

Report

Image-to-Text Conversion and Text Correction System Report

Introduction

In this report, we present a project aimed at extracting text from images using PaddleOCR and performing spell checking using the PySpellChecker library. Additionally, we have developed a web application using Streamlit to provide a user-friendly interface for utilizing these functionalities.

Project Overview

Our project encompasses the following main components:

1. Image Preprocessing for best text extraction.
2. Text extraction from images using PaddleOCR.
3. Spell checking using PySpellChecker.
4. Development of a web application using Streamlit.

Methodology

Image Preprocessing

enhances the contrast of an image by scaling its pixel values to span the full range of intensities. It then applies erosion, a morphological operation and smooth the image slightly.

Text Extraction with PaddleOCR

EasyOCR is a Python library that allows for easy and efficient text extraction from images. We utilize its capabilities to extract text from images uploaded by the user through the web application.

Spell Checking with PySpellChecker

PySpellChecker is a Python library that provides efficient spell checking functionality. After extracting text from images, we pass it through PySpellChecker to identify and correct any spelling errors present in the text.

Web Application Development with Streamlit

Streamlit is a popular framework for building web applications with Python. We leverage Streamlit to create a user-friendly interface where users can upload images, extract text, perform spell checking, and view the corrected text.

Implementation Details

Required Libraries:

```
#pip install paddleocr
#pip install scikit-image
#pip install pyspellchecker
```

Import Libraries:

```
from PIL import Image
from skimage import io as ioski
from skimage.color import rgb2gray
from matplotlib import pyplot as plt
import cv2
from spellchecker import SpellChecker
from paddleocr import PaddleOCR, draw_ocr
# ocr= PaddleOCR(use_angle_cls=True, lang ='en')
```

Image-Preprocessing

```

pxmin = np.min(img)
pxmax = np.max(img)
imgContrast = (img - pxmin) / (pxmax - pxmin)* 100

# Convert pixel values back to uint8
imgContrast = np.uint8(imgContrast)

# Morphological operation: erosion
kernel = np.ones((3, 3), np.uint8)
imgMorph = cv2.erode(imgContrast, kernel, iterations=1)

# Display the processed image
plt.imshow(imgMorph, cmap='gray')
plt.show()

```

Text Extraction

```

# input image in paddleocr
result = ocr.ocr(img , cls=True)

#extract text from that.
tuples_list = []
for sublist1 in result:
    for sublist2 in sublist1:
        tuples_list.append(sublist2[1])

# Print the extracted tuples
string=""
for i in tuples_list:
    string += i[0] +" "

```

Spell Checking

```

# Initialize SpellChecker
spell = SpellChecker()

# Split text into lines
lines = string.strip().split('\n')

# Initialize variables
new_lines = []

# Iterate through each line
for line in lines:
    # Split each line into words
    words = line.split()
    new_line = ""
    # Iterate through each word
    for word in words:
        # Get the correction for the word
        correction = spell.correction(word)
        # Check if the correction is not None
        if correction is not None:
            new_line += correction + " "
        else:
            # If no correction is found, use the original word
            new_line += word + " "
    # Append the corrected line to the list of new lines
    new_lines.append(new_line.strip())

# Join the corrected lines into a single string
corrected_text = '\n'.join(new_lines)

# Print the corrected text

```

```
print(corrected_text)
```

Streamlit Web Application

```
from PIL import Image
from skimage.color import rgb2gray
from matplotlib import pyplot as plt
import cv2
import io
import numpy as np
import os
import streamlit as st
from spellchecker import SpellChecker
from paddleocr import PaddleOCR

ocr= PaddleOCR(use_angle_cls=True, lang='en')

def main():
    # Title and description
    st.title("Image to Text with Text Correction")
    st.write("Upload an image and get the text with spelling
correction!")

    # Define directory to store uploaded images
    upload_dir = "uploaded_images"
    os.makedirs(upload_dir, exist_ok=True)

    # File uploader
    uploaded_file = st.file_uploader("Choose an image...", ty
pe=["jpg", "png"])

    if uploaded_file is not None:
        # Store the uploaded file in the specified directory
        image_path = os.path.join(upload_dir, uploaded_file.n
```

```

ame)
    with open(image_path, "wb") as f:
        f.write(uploaded_file.getvalue())

    # Display uploaded image
    image = Image.open(uploaded_file)
    st.image(image, caption="Uploaded Image", use_column_
width=True)

# Display full path of the uploaded image
full_image_path = os.path.abspath(image_path)
img = cv2.imread(full_image_path, cv2.IMREAD_GRAYSCALE)

E)

# Increase contrast
pxmin = np.min(img)
pxmax = np.max(img)
imgContrast = (img - pxmin) / (pxmax - pxmin) * 100

# Convert pixel values back to uint8
imgContrast = np.uint8(imgContrast)

# Morphological operation: erosion
kernel = np.ones((3, 3), np.uint8)
imgMorph = cv2.erode(imgContrast, kernel, iterations=
1)

result = ocr.ocr(img, cls=True)

```

```

tuples_list = []
for sublist1 in result:
    for sublist2 in sublist1:
        tuples_list.append(sublist2[1])

# Print the extracted tuples
string = ""
for i in tuples_list:
    string += i[0] + " "

st.title("Extracted Text from the image: ")
st.code(string)

```

```

# Initialize SpellChecker
spell = SpellChecker()

# Split text into lines
lines = string.strip().split('\n')

# Initialize variables
new_lines = []

# Iterate through each line
for line in lines:
    # Split each line into words
    words = line.split()
    new_line = ""
    # Iterate through each word
    for word in words:
        # Get the correction for the word

```

```

        correction = spell.correction(word)
        # Check if the correction is not None
        if correction is not None:
            new_line += correction + " "
        else:
            # If no correction is found, use the original word
            new_line += word + " "
        # Append the corrected line to the list of new lines
        new_lines.append(new_line.strip())

    corrected_text = '\n'.join(new_lines)

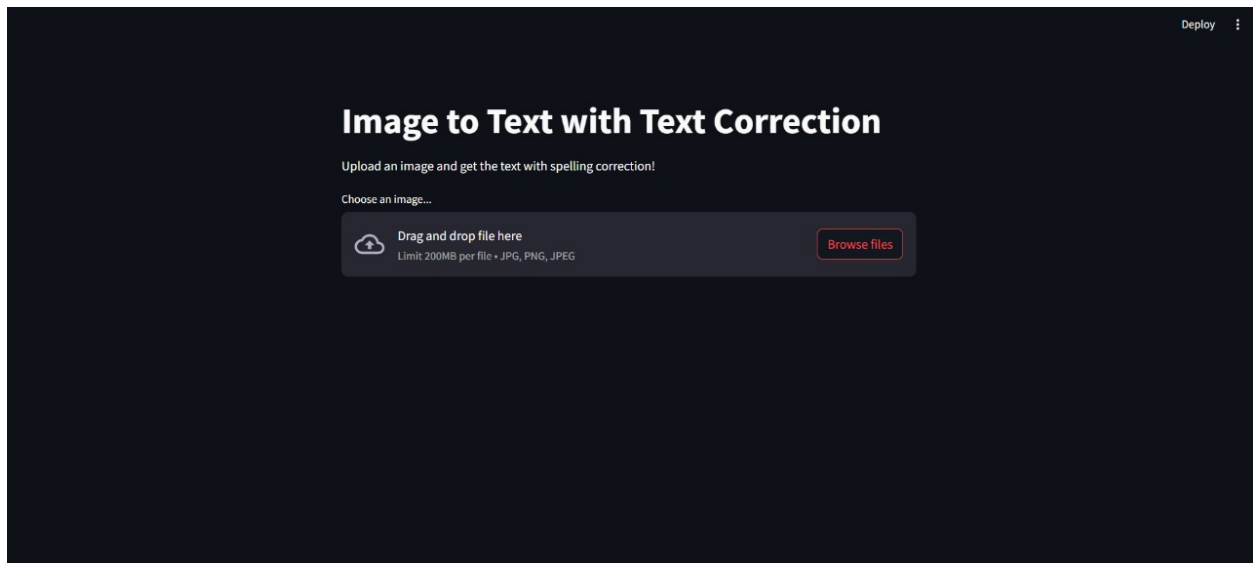
    st.title("Extracted Text from the image: ")
    st.code(corrected_text)

if __name__ == "__main__":
    main()

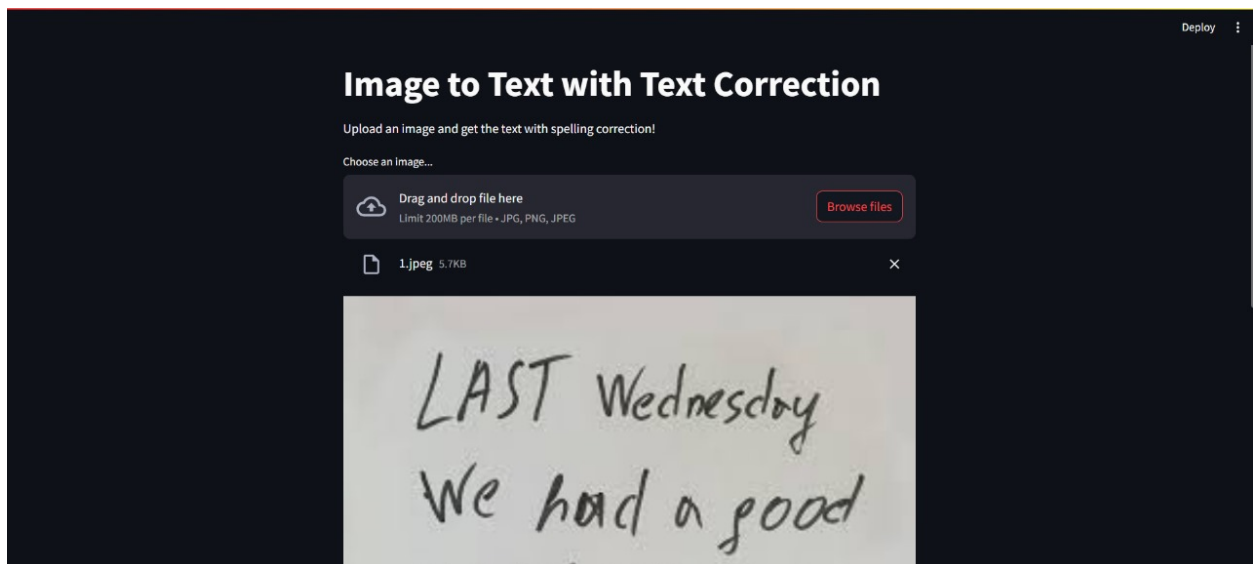
```

Results

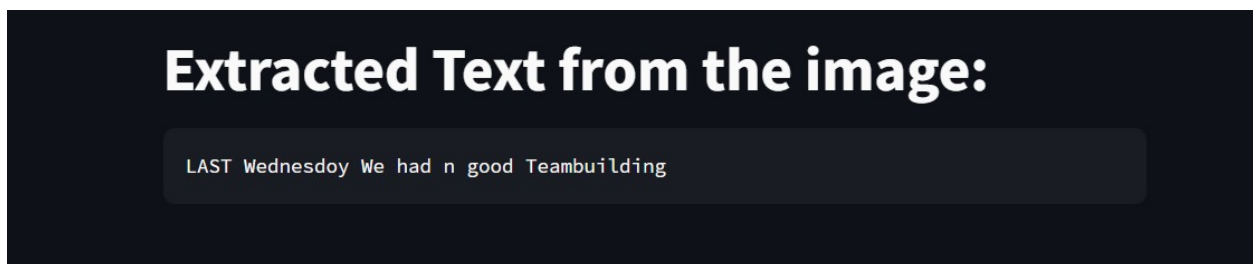
Web Application Interface



Uploading Image



Text Extraction Output



Spell Checking Output

Spelling Corrected Text after extraction:

LAST Wednesday We had n good Teambuilding

Web Application Screenshots

Image to Text with Text Correction

Upload an image and get the text with spelling correction!

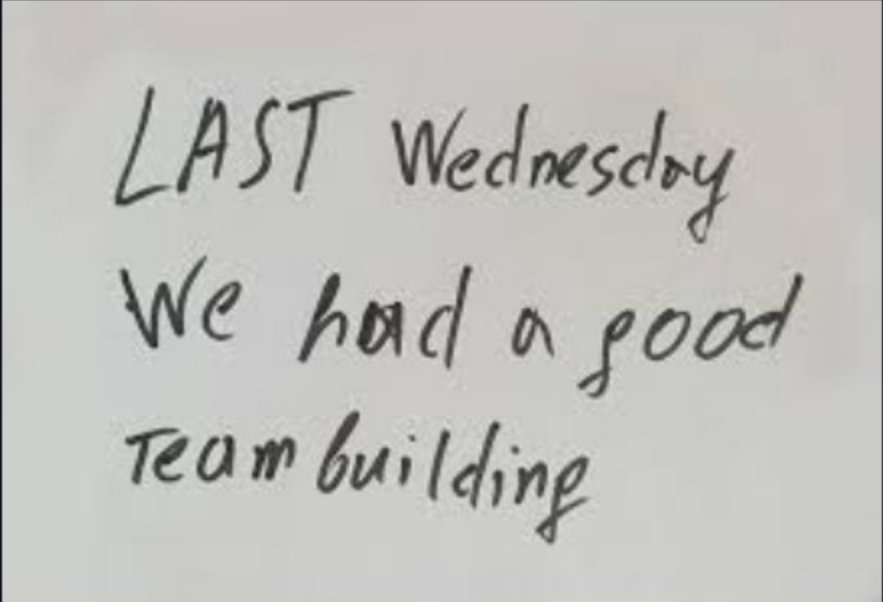
Choose an image...

Drag and drop file here
Limit 200MB per file • JPG, PNG, JPEG

Browse files

1.jpeg 5.7KB

X



Uploaded Image

Extracted Text from the image:

LAST Wednesday We had n good Teambuilding

Spelling Corrected Text after extraction:

LAST Wednesday We had n good Teambuilding

Conclusion

Our project successfully demonstrates the integration of PaddleOCR for text extraction, PySpellChecker for spell checking, and Streamlit for web application development. The combination of these technologies provides a powerful tool for extracting and correcting text from images in a user-friendly manner.