CS583 Final Project

iWildCam 2019 challenge: Wildlife photography classification

Malik Mubeen

Dec. 1, 2019

1 Introduction

Biologists all over the world have been increasingly utilizing Camera Traps, or Wild Cams, that automatically perform the collection of large quantities of image data. These images are used to monitor biodiversity and population density of animal species. Biologists have been pushing for technology that facilitates the automation of species classification with these kinds of Camera Traps; however, expanding these models across various geographic areas has proven difficult, as the biodiversity in the training data might not reflect that of the new region. That is, we observe the problem where test data is vastly different from the data on which these automatic image classifiers are trained.

This particular challenge uses training data from the American Southwest, and test data from the American Northwest. While there is some overlap of species between the two regions, this composition of the training and test data will showcase this problem, and will test our final model's ability to generalize to the test data set.

2 Data

This data was obtained from the Kaggle competition: iWildCam 2019 - FGVC6; Categorize animals in the wild. The training set for this challenge contains 196,157 images from 138 different locations in Southern California, and 153,730 images from 100 locations in Idaho. Each image is of resolution 1024x747, and contains varous metadata including the photographer name, datetime stamp, and location.

Each image in the training data is categorized under one of the following labels:

empty: 0, deer: 1, moose: 2, squirrel: 3, rodent: 4, small mammal: 5, elk: 6, pronghorn antelope: 7, rabbit: 8, bighorn sheep: 9, fox: 10, coyote: 11, black bear: 12, raccoon: 13, skunk: 14, wolf: 15, bobcat: 16, cat: 17, dog: 18, opossum: 19, bison: 20, mountain goat: 21, mountain lion: 22

Below is a sample of images included in this challenge:

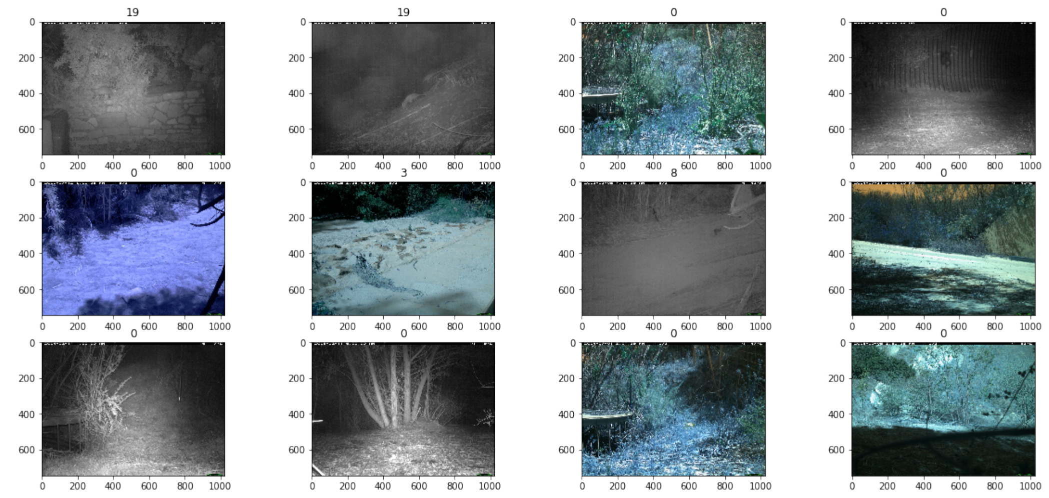
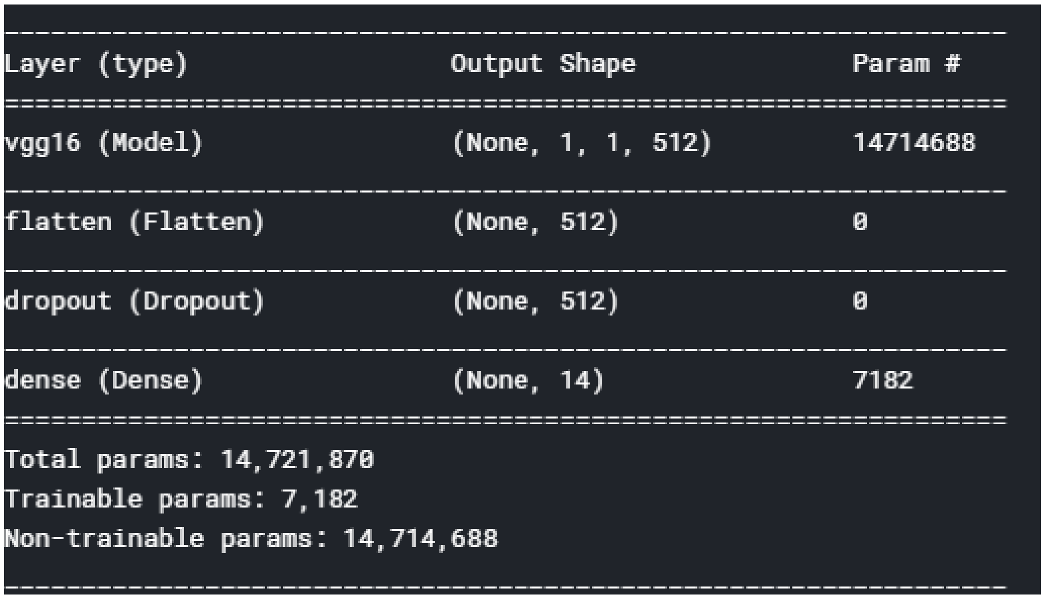


Fig. 1: Sample of wildlife images

3 Analysis

In order to more efficiently process the images, each image in the training and test set was first resized to 32x32. While this does inhibit the model's ability to learn features and classify images, the upside of the model being able to fit in a reasonable amount of time is well worth the decreased precision.

A convolutional neural network was fit to the data. The VGG16 architecture was used as the base of our model. These layers were frozen, and a flatten layer, 33% dropout layer, and final softmax layer were used as the top.



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
vgg16 (Model)                (None, 1, 1, 512)         14714688
_________________________________________________________________
flatten (Flatten)            (None, 512)               0
_________________________________________________________________
dropout (Dropout)            (None, 512)               0
_________________________________________________________________
dense (Dense)                (None, 14)                7182
=================================================================
Total params: 14,721,870
Trainable params: 7,182
Non-trainable params: 14,714,688
_________________________________________________________________
```

Fig. 2: Summary of final model used

This model was trained for 8 epochs on the training set, which yielded a final validation set accuracy of 78.29% (see attached code for full set of model hyperparameters).

4 Comparison to baselines

The training data set was highly imbalanced: the majority of images contained no animals. Thus, we can use the percentage of data points classified as 0, empty, as a naive baseline to evaluate our model. The percentage of images in the validation data set with no animals was 67.65% (i.e., a classifier predicting no animals in each image would have an accuracy of 67.65%). Our model's validation accuracy of 78.29% outperforms this naive baseline accuracy.

Another naive model with just a flatten layer (essentially functioning as a reshaper), and a dense softmax layer was fit on the data. The final validation accuracy was 3.03%. This highlights the effectiveness of the VGG16 convolutional architecture.

5 Test set results and conclusion

Using the model parameters fit on the training and validation data, predictions were made on the test set. The test set accuracy was 11.9% with 68% of the test data, a far lower value than the validation set accuracy. This could be due to a number of reasons. The classifier we fit on the training and validation data was heavily weighted towards images with no animals, and so the proportion of images with no animals in the test set could have been far lower, which would result in our classifier deeming far too may images as having no animals.

Other neural network architectures could have more favorable results for the training and validation data, which could possibly help boost the test set accuracy. Other architectures that could be examined include VGG19 or ResNet.

Finally, it is possible that the biodiversity of the test set is simply too different from the training set for a model to generalize to the two effectively. As evidenced by our relatively higher training and validation set accuracy, deep learning architecture can be used to identify wildlife effectively, but one must be wary of applying a trained model in a one-size-fits-all fashion. In this particular case, the huge discrepancy between training and test set accuracy suggests that the American Southwest and American Northwest perhaps should not be compared to one another in terms of automatic wildlife classification.