DUKE FUQUA SCHOOL OF BUSINESS

DECISION 546Q: Modern Analytics
Section B Team 18
Yinan Chen, Tirth Pravin Gala, Yuhe (Tiffany) Jin, Chenjie (Angelina) Sun

Fruit 360: Image Recognition of Fruits

1. Motivation & Data Understanding

Imagine when you finish a day of work and walk into a grocery store for some fresh products, you have no idea what to purchase. There is a big screen above the racks, and there are numerous fresh products in front of you. As you randomly pick up a tomato, a video demonstrating the recipe for Tomato Tofu Stir Fry Skillet appears on the screen above. You put down the tomato and picked up an apple next to it, the video above changed into the apple pie recipe. Great Idea! You decide to make both the stir fry and the apple pie for yourself tomorrow, so you put zucchini and a few apples into the shopping cart and go on shopping for some tofu, zucchini, eggplant, and basil.

This is an ideal solution in grocery shopping for a store using the technique of artificial intelligence to stimulate customers' appetite for more products. When a customer picks up a vegetable or fruit, the camera on the screen detects the item he/she picks up and uses deep learning techniques to identify the category of the product. Then, a video of a recipe for the chosen product is displayed to the customer. If the customer is entertained and interested in the recipe, he/she will likely purchase other products that are needed for cooking the dish. Finally, the per-customer transaction will increase and improve the performance of the store.

To help stores develop the technique and improve fresh product performance, we use deep learning methods to build models for identifying different kinds of fruits and vegetables. We make use of the Fruits 360 (kaggle.com) from Kaggle with 90,483 images of fruits and vegetables. The size of the training set is 67,692 (one fruit or vegetable per image), and the test

set size is 22,688 (one fruit or vegetable per image). The number of classes is 131 (fruits and vegetables). Images in the dataset share the same size of 100x100 pixels.

2. Data Preparation

In ensuring the effectiveness and accuracy of our model, our first step involved aggregating a dataset of 90,483 fruit and vegetable images. These images, sized at 100x100 pixels, experienced normalization where their pixel values were scaled to a 0-1 scale. We divided the dataset into two sets: 67,692 images for training and 22,688 for validation. The splitting steps balanced evaluation of the model's capabilities and ensured generalizing to unseen data.

For the training set, we apply a series of transformations such as shear (0.3 range), zoom (0.3 range), and horizontal flipping by using Keras's *ImageDataGenerator* method. These methods simulate how fruits and vegetables might appear in real-world scenarios, so we equip the model to recognize these fruits under different conditions. However, for the validation set, we limit processing to simple rescaling (1./255) without additional augmentation. The validation data serves as a consistent basis for evaluating the model's performance by using this step. We found 67,692 images in the training set belonging to 131 classes and 22,688 images in the validation set belonging to 131 classes. Conducting sanity checks involves visually inspecting the augmented images to make sure their realistic portrayal and checking that the class distribution remains balanced post-augmentation. Monitoring the model's performance on the validation set helps detect any signs of overfitting. The below image showcases the types of fruits included in the dataset:

Content

The following fruits and are included:

Apples (different varieties: Crimson Snow, Golden, Golden-Red, Granny Smith, Pink Lady, Red, Red Delicious), Apricot, Avocado, Avocado ripe, Banana (Yellow, Red, Lady Finger), Beetroot Red, Blueberry, Cactus fruit, Cantaloupe (2 varieties), Carambula, Cauliflower, Cherry (different varieties, Rainier), Cherry Wax (Yellow, Red, Black), Chestnut, Clementine, Cocos, Corn (with husk), Cucumber (ripened), Dates, Eggplant, Fig, Ginger Root, Granadilla, Grape (Blue, Pink, White (different varieties)), Grapefruit (Pink, White), Guava, Hazelnut, Huckleberry, Kiwi, Kaki, Kohlrabi, Kumsquats, Lemon (normal, Meyer), Lime, Lychee, Mandarine, Mango (Green, Red), Mangostan, Maracuja, Melon Piel de Sapo, Mulberry, Nectarine (Regular, Flat), Nut (Forest, Pecan), Onion (Red, White), Orange, Papaya, Passion fruit, Peach (different varieties), Pepino, Pear (different varieties, Abate, Forelle, Kaiser, Monster, Red, Stone, Williams), Pepper (Red, Green, Orange, Yellow), Physalis (normal, with Husk), Pineapple (normal, Mini), Pitahaya Red, Plum (different varieties), Pomegranate, Pomelo Sweetie, Potato (Red, Sweet, White), Quince, Rambutan, Raspberry, Redcurrant, Salak, Strawberry (normal, Wedge), Tamarillo, Tangelo, Tomato (different varieties, Maroon, Cherry Red, Yellow, not ripened, Heart), Walnut, Watermelon.

3. Modeling

We build two CNN models using Keras to recognize our dataset. Model 1 is a Sequential model featuring a starting Conv2D layer with 32 filters (3x3), followed by two similar layers, each paired with MaxPooling for dimensionality reduction. A Flatten layer transitions to a Dense layer of 1024 neurons with a dropout rate of 0.5 to prevent overfitting. The model ends with a 131-neuron Dense layer that uses 'softmax' activation, an ideal function for multi-class classification. We used Adam as the optimizer and categorical_crossentropy as the loss function, with a learning rate of 0.001. Model 1 maintains a balance between computing efficiency and the capacity to gather important visual data.

Model 2 increases the initial complexity, beginning with 64 filters in the first Conv2D layer and gradually increasing to 128 filters in the following layers. Similar to Model 1, it ends with a 131-neuron 'softmax' layer. The optimizer and loss function of Model 2 are the same with Model 1. Model 2's larger filter size allows for more detailed feature extraction, which makes it more accurate in classifying different images. One of the con of using this model comes with a higher danger of overfitting.

Model 1 is more suited for scenarios where computational resources are limited or when the dataset does not contain highly intricate patterns. Model 2 is preferred for more complicated datasets where finer details must be captured, providing adequate computer resources and data to avoid overfitting. For our fruit and vegetable image classification project, we eventually selected Model 2 for predictions. While this model demands higher computational resources and poses a greater risk of overfitting, the challenges can be mitigated by incorporating dropout layers and closely monitoring during training. This model aids in maintaining precision in our classification

task despite the availability of significant processing resources. Alternatives such as the Vanilla Model, shown on Kaggle, present a unique opportunity for custom learning and adaptability. It achieved a notable validation accuracy of 97.62% in just 5 epochs. Despite its time efficiency compared to our CNN models, its method remains less familiar to us. Meanwhile, Model 2 performs at detailed feature detection that is appropriate for our dataset, but it takes longer training time.

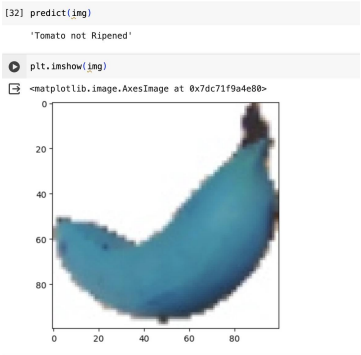
4. Implementation

We feed the training data into the 2 models defined previously, using 50 epochs each. For the implementation part, every time the model is fitted with the training data and validation data, we apply minor changes in hyperparameters to the model and try to improve the model.

The longest running time was the most difficult challenge we faced. The training process of the model takes forever to run. We experimented with a variety of epochs, including 20 and 50. The 20-epoch model did not perform as well as we thought. Then, we tried the 50-epoch model, it performed much better but took more than 2 hours to run. Considering the time cost, the 50-epoch model is the best model we could generate taking into account the tradeoff of running time. Moreover, we tried 3 models and finally decided to choose 2 of them. The model we eliminated had more features including the dropout rate, different activation functions, and more neurons. It seemed to be more sophisticated, but its overall performance did not witness a significant improvement, and the running time was too costly for the training.

Another challenge we encountered was the frustration caused by the “roller-coaster” journey. When we finally trained the model and checked the seemingly perfect training loss curve and accuracy curve, we were confident that the prediction result would not be out of

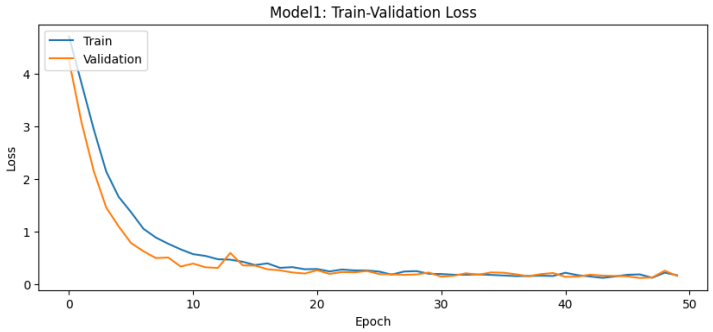
expectation. Then, as our first try, a "Tomato not Ripened" was predicted to be a **blue banana out of blue**, as shown to the right. The shock out of the blue posed pressure to us, especially after we had invested significant effort and thought into designing the model. In a word, the best lesson we learned from model training is to recharge and stay hopeful all the time. It is the key to success.

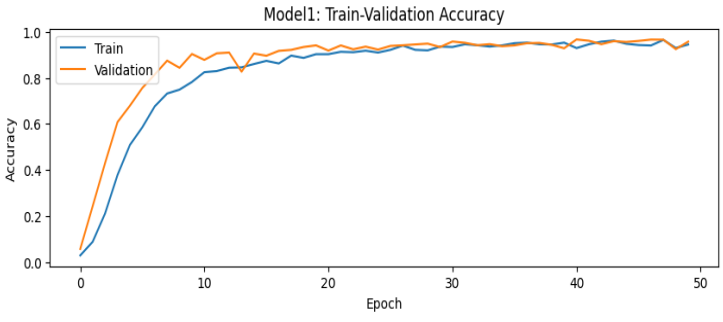


5. Results and Evaluation

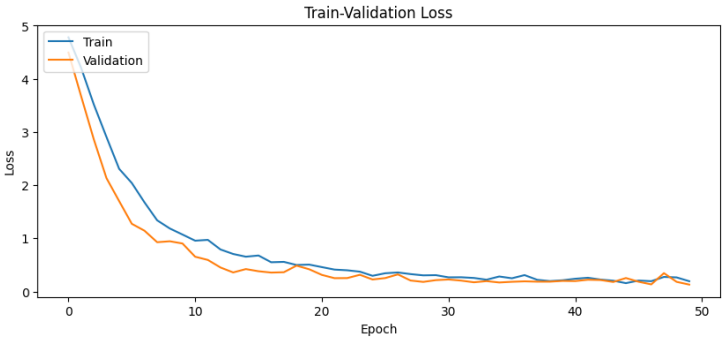
The two models (model 1 and model 2) were trained on the Fruits 360 dataset from kaggle with 131 classes and evaluated on a validation set. Visualizations, including training/validation loss and accuracy plots, were generated to understand model performance which are as follows:

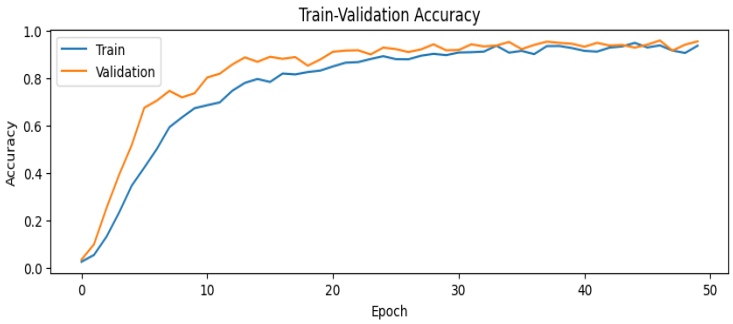
Model 1:





Model 2:





Results of deep learning can be evaluated in 3 common ways:

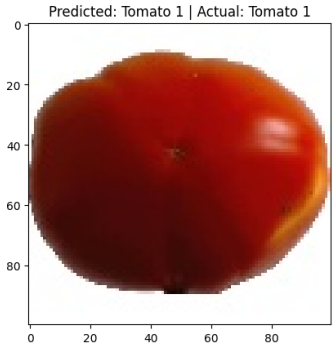
Yinan Chen, Tirth Pravin Gala, Yuhe (Tiffany) Jin, Chenjie (Angelina) Sun

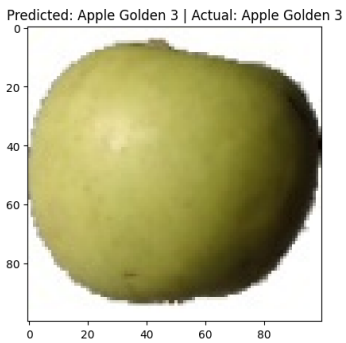
1. Evaluation Metrics:
    a. Loss function: It measures the dissimilarity between predicted and true labels. In our case model1 had the loss of 0.1533 and model2 had the loss of 0.1094. Thus, model2 was the better model as it made less mistakes.
    b. Accuracy: It indicates the proportion of correctly classified samples. In our case accuracy of model1 was 95.74% and model2 was 96.58%. Thus, model2 can be correctly used to classify different images of fruits in its respective categories where it is ~96% precise.
2. Visual Inspection:
    a. Image predictions were visually inspected to ensure alignment with actual fruit categories. Alongside, plots were studied as they visually reflected the learning process and potential overfitting.
3. Model Comparison
    a. Lastly, the two models were compared based on their hyperparameters, loss and accuracy.

	Convolutional Layer	Filter Size	Dense Layer	Dropout Rate	Learning Rate
Model 1	(32, 32, 64)	(3,3)	1024	0.5	0.001
Model 2	(64, 64, 128)				

	Loss	Accuracy
Model 1	0.1533	0.9574
Model 2	0.1094	0.9658

    b. Prediction Result based on Model 2:





Having a benchmark setup with a model implemented 3 years ago on kaggle, we found our model to perform better as accuracy was better. Benchmark model gave a loss of 0.153 and accuracy of 95.81%. The main point of difference between our model and the benchmark model was the optimizer and other hyperparameters of each layer. Below is the model architecture of his model showcasing the difference in his model to ours.

Benchmark model architecture:

```
model.add(Conv2D(32,(3,3),activation = 'relu', input_shape = shape_of_image.shape))
model.add(MaxPooling2D())

model.add(Conv2D(32,(3,3),activation = 'relu', input_shape = shape_of_image.shape))
model.add(MaxPooling2D())

model.add(Conv2D(64,(3,3),activation = 'relu', input_shape = shape_of_image.shape))
model.add(MaxPooling2D())
```

```
model.compile(loss = 'categorical_crossentropy',
              optimizer = 'rmsprop',
              metrics = ['accuracy'])
```

Our model architecture:

```
model2 = Sequential()

model2.add(Conv2D(64, (3, 3), input_shape=(100, 100, 3), activation='relu', padding='same'))
model2.add(MaxPooling2D(pool_size=(2, 2)))

model2.add(Conv2D(64, (3, 3), activation='relu', padding='same'))
model2.add(MaxPooling2D(pool_size=(2, 2)))

model2.add(Conv2D(128, (3, 3), activation='relu', padding='same'))
model2.add(MaxPooling2D(pool_size=(2, 2)))
```

```
optimizer = Adam(learning_rate=0.001)
model2.compile(optimizer = optimizer, loss='categorical_crossentropy',
               metrics=['accuracy'])
model2.summary()
```

Business case which can potentially use this model could be:

1. Integrating screens above product racks allows for an interactive and informative shopping experience. As customers pick up fruits or vegetables, the system identifies the item and displays relevant recipe videos on the screen.

2. As there are fruits recognized in the cart, recipes can be projected and alongside, associated other ingredients to enhance their recipes can be sold. This approach can lead to customers exploring and purchasing a variety of items, contributing to higher transaction values.

3. Businesses can implement this recognition model to various stores across its reach and gather data as to not only which fruit is sold the most in which area but also if a very fresh banana or a

quite ready banana is in demand in this region. This can help in managing inventory, forecasting geographic demand and helping overall scale up the business.

4. Production team can get to know through recognition of each fruit's image if it needs more time to get ripped or it is ready enough to be supplied.

5. Grocery store managers can enhance their visual merchandising strategy by studying individual customer's carts and make shopping experience more convenient by making fruits which are ripe and more in demand easy to pick than others.

6. SOP of offers and discounts can be optimized using fruit recognition models in cart or POS counters. Understanding historic trends based on time series models developed on this can help store managers know at which time of the year which combination of fruits are purchased together. Thus, offers can be optimized.

7. One important aspect to investigate using this model can be to recognise the fruits which are put in carts but not billed and then removed from carts. The online grocery department of grocery stores can use this model and build a business case as to understanding the fruits which are most frequently found in abandoned cart stages of purchase.

These are a few business cases we could think of. Further business understanding and knowledge will open the doors for more problem statements where this model can be useful.

6. Deployment

<u>It can be deployed in the business domains of:</u>

1. Production 2. Inventory Management 4. Customer Relationship Management 5. Assortment and Store Design Planning 6.Sales Operation 7. Procurement Strategy.

<u>Businesses that can use this model are:</u>

1. Grocery Retailers 2. Food & Beverage Companies 3. Online Groceries Platforms 4. Kitchen Appliance Companies 5. Health and Wellness Apps 6. Hospitality and Catering 7. Meal Kit Services.

Applying businesses in these particular domains are as follows:

1. Hardware Integration: Install cameras or sensors above product racks to capture real-time images of selected items. Connect screens or display units to showcase recipe videos.

2. Software Integration: Deploy the trained model on specific servers or edge devices. Implement real-time image recognition algorithms to identify fruits and vegetables.

3. User Interface: Develop a user-friendly interface for the screens to display recipe videos. Ensure seamless interaction for customers, allowing them to navigate through recipes.

4. Integration with Inventory System: Integrate the solution with the store's inventory system to ensure real-time availability of recommended products. Enable automatic updates based on stock levels and new arrivals.

These are some other ways we can ensure that the model helps business apart from what was discussed in the modeling part.

Some of the issues can be:

Our system prioritizes anonymizing data and avoids capturing sensitive information, such as inadvertently displaying images from a customer’s phone. We’re also prepared for technical glitches to address any issues with robust support. Recognizing that some customers may prefer traditional shopping, we aim for non-intrusive digital enhancements. We plan to monitor customer feedback closely and make necessary adjustments to strengthen greater user acceptance.

Basic ethical considerations are essential:

1. Transparency: Clearly communicate to customers that the system uses image recognition for recipe suggestions. Ensuring transparency in data collection and usage is essential.

2. Data Security: Implement strong data security measures to protect customer interactions and preferences. We should adhere to data protection regulations and guidelines.

3. Inclusivity: Ensure that the system is inclusive and accommodates customers with varying preferences and dietary restrictions. Avoid promoting content that may be offensive or culturally insensitive.

Few risks associated with the proposed plan can be:

Our system faces challenges with potential data bias, risking skewed recommendations, and a reliance on technology that could lead to issues during system failures.

Ways to mitigate these issues can be:

1. Continuous Monitoring: Implement continuous monitoring of the system's performance and customer feedback. Regularly update and refine the models based on new data and insights. Images of the new fruits can be added to the dataset.

2. Feedback Loop: Establish a feedback loop with customers to gather insights and address concerns through surveys. Actively seek input on the system's impact and make improvements accordingly.

3. Regular Audits: Conduct regular audits of the system's algorithms to identify and rectify any biases. Involve third-party experts for impartial evaluations and train your staff to complement their skills to the deployment of the model.

Appendix:

Team contributions:

Yinan Chen: Mainly working on the report

Tirth Pravin Gala: Mainly working on the report and PPT

Yuhe (Tiffany) Jin: Mainly working on the coding

Chenjie (Angelina) Sun: Mainly working on the report and PPT