Documentation: Efficient Data Processing with PaddleOCR for Item Label Extraction

Team Name - AIKENS

1. Introduction

In this project, we aimed to extract key details such as weight, volume, wattage, and dimensions from product information labels using Optical Character Recognition (OCR). The primary goal was to automate this extraction process across a large dataset of images (131,000 images) with a focus on identifying numeric values and their corresponding units.

OCR Tools Comparison

We experimented with various OCR tools before finalizing on PaddleOCR due to its superior accuracy and performance. Here's a summary of our testing:

1. Tesseract with PyTesseract:
   We initially tried Tesseract, but it yielded very poor results, with significant inaccuracies in text extraction and its inability to process some images correctly.
2. EasyOCR:
   EasyOCR provided better results compared to Tesseract but posed compatibility issues when deployed across different systems. The tool worked well on some platforms but was challenging to set up uniformly, which led us to seek a more adaptable solution.
3. PaddleOCR:
   After comparing multiple tools, PaddleOCR stood out due to its ease of use, compatibility, and ability to extract text more accurately across different images. It was particularly effective in handling multi-lingual and rotated text, which improved our extraction process.

2. Methodology

Approach to Text and Number Extraction

The primary strategy for extracting relevant numbers and units involved:

1. **Identifying Relevant Units:**
   We first used PaddleOCR to extract all text from the images. From the extracted text, our focus was on identifying units (e.g., "kg," "cm," "W") based on the expected entity name.
2. **Locating Associated Numbers:**
   After identifying a unit, the program searched for the closest numeric value using the coordinates of the extracted text. The search was conducted based on proximity, with a priority given to numbers on the left and top of the unit, and then checking the right and bottom if no number was found. To measure this distance, we calculated the Euclidean distance between the coordinates of text blocks.
3. **Optimizing String Parsing:**
   Using advanced regular expressions (regex) and string transformations, we were able to refine the extraction process, handling edge cases such as numbers embedded within text (e.g., "3.5kg") or separated by special characters.

Machine Learning and Model Choice

Although PaddleOCR is not an ML model in itself, the combination of text detection and our custom parsing logic allowed us to approach the problem from a machine learning perspective. The model's ability to recognize characters in various orientations and styles made it a versatile choice for processing diverse sets of product labels.

3. Challenges and Solutions

1. Large Dataset Processing

Problem:

Processing 131,000 images in a single pass was computationally infeasible, given our resource constraints. Memory limitations caused system crashes when attempting to load the entire dataset.

Solution: Batch Processing

We implemented batch processing to break the dataset into manageable groups of 500 images. This approach reduced memory strain and allowed us to process images incrementally, saving progress at regular intervals. After each batch, results were written to disk to avoid losing data in case of a system failure.

- Batch Size Optimization:
  Through experimentation with different batch sizes (100, 200, 500), we found that 500 images per batch provided the best balance between processing time and resource usage.

2. Memory Management

Problem:

Running operations on large image datasets without clearing memory between batches led to memory leaks and system instability.

Solution: Efficient Memory Clearance

By ensuring that data structures were cleared after every batch, we prevented memory overloads, allowing the system to continue processing large datasets over extended periods without crashing. Additionally, batch-wise saving of results mitigated the risk of data loss.

3. Accuracy vs. Performance Trade-off

Problem:

Initial OCR extraction results were inaccurate, especially when dealing with noisy images or poor-quality labels, yielding only about 40% accuracy.

Solution: Regex Refinements and Adaptive Techniques

We refined our regular expressions and string parsing techniques, improving the accuracy of the extracted data to around 60-70%. We focused on:

- Handling various text formats by modifying regex patterns to identify mixed alphanumeric strings.
- Using distance-based searches to find numbers closest to units.

4. Conclusion

By combining batch processing, effective memory management, and iterative improvements to our parsing logic, we were able to efficiently process a large dataset of product labels with an accuracy of 60-70%. The key to success was:

- Selecting the right OCR tool (PaddleOCR) after thorough experimentation.
- Using batch processing to handle large datasets.
- Adapting regex and parsing techniques to improve the system's accuracy.

This approach allowed us to create a scalable solution that could extract critical information from product labels, balancing resource constraints with accurate text extraction.