Identifying Humans in Drone Footage from Local Beaches

Final Report

Burns, Echelle
Feb 2020

1. INTRODUCTION

Throughout the years, researchers have used a variety of different technologies to address the movement patterns and behavior of particular shark species. In more recent years, however, shark researchers have begun to use drones and helicopters to capture footage of shark species to answer questions like: *How do these animals interact with one another when they are not being actively tagged or followed?* Using drone data presents a lot of issues when it comes to data processing, such as 1) having a suitable reference item to accurately predict the sizes of objects in the drone's field of view, 2) figuring out the orientation of the drone in order to georeference the resulting footage, and 3) deciding whether videos or stills should be collected. However, although drone footage is relatively new to the field of marine biology, data scientists have been analyzing video footage and images for quite a while.

Shark attacks from a variety of species, including White Sharks (*Carcharodon carcharias*), Tiger Sharks (*Galeocerdo cuvier*), and Bull Sharks (*Carcharhinus leucas*) are heightened in the media and instill terror in much of the human population. Although shark researchers are aware that shark attacks do not happen as frequently as the media portrays, no research projects have addressed how frequently sharks and humans actually interact with each other, and how many times those interactions result in a shark bite. A graduate student at California State University, Long Beach (CSULB) is using video footage from drones and helicopters in order to answer this question, by identifying not only when White Sharks and humans are in the water at the same time, but also by categorizing how the sharks respond when approached by their human counterparts. However, gathering drone footage for such a project inevitably produces a lot of raw footage (several 10+ minute videos per day) that likely contains neither sharks nor humans. Going through each frame of a video one-by-one may take up a significant amount of one's time, leaving little time for true data analysis techniques. Therefore, the goal of this project is to take stills from drone video footage and train a deep neural network to identify how many humans are present within each frame. Although this work will be catered to a specific researcher and a specific problem, the methods from this project may also be adapted for researchers who are attempting to answer similar research questions or have similar streams of data.

2. MATERIALS AND METHODS

2.1 Data Sources

A total of 2,465 drone images (3840x2160 pixels) were provided by the California State University, Long Beach Shark Lab. These images were taken at local beaches along the southern California coast and include features such as: humans that are participating in a variety of different activities (e.g. walking on the beach, wading in the shoreline, swimming, paddleboarding, kayaking, or surfing), White Sharks swimming near the ocean's surface, or other forms of marine life (e.g. dolphins, stingrays, kelp). The images were not yet labeled when they were received. Example images can be found in: SpringBoard-Capstone2/Data.

2.2 *Image Labeling and Splitting*

Images were labeled by creating an interactive python script that would display images on the screen and allow the user to place dots (5 px radius) on top of locations where humans were present (SpringBoard-Capstone2/Data_Processing/Labeling_Images). For ease of labeling, the images were resized to 960x540 pixels. This size allowed the labeler to see the entire image at one time, and still allowed for easy identification of human subjects. There was no differentiation between different forms of human activity for this project, but future work should try to implement different labeling methods for different activities (e.g. walking on the beach, wading in the shoreline, swimming, paddleboarding, kayaking, or surfing).

The raw version of each image was then viewed in true color, grayscale, and HSV color formats (SpringBoard-Capstone2/Data_Processing/ DataWrangling_PhotoContrasts, Figures 1-3) to determine which color scale would yield the highest contrast. Labeled images were saved as black and white arrays, where areas of black represented no humans and areas of white (circles with a 5 px radius) represented the location of humans (Figure 4). Both true color raw images and labeled images were then split into 25 smaller images (192x108 pixels) in order to run the model more efficiently (SpringBoard-Capstone2/Data_Processing/ DataWrangling_ImageSlicing, Figure 5). Once split, the images were saved in true color, HSV, and grayscale color schemes. The total number of people in each split image was generated from the labeled image using *skimage*'s *measure.label* function before it was passed to the model.

2.3 *Data Wrangling*

Since data were images, no other data wrangling procedures were implemented.

2.4 *Exploratory Data Analysis*

Of the 2,465 drone images that were provided, only 521 included at least one human subject. Therefore, only images that originally had a human in the frame were used for model training and testing. Once split into 25 smaller images, the dataset was comprised of 13,025 images (some with 0s present) to use for the model. Exploratory data analyses were run on the full size images (960x540 pixels) and split images (192x108 pixels) on this set of data, primarily to determine the underlying distribution of the number of people in each image (SpringBoard-Capstone2/Exploratory_Data_Analysis/Descriptive_Statistics).

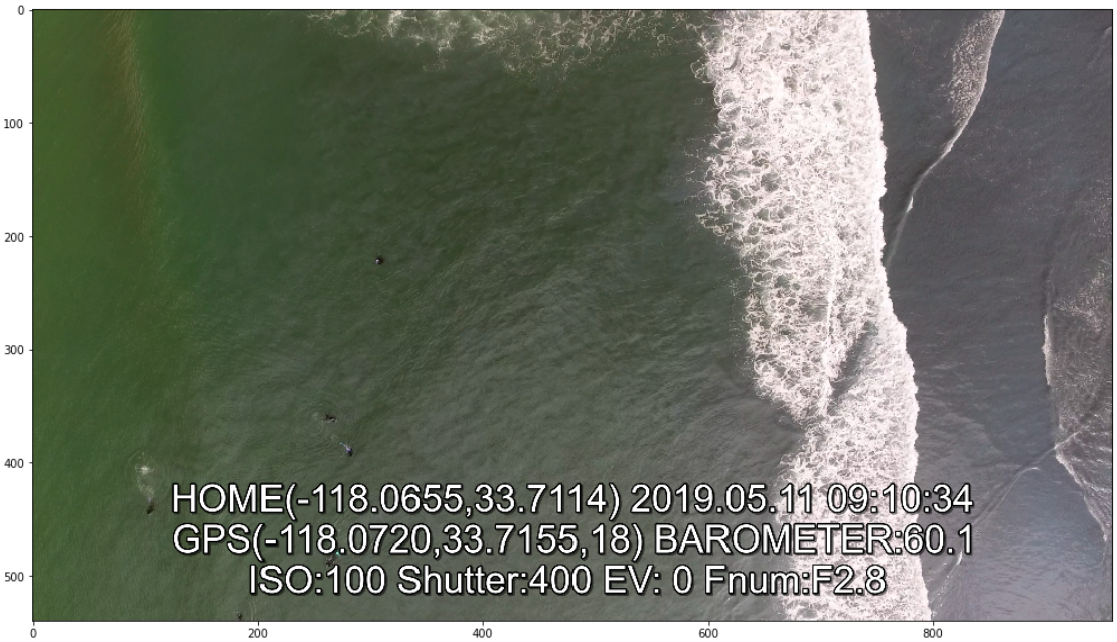

0
100
200
300
400
500
0
200
400
600
800
HOME(-118.0655,33.7114) 2019.05.11 09:10:34
GPS(-118.0720,33.7155,18) BAROMETER:60.1
ISO:100 Shutter:400 EV: 0 Fnum:F2.8

Figure 1. True color image of drone footage that includes human subjects (swimmers) on the left of the figure.

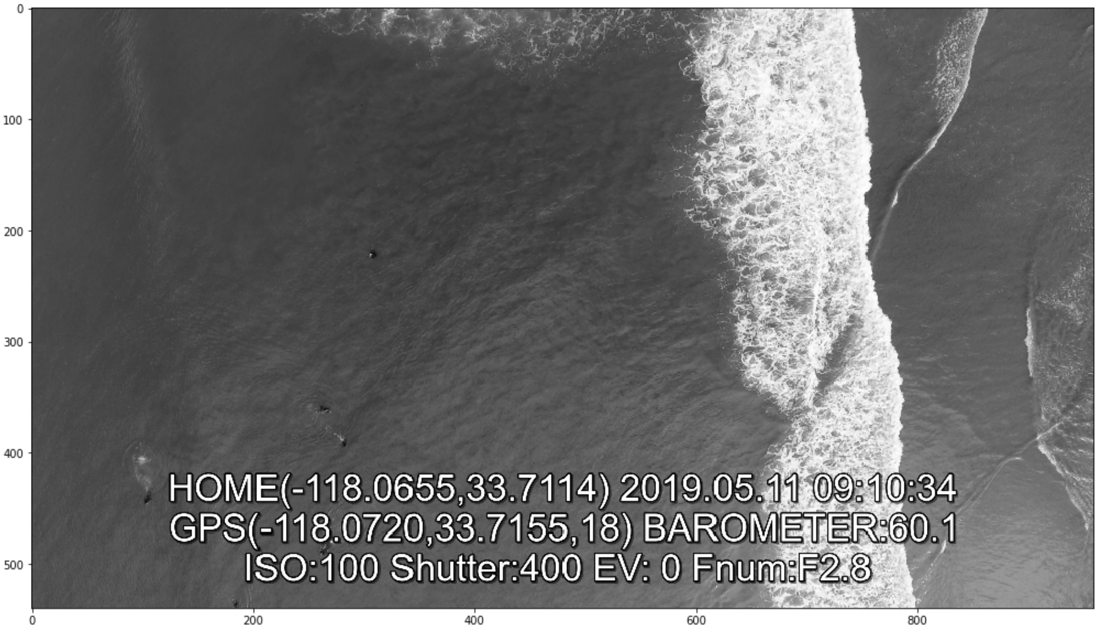

0
100
200
300
400
500
0
200
400
600
800
HOME(-118.0655,33.7114) 2019.05.11 09:10:34
GPS(-118.0720,33.7155,18) BAROMETER:60.1
ISO:100 Shutter:400 EV: 0 Fnum:F2.8

Figure 2. Grayscale image of drone footage that includes human subjects (swimmers on the left of the figure). Content is the same as Figure 1.

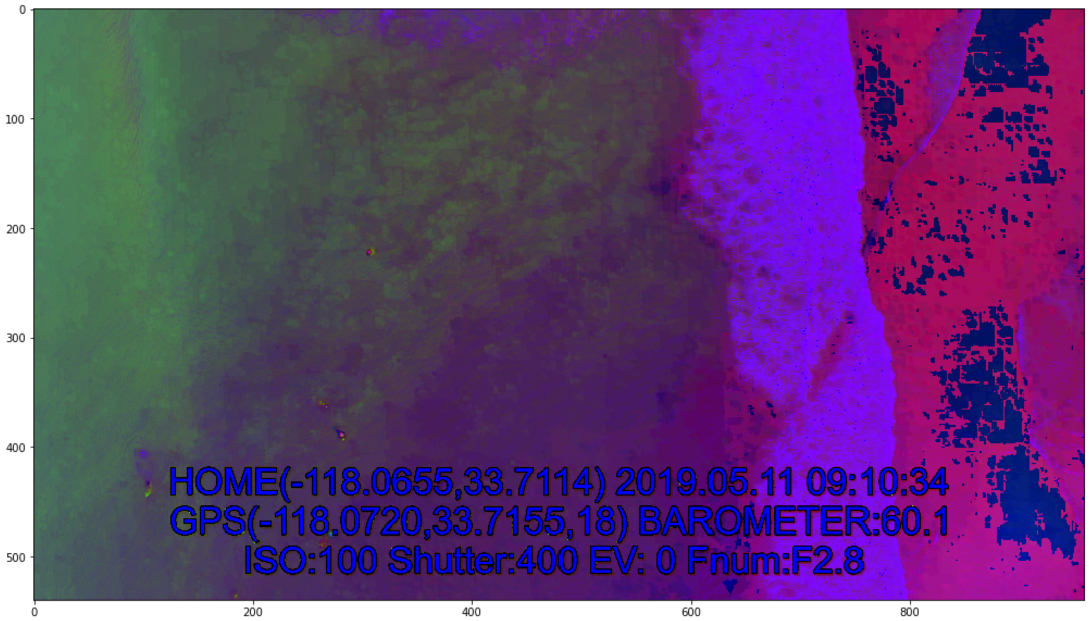

0
100
200
300
400
500
0
200
400
600
800
HOME(-118.0655,33.7114) 2019.05.11 09:10:34
GPS(-118.0720,33.7155,18) BAROMETER:60.1
ISO:100 Shutter:400 EV: 0 Fnum:F2.8

Figure 3. HSV image of drone footage that includes human subjects (swimmers on the left of the figure). Content is the same as Figure 1.

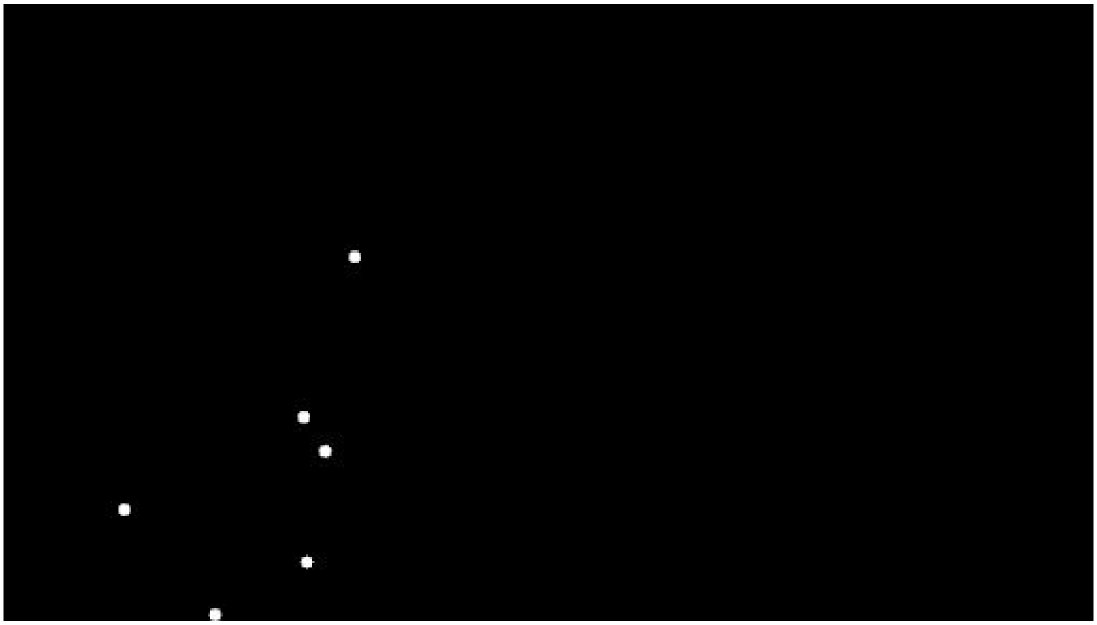


Figure 4. Labeled image of drone footage that includes human subjects, where all black indicates no humans and white dots (5 px radius) indicates the location of a human. Content is the same as Figure 1.

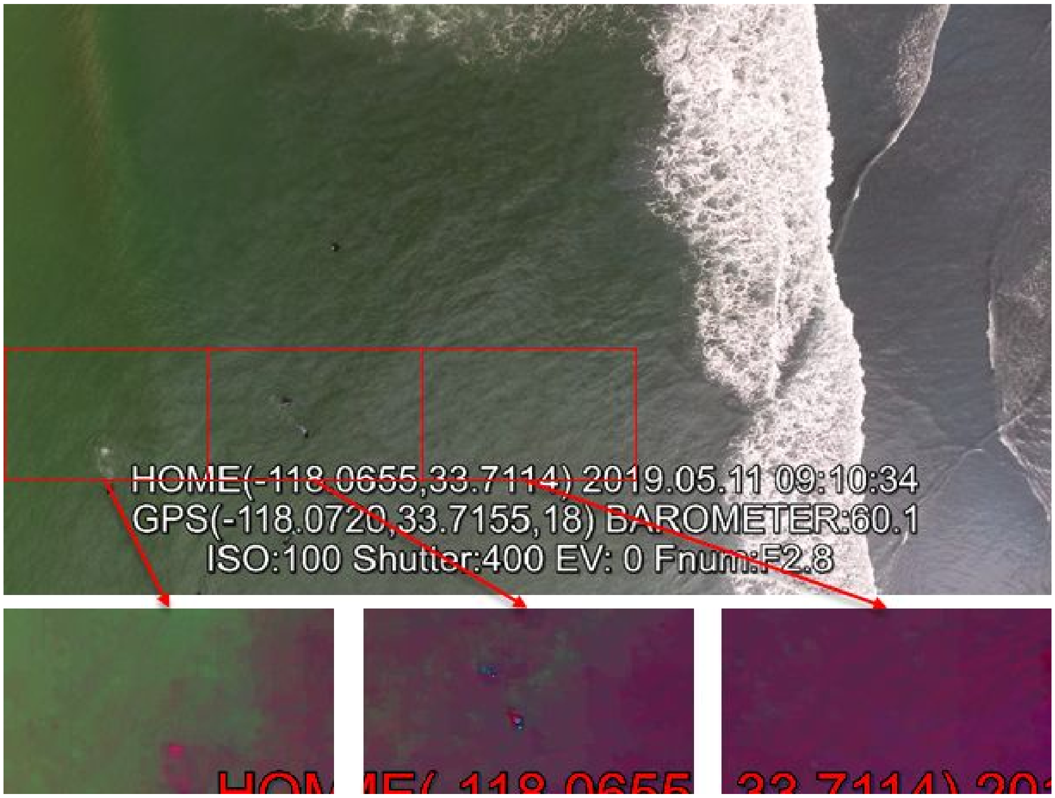

HOME(-118.0655,33.7114) 2019.05.11 09:10:34
GPS(-118.0720,33.7155,18) BAROMETER:60.1
ISO:100 Shutter:400 EV: 0 Fnum:F2.8

Figure 5. Three images from a raw image (960x540 pixels) that was split into 25 smaller images (192x108 pixels) and converted to HSV. Content is the same as Figure 1.

2.5 Convolutional Neural Networks

Prior to constructing a Convolutional Neural Network, images were divided into testing and training datasets, where 20% of all files were saved for the testing group. In addition, 20% of files in the training dataset were saved to validate the model performance at the end of each epoch. A generator function was created to batch upload a number of images at a time and partially train the model, in hopes to allow for faster processing speed. This generator grabs a default of 10 images, groups the raw images together and extracts the number of people in each image from the corresponding labeled images. The result that was passed to the Convolutional Neural Network was a tuple of an array of the color values for each pixel in each image file and an array list of the number of humans in the images.

Several different Convolutional Neural Networks were tested and the best model was chosen using brute-force methods, in which one aspect of the model was changed and the model was re-run. All models were sequential models and had four convolutional layers paired with four max pooling layers. After the fourth max pooling layer, a flattening layer was added and was followed by two dense layers paired with two dropout layers. The final layer was a dense layer with a single node.

Models were changed in a variety of ways, with the most prominent change being adding a normalization layer before the convolutional layers. Different model runs can be found in the Neural_Network folder. The number of kernels and the sizes of the kernels in the convolutional layers were changed, as well as differences in the number of nodes in intermediate dense layers and the activation function of the final dense layer. In addition, the models were run on color images, grayscale images, and HSV converted images, as well as on full-sized images and image splits. When using HSV images, differences in model performance were compared against models that used all color channels or a single color channel (each channel was assessed separately). Lastly, different activation functions were tested across all layers. Tested functions included: ReLU (known for being the most efficient activation function), sigmoid (known to create a smooth gradient and make clear predictions), linear (known to allow multiple outputs greater than 1), tanh (known to be centered around 0 and ideal for datasets that are strongly skewed to negative or positive values), hard sigmoid (known to be computationally faster than a normal sigmoid activation function), PReLU (known to allow for the learning of a negative slope), and swish (a new activation function developed by Google that is said to perform better than the typical ReLU function).

2.6 Model Evaluation

The final model that was chosen performed the best on the training dataset, reaching the lowest loss function, and also performed consistently well on the validation dataset. The data for this model were not balanced prior to model training, and therefore included an unequal number of images that contained no people and that contained one or more people (hereby called ‘unbalanced model’). An additional model was trained and validated on a dataset that used an approximately equal number of images that contained no people and one or more people (hereby called ‘balanced model’). Data were balanced by using a random sample without replacement of the larger dataset (no people present) to match the smaller dataset (1+ people present). The residuals were calculated for each model using the testing datasets that were set aside before the model was developed.

The final, untrained model and generator function can be found in the master folder for this repository as model.py and generator.py, respectively. The trained model with an unbalanced training dataset is saved as model.hd5, whereas the trained model with a balanced dataset is saved as balanced_model.hd5.

3. RESULTS

3.1 Exploratory Data Analysis

All of the full size images used for analyses had one person, and the frequency of images with more than one person decreased as the number of people in the image increased (Figure 6). Most images that were included in this dataset were comprised of images with 7 humans or less. Of the split images, however, over 90% had 0 people in the frame and approximately 7% of the images had only one person in the frame (Figure 7).

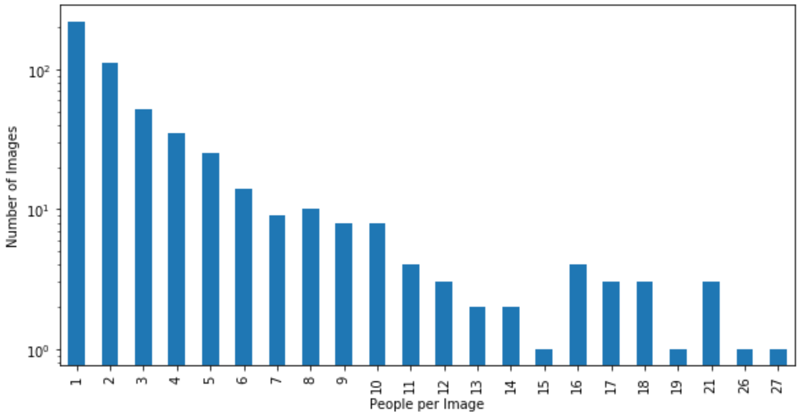


Figure 6. The frequency of raw images (980x540 pixels) that had varying numbers of human subjects. Data were taken only from frames that included at least one person.

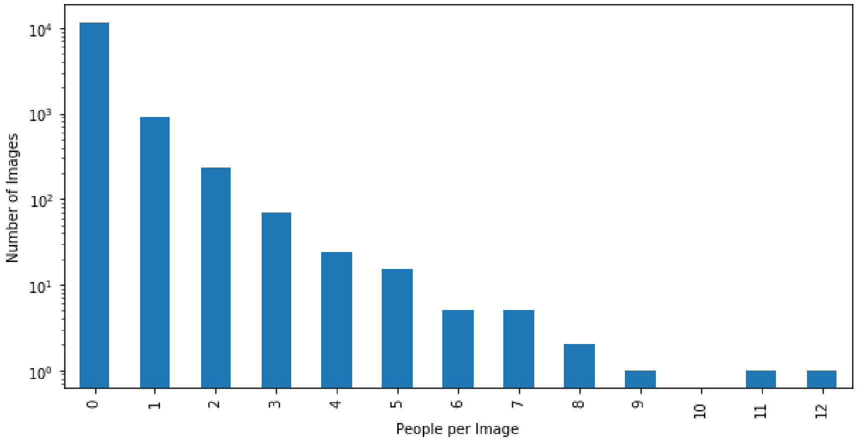


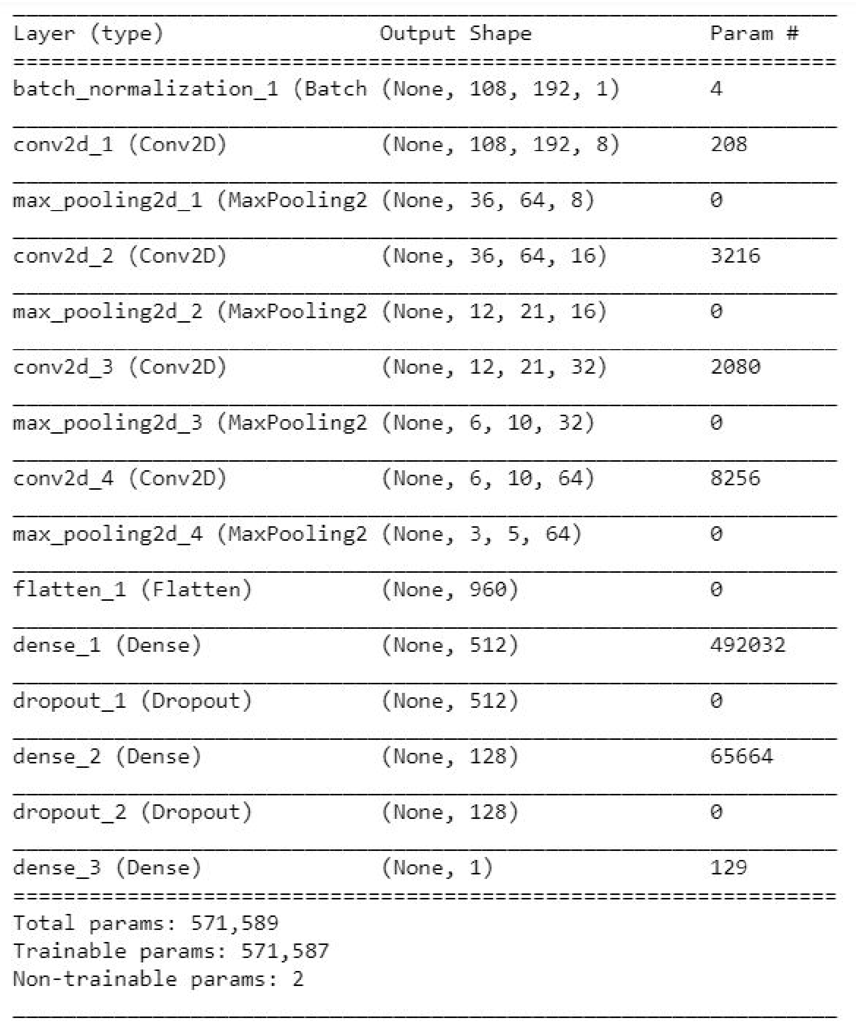
Figure 7. The frequency of images that were split (25 images of 192x108 pixels per raw image) that had varying numbers of human subjects. Raw images included at least one person.

3.2 Convolutional Neural Networks

The final model used the second HSV channel for the images and began with a normalization layer (Table 1, Figure 8). The first convolutional layer included 8 kernels with a kernel size of 5x5 pixels and a ReLU activation function. The first max pooling layer had a kernel size of 3x3 pixels. The second convolutional layer included 16 kernels with a kernel size of 5x5 pixels and a ReLU activation function. The second max pooling layer had a kernel size of 3x3 pixels as well. The third convolutional layer included 32 kernels with a kernel size of 2x2 pixels and a ReLU activation function. The third max pooling layer had a kernel size of 2x2 pixels. The fourth convolutional layer included 64 kernels with a kernel size of 2x2 and a ReLU activation function. The fourth max pooling layer had a kernel size of 2x2 pixels. After the fourth convolutional layer, there was a flattening layer. Following the flattening layer was a dense layer with 512 nodes and a ReLU activation function. The first drop-out layer had a value of 0.2. The second dense layer consisted of 128 nodes and had a ReLU activation function. The dropout layer after this layer had a dropout rate of 0.2 as well. Finally, the last dense layer in the model had a single node and a linear activation function. The model was optimized by looking for the minimum mean squared error value for both the testing and the validation dataset.

The model framework described above yielded a minimum mean squared error of ~ 0.02 for the unbalanced training dataset after nearly 40 epochs with a batch size of 30 images (Figure 9). Throughout each of these epochs, the mean squared error for the unbalanced validation dataset was relatively consistent, around 0.2. Alternatively, the balanced model yielded a minimum mean squared error of ~ 0.1 for the training dataset after nearly 35 epochs with a batch size of 30 images (Figure 10). The mean squared error for the validation dataset varied between 1.0 and 1.4 for all epochs in the balanced model. Although these models appeared to reach convergence after nearly 35-40 epochs, users who use a different set of training images may have different results.

Table 1. Model Summary for the best performing model at identifying human subjects in drone images



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
batch_normalization_1 (Batch (None, 108, 192, 1)       4         
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 108, 192, 8)       208       
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 36, 64, 8)         0         
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 36, 64, 16)        3216      
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 12, 21, 16)        0         
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 12, 21, 32)        2080      
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 6, 10, 32)         0         
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 6, 10, 64)         8256      
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 3, 5, 64)          0         
_________________________________________________________________
flatten_1 (Flatten)          (None, 960)               0         
_________________________________________________________________
dense_1 (Dense)              (None, 512)               492032    
_________________________________________________________________
dropout_1 (Dropout)          (None, 512)               0         
_________________________________________________________________
dense_2 (Dense)              (None, 128)               65664     
_________________________________________________________________
dropout_2 (Dropout)          (None, 128)               0         
_________________________________________________________________
dense_3 (Dense)              (None, 1)                 129       
=================================================================
Total params: 571,589
Trainable params: 571,587
Non-trainable params: 2
_________________________________________________________________
```

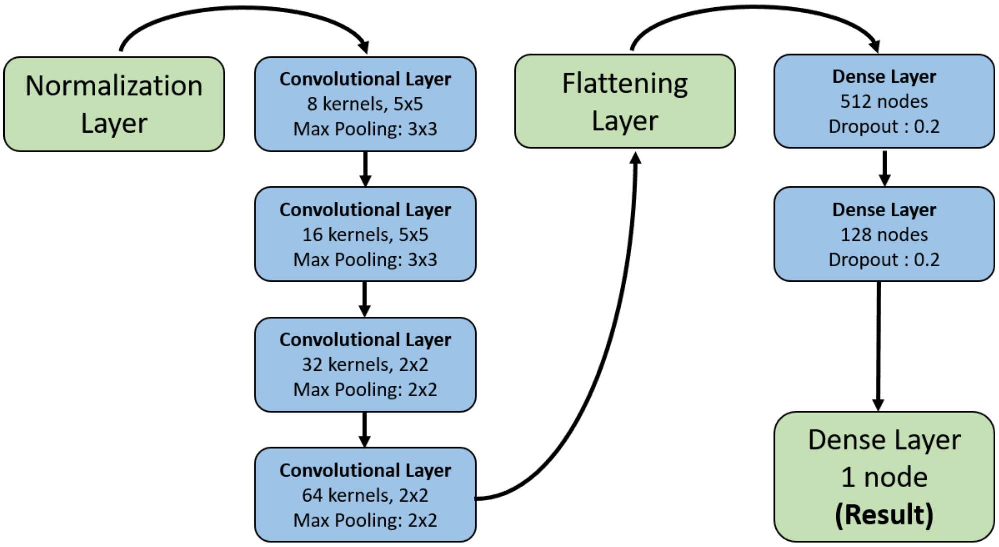


Figure 8. The model architecture that was used to identify human subjects in drone images.

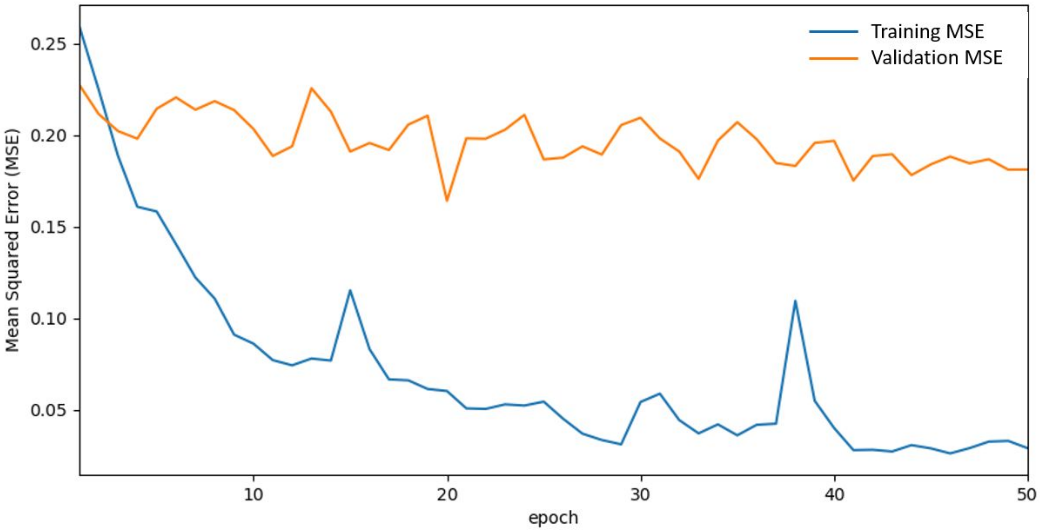


Figure 9. The mean squared error for the training dataset (blue line) and validation dataset (orange line) from the best performing model across 50 epochs. The training dataset was not balanced prior to model training and the mean square error appears to converge at around 40 epochs.

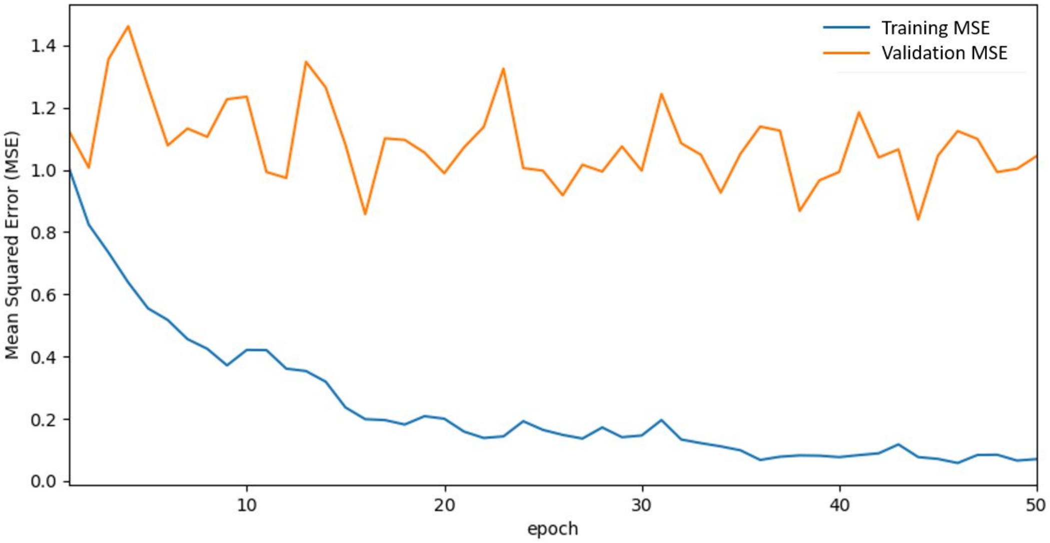


Figure 10. The mean squared error for the training dataset (blue line) and validation dataset (orange line) from the best performing model across 50 epochs. The training dataset was balanced prior to model training and the mean square error appears to converge at around 35 epochs.

3.3 Model Evaluation

The unbalanced model appeared to over-estimate the number of people in a frame in images with no humans present, but under-estimate the number of people in a frame when one or more people were present (Figure 11). This is particularly noticeable when there are five or more humans in an image, when the model estimates less than 1.5 humans present in all cases. In addition, there appeared to be a skewed distribution of residuals, with a very high number of residuals falling between 0 and -1 (Figure 12).

The balanced model showed a similar trend, in which the model over-estimated the number of people in a frame when there were no humans present, but under-estimated the number of people in a frame when there were more than three people present (Figure 13). However, it appears that this model performed better than the unbalanced model when one, two, or three people were present, as the distribution of points appear to scatter randomly across the 1x1 relationship line. Compared to the unbalanced model, however, there appears to be greater variation in the predicted values that were generated for each observed value. In addition, although the residuals appear to have a slight skew towards the left (several instances of residuals around -1), residuals for the balanced model are most frequent around 0 (Figure 14).

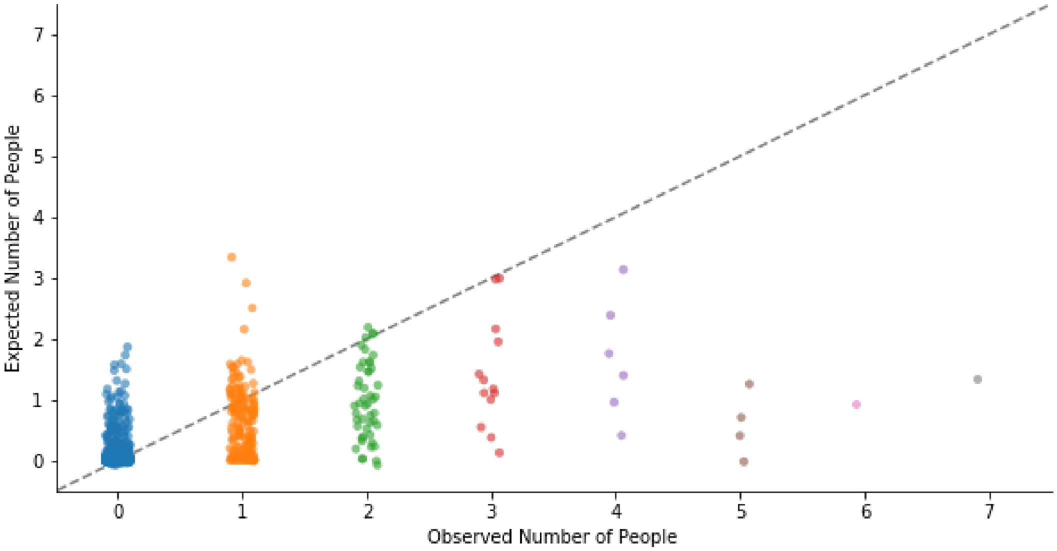


Figure 11. The observed number of people in labeled images in the testing dataset (x-axis) compared to the unbalanced model's prediction (y-axis). The gray dashed line represents a 1 to 1 ratio, which would indicate a model that performed perfectly.

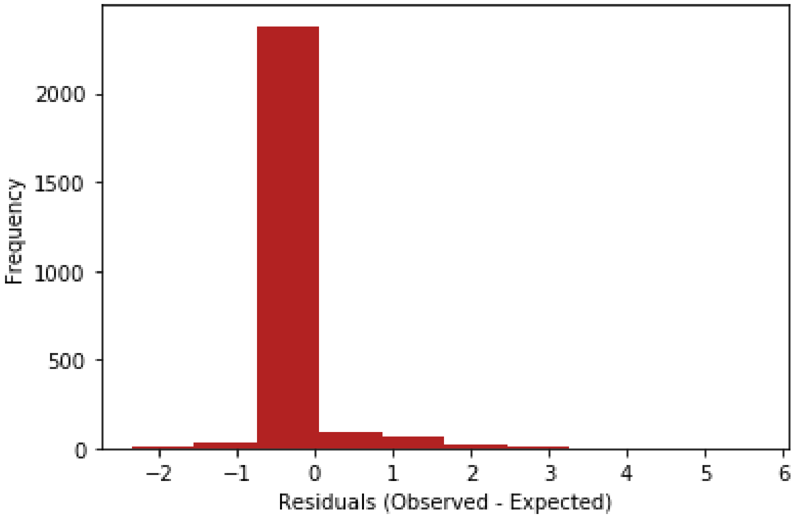


Figure 12. The frequency of residuals calculated from the unbalanced model using the testing dataset.

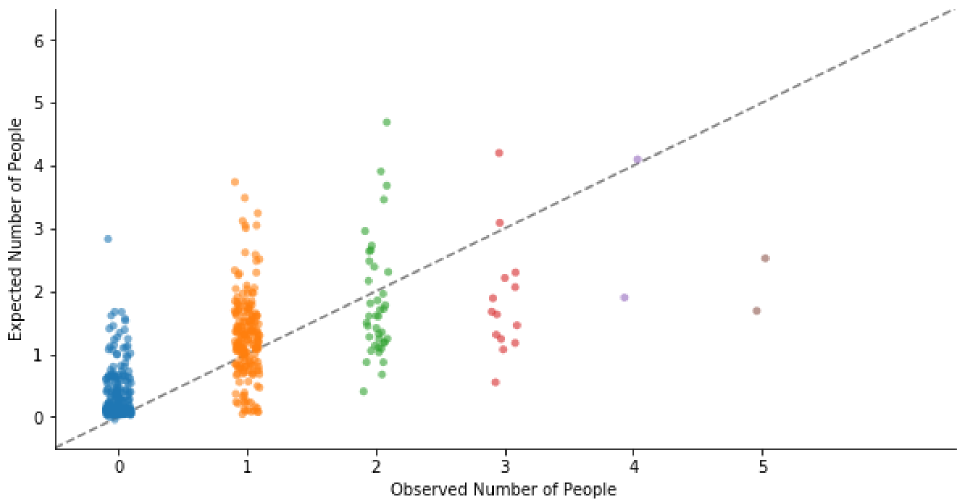


Figure 13. The observed number of people in labeled images in the testing dataset (x-axis) compared to the balanced model's prediction (y-axis). The gray dashed line represents a 1 to 1 ratio, which would indicate a model that performed perfectly.

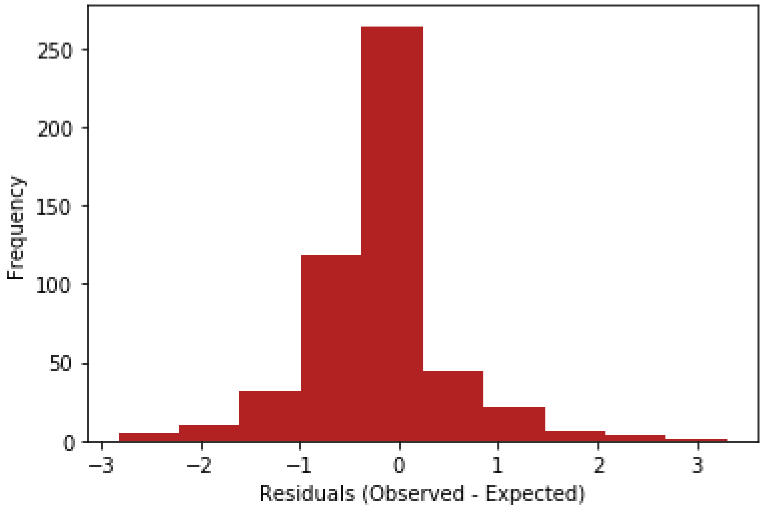


Figure 14. The frequency of residuals calculated from the balanced model using the testing dataset.

4. DISCUSSION

4.1 Sources of Error

The model presented here performs relatively well at identifying how many humans are present in a clipped drone image, but tends to perform worse when more people are present. The main sources of difference in the model predictions and true dataset may arise from the inherent differences among people in the drone images themselves. I hypothesize that the model may have performed better if it were only looking for one particular recreational type (walking on the beach, wading, paddleboarding, etc). There is a vast difference in how a human looks when he or she is walking on the beach compared to when he or she is lying on a surfboard waiting to catch another wave. Although humans can see that both images contain humans doing different activities, the abstract idea of a human likely changes as humans are seen performing such different tasks, making the underlying ‘idea of a human’ more difficult to understand.

Another explanation for the difference in the predicted number of people in an image and the true number of people in an image may arise from splitting the original images into 25 smaller ones. It is possible that, once split, several images contained parts of humans, but not the entire beings. When the images are split and the number of people are generated from the labeled images, incomplete circles (labels for people) still represent an entire person in the models. Therefore, it may be possible that the training, validation, and testing datasets were introducing images that did not have an entire ‘idea of a person,’ causing the model to under-estimate the number of people compared to the number of people that the labeled image suggested. Upon a brief inspection of the first 200 clipped images with people present, it appears that about 18% of the images have at least one labeled human that is not completely in the frame. Of the 18%, however, it appears that most images only contain the one, partially labeled human. In both the balanced and unbalanced models, it appears that instances that include 1 individual perform the best. Therefore, partially labeled humans may be adding to some noise in the dataset that is most noticeable when there is more than one human in the frame. If this is a concern, it may be useful to consider another, more conservative measuring algorithm to generate the number of labeled people in a frame.

4.2 Future Work

Since the model may be reaching a limit in performance based on different human activities, it may be useful to instead create several different models, with each trained to recognize a certain activity (i.e. a model to identify the number of people surfing and a model to identify the number of people walking on the beach). For such models to work, the training dataset will need to be labeled again, only focusing on labeling humans who are performing a particular activity. The current layers that have been defined in the present model may be a good start in training a new model that is focused on identifying a particular human activity. Alternatively, there is a potential for transfer-learning, in which the pre-trained model presented here may be used as a baseline model that is re-trained on new images that have been labeled for specific activities. This may help the new models to reach convergence faster than re-training an entirely new model. It is also important to note that not all images that were used to train the model in this project have been added to the GitHub repository. Several images used in the training dataset included human subjects at different orientations and sizes. Having such diversity in the images used in the training dataset will be vital to producing a useful model in the future.

Another project that would be useful to implement in the future would be a model that identifies the location of humans in a drone image. Instead of a dense layer that would provide a number of people in an image, the new model would predict the center coordinates of where a human is expected to be located. Such a model would be ideal for researchers who are trying to discover behavioral trends between and among different subjects; such outputs would make it easy to calculate the distance between subjects and determine whether those individuals are close enough together to be ‘interacting’ with one another.

4.3 Choosing the Balanced or Unbalanced Model

The primary purpose of this project was to be able to broadly identify how many humans were present in a particular image, and both models were relatively successful in doing so. The unbalanced model produced a smaller mean squared error for the training and the validation datasets when compared to the balanced model. The unbalanced model was also more consistent in its estimates of how many people were present; the variance of predicted values compared to observed values appeared to be lower in the unbalanced model than it was in the balanced model. However, the distribution of residuals were much more skewed to the left (towards -1) for the unbalanced model compared to the balanced model, suggesting that the unbalanced model tended to severely under-estimate the true number of people that were actually present in the image. Alternatively, the balanced model showed residuals closer to 0, indicating that most of the time, the estimates were fairly accurate, even though the mean squared error was greater (high variance in predicted values). Therefore, it is likely that with more training data, the balanced model may be most useful in the long run.

As more data are collected by the CSULB Shark Lab, the balanced model can continue to be trained, and eventually become even more effective at predicting the number of people in an image. Although this model shows direct benefits to students in the CSULB Shark Lab, it can also help researchers that are collecting similar datasets. Ultimately, such a model can save researchers time from manually labeling and sorting their drone images into groups that include human subjects. With this saved time, these researchers can allocate more resources to actively analyzing what the drone images can teach us about animal behavior.