Team Name: snoopygyangg

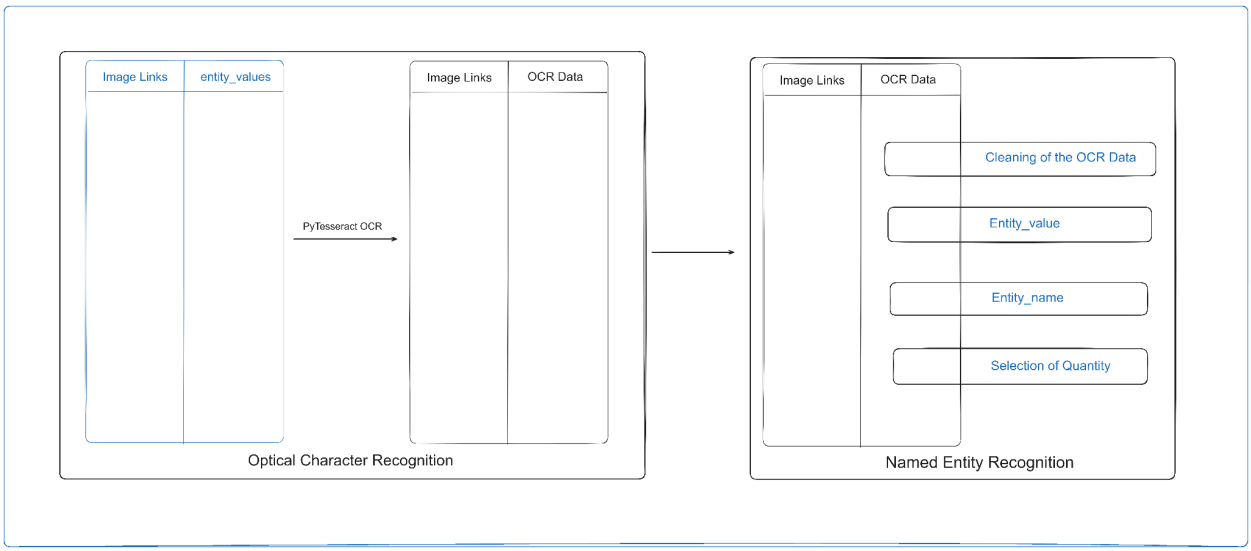
College: XIM University, Bhubaneswar

Overview

Our prediction process involved two key steps:

1. **Image-to-Text Conversion**: We first extracted textual data from the provided test images using Optical Character Recognition (OCR) technology.
2. **Information Extraction with NER:** The extracted text was then cleaned and processed using Named Entity Recognition (NER) techniques. This allowed us to identify and extract the specific information we needed, ensuring a more accurate and targeted prediction.

Flowchart of Processes:



Explanation

Part 1: Image Text Extraction (Using pytesseract)

1. **Import Libraries:** The code starts by importing necessary libraries for making HTTP requests (requests), handling images (PIL), OCR (pytesseract), and data manipulation (pandas).
2. **Load DataFrame:** It loads a CSV file (test.csv) into a pandas DataFrame. This DataFrame likely contains image URLs and other relevant information.
3. **extract_text_from_image Function:** This function takes an image URL as input and performs the following steps:
    - **Download Image:** Sends a GET request to the given URL to download the image.

    - **Open Image:** Opens the downloaded image using PIL.Image.open.
    - **OCR:** Uses pytesseract.image_to_string to perform Optical Character Recognition (OCR) on the image, extracting text from it.
    - **Error Handling:** Includes a try-except block to catch any errors during the process and return an error message.
4.
5. **Apply OCR:** The code applies the extract_text_from_image function to the 'image_link' column of the DataFrame using df['image_link'].apply(...). The extracted text for each image is stored in a new column called 'tesseract'.
6. **Save DataFrame:** The updated DataFrame (with extracted text) is saved to a new CSV file (csv/101000.csv).

Part 2: Entity Extraction and Unit Standardization

1. **Load and Prepare Data:** The code loads a new CSV file (potentially the one created in Part 1) into a DataFrame. It also defines lists for different unit types (length, weight, volume, power).
2. **standardize_unit Function:** This function takes a numerical value and a unit as input. It then attempts to standardize the unit to a predefined format (e.g., "cm" to "centimetre"). This ensures consistent representation of units.
3. **process_text Function:** This is the core function that processes each row of the DataFrame:
    - **Text Lowercasing:** Converts the extracted text to lowercase for case-insensitive matching.
    - **Entity-Specific Logic:** Uses conditional statements (if-elif-else) to apply different extraction logic based on the 'entity_name' column (e.g., 'depth', 'width', 'item_weight').
    - **Regular Expressions:** Utilizes regular expressions (re.search) to find numerical values followed by specific units within the extracted text.
    - **Unit Standardization:** Calls the standardize_unit function to ensure consistent unit representation.
    - **Print Extracted Values:** Prints the extracted value and image link for debugging and progress monitoring.
4.
5. **Apply Processing:** The code applies the process_text function to each row of the DataFrame, storing the extracted and standardized values in the 'extracted_value' column.
6. **Entity-Unit Mapping:** The code creates a dictionary called entity_unit_map to store the unique units found for each entity type. This information can be helpful for analysis or validation.
7. **Save Results:** Finally, the processed DataFrame with the extracted values and standardized units is saved to a new CSV file (extracted_results_from_combined.csv).

Overall, this code automates the process of extracting specific information from images and standardizes the units for easier analysis. The use of regular expressions and entity-specific logic makes the extraction process more robust and accurate.

Summary

We developed a two-step solution for accurate information extraction from product images.

First, we employed Optical Character Recognition (OCR) using the Pytesseract library to convert image data into machine-readable text. This involved downloading images from provided URLs, processing them, and extracting textual content.

Second, we implemented a rule-based Named Entity Recognition (NER) system. This involved:

- **Text Preprocessing:** Cleaning and preparing the extracted text for analysis.
- **Entity-Specific Rules:** Creating tailored logic and regular expressions to extract numerical values and units related to specific entities like depth, width, weight, and volume.
- **Unit Standardization:** Standardizing extracted units (e.g., "cm" to "centimeter") to ensure consistency.

This automated process allowed us to extract targeted information from images, standardize units, and store the results, enhancing the efficiency and accuracy of product information extraction.