SC4031 IoT Course Project

JESSICA DANIELLA GIRSANG
(U2221579J)

Fruit Classification App

1 Classify fruit (**10 classes**):

Apple
Banana
Pineapple
Kiwi
Orange
Avocado
Mango
Strawberries
Cherry
Watermelon

2 Image Input : User Gallery

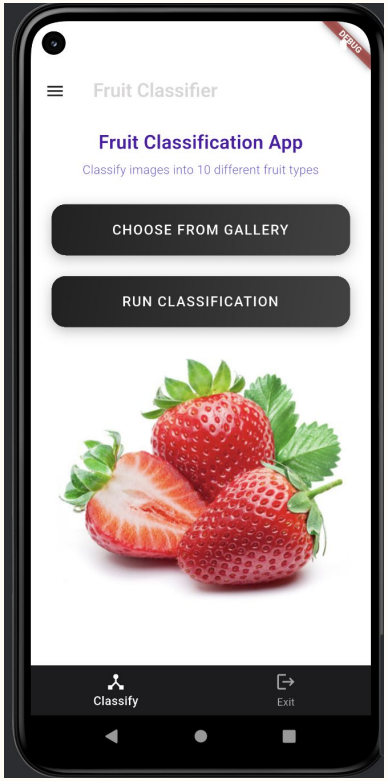
3 Functions: Quick fruit recognition (personal, dietary, etc.)

Architecture

User Input

Upload Image

DISPLAY RESULTS



HTTP POST request
(Send Image)

SERVER
Model inference

HTTP returns
prediction results

Model

DATASET: https://www.kaggle.com/datasets/karimabdulnabi/fruit-classification10-class

```
[*]: model.fit(train_ds, epochs=15, callbacks=callbacks, validation_data=val_ds)

Epoch 1/15
58/58 ━━━━━━━━━━━━━━━━━━━━ 0s 16s/step - accuracy: 0.2589 - loss: 1.9617
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
58/58 ━━━━━━━━━━━━━━━━━━━━ 969s 17s/step - accuracy: 0.2604 - loss: 1.9585 - val_accuracy: 0.0897 - val_loss: 2.3058
Epoch 2/15
58/58 ━━━━━━━━━━━━━━━━━━━━ 0s 16s/step - accuracy: 0.4466 - loss: 1.5672
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
58/58 ━━━━━━━━━━━━━━━━━━━━ 959s 16s/step - accuracy: 0.4471 - loss: 1.5655 - val_accuracy: 0.1050 - val_loss: 2.3255
Epoch 3/15
58/58 ━━━━━━━━━━━━━━━━━━━━ 0s 16s/step - accuracy: 0.4723 - loss: 1.4819
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
```

Epoch: 15+5

Optimizer: Adam

BatchSize = 32

APP DEMO

Local Inference

Marking Criteria (1)

- Collect user input (10)
  - **If** load input from storage (5) ✅
  - **ELSE** collect real-time input by touch screen, microphone, camera, built-in sensors (10)

Marking Criteria (2)

- Infer locally and display result (15)
  - **If** your app offloads the execution to cloud, (10) ✅
  - **ELSE** run heuristic algorithm (not neural network) on the mobile app (5)
  - Display the inference result in real time by screen or synthetic voice (5) ✅

Marking Criteria (3)

- Run on emulated/physical IoT device (15)
  - **If** the program runs natively on a desktop or laptop computer (5)
  - **ELSE** the program runs on an emulated or a physical IoT device (15) ✅
    - iOS emulator provided by Xcode
    - Android emulator
    - Real smartphone
    - Raspberry Pi + add-on sensors (camera, microphone, etc)

Cloud Inference

Marking Criteria (4)

- Run inference in cloud virtual machine (10)
  - **If** deploy server program on a cloud virtual machine, e.g., Azure. (10)
  - **ELSE** deploy server program on your own computer (10)

Marking Criteria (5)

- Communication btw IoT device and cloud (20)
  - Send the user input from the mobile app to the cloud for inference (10)
  - Send the inference result from the cloud to the mobile app (10)

Advanced Task

Marking Criteria (6)

- Train your own model (10)
  - **If** train the used ML model by yourself (10)
    - Training program should be in the code package
    - Training results (e.g., accuracy) in a readme file
    - The training data can be any publicly available dataset
  - **ELIF** use downloaded pre-trained model (7)
  - **ELSE** use heuristic algorithm (not neural network) (3)

Marking Criteria (7)

- Support multiple concurrent users (10)
  - Demonstrate multiple IoT devices can use the cloud service simultaneously

Thank You