EECS 504: Introduction to Computer Vision Team Fruit Ninja
Brendan Adkins (brenadki), Andrew Soong (asoong), Aditya Sundar (adisun), and Hansen Qin (qinh)
Code: https://drive.google.com/drive/folders/1BR6qUUFYLQrzw8tSCPzbB-0jMFYpchIg?usp=sharing

1. Executive Overview

Computer vision continues to open new opportunities for automation in quality control in mass production. This automation can decrease cost and speed up production. This project targets food industries with a vision system that can determine whether a food product is fresh or rotten. Two methods were evaluated and developed. The first of which is blob detection paired with a multi-layered perceptron. The second is image segmentation paired with a convolutional neural network. The chosen foods are apples, bananas, and oranges from a data set found on Kaggle[1].

2. Background and Impact

This system is primarily intended for food processing purposes. A reliable computer vision system that can detect whether a food product has gone bad can perform aspects of quality control in production and packaging. Increased automation can yield lower production costs and longer hours of operation[2]. Additionally, this system may be beneficial for personal use, particularly if the user is vision-impaired and is unable to see the state of freshness of their own food[3].

The project yields two methods for automating the food product quality assurance process, or general food freshness classifiers. It has been shown here that the various methods have different accuracies for different fruits. Neither method on its own performed as well on the oranges because of the differences in ways in which oranges can appear rotten. We recommend running both methods on objects like oranges, which are prone to both mold and bruises, and throw away the orange if either method shows it to be rotten. This will minimize false positives.

The difference in performance between the two methods for different fruits shows that, in order for this product to be expanded to more foods, many methods must be considered and tested for the food product in question. This speaks to the complexity of challenges in computer vision in general, in that the best approach to one problem may not be the best approach for another problem, even though its premise is the same.

3. Methods

We formulated two different methods in this project to classify the images. The first method is a combination of blob detection with a multi-layered perceptron (MLP), and the second method is image segmentation combined with a convolutional neural network (CNN). Each method has its own pros and cons and they will be discussed in the results section.

Fresh and rotten variations of apples, bananas, and oranges were used to train and test the project prototype. The dataset has 2929 images in total, divided into a 2387 image training set and a 542 image test set. The training set has 369 fresh apples, 249 fresh bananas, 324 fresh oranges, 513 rotten apples, 481 rotten bananas, and 351 rotten oranges. The test set has 95 fresh apples, 87 fresh bananas, 88 fresh oranges, 141 rotten apples, 131 rotten bananas, and 93 rotten oranges.

4. Prototype

It was observed that many rotten fruits had similar characteristics. Dark bruises - which can be easily detected by method 1 - appear on most rotten apples and bananas, as well as some oranges. A white mold - better suited to method 2 - also can be found on the oranges, which presents a challenge in that it appears different from the other ways in which fruit can go bad. This difference in appearance is highlighted in Figure 1. Furthermore, it can be seen in Figure 1 that the size and background for the different fruits are not consistent across the images. Depending on the method used, the approach to handling this situation is different.

Figure 1: Difference in appearance of various rotten fruits

4.1. Blob detection with MLP

In this method, we aim to detect whether a fruit is rotten based on the RGB values of the blobs detected. The motivation behind this method is that most bolds would appear on fruits as dark, circular blobs. Therefore, using a blob detection method can effectively select the pixels that are within the rotten areas and only use those pixels to train a classifier. The overall goal for this method is to use blob detection as a preprocessing stage in order to obtain feature vectors that are much lower in dimension than the original image. This reduces the complexity of the classifier and prevents undesirable effects from high dimensional inputs such as curse of dimensionality or overfitting.

The first step of this algorithm is to detect the blobs from the images. We used the difference of Gaussian (DoG) function from the *skimage* library, and set the minimum radius of blobs to be 4 pixels to prevent detecting small fluctuations in the image. The size of the original image does not matter as DoG is scale invariant. The following figure shows some of the detected blobs in the fruits.

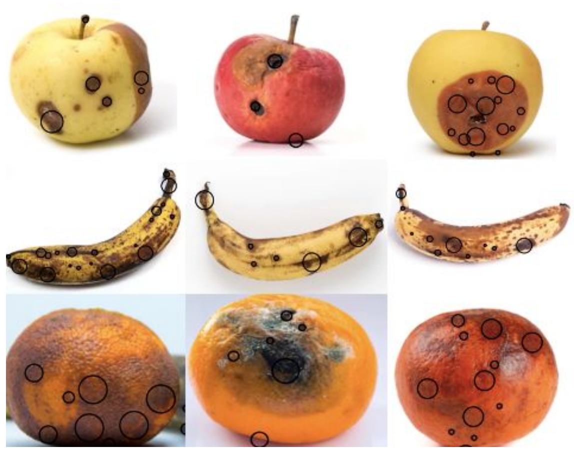


Figure 2. The blob detection algorithm on three types of fruits. The DoG function returns the x and y coordinates of the blobs. The RGB values of the blobs are then extracted and stored.

Next, we used a ranking system to rank the detected blobs from the darkest to the lightest and keep the four darkest blobs. We used the two-norm on the RGB vector to calculate its magnitude, and the lower the magnitude, the darker the blobs, and vice versa for brighter blobs. The reasoning for ranking the blobs is that false detections still exist in the blob detection step. So, by ranking the blobs, we further filter the false detections (which have RGB value brighter than the actual molds) and only keep the RGB values of the actual mold. Next, the average of the RGB values of the four blobs is taken to further simplify the input data, thereby yielding a 3 by 1 RGB vector as the feature vector for each image, as shown in Figure 3. Lastly the feature vectors are fed into a multilayer perceptron with three hidden layers of 100 neurons each. The MLP then outputs a binary result on whether the fruit was rotten or not.

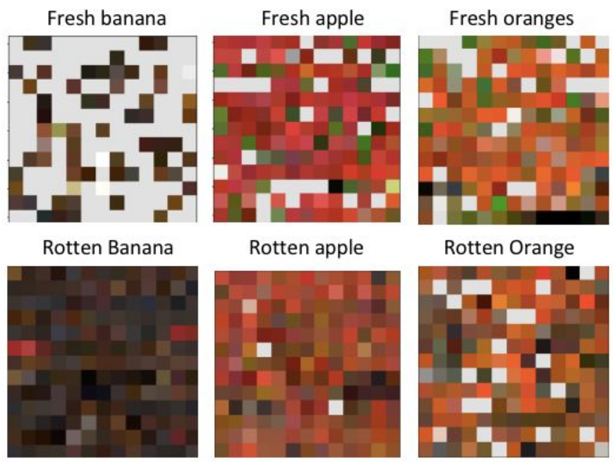
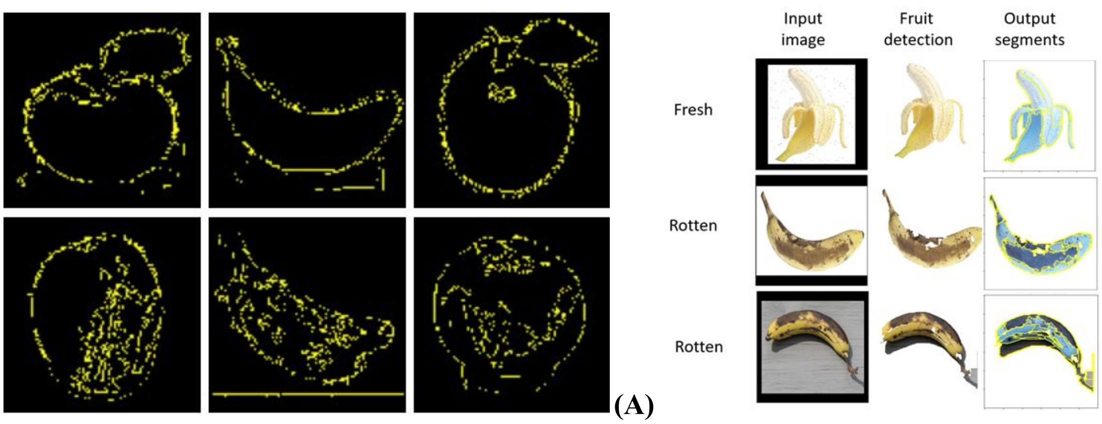


Figure 3. Feature vectors for every fruit. Each pixel represents a training image

3.2 Image Segmentation with CNN

In this method, we use a Convolutional Neural Network with pre-processed images. The pre-processing consists of segmenting each image and remarking the segmentation maps onto a black background. The input images to the CNN need to be the same size so the segmentation maps are padded and resized into 100x100 squares. The goal for this method is to remove color and solely use the shapes of the fruits and the mold on the fruit for classification.



(A) **(B)**

Figure 4. (A) Segmented and resized fruit images. From left to right: apples, bananas, and oranges. The first row is fresh and the second row is rotten. **(B)** Left: input image. Middle: detection and classification of fruit based on pixel value similarity in the input image, Right: felzenszwalb segmentation for the detected fruit.

Segmentation of the images uses the *felzenszwalb* function from the *skimage.segmentation* library. The parameters used were *scale=1000, sigma=0.5, min_size=100*. These parameters were chosen in order for the segmentation to create sizable segments that capture the fruit and large patches of mold, instead of minute differences in surface color from different levels of ripeness. These segmentation maps are then re-marked onto black images and padded with black edges to be square in size. Finally, each image is resized to be 100x100 pixels. A few of these images are in Figure 4 (A).

The next step is to feed these images into a CNN built using the `PyTorch` framework. The CNN consists of four convolutional layers and three fully connected layers and was trained using a batch size of 25 and learning rate of 0.001. The four convolutional layers have 32, 64, 128, and 256 kernels, all size

3x3. The three fully connected layers have 128, 64, and 2 nodes. By visual inspection of the segmentation maps, the edges of the fruits are smooth while the edges of the mold are fuzzy. The goal is for the CNN to learn these features automatically. While training and testing the CNN, it was discovered that the CNN performs better if a separate net was used for apples and oranges and another for bananas. We believe that this is due to the shape differences between apples/oranges and bananas.

Originally, we planned to implement an algorithm to detect, classify and segment the fruits without a neural network. The steps were (i) determining the fruit type by sampling the colour distribution of the pixels, (ii) Felzenszwalb segmentation, (iii) changing the disparate segments to white ([255,255,255]) and separating the fruit from the background, and (iv) segmenting the image from step (iii) to identify defects. Sample results are shown in Figure 4 (B). However, the method was found to be ineffective when the contrast between the fruit and background is lower than a threshold.

5. Results

Both methods yielded acceptable results. In the blob detection method, 120 random testing images were used(40) of each fruit. The MLP was able to yield a 94.3% accuracy for apples, 96.1% accuracy for bananas, and 82.4% on oranges. The lower accuracy of oranges is possibly due to the orange molds having non-blob shapes and a lighter color, so the same parameters used for apples and oranges does not work ideally on oranges.

The segmentation method yielded lower accuracies than the blob detection method. After training for 22 epochs, the net used to classify apples and oranges achieved 77.7% accuracy on testing of 417 fresh and rotten apples/oranges. After training for 30 epochs, the net used to classify bananas achieved 81.6% accuracy on testing of 218 fresh and rotten bananas. It might be reasoned that a larger CNN would yield better results, but we believe that the current CNN performs worse than the blob detection method because it is trained without color. As humans, we know color plays a very important role in classifying fresh versus rotten fruit. Furthermore, the segmentation map may contain unwanted artifacts and/or lose fidelity after resizing. Both of which can confuse the CNN. Examples can be seen in Figure 5.

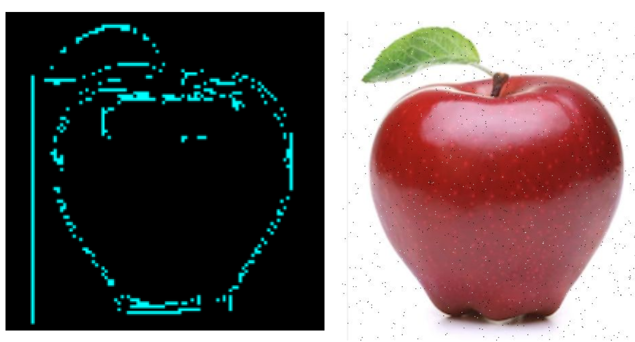


Figure 5. Fresh apple that was incorrectly classified as rotten. The segmentation map created for this apple was not perfect, as seen by the vertical line.

References

1. https://www.kaggle.com/sriramr/fruits-fresh-and-rotten-for-classification
2. https://www.brookings.edu/blog/future-development/2020/12/11/what-is-the-future-of-work-in-agri-food/
3. Kostyra E, Żakowska-Biemans S, Śniegocka K, Piotrowska A. Food shopping, sensory determinants of food choice and meal preparation by visually impaired people. Obstacles and expectations in daily food experiences. Appetite. 2017 Jun 1;113:14-22. doi: 10.1016/j.appet.2017.02.008. Epub 2017 Feb 7. PMID: 28188864.