Apple Fruit Detection and Maturity Status Classification

Deepika Srinivasan, Mahmoud Yousef

Abstract: ***Identifying the quality of fresh produce while procuring is a major task that involves time and human effort in the retail industry. The main objective of this project is to identify and classify whether the apple fruit is fresh or rotten using Convolutional Neural Networks. The outcome of our study resulted in 97.92 percent accuracy for the 2 classes of approximately 5031 images in the classification, by identifying apples using Resnet 50 and then classifying them using the proposed model.***

Keywords: Convolution Neural Network, Resnet50, Fresh Produce, Flask, and Tkinter.

I. INTRODUCTION

Retail and many supermarkets require manual labor to sort and classify fruits depending on the maturity level. This includes not just labor costs, but also the time spent on these operations. In this rapidly expanding sector, fruit identification is required followed by maturity level determination.

Until Convolution Neural Network was implemented, other methods were used for the assessment of maturity level. Wajid et al. used Border / Interior Pixel Classification (BIC) to extract features and Naïve Bayes, Artificial Neural Network, and Decision Tree-Comparison to classify oranges into ripe, unripe, and unscaled classes [1]. While the precision is about 93.13 percent, the number of images in the dataset is 335 which is considered small. Using color and scale as characteristics, Ronald and Evans used Naive Bayes to classify three varieties of apples and achieved 91 percent accuracy [2]. Tiger and Verma have proposed a two-layer feed-forward network to distinguish normal and infected apples using MATLAB [3], but they have not given accuracy measures nor network details. SVM is considered to be the best model for distinguishing healthy and infected apples, along with grade identification by Moallem, Serajoddin, and Pourghassem with a recognition rate of 92.5 percent and 82.5 percent respectively [4]. The maturity of fruits such as apples, melons, and bananas were measured using K Nearest Neighbors algorithm and Brix units by Iswari, Wella, and Ranny [5]. However, this analysis uses a small dataset.

Infected mango leaves are detected by Singh et al. using the Multilayer Convolutional layer [6]. While the accuracy is 97.13 percent, image preprocessing and several epochs are required in this analysis. Lu et al. suggested the six-layer Convolution Neural Network Model to identify nine different fruit types and obtained 91.44 percent accuracy [7]. It should be noted that only 900 images were used for training and testing the model, respectively. Khaing, Naung, and Htut [8] applied the Convolutional Neural Network model to identify 30 different types of fruit using a CNN model with 8 layers and 1000 epochs and obtained 94 percent accuracy, although the number of fruits involved in the training is remarkably small. Singla, Yuan, and Ebrahimi [9] used the GoogleNet pre-trained model to identify food and non-food images and have achieved 99.2 percent accuracy. Zhang et al. [10] proposed a 13-layer CNN model for a 3600-image dataset with 200 images in each class for the Fruit classification analysis and achieved 94.94 percent accuracy. However, the images used for the study came from a clean dataset, and there were no practical images. Hou et al. [11] developed a Convolutional Neural Network model based on fruit recognition by extracting the features using a selective search algorithm and then feeding them into the model—achieving 99.77 percent accuracy. The above methods obtained good results, but there are also shortcomings: some methods used a small dataset while others used complicated models and intensive training to obtain better results. To resolve these defects, a novel method of extraction and classification of fruit is suggested using fruit images and a convolutional neural network (CNN) with two layers is proposed in this study.

II. CONVOLUTIONAL NEURAL NETWORK

Images are made of large dimensions, and, thus, it is difficult to implement classical neural networks for both image recognition and classification problems due to the wide number of neurons in hidden layers and the required computational costs needed to train the network. Yann LeCun suggested Convolutional Neural Network (CNN) architecture to address these challenges. CNN can be used in a wide variety of applications such as image and video recognition, image detection, medical image interpretation, natural language processing, and financial time series.

A CNN is a deep learning algorithm capable of capturing an input image, assigning value to various objects in the image, and distinguishing between classes. The typical Structure of the CNN model is shown in Fig 1. This consists of three primary types of layers: convolution layers, sub-sampling or the pooling layers, and an output layer—most commonly a fully connected neural network.

Revised Manuscript Received on June 22, 2020.

Deepika Srinivasan, School of Computer Science and Mathematics, University of Central Missouri, Warrensburg, USA. E-mail: dxs30730@ucmo.edu

Dr. Mahmoud Yousef, School of Computer Science and Mathematics, University of Central Missouri, Warrensburg, USA. E-mail: yousef@ucmo.edu

The Convolutional Layer, along with the Pooling Layer, forms a layer of a CNN. Depending on the complexity of the images, the number of these layers can be increased much more to capture low-level data at the expense of more computing resources. Next, we must flatten the final output and then feed it to a neural network feed. This fully connected layer looks at the performance of the previous layer and decides which features are more associated with a given class.

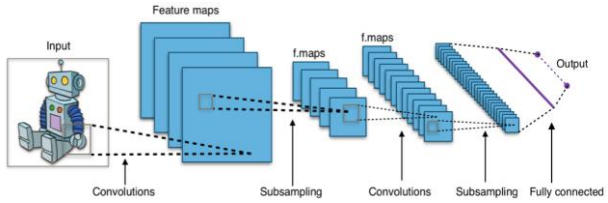

Fig 1: Typical CNN Architecture [13]

CNN's function is to reduce images to a type that is easier to process without missing vital functionality to get an accurate result. This is critical when developing an architecture that is not only good in terms of learning features but also is scalable to large datasets. Owing to the reduction of the number of parameters involved and the reusability of weights, CNN architecture performs better fitting to the image dataset. One distinctive characteristic of CNN is that several neurons will use the same filter. This decreases memory footprint since a single bias and a single weight vector are used for all receptive fields that share the filter.

III. USING CNN IN IMAGE RECOGNITION AND IMAGE CLASSIFICATION

A. Dataset

The initial step to create the model is to collect the data set for the image. In this research, we used Kaggle's Fresh and Rotten fruits Image dataset to train and test our model with a specific emphasis on apples [12]. Dataset consists of 2088 fresh apples while 2943 photos are rotted apples.

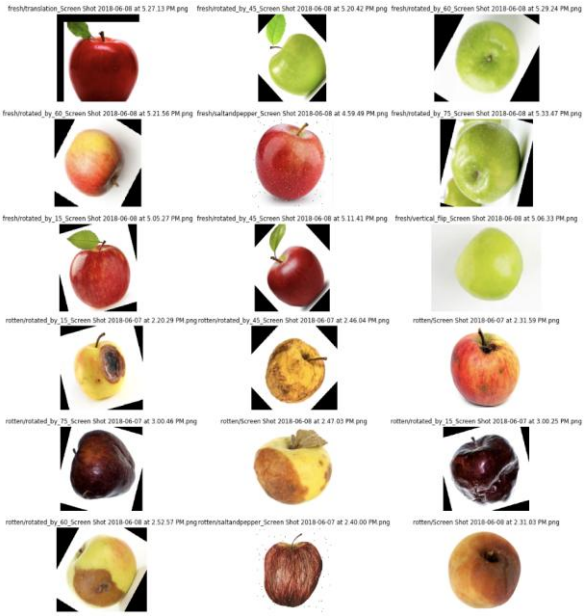

Fig 2: Fresh and Rotten Apple Images from Dataset

Dataset was separated into Training and Testing images based on a 75:25 split ratio. This was done by experimenting with various ratios and choosing the best performing ratio. To prevent overfitting issues, we have made use of Data Augmentation as shown in Fig. 3 below

Fig 3: Original and Data Augmented Images

B. Implementation of Apple Recognition using Transfer Learning

In this paper, we used Resnet50 to detect and extract the apple from the main image. ResNet50 is based on a convolutional neural network with 50 layers. We can load a pre-trained version of the trained network from the ImageNet database on more than a million images. The pre-trained network is capable of classifying pictures into 1000 types of objects, such as a keyboard, mouse, pencil, and several other classes. As a result, the network has acquired rich representations of features for a large variety of images. ImageAI provides classes and functions that are highly effective and simple to use to perform Image Object Detection and Extraction. State-of-the-art deep learning algorithms such as RetinaNet, YOLOv3, and TinyYOLOv3 can be used. ObjectDetection class helps in detecting 80 common everyday objects of different kinds on any image or collection of images using pre-trained models trained on the COCO dataset. ResNet uses skip connection to attach the output to a later layer from an earlier layer. The identity and convolution blocks are then combined with the architecture shown below in Fig. 4 to create a ResNet-50 model.

The ResNet-50 model consists of 5 stages—each with a convolution and identity component. Each block of convolution has 3 layers of convolution, and each block of identity always has 3 layers of convolution.

The ResNet-50 has over 23 million parameters that are trainable.

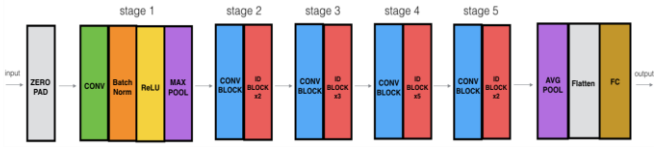

Fig 4: ResNet50 Architecture [14]

Our goal here was to extract only apple images from the original image that could consist of any object. The pre-trained ResNet model was applied for this purpose and the images retrieved were saved to a local directory.

C. Implementation of Apple Classification using CNN

CNN Model proposed for this study consists of 4 layers. Data Augmentation will scale the images to the size of 64x64x3.

The first convolutional layer is made of 32 filters of size 3x3. The first convolutional layer's output is fed into the second convolutional layer, which also consists of the same number of filters and filter size as the previous. Following the first and second convolutional layer is the sub-sampling layer making use of Maxpool with size 3x3. Finally, we have two layers that are fully connected to help classify fresh apples from rotten ones.

TABLE I. CNN Model

Layer	*Architecture*	*Output Size*	*Parameter Count*
Input	64*64*3		
Conv_1	3*3*32	62*62*32	896
Maxpool_1	3*3	20*20*32	
Conv_2	3*3*32	18*18*32	9248
Maxpool_2	3*3	6*6*32	
Fully_Connected_1	128		147584
Output	1		129

Batch size 32, epochs 10, Stride 1, Relu Activation, and Train and Test Ratio 75:25 were selected in our model based on performance comparison. For each layer, we initialized the values of the weights and biases randomly. The network is trained with back-propagation using Adam as an optimizer. The feature map of the two convolutional layers is shown as seen in Fig. 5 below.

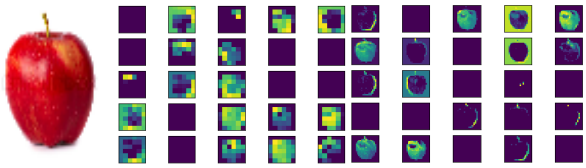

Fig 5: Feature Map Visualization

TABLE II. Table showing results with different Training and Test Split Ratio

Split Ratio	*Epochs*	*Training Accuracy*	*Testing Accuracy*
40:60	10	94.28%	93.81%
50:50	10	94.36%	96.05%
60:40	10	96.48%	95.68%
70:30	10	95.68%	96.07%
75:25	10	96.40%	97.59%
80:20	10	96.24%	94.56%

TABLE III. Table Showing Results With Different Batch Sizes

Batch size	*Training accuracy*	*Testing accuracy*
16	96.19%	96.74%
32	96.13%	97.12%
64	92.61%	92.36%

TABLE IV. Table Showing Results With Different Pooling Techniques

Batch size	*Pooling Layer*	*Training accuracy*	*Testing accuracy*
16	Maxpool	96.19%	96.74%
16	Avgeragepool	94.94%	96.34%
32	Maxpool	96.13%	97.12%
32	Averagepool	93.69%	95.83%
64	Maxpool	92.61%	92.36%
64	Averagepool	93.98%	91.35%

TABLE V. Table showing results with different maxpool size

Batch Size	*Maxpool Size*	*Training Accuracy*	*Testing Accuracy*
32	2*2	96.13%	97.12%
32	3*3	96.66%	97.76%
64	2*2	92.61%	92.36%
64	3*3	95.52%	93.24%

TABLE VI. Table Showing Results With Different Strides

Batch Size	*Stride*	*Training Accuracy*	*Testing Accuracy*
32	1	96.13%	97.12%
32	2	95.39%	92.95%
64	1	92.61%	92.36%
64	2	94.65%	96.26%

TABLE VII. Table Showing Results With Different Filter Count

Conv Layer and Filter Count	*Training Accuracy*	*Testing Accuracy*	*Parameter count*
Layer 1=32 Layer 2=32	96.40%	97.59%	720,545
Layer 1=32 Layer 2=16	95.44%	95.67%	407,185
Layer 1=32 Layer 2=8	95.13%	92.14%	204,169
Layer 1=16 Layer 2=32	95.49%	94.23%	808,161
Layer 1=16 Layer 2=16	94.88%	95.75%	404,433

TABLE VIII. Table Showing Results With Different Filter Size

Filter size	*Epochs*	*Training Accuracy*	*Testing Accuracy*
3*3	10	96.40%	97.59%
5*5	10	93.14%	95.03%
7*7	10	93.14%	94.07%

TABLE IX. Table showing results with different number of neurons in dense layer

Neuron count	*Training Accuracy*	*Testing Accuracy*	*Parameter Count*
64	95.95%	96.15%	411,681
128	95.92%	96.39%	813,217
256	93.82%	93.67%	1,616,289

TABLE X. Table Showing Results With Different Activation

Activation	*Training Accuracy*	*Testing Accuracy*
Sigmoid	58.76%	58.41%
Tanh	94.17%	93.43%
Relu	95.92%	96.39%

TABLE XI. Table showing parameter count reduction with selected hyperparameters

Activation	*Maxpool size*	*Training Accuracy*	*Testing Accuracy*	*Parameter count*
Relu	2*2	94.70%	96.07%	813,217
Relu	3*3	96.37%	97.92 %	157,857

D. Methodology

Step 1: Imported packages such as Tensorflow, Keras, Numpy, Scipy, Matplotlib, Imageai, Image, Flask, and Tkinter.
Step 2: Experiment-based model parameters and hyper-parameters are initialized.
Step 3: The generator of image data is used to retrieve the images from the Dataset.
Step 4: The number of classes in the image generated array has been extracted and verified.
Step 5: 75:25 Train Test Split ratio has been selected and as a result, 3773 images are taken as the train data, and 1258 images are taken as the test data.
Step 6: Sequential Model is developed. Training is repeated with 10 Epochs and tested against the test data. Test accuracy and loss are calculated at each step which will result in the final CNN model.
Step 7: Run the python program with the test image to detect apples along with accepted probability.
Step 8: The uploaded image is detected for apples based on the user-defined probability, and the extracted images are saved.
Step 9: For each extracted apple image saved locally, CNN Model is called using the predict function, and the classification is displayed based on the trained model.

E. Results and Analysis

In our experiment, we used a test image of various varieties of fruit, which, in effect, would be used for object detection with a focus on apples alone. The probability of which apples to be identified will be given by the user. Next, the extracted apple images are used as input for our CNN model to be labeled as ‘fresh’ or ‘rotten’.

For example, in the sample input image given below (Fig. 6a), we can see that there is only one apple present in the whole picture so that apple is identified (Fig. 6b) and extracted (Fig. 6c). The CNN model created earlier would select the extracted images and label them as ‘fresh’ or ‘rotten’ fruit.

(a) (b) (c)

Fig 6: Apple Detected and Extracted

F. Graphical User Interface

Tkinter is Python's Standard Interface library for developing Graphical user interface-based applications. Tkinter offers a powerful object-oriented interface to the Tk GUI toolkit. To use this application, the user must first upload the apple to be labeled by clicking on 'Upload an image', as shown in Fig. 7. Eventually, the user must click on 'Classify apple' to view the prediction results.

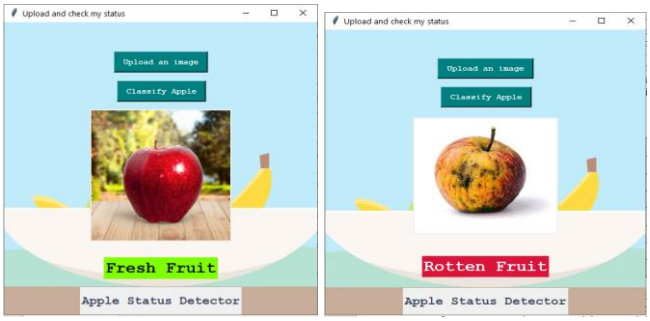

Fig 7: GUI Application

G. Web Application

Flask is a Python micro application system developed with a small core and easy-to-extend philosophy. It is designed to make it fast and easy to use with the potential to scale up to more complicated applications.

Users would have to enter the Image URL in our web application which consists of the location of the test image and click on 'Predict.' This, in turn, will trigger the model, producing a prediction as seen in Fig. 8

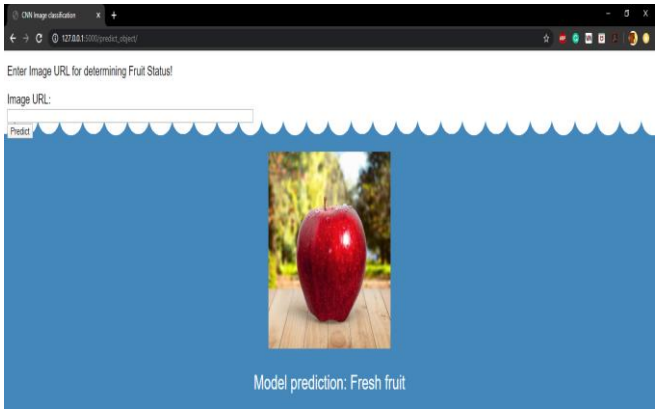

Fig 8: Flask based web application

IV. CONCLUSION

In this article, apple identification accompanied by classification is carried out using a convolutional neural network (CNN). In our study, we used Resnet50 to help identify apples and extract them. Apple classification model based on CNN was developed to classify extracted apple images. The model was compared with different hyperparameters and achieved 97.92 percent classification accuracy along with parameter count reduction. Finally, for our project, we proposed a Graphical User Interface and Web-based application. In fact, we can also apply this concept to other fruit or vegetables by training the model accordingly. The cloud-based use of the model could also be part of the future study so that many people can access the model at the same time in a distributed environment.

REFERENCES

1. A. Wajid, N. K. Singh, P. Junjun and M. A. Mughal, “Recognition of ripe, unripe and scaled condition of orange citrus based on decision tree classification,” 2018 International Conference on Computing, Mathematics and Engineering Technologies (iCoMET), Sukkur, 2018, pp. 1-4.

2. M. Ronald and M. Evans, “Classification of Selected Apple Fruit Varieties Using Naïve Bayes,” Indian Journal of Computer Science and Engineering (IJCSE), vol. 7, no. 1, pp. 13–19, 2016.
3. B. Tiger and T. Verma, "Identification and classification of normal and infected apples using neural network," International Journal of Sciences and Research, vol. 2, Issue. 6, pp. 160-163, 2013.
4. P. Moallem, A. Serajoddin, and H. Pourghassem, “Computer vision-based apple grading for golden delicious apples based on surface features,” Information Processing in Agriculture, vol. 4, pp. 33–40, 2017.
5. N. M. S Iswari, Wella and Ranny, "Fruitylicious: Mobile application for fruit ripeness determination based on fruit image," 2017 10th International Conference on Human System Interactions (HSI), Ulsan, 2017, pp. 183-187.
6. U. P. Singh, S. S. Chouhan, S. Jain, and S. Jain, "Multilayer Convolution Neural Network for the Classification of Mango Leaves Infected by Anthracnose Disease," in IEEE Access, vol. 7, pp. 43721-43729, 2019.
7. S. Lu, Z. Lu, S. Aok and L. Graham, "Fruit Classification Based on Six Layer Convolutional Neural Network," 2018 IEEE 23rd International Conference on Digital Signal Processing (DSP), Shanghai, China, 2018, pp. 1-5.
8. Z. M. Khaing, Y. Naung and P. H. Htut, "Development of control system for fruit classification based on convolutional neural network," 2018 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus), Moscow, 2018, pp. 1805-1807.
9. A. Singla, L. Yuan, and T. Ebrahimi, T, “Food/Non-food Image Classification and Food Categorization using Pre-Trained GoogLeNet Model,” Proceedings of the 2nd International Workshop on Multimedia Assisted Dietary Management - MADiMa, 2016, pp. 3-11.
10. Y. Zhang, Z. Dong, X. Chen, W. Jia, S. Du, K. Muhammad, S. Wang, “Image based fruit category classification by 13-layer deep convolutional neural network and data augmentation”, Multimed Tools Appl 78, 3613–3632, 2019.
11. L. Hou, Q. Wu, Q. Sun, H. Yang and P. Li, "Fruit recognition based on convolution neural network," 2016 12th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD), Changsha, 2016, pp. 18-22.
12. Kaggle.com. 2018. Fruits Fresh and Rotten For Classification. [online] Available at:<https://www.kaggle.com/sriramr/fruits-fresh-and-rotten-for-classification> [Accessed 1 February 2020].
13. Elitedatascience.Com Keras Tutorial: The Ultimate Beginner’s Guide to Deep Learning in Python. [online] Available at:<https://elitedatascience.com/keras-tutorial-deep-learning-in-python> [Accessed 19th July 2020]
14. Towardsdatascience.Com. 2019.Understanding and Coding a ResNet in Keras. [online] Available at:<https://towardsdatascience.com/understanding-and-coding-a-resnet-in-keras-446d7ff84d33> [Accessed 19th July 2020]

AUTHORS PROFILE

Deepika Srinivasan completed her Master’s in Computer Science from the University of Central Missouri, Warrensburg, USA. Her research interests are Machine Learning and Deep Learning.

Dr. Mahmoud Yousef is a Professor in the School of Computer Science and Mathematics, University of Central Missouri, Warrensburg, USA. He is a member of the Association for Computing Machinery (ACM), a member of the American Mathematical Society (AMS), a member of the Steering Committee of Consortium for Computing Sciences in Colleges-Central Plains (CCSC-CP), and the managing editor of the Missouri Journal of Mathematical Sciences. His research area includes Computer Simulation and Modeling, Data Mining, and Computer Science Education.