U ▪
S T

Python Developer – Initial Screening for UST (SEMICON) – Oct 2023

Problem Statement

1.OCR

Take a look at the photos in

https://www.google.com/search?q=Food+nutrition+labels&tbm=isch&safe=active&hl=en-US&sa=X&ved=2ahUKEwiJhZnIv6mAAxVp7DgGHVJ8AcQQBXoECAEQZQ&biw=1739&bih=894

or take your own photos from food packets at home. Then list down all the possible nutrients available combined for all the photos and create a database.

a. Use OCR algorithms to detect only the nutrient information from the images.

b. Test your method for a new image from net and verify.

c. Think about and apply corrections to the OCR predictions using the combined list of all possible nutrients (This is most important to address)

Description of project

In this project I have developed a solution for detecting only the nutrient information from the images. I have used OCR algorithm to list down all the possible nutrients available combined for all the photos.

Requirements

- Tools
  - Python
  - MongoDB
  - MongoDB Compass
  - PyCharm
  - Tesseract
  - Git, GitHub
- **Libraries**
  - pymongo
  - pytesseract
  - open-cv
  - regex
  - pillow

U ▪
S T

Input: Take photos from food packets

Output: List down all the possible nutrients available and store in database.

Task Lists

Step1: Installed all requirements required for this project.

Step2: Created a database and collection using MongoDB.

Step3: Take images from the link.

Step4: Used PyCharm to develop code for this project.

Step5: I have tried two libraries:

- Using OpenCV (Open-Source Computer Vision Library)
- Using Pillow (Python Imaging Library)

I tested out two libraries. I ultimately decided on OpenCV.

Step6: Created a list of possible nutrient keywords. The list contains a collection of keywords that are commonly associated with nutrients and nutritional information found in food products.

Step7: Created a regular expression pattern using the re module. This pattern is intended for extracting nutrient information from a text string based on list of possible nutrient information.

Step8: Converted the input image from a colored image to a grayscale image. Grayscale images contain only shades of gray and are often used in image processing and computer vision tasks because they simplify the image while retaining important information like intensity.

Step9: Created a function which is intended to take a nutrient name as input and check if it closely matches any of the predefined possible nutrient keywords. If it finds a close match, it returns the best-matching keyword as the corrected nutrient name; otherwise, it returns the original nutrient name. This can be useful for data standardization or validation in applications related to nutrition or food data processing.

Step10: Injected data into the database.

Step11: I have pushed the code to git.

Step12: To find the solution, I have referred YouTube videos, ChatGPT and Google.

Step13: Detected the nutrient information only from the images.

U ▪
S T

Optical Character Recognition (OCR)

Optical Character Recognition (OCR) can be performed using different methods and technologies, depending on the specific requirements and characteristics of the documents or images being processed. Optical character recognition is a technology that converts typed or handwritten text and printed images containing text into machine-readable digital data format.

Here's a simplified overview of how OCR works:

- Image Acquisition: OCR begins with the acquisition of an image or document that contains text. This can be a scanned document, a photograph of text, a screenshot, or any visual representation of text.
- Preprocessing: Before performing OCR, the image is preprocessed to improve its quality and enhance the chances of accurate text extraction. Preprocessing steps may include:
    - Image enhancement: Adjusting contrast, brightness, and sharpness.
    - Noise reduction: Removing unwanted artifacts or speckles.
    - Binarization: Converting the image to black and white for better character separation.
- Text Detection: In some OCR systems, a text detection step is performed to identify regions of the image that likely contain text. This step helps reduce the area that needs to be processed and improves efficiency.
- Character Segmentation: OCR algorithms analyze the image to identify individual characters or symbols.
- Postprocessing: After recognizing individual characters, OCR software may apply postprocessing techniques to improve the overall accuracy and coherency of the recognized text. This can involve:
    - Spell checking: Correcting recognized words based on a dictionary.
    - Data validation: Ensuring that the recognized text conforms to expected formats.
- Output: The final output of the OCR process is a machine-readable text document.

Now, let's address why Tesseract OCR is a popular choice:

Tesseract is an open-source optical character recognition (OCR) engine developed by Google. OCR technology is used to recognize and extract text from images or scanned documents, essentially converting images of text into machine-readable text.

- **Open Source:** Tesseract OCR is open-source, meaning it is freely available for use and can be modified and extended by the community.
- **Accuracy:** Tesseract OCR has improved significantly in terms of accuracy over the years.

- **Language Support:** It supports recognition of text in multiple languages, with support for over 100 languages.
- **Cross-Platform:** Tesseract is designed to work on multiple platforms, including Windows, macOS, Linux, and mobile operating systems like Android and iOS.

Overall, Tesseract OCR is a powerful OCR engine that has gained popularity due to its accuracy, language support, and open-source nature, making it suitable for a wide range of text recognition applications. **pytesseract** is a Python library that provides an interface to the Tesseract Optical Character Recognition (OCR) engine. **pytesseract** allows you to perform OCR on images and extract text content from them.

OpenCV (Open Source Computer Vision Library)

OpenCV is primarily designed for computer vision tasks, including image and video processing, object detection, and feature extraction. While OpenCV is not an OCR engine, it can be used for preprocessing and enhancing images before OCR. For example, you can use OpenCV to adjust image contrast, perform noise reduction to improve OCR accuracy.

- **Strengths:** OpenCV is powerful for computer vision tasks and offers a wide range of image processing functions

Pillow (Python Imaging Library):

Pillow is a library specifically designed for working with images in Python. Its main focus is on image file format support, image manipulation, and basic image processing. Pillow can be used for opening, saving, resizing, cropping, and drawing on images. It is valuable for loading and preprocessing images before OCR.

- **Strengths:** Ease of use and extensive image file format support.

When it comes to OCR, both libraries can play complementary roles. OpenCV can be used for preprocessing and enhancing images to improve OCR accuracy, while Pillow can be used for image loading, resizing, and basic manipulation tasks. The choice between them depends on your specific OCR requirements and the tasks you need to perform before feeding images into an OCR engine. There is no "better" library overall; it depends on the context and use case.