**1. Understanding the Data**

* **Dataset**: Two key files are provided: train.csv (with labels) and test.csv (without labels).
  + **Columns**:
    - index: Unique identifier for each sample.
    - image\_link: URL to download images.
    - group\_id: Product category.
    - entity\_name: The type of attribute being extracted (e.g., weight, volume).
    - entity\_value: The actual value (e.g., 34 grams).
* **Sample Output**: Use sample\_test\_out.csv as a guide for formatting predictions.
* **Allowed Units**: These are specified in constants.py and the appendix.

**2. Data Preprocessing**

* **Image Downloading**: Use the function download\_images from src/utils.py to download images linked in the dataset.
* **Exploratory Data Analysis (EDA)**:
  + Check the distribution of different entity types (e.g., weight, volume) in the train.csv.
  + Analyze variations in how values are presented (e.g., different units for the same entity).
* **Image Augmentation (if needed)**: Augment data using techniques such as rotation, scaling, or contrast adjustments to improve model generalization.

**3. Feature Extraction**

* **OCR (Optical Character Recognition)**: Use OCR to extract text from images. Options include:
  + **Tesseract**: Free and open-source OCR engine.
  + **Google Vision API**: A cloud-based service for text extraction.
  + **EasyOCR**: A popular Python library.
* **Text Preprocessing**:
  + Clean the extracted text to remove noise (unwanted symbols, numbers, etc.).
  + Focus on extracting key units (grams, liters, inches, etc.).

**4. Model Selection**

* **Entity Recognition Model**:
  + Consider using **Natural Language Processing (NLP)** techniques to extract the relevant entity and value from the OCR output. For instance, a Named Entity Recognition (NER) model trained on domain-specific data might help detect patterns like "34 grams" or "200 ml".
  + Pre-trained models (e.g., from Hugging Face's transformers library) can be fine-tuned on the dataset to identify patterns related to units and values.
* **Object Detection Models**:
  + If the entity values appear in specific regions of the image, use object detection models like **YOLO** or **Faster R-CNN** to locate and extract relevant features.

**5. Model Training and Evaluation**

* **Train the Model**: Fine-tune the chosen models (OCR + NLP/object detection) on the training dataset. Ensure the model is able to correctly detect entity values for various categories.
* **F1 Score Evaluation**:
  + The submission will be evaluated using the **F1 score**, which balances precision and recall.
  + Precision = True Positives / (True Positives + False Positives)
  + Recall = True Positives / (True Positives + False Negatives)
  + Optimize the model by minimizing both false positives and false negatives.

**6. Output Formatting**

* Predictions should follow the format: "x unit", where x is a float number, and unit is one of the allowed units (as per constants.py).
* Invalid formats, such as “60 ounce/1.7 kilogram” or scientific notations like “2.2e2 kilogram”, are not allowed.
* Use src/sanity.py to ensure the final output file passes all formatting checks.

**7. Post-Processing**

* Implement any necessary post-processing steps to ensure the extracted values are correctly formatted, e.g., rounding numbers to specific decimal places and converting between units if required.
* If no entity value is found, return an empty string ("").

**8. Testing and Validation**

* **Test on the provided test.csv** without labels and generate predictions.
* **Validate Output**: Run the output through the sanity.py script to ensure it is correctly formatted and includes predictions for all test samples.

**9. Submission**

* Submit the final test\_out.csv file with predictions matching the format of sample\_test\_out.csv.

**10. Optional Improvements**

* **Model Ensemble**: Combine predictions from multiple models (e.g., using both OCR and object detection models) for better accuracy.
* **Hyperparameter Tuning**: Use techniques like grid search or random search to optimize model hyperparameters.
* **Advanced OCR Techniques**: Consider improving the text extraction process with advanced techniques like attention-based OCR models.