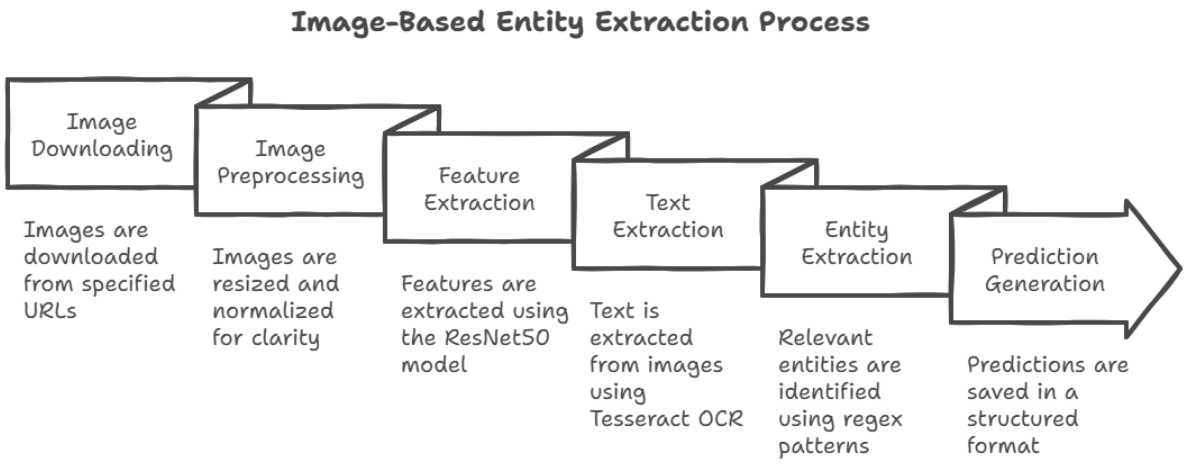
Image-Based Entity Extraction Model



Introduction

The objective of this project was to design a machine learning-based system capable of extracting specific entities—such as dimensions (width, height, depth), weight, and wattage—from images by utilizing Optical Character Recognition (OCR) and deep learning techniques. The goal was to automate the detection of measurements and specifications from images that typically contain textual data, such as product labels, technical drawings, or blueprints.

Model Approach

The model combines two primary techniques:

1. Optical Character Recognition (OCR): Using Tesseract, a widely used open-source OCR engine, the model extracts numeric values and associated units (e.g., "15 cm," "2.5 kg") from the images.

2. Deep Learning-Based Image Processing: Images are pre-processed and analysed with a ResNet50 model, pre-trained on the ImageNet dataset.

ResNet50 standardizes images through resizing and normalizing, enhancing text clarity and OCR accuracy.

Workflow

1. Image Downloading and Preprocessing

- Downloading Images: The `**download_image**` function retrieves images from URLs, saving each in a specified directory. If the directory doesn't exist, it is created. Each image is saved under a unique name, either derived from its URL or assigned sequentially (e.g., image_0.jpg, image_1.jpg). The function `download_images_from_csv` handles image downloads from URLs listed in a CSV file.

- Image Preprocessing for Feature Extraction: The `**preprocess_image**` function resizes each image to 224x224 pixels, a standard input size for pre-trained models like ResNet50, normalizes pixel values to a 0–1 range, and adds a batch dimension, preparing it for neural network input.

2. Feature Extraction Using Pre-trained ResNet50

- Loading ResNet50: The ResNet50 model is initialized using the `ResNet50` class from Keras, with pre-trained weights from the ImageNet dataset. Configured without its final classification layer, it serves as a feature extractor, producing a high-dimensional representation of each image.

- Extracting Features: The `**extract_image_features**` function takes a preprocessed image and extracts its features through ResNet50. Using the `preprocess_input` function ensures compatibility with ResNet50's input requirements, resulting in a rich feature vector representing the image's content.

3. Optical Character Recognition (OCR) with Tesseract

- Loading Tesseract: The OCR engine Tesseract is configured by setting its executable path with `pytesseract.tesseract_cmd`. Tesseract enables text extraction from images.

- Extracting Text from Images: The `**extract_text_tesseract**` function reads an image with the Pillow library and uses Tesseract's `image_to_string` method to convert the image content into readable text. This step is essential for interpreting images with labels, product details, or other textual information.

4. Entity Extraction from Text

- Regex-based Entity Extraction: The `**extract_entity**` function applies regular expressions (regex) to locate numerical values and units (e.g., grams, centimeters, inches) within the text extracted by Tesseract. By matching patterns such as "number + unit" (e.g., "100 grams," "5 kg"), this function identifies relevant values, such as weight or dimensions, which can then be processed further.

5. Image Processing Pipeline

- Image Processing and Prediction: The `**process_image**` function orchestrates image preprocessing, OCR-based text extraction, and regex-based entity identification. Each image is preprocessed for feature extraction, run through OCR to retrieve text, and then analyzed for relevant entities. The function returns extracted entity values as predictions.

- Generating Predictions for a Dataset: The `**generate_predictions**` function ties all components together. It downloads images from a CSV file, processes each image using `process_image`, and records predicted entity values. Results are saved to a new CSV file, `testpredictfinal.csv`, allowing efficient processing of large datasets and structured extraction of relevant attributes.

System Workflow

- Input: The system takes a CSV file (`test.csv`) containing a list of image URLs (in the `image_link` column). Each image represents a product and potentially displays information like weight, volume, or dimensions.

- Image Downloading: Using URLs in the CSV, images are downloaded to a local folder (`images/`) with sequentially generated names.

- Image Processing: Each downloaded image undergoes preprocessing (resizing and normalizing), OCR text extraction, and regex-based entity extraction. Relevant information, such as weight or dimensions, is identified and processed.

- Prediction Generation: Extracted entities are saved in `testpredictfin.csv`, with each prediction indexed according to the original CSV. The file stores extracted values, facilitating a structured view of product attributes.

Conclusion

This model effectively automates the extraction of key measurements from images, significantly reducing manual data entry for measurement or specification extraction tasks. The combination of deep learning (ResNet50) and OCR (Tesseract) ensures accurate text retrieval, while regular expressions and custom mapping filter relevant data efficiently. Demonstrating robustness across diverse entity types (such as dimensions, weight, and wattage), the model holds promise for practical applications, including product cataloging and technical documentation analysis.