Capstone 2 - Milestone Report II
Burns, Echelle
Feb 2020

Identifying Humans in Drone Footage from Local Beaches

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 itself presents a lot of issues when it comes to data processing, such as having a suitable reference item to accurately predict the sizes of objects in the drone's field of view and figuring out the orientation of the drone in order to georeference the resulting footage. 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 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*

Deep Neural Networks

Prior to constructing a Convolutional Neural Network, image splits from larger images that contained at least one human subject were divided into testing and training datasets, where 20% of all files were saved for the testing group. In addition, 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 raw image file and an array list of the number of humans in the raw 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 activator functions were tested across all layers. Tested functions included: ReLU (known for being the most efficient activator 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 activator), PReLU (known to allow for the learning of a negative slope), and swish (a new model developed by Google that is said to perform better than the typical ReLU function).

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 testing/validation dataset.

3. *Results*

Deep Neural Networks

The Sequential model that performed the best at identifying human subjects in the split drone images yielded a minimum mean squared error of ~ 0.05 for the training dataset after nearly 35 epochs with a batch size of 30 images (Figure 1). Throughout each of these epochs, the mean squared error for the testing dataset was relatively consistent, around 0.17. The final model used the second HSV channel for the images and began with a normalization layer. 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 training dataset. The model appeared to reach convergence after nearly 35 epochs for this dataset, but may differ for users who use a different set of training images.

Figures

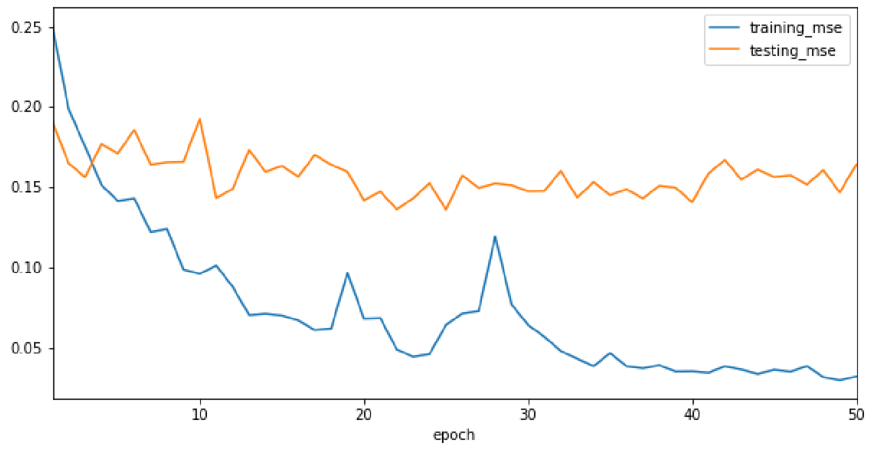

Figure 1. The mean squared error for the training dataset (blue line) and testing dataset (orange line) from the best performing model across 50 epochs. The training dataset mean square error appears to converge at around 35 epochs.