Ai for AgriTech Hackathon – Python Code Files on Project.

CNN-Based Plant Disease Detection – Python Code

This example uses Keras + TensorFlow and works with any dataset structured like:

```
dataset/
├── train/
|   ├── healthy/
|   ├── bacterial_spot/
|   └── early_blight/
└── validation/
    ├── healthy/
    ├── bacterial_spot/
    └── early_blight/
```

Step 1: Import Required Libraries

```
import os
import numpy as np
import matplotlib.pyplot as plt
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout
from tensorflow.keras.optimizers import Adam
```

Step 2: Set Parameters and Paths

```
IMG_HEIGHT = 128

IMG_WIDTH = 128

BATCH_SIZE = 32

EPOCHS = 10


train_dir = "dataset/train"

val_dir = "dataset/validation"
```

Step 3: Data Preprocessing

```
train_datagen = ImageDataGenerator(rescale=1./255)

val_datagen = ImageDataGenerator(rescale=1./255)


train_data = train_datagen.flow_from_directory(
    train_dir,
    target_size=(IMG_HEIGHT, IMG_WIDTH),
    batch_size=BATCH_SIZE,
    class_mode='categorical'
)


val_data = val_datagen.flow_from_directory(
    val_dir,
    target_size=(IMG_HEIGHT, IMG_WIDTH),
    batch_size=BATCH_SIZE,
```

```
    class_mode='categorical'
)
```

Step 4: CNN Model Architecture

```
model = Sequential([
    Conv2D(32, (3,3), activation='relu', input_shape=(IMG_HEIGHT, IMG_WIDTH, 3)),
    MaxPooling2D(2,2),

    Conv2D(64, (3,3), activation='relu'),
    MaxPooling2D(2,2),

    Conv2D(128, (3,3), activation='relu'),
    MaxPooling2D(2,2),

    Flatten(),
    Dense(256, activation='relu'),
    Dropout(0.5),
    Dense(train_data.num_classes, activation='softmax')
])

model.compile(
    optimizer=Adam(learning_rate=0.0001),
    loss='categorical_crossentropy',
    metrics=['accuracy']
)
```

```
model.summary()
```

Step 5: Model Training

```
history = model.fit(
    train_data,
    epochs=EPOCHS,
    validation_data=val_data
)
```

Step 6: Save the Model

```
model.save("plant_disease_model.h5")
```

Step 7: Plot Accuracy & Loss Graphs

```
plt.plot(history.history['accuracy'], label='Training Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.legend()
plt.title('Accuracy')
plt.show()


plt.plot(history.history['loss'], label='Training Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.legend()
plt.title('Loss')
```

```
plt.show()
```