MSDS462 Final Project Summary

Brian Tieskoetter

1 Project Description and Goals

This project was an exploration of a strategy of using a system of layers of small, simple, focused computer vision models to achieve similar performance to what can be done with complex general purpose models that were trained on huge generalized datasets. The smaller models can be trained using transfer learning on small image datasets, and since they are specialized and focused on a specific task they can be small enough to deploy directly on an edge device with little power consumption.

This project implemented a two-layer model system on an Amazon Web Services (AWS) Deeplens camera. Because it is easy to attract large numbers of birds conveniently, the system was written to generate a list of the species of birds in my back yard. An object detection model is used to detect the presence of birds and where they are; then the image immediately around the bird is isolated, scaled, and passed to the classification model for classification. Notebooks with project code are here.

The Deeplens does not support multiple models directly. An alternate strategy was to use an Intel Movidius Neural compute Stick to support the second model; but software incomaptablilties also prevented calling the neural stick software from the DeepLens camera code by the end of the class. The end solution was to implement the object detection model on the camera and write the isolated bird images to an AWS S3 bucket; then an AWS Lambda function was triggered when the file was written to S3 that called the classification model.

AWS DeepLens sample models were used as a baseliine for comparison. The example object detection model was unable to identify birds or squirrels almost 100% of the time when aimed at my back yard; it would occasionally identify a chair or plant but nothing else. The example bird classification model was worse, with zero success.

2 Implementation

The project had two phases: use the camera to gather training images from my yard directly, then train final models with those images for better accuracy.

Object Detection Model

The original plan was to retrain the object detection model for better accuracy on the objects in my yard. To support this, the full camera image was cropped and the camera lambda was modified to save captured images to an S3 bucket. However, just cropping the images dramatically improved the accuracy of the object detectin model and there was little value in retraining. That model was kept as is. Whenever a bird is detected, the routine returns an identification and a bounding box. For each object

detected, if it is a bird, dog, or cat the full image is cropped again to a region slightly larger than the bounding box. Then it is scaled to 224 x 224 pixels, converted to jpeg, and saved to an s3 bucket.

The camera frame is cropped to 640 x 640 before passing to the object detection routine to get a reasonable compromise between the field of view and minimal compression to get the isolated bird images to 224 x 224 pixels.

Classification Model

The captured bird images in the S3 bucket were manually sorted into species and combined with a subset of images from the Caltech UCSD CUB_200_2011 image data set to get a total of 16 categories with one 'no bird' category. A transfer learning process was run on the ResNet 50 model on Sagemaker. To keep the model small, only 18 model layers were kept. After transfer learning, the model could attain about 70% accuracy on training and validation sets with this data. I think further improvement is unlikely with this data because subtle differences between species are lost when the images are scaled to 224 x 224 pixels.

System Implementation

The routine on the camera was largely reused from the image collection phase of the project. When birds are detected, the bird is isolated in the image and sent to the S3 bucket. A trigger was placed on the S3 bucket to call a lamba function that downloads that file, passes it to the classification routine, and then writes the results to a log file also in an S3 bucket. This will let me assess the effectiveness of the system over time.

3 Results and Future Actions

As the vagaries of nature go, I have only had one type of bird in my yard since deploying for tes so results are anectodal; but the model appears to be >90% accurate identifying that bird. This is much better than the original example at much lower cost.

The next step is to work out software incompatabilities to support a second model directly on the device with the Movidius chip; this seems mainly due to a difference in Python versions. Supporting more than one model in the device memory (i.e. CPU and GPU) is intriguing but would probably require me to write my own code to load the models and do inference.

I can also implement a similar cropping process to the CUB-200-2011 data set as I did on my gathered images to improve classification accuracy. I think the model is limited because subtle differences between bird species become invisible when the images are shrunk to the 224 x 224 pixel size.

Wah C., Branson S., Welinder P., Perona P., Belongie S. “The Caltech-UCSD Birds-200-2011 Dataset.” Computation & Neural Systems Technical Report, CNS-TR-2011-001.