion of “Ghost” images, including images from Ecuador did not introduce more images in the under-represented animal classes or orders. This shows that the current distribution of images is a true reflection of the natural taxonomy of the animal species, which are also consistent across these rainforest locations.

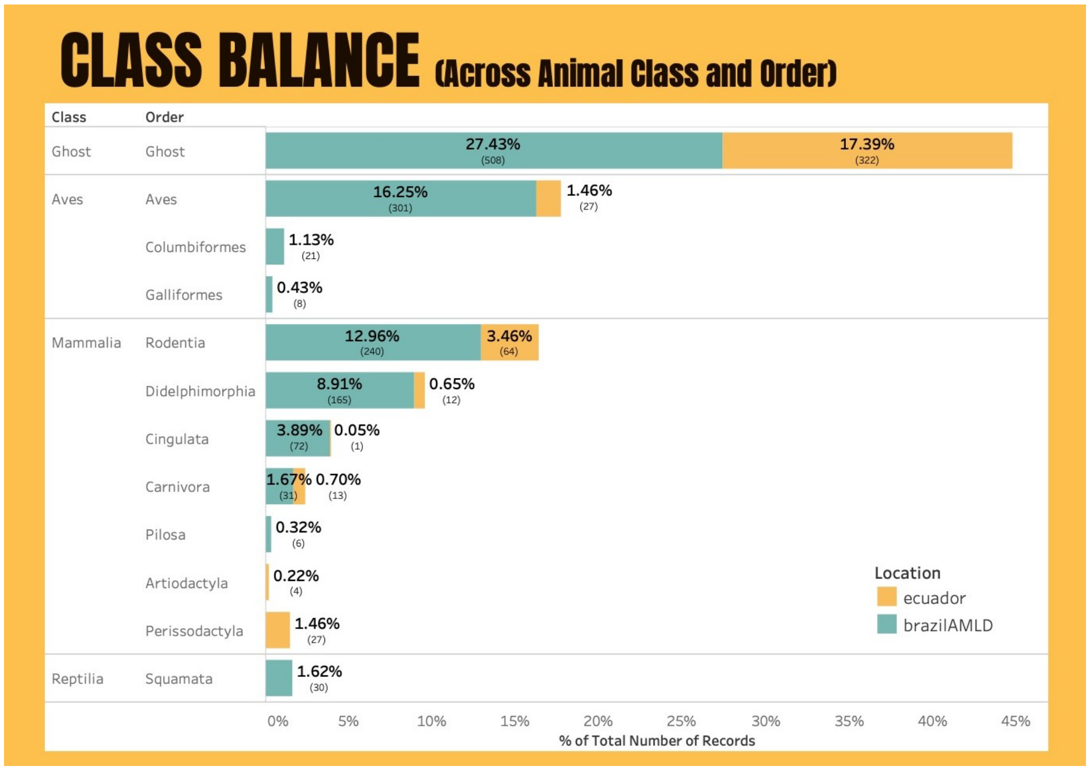


Figure 4: Class Balance Across Animal Class and Order

3.3 Limitations of Data

Class Imbalance

In order for any machine learning algorithms to effectively classify across various categories, the data distribution across these categories need to be balanced. However, the distribution of images across animal species is hugely imbalanced, which is one of the major challenges we face in our data. Even with more data from different locations and efforts to group animal species at different taxonomy levels, as described in the previous section, the class balance is still not improved. This has a significant impact on the performance of our multi-class models to correctly identify the images by animal categories, which will be elaborated in Section 4.

To overcome this inherent constraint in our data, we intend to employ common image processing methods to increase the sample size of the under-represented animal categories. Image augmentations and oversampling can be applied to generate samples from existing data, while extracting still frames of videos can be used to generate out-of-sample images.

Poor Quality of Captures

Another limitation of our data is the poor quality of some of the camera trap captures, which makes it difficult to identify the animal species within. As the camera traps are set in the wild rainforests and are triggered by motion, there are many uncertainties related to the quality of the images and videos captured. The camera settings, the lighting condition of the environment, as well as the speed of movement of the animal species, can cause the exposure quality of the captures to vary, which may result in overexposed or underexposed images and videos. While the camera traps are generally set in locations with clear frames of view, it is possible for this view to be obstructed by any natural occurrences, such as fallen leaves or branches, throughout the lifetime of the camera traps. This in turn causes the images and videos to be largely obstructed by non-animal objects. Another quality issue is the partial captures of the animal bodies, especially those with only the extremities of the animals, which is a result of the lag time of the camera trigger. All these quality issues can cause the captures to contain significantly less information about the animal species and thus are ineffective samples for the machine learning algorithm to learn from. This can be a major source of the errors for our model performances, as can be seen in Section 4. These sub quality samples can either be transformed or removed to improve the overall data quality.

4 Methods

4.1 Data Augmentation

Since the given dataset was fairly small, we applied several data augmentation techniques to the images. Among these augmentation techniques are rotation, reflection, and “shifting” - moving rows of pixels from one side of the image to another. This helped prevent our algorithms overfitting to coincidental properties of the location of the animals in the provided images. For example, it may have been that the animal only shows in the bottom right corner - such coincidences become quite common with smaller datasets. Reflecting, rotating and shifting the images to artificially generate more training data combats these issues so as to increase the ability of the model to recognize target animals in images of varied size, contrast, angles and so on. Some visual examples of this process are given in Figure 5.

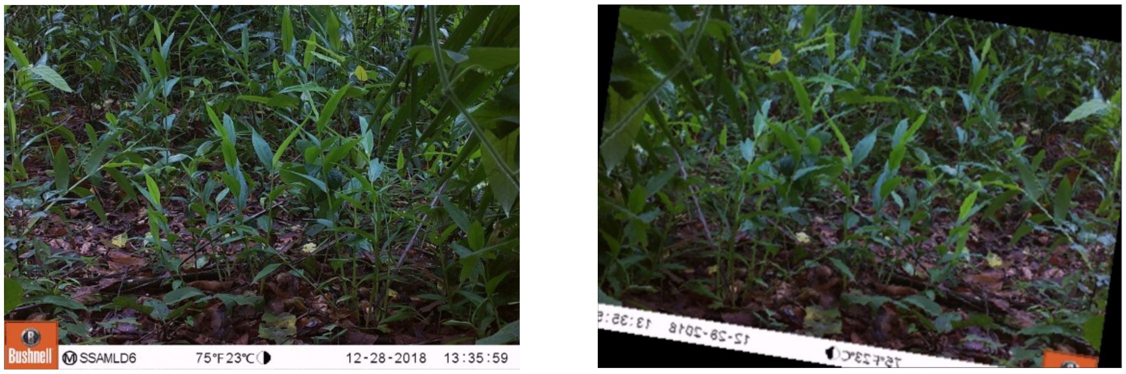


Figure 5: Data Augmentation Examples: original image (left) and flip/rotation image (right)

4.2 Transfer Learning

In recent years, deep learning architectures, especially the convolutional neural networks (CNNs), are the most successful type of methods that has been applied to computer vision tasks such as classification and object detection. Almost all classic and modern computer vision models (VGG, AlexNet, ResNet, Inception, etc.) are built on CNN architectures.

However, there are several major problems arising when we tried to build deep learning models from scratch. First, it requires a large amount of data to prevent overfitting issues. Most neural network models with more than a few hidden layers have more than a million weight coefficients to learn from data and even though after implementing several possible regularization like batch normalization and dropping techniques, it is still likely that the final model will overfit the noise. Moreover, time and expertise are required to obtain an optimal model as well. Constructing a neural network architecture and fine-tuning hyperparameters are both time-consuming and computationally expensive tasks. Therefore, we applied transfer learning to our task, an idea to leverage knowledge (weights, features, etc.) from a pre-trained model solving one problem and reuse it to a different but related problem. Compared with training a deep neural network from scratch, using transfer learning can accelerate the model building process, and can result in a more accurate and effective baseline performance with far less data.

There are many pre-trained models that use a large corpus of images in the training process for us to employ. In this project, we use them as the baseline model and retrain some of the layers to “adapt” the parameters to this new domain. Most of those models like ResNet and RetinaNet were trained on clear images and not on the poor quality camera trap captures containing small target objects; therefore, the domain of the input can be different and the pretrained features extracted might not be useful. Re-training some of the layers can help the model readjust to this new goal better.

4.3 Image Classification

In this project, we applied transfer learning models to do binary classification and multi-class classification.

Binary classification refers to assigning an object to one of the two categories (e.g., “Animal” or “Ghost”). We defined “Animal” as the normal state assigned with the class label 1 and “Ghost” as the abnormal state assigned with the class label 0. The model predicts a probability of a target object in the image belonging to which class state.

Unlike binary classification, multi-class classification involves predicting more than two classes for each object, which means there are no normal and abnormal state labels. Instead, the model predicts the probability of a target object belonging to each categorical outcome and makes the assumption that each object is assigned to one and only one label with the highest predicted probability. We trained our models using two versions of classes: animal class (“Aves”, “Mammalia”, “Reptilia”) and animal class-order (“Aves”, “Mammalia”, “Reptilia”, “Rodentia”, "Didelphimorphia", "Cingulata"). The number of classes to be classified would increase as more data becomes available soon.

Our binary and multiclass classification models are both implemented in PyTorch using a pre-trained version of ResNet-101 which has been trained on more than a million images from the ImageNet database. The training process was conducted on 25 epochs, with batch size of 4. Stochastic Gradient Descent optimizer was applied with momentum of 0.9 and the initial learning rate of 0.01, which is decayed every 7 epochs by a factor of 0.1.

4.4 Image Detection

For image detection, we applied pre-trained object detection models using images as input. Similarly to image classification, we experimented with object detection models at both binary and multi-class levels. In addition to the classification of objects as in image classification, object detection model adds an additional task of localizing the object instance within the images, using bounding boxes. Hence, the inputs to object detection models, in addition to the object classes to predict, like in the image classification, the annotated bounding boxes locations will also need to be provided to the models.

Our object detection models are implemented, with Detectron2 Python package, using transfer learning of RetinaNet with ResNet-50 backbone, using the pretrained weights on ImageNet dataset as initialisation. Training process was conducted on 25 epochs, with batch size of 4. Stochastic Gradient Descent optimizer is applied with momentum of 0.9 and initial learning rate of 0.01, which is warmed up for 1000 iterations and decayed every 3 epochs by a factor of 0.1. The number of classes is changed as 1, 3, and 6 for models at unary ("Animal"), animal class ("Aves", "Mammalia", "Reptilia") and animal class-order ("Aves", "Mammalia", "Reptilia", "Rodentia", "Didelphimorphia", "Cingulata") levels.

4.5 Performance Evaluation Metric

4.5.1 Confusion Matrix

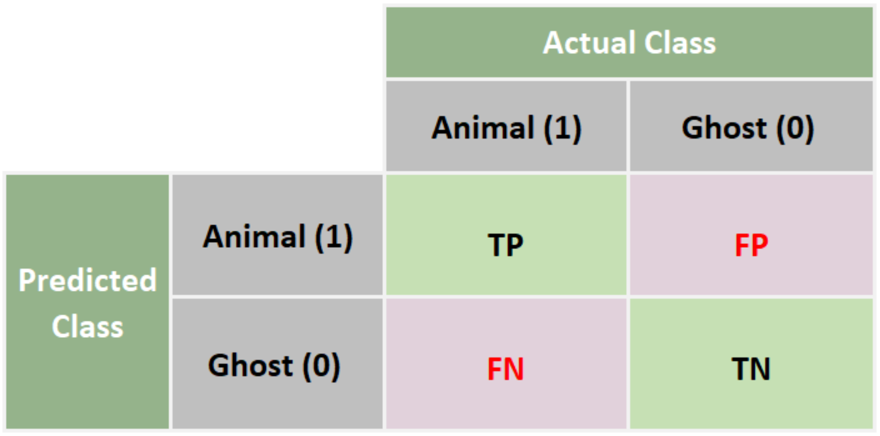
A confusion matrix is often used to evaluate the performance of a classification model (Figure 6). There are four parameters in the confusion matrix of two-class.

1. True Positive (TP): These are cases in which the model correctly predicts the "Animal" class.
2. False Positive (FP): These are cases in which the model incorrectly predicts the "Ghost" class.
3. False Negative (FN): These are cases in which the model incorrectly predicts the "Animal" class.
4. True Negative (TN): These are the cases in which the model correctly predicts the "Ghost" class.

Besides, there are some important rates calculated from a confusion matrix for a classifier.

1. Precision: A ratio of correctly predicted "Animal" to the total predicted "Animal", which is a good measure to determine when the costs of FP is high

2. Recall: A ratio of correctly predicted “Animal” to all actual “Animal”, which is a better measure to evaluate the model when there is a high cost of FN
3. F-1 Score: The weighted average of Precision and Recall, which is a better measure to use if we need to seek a balance between Precision and Recall
4. Accuracy: A ratio of correctly predicted objects to the total object, which is a great measure when the dataset is balanced



		Actual Class	
		Animal (1)	Ghost (0)
Predicted Class	Animal (1)	TP	FP
	Ghost (0)	FN	TN

Figure 6: Confusion Matrix

4.5.2 Intersection over Union (IoU)

In object detection tasks, the object in an image (e.g., animals) is detected with a “bounding box” plotted around it. In reality, the (x,y) coordinates of the predicted bounding boxes from a model is extremely unlikely to completely match the (x,y) coordinates of the labeled ground-truth bounding boxes. Therefore, Intersection over Union (IoU) has been widely used to measure the accuracy of the predicted location for the target object in object detection area. The IoU metric computes the intersection over union between the predicted bounding box and the ground-truth bounding box (Figure 7). In general, an IoU score greater than 0.5 is considered a “good” prediction.

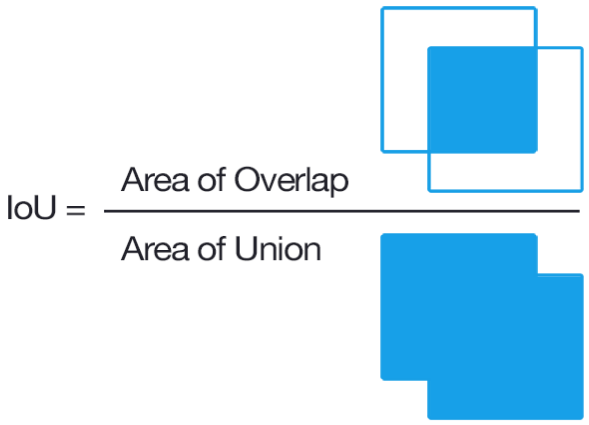


Figure 7: Intersection over Union (IoU)

4.6 Video Detection

Image detection has been researched extensively through the years, comparatively, not so much has been done to video detection. One of the reasons is that video data can be easily transformed into image data if ignoring the audio part. This transformation indicates the only difference between video and image. With an additional component of time, a sequence of images can form a video like the example shown here in Figure 8. As one skims through figure 1 from left to right, it is a sequence of images that has been cropped from a video.

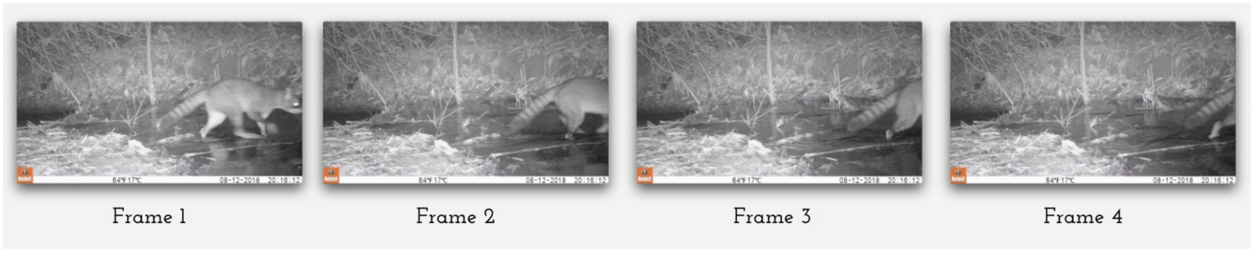


Figure 8. Image representation of a video

Once knowing the only difference, we are able to solve this problem by building the solution on top of image detection models. Referring to Han, Khorrami, and Paine's research paper *Seq-NMS for Video Object Detection*, video data handling will be done through the post-process stage after applying image detection models. This means we need to apply the image object detection model to each of the frames first as shown in Figure 9. The bounding boxes together with the confidence score is the output of the image detection model. Next, we can set up a rule to link the boxes across the frames. Assume we use the rule that says if a box appears in the current frame overlaps with a box in the next frame about 70% of the area, we can link the two boxes.

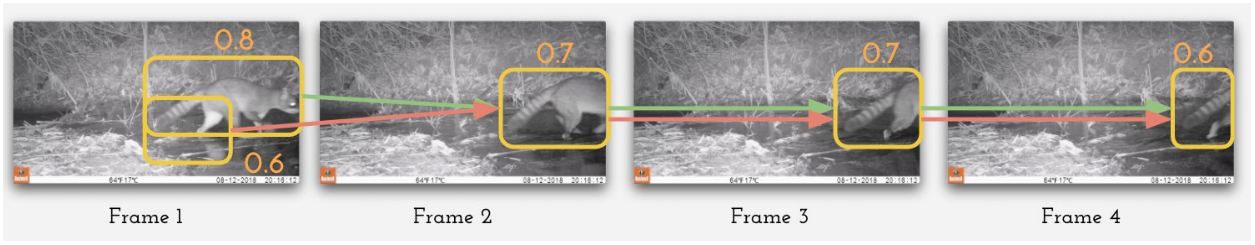


Figure 9. Video Detection Method

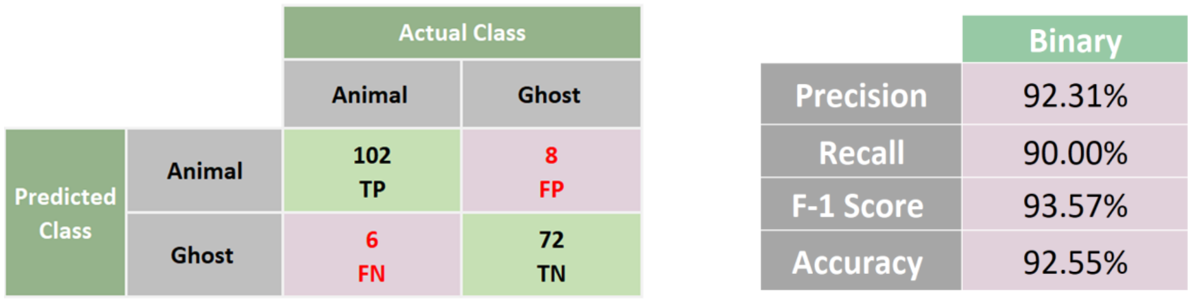
Applying this rule can potentially result in 2 sequences. The first sequence could link the larger bounding box in frame 1 with the rest of the boxes in the later frames as shown in the green arrows. The second sequence could link the smaller bounding box with the rest of the boxes in the later frames as shown in the red arrows. Next, we could apply a rule between the sequences to select only one sequence to be the final result. This rule could be comparing the total confidence score for each of the sequences by summing up the scores of each box. Sequence 1 will get a score of 2.8 and sequence 2 will be 0.2 points shorter. Sequence 1 will be selected as the final output. Another rule could be comparing the average confidence score and it would result in the same output inside this example. Design different rules on how to link the frames and on how to select the final sequence will be the tasks we have to work on extensively to get the best result of the video detection model.

5 Results

5.1 Image Classification

5.1.1 Binary Classification

Figure 10 reports the confusion matrix and a list of key metrics calculated from it for our binary classification model. Among the validation data, 102 out of 108 labeled as "Animal" were predicted correctly, and 72 out of 80 labeled as "Ghost" were predicted correctly. There were 8 "Ghost" incorrectly predicted as "Animal" (Type I error) and 6 "Animal" classified as "Ghost" (Type II error). In our case, the Type I error is preferable to a Type II error because our goal is to help the scientists detect the animals so that the error of missing recognizing an animal in the camp traps can be "dangerous".



		Actual Class	
		Animal	Ghost
Predicted Class	Animal	102 TP	8 FP
	Ghost	6 FN	72 TN

	Binary
Precision	92.31%
Recall	90.00%
F-1 Score	93.57%
Accuracy	92.55%

Figure 10: Confusion Matrix and Related Rates for Binary Classification

While the confusion matrix is used to measure the performance of the binary classifier with a fixed threshold (0.5), the ROC curve examines it over all possible thresholds. As shown in Figure 11, the curve is closer to the top-left corner with AUC score of 0.98. The higher the AUC, the better performance of our model at distinguishing the "Animal" and "Ghost" classes. Based on the metrics above, our classifier works well on binary classification.

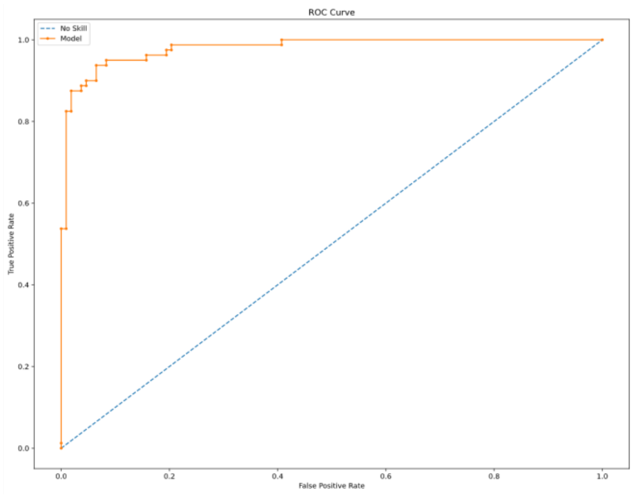


Figure 11: ROC Curve for Binary Classification

5.1.2 Multiclass Classification

We trained and tested our multi-class model based on three different versions of classes (Figure 12). As mentioned in section 5.3, the animal class contains more general classes: “Aves”, “Mammalia”, and “Reptilia”; the animal class-order includes the second level of the taxonomy: “Aves”, “Mammalia”, “Reptilia”, “Rodentia”, “Didelphimorphia”, “Cingulata”. Since categories in the high level contain a broader range of animals, which means more sufficient samples are available in each category, the model with animal class (plus “Ghost”) performed better than the model with animal class-order (plus “Ghost”) as expected. However, if we excluded the “Ghost” and only considered classifying the animal class-order, our model achieved a higher overall accuracy.

		Multiclass (animal class with "Ghost")	Multiclass (animal class-order with "Ghost")	Multiclass (animal class-order without "Ghost")
Micro-average		89.29%	86.16%	89.81%
Weighted-average	Precision	89.40%	86.20%	89.81%
	Recall	89.29%	86.17%	89.81%
	F-1 Score	84.32%	85.90%	89.63%

Figure 12: Performance Comparison of Three Multi-class Models

Particularly, we can see from Figure 13 that by removing “Ghost” from the classes, more species were predicted correctly in each category. For example, in animal class-order with the “Ghost” model, only 2 out of 4 Reptilia were classified as “Reptilia”, an accuracy of only 50%. We checked back the images containing the Reptilia and found that first there are only 20 such images and second this animal is really good at camouflaging itself with the background so that it is reasonable that the model incorrectly predicted it as “Ghost”. In order to make our model less confused by the “noise”, we decided to only choose the 6 animal class-order as the categories for our multi-class models. In the next step, we will join the binary and multi-class models together to make a robust and flexible classification model. Moreover, the training set size is not particularly large (1,478 images), especially for some species, thus a much larger dataset is recommended for generalizable performance.

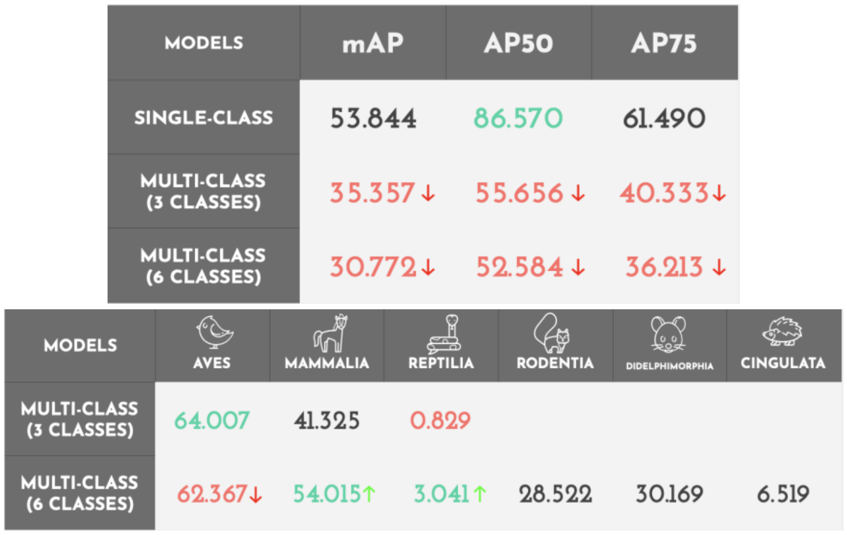
		Actual Class					
		Aves	Cingulata	Didelphimorphia	Mammalia	Reptilia	Rodentia
Predicted Class	Aves	34 -> **35**	0	0	0	0	1
	Cingulata	0	3 -> **4**	0	0	0	1
	Didelphimorphia	0	2	16 -> **19**	0	0	2
	Mammalia	0	0	0	7 -> **8**	0	0
	Reptilia	0	0	0	0	2 -> **4**	0
	Rodentia	1	1	1	0	0	25 -> **27**

Figure 13: Confusion Matrix for Multi-class Model (animal class-order)

5.2 Image Detection

Figure 14 summarizes the results for the different object detection models that we explored, in terms of average precision statistics. Average Precision (AP) is defined as the area under the Precision-Recall curve across different thresholds of prediction confidence scores, which can also be calculated average precision across different IOU thresholds. For object detection, one of the most commonly used evaluation metrics is the COCO metrics, which includes the mean AP (mAP) that is averaged AP across 11 different IOU thresholds, AP at IOU threshold of 50% and 75%. Amongst 3 object detection models, the Single-class model performs the best with average precision of 86.570% calculated with IOU threshold of 50%.

For the multi-class models, the performance is sub-optimal, as all of the AP statistics decreased in comparison to that of the single-class model. In addition, the AP at per class level is also less than ideal, with only the “Ave” class having an AP above 50%. This shows that the models are only able to detect birds relatively accurately, while failing to detect the other animal classes and orders, which may be due to the abundance of birds images in the dataset, with the animal body completely in frame most of the time. The models are performing extremely badly at detecting reptiles, with less than 1% and 5% AP respectively, which could be due to the lack of samples in the dataset.



MODELS	mAP	AP50	AP75
SINGLE-CLASS	53.844	86.570	61.490
MULTI-CLASS (3 CLASSES)	35.357↓	55.656↓	40.333↓
MULTI-CLASS (6 CLASSES)	30.772↓	52.584↓	36.213↓

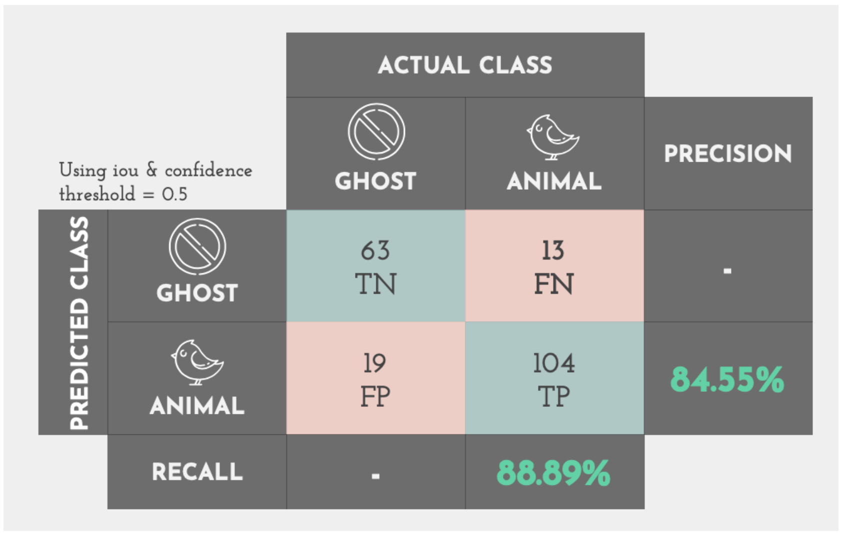
MODELS	AVES	MAMMALIA	REPTILIA	RODENTIA	DIDELPHIMORPHIA	CINGULATA
MULTI-CLASS (3 CLASSES)	64.007	41.325	0.829			
MULTI-CLASS (6 CLASSES)	62.367↓	54.015↑	3.041↑	28.522	30.169	6.519

Figure 14: Average Precision (AP) statistics for Image Detection Models (Top: Mean AP, AP at IOU=50%, IOU=75% for 3 object detection models; Bottom: Per-class AP for multi-class models)

5.2.1 Single-Class Detection

With the single-class object detection model, we further examine the results with the confusion matrix calculated using both IOU and confidence threshold set at 50%.

As shown in Figure 15, the precision and recall are both relatively high above 80%, while more specifically the model is able to detect an animal when in an Animal image 84.55% of the time, while correctly detecting an animal 88.89% of the time. However, there are still quite a significant number of false positives (the model detected an animal where there is not) and false negatives (the model missed out on an animal instance in the image), which we further explored using samples from the validation set.



		ACTUAL CLASS		
	Using iou & confidence threshold = 0.5	GHOST	ANIMAL	PRECISION
PREDICTED CLASS	GHOST	63 TN	13 FN	-
	ANIMAL	19 FP	104 TP	84.55%
	RECALL	-	88.89%	

Figure 15: Confusion Matrix for Single Class object detection model at IOU and confidence threshold = 0.5

From the mis-classified samples, we seem to be able to generalize some common sources of errors. For the false positives, there are mainly 2 categories, in which the model is either mis-detecting leaves as animals or correctly detecting the presence of the animal but wrongly locating it. Both of these seem to be due to the lack of contrast in the background due to either low lighting condition of similar color scheme. For the false negatives, there are mainly 4 categories of errors. For images where the animal is either partially in frame, such as with only a tail, or blocked by plants or leaves, the model appears to be having trouble detecting the animal instances. Another potential source of error could be due to the camouflaged nature of the animal where it is intentionally immersed into the background, which makes it difficult to detect. Lastly, there are many images with poor exposure quality, either over-exposed or under-exposed, which significantly reduced the amount of information in the image for the models to learn from.

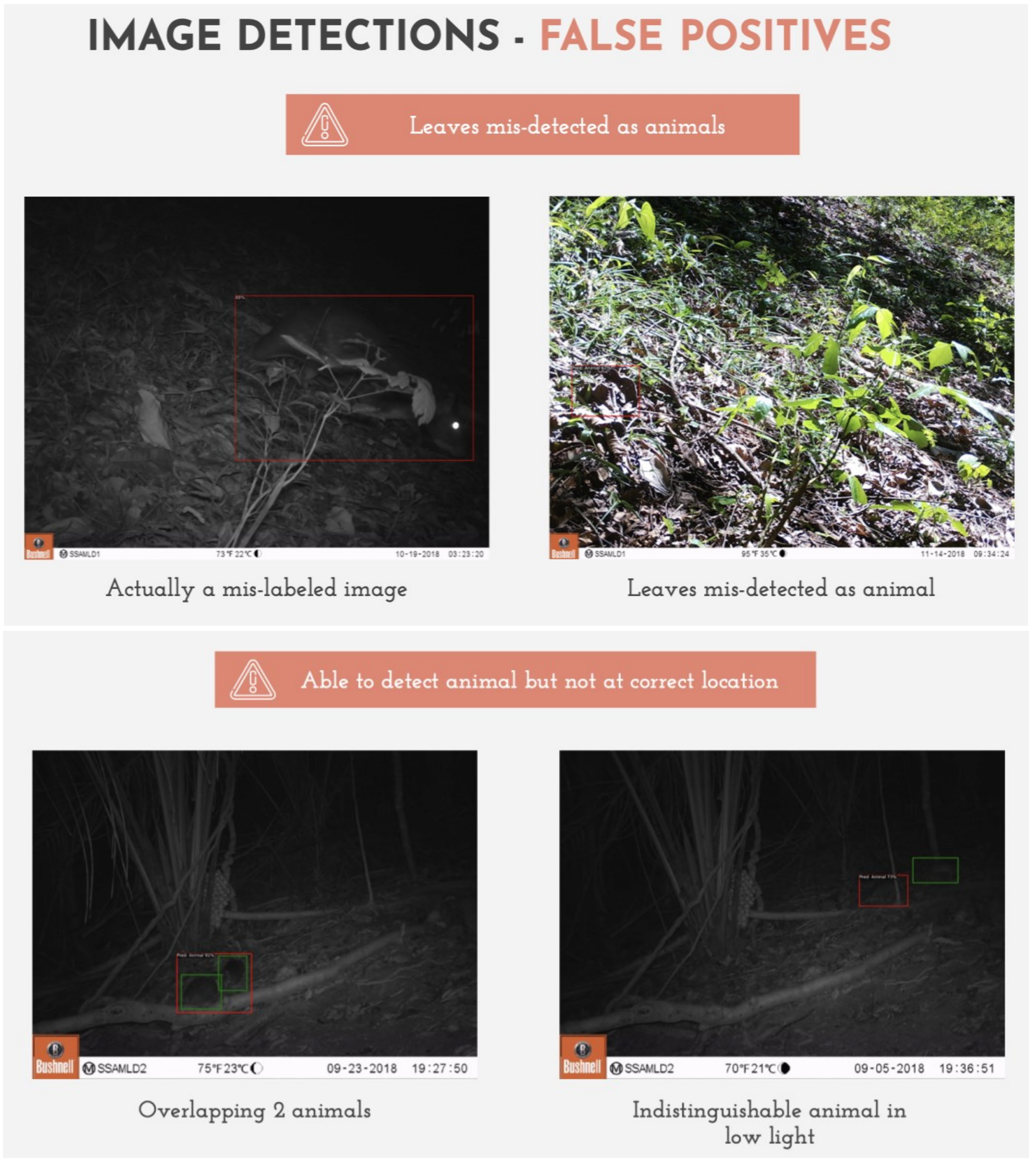


IMAGE DETECTIONS - FALSE POSITIVES

Leaves mis-detected as animals

Actually a mis-labeled image

Leaves mis-detected as animal

Able to detect animal but not at correct location

Overlapping 2 animals

Indistinguishable animal in low light

Figure 16: False Positives Samples from the validation set for the Single-Class object detection model

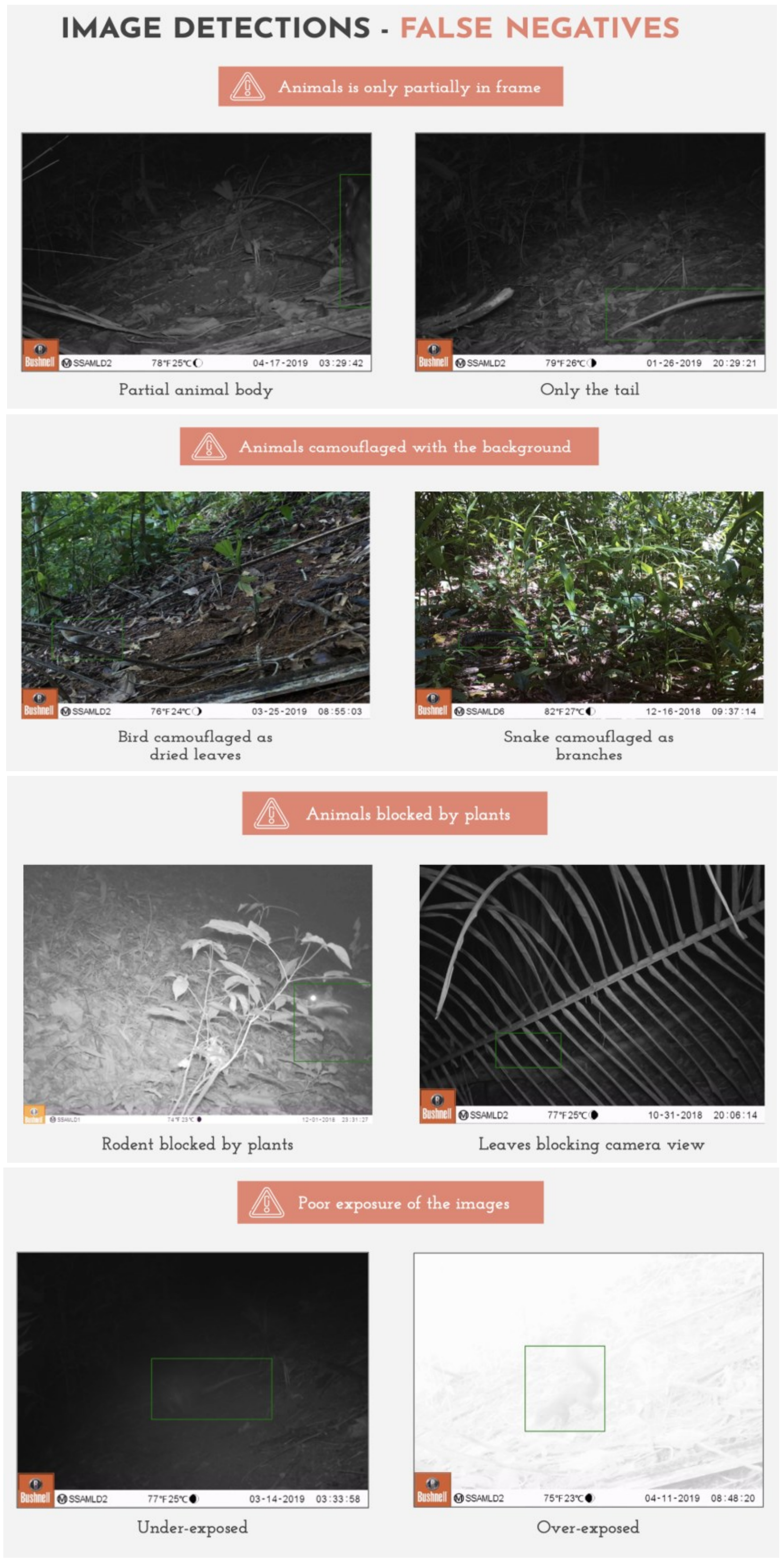
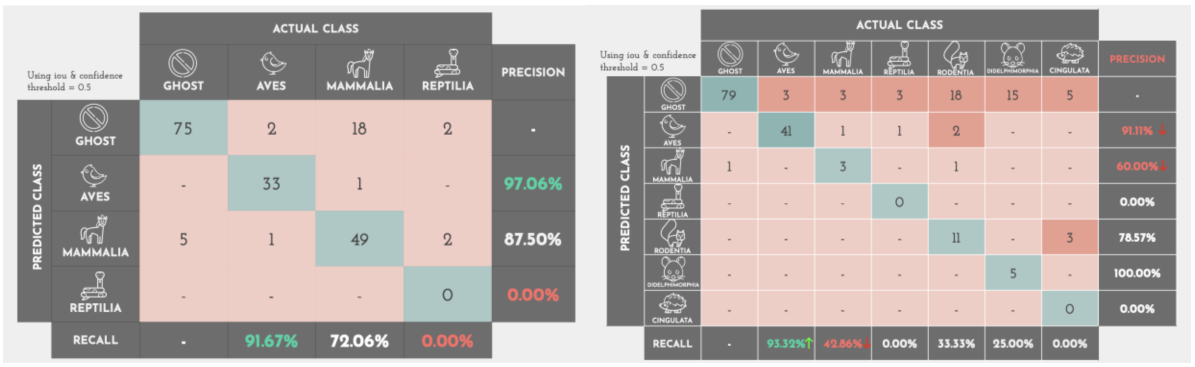


IMAGE DETECTIONS - FALSE NEGATIVES
Animals is only partially in frame
Partial animal body
Only the tail
Animals camouflaged with the background
Bird camouflaged as dried leaves
Snake camouflaged as branches
Animals blocked by plants
Rodent blocked by plants
Leaves blocking camera view
Poor exposure of the images
Under-exposed
Over-exposed

Figure 17: False Negatives Samples from the validation set for the Single-Class object detection model

5.2.2 Multi-Class Detection

Similarly with the single-class model, we also explored the confusion matrix for the multi-class models, at the same confidence and IOU threshold of 0.5, as shown in Figure 18. The findings are consistent with that from the AP statistics that the models are only performing well at detecting birds (“Aves”) at precision and recall of above 90%. More experimentations need to be explored to improve the performance of the multi-class detection models.


Using iou & confidence threshold = 0.5

PREDICTED CLASS \ ACTUAL CLASS	GHOST	AVES	MAMMALIA	REPTILIA	PRECISION
GHOST	75	2	18	2	-
AVES	-	33	1	-	97.06%
MAMMALIA	5	1	49	2	87.50%
REPTILIA	-	-	-	0	0.00%
RECALL	-	91.67%	72.06%	0.00%	

Using iou & confidence threshold = 0.5

PREDICTED CLASS \ ACTUAL CLASS	GHOST	AVES	MAMMALIA	REPTILIA	RODENTIA	DIDELPHIMORPHIA	CINGULATA	PRECISION
GHOST	79	3	3	3	18	15	5	-
AVES	-	41	1	1	2	-	-	91.11% ↓
MAMMALIA	1	-	3	-	1	-	-	60.00% ↓
REPTILIA	-	-	-	0	-	-	-	0.00%
RODENTIA	-	-	-	-	11	-	3	78.57%
DIDELPHIMORPHIA	-	-	-	-	-	5	-	100.00%
CINGULATA	-	-	-	-	-	-	0	0.00%
RECALL	-	93.32% ↑	42.86% ↓	0.00%	33.33%	25.00%	0.00%	

Figure 18: Confusion Matrix for Multi-Class object detection models at IOU and confidence threshold = 0.5 (Left: Multi-class with 3 classes; Right: Multi-class with 6 classes)

6 Conclusions

The main goal of this project is to help Saving Nature effectively automate the process of identifying important visual data. At the current stage with the limited data source, this project has successfully constructed high performing models dealing with the image data. This current achievement sets up a solid foundation for our next stage on video detection. Our video detection model will be built upon the image detection model with an extra post-processing step. The current data sample size does not allow us to continually improve our image models. However, the problem of having a relatively small sample size will be resolved soon. There will be more labeled images together with labeled video data available to us in the next semester. Aside from the currently limited sample size, human-made error on labeling the data will be something we have little control over. This means that even with a sufficient amount of sample size, our final model’s performance will hit a glass ceiling at some point before reaching 100% accuracy. With this limitation, our work will still help Saving Nature save a great amount of effort on manually processing the visual data. Continued efforts are needed from us in the upcoming semester to make the most accurate model we can for Saving Nature.

Citations

Han, Wei, et al. “Seq-NMS for Video Object Detection.” *ArXiv*, 2016, arxiv.org/pdf/1602.08465v3.pdf.

Hoang, T.M.; Nguyen, P.H.; Truong, N.Q.; Lee, Y.W.; Park, K.R. Deep RetinaNet-Based Detection and Classification of Road Markings by Visible Light Camera Sensors. *Sensors* **2019**, *19*, 281.

Li, Yixing; Ren, Fengbo. Lightweight retinanet for object detection. arXiv preprint arXiv:1905.10011, 2019.

Norouzzadeh,Mohammad Sadegh;Nguyen,Anh; Kosmala, Margaret; Sawnson, ALexandra; Palmer, Meredith S.; Packer, Cragi; Clune, Jeff.Automatically identifying, counting, and describing wild animals in camera-trap images with deep learning. National Academy of Sciences 2018, 115, 25, E5716--E5725.

Wu, Yuxin; Kirillov, Alexander; Massaand, Francisco; Lo, Wan-Yen; Girshick Ross. Detectron2.https://github.com/facebookresearch/detectron2. 2019.