**Approach for Feature Extraction from Images**

**Problem Statement**

The goal of this hackathon is to develop a machine learning model that extracts entity values (e.g., weight, dimensions, voltage) directly from product images. This capability is particularly useful in areas such as healthcare, e-commerce, and content moderation, where obtaining key information from images is crucial. Digital marketplaces often lack detailed textual descriptions, making it necessary to extract important details, such as weight, volume, and wattage, directly from images.

The dataset consists of product images and metadata, such as:

* **index**: A unique identifier for each product.
* **image\_link**: A public URL where the product image can be accessed.
* **group\_id**: The category code of the product.
* **entity\_name**: The type of entity (e.g., "item\_weight", "height") associated with the product.
* **entity\_value**: The actual entity value (e.g., "34 gram"), which is only available in the training data.

The test data consists of similar columns but without the entity\_value, which we need to predict. The output must contain two columns: index and prediction, where prediction is a string with the value followed by its unit (e.g., "34 gram").

**Problem Analysis**

The main challenge in this problem was twofold:

1. **Extracting text from images**: Since images contain important information, we needed a reliable way to extract this data.
2. **Extracting the correct entity values from the text**: Once the text was extracted, the relevant values had to be retrieved based on the type of entity (e.g., weight, height, voltage).

Our approach focused on two primary stages:

1. **Text extraction from images** using AWS API calls.
2. **Entity value extraction** from the extracted text using regular expressions.

**Approach**

**1. Text Extraction from Images**

We used AWS Textract to extract text from the images. Textract is an AWS service that uses machine learning to extract printed text, handwriting, and data from documents and images. It provided us with a fast and reliable way to process over 131,000 images in the test set.

* **API Integration**: We made API calls to AWS Textract for each image. We processed both the training and test images (around 2 lakh images for training and 131,187 for testing).
* **Storing Extracted Text**: The extracted text for each image was stored in a separate CSV file. This file included:
  + image\_path: Path of the image file.
  + group\_id: The category code of the product.
  + entity\_name: The type of entity (e.g., item weight, height).
  + extracted\_text: The text extracted from the image using AWS Textract.

**2. Entity Value Extraction Using Regular Expressions**

The next step was to extract the entity values from the text based on the entity type (e.g., weight, volume, dimensions). The challenge was to handle different formats in which these values might appear in the text. We solved this problem using regular expressions tailored to each entity type.

**Regular Expressions for Entity Types**

The entity\_name had 8 categories:

* height
* depth
* item\_weight
* max\_weight\_recommendation
* item\_volume
* voltage
* wattage

We created specific regular expressions for each entity type to capture the possible ways the entity value could be represented in the extracted text. For example:

* **Weight**: We accounted for different formats such as "34 gram", "2.5 kg", "500g".
* **Dimensions**: For height, depth, and volume, we considered patterns like "12 cm", "3.5 inch", "100mm".
* **Voltage/Wattage**: We searched for patterns like "220V", "60 watts".

Each regular expression was designed to identify the numeric value and the corresponding unit. If a valid match was found, the value and unit were concatenated as a single string (e.g., "34 gram").

**Applying Regular Expressions to Extract Entity Values**

We wrote a function that applied the regular expressions for each entity type. This function was executed on the test dataset to extract the entity value based on the given entity\_name. The process included:

1. **Text preprocessing**: Cleaning the extracted text to remove noise.
2. **Regular expression matching**: Using the appropriate regular expression for the entity type to find the relevant value and unit.
3. **Handling cases with no match**: If no match was found, an empty string "" was returned.

**3. Generating the Submission File**

Once we had the predicted entity values, we needed to format the output according to the given guidelines. The submission file had to contain two columns:

* **index**: The unique identifier from the test file.
* **prediction**: The predicted entity value in the format "x unit" (e.g., "34 gram").

If no value was found for a particular image, we returned an empty string for that row.

We ensured that the output was formatted correctly using the sanity checker provided in src/sanity.py. This step helped us validate the format before submission.

**Conclusion**

Our approach relied on a combination of AWS Textract for accurate text extraction and regular expressions for extracting entity values based on patterns in the extracted text. By using a robust process for text extraction and carefully designed regular expressions, we were able to efficiently extract the required entity values for over 130,000 test images.

Key points of the solution:

* **Scalable text extraction** using AWS API calls.
* **Entity-specific regular expressions** to handle different units and formats.
* **Preprocessing and output validation** to ensure compliance with the required submission format.

This structured approach helped us solve the problem efficiently and meet the submission criteria.

* Nagendra
* Bharath Kumar
* Mukunda
* Srihari