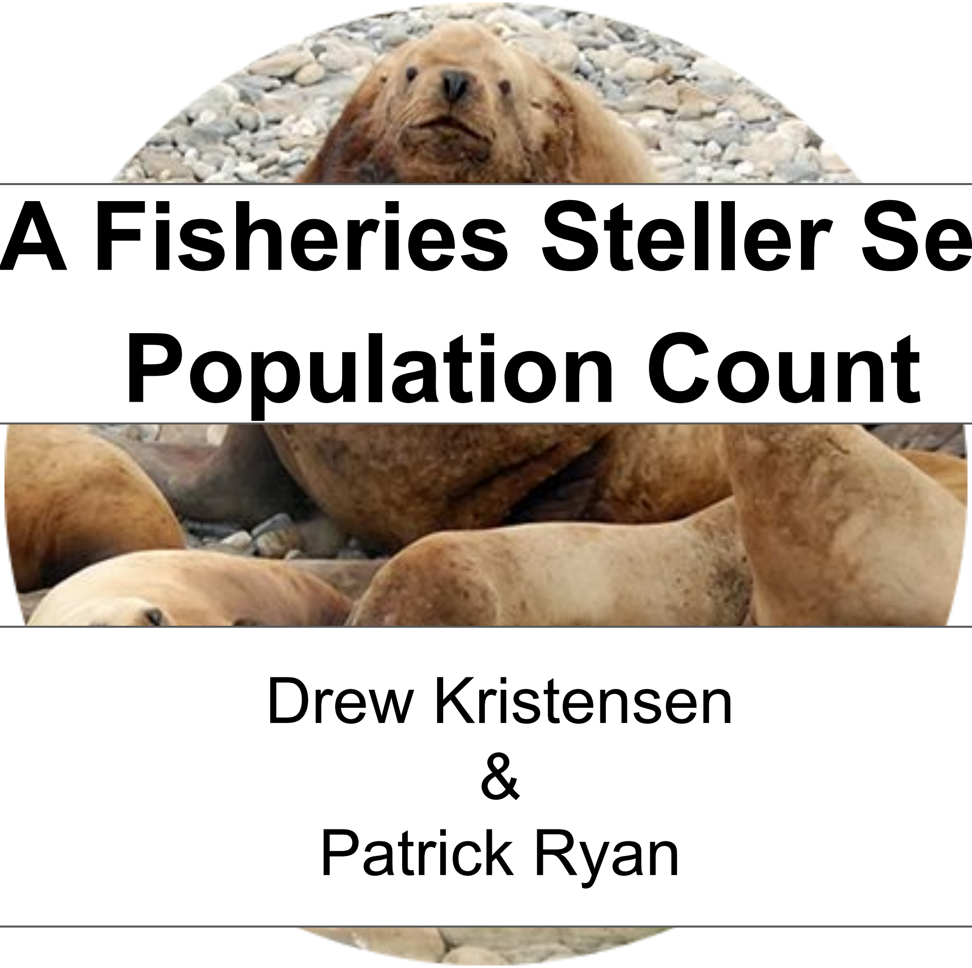


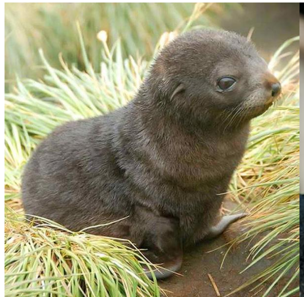
NOAA Fisheries Steller Sea Lion Population Count

Drew Kristensen
&
Patrick Ryan

Motivation

Problem: Steller sea lions in the western Aleutian Islands have declined 94 percent in the last 30 years. Only way to collect accurate data on sea lion population is to take aerial photos. To save the population scientists need to track and have accurate numbers on the total population. But this can take scientist months to count all the sea lions in the thousands of photos they need to go through.

Solution: A neural network algorithm could be used that could count the seal lions in the photos saving time for scientists so they can focus on saving the sea lion population. DO IT FOR THE BABY SEALS!!!



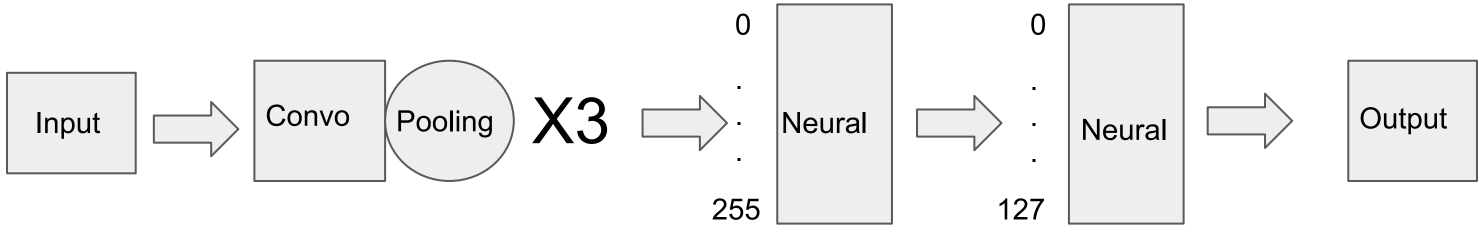
Technical Problem

Enumerate all sea lions present in an image taken from above

- Identify which portions of the image contained sea lions
- Find a method to pass over the image to be ensure we would detect a sea lion in an image
- Minimize the amount of background that our network detected as a sea lion
- Kaggle testing input is 96.5 GB...zipped... (FYI that is big)

How we did it

- Made a neural network of sequential layers. Three convolutional layers each followed by a pooling layer. Into 2 neural layers, first 256 neurons, the second 128 neurons. Resulting in one final output layer. Each feeds sequentially into the next
- Filtered the image in 64 by 64 pixel ranges, shift the window by 48 pixels over to ensure we don't miss any seals. Found this to the best size that was fast but still gave use accurate data.
- Had to set up computers in Thompson to train the input.



Training

- Training turned out to be one of our biggest challenges. (surprise!)
- Full data input was to big to run on the computer we set up in a reasonable time.

Future solutions:

- Set up dedicated training computer days in advance.
- Switch to running on GPU instead of CPU
  - CNN have been shown to run much faster on GPU's
  - Keras can be implemented to run on GPU

Making the Choice

Chose because CNNs have been shown to demonstrate low error on image classification problems

- Highly recommended for image recognition and classification due to how convolution, ReLu (non-linearity) and pooling layers work.

Easy(ish) to implement using python frameworks

- Keras
- Theano (Backend)
- A lot of good documentation online

Results

Kaggle provided two data sets, a large and small. Had to use smaller data set to measure our CNN due to time constraints. Large data set would have hopefully given better results.

Test Set:

- Achieved 91% accuracy.
- Our goal was 90%

Small Official Dataset:

- Precision: 80%
- Accuracy: 50%