Water Meter Agent: Version 0

Fernan Frans B. Pelobello

12/8/2025

A. Project Description

In this project, we developed a water meter reading system using YOLOv8 for detecting meter windows and a simple CNN model for digit recognition. The goal is to demonstrate a proof of concept for a non-agentic approach to automated water meter reading.

B. Dataset

The dataset used in this project was obtained from Roboflow. It contains 351 water meter images with 1,755 annotations. Only one class was used—the water meter window.

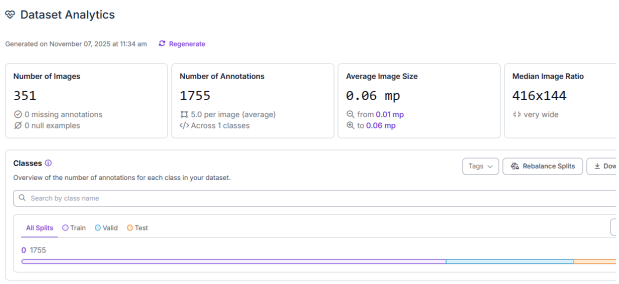

Fig 1: Dataset Analytics

C. System Flow

Figure 2 illustrates the system flow of the Water Meter Agent. The process begins by detecting all water meter windows using a custom-trained YOLOv8 model. Each detected window is then processed individually:

1. Crop the detected window
2. Preprocess the image to clean and enhance visibility
3. Use a CNN model to detect and classify the digit inside the window
4. Combine all predicted digits to produce the final water meter reading

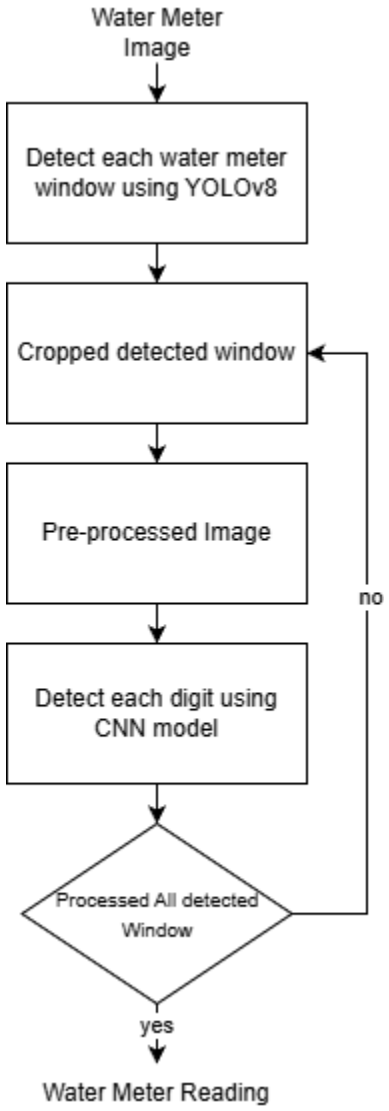

Figure 2: System Flow

D. Detecting Windows using YOLOv8

To locate the water meter windows, we used YOLOv8, a state-of-the-art, lightweight, and fast object detection model. The model was trained in Roboflow and achieved high accuracy.

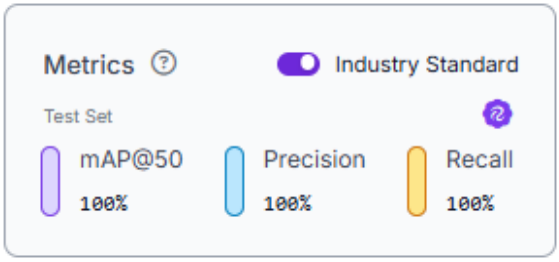

Figure 3: Results Metrics of the Trained YOLOv8 Model

The model reached 100% mAP@50, precision, and recall, largely because of the limited dataset and the similarity of the images. However, despite the high metrics, the model did not exhibit overfitting.

As shown in Figure 4, the training curves demonstrate stable convergence, with validation metrics improving alongside training metrics and loss values remaining low.

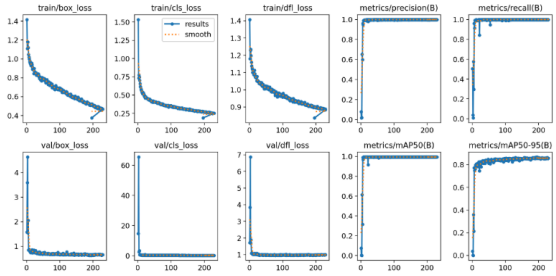

Figure 4: Training Graphs

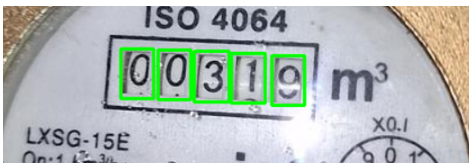

Figure 5: Sample Detected Water Meter Windows

E. Preprocessing of Detected Windows

a. Cleaning and Grayscale Conversion

The first preprocessing step applies Gaussian blur to reduce noise, followed by conversion to grayscale. The image is then binarized using cv2.adaptiveThreshold to adapt to varying lighting conditions. This converts the background to black and keeps the digit highlighted.

Fig 6: Sample Cropped Water Meter Window

Fig 7: Cleaned and Binary Image

b. Keeping the Largest Component

To remove noise, we extract only the largest connected component using cv2.connectedComponentsWithStats. This isolates the most prominent digit shape.

Fig 8:Largest Component Extraction

c. Cropping the Digit

The largest component is then tightly cropped to further isolate the digit.

Fig 9: Cropped Largest Component

d. Resizing the Image

The digit image is enlarged proportionally to ensure better visual clarity for recognition.

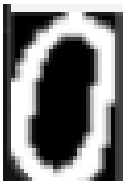

Fig 10: Scaled Image

e. Adding Padding to Create a Square Image

Padding is added to center the digit and ensure the image is square.

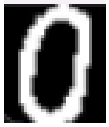

Fig 11: Padded Image

f. Resize to 28x28 pixels

Finally, the image is resized to 28×28 pixels, the input size required by the CNN model.

F. Detecting of Digits using MNIST Model

After preprocessing, digit recognition is performed using a pre-trained MNIST CNN model, publicly available online.

The MNIST model was trained on 70,000 handwritten digit images.

Repository:
https://github.com/lorossi/mnist-model

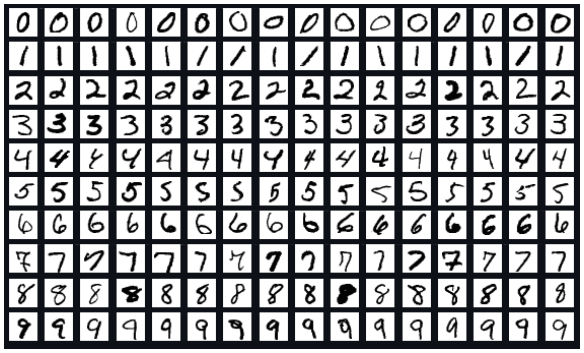

Fig 12: MNIST sample Dataset

G. Results and Discussion

We successfully developed an end-to-end water meter reading system using YOLOv8 for window detection and the MNIST CNN model for digit classification.

Currently, the average processing time per image is around 10 seconds.

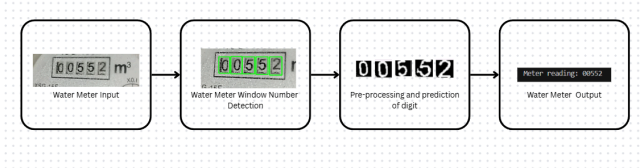

Fig 13: End-to-End Water Meter Prediction

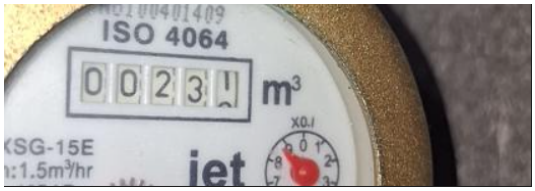

Fig 14: Sample Water Meter Image

```
(wizy) F:\Water-Meter-Agent>python predict.py
Detected 5 windows.
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 617ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 62ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 40ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 31ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 48ms/step
Meter reading: 00231
Prediction completed in 8.36 seconds.
```

Fig 15: Sample Output from Fig. 14 Input

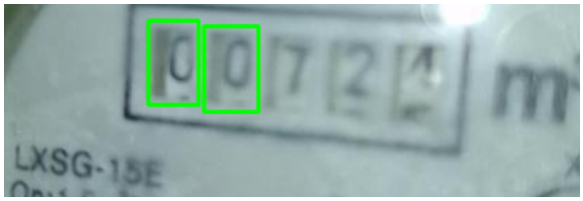

Fig 16: Failed detection of water meter windows because of glare of light

Fig 17: Sample of distorted image because of noise

However, some limitations remain:

- Glare, water droplets, and noise can cause failed detections (Fig. 16–17).
- The MNIST model struggles with digits that are not handwritten, affecting accuracy.
- The dataset is still small, limiting the robustness of the detector.

A more accurate system would require:

- A larger and more diverse dataset
- A custom CNN model trained specifically for water meter digits
- More advanced preprocessing for low-quality images

Despite these limitations, the project demonstrates a working proof of concept for automated water meter reading using YOLOv8 and CNN-based digit detection.

H. Conclusion

In conclusion, this project successfully implemented an end-to-end water meter reading pipeline using YOLOv8 and a CNN MNIST model. Although the performance is not yet perfect, this serves as a strong baseline for further development.

Improvements such as dataset expansion, enhanced preprocessing, and a custom digit recognition model can significantly increase accuracy and reliability.