Solution report for the Amazon ML challenge*

(subject to change)

dhAI kilo

Observations on the Dataset:

- The training dataset comprised 750 distinct groups of items under group_id.
- Upon reviewing the images associated with each item, we observed that there were 750 unique product types, which included products like furniture, batteries, wallets, pipes, etc.
- Each product had associated entities listed under entity_name. Even though there were 8 different entity types listed, we found that the entities varied depending on the product type in the training dataset. For instance,
    - Furniture products only had dimensional entities like length, width, and height.
    - Battery products had entities related to voltage and wattage.
- After examining several images from the provided image links:
    - Some images did not contain the expected numerical values or entities, leading us to believe there may be errors in the training set.
    - Certain images displayed multiple values of the same unit (e.g., 15kg and 30kg in the same image), making it necessary to apply filtering techniques to extract the correct value.
    - Images with dimensional entities had texts written in different orientation, so we assumed we needed to take a orientation invariant image processing models and techniques.

Processing the dataset + solution

- We began by exploring various OCR (Optical Character Recognition) tools to extract text from the images.

- Since the orientation of the text in the images varied, we needed an OCR that could detect and recognise text irrespective of its orientation.

- After evaluating several models, we found that PaddleOCR was the most suitable for our needs. It delivered accurate and fast results when run on Google Colab's T4 GPU, handling text orientation effectively.

- Once the text was extracted from the images, it needed cleaning. Due to image orientation and inconsistencies in the data, the raw text initially appeared disorganised and did not make contextual sense.

- To process the text, we decided to structure it based on the ‘entity_name’ we were predicting. Implemented the following steps:
    1. We took the entire extracted text along with the specific query (entity_name) we needed to predict.
    2. For each query, we mapped it to all possible units that could be associated with that entity (e.g., if the `entity_name` was `item_weight`, we mapped it to units such as kg, g, mg, etc.).
    3. While mapping the queries, we ensured that various forms of the same unit (e.g., kilograms, kg, kilogram) were standardised to a single word, like kilogram.
    4. We repeated this process for each of the eight entities in the dataset.

- Once the query was mapped to its relevant units, we used regular expressions (regex) to parse the extracted text and obtain a list of relevant values associated with that entity.

- The algorithm we developed for extracting relevant entities was based on key observations made from analyzing the images:

- For items requiring depth, width, and height, we observed that:
    1. The longest numerical value in most cases corresponded to height.
    2. The second longest value was typically the width.
    3. The third longest value was usually the depth.

4. When images presented a range of values (e.g., 5-10 cm), we noticed that in most cases, the higher value was used as the entity value.

- With these insights, we tried the following algorithm:

    1. Sort the extracted entities based on their units and numerical values.
    2. For every entity except for depth and width, select the maximum value.
    3. If the query is for width, return the second largest value.
    4. If the query is for depth, return the third largest value.
    5. Exceptions:

        A. If no text was detected, return a blank value.

        B. If text was detected but had no numerical values, return a blank value.

        C. If text and numbers were detected, but no units matched the query mapping, return a random numerical value paired with a random unit from the query.