Machine Learning Engineer Nanodegree

Capstone Proposal

Siddharth Baijal
January 25th, 2019

Proposal

Domain Background

Weed management is one of the biggest issues in the agricultural domain. It becomes an even bigger issue when a country's GDP heavily depends on agriculture. For many years researchers have tried to find ways to perform site specific weed control. If we are able to identify the plants at an early stage, then it will allow the farmers to perform weedling before the weeds start competing with the crops. Being able to manage weeds over a continuous period of time would result in better harvest in years to come.

I, am an India citizen. My country's GDP heavily depends on agriculture. Yet year on year the crop yield of many of the farmers suffers from various reasons. These reasons vary from economic conditions of the farmers, unpredictable weather conditions in the recent years due to global warming, not being able to take proper care of their crops due to lack of knowledge (which includes not being able to protect crops from weed efficiently). As per a report[1], India loses around $11 billion to weed every year. Moreover, this is not just an issue in India, losses due to ineffective weed management are a cause of economic issue worldwide. This is my personal motivation to work on this particular problem.

Problem Statement

The problem is to be able to differentiate weed from a crop seedling. The ability to do so effectively can mean better crop yields and stewardship of the environment [2]. The solution is to be able to identify the plants in one of the 12 categories of plants given in the data set.

Datasets and Inputs

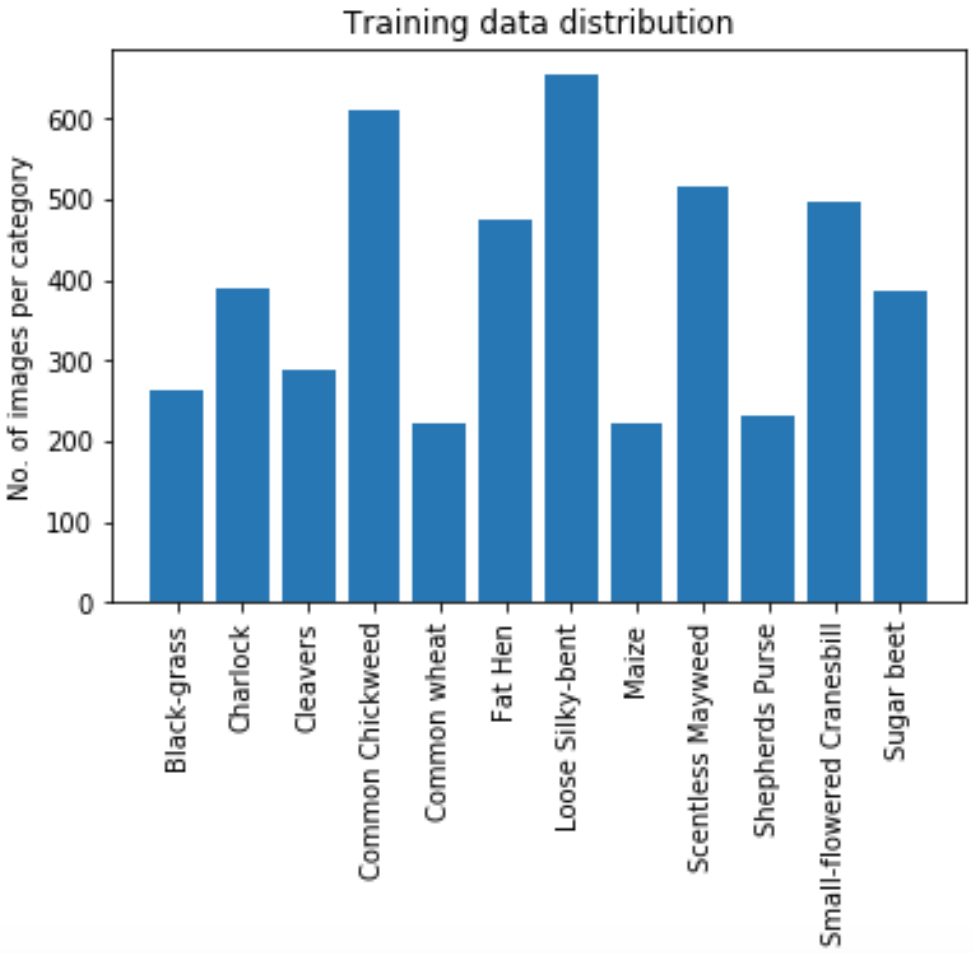
The data set is hosted by Kaggle [2]. The data set consists of training and test set both having images of plants. The images are in png format and are coloured images.

The below description is cited from the Kaggle competition itself.

Data set provided contains a training set and a test set of images of plant seedlings at various stages of grown. Each image has a filename that is its unique id. The dataset comprises 12 plant species. The goal of the competition is to create a classifier capable of determining a plant's species from a photo. The list of species is as follows:

1. Black-grass
2. Charlock
3. Cleavers
4. Common Chickweed
5. Common wheat
6. Fat Hen
7. Loose Silky-bent
8. Maize
9. Scentless Mayweed
10. Shepherds Purse
11. Small-flowered Cranesbill
12. Sugar beet

I did a small test on the training data set to find the the count of samples and distribution across each category and following are the results:-



The total number of sample images in training and test data are 4750 and 794 respectively.

It is quite evident from the graph above that the distribution of the training data across categories is not normal and seems to be unbalanced.

I will be using the above training data to train my model. Moreover I plan on splitting the training data set in training and validation sets. Finally I will use the trained model to identify plant seedlings as a part of one of the 12 categories using the test data set.

Solution Statement

The aim of this project is to identify plant breeds which will help isolate them from the weeds and result in effective weed management.

For the solution I plan on using deep neural networks to identify the plant breeds. More specifically, I would make use of CNN (Convolutional Neural Networks) as they are able to effectively identify patterns within images. To add to this, I would make use of transfer learning which involves taking a pre-trained neural network and adapting the neural network to a new, different data set.

The result would be that the trained CNN model is able to accurately identify the images from the test data set into one of 12 categories.

Benchmark Model

Since I plan to make use of CNN along with transfer learning as a solution to identifying the plants, it would be ideal to use a simple CNN model (without a pre-trained network) as a benchmark model to compare my results. Once I pass the results of the benchmark model using CNN created with transfer learning, I will compare my results with kernels of this Kaggle competition and try to build a competing model myself. The highest accuracy is around 0.979. I don't intend to use any of the code provided in those kernels itself for my solution.

Evaluation Metrics

Evaluation metric used to quantify the performance of the benchmark and the solution model is F1-score. This is the evaluation method as per the Kaggle competition [3] as well. To define F1-score we need to first define the following terms [4]:

True Positive (TP)

A true positive test result is one which detects condition, when the condition is present.

True Negative (TN)

A true negative test result is one which does not detect a condition, when the condition is absent.

False Positive (FP)

A false positive test result is one which detects condition, when the condition is absent.

False Negative (FN)

A false negative test result is one which does not detect a condition, when the condition is present.

Precision

Is the ability of the test to detect a condition when the condition is present.

$$TP / (TP+FN)$$

Recall

Recall is the proportion of positives that correspond to a presence of a condition.

$$TP / (TP+FP)$$

F1-Score

F1-score is the harmonic mean of precision and recall, where F1 reaches the highest value at 1 and the worst at 0.

$$2 * (Precision * Recall / Precision + Recall)$$

Project Design

I have broken down project design into following 3 sections:

- Data Exploration
- Pre-Processing
- Create CNN Model

Data Exploration

In my approach to the given problem, the first and foremost thing will be data exploration. Here I will try to extract as much information about the dataset (training and test) as possible like the distribution of data, image size etc.

Pre-Processing

Next step would be to perform the necessary pre-processing on the data set. I would check if we there is any noise in the images which can be removed or if I need to resize the images to the same size. I might convert the images to grayscale as well (if required). Since, the images under different categories are not well distributed, I might use augmentation technique to widen the variety of data presented to the model.

Create CNN Model

Once we are through with the, data exploration and the image pre-processing step, I would move towards designing the CNN architecture of the data model. For designing the neural network, I want to make use of transfer learning instead of designing the network from scratch. In particular I plan on using Inception, Xception and ResNet50 models for transfer learning. I would compare the results of these models and select the one with best performance.

[5]Transfer Learning involves taking a pre-trained neural network and adapting the neural network to a new, different data set. Based on the size and the similarity of the data set from the original data there are 4 main approaches to transfer learning:-

Small dataset and similar data	**Small dataset and different data**
Large dataset and similar data	**Large dataset and different data**

Case 1 : Small dataset and similar data

If the new data set is small and similar to the original data set:

- slice off the end of the neural network
- add a new fully connected layer that matches the number of classes in the new data set
- randomize the weights of the new fully connected layer; freeze all the weights from the pre-trained network
- train the network to update the weights of the new fully connected layer

Case 2 : Small Data Set, Different Data

If the new data set is small and different from the original training data:

- slice off most of the pre-trained layers near the beginning of the network
- add to the remaining pre-trained layers a new fully connected layer that matches the number of classes in the new data set
- randomize the weights of the new fully connected layer; freeze all the weights from the pre-trained network
- train the network to update the weights of the new fully connected layer

Case 3: Large Data Set, Similar Data

If the new data set is large and similar to the original training data:

- remove the last fully connected layer and replace with a layer matching the number of classes in the new data set
- randomly initialize the weights in the new fully connected layer
- initialize the rest of the weights using the pre-trained weights
- re-train the entire neural network

Case 4: Large Data Set, Different Data

If the new data set is large and different from the original training data:

- remove the last fully connected layer and replace with a layer matching the number of classes in the new data set
- retrain the network from scratch with randomly initialized weights
- alternatively, you could just use the same strategy as the "large and similar" data case

From the above cases, our case is Small dataset, Different data. This is because the seedling data set provided is relatively small and the images are different than the ones on which our transfer learning models have been trained. Therefore as mentioned in Case 2, I will slice off most of the layers of the pre-trained near the beginning of the network and add to the remaining pre-pre-treated layers a new fully connected layer that matches the number of classes in the new data set (i.e. 12 in our case).

Once I successfully created the CNN model using transfer learning, I might perform some fine tuning of parameters to achieve the desired accuracy. The fine tuning would involve changing the learning rate, optimiser and also make use of augmentation to provide better diversity of data to the model.

References

[1] https://www.thehindubusinessline.com/economy/india-loses-farm-produce-worth-11b-to-weeds-every-year-icar/article10033566.ece

[2] https://www.kaggle.com/c/plant-seedlings-classification/data

[3] https://www.kaggle.com/c/plant-seedlings-classification#evaluation

[4] https://groups.bme.gatech.edu/groups/biml/resources/useful_documents/Test_Statistics.pdf

[5] Referenced from Udacity course work