AI Final Research Paper
GuavaScript: a Guava Quality Predictor

Group 3 110207404 Julia 林佳瑢 111ZU1003 Anita 林品萱
111ZU1012 Diana 魏彤芸 111ZU1016 Jenny 郭怡均 111ZU1060 Chloe 陳恩愛

Project description

Guavas, an important agricultural product in Taiwan, play a vital role in both export and domestic sales. This AI project aims to address several societal problems. Firstly, people often touch and inspect every fruit to find the "perfect" ones, but they do not end up purchasing them, causing damage during selection and picking. Secondly, there is wastage of good-tasting but unsalable fruit because people tend to buy beautiful-looking fruit assuming it is tasty, which is a myth. Lastly, Taiwan faces a growing problem of leftover food, despite claiming to be thrifty and simple. It is an urgent issue that needs to be addressed.

This AI project, called "GuavaScript: A Guava Quality Predictor," aims to develop and implement deep learning models using the Un2net model for image recognition and VGG16, a Convolutional Neural Network (CNN) model. Customers can upload a picture of a guava and provide traceability information, and the AI will provide information about the guava. If successful, our AI can help people purchase guavas that match their taste, reduce the number of bruised guavas, increase sales of good tasting but unsalable guavas, address the wastage of all types of guavas, and reduce the leftover food problem in Taiwan. Additionally, increased domestic sales can help mitigate the impact of China's boycott on Taiwanese fruits.

Here are two studies related to our project. The first study, "Non-destructive Fruit Sugar Content Evaluation Using Deep Learning Network" [1], conducted by the Information Management department at Chaoyang University of Technology, inspected 130 Fuji apples from New Zealand, Chile, and other origins. They took pictures of the apples from eight different angles, including light-colored side, dark-colored side, top, and bottom. By connecting the data on the appearance and sweetness of the apples, they trained an AI model to predict the taste, proving the high probability of fruit recognition and the importance of different angles of the fruits. The second study, "Quality Evaluation in Guavas using Deep Learning Architectures: An Experimental Review" [2], collected 491 pictures of 24 kinds of guavas over a span of 15 days in a controlled environment. They used the Mask R-CNN technique PixelLib for image segmentation and trained three CNN models (VGG16, Inception, and V3Xception) for sample predictions. This research emphasized the importance of clear images and the degree of flavor determinants.

Our AI project process is divided into six parts, which will be described in detail later.

1. Collecting Data: Gather guava images from different days and corresponding excel data, including features like Origin, Day (From Harvest), Annual Rainfall, Irrigation/Watering Frequency, as well as quality attributes such as Sweetness, Sourness, Crunchiness, Hardness, and Flavor, which will be determined on a scale of 0-5.

2. Data Preparation: Match the images with their respective data by ensuring proper alignment between the image files and the data records.

3. Data Preprocessing: Preprocess the images by removing their backgrounds.

4. Build the Model: Use Convolutional Neural Networks (CNNs) or deep learning models to predict guava's quality attributes.

5. Train the Model: Split the data into training and testing sets. Train the model using the training set and evaluate its performance using the testing set.

6. Predict Quality: Use the trained model to predict the quality attributes of new images and related data.

Data Preparation and Preprocessing:

For data collection, we divide it into two parts. First, we collect images of guavas. Each group member buys about five guavas, takes four angles of pictures per day, and records the taste results, including "sweetness," "sourness," "crunchiness," "hardness," and "flavor." The taste results are rated on a scale of 0-5. The recording days for each member may differ. Second, we collect guava information such as Origin, Day (From Harvest), Average Yield (per hectare), Annual Temperature, Annual Rainfall, Irrigation/Watering Frequency, Fertilizer Frequency, Field Cultivation/Land Management Frequency, and Pest Control Frequency from Agriculture and Food Agency, Council of Agriculture, Executive Yuan, and Central Weather Bureau Data Inquire Services.

For data preprocessing, here are the steps process in GuavaScript:

1. **Background Removal of Guava Photos**:
    - The preprocessing process begins by removing the background from guava photos. This step is performed separately for different folders named Anita, Chloe, Jenny, and Julia, each containing guava photos.
    - The AI utilizes a tool called "rembg" to perform the background removal. It loads the guava photos and applies the background removal algorithm to produce images with transparent backgrounds.
    - The background removal is done using a loop that iterates over the files in each folder. The input path of the guava photo is read, and the output path for the resulting image with the removed background is determined.
    - The AI uses the "rembg" tool's functionality to remove the background from the guava photo, and the resulting image data is written to the output path.
2. **Creating Photo Tags**:
    - After the background removal, the AI creates a CSV file called "Photo_Tag.csv" to store the information about the guava photos. This file will be used later for data processing and analysis.
    - The AI opens the CSV file in write mode and creates a csv writer to write the data.
    - It iterates over the guava photo files in the "PhotoData" folder (the folder where the background-removed images are stored). For each file, it extracts the relevant information, such as the file name, origin, and day.
    - The AI writes a row to the CSV file for each guava photo, containing the file name, origin (initially left blank), and day of the guava.
3. **Preparing the Training and Testing Data**:

- To train and evaluate the machine learning models, the AI needs to split the data into training and testing sets.
- The AI loads the necessary CSV files containing the guava data. Specifically, it loads "Machine1.csv" and "Machine2.csv" as two DataFrames to get the training data, and "Machine1.csv" and "Machine3.csv" to get the testing data.
- "Machine1.csv" is added from the “Photo_Tag.csv”, which contains the correct **Day**, Origin, Average yield, Annual rainfall, Annual average temperature, irrigation frequency, fertilization frequency, land management frequency, pest control frequency collected from Agriculture and Food Agency, Council of Agriculture, Executive Yuan, and Central Weather Bureau Data Inquire Services.
- "Machine2.csv" contains the correct Origin, **tasting results**, Average yield, Annual rainfall, Annual average temperature, irrigation frequency, fertilization frequency, land management frequency, pest control frequency collected from Agriculture and Food Agency, Council of Agriculture, Executive Yuan, and Central Weather Bureau Data Inquire Services.
- For the training data, the AI performs a join operation on the two loaded DataFrames, merging them based on specified columns like origin, day, irrigation frequency, fertilization frequency, field management frequency, and pest control frequency. The resulting merged DataFrame is saved as "Machine2_new.csv".
- "Machine3.csv" is a file which deletes the redundant columns from "Machine2_new.csv".
- For the testing data, the AI compares the file names in the two CSV files and removes the matching lines from the first DataFrame, leaving the remaining lines as the testing data. The filtered DataFrame is saved as "TestData.csv".

These preprocessing steps are essential to prepare the guava photos and associated data for further analysis and modeling. The background removal enhances the quality of the images by removing unnecessary elements, while the creation of photo tags and data splitting ensure that the AI has the necessary information for training and evaluating the machine learning models accurately.

File Structure

The file structure of our project includes:

- Folders contain guava photos used for preprocessing and analysis
  - Anita
  - Chloe
  - Jenny
  - Julia
- Folder stores the preprocessed guava photos with background removed
  - PhotoData
- Main code file containing the entire project code
  - GuavaScript.html
  - GuavaScript.ipynb
  - GuavaScript.py
  - requirements.txt
- CSV files contain the dataset and guava attributes used for training and testing
  - GuavaScript.xlsx (raw data)

    - Photo_Tag.csv
    - Machine1.csv
    - Machine2.csv
    - Machine2_new.csv
    - Machine3.csv
    - CodeData.csv: Integrates traceable barcode and guava production data. If the "traceability barcode" option is selected when running Gradio, these production data will be input together with the photos.
    - - DayAI & TasteAI: AI models developed for the project.
- CSV files contain the testing data and the predicted guava attributes, respectively
    - TestData.csv
    - ResultData.csv
- Trained models for day prediction and taste prediction, respectively
    - DayAI.h5
    - TasteAI.joblib

Introduction of modules

1. **os**: The **os** package is used to interact with the operating system. It is necessary for performing file and folder operations, such as getting absolute paths and creating directories. In GuavaScript, it is used to manage file and folder paths during preprocessing and data handling.

2. **tempfile**: The **tempfile** package provides functions for working with temporary files and directories. In GuavaScript, it is used to create temporary files for storing images during preprocessing. Temporary files are useful for managing resources efficiently and avoiding cluttering the system with unnecessary files.

3. **cv2 (OpenCV)**: OpenCV is a powerful computer vision library that provides various image processing and computer vision algorithms. It is extensively used in GuavaScript for image loading, color conversion, resizing, and other image-related operations. OpenCV offers efficient and optimized functions for working with images, making it an essential package for computer vision tasks.

4. **numpy (np)**: NumPy is a fundamental package for scientific computing in Python. It provides support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions. In GuavaScript, NumPy is used for array operations, data manipulation, and efficient storage and computation of image data and numerical features.

5. **pandas (pd)**: Pandas is a popular library for data manipulation and analysis. It offers data structures like DataFrames and functions to efficiently work with structured data. In GuavaScript, Pandas is used for loading, merging, and manipulating data from CSV files. It provides convenient methods for handling and preprocessing tabular data.

6. **tensorflow (tf)**: TensorFlow is a widely-used open-source machine learning framework. It provides tools and libraries for building and training machine learning models, including deep learning models. In GuavaScript, TensorFlow is used for various deep learning tasks, such as training and using the VGG16 model for predicting the day of the guava.

7. **tensorflow.keras**: TensorFlow's Keras API is a high-level neural networks API that provides a user-friendly interface for building and training deep learning models. In GuavaScript, it is used for various tasks, such as image preprocessing, model creation, and evaluation. The **keras.preprocessing.image** module from TensorFlow is used for image data augmentation and preprocessing.

8. **ImageDataGenerator**: The **ImageDataGenerator** class from **tensorflow.keras.preprocessing.image** module provides real-time data augmentation for image data. In GuavaScript, it is used to generate augmented training data by applying various transformations like rotation, shift, shear, zoom, and flip. Data augmentation helps improve model performance and generalization by creating additional training examples.

9. **from tensorflow.keras.applications.vgg16**: The **tensorflow.keras.applications.vgg16** module provides the VGG16 model, a pre-trained convolutional neural network architecture commonly used for image classification tasks. In GuavaScript, the VGG16 model is used as the base model for predicting the day of the guava based on its photo. The pre-trained weights of the VGG16 model are used to leverage its learned features for transfer learning.

   We tried VGG16, ResNet50, and InceptionV3, which are common pre-trained CNN models used for transfer learning. Eventually, we chose VGG16 because it performed the best. We used VGG16 to predict the number of days the guava had been harvested. After obtaining the days, we employed Random Forest to predict its palatability.

   1. ResNet50 is a convolutional neural network with 50 layers. It can effectively train deep neural networks while minimizing the parameter count through the bottleneck design.

   2. InceptionV3 features 7x7 factorized convolutions, which decompose convolutions into asymmetric convolutions. Increasing the model's depth while reducing computational requirements leads to improved efficiency.

10. **decode_predictions**: The **decode_predictions** function from the **tensorflow.keras.applications.vgg16** module is used to decode the predictions made by the VGG16 model into a list of labels. It helps in interpreting the predictions made by the VGG16 model and mapping them to meaningful class labels.

11. **train_test_split**: The **train_test_split** function from the **sklearn.model_selection** module is used to split the data into training and testing sets. In GuavaScript, it is used to divide the guava photo dataset into training and validation sets for model training and evaluation. It helps assess the model's performance on unseen data.

12. **LabelEncoder**: The **LabelEncoder** class from the **sklearn.preprocessing** module is used to encode categorical features into numerical values. In GuavaScript, it is used to encode the origin feature in the taste prediction model. Label encoding is necessary for representing categorical variables as numerical inputs for machine learning algorithms.

13. **StandardScaler**: The **StandardScaler** class from the **sklearn.preprocessing** module is used to standardize features by removing the mean and scaling to unit variance. In GuavaScript, it is used to scale the features in the taste prediction model. Feature scaling is important for

ensuring that different features contribute equally to the model and avoiding issues with feature magnitudes.

14. **RandomForestRegressor**: The **RandomForestRegressor** class from the **sklearn.ensemble** module is used to implement a random forest regressor model. In GuavaScript, it is used for training the taste prediction model, which predicts the taste attributes of guavas. Random forests are powerful ensemble models that can capture complex relationships and provide accurate predictions.

    There are several reasons why we chose Random Forest. First, it exhibits robustness to overfitting, which occurs when a model fails to generalize beyond the training data, leading to decreased accuracy. Second, Random Forest demonstrates excellent capability in handling high-dimensional data. It effectively deals with datasets that contain a large number of features. Third, it is capable of handling and classifying mixed data types, accommodating variables of different types in the same model. Fourth, Random Forest provides feature importance, enabling us to assess the relevance of individual features in the classification process. This information helps identify any uninformative or redundant features. Lastly, Random Forest is well-suited for capturing non-linear relationships in data, making it suitable for datasets with complex patterns.

15. **r2_score**: The **r2_score** function from the **sklearn.metrics** module is used to compute the R-squared score, a statistical measure of how well the regression predictions fit the actual values. In GuavaScript, it is used to evaluate the performance of the taste prediction model. R-squared helps assess the model's ability to explain the variance in the target variables.

16. **dump, load**: The **dump** and **load** functions from the **joblib** module are used to serialize and deserialize Python objects. In GuavaScript, they are used to save the taste prediction model to a file and load it for making predictions. Serialization allows models to be saved and reused without retraining.

17. **hog**: The **hog** function from the **skimage.feature** module is used to compute Histogram of Oriented Gradients (HOG) features for an image. In GuavaScript, it is used to extract features from guava photos for the taste prediction model. HOG features capture the local structure and texture information of an image and are useful for object detection and recognition tasks.

18. **Path**: The **Path** class from the **pathlib** module provides a convenient way to work with file and directory paths. In GuavaScript, it is used to work with file paths and directories, such as accessing and iterating over files in a directory.

19. **Image**: The **Image** class from the **PIL** (Python Imaging Library) module represents an image. It provides various image manipulation and processing functionalities. In GuavaScript, the **Image** class is used for image-related operations, such as resizing and saving.

20. **remove, new_session**: The **remove** and **new_session** functions from the **rembg** module are used for background removal functionality for images. In GuavaScript, they are used to remove the background from guava photos in the preprocessing step. Background removal helps isolate the guava from the surroundings, making it easier to extract features and analyze the guava image.

21. **gradio**: Gradio is a library for creating customizable and interactive UI components for machine learning models. In GuavaScript, it is used to create the user interface for predicting guava quality based on photos and traceable information. Gradio simplifies the process of building user interfaces, allowing users to interact with the AI model easily.

These packages and modules collectively provide a comprehensive set of tools and functionalities required for image processing, machine learning, data manipulation, and user interface creation, enabling the GuavaScript AI to perform its tasks effectively and efficiently.

Development

1. **Preprocessing the Dataset**:
    - The first step is to remove the background of the guava photos. This is achieved using the "rembg" tool, which helps in separating the guava from its background. The background removal process is performed separately for different folders containing guava photos (Anita, Chloe, Jenny, and Julia).
    - After background removal, a photo tags file is created. This CSV file contains information about the guava photos, including the file name, origin, and day. It provides a structured representation of the guava photo dataset.
    - The training and testing data are obtained by loading and merging data from different CSV files that contain information about guava characteristics and features. The data is split into training and testing sets to facilitate model training and evaluation.
2. **Identifying the Day of the Guava**:
    - The AI employs the VGG16 model, a pre-trained convolutional neural network (CNN) architecture, to predict the day of the guava based on its photo. The VGG16 model is loaded using TensorFlow's Keras API.
    - The guava photos are preprocessed using techniques such as resizing and normalization to match the input requirements of the VGG16 model.
    - Data augmentation is applied using the **ImageDataGenerator** class from TensorFlow to generate augmented training data. Augmentation techniques include rotation, shifting, shearing, zooming, and flipping of images.
    - The VGG16 model is fine-tuned by adding custom top layers that include global average pooling, dense layers, and a linear activation layer. The model is compiled with appropriate optimizer and loss function settings.
    - The model is then trained using the augmented training data, and its performance is evaluated using validation data. The training process involves optimizing the model's weights based on the calculated loss.
    - The trained model is saved in the H5 file format for future use in predicting the day of guavas.
3. **Identifying the Quality of the Guava**:
    - The taste prediction model is built using the Random Forest Regressor algorithm from the scikit-learn library. The model predicts the taste attributes of guavas, including sweetness, sourness, crunchiness, hardness, and flavor, based on guava photos and other features.
    - The guava photos are preprocessed by converting them to grayscale, resizing them to a desired size, and extracting Histogram of Oriented Gradients (HOG) features. The HOG features capture the shape and texture information of the guava photos.

- The taste prediction model combines the HOG features with other features like origin, average yield, rainfall, temperature, and various agricultural management frequencies. The combined features are preprocessed by scaling and encoding categorical variables.
- The preprocessed features are used to train the Random Forest Regressor model. The model learns the relationships between the input features and the taste attributes through the ensemble of decision trees.
- The trained taste prediction model is saved as a serialized file using the joblib library. This allows the model to be loaded and used for predicting guava taste attributes in the future.

4. **Using GuavaScript to Predict Testing Data**:
    - The testing data, which consists of guava photos and corresponding information, is loaded from a CSV file.
    - For each guava photo in the testing data, the day is predicted using the trained VGG16 model. The photo is preprocessed, and the VGG16 model is used to predict the day.
    - The predicted day and other relevant features are combined to create an input dataset for the taste prediction model. The features are preprocessed in the same manner as during model training.
    - The taste prediction model, which was trained using the Random Forest Regressor, is loaded. The model takes the preprocessed features as input and predicts the taste attributes of the guava.
    - The predicted taste attributes are added as new columns to the testing data. The results, including the guava photo, input features, and predicted taste attributes, are saved in a new CSV file.
5. **Becoming Gradio**:
    - The Gradio library is utilized to create an interactive user interface for the GuavaScript AI.
    - The AI models and relevant data are loaded, including the VGG16 model for predicting the day and the taste prediction model for predicting the taste attributes.
    - The Gradio interface allows users to upload a guava photo and select its origin (traceability code) as inputs. The interface provides real-time feedback and visualizations of the predicted taste attributes.
    - The user interface provides a user-friendly way for individuals to interact with the AI and obtain predictions of guava quality based on photos and traceable information.

Analysis

To identify the number of days after the guavas have been picked, we follow a series of steps. First, we read the CSV file containing the relevant data. Next, we build access to the columns to extract the necessary information. We then load all the guava photos into Google Drive for convenient storage and management. Afterwards, we convert the columns from the CSV file into a NumPy array, enabling efficient data processing. To evaluate the model's performance, we split the data into training and testing datasets. Moving on, we create an Image Data Generator to perform data augmentation, generating augmented training data with variations. While the augmented data is essential for training, we use the original validated data to evaluate the model's accuracy. To build the model architecture, we create a VGG16 model, adding custom top layers suitable for regression. The model is compiled with the appropriate loss function, optimizer, and evaluation metrics. We then train the model using

the augmented data, optimizing its performance. After training, we evaluate the model using the loss value as a measure of its effectiveness. The model is saved as an H5 file for future use. We obtain predicted ages using the trained model and calculate the R-squared value, providing an assessment of the model's predictive capability. By following these steps, we can accurately determine the number of days after guavas have been picked, enabling better quality control and freshness management.

To identify the taste of guavas, we follow a series of steps. First, we load the guava information dataset, which provides us with valuable data for our analysis. Additionally, we gather photos of guavas, as visual information can play a crucial role in determining taste. Once the photos are loaded, we ensure that the images are properly prepared and adjust their size if necessary. To extract meaningful features from the original photos, we employ the Histogram of Oriented Gradients (HOG) technique. This allows us to capture relevant information about the texture and shape of the guavas. We combine these image features with our observed features, which may include variables such as ripeness, color, and size. Next, we convert the column names to strings and encode categorical variables, as machine learning algorithms typically require numerical inputs. Additionally, we scale the features to ensure they are on a similar scale, avoiding any bias that may arise from different measurement units. To assign appropriate weights to the features, we carefully analyze their importance and relevance to the taste identification task. This step helps us prioritize the most informative features in our analysis. We then select the non-scaled features and combine them with the scaled ones. Before training our model, we split the data into training and testing datasets. This separation allows us to assess the model's performance on unseen data. Using a Random Forest Regressor model, we train our model on the training dataset, leveraging the algorithm's ability to capture complex relationships between features. As soon as the model is trained, we make predictions on the testing dataset and evaluate its performance. To measure accuracy, we compare the predicted taste values with the actual taste values. This evaluation step provides valuable insights into the model's ability to accurately identify the taste of guavas. Finally, we save the trained model as a file, allowing us to use it for future taste identification tasks without the need for retraining. This step ensures efficiency and convenience in deploying our model for real-world applications. By following these steps, we can leverage data analysis and machine learning techniques to gain valuable insights into the taste of guavas.

When it's time to use GuavaScript to predict the testing data, we follow a series of steps to ensure accurate and reliable predictions. First, we load the guava information dataset, which contains the relevant features for our prediction task. Additionally, we load the corresponding photos of guavas, which will be used in conjunction with the observed features. Next, we prepare the images by adjusting their size if necessary. To extract meaningful features from the original photos, we employ the Histogram of Oriented Gradients (HOG) technique. These image features are then combined with the observed features from the dataset. To ensure compatibility, we convert column names to strings and encode categorical variables appropriately. Scaling the features is essential for proper model training, and we assign weights to the features to account for their relative importance. The non-scaled features are selected and combined with the scaled ones. Finally, we split the data into training and testing datasets. Using a Random Forest Regressor model, we train our model on the training data and make predictions on the testing data. To ensure compatibility with the dataset, we convert the predictions into integers and add them as a new column. Finally, we save the results into a CSV file for further analysis and utilization.

To facilitate the photo reprocessing for our guava day prediction task, we can create a function that streamlines the necessary steps. First, we resize the images to a standardized size, ensuring uniformity

in our dataset. Next, we convert the resized images into a NumPy array, which allows for efficient manipulation and processing. With the array in place, we create an Image Data Generator, which will be responsible for applying data augmentation techniques to our images. To prepare the augmented data for model training, we expand the dimensions of the images to match the expected input shape of the model. This step ensures compatibility and seamless integration with the subsequent stages. Now that we have the augmented image data ready, we can build access to it, enabling easy retrieval and utilization. Finally, we store the augmented images in a batch, ready to be fed into the model during the training process. By incorporating these steps within a dedicated function, we can streamline and automate the photo reprocessing pipeline, saving time and effort in our guava day prediction workflow.

To facilitate the prediction of the number of days after guavas have been picked, we can encapsulate the necessary steps into a function. This function will handle the image processing and utilize a TensorFlow (TF) Keras model for the prediction. First, we write a function specifically designed for the picked-days prediction task. Within this function, we perform the required image processing operations, such as loading and preprocessing the guava images. Then, we pass these processed images to the TF Keras model, which has been trained to predict the number of days after picking. The model utilizes the learned patterns and features from the training data to make accurate predictions. Finally, we return the predicted number of days as the output of our function. By encapsulating these steps within a function, we can easily apply this prediction process to new guava images and obtain the desired result.

To facilitate the prediction of the taste of guavas, we can follow the following steps. First, we write a function specifically designed for taste prediction. Next, we load the prepared images of guavas into our system. We then adjust the image size if necessary to ensure consistency in our analysis. To extract meaningful features from the original photos, we utilize the Histogram of Oriented Gradients (HOG) technique. This helps us capture relevant information about the guava images. To ensure we have the necessary data for taste prediction, we fetch the required information from the given dataset. If there is any missing data, we handle it by returning the value "missing." After obtaining the dataset, we proceed to encode any categorical variables present to facilitate further analysis. Next, we scale the features to ensure they have a consistent range, which is crucial for training our model effectively. To assign appropriate weights to the features, we carefully consider their significance in taste prediction. We then select the non-scaled features and combine them with the scaled ones. This step allows us to capture the full range of information available. Finally, we scale the combined features to ensure uniformity before proceeding.

To assess the performance of our model, we split the data into training and testing datasets. The training dataset will be used to train a Random Forest Regressor model, a powerful algorithm for regression tasks. Once the model is trained, we make predictions on the testing dataset to evaluate its accuracy and generalization capabilities.

To demonstrate the capabilities of our machine learning model with Gardio, we need to follow a few essential steps. First, we install and import Gardio, a powerful library for building interactive user interfaces. Next, we load the first AI model for predicting the number of days since the guavas were picked. Additionally, we load the second AI model for predicting the taste of the guavas, which is another crucial factor for consumers. Once the models are loaded, we read the CSV file containing the guava data after conducting thorough testing. From this data, we extract the list of unique origins, providing valuable insights into the guava's geographic diversity. Finally, we define, set up, and

launch the user interface using Gardio, allowing users to interact with our machine learning model seamlessly. Through this integrated approach, we can showcase the accuracy and usability of our AI model in predicting guava freshness and taste, enhancing the overall user experience with Gardio.

Result and Discussion

Through the prediction of the AI, the accuracy is 0.48, which is lower than we expected. It is also less than the research we cited, which is 0.7. Here is our discussion. During the development of this solution, numerous challenges were encountered, with data collection proving particularly difficult. These challenges included capturing guavas from various angles, replicating supermarket lighting conditions, and testing for sweetness based on our team member's subjective feeling since we don't use any instrument to test the guavas, which is a serious problem causing the accuracy to be so low. A significant hurdle was the variability in photo parameters due to the diverse resolutions and RGB settings of different phones, potentially impacting the accuracy of the results.

To address this issue and improve the project, it would be beneficial to use a variety of phones to capture images during the training data preparation phase. It is also helpful to increase our accuracy by collecting more images of guava and to use professional equipment to test guava instead of relying on our subjective feeling. Despite these challenges, the solution offers four key benefits: reducing consumer confusion when selecting fruits, enhancing interaction and experience between businesses and consumers, lowering operating costs for businesses, and minimizing resource waste.

Looking ahead, this approach holds promise not only for supermarkets but also for smart agriculture and quality monitoring. Unmanned drones can be employed in smart agriculture to monitor guavas, enabling the automated harvesting of ripe fruits when the sweetness reaches a predetermined value. In terms of quality monitoring, guavas can be graded based on their sweetness, providing a reliable measure of quality for e-commerce platforms. Consumers can scan the image results to determine whether to purchase a particular guava, and retailers can substantiate the quality of their guavas based on the findings of this research.

Conclusion

In conclusion, artificial intelligence is so important in today's world; it can help us identify those slight changes people are hardly able to observe and get the result in a few seconds. Our AI project, GuavaScript: A guava quality detector, is a great improvement that AI and people can collaborate together to create a better environment. It is a process for predicting guava properties using machine learning. We explored two main prediction tasks: days since picking and guava taste. In terms of predicting the number of days after picking, we used image and data analysis techniques to build a model that can predict the freshness of guava based on images and other observed characteristics. Similarly, in predicting the taste of guava, we used a combination of images and other features to train a model that can accurately identify the taste of guava. The building and training process of these models is carefully designed and tuned to ensure the best predictive performance.

The main contribution of this AI is to show how to use machine learning and data analysis techniques to improve guava quality control and taste management. By combining imagery and other observational features, we were able to build accurate and reliable predictive models to help farmers and operators better manage guava freshness and quality, and customers to match their taste. These predictive models not only provide instant data analysis but also provide valuable guidance for business decision-making and customers.

Overall, by applying machine learning techniques and data analysis methods, we were able to improve the ability to accurately predict guava properties, leading to better quality control and business management. This is an important discovery and contribution for guava growers, wholesalers, and retailers. In the future, we can further improve and optimize these predictive models and apply them to predictive analysis of other fruits and agricultural products.

References

[1] 周治辰(2021)。使用卷積神經網路分類蘋果甜度等級之研究。[碩士論文。朝陽科技大學] 臺灣博碩士論文知識加值系統。 https://hdl.handle.net/11296/65ekn7.

[2] T. Choudhury et al., "Quality Evaluation in Guavas using Deep Learning Architectures: An Experimental Review," 2022 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA), Ankara, Turkey, 2022, pp. 1-6, doi: 10.1109/HORA55278.2022.9799824.

[3] Kazuki Kyakuno (2020). "U2Net: A machine learning model that performs object cropping in a single shot." Medium. https://medium.com/axinc-ai/u2net-a-machine-learning-model-that-performs-object-cropping-in-a-single-shot-48adfc158483

[4] 劉智皓(2021)。機器學習_學習筆記系列(37)：隨機森林回歸(Random Forest Regressor). Medium. https://tomohiroliu22.medium.com/%E6%A9%9F%E5%99%A8%E5%AD%B8%E7%BF%92-%E5%AD%B8%E7%BF%92%E7%AD%86%E8%A8%98%E7%B3%BB%E5%88%97-37-%E9%9A%A8%E6%A9%9F%E6%A3%AE%E6%9E%97%E5%9B%9E%E6%AD%B8-random-forest-regressor-a0f7a57c06c4

[5] 農情報告資源網。111 年鄉鎮作物查詢。https://agr.afa.gov.tw/afa/afa_frame.jsp

[6] 產銷履歷農產品資訊網。
番石榴--履歷燕巢芭樂：https://tqr.tw/?t=2305210094508639
履歷番石榴。https://tqr.tw/?t=2305210094508639
番石榴--珍珠芭樂。https://taft.coa.gov.tw/sp-resume-code-7075595-2305190236628545-1.html
番石榴-履歷芭樂。https://tqr.tw/?t=2305220044418197

[7] CODIS 氣候資料服務系統。111 年年報表。https://codis.cwb.gov.tw/StationData#