

# OBJECT CLASSIFICATION ON PLANTS

Soham Ashtekar

St. Clair College, Windsor

sa24@myscc.ca

Amandeep Kaur

St.Clair College, Windsor

aa52@myscc.ca

## 1. INTRODUCTION

As computing power, we have in hand has increased tremendously, we have reached new milestones in the field of machine learning and artificial intelligence. One of the important researches we have done in this field is object detection and classification. In recent years we have seen lots of implementation of machine learning in various fields, such as google ad's, tesla's autopilot, IBM's Watson and many more.

Tesla uses it's sensors and cameras in the car to classify objects, and by using machine learning they predict how the input to the steering wheel will affect the direction of car or how input to the break will stop or slow down the car. For tesla vehicle to be aware of its environment they are using multiple cameras around the car, they grab video feed of the surrounding and the computer inside the car processes it, if there are any stop signs or speed limits the car reads that and shows it in the screen inside the car. The car also shows its surroundings in 3D world on the screen put in the car.

The purpose of this project is to learn how can we use different types of computer vison and object detection techniques, how they work and possible implementation in the real world. The image classification is done by using convolutional neural networks is a part of deep neural networks, they are also known as shift invariant or space invariant artificial neural networks.

For this study we will be using data from different sources, for some plants images we will be taking images in store such as Walmart, Food Basics, Fresh co these stores are widely known stores in Canada. They sell different types of plants for decoration. We went in these stores and grabbed 50-60 images of each object in different lighting environment and different angles. Different angles and different lighting creates more variation in the dataset which makes the model more robust and it is generalized.

After grabbing the pictures we will do some preprocessing to make the data ready for the model to learn, and then we will try bunch of different convolutional models such as Convo2D, vgg16, Resnet V2. We are using ready solutions because to tune and test newly created neural networks takes a lot of time, the higher the image resolution the more time it takes to train the model, and in our case resolution will be

416\*312. The basic Conv2d network is moderately tuned for our dataset and other models are used as it is from the Keras library. Our choice of backend was Tensorflow version 1.15, and we used GPU(Graphics Processing Unit) for training our model as it increases the speed of the training tremendously. We will discuss why in methods section.

## 2. Related Work

There are several researches and projects done on the image classification and object detection which are very robust too, but most of those studies do not have a complete solution for my problem, as I wanted to know more about how parallel computing works and how can we use the power of GPU to train these neural networks. Also I have used techniques where I have imported models from Keras and changed the input and output layers to match my dataset.

**Plants Images Classification Based on Textural Features using Combined Classifier[1]** is a study done by **M. Z. Rashad, B.S.el-Desouky and Manal S.Khawaski** from the **Mansoura University Egypt**, they grabbed pictures of plants using a 6Mp camera and then as preprocessing they downscaled the image to 128\*128 pixels, then they converted image to grayscale, for classification they used Combined Classifier from LVQ(Learning Vector Quantization) ad RBF(Radial Based Function) as their proposed model, they compared the resulted probabilities from multiple classifiers and the result was the one with maximum vote. They achieved around 88% accuracy using this method. In their method they trained the model for very long time which is around 2000 epochs and they used smaller database, and that is where I tried to improve the approach.

**Recognition Of Plants Using Leaf Image With Neural Network and Computer Vision[2]** is a recent study done by **Shivam Upadhyay, Aakash Yadav, Kaushik Yadav, Sonal Chaudhari**, they used images of leaf from different plants and then using the features extracted from images they classified the tree. In the first stage they obtained images using a digital scanner and then they extracted morphological features which act as input to the classifier. After grabbing bunch of images they flipped images in multiple angles and then crated more data which has variation, after that in pre processing they reduced the pixel size of the images to 100\*100 and then converted those to grayscale and at the end the images were converted to binary scale. They applied some filters to detect the edges and extract images from the background. Then using those features as input they trained a neural network using back propagation and tuning the weights. They used Tensorflow and a local computer for this research which I will improve in my own studies.

### 3. Methods

**Data Collection:** - We went in to the stores like Walmart, Food Basics and Fresh co and grabbed around 400 images. These images were grabbed from mobile phone, because we grabbed pictures using two different phones we had different image quality and also different image resolution which introduced further variance in our dataset. Then we copied those images into a personal computer and created different folders for each type of plants.

**Data Preprocessing:** - First we created a method which takes class and image directory as two arguments and then creates a X(Feature array) Y(Target array). To read the images we used OpenCV. In the make train data we also resized the images to 416\*312 which gave us images with enough details to classify accurately. After the feature array was created which had information about luminance of each Red, Green and Blue pixel we scaled the values by doing division with 255. This works like min max scaler. 255 is the highest value for a Red, Green or a Blue pixel, by dividing values by 255 we are scaling the data.

**Data Generator:** - After scaling the data we used Image Data Generator which generates variation in data by flipping and zooming on the images. For data generator we set 11 parameters which are given below.

- featurewise\_center = False (set input mean to 0 over the dataset)
- samplewise\_center=False ( set each sample mean to 0)
- featurewise\_std\_normalization=False (divide inputs by std of the dataset)
- samplewise\_std\_normalization=False (divide each input by its std)
- zca\_whitening=False (apply ZCA whitening)
- rotation\_range=10 (randomly rotate images in the range (degrees, 0 to 180))
- zoom\_range = 0.1 (Randomly zoom image)
- width\_shift\_range=0.2 (randomly shift images horizontally (fraction of total width))
- height\_shift\_range=0.2 (randomly shift images vertically (fraction of total height))
- horizontal\_flip=True (randomly flip images)
- vertical\_flip=False ( randomly flip images)

**Neural Networks:** - After the data was completely prepared for the model we trained the data for three different convolutional neural networks.

- I. **ConvoNet:** - A ConvoNet model was used from the Kaggle kernels written by **Raj Mehrotra[3]** and then the model was tuned for our data set. After importing the model, first layer of the model was deleted, and we put our own input layer with

shape 416\*312 and 32 filters with relu activation. Then we removed his output layer and added our own Dense layer with 6 classes and Sigmoid activation. After chose changes we got a accuracy boost of 60%.

- II. **Vgg16:** - A vgg16 model was imported from Keras [4] applications and for arguments we included the top layer and did not import the weights. Then we gave an input shape of (416,312,3) which we were using for all our models. Then we compiled model with Optimizer Adam with learning rate of 0.001. And for our loss function we used ‘Categorical\_crossentropy’ and chosen metrics for learning was accuracy.

#### **VGG16 implementation**

```
from keras.applications.vgg16 import VGG16  
  
model_vgg16 = VGG16(include_top=True,input_shape=(416,312,3),weights=None,  
classes=6, pooling=None)
```

- III. **ResNet v2:** - We imported resnetv2 from Keras [5] and we did similar steps which we did for vgg16, removal for first and last layer and compiling with learning rate of 0.001, at the end we trained the model on our dataset.

#### **Resnetv2 Implementation**

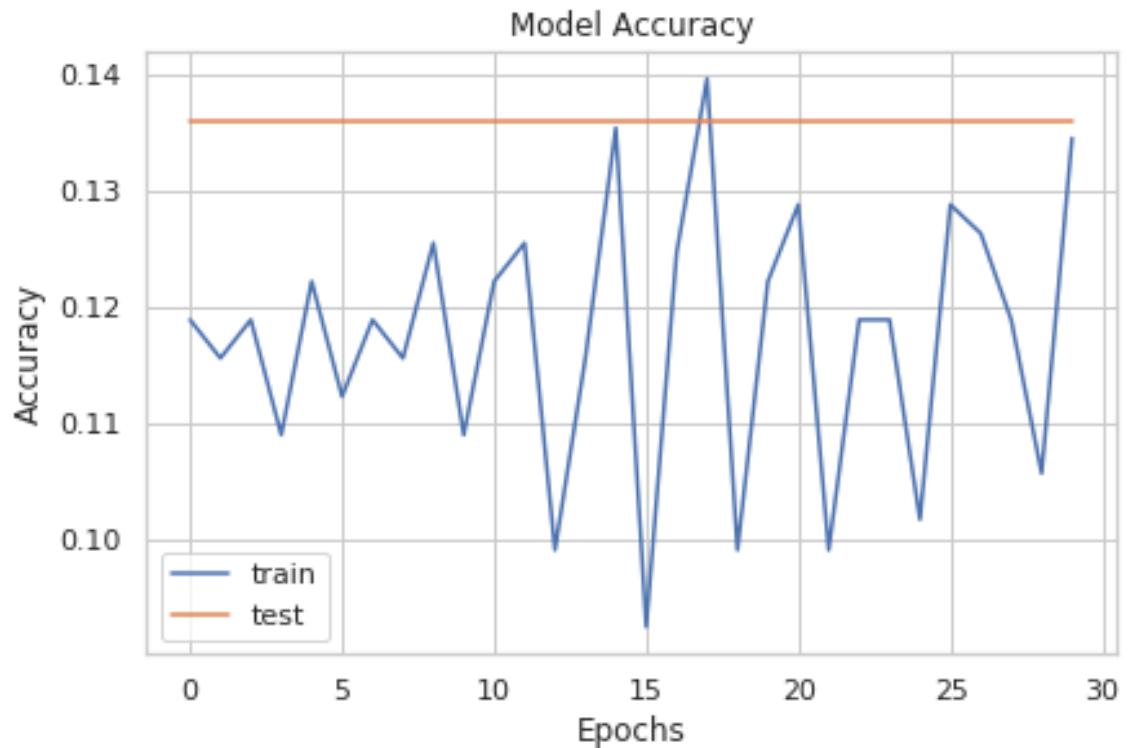
```
from keras.applications.resnet_v2 import resnet_v2  
  
model_resnet = keras.applications.resnet_v2.ResNet50V2(include_top=True,input_shap  
e=(416,312,3),weights=None, classes=6, pooling=None)
```

## **4. Results**

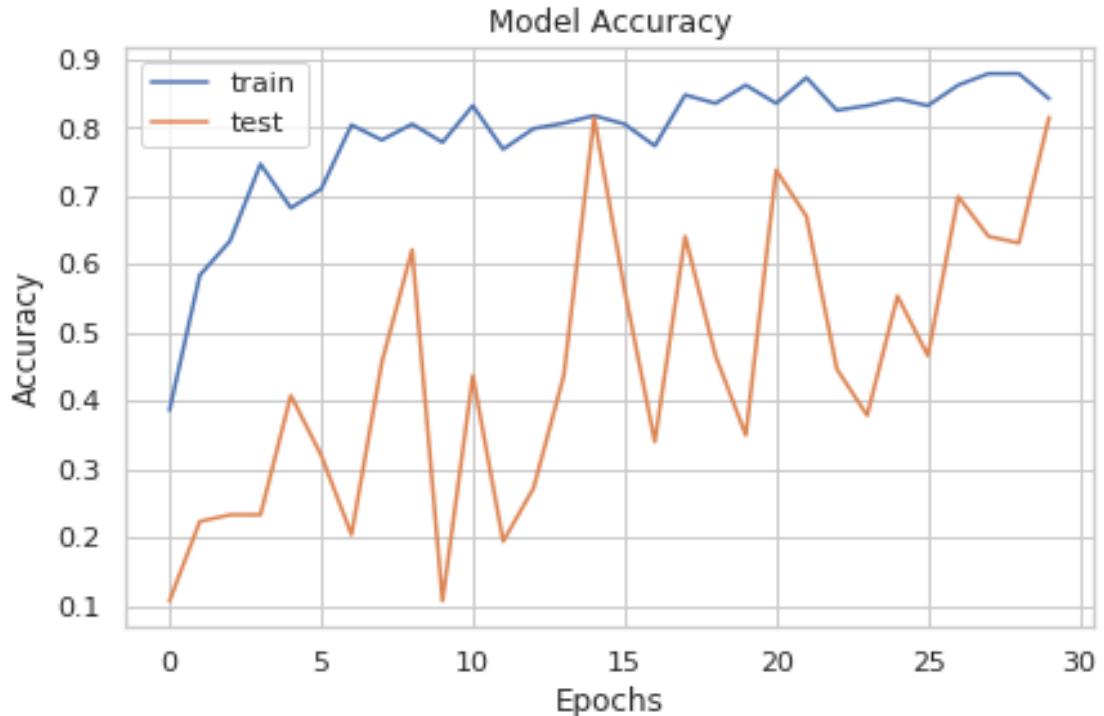
After training and testing multiple models our proposed method is ConvoNet convolutional neural network which was provided by Raj Mehrotra and improved by us. Model tuned by us provided above 90% accuracy for the train set and 93% accuracy for the validation set. When the model was in pure state(Without changes) the accuracy was 30%. The 50-60% jump in accuracy was a good milestone. The Learning graph for our model is as follows.



The vgg16 model we testing didn't learn at all, the main reason to that can be number of epochs. Vgg16 gave us accuracy score of 10-20% and the accuracy score and loss remained same for its entire training time. The graph explains further



The last model in our testing was resnet, it performed upto par with our model, but the consistency was not their and the learning did not seem to be linear, there werea lot of variations in the learning.



## 5. Discussion

As compared to previous studies we trained our models using google Collab which reduced the training time more than 800% because of the computational power of GPU. Not many studies used multiple models based on convolutional neural networks. Compared to study done by **Shivam Upadhyay** and his colleagues was were like the study we did, related to their studies we got better result for our proposed model by 5%. Our dataset had better quality pictures (Greater Resolution) which creates more generalized model due to more data.

The study done by **M. Z. Rashad** and his colleagues can be called ancient due to revolutionary technological advancements. Even though their model accuracy was very good, to get that type of accuracy they had to train neural network for very long, we obtained the same kind of results in much lesser time.

## 6. Conclusion

Artificial neural networks and machine learning is not a new field, but the advancements in technologies like Graphics Cards, CPU and datacenters created by companies like Microsoft (Azure)[6] and Google(Could, Collab)[7] which are freely

available make training this huge Neural Networks much easier. We are generating 2 times the data every day which we used to generate yearly a decade ago and now we have the power to analyze that data and make use of it.

Models like ResnetV2 and Vgg16 can provide excellent accuracy on Mobilenet data but to create solution for your own problem you may have to change these networks as we did with ConvNet. We can apply object detection or classification to many fields, the applications are endless.

GPU makes it easier and faster to train the network so when training network with TensorFlow 1.15 or previous you should use.

```
#Checking if we have compatible gpu
from tensorflow.python.client import device_lib
print(device_lib.list_local_devices())
```

## 7. Contribution

**Soham Ashtekar**:- The task assigned was to preprocess the data and train the Neural network. After the code was written the documentation was created which lead into report creation.

**Amandeep Kaur**:- The main idea to do this study was of her, she gathered the data and researched for the report and solution.

## 8. References

- [1] [https://www.researchgate.net/profile/Beih\\_El-Desouky/publication/274175414\\_Plants\\_Images\\_Classification\\_Based\\_on\\_Textural\\_Features\\_using\\_Combined\\_Classifier/links/554364190cf23ff71683a199/Plants-Images-Classification-Based-on-Textural-Features-using-Combined-Classifier.pdf](https://www.researchgate.net/profile/Beih_El-Desouky/publication/274175414_Plants_Images_Classification_Based_on_Textural_Features_using_Combined_Classifier/links/554364190cf23ff71683a199/Plants-Images-Classification-Based-on-Textural-Features-using-Combined-Classifier.pdf)
- [2] [https://www.researchgate.net/profile/Beih\\_El-Desouky/publication/274175414\\_Plants\\_Images\\_Classification\\_Based\\_on\\_Textural\\_Features\\_using\\_Combined\\_Classifier/links/554364190cf23ff71683a199/Plants-Images-Classification-Based-on-Textural-Features-using-Combined-Classifier.pdf](https://www.researchgate.net/profile/Beih_El-Desouky/publication/274175414_Plants_Images_Classification_Based_on_Textural_Features_using_Combined_Classifier/links/554364190cf23ff71683a199/Plants-Images-Classification-Based-on-Textural-Features-using-Combined-Classifier.pdf)
- [3] <https://www.kaggle.com/rajmehra03/flower-recognition-cnn-keras>
- [4] <https://keras.io/applications/#vgg16>
- [5] <https://keras.io/applications/#resnet>
- [6] <https://notebooks.azure.com/>
- [7] <https://towardsdatascience.com/getting-started-with-google-colab-f2fff97f594c>

## 9. Appendices

- <https://github.com/sohamashtekar/Plants-Classification> Required ipynb notebook and dependencies and installation instructions.