rice-disease-classifier

May 7, 2024

[1]:
```
import tkinter as tk
from tkinter import filedialog
from PIL import Image, ImageTk
import torch
import torchvision.transforms as transforms
import torchvision.models as models
import pickle
```

[2]:
```
# Define class labels
class_labels = ['Bacterial leaf blight', 'Brown spot', 'Leaf smut']  # Replace␣
 ↪with your actual class labels

# Define image transformations
transform = transforms.Compose([
    transforms.Resize(256),
    transforms.CenterCrop(224),
    transforms.ToTensor(),
    transforms.Normalize([0.485, 0.456, 0.406], [0.229, 0.224, 0.225])
])

# Load the saved AlexNet model
def load_model():
    global model
    with open("rice_disease_model_final.pkl", 'rb') as f:
        model = pickle.load(f)
    model.eval()

def save_model():
    with open("rice_disease_model_final.pkl", 'wb') as f:
        pickle.dump(model, f)

load_model()  # Load the model when the application starts

# Function to classify image
def classify_image(image_path):
    image = Image.open(image_path)
    image = transform(image).unsqueeze(0)
```

```
    with torch.no_grad():
        outputs = model(image)
        _, predicted = torch.max(outputs, 1)
    return class_labels[predicted.item()]

# Function to open file dialog and classify selected image
def classify_selected_image():
    file_path = filedialog.askopenfilename()
    if file_path:
        predicted_class = classify_image(file_path)
        result_label.config(text="Predicted class: " + predicted_class)

# Create the Tkinter window
root = tk.Tk()
root.title("Image Classifier")

# Create a button to select an image
select_button = tk.Button(root, text="Select Image",␣
 ↪command=classify_selected_image)
select_button.pack(pady=10)

# Create a button to save the model
save_button = tk.Button(root, text="Save Model", command=save_model)
save_button.pack(pady=5)

# Create a label to display the result
result_label = tk.Label(root, text="")
result_label.pack(pady=10)

# Run the Tkinter event loop
root.mainloop()
```