**OCR Application with EasyOCR, PaddleOCR, and Gradio**

**Overview**

This script uses EasyOCR and PaddleOCR to extract text from images, providing a simple Gradio interface for comparison. Users can upload images and view extracted text with a basic accuracy estimate.

**Features**

* **Dual OCR Models**: Extract text using EasyOCR and PaddleOCR.
* **Interactive UI**: Gradio-based interface for real-time image uploads.
* **Error Handling**: Displays error messages for OCR failures.

**Setup**

1. **Install Dependencies**:
2. pip install gradio easyocr paddleocr Pillow numpy
3. **Run the Script**:
4. python script\_name.py
5. Access the Gradio interface via browser or shareable link.

**Inputs and Outputs**

**Inputs:**

* An image (PNG/JPG).

**Outputs:**

1. EasyOCR and PaddleOCR extracted text.
2. Basic length-based accuracy estimates.

**Future Enhancements**

* Add multi-language support.
* Improve accuracy calculation using advanced metrics.

This application provides an intuitive way to compare OCR models and visualize results in real time.

Code

import gradio as gr

import easyocr

from paddleocr import PaddleOCR

from PIL import Image

import numpy as np

# Initialize OCR models

easyocr\_reader = easyocr.Reader(['en'])

paddleocr\_reader = PaddleOCR(use\_angle\_cls=True, lang='en')

# Function to calculate accuracy (for demo, using length of extracted text vs. expected text length)

def calculate\_accuracy(extracted\_text, expected\_text):

    if not expected\_text:

        return 0

    common\_words = len(set(extracted\_text.split()).intersection(set(expected\_text.split())))

    total\_words = len(set(expected\_text.split()))

    return (common\_words / total\_words) \* 100 if total\_words else 0

# OCR function using EasyOCR and PaddleOCR with error handling

def ocr\_function(image: Image):

    img = np.array(image)

    try:

        # EasyOCR

        easyocr\_result = easyocr\_reader.readtext(img)

        easyocr\_text = " ".join([result[1] for result in easyocr\_result])

        easyocr\_accuracy = f"{len(easyocr\_text) / 10}%"

    except Exception as e:

        easyocr\_text = f"Error: {str(e)}"

        easyocr\_accuracy = "Error"

    try:

        # PaddleOCR

        paddleocr\_result = paddleocr\_reader.ocr(img, cls=True)

        # Fixing the handling of PaddleOCR output

        paddleocr\_text = " ".join([line[1][0] for line in paddleocr\_result[0]])  # Extracting text from tuples

        paddleocr\_accuracy = f"{len(paddleocr\_text) / 10}%"

    except Exception as e:

        paddleocr\_text = f"Error: {str(e)}"

        paddleocr\_accuracy = "Error"

    # Returning the output

    return (easyocr\_text, easyocr\_accuracy, paddleocr\_text, paddleocr\_accuracy)

# Gradio Interface

interface = gr.Interface(

    fn=ocr\_function,

    inputs=gr.Image(type="pil"),

    outputs=[

        gr.Textbox(label="EasyOCR Result"),

        gr.Textbox(label="EasyOCR Accuracy"),

        gr.Textbox(label="PaddleOCR Result"),

        gr.Textbox(label="PaddleOCR Accuracy"),

    ],

    live=True,

    allow\_flagging="never"

)

# Launch Gradio interface in the browser

interface.launch(share=True)