A Neural Network Approach to Classifying Banana Ripeness

Kyle Demeule
Department of Computing Science
Faculty of Applied Sciences
Simon Fraser University
8888 University Drive
kdd2@sfu.ca

Bernard S Chan
Department of Computing Science
Faculty of Applied Sciences
Simon Fraser University
8888 University Drive
bernardc@sfu.ca

Saeed Soltani
Department of Computing Science
Faculty of Applied Sciences
Simon Fraser University
8888 University Drive
saeeds@sfu.ca

Abstract

Automatic fruit ripeness detection through machine learning techniques is an active and widely applicable research topic. In this paper, a new technique to detect banana ripeness and non-banana objects is introduced. First, an appropriate data set on bananas was generated under controlled conditions. Instead of histogram or segmentation based methods, we use AlexNet, a convolution neural network model supported by Caffe, to extract features. Through the extracted features, support vector classifiers with various kernels, such as linear, radial basis functions, sigmoid and polynomial, were used to classify the different objects. Features extracted from different exit points of AlexNet were also considered in the experiment. Overall, the combination of features extracted from FC6 and support vector classifier with radial basis function kernel provided the best results. In strictly classifying the ripeness of bananas, this combination has 90.2% accuracy in testing.

1 Introduction

Traditionally, humans detect ripeness of fruit through sight, odour, taste and touch. While people and animals are naturally equipped with these senses, machines are not, so automating fruit ripeness detection is a difficult task. Given that odour sensors and image processing are more reliable and developed than taste and touch, machine learning research in detecting fruit ripeness have been based around odour and sight. Through different type of odour sensors, Llobet et al. [6] and Li et al. [5] collected smell information on ripening bananas and apples. Then, they applied various supervised classifier to classify their states.

Instead of odour, we are interested in integrating imaging and deep learning techniques to classify the ripeness of bananas. Based on reviews by Dadwal and Banga [1] and Kodagali and Balaji [3], typical computer vision based ripeness detection methods are based on histogram matching or image segmentation. For example, Paulraj et al. [7] proposed a histogram-based neural network classifier for evaluating ripeness banana. Each image in the data set is decomposed into its RGB component and the number of pixels in each channel of varying intensities were counted. This

information is then vectorized and fed into a neural network to be classified as unripe, ripe, or overripe. Segmentation based methods build on top of histogram-based methods. As mentioned in Dadwal and Banga [1], images are preprocessed so that the fruits in the picture are separated from the rest of the image. Then, classifying techniques, such as histogram-based neural network or clustering, are applied to classifying the ripeness. Segmentation based techniques are theoretically superior because features used for classification are only extracted from the relevant segmented region.

With the works we have reviewed, the models only classify ripeness of fruits. In application, it is likely that there are other objects aside from the fruits on interest. Therefore, our first goal for this project is to implement a new method that could detect the ripeness of bananas. Secondly, our model will differentiate between banana and non-banana objects. Because the data sets from Paulraj et al. [7] is not available, our initial task is to create an appropriate data set. Then, instead of using histogram or segmentation based methods, we will use convolutional neural network (CNN) to extract features. With the features of each image at hand, we construct support vector machine (SVM) classification model to classify ripeness and whether the object in question is a banana.

The remainder of this paper is organized as follows. In Section 2, we detail the steps in data generation, features extraction and label classification. The results from our experiment are discussed in Section 3 and we conclude the paper with a discussion on our results and future work in Section 4.

2 Methodology

For this project, we created our own data set on banana and non-banana objects because the previous data sets from Saad et al. [9] and Paulraj et al. [7] were not available. After establishing our own data set, we extracted the features of the images using a pre-trained convolution neural network. The Caffe deep learning framework by Jia et al. [2] allowed us to access to many existing models. Given that we have a visualization task with different types of objects, we chose AlexNet by Krizhevsky et al. [4] to extract the features. After obtaining the features of the images, we used the SVM library provided in SciKit Learn [8] to classify the objects. A workflow of this project is shown as a flowchart in Figure 1. In this section, we will discuss the details in each step of our work.

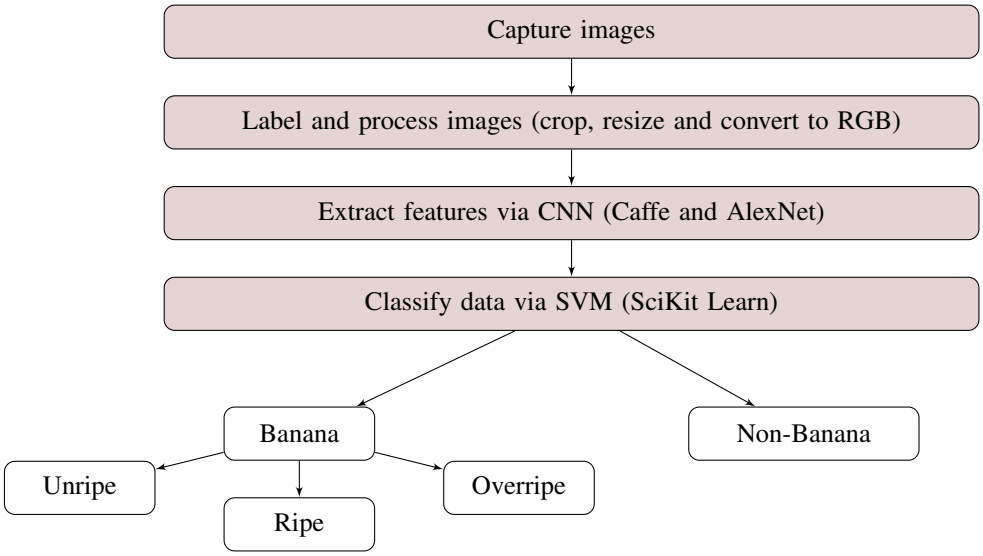

Figure 1: A flow chart of project development.

2.1 Generating and Preparing the Data set

Since the data sets used by Saad et al. [9] and Paulraj et al. [7] were not publicly available, we decided to create on our own data set. Controlling for lighting, background and camera (Canon S90), we took pictures of banana and various non-banana objects. After the pictures were taken,

we incorporated each picture at $0\,^\circ$, $90\,^\circ$, $180\,^\circ$ and $270\,^\circ$ of rotation to increase the number of pictures in the data set by four fold. Also, there were equal number of pictures for each of the four labels: unripe banana, ripe banana, overripe banana and non-banana. For the banana data set, twelve unique bananas were used. We used apples, tomatoes, lemons, limes, mushrooms, broccolis, potatoes, pears and green peppers as non-banana objects. In total, there were 928 images generated for the data set and sample pictures of this data set are shown in Figure 2. After obtaining the data set, we used various Python scripts to standardize the images so that each one is resized and cropped to 256×256 pixels, and applied the class labels manually. Furthermore, each picture is decomposed into the RGB channels for features extraction.

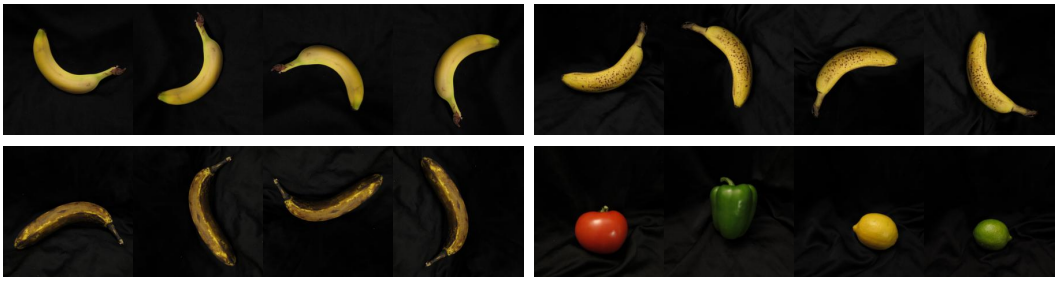

Figure 2: Upper left: unripe bananas. Upper right: ripe bananas. Lower left: overripe bananas. Lower Right: non-banana objects.

2.2 Features Extraction via AlexNet

To better classify the images, a CNN was used to extract features so that other classification methods could later be used. CNNs are neural networks in which each neuron in a layer is connected to a small, adjacent portion of the previous layer. These adjacent portions overlap, so each neuron on the previous layer is connected to multiple neurons on the next, but significantly less than if the layers were fully connected. AlexNet by Krizhevsky et al. [4] is a CNN that was developed for image classification. It was trained on 1.2 million high resolution images from a subset of the images from the LSVRC-2010 and LSRVC-2012 competitions. It achieved a score that outscored the previous LSVRC-2010 champion. Figure 3 shows the structure of AlexNet. This CNN has five convolutional layers and three fully connected layers. Several of the convolutional layers are connected through max-pooling layers, where a layer is segmented and the max value for each segment is taken. We chose to use AlexNet because it was designed for image classification and had very high accuracy in the LSVRC-2010 challenge.

Instead of building and training our own model we decided to look at existing solutions. Caffe is a deep learning framework developed by Jia et al. [2] to aid in the training and use of neural networks. It has libraries that allow it to be used by a variety of languages, including Python, and has a "model zoo" that allows pre-trained models to be downloaded and used for feature extraction or as the basis for new networks. For this work, we used an existing model in Caffe's "model zoo" that is a pre-trained model based on the original AlexNet.

Using this pre-trained CNN, we input all the images from our dataset and extracted the internal representation of the data from the last three layers: FC6, FC7 and FC8. Layers FC6 and FC7 have 4096 neurons each, so their output is a vector of length 4096. FC8 is similar but has 1000 neurons.

2.3 Classification

The format of the features extracted from an image using the CNN is a vector of length 4096 or 1000. Since each object is now a vector, we can use a traditional classification approach, for example support vector machines (SVM). C-Support vector classification (C-SVC) [**?** **?**] is a form of SVM that includes a penalty on the error term. In this model, the C parameter is added to prevent misclassification in the training set. A large value for C will result in a smaller-margin hyperplane and eventually better classification. On the other hand, a small value of C could lead to misclassified sets, even if the training data is linearly separable. This algorithm also allows the use of kernel functions.

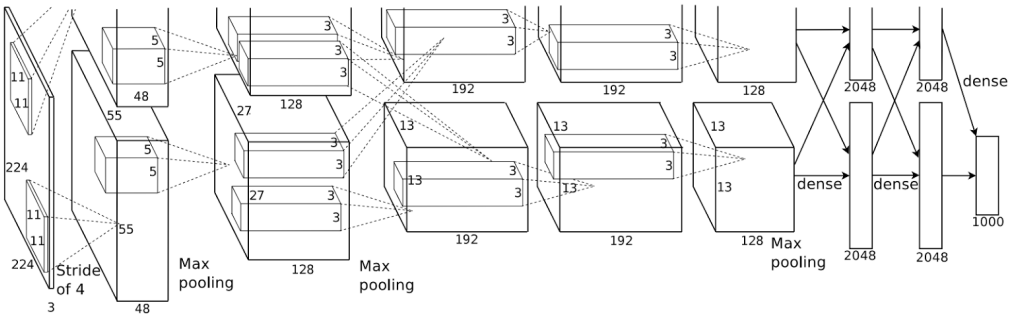

Figure 3: Architechture of AlexNet.

Given our previous experience with python, and the previous use of python in processing and extracting features from the images, we wanted to continue to use python for classification. We chose to use scikit-learn [8], a mature, open source library that implements multiple machine learning algorithms including classification, regression, and clustering. It has a *sklearn.svm.SVC* library for C-support vector classification, an offers multiple kernel methods, as well as a grid search package for optimizing parameters.

There are several ways to configure the C-SVC models. The classification can be changed by specifying the layer of the CNN we use as input (FC6, FC7, FC8), what kernel method we choose for the SVM (linear, RBF, sigmoid, polynomial), and the value of the parameters (γ, C, and degree for polynomial kernels). To test these options we used the grid search functionality of the scikit-learn package to exhaustively determine the best parameters for each kernel/layer combination. For each layer, for each kernel method, we use 10-fold cross validation on the training set to learn the best parameters for the SVM. Then with the best parameters we compared the results for each layer/kernel combination on the training data. From those results we can pick which layer, kernel, and parameter combination we think will have the best result on the test set.

A training and testing split of 2/3 and 1/3 was established at the beginning of experimentation. Each original image was rotated four ways, but all four images were either put in the training or testing set, they were never split into the two groups. The same training and testing split was used for all experiments.

3 Results

As mentioned in Section 2.3, we applied existing SVM algorithms from SciKit Learn's library to classify the data in our set. For the experiment, we compared the performance of different kernels and feature extraction points from AlexNet. As highlighted in red in Table 1, using features extracted from RC6 and RC7 along with the radial basis function kernel both provided 100% accuracy in training. Testing results were consistent with our training results and the $\{$FC6, RBF$\}$ pair outperformed all other classifiers at 87.8% accuracy. Therefore, the remainder of this report focuses on $\{$FC6, RBF$\}$ as the chosen classifier.

Table 1: Overall accuracy of correctly classified objects from training and testing of SVM models with various kernels. Features were obtained from FC6, FC7 and FC8 exits of AlexNet. (Lin = linear, RBF = radial basis function, Sig = sigmoid, Poly = polynomial)

	Training				Testing			
	Lin	RBF	Sig	Poly	Lin	RBF	Sig	Poly
FC6	0.942	1.000	0.266	0.911	0.821	0.878	0.218	0.814
FC7	0.876	1.000	0.266	0.872	0.788	0.862	0.218	0.804
FC8	0.768	0.998	0.266	0.807	0.676	0.843	0.278	0.696

To better understand the performance of the {FC6, RBF} classifier, we present the results on specific labels in Tables 2 and 3. As seen in Table 2, there was no confusion between unripe and overripe bananas. All errors in classifying banana centered around ripe bananas. As for the performance of classifying bananas versus other objects, Table 3 shows that there are more objects that are misidentified than bananas misidentified as other objects.

Table 2: Confusion matrix on banana ripeness with the {FC6, RBF} classifier.

		Predicted		
		Unripe	Ripe	Overripe
Actual	Unripe	0.913	0.095	0.000
	Ripe	0.068	0.836	0.096
	Overripe	0.000	0.033	0.967

Table 3: Confusion matrix on banana versus other objects with the {FC6, RBF} classifier.

		Predicted	
		Banana	Other
Actual	Banana	0.953	0.046
	Other	0.066	0.934

Table 2 showed that we were able to distinguish between unripe and overripe bananas successfully. Though, the system had difficulty in distinguishing ripe bananas from the other two classes. Overall, our accuracy on classifying banana ripeness was 90.2% and this was an improvement on the work by Saad et al. [9] over a much larger data set.

To better understand the incorrectly classified objects from {FC6, RBF}, images of correctly and incorrectly classified objects are shown in Figure 4. Given the size of the data set, it is impractical to present the entire list of correctly or incorrectly identified objects for each class label in the figure. While we have only shown a selected few of the incorrectly classified objects, they are generally representative of the objects that were misclassified.

From the first two rows of Figure 4, we see that unripe and ripe bananas are often misclassified as each other. This observation correspond with the results shown in Table 2. Furthermore, from the fourth row of Figure 4, we see that the objects that are misclassified as non-bananas are mostly overripe bananas. This information cannot be directly obtained from Table 3.

4 Discussion

Our method outperformed the performance of Saad et al. [9] in classifying banana ripeness with significantly larger data set. In addition to improving the accuracy of classification over previous works, we also added detection of non-banana objects as an extra feature in this project.

There are several future directions for this project. Since we used a pre-trained model to extract features from our data set, we can improve the feature extraction process by training the model and fine tuning the parameters of the model. Features extracted from a refined model should provide more representative feature sets that would improve the classification results.

Aside from simply detecting ripeness of bananas, the core idea of this algorithm could be expanded for ripeness detection for other fruits and vegetables. This project can also be expanded into other environment. We aim to implement this as an application for mobile operating systems. A mobile app that could detect fruit ripeness through computer vision is an useful tool for individuals who have impaired senses. From an industrial point of view, this program could be essential in automatic large scale sorting.

The data set used in this project as well as the accompanying code are available at bit.ly/BananaRipe.

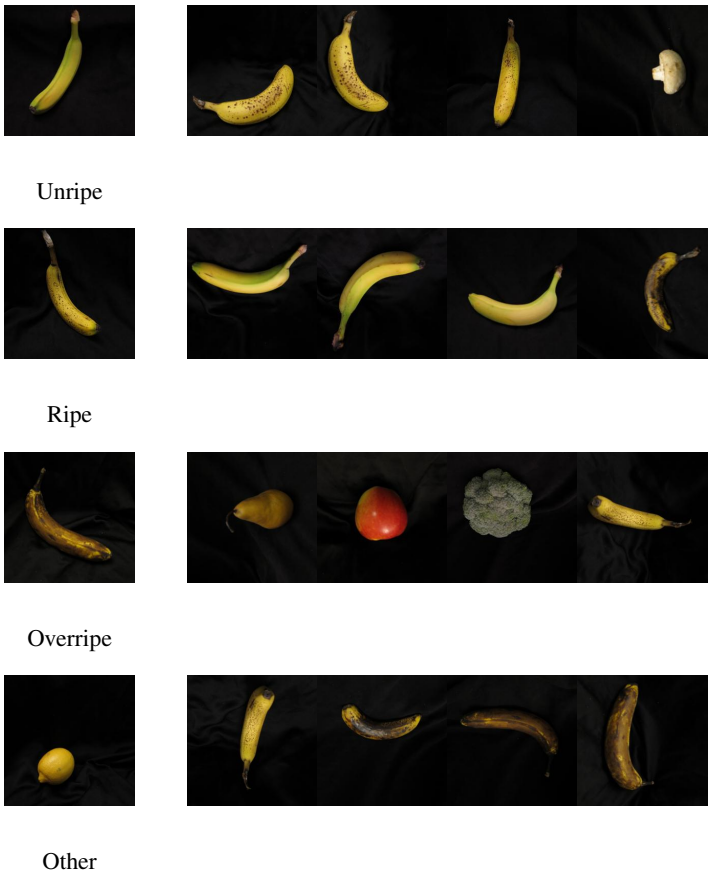

Figure 4: Samples of correctly classified data from each class are presented in the first column. For each row, the remaining columns contain samples of incorrectly classified objects that have been given the same label as the object in the first column.

Acknowledgments

We thank our TA Zhiwei (Lucas) Deng and for his guidance on using Caffe as well as AlexNet. We also thank Dr. Mori for his insights on solving this problem.

Contributions

Contributions were as follows. Kyle Demeule: Creation and processing of data set, feature extraction with Caffe, SVM implementation and optimization, editing of report. Bernard Chan: SVM implementation, literature review, poster creation, and majority of report. Saeed Soltani: SVM optimization, PCA research, literature review, and report contributions.

References

[1] Meenu Dadwal and VK Banga. Color image segmentation for fruit ripeness detection: A review. In *2nd International Conference on Electrical, Electronics and Civil Engineering (ICEECE'2012) Singapore April*, pages 28–29, 2012.

[2] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

[3] Jyoti A Kodagali and S Balaji. Computer vision and image analysis based techniques for automatic characterization of fruits–a review. *International Journal of Computer Applications*, 50 (6), 2012.

[4] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[5] Changying Li, Paul Heinemann, and Richard Sherry. Neural network and bayesian network fusion models to fuse electronic nose and surface acoustic wave sensor data for apple defect detection. *Sensors and Actuators B: Chemical*, 125(1):301–310, 2007.

[6] Eduard Llobet, Evor L Hines, Julian W Gardner, and Stefano Franco. Non-destructive banana ripeness determination using a neural network-based electronic nose. *Measurement Science and Technology*, 10(6):538, 1999.

[7] Murugesapandian Paulraj, Chengalvarayan Radhakrishnamurthy Hema, Krishnan R Pranesh, and Mohd Radzi Siti Sofiah. Color recognition algorithm using a neural network model in determining the ripeness of a banana. In *Proceedings of the International Conference on Man-Machine Systems (ICoMMS)*. Universiti Malaysia Perlis, 2009.

[8] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[9] Hasnida Saad, Ahmad Puad Ismail, Norazila Othman, Mohamad Huzaimy Jusoh, Nani Fadzlina Naim, and Nur Azam Ahmad. Recognizing the ripeness of bananas using artificial neural network based on histogram approach. In *Signal and Image Processing Applications (ICSIPA)*, pages 536–541. IEEE, 2009.