MACHINE LEARNING NANODEGREE PROGRAM

CAPSTONE PROPOSAL IMPLEMENTATION

Satish Fulwani

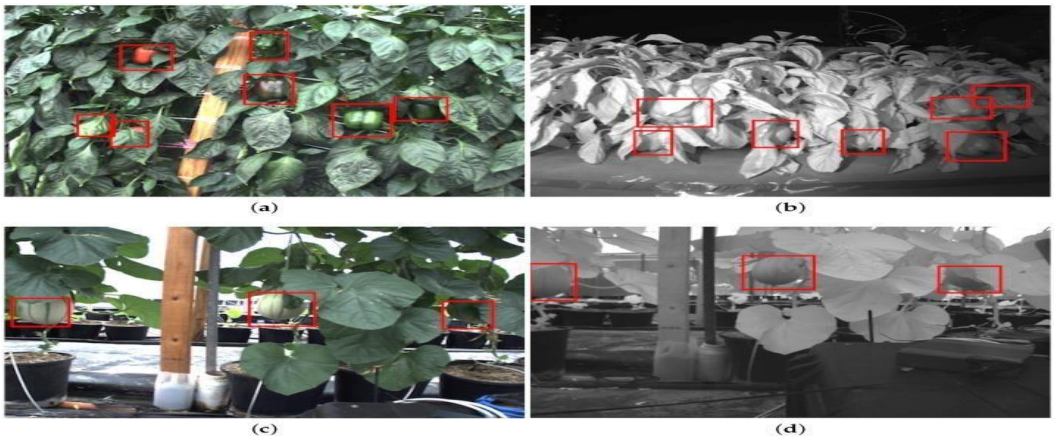
02/01/2019

FRUIT CLASSIFIER

I. Definition

Project Overview

Skilled robots are being used for fruit harvesting in farms, this helps in reduction of man power and to increase the speed of work. The main part in robot harvesting is detection of the fruit , if robot cannot detect the correct fruit it cannot do the estimated work. So fruit detection is one part of robot harvesting and we will use Deep Convolutional Neural Networks (DCNN) to develop fruit classifier. So here we will develop Fruits classifier that will classify the fruits present in an image and will predict the percentage if there is no fruit in that image.



Although many researchers have tackled the problem of fruit detection, the problem of creating a fast and reliable fruit detection system persists, as found in the survey by "http://www.inderscience.com/offer.php?id=46419". This is due to high variation in the appearance of the fruits in field settings, including colour, shape, size, texture and reflectance properties. Furthermore, in the majority of these settings, the fruits are partially abstracted and subject to continually-changing illumination and shadow conditions. To solve this problem DCNN provides good performance in images with low quality, different shapes, different views.

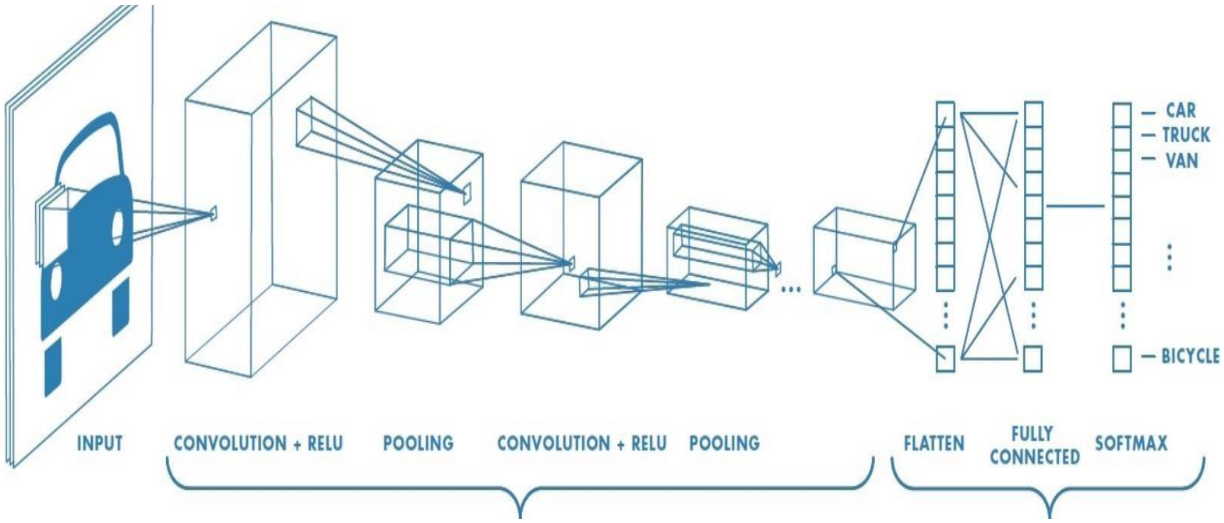
Problem Statement

In this project, we will provide an image to the model and it will predict the fruit present. The project will be using Conventional Neural Network. And it can be directly linked to an webpage and used to predict the fruits present. The project is referenced from Kaggle, it has 81 types of fruit images.

Domain Background

This project is mainly dependent on 2 domains:

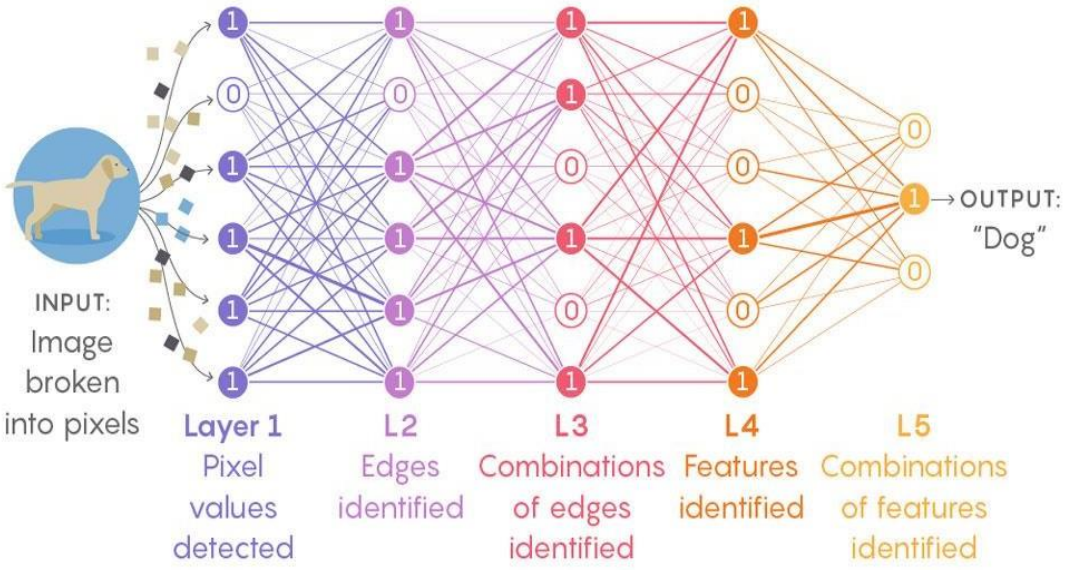
1) Convolutional NeuralNetwork



2) Deep Learning

Learning From Experience

Deep neural networks learn by adjusting the strengths of their connections to better convey input signals through multiple layers to neurons associated with the right general concepts.



When data is fed into a network, each artificial neuron that fires (labeled "1") transmits signals to certain neurons in the next layer, which are likely to fire if multiple signals are received. The process filters out noise and retains only the most relevant features.

Metrics

Existing models and study available in the domain to compare the accuracy.

- Previously available models to enhance theaccuracy.
- Results available in thepaper.

Table 3: Results of training the neural network on the fruits-360 dataset.

Scenario	Accuracy on training set	Accuracy on test set
Grayscale	99.53%	91.91%
RGB	99.51%	95.59%
HSV	99.32%	95.22%
HSV + Grayscale	98.72%	94.17%
HSV + Grayscale + hue/saturation change + flips	99.46%	96.41%

- Comparing another implementation onKaggle.

II. Analysis

Data Exploration

There in all 81 fruits with 55244 images. The following fruits are included: Apples (different varieties: Golden, Golden-Red, Granny Smith, Red, Red Delicious), Apricot, Avocado, Avocado ripe, Banana (Yellow, Red), Cactus fruit, Cantaloupe (2 varieties), Carambula, Cherry (different varieties, Rainier), Cherry Wax (Yellow, Red, Black), Clementine, Cocos, Dates, Granadilla, Grape (Pink, White, White2), Grapefruit (Pink, White), Guava, Huckleberry, Kiwi, Kaki, Kumsquats, Lemon (normal, Meyer), Lime, Lychee, Mandarine, Mango, Maracuja, Melon Piel de Sapo, Mulberry, Nectarine, Orange, Papaya, Passion fruit, Peach, Pepino, Pear (different varieties, Abate, Monster, Williams), Physalis (normal, with Husk), Pineapple (normal, Mini), Pitahaya Red, Plum, Pomegranate, Quince, Rambutan, Raspberry, Salak, Strawberry (normal, Wedge), Tamarillo, Tangelo, Tomato (different varieties, Maroon, Cherry Red), Walnut.

Label	Number of training images	Number of test images
Apple Braeburn	492	164
Apple Golden 1	492	164
Apple Golden 2	492	164
Apple Golden 3	481	161
Apple Granny Smith	492	164
Apple Red 1	492	164
Apple Red 2	492	164
Apple Red 3	429	144
Apple Red Delicious	490	166
Apple Red Yellow	492	164
Apricot	492	164
Avocado	427	143
Avocado ripe	491	166
Banana	490	166
Banana Red	490	166
Cactus fruit	490	166
Cantaloupe 1	492	164
Cantaloupe 2	492	164
Carambula	490	166
Cherry 1	492	164
Cherry 2	738	246
Cherry Rainier	738	246

2)

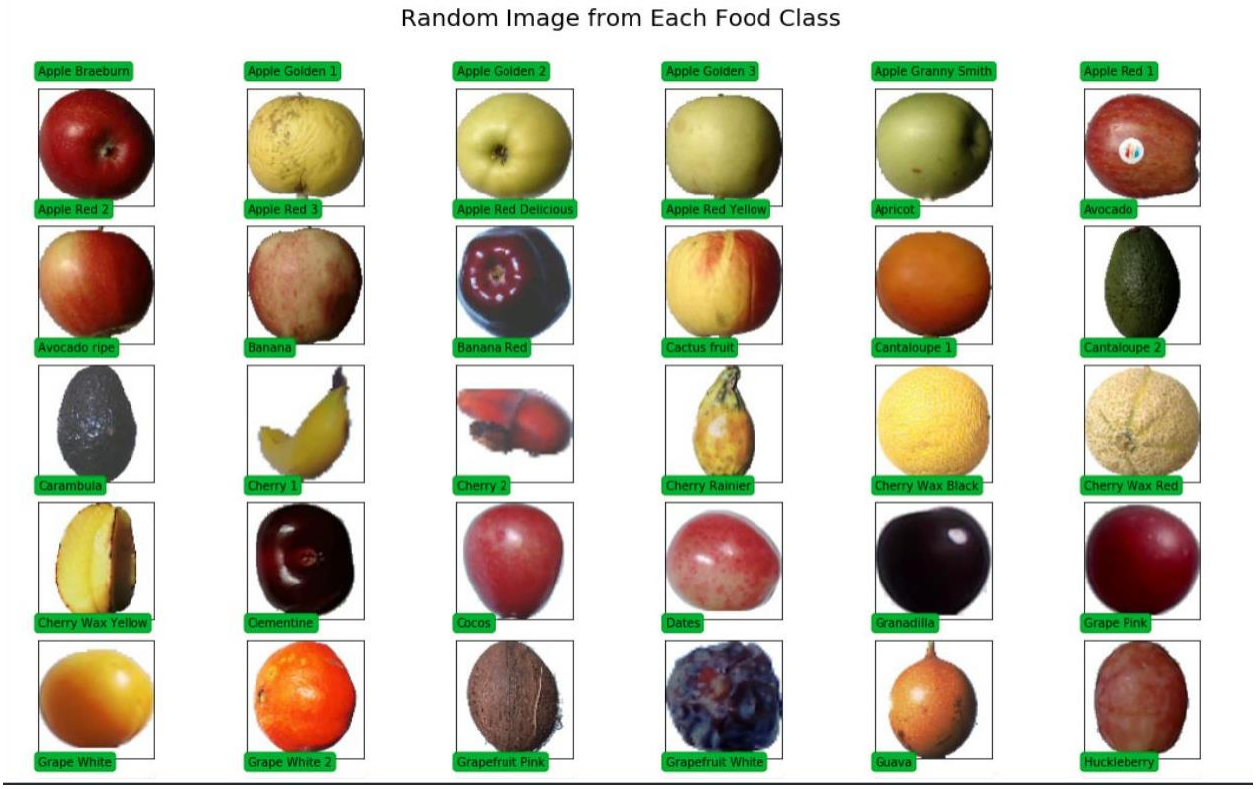
Label	Number of training images	Number of test images
Cherry Wax Black	492	164
Cherry Wax Red	492	164
Cherry Wax Yellow	492	164
Clementine	490	166
Cocos	490	166
Dates	490	166
Granadilla	490	166
Grape Pink	492	164
Grape White	490	166
Grape White 2	490	166
Grapefruit Pink	490	166
Grapefruit White	492	164
Guava	490	166
Huckleberry	490	166
Kaki	490	166
Kiwi	466	156
Kumquats	490	166
Lemon	492	164
Lemon Meyer	490	166
Limes	490	166
Lychee	490	166
Mandarine	490	166
Mango	490	166
Maracuja	490	166
Melon Piel de Sapo	738	246
Mulberry	492	164
Nectarine	492	164
Orange	479	160
Papaya	492	164
Passion Fruit	490	166
Peach	492	164
Peach Flat	492	164
Pear	492	164
Pear Abate	490	166

3)

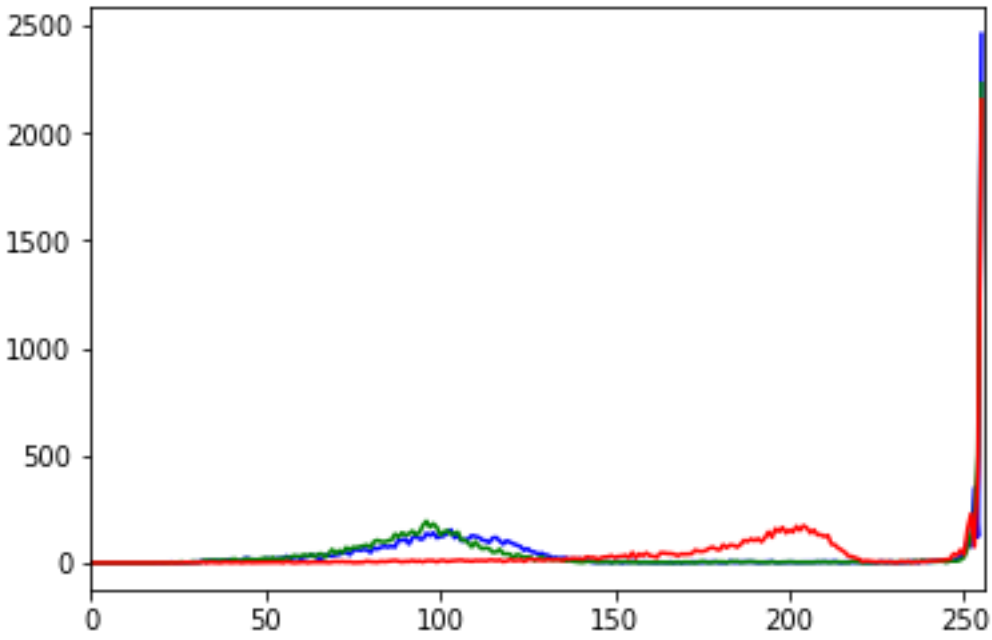
Label	Number of training images	Number of test images
Pear Monster	490	166
Pear Williams	490	166
Pepino	490	166
Physalis	492	164
Physalis with Husk	492	164
Pineapple	490	166
Pineapple Mini	493	163
Pitahaya Red	490	166
Plum	447	151
Pomegranate	492	164
Quince	490	166
Rambutan	492	164
Raspberry	490	166
Salak	490	162
Strawberry	492	164
Strawberry Wedge	738	246
Tamarillo	490	166
Tangelo	490	166
Tomato 1	738	246
Tomato 2	672	225
Tomato 3	738	246
Tomato 4	479	160
Tomato Cherry Red	492	164
Tomato Maroon	367	127
Walnut	735	249

Data Visualization

- After loading data from dataset, using python's matplotlib Iplotted the images according to their classes.



- Here is a graph which shows the color representation in an training image using CV2, which shows that blue has some high value areas in the image.



Algorithm and Techniques

- The base of the model being made is Deep Learning.
- The technique used is Convolutional Neural Networks, the model is being developed from scratch.
- We will be designing Convolutional Neural Networks and connect it to deep neuralnetwork.
- Various filters and pooling layers for predicting the image more properly.
- The input to the model will be an image of size 224*224 , which is converted to an array of dimension 4.
- The dataset being to large it is not possible to convert and train the model on local machine, so platform used here is the instance at google cloud.
- The end layer of the model is using ‘Softmax function’ which gives the probabilities of the type of fruit in the Image.

Detailed Architecture:

My first thought was to give perfect result with less errors so I thought to go deep and increase the number of filters.

1) First I started with 16 filters and kept padding as 'same' as it may help me not losing the data.

2) Then a MaxPooling layer with pool_size=2 to reduce our data width wise.

3) Then to go deep inside and classify I went until the layer with 128 filters.

4) A droupout layer to reduce chances of overfitting.

5) To connect CNN layers to fully connected network flattering layer is used.

6) Then a hidden layer with 500 nodes is connected having 'relu' activation function.

7) As there are 133 breed the output layer having 133 nodes having 'softmax' activation function is connected.

Benchmark

- The benchmark for the model being developed is calculated from the accuracy obtained from training set.
- There are various already created models available on Kaggle which will be used to compare the accuracy, here are some models:

Table 3: Results of training the neural network on the fruits-360 dataset.

Scenario	Accuracy on training set	Accuracy on test set
Grayscale	99.53%	91.91%
RGB	99.51%	95.59%
HSV	99.32%	95.22%
HSV + Grayscale	98.72%	94.17%
HSV + Grayscale + hue/saturation change + flips	99.46%	96.41%

III. Methodology

Data Preprocessing

- LoadingDataset:
  To load dataset, we populate a few variables with the load_files function from the scikit-learn library:
  - train_files, valid_files, test_files - numpy arrays containing file paths to images
  - train_targets, valid_targets, test_targets - numpy arrays containing onehot-encoded classificationlabels.
  - fruit_names - list of string-valued dog breed names for translatinglabels.

- Now the data obtained is in image format, we must convert it in the form the CNN model takes input that is Numpy Array. To convert the steps followed are:
  - The input image is first converted to numpy array of dimension 3 by using img_to_array function.
  - Then the array from the previous array is converted to dimension 4.

Implementation

- Deep Learning is used for training the model.
- Convolutional Neural Networks was designed and connected to deep neural network.
- The model takes an input array of dimension 4, processes it further through the network.
- The filter size increases in following sequence 8,16,32,64.
- Model has dropout layers to avoid overfitting.
- The last layer is dense layer of 81 nodes as we have 81 types of fruits.
- Now model is ready for training.

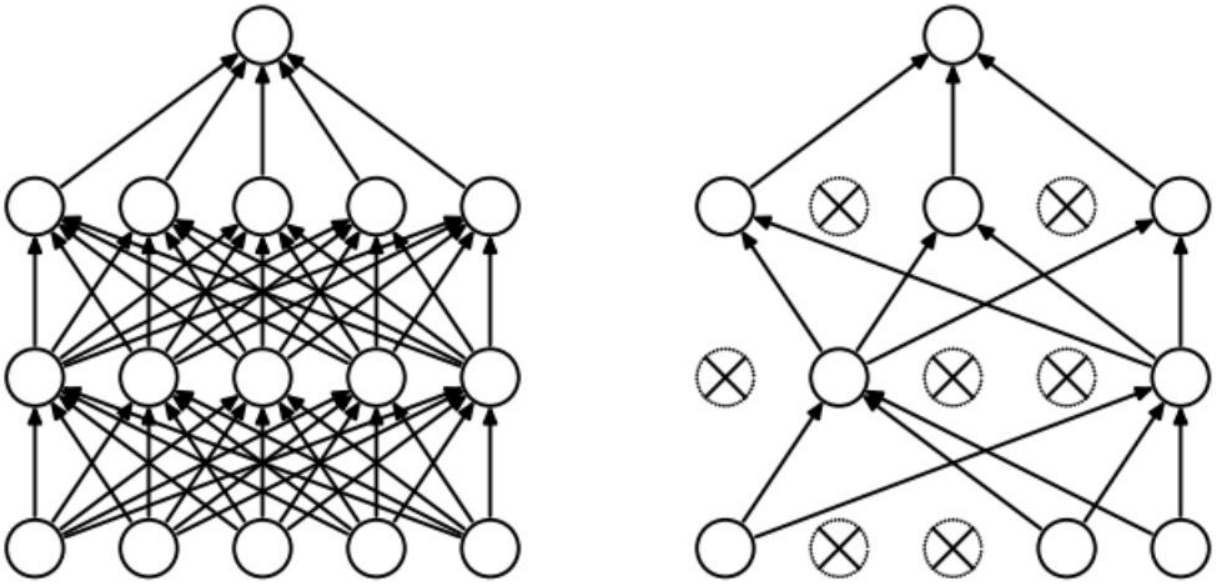
Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 224, 224, 16)	208
max_pooling2d_1 (MaxPooling2	(None, 112, 112, 16)	0
conv2d_2 (Conv2D)	(None, 112, 112, 32)	2080
max_pooling2d_2 (MaxPooling2	(None, 56, 56, 32)	0
conv2d_3 (Conv2D)	(None, 56, 56, 64)	8256
max_pooling2d_3 (MaxPooling2	(None, 28, 28, 64)	0
conv2d_4 (Conv2D)	(None, 28, 28, 128)	32896
max_pooling2d_4 (MaxPooling2	(None, 14, 14, 128)	0
dropout_1 (Dropout)	(None, 14, 14, 128)	0
flatten_1 (Flatten)	(None, 25088)	0
dense_1 (Dense)	(None, 500)	12544500
dropout_2 (Dropout)	(None, 500)	0
dense_2 (Dense)	(None, 81)	40581

Total params: 12,628,521
Trainable params: 12,628,521
Non-trainable params: 0

Fig: Model Summary

Refinement

- After implementation of the model the changes made were few dropouts layer were added to the model as the dataset is so size 64000 image.
- This is how a dropout layers helps in Neural network:



(a) Standard Neural Net (b) After applying dropout.

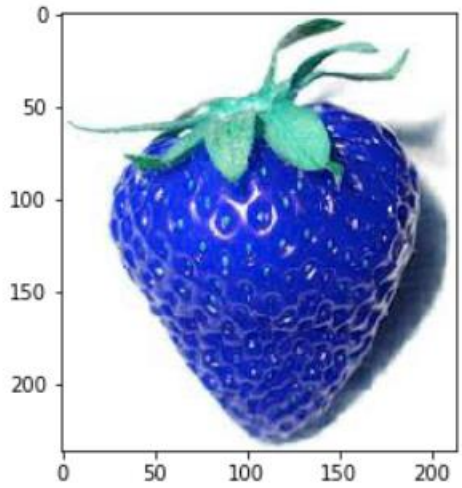
- The dropout values we used are 0.2 and 0.4, with increasing the dropout, there is some increase in validation accuracy and decrease in loss.

IV. Results

- ## Model Evaluation and Validation

  The accuracy obtained from the training set is 99.47% and from testing set is 97.68%.And when the model is tested on image from real world it failed to predict perfectly the type of fruit.

```
In [20]: test_model('strawberry.jpg')
```




```
Out[20]: 'Tomato 2'
```

- ## Justification

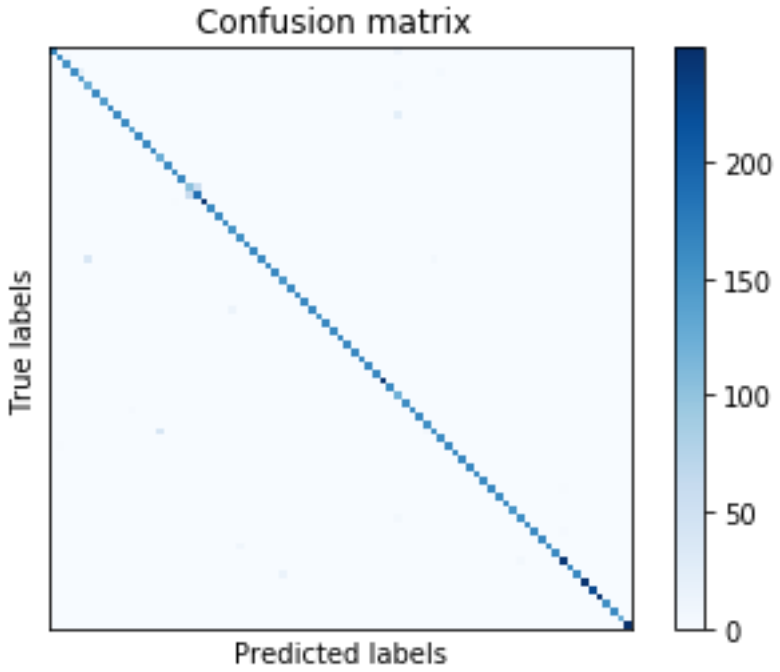
  According to our benchmark models our model proved to be very effective on testing data and our goal is achieved but still our model is not up to mark and still can improve more.

V. Conclusion

Free-Form Visualization

Confusion Matrix:

- Confusion Matrix, also known as an error matrix, is a specific table layout that allows visualization of the performance of an algorithm.
- Here is the confusion matrix of my model:



Reflection

The overall process of development of the model from starch was little though but due to this project I learnt many new techniques and cleared my concept regarding machine learning. The most interesting aspect was to preprocess the data and to do the same it requires large memory. The model takes an input of 224*224, and predicts the output accordingly. This can be used as a part for many application and automation systems.

Improvement

The improvement in this project is training the model for more epochs and increasing the prediction accuracy for image on unseen data.