Fruits and Vegetables Classification using Progressive Resizing and Transfer Learning

Kishore M, S. B. Kulkarni, K. Senthil Babu
KS School of Engineering and Management, Karnataka, India
Associate Professor, Department of CSE, SDMCET, Dharwad

Abstract. *The customer satisfaction is the key to succeeding business. In order to work on this, time consumption is the most important factor. Currently, the method of segregation of fruits and vegetables based on category and grade is done manually. Though implemented, this method is time consuming, costly, inefficient, and a considerably difficult task. In places where food is prepared in bulk, the quality of the fruits and vegetables is often overlooked, and the overall quality of the food consumed is reduced. In order to overcome this drawback, computer vision has been introduced and implemented to segregate the products based on features like color, size, quantity, and so on. We are proposing an automatic and an effective method of evaluation for fruits and vegetables using Machine Learning techniques. The algorithm used here does not require human intervention, and the system has higher accuracy compared to the human-involved systems because it uses an automated computer algorithm. The fruits and vegetables classifier is efficient, non-destructive, and accurate which reduces man power. The existing classifier can be used in various applications such as user friendly, a self-checkout system for the visually impaired customers in supermarkets, sorting of products before releasing it to the markets.*

Keywords: Classification, Computer Vision, Deep Learning, VGG16, Progressive Resizing, Transfer Learning.

1 Introduction

Artificial Intelligence is outgrowing in demand for the agriculture sector in recent years. The application of the system consists of computer vision, machine learning algorithm which is pre-fed, image acquisition, pre-processing, and interpretation to enable the non-destructive assessment of quality in agricultural food products. The automated method in particular computer vision is used the most in classification and quality inspection of fruits and vegetables, as its reliable, faster, and efficient than the manual examination. Deep learning, being a part of machine learning, is used to build the classifier which analyses based on features like shape, color, and size. Since the color of fruits and vegetables is evident, it is chosen as the main attribute. The classifier is applicable for places where the food products are produced in bulk, in order to maintain the quality of the fruit or vegetable by classifying based on color. It can be used as a self-checkout system at supermarkets and sorting will be accurate.

Software

The software used to build a fruits and vegetables classifier is Keras. It is an open-source library that is written in the programming language Python. It focuses on being user-friendly, modular, and extensible. The interpreting language is Python. Python is an interpreted, extensible, beginner's code, with a large standard library. The dataset is collected from google.

Deep Learning

Deep Learning, a subfield of machine learning, is concerned with algorithms inspired by the structure and function of the brain and is called artificial neural networks. When given a large dataset of input and output pairs, a deep learning algorithm will try to reduce the difference between its prediction and expected output. By doing this, it tries to learn the association between given inputs and outputs, this allows a deep learning model to generalize to inputs it hasn't seen before. Deep learning methods aim at learning feature hierarchies with features that belong to higher levels of the hierarchy formed by the composition of lower-level features. Features that learn automatically at multiple levels of abstraction allows systems to map inputs to outputs, and learn complex functions directly from the data, without depending on human-crafted features.

2 Literature Review

The identification of fruits and vegetables is implemented in different areas. The most common areas are, identification in the retail business, and in areas where the purpose is to ease the harvest from the perspective of agriculture. In the retail business, the identification is mostly done manually by a cashier, or via the self-service systems in a store. [1] Classification is a fundamental research work in the field of Agriculture and Botany. Up to now, it has been found that there are hundreds of thousands of species of vegetables [4]. People will get confused because they don't know the species of vegetables. Hence, the design of a vegetable classifier will also bring ease to people's lives [1]. There are some challenges in vegetable classification, the background of a vegetable image is complex, there is a similarity between the different species of vegetables, so we cannot just rely on a single feature, such as color, shape or texture to distinguish the species of vegetables [11][8], and the same species of vegetables will be different because of the shape, scale, viewpoint and so on [9][1].

Literature on Object Classification

A classifier is an algorithm that takes a set of features characterizing objects and uses them to determine the class of each of the objects. In fruits and vegetables, these are features like shape, color, size, etc. The classifier then utilizes these attributes to determine the fruit or vegetable. There are two kinds of classification techniques: unsupervised and supervised. In supervised classification, the human expert has determined into what classes an object may be categorized and has also provided a set of samples with known classes. This set of known objects is known as the training set, as it is used by the classification programs to learn how to classify objects. Unsupervised classification is a form of pixel-based analysis. There are neither training sets nor pre-determined classes. It looks for previously undetected patterns in a data set, with no pre-existing labels, and with a minimum of human supervision. There are four steps to develop classifiers:

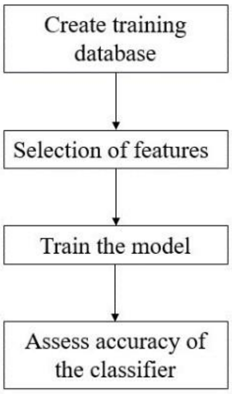


Fig. 1. Block diagram of Image Classifier.

Literature on Convolutional Neural Networks

A Convolutional Neural Network is a Deep Learning algorithm that takes in an input image, allocates weights and biases to the various objects or aspects of the image, and later returns the differences of one from the other. The pre-processing needed in the case of CNN is significantly lower when compared with other classification algorithms. While in primitive methods, filters are hand-engineered, with sufficient training, CNN has the ability to learn these filters/features.

Convolutional layers hold a vital spot on CNNs. Convolution is the mere use of a filter on an input that leads to an activation. Repeated application of the same filter on an input produces a map of activations referred to as a feature map. Feature maps indicate the strength locations and a detected feature in the input (such as an image).

The advantages of CNN versus traditional methods can be summarized as follows:

- Hierarchical feature representation. It is a multilevel representation from pixel to high-level semantic features, which is learned by a hierarchical multistage structure. It can be determined from data automatically, and hidden factors of input data can be freed through multilevel non-linear mappings.
- Compared to traditional depthless models, a deeper architecture gives rise to an exponentially increased capability to express.
- The architecture of CNN presents a chance to jointly optimize several related tasks synchronically (for example, fast R-CNN couples classification and bounding box regression into a multitask learning manner).
- Availing from the comprehensive learning capacity of deep CNNs, few computer vision challenges can also be defined as high-dimensional data transform problems and can be solved from another perspective.

Due to the above-mentioned advantages, CNN has been extensively applied in diverse research fields, such as image reconstruction, classification, image retrieval and recognition, and so on.

Literature on VGG16 Model

The VGG model is a typical and powerful CNN with a high classification rate and a high recognition rate. It goes deeper than the earlier traditional CNN architecture. To be precise, the VGG model enhances the depth of the network steadily by adding more convolutional layers, and small convolution filters (3*3) make it work successfully. [3]

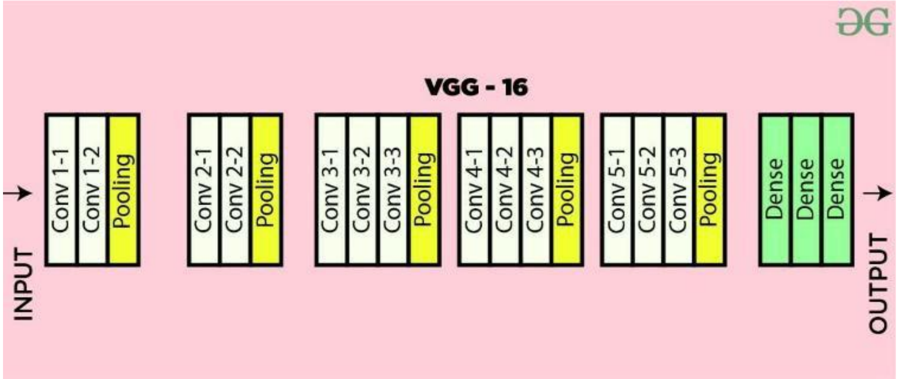


Fig. 2. Structure of VGG16 Model.

The input to the conv1 layer is a fixed 224 x 224 sized RGB image. The image is carried through a stack of convolutional layers, where the filters are used with a small 3×3 receptive field. In one of the configurations, it also uses 1×1 convolutional filters, which can be viewed as a linear transformation of input, followed by non-linearity. The stride is fixed to 1 pixel. The spatial padding of the input convolutional layer is such that the spatial resolution is retained after convolution, that is, the padding is 1-pixel for 3×3 convolutional layers. Spatial pooling is carried out by 5 max-pooling layers, which follow a few of the convolutional layers. Although, not all the convolutional layers are followed by max-pooling. Max-pooling is done over a 2×2-pixel window with stride 2.

The prediction vector has dimensions (1000, 1) and can be represented as,

$$\hat{\mathbf{y}} = \begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \vdots \\ \hat{y}_i \\ \vdots \\ \hat{y}_{1000} \end{bmatrix}$$

The loss function used for a single training example in a fully connected network is:

$$\mathcal{L}(\hat{y}, y) = -ylog(\hat{y}) - (1-y)log(1-\hat{y})$$

The loss function used for a single training example for a VGG16 model is very similar:

$$\mathcal{L}(\hat{\mathbf{y}}, \mathbf{y}) = -\sum_{i=1}^{1000} y_i log(\hat{y}_i)$$

Three Fully-Connected (FC) layers follow a stack of convolutional layers that has a specific depth depending on the architectures: the first two has 4096 channels each, the third performs 1000-way ILSVRC classification and thereby contains 1000 channels - one for each class. The final layer is a SoftMax layer, which is used in a classifier only when the classes are mutually exclusive. The configuration of the fully connected layers is fixed in all the networks.

3 Implementation

A convolutional neural network is a complex type of neural net which builds incrementally deeper-level features out of groups of pixels that is present in the input dataset images. The features that are extracted from these images are then weighted to generate a classifier and should be significant to the model's requirement. CNN's are the best among all other classifier models and perform undoubtedly well with huge data to work with.

Dataset

We consider four classes of fruits and vegetables (2 each) in our study. The four classes are Banana, Tomato, Carrot, and Potato. The limit to four is because of the amount of training time and processor power that the model takes up. The dataset has 600 images of each class for both training and validation. Also, we use 18 new images from each class for predictions. The dataset is split into training and testing sets from within the code. The dataset is extracted in such a way that the images are of almost similar resolutions and that there is no image background noise. Each class has 600 images and 18 new images for training/validation, and prediction respectively. And thus, creating a balanced dataset.

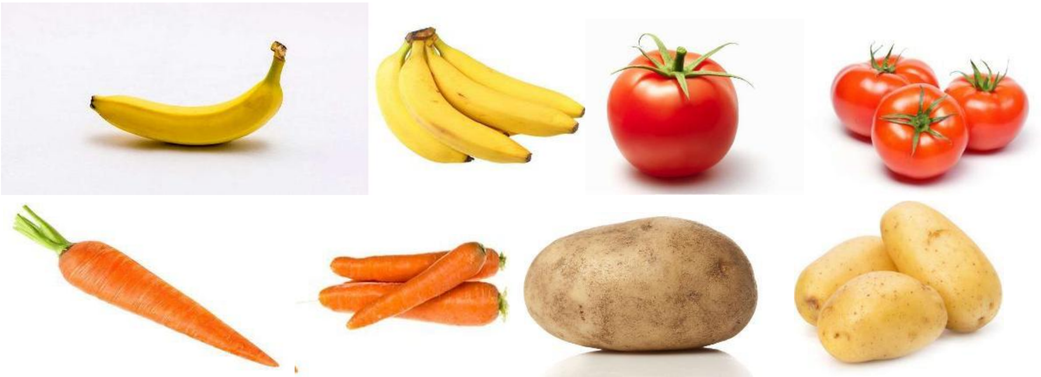


Fig. 3. Datasets (Samples) of Banana, Tomato, Carrot, Potato.

Image Preprocessing

The aim of pre-processing is to improve the image data that suppresses unwanted distortions or enhances some of its features that are crucial for further processing. Image data augmentation is a method that can be used to expand the size of a training dataset artificially by creating modified versions of images in the dataset. Augmenting the data provides much better results, especially when the dataset consists of a smaller number of images. Training deep learning neural network models on a large amount of data can result in more skillful models, and the augmentation methods can create variations of the images that can enhance the ability to the fit models in order to generalize what they have learned to new images. It is during image pre-processing that we split the dataset into two subsets, that of training and testing.

Data augmentation is a strategy that allows practitioners to significantly expand the diversity of the data available for training the models, without actually collecting any new data. Data augmentation techniques like padding, cropping, and horizontal flipping are the most commonly used techniques to train large neural networks. We use data generators, which are provided by Keras, to augment data. Data generators are on-the-spot image transformers and are said to be one of the most efficient ways of feeding image data to models in Keras. They let you work with on-disk image data which are overlarge to fit all into the memory. Pre-processing, a key approach for improving model performance is allowed especially when the model sees random image transformations and standardizations. The training data uses a mixed bag of data transformations to try and compress as much variety as possible. For the validation data, we apply just one modification, rescaling, because the validation set has to reflect "real world" performance. We are using an 80/20 train/validation split.

Progressive Resizing

Progressive Resizing is a unique technique used to build CNN models that improvise the training phases and optimization phases yielding better results.

We use three sections as follows:

- Firstly, a base model that works with input images scaled to 48x48.
- Secondly, a progressive model that works with input images scaled to 96x96.
- Lastly, a second progressive model that works with input images scaled to 192x192.

The resultant model is a “three-layer cake”: each larger-scale model subsumes the previous smaller-scale model layers and weights in its architecture. We’ve used progressive resizing to build a CNN that can learn to distinguish between four novel kinds of fruits and vegetables.

One trouble with a single neural net is that it can only work with standardized image sizes; images too small must be scaled up and those too large must be scaled down. For better model accuracy, the larger the image, the better. But there are a lot of advantages to beginning with smaller images.

To understand why we must first know that the most valuable features of an image classification problem are large. A well-tuned gradient descent typically favors robust and well-supported features in its decision-making. This optimization method tries to translate images into features holding as many pixels in as many of the samples as possible in an image classification model. While practically, when a model is trained on a set of really small images, it will learn fewer features than the one trained on a set of very large image inputs but, the traits that it learns will be the most prominent ones. Thus, a model architecture that works on an image dataset with smaller images will particularize to the larger images. Also, small-image-models are much faster to train. After all, a picture input size twice as large has four times as many pixels to learn on.

Since small-image-models stereotype well to larger input sizes, and since they take a shorter time to train, we start by training the model on a smaller sized data and scale up the images and the models later.

Model

The core of our model is VGG16, which is a pre-trained CNN architecture. This version of VGG16 is the one trained on the famed ImageNet, which includes some fruits and vegetables in its list of classes. We add a new top layer consisting of quite a large, fully connected layer with modest regularization in the form of dropout. Since there are four output classes, the output layer has four nodes. We then freeze the VGG16 model, i.e., we do not train any of the weights; we use it as-is.

Applying Progressive Resizing

Initially, we build a classifier that performs well on small n x n (48x48) images. The next step is to scale our model up to 2n x 2n (96x96) images. This is done using Transfer Learning. Transfer learning is a technique of re-using layers and weights from previous models when building new ones. In our case, this means taking the model we just created, freezing it so that further training won’t change the existing weights, and injecting its layers into a new model that takes upscaled 96x96 images as input. The work done at this stage is limited to finding a configuration of good feeder layers we can prefix our old model with. These new layers can concentrate on finding the further features findable in 96x96 scaled images that weren’t in the 48x48 scaled images. We first save the architecture and the weights of the model that was trained on 48x48 images.

We import the saved model and remove the first two layers of the model, i.e., the input layer and the first convolutional layer that took the 48x48 image and convolved it. Next, we reattach the rest of the layers of our old model upon the new one and fix the old convolutional and fully-connected layer weights in place. Basically, we create a new model that trains on an imageset of 96x96 pixel images which reuses the old 48x48 classifier internally. The model is trained, and we then save the architecture and weights of the model.

Progressive Resizing can be applied one more time, this time scaling from 96x96 to 192x192. The procedure remains the same, differing only in scale.

4 Result and Discussion

Comparison Model

A comparison model was generated with the following algorithms using the same datasets:

- Logistic Regression (LR)
- Linear Discriminant Analysis (LDA)
- K – Nearest Neighbor Algorithm (KNN)
- Decision Tree Learning (CART)
- Random Forest Algorithm (RF)
- Naive Bayes Algorithm (NB)
- Support Vector Machine (SVM)

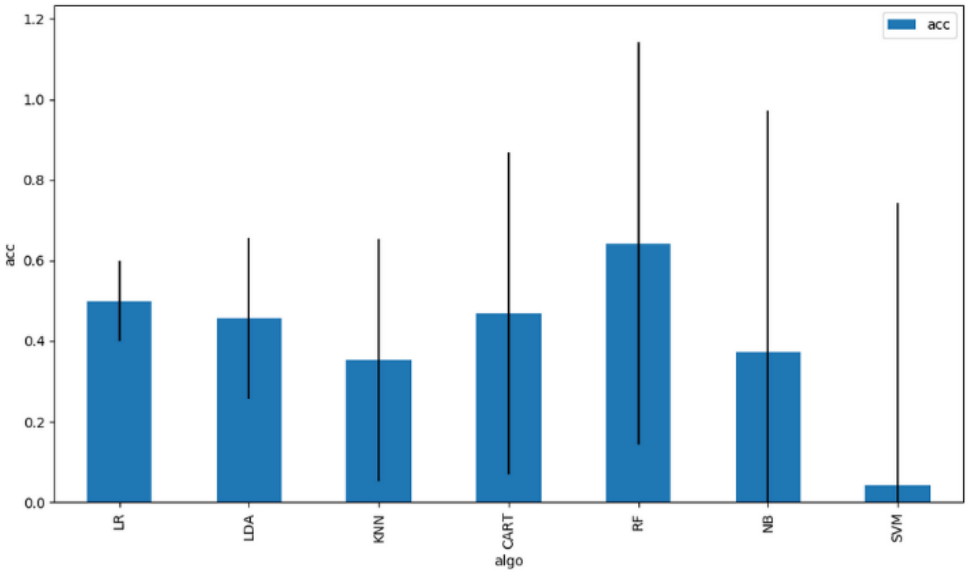
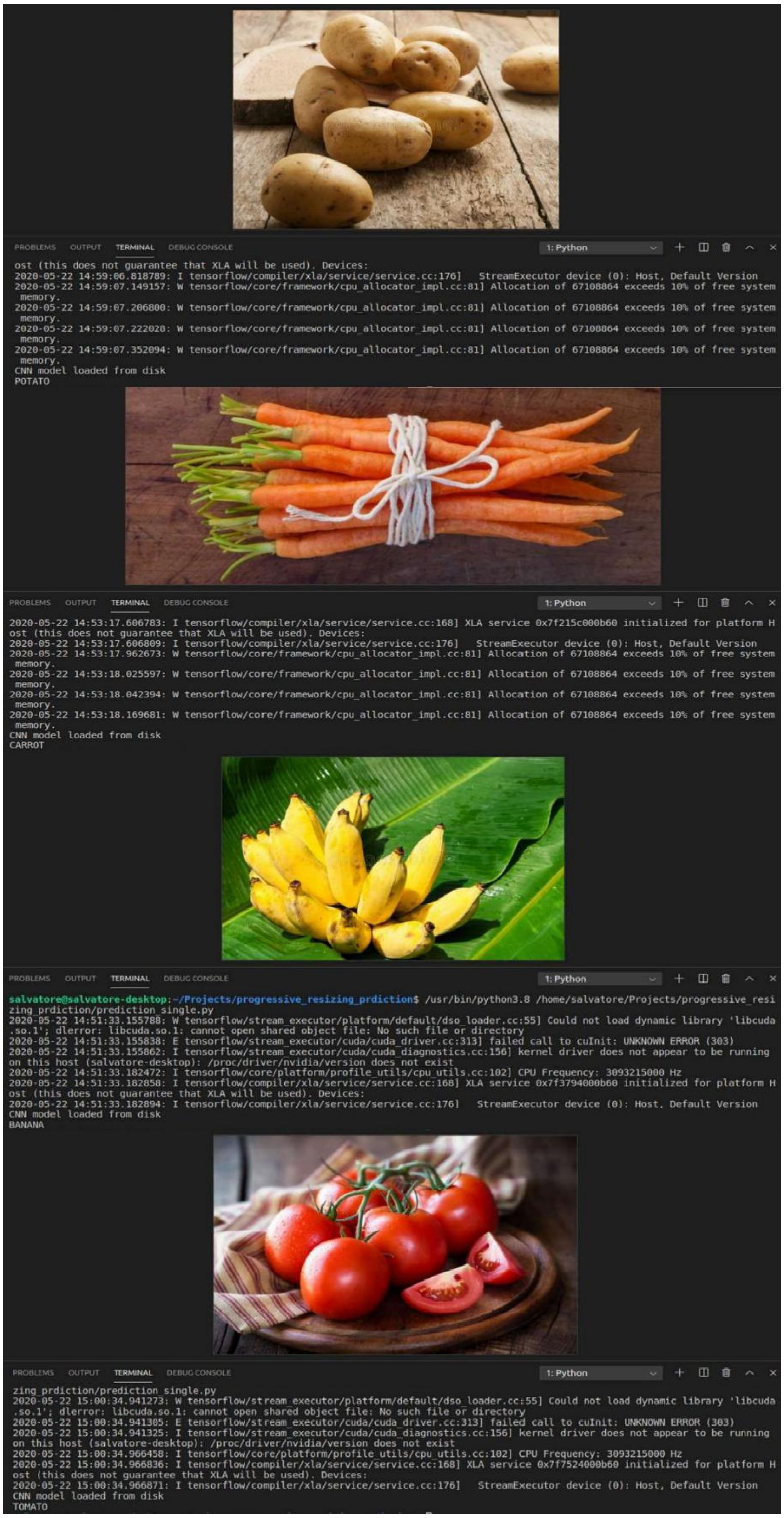


Fig. 4. Comparison Model.

By this analysis, we decided to confirm our model using CNN algorithm, and build a VGG16 model to classify the fruits and vegetables.

Prediction Model

The prediction model was built on 4 classes of fruits and vegetables – Tomato, Banana, Carrot and Potato. It has been trained to classify and recognize the fruit or vegetable displayed or provided as the input.



```
ost (this does not guarantee that XLA will be used). Devices:
2020-05-22 14:59:06.818789: I tensorflow/compiler/xla/service/service.cc:176]   StreamExecutor device (0): Host, Default Version
2020-05-22 14:59:07.149157: W tensorflow/core/framework/cpu_allocator_impl.cc:81] Allocation of 67108864 exceeds 10% of free system
 memory.
2020-05-22 14:59:07.206800: W tensorflow/core/framework/cpu_allocator_impl.cc:81] Allocation of 67108864 exceeds 10% of free system
 memory.
2020-05-22 14:59:07.222028: W tensorflow/core/framework/cpu_allocator_impl.cc:81] Allocation of 67108864 exceeds 10% of free system
 memory.
2020-05-22 14:59:07.352094: W tensorflow/core/framework/cpu_allocator_impl.cc:81] Allocation of 67108864 exceeds 10% of free system
 memory.
CNN model loaded from disk
POTATO
```

```
2020-05-22 14:53:17.606783: I tensorflow/compiler/xla/service/service.cc:168] XLA service 0x7f215c000b60 initialized for platform H
ost (this does not guarantee that XLA will be used). Devices:
2020-05-22 14:53:17.606809: I tensorflow/compiler/xla/service/service.cc:176]   StreamExecutor device (0): Host, Default Version
2020-05-22 14:53:17.962673: W tensorflow/core/framework/cpu_allocator_impl.cc:81] Allocation of 67108864 exceeds 10% of free system
 memory.
2020-05-22 14:53:18.025597: W tensorflow/core/framework/cpu_allocator_impl.cc:81] Allocation of 67108864 exceeds 10% of free system
 memory.
2020-05-22 14:53:18.042394: W tensorflow/core/framework/cpu_allocator_impl.cc:81] Allocation of 67108864 exceeds 10% of free system
 memory.
2020-05-22 14:53:18.169681: W tensorflow/core/framework/cpu_allocator_impl.cc:81] Allocation of 67108864 exceeds 10% of free system
 memory.
CNN model loaded from disk
CARROT
```

```
salvatore@salvatore-desktop:~/Projects/progressive_resizing_prdiction$ /usr/bin/python3.8 /home/salvatore/Projects/progressive_resi
zing_prdiction/prediction_single.py
2020-05-22 14:51:33.155788: W tensorflow/stream_executor/platform/default/dso_loader.cc:55] Could not load dynamic library 'libcuda
.so.1'; dlerror: libcuda.so.1: cannot open shared object file: No such file or directory
2020-05-22 14:51:33.155838: E tensorflow/stream_executor/cuda/cuda_driver.cc:313] failed call to cuInit: UNKNOWN ERROR (303)
2020-05-22 14:51:33.155862: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:156] kernel driver does not appear to be running
on this host (salvatore-desktop): /proc/driver/nvidia/version does not exist
2020-05-22 14:51:33.182472: I tensorflow/core/platform/profile_utils/cpu_utils.cc:102] CPU Frequency: 3093215000 Hz
2020-05-22 14:51:33.182858: I tensorflow/compiler/xla/service/service.cc:168] XLA service 0x7f3794000b60 initialized for platform H
ost (this does not guarantee that XLA will be used). Devices:
2020-05-22 14:51:33.182894: I tensorflow/compiler/xla/service/service.cc:176]   StreamExecutor device (0): Host, Default Version
CNN model loaded from disk
BANANA
```

```
zing_prdiction/prediction_single.py
2020-05-22 15:00:34.941273: W tensorflow/stream_executor/platform/default/dso_loader.cc:55] Could not load dynamic library 'libcuda
.so.1'; dlerror: libcuda.so.1: cannot open shared object file: No such file or directory
2020-05-22 15:00:34.941305: E tensorflow/stream_executor/cuda/cuda_driver.cc:313] failed call to cuInit: UNKNOWN ERROR (303)
2020-05-22 15:00:34.941325: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:156] kernel driver does not appear to be running
on this host (salvatore-desktop): /proc/driver/nvidia/version does not exist
2020-05-22 15:00:34.966458: I tensorflow/core/platform/profile_utils/cpu_utils.cc:102] CPU Frequency: 3093215000 Hz
2020-05-22 15:00:34.966836: I tensorflow/compiler/xla/service/service.cc:168] XLA service 0x7f7524000b60 initialized for platform H
ost (this does not guarantee that XLA will be used). Devices:
2020-05-22 15:00:34.966871: I tensorflow/compiler/xla/service/service.cc:176]   StreamExecutor device (0): Host, Default Version
CNN model loaded from disk
TOMATO
```

Fig. 5. Prediction Screenshots.

The validation accuracy progressed from 90.85% for the 48 x 48 classifier to 95.77% for the 96 x 96 classifier and finally, the model gained a 97.23% accuracy for the 192 x 192 classifier. The prediction accuracy came up 95.83% with loss in accuracy observed for the Potato class. This can be overcome by using a larger dataset or by scaling the model further to 384 x 384. The model performance and methodology is effective for real-time use.

Since VGG16 on its own is a very dense CNN model, progressive resizing on this model is time-consuming and hardware intensive. Each progression took around 3 to 6 hours of training time on Google Colab, which provides a single 12GB NVIDIA Tesla K80 GPU. Therefore, the training was restricted to just four classes and 600 images per class.

By increasing the number of classes and compressing the neural net with techniques like Weight Pruning or Quantization, or SVD decompositions of Weight Matrices, the model can be called to higher resolutions and thus can achieve better accuracies for significantly lesser training time.

Discussion

Through our analyses, we were able to develop several models that classify fruits based on an image. Given the nature of our dataset, it was crucial that we first started by pre-processing our data using progressive-resizing. By implementing this feature extraction method, which reduced the dimensionality of our dataset on scaling them up and fine- tuning performance, we were able to achieve high accuracies, even with our plain dataset models. From there, we developed more complex neural networks, which gave us our best accuracy scores.

Utilizing neural networks made it easier to further train and increase accuracy, and over time, images with more noise could be added to train our models to classify the fruit/vegetable in the noisy image. Also, more fruits and vegetables can be added and trained- even things like seeds, nuts, bread, and pastries could be added for classification purposes (This point can be used in future work as well). While our highest performing model is a customized transfer learning model that is the VGG16 model providing the best accuracies for classification along with progressive resizing. This custom model can provide the best results for high dimensional and noisier images and for a huge image dataset.

In the project, the implementation has been done as a Python script. This can be converted into a standalone application so that it can be run on any computer. The fruits and vegetable classifier model can be deployed into a processor or microcomputer, like the Raspberry Pi Module which can be interfaced with a camera and screen. The camera detects the real-time object and the module can classify the product, displaying the name. This makes the system more compact and portable This can be enhanced with the value of the product, which can make a more efficient and easier self-checkout system in the market. Features like dark spots, ripe color, and so on can also be considered to evaluate the quality of the fruit or vegetable, thereby, increasing the quality analysis of the same.

References

1. Frida Femling, Adam Olsson, Fernando Alonso- Fernandez **'Fruit and Vegetable Identification Using Machine Learning for Retail Applications'** School of Information Technology, Halmstad University, October 23 2018.
2. Guoxiang Zeng **'Fruit and Vegetables Classification System Using Image Saliency and Convolutional Neural Network'** 2017 IEEE 3rd Information Technology and Mechatronics Conference (ITOEC), October 3-5, 2017.
3. Mayar Haggag, Samaa Abdelhay, Asma Mecheter, Samer Gowid, Farayi Musharavati, Saud Ghani **'An Intelligent Hybrid Experimental-Based Deep Learning Algorithm for Tomato-Sorting Controllers'** IEEE Access (Volume 7), August 2nd, 2019.

4. Web Article: **Boost your CNN image classifier performance with progressive resizing in Keras**.
5. L. P. Deshmukh, M. S. Kasbe, T.H.Mujawar, S.S.Mule, A.D.Shaligram **'A Wireless Electronic Nose (WEN) for the Detection and Classification of Fruits (A Case Study)'** 2016 International Symposium on Electronics and Smart Devices (ISESD), November 29-30, 2016.
6. Toshiki Nishi, Shuichi Kurogi, Kazuya Matsuo **'Grading Fruits and Vegetables using RGB-D images and convolutional neural network'** 2017 IEEE Symposium Series on Computational Intelligence (SSCI), November 27-December 1, 2017.
7. Ni Made Satvika Iswari, Wella, Ranny **'Fruitylicious: Mobile Application for Fruit Ripeness Determination based on Fruit Image'** 2017 10th International Conference on Human System Interactions (HSI). July 17-19, 2017.
8. Md Khurram Monir Rabby, Brinta Chowdhury, Jung H. Kim **'A Modified Canny Edge Detection Algorithm for Fruit Detection & Classification'** 2018 10th International Conference on Electrical and Computer Engineering, December 20-22, 2018.
9. Izadora Binti Mustaffa, Syawal Fikri Bin Mohd Khairul **'Identification of Fruit Size and Maturity Through Fruit Images Using OpenCV-Python and Rasberry Pi'** 2017 International Conference on Robotics, Automation and Sciences (ICORAS), November 27-29, 2017.
10. Om Patil, Prof. (Dr.) Vijay Gaikwad **'Classification of Vegetables using TensorFlow'** International Journal for Research in Applied Science & Engineering Technology (IJRASET), April, 2018.
11. R. Shantha Selva Kumari, V. Gomathy **'Fruit Classification using Statistical Features in SVM Classifier'** 2018 4th International Conference on Electrical Energy Systems (ICEES), February 7-9, 2018.
12. Siyuan Lu, Zhihai Lu2, Soriya Aok, Logan Graham **'Fruit Classification Based on Six Layer Convolutional Neural Network'** 2018 IEEE 23rd International Conference on Digital Signal Processing (DSP), November 19-21, 2018.
13. Sitti Wetenriajeng Sidehabi, Ansar Suyuti, Intan Sari Areni, Ingrid Nurtanio **'Classification on Passion Fruit's Ripeness using K-Means Clustering and Artificial Neural Network'** 2018 International Conference on Information and Communications Technology (ICOIACT), March 6-7, 2018.
14. Anuja Bhargava, Atul Bansal **'Fruits and vegetables quality evaluation using computer vision - A review'** Journal of King Saud University, 2018.
15. AKNS Aishwarya Chandini, Uma Maheswari B **'Improved Quality Detection Technique for Fruits Using GLCM and MultiClass SVM'** 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI), September 19-22, 2018.