Deep Learning - MSAI 495

Know Your Food:

Image Based Classification and Nutrition Retrieval

Final Project Report

February 21st, 2022

Professor
Reda Al-Bahrani

Students
Shubham Shahi
Geethanjali Vasudevan
Shraddha Bangad

1. ABSTRACT

Recently we have seen a rapid increase in dietary ailments, caused due to unhealthy food routines. Dietary assessment systems which can record the real time images of the meal and examine its nutritional content can be very useful and improve the dietary habits of people. Fruits and vegetables play a significant role in human nutrition, especially as sources of vitamins, folacin, minerals, and dietary fiber. Without being aware of their nutritional value, we often tend to undervalue them in our diet. This project proposes a hybrid image based nutritional assessment system using Convolutional Neural network. Our experiments are conducted on images collected from the kaggle datasets. To achieve our goal, we proposed two models. The first model was custom built using residual block and batch normalization and the other model was built using RESNET 34 which is a 34 layer convolutional neural network that can be utilized as a state-of-the-art image classification model. Performance of the model was evaluated by comparing it to a baseline model. We achieved an accuracy of 99% for our custom model and an accuracy of 94% for RESNET 34.

KEYWORDS

Convolutional neural network, transfer learning, nutritional value, fruit recognition, vegetable recognition.

2. LITERATURE SURVEY

Fruit and vegetable recognition task leverages grocery stores with automating labeling and computation of the price. It is an important and difficult task, since it is necessary for the cashier to know the categories of fruit and vegetables available at a store at a given time in order to determine its price. Even though bar codes have partially elevated this work for packaged products, unpackaged items picked by customers and weighed still face a major issue.

Many researchers have come up with different kinds of solutions for this problem. From extracting features using scale-invariant feature transform to extract interest point descriptors of fruits and vegetables, clustering algorithm for fruit and vegetable classification[2], applying KNN euclidean distance metric to measure the distance between the attributes of the unknown fruit with the stored fruit examples. [3], combining three different features- color, shape, and size to perform sequential pattern classification. [4], and Fisher vectors[5]. The aforementioned techniques may have one or several of the following shortcomings. Some required extra sensors like invisible light or weight sensors, some were not able to classify various fruits and vegetables from different angles and position, the same fruit while others were not robust and scalable.

3. DATASET

For our project, our goal was to find data which has multiple images of fruits and vegetables. We choose a dataset from kaggle which has 90483 images labeled across 131 classes of fruits and vegetables. The image size throughout the dataset is 100x100 pixels.

The Training set consists of 67692 images. It has one fruit or vegetable per image. We also have 103 images that have multiple fruits per image. This diversifies our dataset and trains models to differentiate and classify efficiently when multiple classification is required for an image. We would also be using image augmentation to diversify the dataset more.

The Test set size has a total of 22688 images and would be used for final testing only.

Image 1: Sample image of the dataset

4. EDA & DATA CHALLENGES

1. Image Size:

The image size in the dataset is of 100x100 was small which made the images visually of low quality and are similar in terms of colors, shapes and sizes.

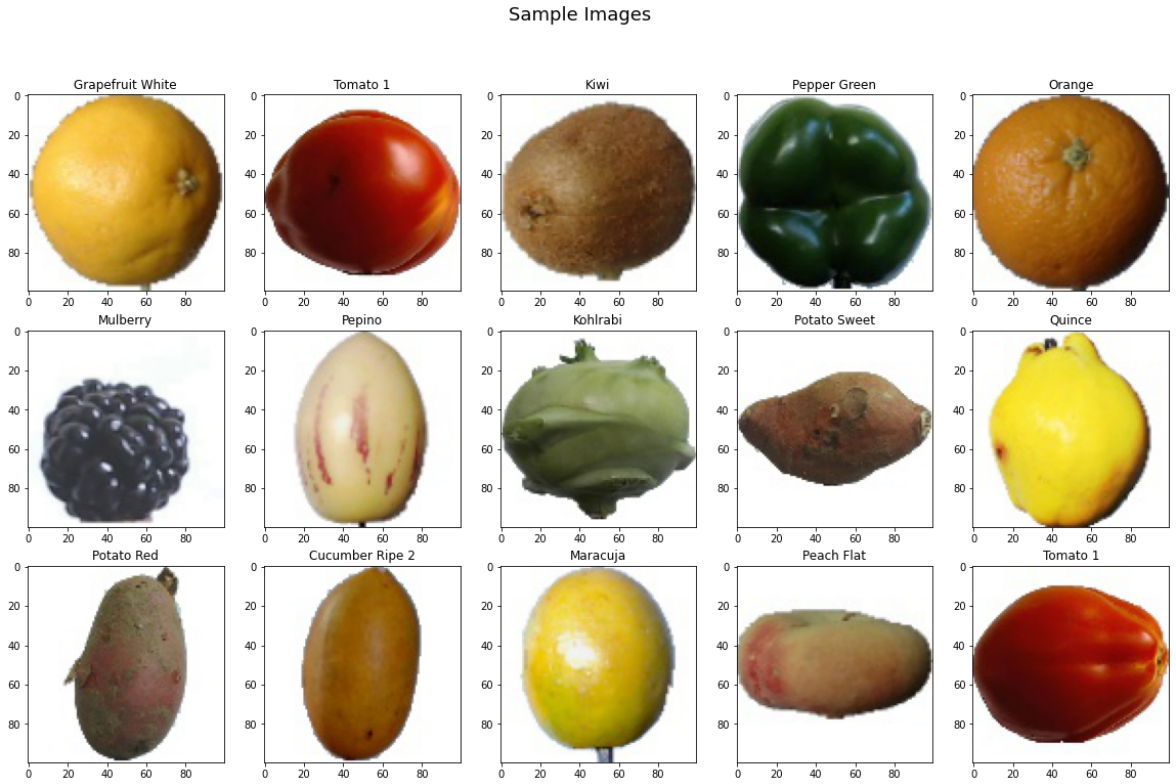


Image 2: Sample image to scale the fruits and vegetable images in the dataset

2. Orientation:

With 131 classes to identify, we observed that the dataset has images of the same fruit in different orientations.

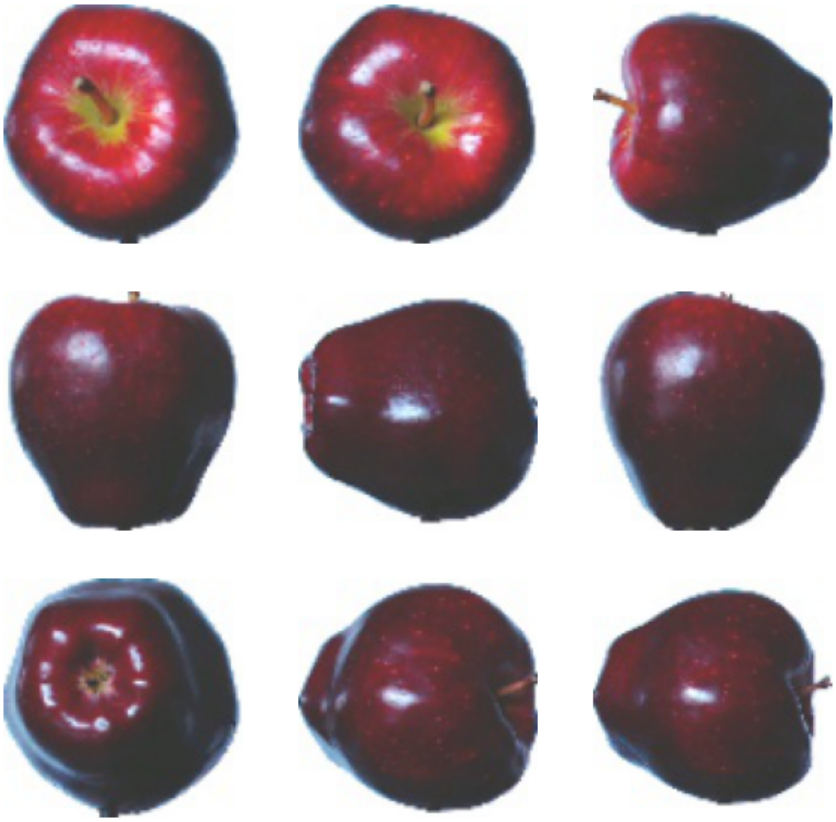
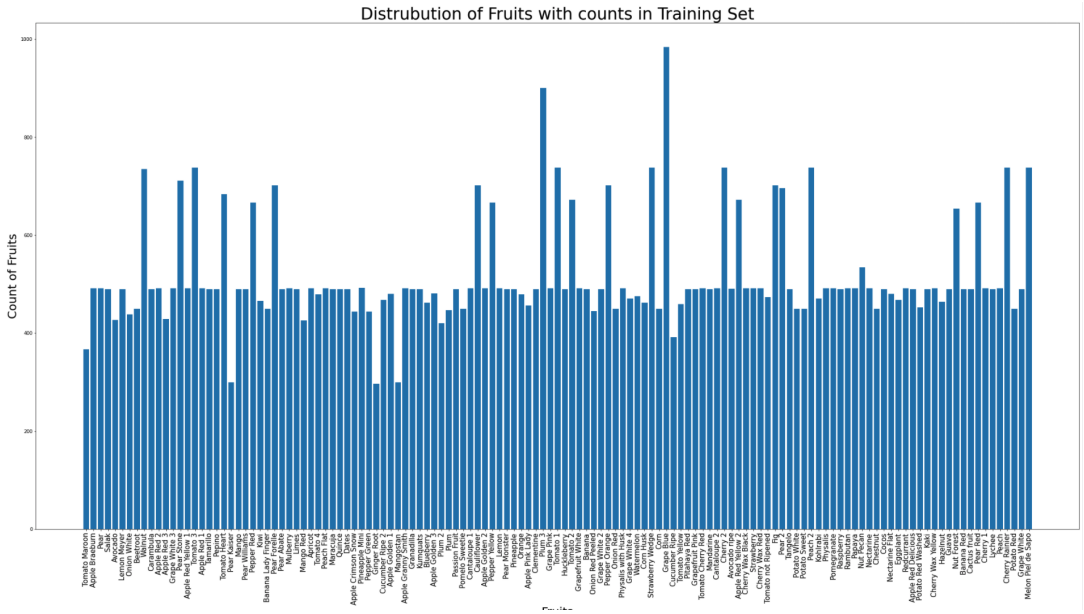


Image 3: Image showing different orientations of an apple

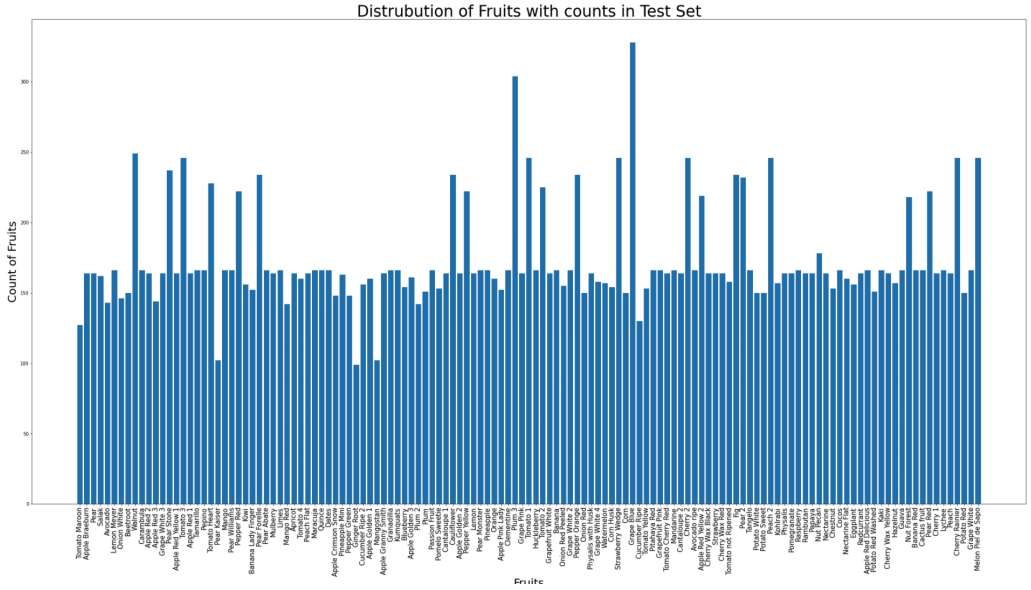
It was crucial that we take into account to train the model to identify the same apple from different angles and orientation. To overcome this challenge we have build our own classifier that recognizes the fruit irrespective of shape, position and orientation

3. Data Distribution:

The Dataset is evenly distributed but we observe that there were few classes of fruits which had more images. This posed a challenge as without pre-processing the model would be predicting a few classes better than the others.



Graph 1: Data Distribution of fruits in Training Set



Graph 2: Data Distribution of fruits in Test set

5. DATA PREPROCESSING

5.1 Splitting train data into train and validation sets

Splitting the data is a necessity to eliminate bias to training data in ML algorithms. Revising parameters of a ML model to best fit the training data commonly results in an overfit algorithm that performs poorly on actual test data. For this reason, we split the dataset into multiple subsets on which we train different parameters. We split the dataset into 3 parts:

1. **Training set:**

   This set of data is used to train and make the model learn the hidden patterns in the dataset. The training set helps us compute the losses and adjust the weights of the model using gradient descent

2. **Validation set:**

   This set is used to validate our model performance during the time of training. This validation process gives information that helps us tune the model's hyperparameters and configurations accordingly.

3. **Test set:**

   This set is used to test the data after the training is complete. It provides an unbiased final model performance metric in terms of accuracy, precision, etc. It answers the question of "How well does the model perform?"

5.2 Data Loader

We have used a data loader to help in loading and transforming the data, which is almost ready for training. Shuffle is set to TRUE for training data. This was to make sure that the batches generated in each epoch are different. Through this randomization we were able to generalize & speed up the training process.

Moreover, the Validation Set is used for evaluating the model, hence we decided not to shuffle the images.

5.3 Data Preprocessing

1. **Data Normalization:**

   While going through the dataset we observed different pixel intensities like shown in Fig: .

   To tackle this we have normalized the dataset to visually enhance the images.

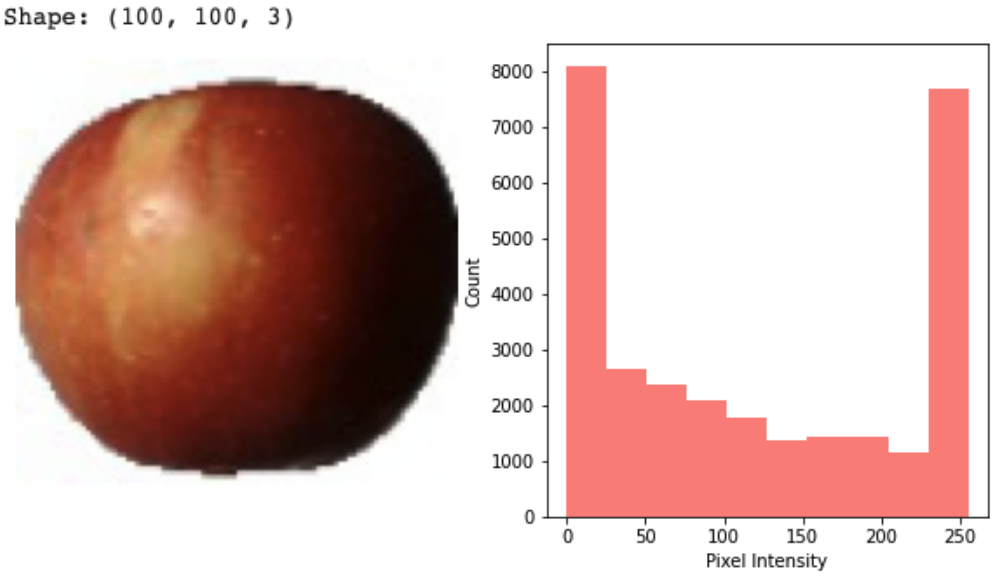


Image 4: Image showing pixel intensity

2. **Random Transformation:**
   For Random transformation we have randomly changed the brightness/contrast of the images, Crop images at random location.

3. **Flip the image horizontally:**
   Changed the orientation of the image.

4. **Padding by 10%:**
   Added padding to the sides by 10%.

5. **Random Rotations of 20 degrees:**
   Rotated the images to change the angle and orientation of the images

After the transformations the data looked like:

Image 5: Sample image of the Dataset after applying different transformations

6. METHODOLOGY

As we have a total of 67692 training images, which we have split into smaller batches and used a batch size of 32 for our problem.
We have experimented with two different models.

1. Custom model with Residual blocks and Batch normalization
2. Residual neural networks, which are pre-trained models with the ImageNet dataset.

1. Custom CNN Model:

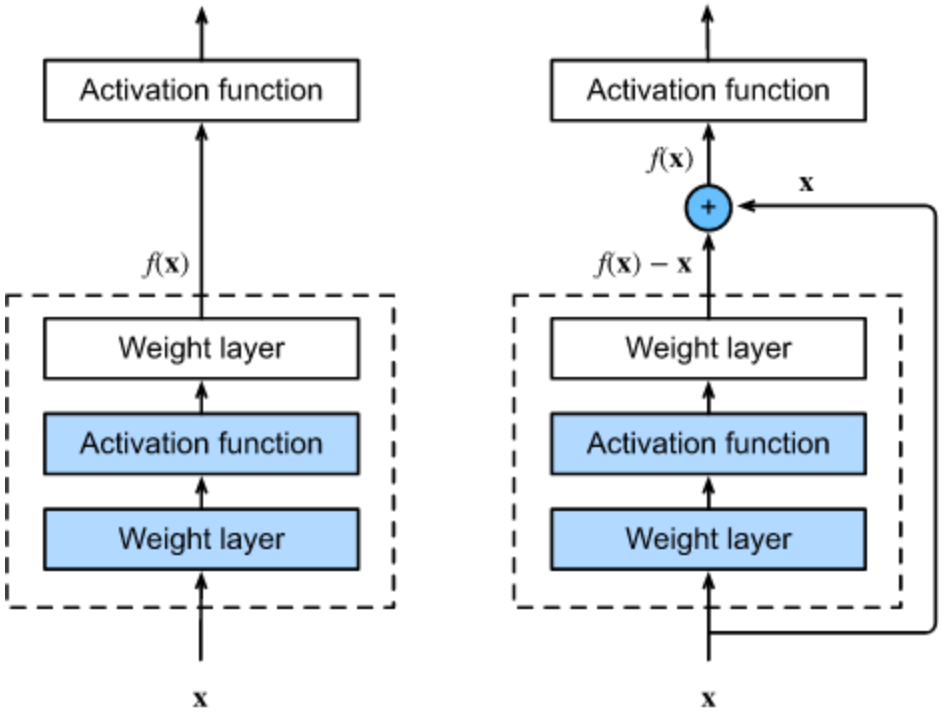


Image 6 : difference between regular block(left) and residual block(right)

The main purpose of using residual blocks is that it adds the original input back to the output feature map, which is computed by passing it through one or more convolutional layers. Here, we have used 3

convolutional layers before every residual block. In our Custom CNN, we have a total of 9 convolutional layers and a residual block after every 3 convolutional layers.

2. ResNet-34 CNN Model:

ResNet stands for Residual Neural Network. Resnet34 is a 34 layer convolutional neural network. This is a model that has been pre-trained on the ImageNet dataset. However, it is different from traditional neural networks in the sense that it takes residuals from each layer and uses them in the subsequent connected layers.

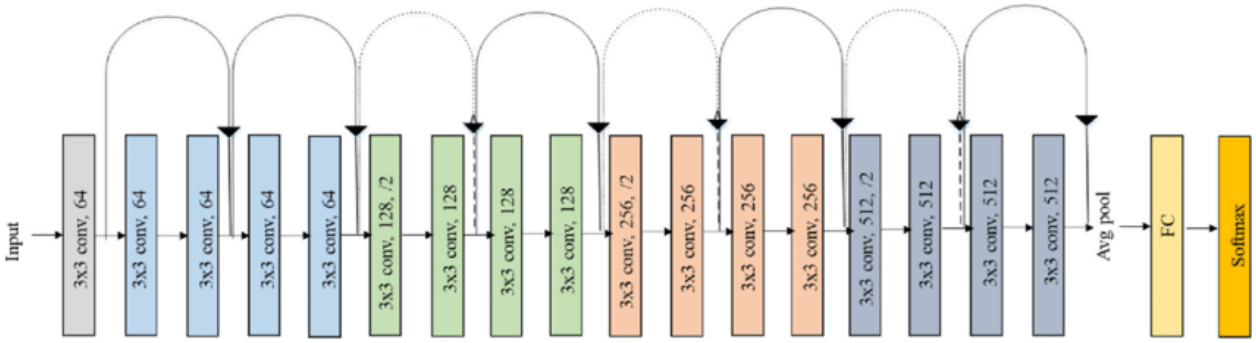


Image 7 : Resnet model architecture

7. MODEL TRAINING

Hyperparameter tuning:

- **Learning rate scheduling:**
  - Model uses a learning rate scheduler to change the learning rate after every batch of training. Amongst the various learning rate scheduler strategies, we chose “One Cycle Learning Rate Policy”. This scheduler starts with a low learning rate, gradually increasing it batch-by-batch to a

high learning rate for about 40% of epochs, then gradually decreasing it to a very low value for the remaining epochs.
    - This helps us in picking the right learning rate at different iterations, thus making the model converge quickly. The one cycle policy gives very fast results when training complex models. It follows the Cyclical Learning Rate (CLR) to obtain faster training time with regularization effect but with a slight modification.
- **Weight Decay:**
    - This regularization technique prevents the weight from becoming too large. To do this, it adds an additional term of the loss function.
- **Gradient Clipping:**
    - This technique prevents undesirable changes in parameters caused by large gradient values. This is achieved by limiting the values of gradients to a small range.

Batch Size	32
Epochs	10
One cycle Learning rate Policy	(max) 1e-3
Weight Decay	1e-4
Gradient Clipping	1e-1
Optimizer	Adam

Table 1:Model hyperparameter Specification

RESULTS

After training these are the results:

Custom Model:

- Training Loss: 0.001

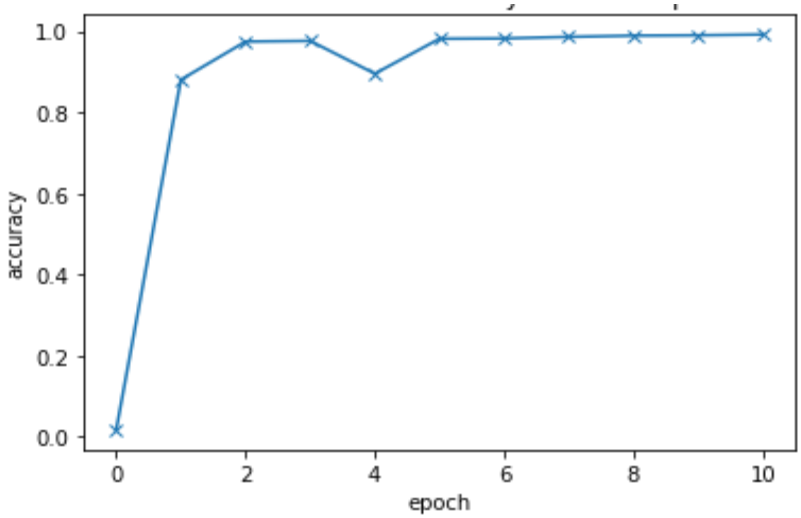
- Validation Loss: 0.04
- Validation Accuracy: ~99%

The ResNet model :

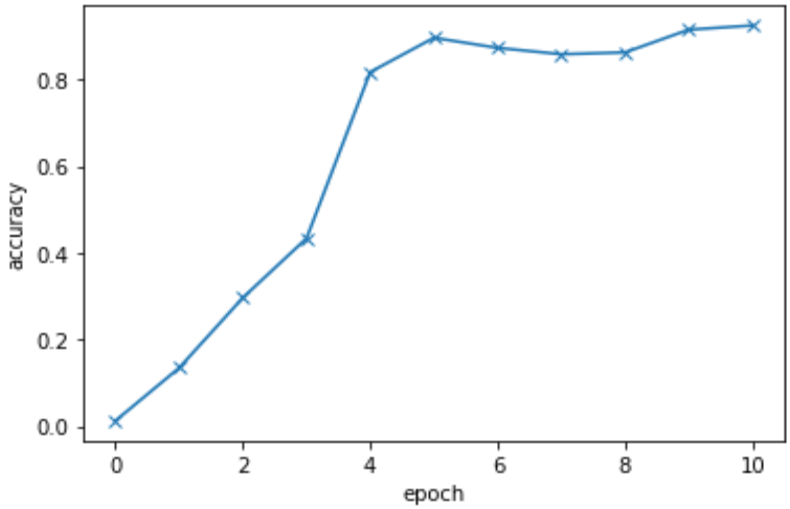
- Training Loss: 3.810
- Validation Loss: 3.896
- Validation Accuracy: ~94%

Architecture	**Custom**	**Resnet**
Training loss	0.001	3.810
Validation loss	0.04	3.876
Validation accuracy	~99%	~94%

Table 2: Comparison of different architecture

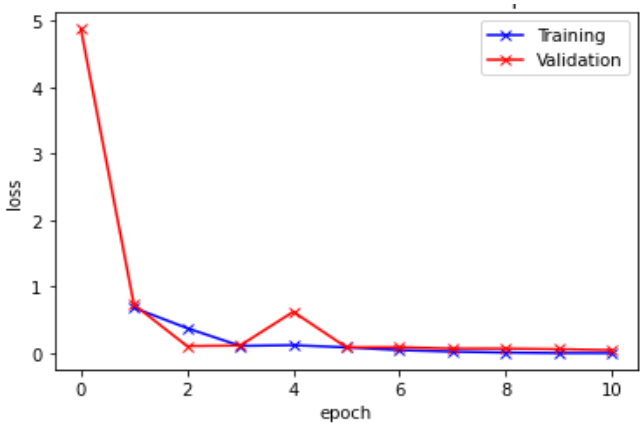


Graph 3: Custom CNN Model: accuracy vs No of Epochs

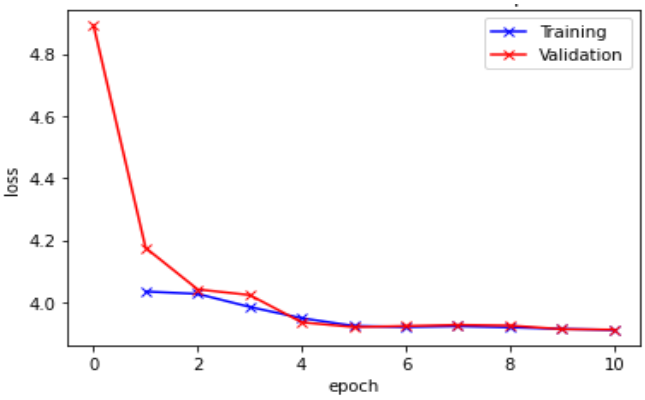


Graph 4: ResNet CNN Model: Accuracy vs No. of Epochs

The custom model training took 2.5 times, the time consumed by the ResNet model, but the custom model had better results which is evident in the graphs- **the custom model** reached ~**99%**, meanwhile the **ResNet model** only got ~**94.**



Graph 5: Custom CNN Model: Loss vs No. of Epochs



Graph 6: ResNet CNN Model: Loss vs no. of Epochs

Another considerable thing is that the custom model **reduced quite well** the training and validation losses, while the ResNet model did not perform as well.
Some of the predictions from the models are presented below:

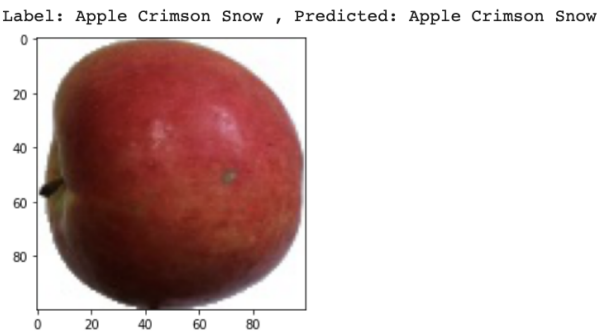
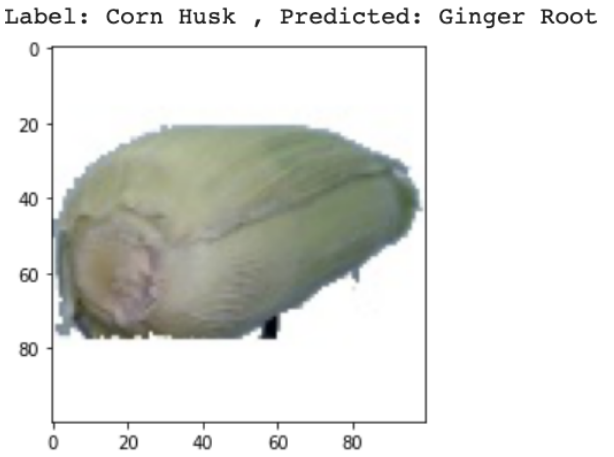
Result 1: Correct prediction	**True Label**: Apple Crimson Snow **Predicted Label:** Apple Crimson Snow	
Result 2: Wrong prediction	**True Label :** Corn Husk **Predicted Label** : Ginger Root	

Table 3: Prediction Result

FUTURE WORK

1. Although we experimented with a number of pretrained models, there was a performance, computation time trade off. So, now we are trying to find models that are faster and produce better results.

2. Once we finalize the model, we are planning to develop a flask, nginx application that can connect nutrition and image classification engines into one.

3. Currently we connect to a nutrition api to get nutritional information. But we are trying to develop an application that can count and classify different types of fruits in an image and create a nutritional value chart based on the aggregation of individual types.

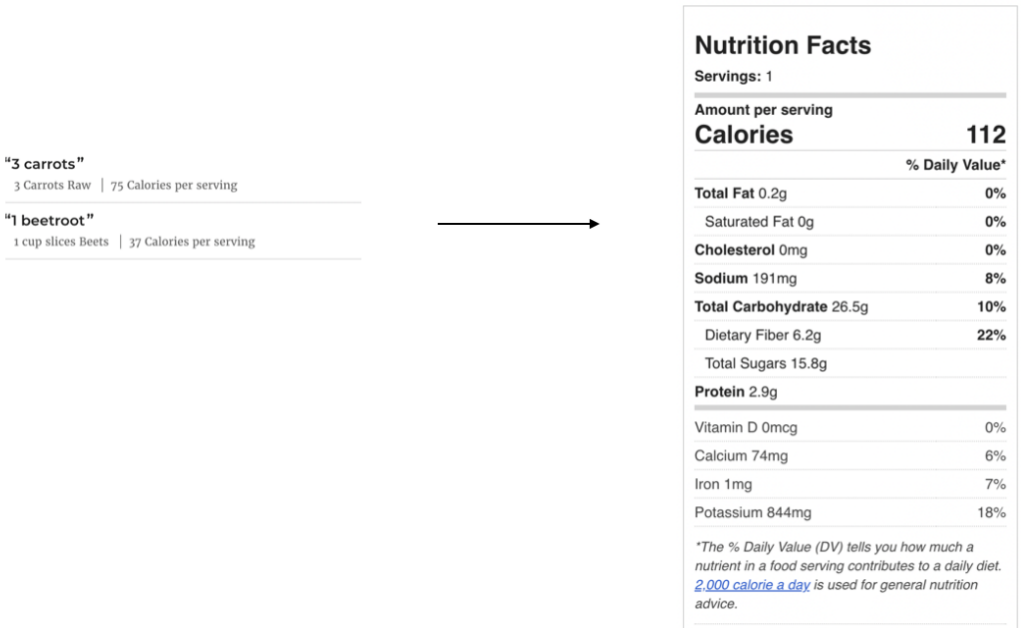


Image 7 : System that counts the number of each fruit/vegetable and aggregates the nutritional information

CONCLUSION

We started out with the project to work on a real life problem in order to apply the techniques learnt in the class as well as build an application that creates value along the way. For now the project is limited to the image classification of raw fruits and vegetables, but we wish to scale this to local cuisines and dishes as well. This is to increase the range of image classification that the model will do along with nutrition retrieval.

REFERENCE

[1] Giovanni Maria Farinella, Dario Allegra, Marco Moltisanti, Filippo Stanco, and Sebastiano Battiato. Retrieval and classification of food images. Computers in Biology and Medicine, 77:23 – 39, 2016.

[2] Keigo Kitamura, Toshihiko Yamasaki, and Kiyoharu Aizawa. Food log by analyzing food images. In Proceedings of the 16th ACM International Conference on Multimedia, MM '08, pages 999–1000, New York, NY, USA, 2008. ACM.

[3] Woo Chaw Sang and Seyed Hadi Mirisaee, "A New Method for Fruit Recognition System", MNCC Transaction on ICT, Vol1- No 1, June 2009.

[4] E. Umbarg, S. "computer vision and image processing", Prentice Hall Professional, Hall Professional Technical Reference, 1998.

[5] Jorge Sanchez, Florent Perronnin, Thomas Mensink, and Jakob Verbeek. Image Classification with the Fisher Vector: Theory and Practice. International Journal of Computer Vision, 105(3):222–245, December 2013.