A Learning Based Classification Model for Vegetables Recognition and Recipes Recommendation

Gaurang Solanki[1], Kartik Mahawar[2], Piyush Singh[3], Puneet[4]

School of Computer Science and Engineering

Lovely Professional University. Phagwara, Punjab, India

gorangsolanki111@gmail.com, km9300102853@gmail.com

piyushsingh.11712746@gmail.com, puneetbhardwaj616@gmail.com

Abstract

This paper presents the building of a classification model using Transfer learning, deployment of the model on the webpage, and recipes Recommendation using web scrapping. First, the vegetable classification model was built using transfer learning on the Pre-trained model MobilenetV2, which was originally trained on the ImageNet dataset with 1.4 Million images and 1000 classes of web images. We used this as our base model to train our vegetable dataset and classify the images of 26 different vegetables. Second, the Model was deployed on a webpage using Tensorflow. JS. The model takes an image as input, classifies the image, and gives the vegetable name as output. Images can be taken from the webcam as well as from the storage. Third, basedon model outputs, the web scraping script is used to fetch links for top-rated recipes from the internet.

1. INTRODUCTION

This proposal is aimed at people who don’t have that much knowledge about vegetables and help them ease their cooking. A system that can recognize local vegetables using Deep learning, under this reference after the recognition, the system provides the related recipes which can be prepared by using those vegetables. To enhance the classification process and promote the usability of the graphical user interface compared to the existing manual system is the main motive of the system. It’s our goal to implement a system that can classify vegetables with high accuracy.

There are certainly many ways to solve classification problem, most common and affective is Convolutional Neural Network (CNN). CNN is Deep learning algorithm that is respectively created to process pixel data. CNN contain 3 types of layers; Convolutional layers, Pooling layers, and Fully-connected layers. When filter or kernel of specific size is passed over pixel data of image with some computation, to detect feature. This process is known as convolution. Convolutional output can be affected by three components, also known as

Hyper parameters, those are, Number of filters, Stride, and Padding. Pooling layers, these layers are used to reduce the dimensionality of input. Two main types of pooling are, Max Pooling, and Average Pooling. Fully-connected layers (FC), also known as Dense layers or Top layers, these layers come at end of the architecture, every neuron of FC is directly connected to each neuron of previous layer, last FC layer with specific activation function (SoftMax or sigmoid)is responsible for models' output.

There are multiple pretrained models present for classification and detection problems, models like, MobileNetV2, VGG16, VGG19, ResNet50, InceptionV3, ExceptionV3 etc. These models are trained on millions of images, these pretrain models can be used to get effective result. We have used transfer learning on the MobileNetV2 pretrained model. To givethe user the simplest way of using the model, we have deployed this robust model into a web application with the help of TensorFlow.JS. This webpage shows the related recipes torecognized vegetables and gives the user a flawless experience of finding recipes.

2. BACKGROUND

Training inappropriate model with unprocessed data gives inferior generalization. Building affective and robust model cause thorough research and understanding of solution to similar problem. There are many stages to build successful model, which includes Data collection, Data processing, model building, model training, model evaluation, and experimenting. Following paragraphs contain references from the different papers that helped us ease the process of building, training, and deploying the model.

When tried to train the model on image with no other pre-processing than resizing, results were unpleasant. In reference to [1], before training, Data were passed through augmentation process, there was significant difference in outcome.

Most important part of creating model is model evaluation, and many times classification problem in Deep learning involves more than two class known as “multi-class classification”. Using appropriate metrics for evaluation can help easily understand the performance of the model. In [2], authors present different type of Metrics that can be used to evaluate the performance of the model, it includes, Confusion Matrix, Balance Accuracy, Accuracy, Precision and Recall, F1-scores and etc. According to the problem, Confusion Matrix, and Accuracy best suit for this project.

To achieve desire outcome on webpage, working of deployed model is most important. In [3], author presents, methodology of using client-side network using Tensorflow.JS, and explained the benefits of client-side network, which involves, keeping user data local, No dependencies, Interactive, and easy to use.

3. DATA FLOW MODEL

The initial process involves data collection where thousands of images of vegetables are gathered to form a dataset. Then, Images are split into 3 parts; train, validation, and test. Next, Data is pre-processed and augmented before feeding into the network. The necessity of each dataset and augmentation process is described in the Dataset section. Now, the MobileNetV2 model is used as the base model, after modification, it would be used as a classification model for our vegetable Dataset. The training set is used to train the model, the validation set is used to test the evaluation model. The best-suited model was achieved through the process of experimental adjustment of the batch size parameter. Once the accuracy is satisfactory the model is deployed to the web application which is the User Interface where the user will take the photo and it will be used as input to the model to classify the vegetable. The user will be prompted with output from the model’s prediction displaying the name of the vegetable and further given an option to recommend recipes based on that vegetable.

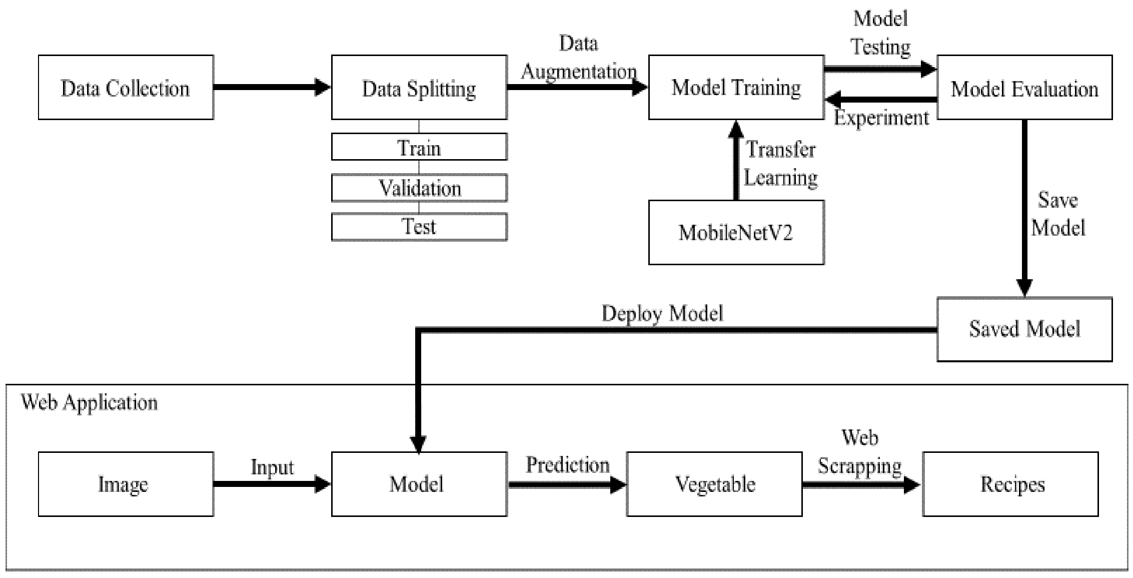


Figure 3.1: Data Flow Model

4. DEVELOPMENT OF CLASSIFICATION MODEL

The whole procedure of developing the model for the vegetable classification model is described in this section. The complete process is divided into several necessary stages.

4.1 DATASET

Data plays an essential part to develop a robust classification model, without an appropriate dataset model becomes purposeless. The complete dataset was downloaded from Kaggle datasets under the name ‘Fruits and Vegetables Image Recognition Dataset’ by Kritik Seth. According to the description, the images in the dataset were scraped from Bing Image Search. This dataset contains images of the following vegetables - cucumber, carrot, capsicum, onion, potato, lemon, tomato, radish, beetroot, cabbage, lettuce, spinach, soybean, cauliflower, bell pepper, chili pepper, turnip, corn, sweetcorn, sweet potato, paprika, jalapeño, ginger, garlic, peas, and eggplant. This dataset contains three folders: train, test, and validation, each of the folders contain subfolders for different vegetables wherein the images for respective vegetables are present. However, there is a small problem with the dataset. To develop a robust model, images on the test and validation set should be different from images on the train set. But, in this dataset images on the test and validation set were copied from the train set, which means evaluating the model with the same images would be inappropriate and makes the model impractical for real-life images. Therefore, this issue was corrected by removing duplicate images and adding unique images into the test and validation set using python script. Now, a total of almost 2600 images are divided in the following manner, the train set contains 80 images each, the test set contains 10 images each, and the validation set contains 10 images each.

4.2 AUGMENTATION PROCESS

What is Data Augmentation? Data augmentation is a technique used to artificially expand training data by modifying existing training data. Data augmentation is useful to improve the performance and outcome of machine learning models by creating new and different kinds of images to train datasets. How does it work? The transformation function sequence is a set of functions that help to modify the image into a new image. Data is passed through a transformation function sequence to get augmented data. In this project, the following transformation functions are used; rotation, zoom, width shift, height shift, shear, and horizontal flip. Also, images are converted to a fixed shape of (244x244) using the target size transformation function as MobileNetV2 was trained on images with the same shape.

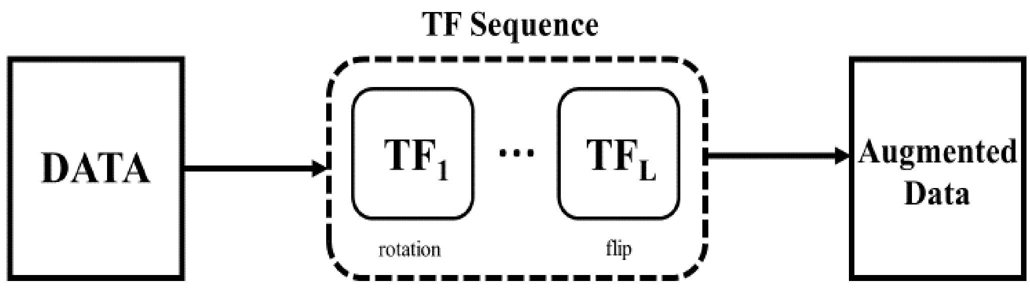


Figure 4.1: Transformation Function Sequence

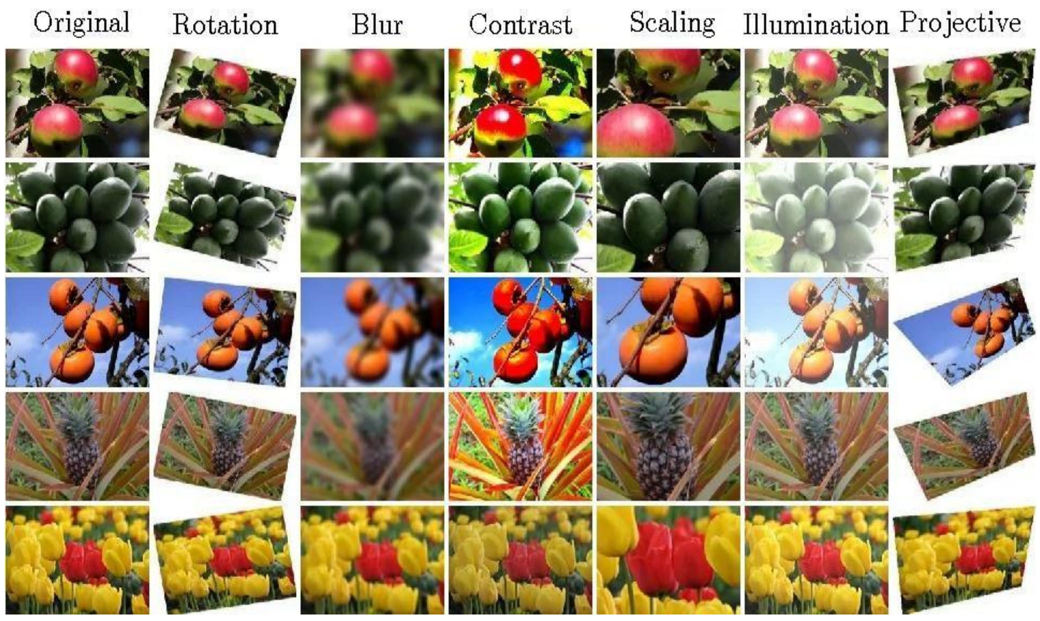


Figure 4.2: Augmentation Example

4.3 MODEL ARCHITECTURE

There are several pre-trained models are present which can be used as a base model to develop an image classification model. Why MobileNetV2? MobileNetV2 is a very effective feature extractor for classification, detection, and segmentation. Moreover, MobileNetV2 given various advantages over other networks such as VGG16, VGG19, ResNet50, InceptionV3, and ExceptionV3. MobileNetV2 is a lightweight deep neural network best suited for mobile and embedded vision applications. Reduced network size, Reduced number of parameters, and therefore, faster performance. Now, As mentioned before, the MobileNetV2 model was trained on a large dataset called ImageNet, containing over 14 million images and 1000 classes. Fully connected layers of the architecture are called Top layers, and they are responsible for the classification in the model. To use this model for our dataset to classify vegetables, some modifications needed to be done. Top layers were excluded from the original model and customized fully connected layers were added based on the number of classes in the dataset. However, ImageNet weights are used on the rest of the layers, which means weights parameters from only the top layers will be participating in the training.

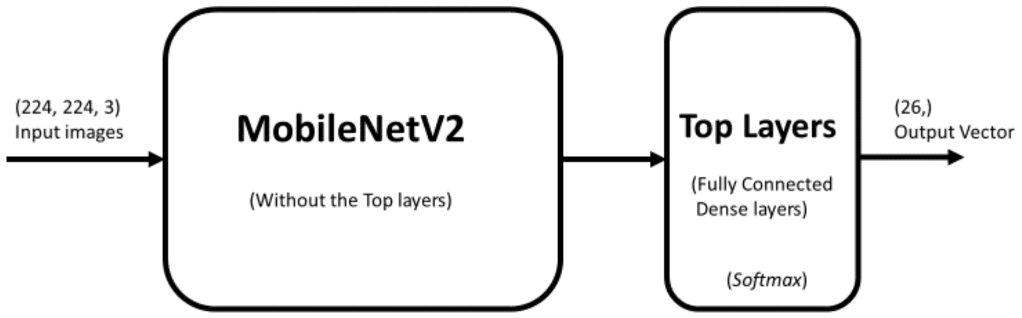


Figure 4.3: Model Architecture

4.4 MODEL TRAINING

We trained our model using TensorFlow. We used the standard Adam Optimizer with a 0.01 Learning Rate, and we used categorical cross-entropy as our loss function. The model was trained on a training dataset with over 2000 images in the batch size of 32 for 7 epochs. On Google Colab GPU, the model was trained at the rate of 10s per step for 1st epoch, then 1s per step for the rest of the epochs.

4.5 MODEL EVALUATION

The model was evaluated on the bases of accuracy. Accuracy is one of the most popular metrics in multi-class classification and it is directly computed from the confusion matrix.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

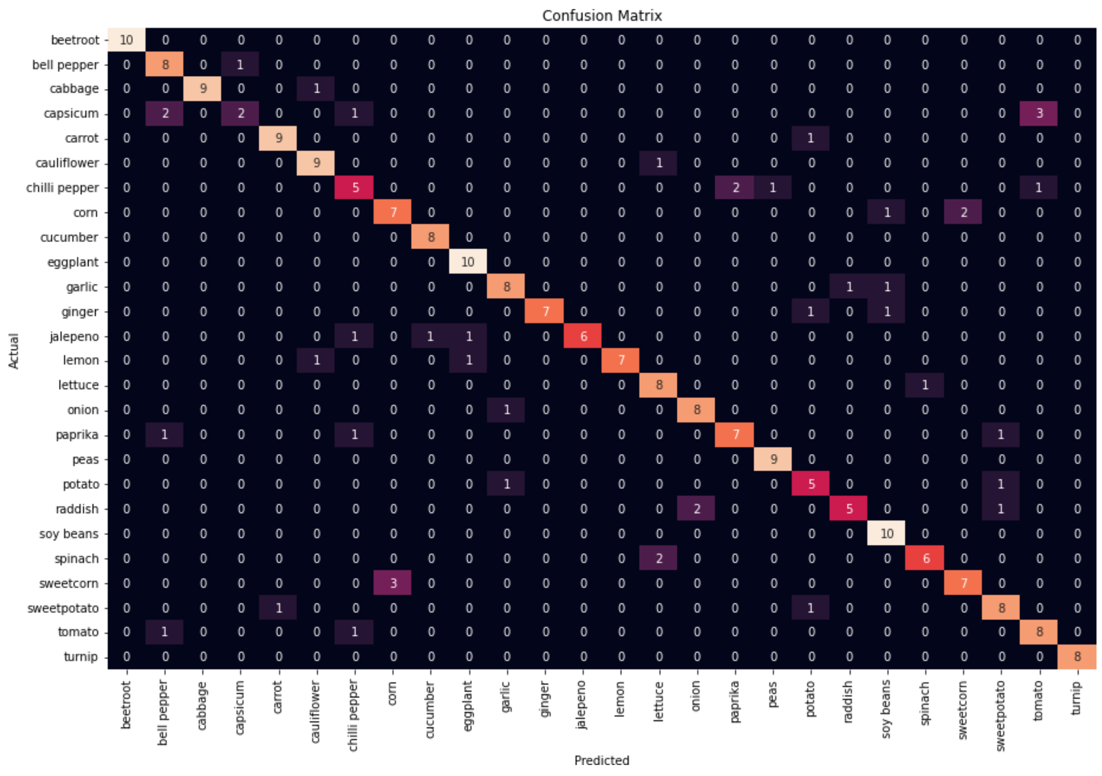
TP: True Positive

TN: True Negative

FP: False Positive

FN: False Negative

The formula of the Accuracy considers the sum of True Positive and True Negative elements at the numerator and the sum of all the entries of the confusion matrix at the denominator. True Positives and True Negatives are the elements correctly classified by the model and they are on the main diagonal of the confusion matrix, while the denominator also considers all the elements out of the main diagonal that has been incorrectly classified by the model. In simple words, consider choosing a random unit and predicting its class, Accuracy is the probability that the model prediction is correct.



	beetroot	bell pepper	cabbage	capsicum	carrot	cauliflower	chilli pepper	corn	cucumber	eggplant	garlic	ginger	jalepeno	lemon	lettuce	onion	paprika	peas	potato	raddish	soy beans	spinach	sweetcorn	sweetpotato	tomato	turnip
beetroot	10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
bell pepper	0	8	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
cabbage	0	0	9	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
capsicum	0	2	0	2	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0
carrot	0	0	0	0	9	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
cauliflower	0	0	0	0	0	9	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
chilli pepper	0	0	0	0	0	0	5	0	0	0	0	0	0	0	0	0	2	1	0	0	0	0	0	0	1	0
corn	0	0	0	0	0	0	0	7	0	0	0	0	0	0	0	0	0	0	0	0	1	0	2	0	0	0
cucumber	0	0	0	0	0	0	0	0	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
eggplant	0	0	0	0	0	0	0	0	0	10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
garlic	0	0	0	0	0	0	0	0	0	0	8	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0
ginger	0	0	0	0	0	0	0	0	0	0	0	7	0	0	0	0	0	0	1	0	1	0	0	0	0	0
jalepeno	0	0	0	0	0	0	1	0	1	1	0	0	6	0	0	0	0	0	0	0	0	0	0	0	0	0
lemon	0	0	0	0	0	1	0	0	0	1	0	0	0	7	0	0	0	0	0	0	0	0	0	0	0	0
lettuce	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8	0	0	0	0	0	0	1	0	0	0	0
onion	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	8	0	0	0	0	0	0	0	0	0	0
paprika	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	7	0	0	0	0	0	0	1	0	0
peas	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	9	0	0	0	0	0	0	0	0
potato	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	5	0	0	0	0	1	0	0
raddish	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	5	0	0	0	1	0	0
soy beans	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10	0	0	0	0	0
spinach	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	6	0	0	0	0
sweetcorn	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	7	0	0	0
sweetpotato	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	8	0	0
tomato	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8	0
turnip	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8

Figure 4.4: Confusion Matrix

As the model is pre-trained and we are using original weights, there is not much to play with hyperparameters. However, the Model was trained several times with different batch sizes and the accuracy was as follows,

Table 4.1: Batch Vs Accuracy

Batch Size	Train Accuracy	Test Accuracy
1	88.91%	74.90%
4	93.36%	74.29%
16	95.34%	81.59%
32	96.89%	85.26%
64	96.17%	80.75%

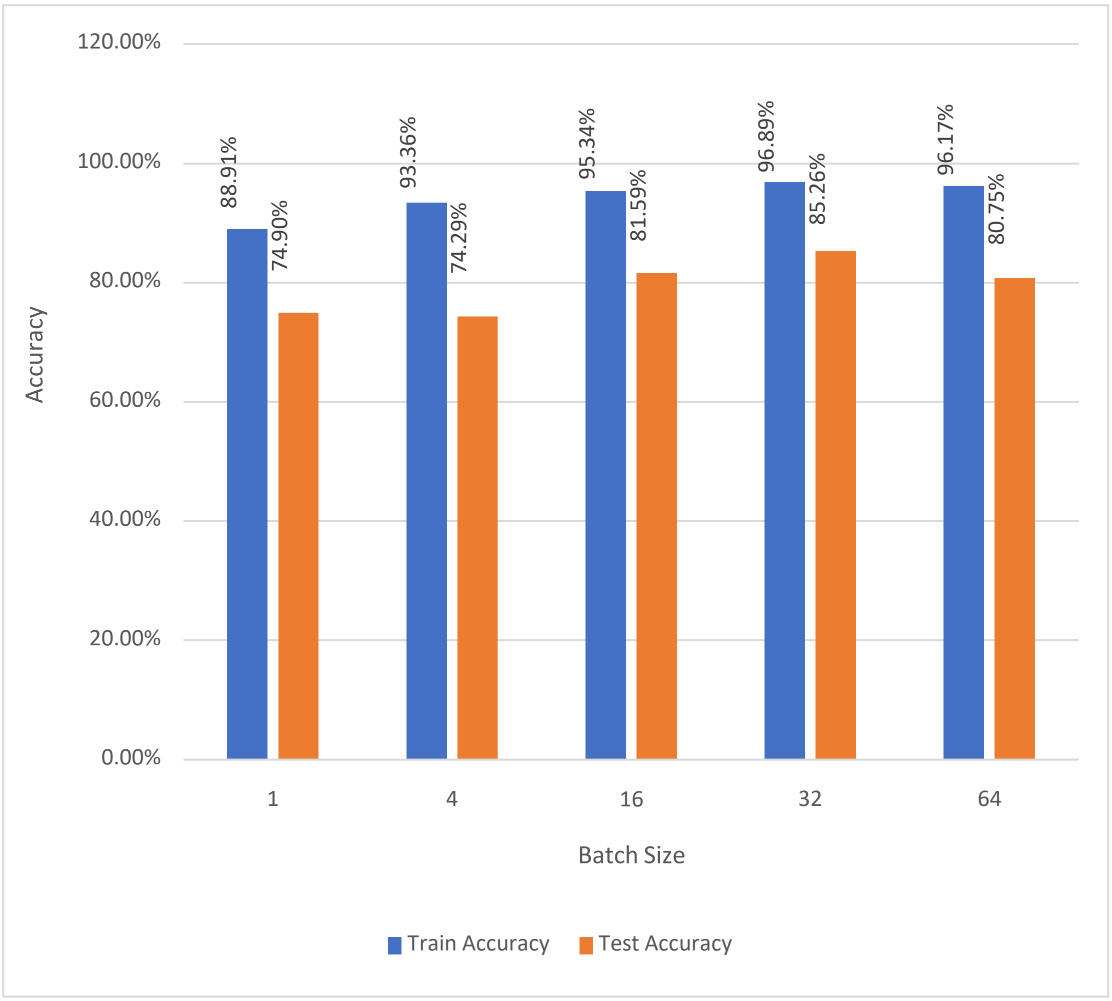


Figure 4.5: Batch Vs Accuracy Graph

From Table 4.1 and Figure 4.5, we can say that, as the size of the batch increases train accuracy increases. However, when the batch size is 32 Test Accuracy is highest i.e. 85.26%, further increment in batch size causes overfitting. Therefore, Experimental results verify the effectiveness of the model when the batch size is 32. The current model gives a test accuracy of 85.26%, which means out of 100 images 85 images will be classified correctly. The performance of the model can always be improved by using a larger dataset.

5. WEB APPLICATION

Deployment is the method to integrate a machine learning model into a webpage. To use the model in the simplest way, we deployed the model on a webpage using Tensorflow.JS. Now, a JavaScript library called Webcam.js allows users to capture a picture from the webcam, then the picture is used as input for the model, the model gives a vector of the probability of vegetables, the vegetable with the highest probability is our final output. Once the user is done scanning all the vegetables, with a click of a button, top recommended recipes from YouTube and Nutritional facts of vegetables from the U.S. Department of Agriculture’s official website are shown.

To scrape the videos from YouTube, we used YouTube Data API, with the help of this API we can easily search any query, and fetch result videos. Now to get nutritional facts, we used the U.S. Department of Agriculture food data API, we can search for any fruit, vegetable, or food for its nutritional information like Protein, Cholesterol, Fat, Energy, etc.

Styling and Designing of the web application are done using HTML and CSS, and Interactive functionality is accomplice by using JavaScript and jQuery.

REFERENCES

[1] Srdjan Sladojevic, Marko Arsenovic, Andras Anderla, Dubravko Culibrk, and Darko Stefanovic. Deep Neural Networks Based Recognition of Plant Diseases by Leaf Image Classification, 2016.

[2] Margherita Grandini, Enrico Bagli, and Giorgio Visani. Metrics for multi-class classification: An overview, 2020.

[3] Client-side neural network. **https://medium.com/agara-labs/image-classification-**