Detecting and Classifying Food Produce Using AI to Enhance Crop Sustainability

Artificial Intelligence

ENGTECH 4AI3

Daniel Ebere

[chikezd@mcmaster.ca](mailto:chikezd@mcmaster.ca)

Muhammad Malik

[Malikm56@mcmaster.ca](mailto:Malikm56@mcmaster.ca)

Nisaanth Gunaratnam

[Gunaratn@mcmaster.ca](mailto:Gunaratn@mcmaster.ca)

Ali Rezaie

[rezaiea@mcmaster.ca](mailto:rezaiea@mcmaster.ca)

Abstract *- Integrating artificial intelligence (AI) in the agricultural sector enables real-time monitoring and management of crops. This involves identifying and separating substandard produce, classifying them based on quality, allocating them into designated containers, and transporting them to field collection points. AI-driven advancements have the potential to speed up on-farm sorting and transportation processes, ensuring greater precision and resilience while significantly reducing post-harvest losses.*

***Keywords: Foods, Feature Extraction, Conventional Neural Network (CNN), Food Classification***

# INTRODUCTION

New technologies that seek to increase sustainability and production are causing a rapid transformation in the agricultural sector. Artificial intelligence (AI), particularly the application of neural networks to optimize harvesting procedures, is one significant innovation.

This paper describes the operation of neural networks and how they might be used to interpret intricate agricultural data. AI in farming signifies a substantial change toward data-driven decision-making and is not merely a passing fad. Crop quality and output may be significantly impacted by this change. The necessity for effective and sustainable farming methods only increases with the world's population. Large volumes of data, like photos of crops, can be analyzed using neural networks, which are built like the human brain.

This paper will outline the types of data needed for effective classification and how to train neural networks to identify whether tomatoes and apples are ripe or unripe and to assess the overall health of the crop. By using AI-powered neural networks to increase yields and decrease resource waste, farmers may achieve sustainability goals. This study looks at AI's potential in agriculture, how it will impact farming practices going forward, and how it could contribute to the development of a more sustainable food system.

Understanding the connection between productivity and technology can help stakeholders better manage the challenges of modern farming in a data-driven environment.

# HOW TO ACHIEVE

This process outlines the steps to develop an AI system for identifying, sorting, and transporting produce in agricultural settings. Here’s a breakdown of each step:

## Data Collection and Annotation:

The first step involves gathering a large amount of image and sensor data from various types of produce. This data will serve as the foundation for training the AI model. Once the data is collected, it's important to annotate it, which means labeling images based on quality. This includes marking both acceptable and substandard produce, with specific tags for defects like discoloration or deformities. The annotated data is used to create a dataset that classifies produce according to industry standards. This serves as a reference for the AI model.

**2. Model Training and Object Detection**: Using machine learning techniques, particularly Convolutional Neural Networks (CNNs), the system is trained to recognize and classify the quality of the produce based on its visual features. The model is developed to detect and locate individual produce items effectively, facilitating the classification process.

**3. Real-Time Produce Sorting System**: The computer vision model is then integrated with sorting hardware, enabling automated processes. Robotic arms or conveyor belts are employed to separate substandard produce from acceptable items based on the classifications made by the AI.

**4. Container Allocation and Organization**: Different types of containers are defined to hold the sorted produce according to its quality. An automated system is created for the easy and efficient distribution of produce into the designated containers. The system then tracks the contents of each container to ensure proper classification and volume limits.

**5. Transportation Coordination to Collection Points**: An AI-driven system plans the logistics for transporting containers back to field collection points, ensuring efficient movement. Sensors are used to keep track of container status, providing alerts when transportation is necessary. Transportation routes and schedules are optimized to enhance efficiency.

**6. Testing and Calibration**: The entire system is tested under various conditions to ensure accuracy and effectiveness. Calibration is done to minimize errors in classification.

**7. Deployment and Continuous Monitoring**: After adjustments, the AI system is deployed in real agricultural settings, with real-time monitoring capabilities for performance assessment. Feedback loops are implemented to allow the AI model to learn and adapt to new conditions or updated standards, improving its performance over time.

By following these steps, a comprehensive AI system is built to efficiently identify, sort, classify, and transport produce in real-world environments, ultimately enhancing productivity and quality control in agriculture.

**2.1 Data Collection and Annotation Selection Criteria:**

* **Color:** AI analyzes color consistency to detect ripeness, freshness, or signs of spoilage. For instance, unusual discoloration (like brown or black spots) might indicate rot or bruising.
* **Shape and Size:** Regular shape and uniform size are often associated with high-quality produce. Deviations from the expected shape or size can indicate poor growth conditions, pest damage, or defects.
* **Texture and Surface Features:** AI can detect irregularities on the surface, such as blemishes, bruises, or scars, indicating handling damage or disease.
* **Bruising and Damage**: Impact damage from harvesting or handling can cause bruising. AI models identify softer areas, indentations, or changes in color that indicate bruising.
* **Insect and Pest Damage**: AI systems can recognize common pest damage patterns, like small holes, scars, or darkened areas left by pests.
* **Color Analysis for Ripeness**: For fruits and vegetables, color changes indicate ripeness. AI models assess maturity by comparing produce color with a standard ripeness scale.
* **Size Relative to Expected Growth Stage**: Size and weight of produce can be compared to expected benchmarks for optimal harvest. Out-of-spec sizes often indicate immaturity or over-ripeness.
* **Spotting Symptoms of Disease**: Specific patterns or colors on leaves and produce indicate infections (like blight or mold). AI recognizes these disease signatures, using both visual data and sometimes multispectral imaging.

**2.2 Don’t include:**

* **Moisture Content**: Using near-infrared or hyperspectral imaging, AI can estimate moisture levels, which correlates with freshness. Low moisture can indicate that produce has been in storage too long or is drying out.
* **Mold and Mildew**: Using hyperspectral imaging, AI can detect early signs of mold or mildew on the surface of produce, which may not be visible in regular RGB images.
* **Firmness**: For crops where firmness is a quality indicator, AI integrates with tactile sensors to check for softness or mushiness, signaling degradation.
* **Infrared Scanning for Temperature Variations**: AI uses infrared imaging to detect temperature anomalies, which can indicate bacterial or fungal infections on produce.

**2.3 Separation of Crops:**

* **Grade Sorting**: Based on standards for premium, standard, and substandard grades, AI separates produce by quality classes. Criteria are set to match requirements for fresh markets, processing, or disposal.
* **Intended Use**: For produce destined for processing (e.g., juicing), less stringent criteria may be applied, whereas fresh market produce requires a high-quality standard.

# CONVENTIONAL NEURAL NETWORK (CNN):

Conventional Neural Network (CNN) is a type of deep learning algorithm that is used for handling images and producing features from its input data. The CNN comprises of an input layer, hidden layer and an output layer. The input receives the raw image while the hidden layer executes the convolution. The inner layer has filters which detects specific features in the images. The CNN operation moves a filter over the input data and calculates the results between the inputs which filters weights at every position. CNN scans the entire image with pooling layers which further decreases the convolved feature reducing the spatial size of the CNN[1] .

Figure 1: CNN Process

## Fully Connected Layer

After pooling layers, the Fully connected layers interpret the prior layers extracted from the convolutional layers by connecting the neurons between layers

## Output Layer

The output layer shows the final prediction. Its structure transforms data from previous layers into the desired output format.

## TensorFlow

It is an open-source Machine Learning (ML) framework developed by Google for building, training, and developing ML models. Tensorflow provides neural network architectures and scripts to retain the networks for users[2].

## Keras

An open-source deep learning (DL) framework that provides an interface for developing ML models particularly neutral networks[2]. Keras is a google product which provides TensorFlow deep learning architectures such as I DenseNet121 used in the project.

## DenseNet121

DenseNet121 is a convolutional neural network (CNN) architecture that is particularly known for its dense connectivity pattern. It has 121 hidden layers which means that each layer in the network is directly connected to every other layer that comes after it. This dense connectivity has a few benefits like feature reuse, stronger gradients, and reduced overfitting[3].

A diagram of a diagram of a diagram

Description automatically generated with medium confidence

Figure 2: Dense Blocks [4]

# ALGORITHM:

The algorithm used in the project did a variety of functions which involved extracting images from train data to learn the features in the image sets to produce an accurate estimate. The algorithm is grouped into these steps.

* Select the images of foods to use in the dataset.
* Resize all fruit images to a standard size.
* Convert the dataset format from for compatibility with the model.
* Split the dataset into training, testing, and validation sets.
* Normalize pixel values from 0-240.
* Prepare class labels for training.
* Define the architecture of the model to be used for classification (Densenet121).
* Compile the model with the following: Optimizer, Loss Function, Metrics
* Fit the model for the training data.
* Evaluate the model on the test dataset.
* Calculate the confusion matrix to assess classification performance.

## Data Acquisition and Preparation:

The code starts by using the Kaggle API to download the dataset, which contains images of foods in various conditions (fresh, rotten). It then organizes this dataset into a directory structure with separate folders for training, validation, and testing data. Images are divided into subfolders according to their class (e.g., fresh apples, rotten tomatoes).

## Data Preprocessing:

“ImageDataGenerator" from Keras is used to prepare the images for training and validation. It applies rescaling (normalizing pixel values between 0 and 1) and optionally data augmentation (like rotations, flips) to increase training data diversity. The “flow\_from\_directory” method creates iterators (“train\_dataset”, “val\_dataset”) to feed data to the model in batches.

## Model Selection and Creation:

DenseNet121 is chosen as the model architecture for image classification. DenseNet is known for its dense connections between layers, promoting feature reuse and efficiency.

The model is instantiated with specific parameters: “include\_top=True for classification”, “weights=None” to train from scratch, and “classes=4 “corresponding to the four food classes in the dataset.

## Model Compilation and Training:

The model is compiled with the “categorical\_crossentropy” loss function (suitable for multi-class classification), the Adam optimizer, and accuracy as the evaluation metric.

Training is performed using model.fit, specifying training and validation data, epochs, and callbacks:

annealer: Adjusts the learning rate during training.

mc: Saves the best model based on validation loss.

es: Stops training early if validation accuracy plateaus.

## Model Evaluation:

After training, the model is evaluated on a separate test dataset using” model.evaluate”. This provides metrics like loss and accuracy on unseen data.

Predictions are generated with model.predict, and classification reports and confusion matrices are used to analyze the model's performance in detail.

## Visualization:

The code includes sections for visualizing the model architecture using “plot\_model “and “visualkeras.” This helps understand the structure and layers of the model.

The algorithm downloads a dataset of fruit images, preprocesses them, trains a DenseNet121 model to classify the foods based on their condition, and then evaluates the model's performance on a separate test set.

# PROCESS

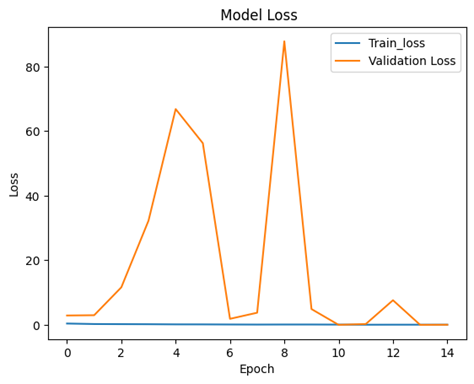
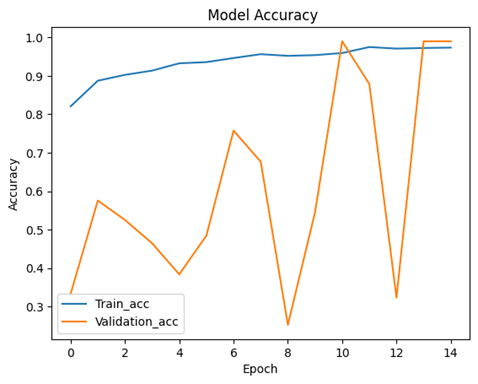
This AI will separate tomatoes and apple into different categories. First it will separate based on ripe or unripe. Then it will separate the ripe one into disease and not disease and the unripe ones into disease or not disease.

# RESULTS

After running the algorithm and generating the following results, the program works well. In the figures shown below, the difference can be seen between the training and validation sets during the repeated learning.

In the model accuracy graph, the training curve improves steadily over each cycle that indicates that the algorithm is learning well from the training dataset. Were as for the validation curve, it fluctuates often below high and low values over each cycle indicates the model struggling to perform with new data. After 10 cycles, we have a model accuracy of 94 % which the new data set.

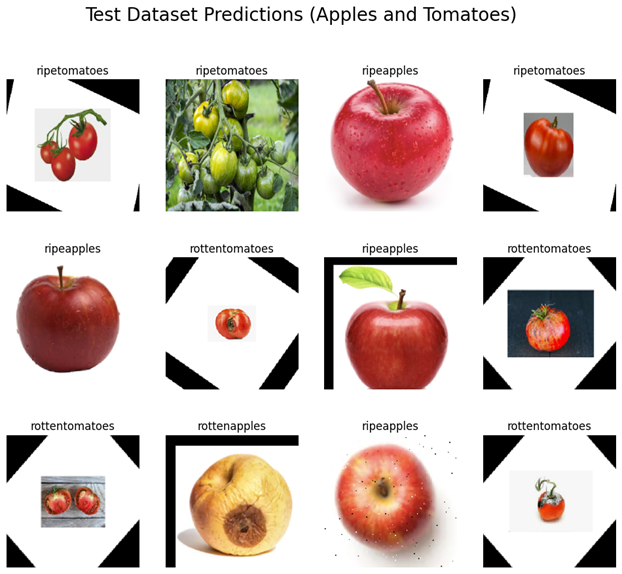
In the model loss graph, it can be seen the training loss curves remains consistent low throughout the cycles of learning which indicates the models fits the training data well. Were as for the validation loss curve, it can be seen to fluctuate significantly which can be seen as the model struggling to perform with new data. This can be the result of the model overfitting.

***Figure 1 & 2:*** Classification accuracies and total losses for training and validation of the dataset

In figure 3, represents the test dataset predictions made by the algorithm for both apples and tomatoes. Each section represents an image from the dataset and the predicted label which shows how the algorithm performed. The algorithm performs well in predicting between ripe and rotten apples and tomatoes from the dataset. It correctly identifies mold on the apples and tomatoes and predicted them to be rotten and the identify the colour of ripe apples and tomatoes.

In the second image, we can see that the algorithm has a issue with misclassifying green tomatoes as ripe which can be seen as the model not being trained enough to distinguish between unripe and ripe tomatoes.

In the bottom row, the first image, the model predicted this image to be rotten but in truth, it is ripe. They can be caused by the models thinking that due to the colour and texture from it being cut in half, it is rotten.

***Figure 3:*** Test database prediction from the algorithm

In figure 4, the confusion matrix represents the evaluation of the prediction for the classification of apples and tomatoes. The true label represents the labels for the actual class labels and the predicted labels are the labels predicted by the model. The diagonal cells in yellow and gold shows the correct predictions by the model and the others show the incorrect predictions for each of the labels. The analysis for each of the classes can be show as follows:

1. **Rotten Tomatoes:**
   * 263 instances are correctly classified as rotten tomatoes.
2. **Rotten Apples:**

* 198 instances are correctly classified as rotten apples.
* 4 are misclassified as rotten tomatoes.
* 42 are misclassified as ripe apples.
* 19 are misclassified as ripe tomatoes.

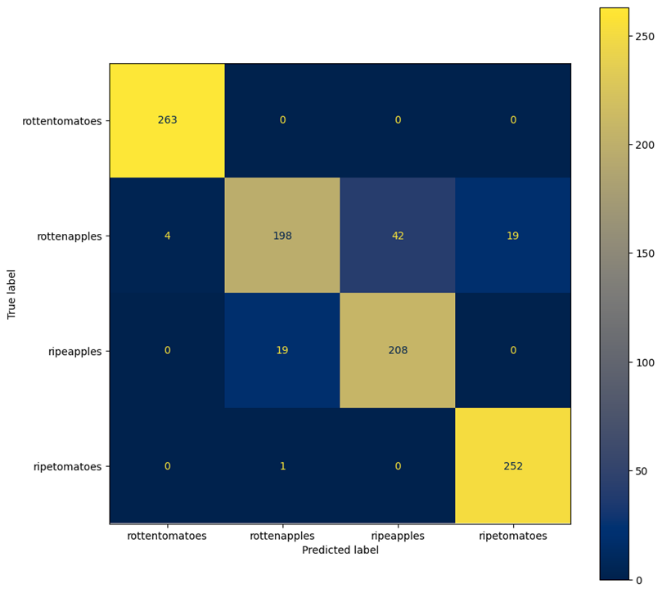
1. **Ripe Apples:**

* 208 instances are correctly classified as ripe apples.
* 19 are misclassified as rotten apples.

1. **Ripe Tomatoes:**

* 252 instances are correctly classified as ripe tomatoes.
* 1 instance is misclassified as rotten apples.

This shows the models preforms well as majority of the labels fall under the diagonal cells indicating they are correct. We can see the misclassification occurs between rotten apples and ripe apples which may be caused both being like each other. Otherwise, rotten tomatoes and ripe tomatoes have very few misclassifications.

***Figure 4:*** Confusion Matrix for apples and tomato classification

# CONCLUSION

The algorithm achieves ~ 94% accuracy in classifying apples and tomatoes after 10 training cycles, but validation accuracy fluctuations indicate overfitting. It performs well with tomatoes but struggles with apples. Future improvements will focus on reducing overfitting and enhancing accuracy through:

1. Increasing training data via diverse datasets and data augmentation. 2. Adjusting model architecture with alternative models and hyperparameter tuning. 3. Applying regularization techniques like dropout and weight decay. 4. Enhancing data preprocessing through normalization and feature engineering.

# References

[1] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the Inception Architecture for Computer Vision,” Dec. 2015, [Online]. Available: http://arxiv.org/abs/1512.00567

[2] H. Lee and J. Song, “Introduction to convolutional neural network using Keras; an understanding from a statistician,” *Commun Stat Appl Methods*, vol. 26, no. 6, pp. 591–610, Nov. 2019, doi: 10.29220/CSAM.2019.26.6.591.

[3] J. Naranjo-Torres, M. Mora, R. Hernández-García, R. J. Barrientos, C. Fredes, and A. Valenzuela, “A review of convolutional neural network applied to fruit image processing,” May 01, 2020, *MDPI AG*. doi: 10.3390/app10103443.

[4] X. Li, X. Shen, Y. Zhou, X. Wang, and T.-Q. Li, “Classification of breast cancer histopathological images using interleaved DenseNet with SENet (IDSNet),” *PLoS One*, vol. 15, no. 5, p. e0232127, May 2020, doi: 10.1371/journal.pone.0232127.