

Steve Greenberg - Capstone Proposal

Machine Learning NanoDegree - December 18, 2017

Introduction

My Capstone Project will attempt to classify twelve different species of plants by image analysis. This project will utilize the dataset from Kaggle's [Plant Seedlings Identification](#) competition.

Domain Background

There is a long history of using quantitative analysis to differentiate between species of plants. Wikipedia highlights the [Iris Flower Dataset](#) as one of the [classic data sets](#) used to teach statistics. More recently, a [Flowers dataset has been used to introduce students to Transfer Learning on TensorFlow](#).

Classifying plant species present an obvious benefit and challenge. Similarly looking plants can sometimes require dramatically different treatments. Harmful, invasive species should be removed. Beneficial species should be cultivated and supported. Humans are often only able to differentiate between the two if they have uncommon domain expertise.

Aarhus University in Denmark has [made this dataset available for the public](#). They write:

A database of images of approximately 960 unique plants belonging to 12 species at several growth stages is made publicly available. It comprises annotated RGB images with a physical resolution of roughly 10 pixels per mm. To standardise the evaluation of classification results obtained with the database, a benchmark based on f1 scores is proposed.

Their motivation is to "provide researchers a foundation for training weed recognition algorithms".

More details are provided in the authoritative link to the data - <https://arxiv.org/abs/1711.05458>

Kaggle is re-hosting the dataset as a competition to give it greater exposure.

My Motivation

One side of my yard has a mix of ivy and bindweed. It's critical to distinguish between the two because the ivy is desired, and the bindweed is an invasive plant which can take over lawns. Nonetheless, the two are quite similar from a distance:



from

<https://owlcation.com/stem/The-Hedge-Bindweed-or-Morning-Glory-An-Invasive-Plant>



from <http://www.wisegeek.com/what-is-ivy.htm>

So I can appreciate how useful a solution to this problem would be!

Another motivation is that the **type** of problem appeals to me because it will help in my professional development. I want to tackle a neural networks problem that can take advantage of public cloud resources. Image Classification problems lend themselves to CNNs and can perform well on GPUs.

I manage a team of engineers at Google who build models on TensorFlow and run them on Google Cloud Platform. My team's work can be found in these two Github repos:

- [GoogleCloudPlatform/ml-on-gcp](https://github.com/GoogleCloudPlatform/ml-on-gcp)
- [GoogleCloudPlatform/cloudml-samples](https://github.com/GoogleCloudPlatform/cloudml-samples)

My rationale for completing this nanodegree has been to improve my ability to lead my team. Tackling the Plant Seedlings Identification problem will require me to use Neural Networks. I will use TensorFlow and run the training on Google Cloud Platform. I intend to run my training both in the managed environment of Google Cloud ML Engine and on Google Compute Engine. In both cases, I will use nvidia GPUs.




Problem Statement





The problem is to classify an image of a seedling into one of twelve categories. Evaluation of a solution will be done by measuring the mean multi-class F1 score run against a test dataset.

Datasets and Inputs

Kaggle is hosting both the training and test datasets for this challenge.

The training dataset ([link](https://vision.eng.au.dk/plant-see-dlings-dataset/)) consists of 4,762 labeled images from each of the twelve categories.

English Name (Pictures from https://vision.eng.au.dk/plant-see-dlings-dataset/)	EPPO code	Number of Training Samples
Maize 	ZEAMX	222
Common wheat 	TRZAX	222
Sugar beet 	BEAVA	386
Scentless Mayweed	MATIN	517

		
Common Chickweed 	STEME	612
Shepherd's Purse 	CAPBP	232
Cleavers 	GALAP	288
Charlock	SINAR	391

		
<p>Fat Hen</p> 	CHEAL	476
<p>Small-flowered Cranesbill</p> 	GERSS	497
<p>Black-grass</p> 	ALOMY	264
<p>Loose Silky-bent</p> 	APESV	655

The [Test Dataset](#) consists of 795 unlabeled images.

Solution Statement

A solution to this problem could be a model trained to predict a species for any image.

- Input: Input to the model would be an arbitrary image.
- Output: Output from the model would be a predicted label - one of the twelve EPPO codes listed in the table above.

Alternatively, the model could output twelve probabilities - one for each of the species. This could be in the form of an array, sorted with the highest probability label first.

- The model's success can be quantified by running the 795 test images through the model, combining the results into a single CSV and submitting this CSV to kaggle for evaluation. Kaggle will evaluate the mean multi-class F1 score.

Benchmark Model

Results from my model can be benchmarked against other submissions in the Kaggle [Leaderboard](#) for this contest. In particular, another user has put together an [example CNN using Keras](#) and is hosting it as a Kaggle Kernel. Once I've completed my solution I can run both their code and my code, and see which returns better results. (In adherence with the honor code, I have not yet looked at their solution.)

Before submitting my results to Kaggle I can evaluate my model locally through k-folds cross-validation. I will separate the data into five folds, and run training five times, holding out one of the folds for validation each time.

Evaluation Metrics

Because kaggle uses the mean multi-class F1 score as the evaluation metric for this contest, I will use this metric to evaluate my solution.

[scikit-learn's documentation on the F1 Score](#) provides a useful explanation: It "can be interpreted as a weighted average of the precision and recall, where an F1 score reaches its best value at 1 and worst score at 0".

The F1 score for a multi-class problem is the weighted average of each class's F1 Score. Because we have an unbalanced training set, we will want to weight by the number of true instances for each label.

The formula for the F1 score for any class is:

$$\frac{2 \times (\textit{precision} \times \textit{recall})}{(\textit{precision} + \textit{recall})}$$

Precision measures False Positives. The formula for precision is:

$$\frac{\textit{True Positives}}{(\textit{True Positives} + \textit{False Positives})}$$

Recall measures False Negatives. The formula for recall is:

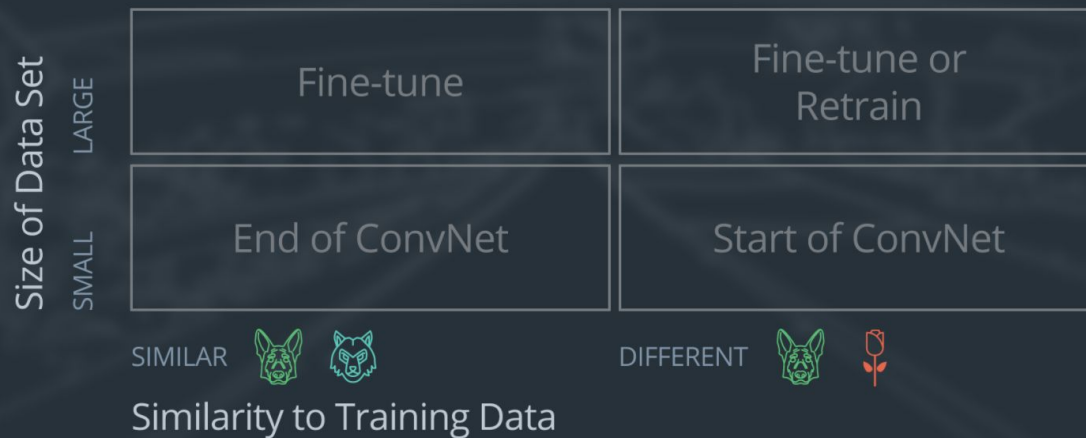
$$\frac{\textit{True Positives}}{(\textit{True Positives} + \textit{False Negatives})}$$

Project Design

I plan to attempt to solve the problem using a Convolutional Neural Network. Rather than build a network from scratch, I will use Transfer Learning on top of the InceptionV3 network.

In the Udacity lesson on Transfer Learning, we were taught that the best strategy depends on both the size of the new dataset and the similarity to the pre-trained network.

Guide for How to Use Transfer Learning



Four Cases when using Transfer Learning

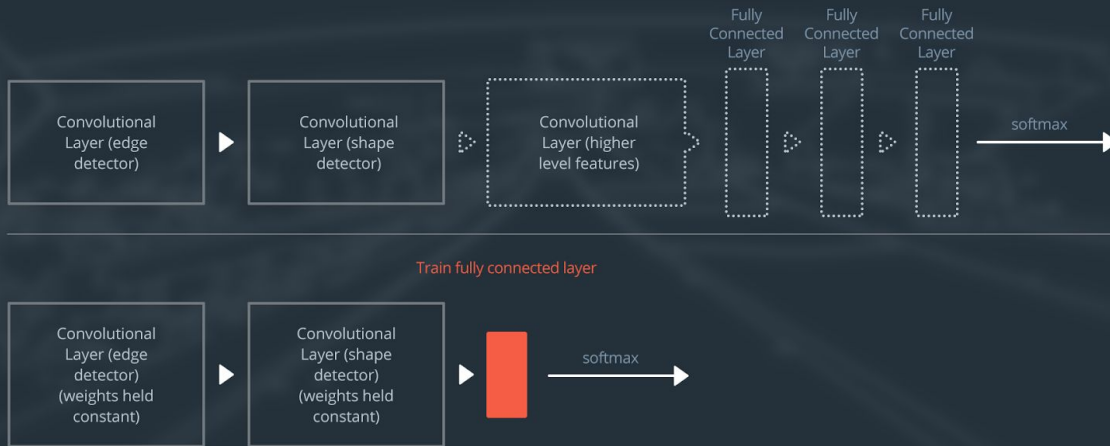
- For my use case, the **size of the new dataset is fairly small**. We have fewer than 5000 images.
- Additionally, **images in my dataset are dissimilar** from the ImageNet images used to train the Inception network.

As a result, I will follow this approach:

- I will remove most of the pre-trained layers from the Inception V3 network, retaining only the beginning of the network which detects generic aspects of images - like edges and shapes.
- I will add a new fully connected layer with 12 nodes - matching the number of classes in the seedling dataset.
- I will randomize the weights of the my new fully connected layer. I will freeze all the weights from retained layers of the Inception V3 network.
- I will train the network to update the weights of the new fully connected layer

Here is a visualization of this approach:

Case: Small Data Set, Different Data



Neural Network with Small Dataset, Different Data

Based on my initial results, I may need to tweak my model in a number of ways.

- I will experiment with removing various numbers of pre-trained layers from Inception V3, effectively experimenting with where I "cut the network in half".
- I may substitute another pre-trained network for Inception V3. Keras offers [seven pretrained models](#).
- I may experiment with data augmentation - for instance rotating the training images - in order to improve the training.

Although the F1 Score will be my main evaluation metric, I will use a confusion matrix to understand intermediate results of my training and fine tune it.

