Developing a Model for Utility Meter

Developing a machine learning model to automate the reading of utility meter images requires a comprehensive pipeline. Here's an outline of the approach

### **1. Problem Understanding and Data Collection**

* Dataset:Due to the limited availability of data, only 10 source images were provided for training. Efforts were made to identify additional datasets to enhance model generalization; however, the identified datasets significantly differed in characteristics such as resolution, orientation, and environmental conditions, making them incompatible with the provided images.
* Annotations: The annotation process was carried out in two stages to enhance the model's ability to accurately detect and interpret utility meter readings:

1. Stage 1 - Utility Meter Segmentation   
   The utility meter images were first annotated for meter detection using Roboflow, where bounding boxes were drawn to segment the entire utility meter region. This stage focused on detecting the meter as a whole within the image.
2. Stage 2 - Number Detection   
   Following the first stage, the detected utility meter regions were cropped and used as inputs for the second stage. These cropped images were re-uploaded to Roboflow, where bounding boxes were further annotated to identify and label the individual digits or numbers displayed on the meter.

This two-stage annotation process ensured precise detection of the utility meter and its readings, allowing the model to first localize the meter in diverse environments and subsequently focus on digit recognition within the detected regions.

2. Preprocessing

* For utility meter detection and number recognition, the following augmentation techniques are performed to increase the number of images as well as to introduce the robustness to the model by introducing different variability to the dataset. Following Augmentations are performed to increase the robustness of the model.
  + Geometric Transformations such as random rotations, and scaling
  + Photometric Transformations such as adjusting brightness, contrast, saturation
  + Affine Transformations such as Translation, Shearing

3. Model Development

Utility meter images were annotated using instance segmentation techniques. Various instance segmentation models were evaluated, including Segmentation Models Library, YOLOv8n-seg, YOLOv8m-seg, YOLO11n-seg. Among these, YOLOv8m-seg was identified as the most effective model for detecting utility meter readings based on performance metrics such as precision, recall, and mAP.

OCR Model Training:

To prepare for OCR number detection, the segmented utility meter images were downloaded with their corresponding masks.Bitwise XOR operations were performed to crop and isolate the regions containing meter readings.These cropped regions were annotated with bounding boxes for individual digits.A Faster R-CNN model was used to train the OCR for detecting numbers in the cropped images.The model was trained for 40 epochs, leveraging the annotated data to learn digit recognition effectively.The trained Faster R-CNN model was evaluated to identify the optimal checkpoint based on validation metrics.The best-performing model was then used to detect numbers in new images. The results demonstrated accurate number detection and recognition in utility meter readings.

4. Model Deployment

The trained models for utility meter segmentation and OCR detection were deployed using Streamlit, enabling a user-friendly interface for real-time processing. Users can upload an image of a utility meter, and the uploaded image is processed through the deployed models such as YOLOv8m-seg to detects and segments the utility meter from the image and faster R-CNN to Identify and recognizes the numbers within the segmented region. The OCR results (detected numbers) are displayed alongside the segmented image.



**Key Features:**

* Real-time image upload and processing.
* Visual display of the segmented meter and detected numbers.
* Intuitive interface powered by Streamlit, making it accessible for non-technical users.

Inference:

The model, despite being trained on a limited number of images, demonstrates the capability to detect numbers with reasonable accuracy. However, to enhance its robustness and generalization, it is essential to augment the dataset by incorporating a greater variety of images captured under diverse environmental conditions. This will enable the model to perform reliably across different scenarios and improve its overall accuracy and effectiveness.

Code files Included:

**train\_yolov8.py** – Script for training YOLOv8 models for object detection tasks.

**image\_segmentation\_ocr.ipynb** – Notebook for Exploratory Data Analysis (EDA) on image segmentation and OCR training using Faster R-CNN.

**deploy.py** – Streamlit application for deploying the trained models in an interactive web interface.

**app.py** – Flask-based application for deploying models via a lightweight web server.

**Dockerfile** – Configuration file for containerizing the Flask application for streamlined deployment.

References

1. https://docs.ultralytics.com/models/yolov8/

2. https://www.kaggle.com/datasets/merrickolivier/water-meter-ocr-images/data

3. https://pytorch.org/vision/stable/models/faster\_rcnn.html

4. https://streamlit.io/

5. https://roboflow.com/annotate