Team_Ctrl+Alt+MAAL

This script processes a CSV file containing image URLs to extract text from the images using Optical Character Recognition (OCR). The extracted text is then saved to a new CSV file.

Dependencies

- `pandas`: For reading and writing CSV files.
- `PaddleOCR`: For performing OCR on images.
- `PIL (Pillow)`: For handling image files.
- `requests`: For downloading images from URLs.
- `numpy`: For manipulating image data as arrays.
- `tqdm`: For displaying a progress bar during processing (optional).

Initialization

- **PaddleOCR Model:**
  - The OCR model is initialized with English language support, angle correction, and GPU acceleration.

Functions

`load_image_from_url(url)`

- **Purpose:**
  - Downloads an image from the specified URL, converts it to RGB format, and then converts it into a NumPy array.
- **Parameters:**
  - `url` (str): The URL of the image to be downloaded.
- **Returns:**
  - A NumPy array representing the image.

`extract_text_from_image_url(image_url)`

- **Purpose:**
  - Extracts text from an image at the specified URL using the PaddleOCR model.
- **Parameters:**
  - `image_url` (str): The URL of the image from which text needs to be extracted.
- **Returns:**

    - A string containing the extracted text from the image.

`process_csv(input_csv_path, output_csv_path)`

- **Purpose:**
    - Processes a CSV file where each row contains an image URL. Extracts text from each image and writes the results to a new CSV file.
- **Parameters:**
    - `input_csv_path` (str): The path to the input CSV file containing image URLs.
    - `output_csv_path` (str): The path where the output CSV file with extracted text will be saved.
- **Functionality:**
    - Reads the input CSV file into a pandas DataFrame.
    - Iterates over each row, extracts text from the image URL using the `extract_text_from_image_url` function, and stores the result.
    - Saves the DataFrame with the extracted text to a new CSV file.

Usage

To use the script, specify the paths to the input and output CSV files and call the `process_csv` function. For example:

python
Copy code

```
input_csv = '/path/to/input.csv'
output_csv = '/path/to/output.csv'
process_csv(input_csv, output_csv)
```

Error Handling

- If there is an error processing an image URL (e.g., due to connectivity issues or invalid URLs), an error message is printed, and the extracted text is left blank for that row.

Output

- The output CSV file will contain the original data along with a new column for the extracted text from the images.

From here onwardss

extract_entity_value(entity_name, extracted_text) function:

Purpose:

This function extracts a specific entity's numeric value (like width, weight, voltage, etc.) and its unit from a block of text.

How it works:

1. **Text Preprocessing:**
    - The extracted_text is converted to lowercase for uniformity.
    - Commas in numbers (e.g., "4,3") are replaced with periods to ensure consistency in numeric formats.
2. **Unit Mapping:**
    - A dictionary (unit_map) is defined to map various abbreviations or variations of units (e.g., "mm" for millimeter, "g" for gram) to their standardized forms (e.g., "millimetre", "gram").
3. **Regular Expression Patterns:**
    - A dictionary of regex patterns (patterns) is used to extract numeric values along with their associated units based on the entity type (e.g., width, height, weight).
    - Each entity type (like width, depth, or wattage) has a specific regex pattern that matches a number followed by a unit.
4. **Matching Logic:**
    - The function selects the appropriate regex pattern based on the entity_name and uses re.findall() to search the extracted_text for matches.
    - If matches are found, they are sorted in descending order based on the numeric value, and the largest value is selected.
    - The corresponding unit is standardized using the unit_map, and the function returns the extracted value along with the unit.
5. **Special Cases:**
    - For wattage, if a direct match is not found, the function calculates wattage using the formula voltage * current, provided both voltage and current are found in the text.
    - If wattage is not found, but voltage is requested, or vice versa, the function attempts to retrieve the corresponding value based on the available data.
6. **Return Value:**
    - If a match is found, the function returns the largest value along with the standardized unit.
    - If no match is found, it returns an empty string.

`process_csv(input_file_path, output_file_path)` function:

Purpose:

This function processes a CSV file to extract entity values for each row and writes the results to a new CSV file.

How it works:

1. **Reading the Input CSV:**
   - The function reads the CSV file from `input_file_path` using `csv.DictReader()`, which allows each row to be accessed as a dictionary.
2. **Processing Each Row:**
   - For each row, the function retrieves the `entity_name` (the type of entity to extract, like width or weight) and the `extracted_text` (the block of text to search).
   - It calls the `extract_entity_value()` function to get the numeric value and unit for the entity.
   - The extracted value is then stored in a list of results along with the row's `index` and `entity_name`.
3. **Writing the Output CSV:**
   - The results are written to a new CSV file at `output_file_path`, with columns `index`, `entity_name`, and `extracted_value`.
   - The function prints a message indicating that the results have been saved.
4. **Sample Output:**
   - After processing, the first five results are printed to the console as a sample.

Usage:

The script reads data from the input CSV file at `/kaggle/input/test-pred/updated_with_new_predictions.csv`, processes it to extract numeric values based on the entity name, and saves the results to `output7.csv`.