International Journal of Engineering Inventions
e-ISSN: 2278-7461, p-ISSN: 2319-6491
Volume 15, Issue 6 [June 2026] PP: 47-52

A Two-Stage Deep Learning Pipeline for Automatic Reading of Seven-Segment Digital Caliper Displays

Hong-Thai Nguyen
Faculty of Mechanical Engineering, Thai Nguyen University of Technology, Thai Nguyen, Vietnam

Abstract—*Digital calipers are ubiquitous precision-measurement instruments in manufacturing, yet their readings are still transcribed manually in most workflows, introducing errors and breaking traceability. This paper presents a lightweight two-stage deep-learning pipeline that reads the value shown on a seven-segment LCD of a digital caliper from an ordinary photograph. The first stage uses a YOLOv8 segmentation model to localise the display region; the second stage uses a YOLOv8 detector to recognise individual digits and the decimal point, after which a geometric rule reconstructs the numeric value. On a self-collected dataset, the localisation stage achieves a box mAP@0.5 of 0.99 and a mask mAP@0.5 of 0.96 from only 63 annotated images, while the digit-recognition stage reaches an mAP@0.5 of 0.93. The complete pipeline runs in approximately 100 ms per image on a consumer GPU, making it suitable for low-cost shop-floor deployment.*
Keywords—*seven-segment display, optical character recognition, YOLOv8, object detection, digital caliper, convolutional neural network, industrial metrology*

Date of Submission: 06-06-2026 | Date of acceptance: 16-06-2026

I. Introduction

The digital caliper is a ubiquitous hand-held instrument used to measure dimensions to a resolution of 0.01 mm. Low-cost calipers rarely expose a digital data interface; operators read and record values by hand, introducing transcription errors and breaking the traceability chain that modern quality-management standards require [1].

Automating this transcription is a specialised case of optical character recognition (OCR). General-purpose OCR engines perform poorly on seven-segment displays due to disconnected segments and variable contrast. Dedicated recognisers have evolved from hand-crafted heuristics [2] to deep neural networks [3-5].

We propose a two-stage pipeline that decouples localisation from recognition: (1) a compact YOLOv8 segmentation model [6, 7] isolates the LCD region, and (2) a second model reads the digits inside the crop, for which we evaluate both a YOLOv8 detector and a classical contour+CNN classifier.

Contributions: (1) a complete pipeline trainable from ~60 annotated images; (2) an empirical comparison of detection-based vs. contour+CNN digit recognition; and (3) a characterisation of the accuracy/latency trade-off on consumer hardware.

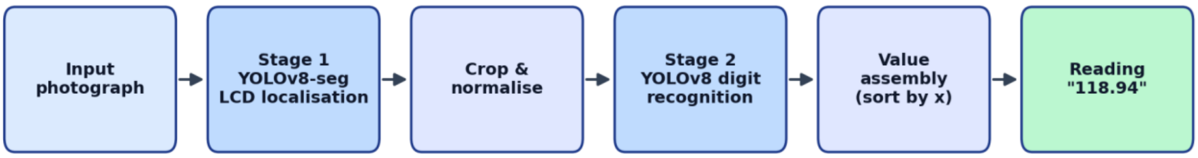

Figure 1. *Overview of the proposed two-stage caliper-reading pipeline.*

II. Related Work

2.1 Seven-segment recognition

Early seven-segment readers relied on fixed-geometry heuristics: after binarisation and deskewing, segment on/off patterns are mapped to digits via a truth table [2, 6]. These methods are fast but brittle under perspective distortion and glare.

Learning-based approaches overcome these limitations. Kanagarathinam and Sekar [3] used connected-component analysis with a trained classifier. More recent works apply single-shot detectors [4, 5, 7], treating each digit as an object class and achieving robustness to rotation and scale change.

2.2 Compact object detectors
The YOLO family [8, 9] provides a favourable accuracy/speed trade-off with small variants running in real time on consumer GPUs. YOLOv8 [10] additionally offers an instance-segmentation head, which we exploit to obtain a tight LCD mask. Starting from COCO-pretrained weights, strong localisation can be learned from only a few dozen domain images.
2.3 Reading instruments from images
Automatic reading of analogue and digital instruments has been studied broadly [11, 12] for remote monitoring and inspection. Our work targets specifically the hand-held digital caliper, which has received little dedicated attention despite its industrial ubiquity.

III. Methodology

3.1 Pipeline overview
The system processes a single RGB photograph in four steps (Figure 1): (1) localise and crop the LCD; (2) normalise the crop; (3) recognise digits and the decimal point; (4) assemble symbols left-to-right into a signed decimal number.

3.2 Stage 1: LCD localisation
We fine-tune a YOLOv8s-seg model initialised from COCO-pretrained weights to predict a single class (digital_display) with its instance mask. Using a segmentation head rather than a plain bounding-box detector lets us crop along the true display boundary, suppressing reflective metal borders that would otherwise confuse digit extraction.
Input images are resized to 640×640. Training uses SGD (lr=0.01, momentum=0.937, weight decay=5×10^{-4}), batch size 16, standard augmentation (mosaic, HSV jitter, flips), up to 200 epochs with early stopping (patience 25).
3.3 Stage 2A: Digit recognition by detection
The cropped display is fed to a YOLOv8n model fine-tuned on a public seven-segment dataset (eleven classes: digits 0–9 and the decimal point). Each detected box carries a class label and confidence score passed to the assembly module.
3.4 Stage 2B: Digit recognition by contour + CNN
As a comparison baseline we implement a classical alternative. The crop is converted to greyscale, contrast-enhanced with CLAHE, binarised with Otsu's method, and cleaned with morphological opening. Connected contours are filtered by area and aspect ratio into candidate digits (large, tall blobs) and dots (small, round blobs). Each candidate is padded to a square, resized to 28x28, and bordered to 32x32. A compact CNN with three convolutional blocks (32-64-128 filters, each with batch normalisation, ReLU, and 2x2 max-pooling) followed by a 256-unit fully connected layer with dropout classifies each candidate into one of twelve classes (0-9, dot, minus). This route is attractive when no labelled digit-detection dataset is available, since the operator can bootstrap a classifier from digits cropped by the contour stage itself.
3.5 Value assembly
Both variants emit symbols with horizontal positions, sorted by x-coordinate, mapped to characters, concatenated, and parsed as a floating-point number (e.g. {1,1,8,dot,9,4} → "118.94").

IV. Experimental Setup

4.1 Datasets
Caliper-LCD (self-collected): 126 photographs of a digital caliper under varied angles and lighting; 63 annotated with polygon masks (class digital_display) using labelme and converted to YOLO segmentation format. Images were downscaled to 1280 px on the longest side.
Seven-segment digits (public). For the digit-recognition stage we use a public seven-segment digit dataset with 2,040 training, 168 validation, and 100 test images across eleven classes (0-9 and the decimal point).
4.2 Implementation and hardware
All models use the Ultralytics framework [10] on PyTorch [13]; classical image processing uses OpenCV [14]. Training was on an NVIDIA RTX 3060 (12 GB) GPU; inference timings are measured on the same device.
4.3 Metrics
Performance is reported with COCO-style mAP@0.5 and mAP@0.5:0.95, precision and recall. End-to-end accuracy requires every symbol including the decimal point to match the ground truth.

V. Results and Discussion

5.1 LCD localisation

Despite training on only 63 annotated images, the model achieves a box mAP@0.5 of 0.99 and a mask mAP@0.5 of 0.96 with recall 0.997 (Table 1; Figures 2–4). High recall is critical since a missed display causes total pipeline failure. The result confirms efficient transfer from COCO-pretrained weights.

Table 1. LCD LOCALISATION PERFORMANCE (YOLOv8s-seg, 63 TRAINING IMAGES)

Output	Precision	Recall	mAP@0.5	mAP@0.5:0.95
Box	0.82	0.997	0.99	0.89
Mask	0.82	0.958	0.96	0.88

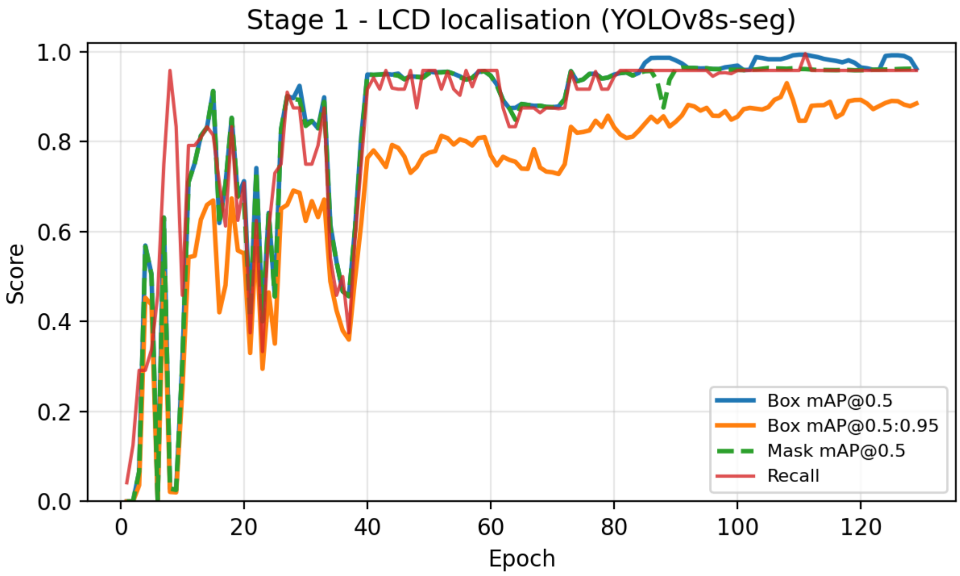

Figure 2. *Stage-1 localisation metrics versus training epoch.*

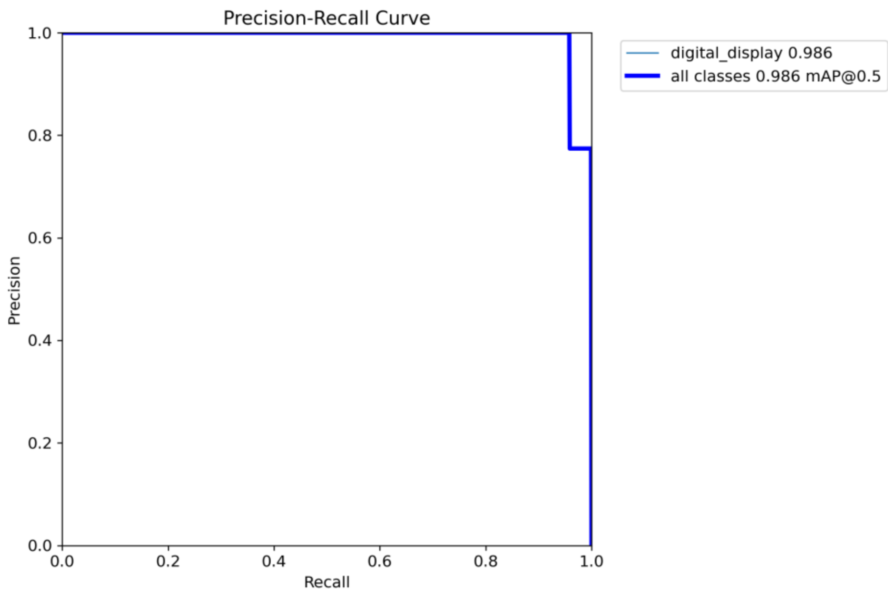

Figure 3. *Stage-1 box precision-recall curve (mAP@0.5 = 0.986).*

Figure 4. *Qualitative LCD detections on held-out caliper photographs. Each detection is labelled digital_display with its confidence.*

5.2 Digit recognition

The YOLOv8n detector achieves mAP@0.5 of 0.93, precision 0.91, and recall 0.93 (Table 2; Figure 5). The mAP@0.5:0.95 of 0.54 is immaterial since the assembly rule uses only class labels and horizontal ordering. The contour+CNN baseline reached mid-90% classification accuracy but was less robust end-to-end, as the contour stage loses dots and merges adjacent digits under glare.

Table 2. SEVEN-SEGMENT DIGIT DETECTION (YOLOv8n, 11 CLASSES)

Model	Precision	Recall	mAP@0.5	mAP@0.5:0.95
YOLOv8n	0.91	0.93	0.93	0.54

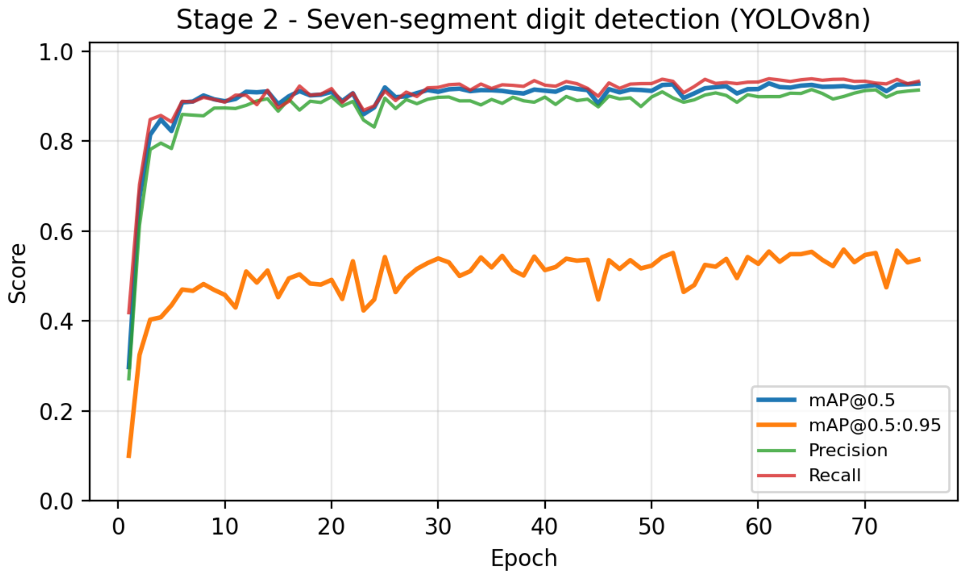

Figure 5. *Stage-2 digit-detector metrics versus training epoch.*

5.3 End-to-end behaviour and latency

The pipeline correctly reads clear, near-frontal captures (e.g. "118.94" from six high-confidence symbols). Remaining failure modes are specular glare and extreme viewing angles. End-to-end latency is ~100 ms on the RTX 3060, supporting interactive and real-time webcam use.

5.4 Discussion

Three findings stand out: (1) decoupling localisation from recognition confines expensive annotation to a single-class task while reusing a large public digit dataset; (2) instance segmentation for the display suppresses reflective borders better than a plain bounding box; (3) an end-to-end digit detector is more robust than a contour+CNN cascade by avoiding hand-tuned binarisation thresholds. The main limitation is the small, single-instrument evaluation set; a larger multi-caliper benchmark is left for future work.

VI. Conclusion

We presented a two-stage deep-learning pipeline that reads digital caliper displays from single photographs. A YOLOv8 segmentation model localises the LCD with box mAP@0.5 of 0.99 from 63 annotated images; a YOLOv8 detector recognises digits with mAP@0.5 of 0.93; a geometric rule assembles the numeric reading in ~100 ms on a consumer GPU. Future work will expand evaluation across caliper models and lighting conditions, add glare-robust preprocessing, and export models to ONNX for mobile deployment.

Acknowledgment

The work presented in this paper is supported by Thai Nguyen University of Technology, Thai Nguyen University, Vietnam.

References

[1] L. S. Dalenogare et al., "The expected contribution of Industry 4.0 technologies for industrial performance," Int. J. Production Economics, vol. 204, pp. 383-394, 2018.
[2] V. N. Shenoy and O. M. Aalami, "Utilizing smartphone-based machine learning in medical monitor data collection: Seven segment digit recognition," in Proc. AMIA Annual Symposium, 2017, pp. 1564-1570.
[3] K. Kanagarathinam and K. Sekar, "Text detection and recognition in raw image dataset of seven segment digital energy meter display," Energy Reports, vol. 5, pp. 842-852, 2019.
[4] E. Finnegan et al., "Automated method for detecting and reading seven-segment digits from images of blood glucose metres and blood pressure monitors," J. Medical Engineering & Technology, vol. 43, no. 6, pp. 341-355, 2019.
[5] J. C. Popayan et al., "Seven-segment display recognition based on deep learning for industrial measurement instruments," in Proc. IEEE Int. Conf. Industrial Technology (ICIT), 2021, pp. 1-6.
[6] R. P. Ghugardare et al., "Optical character recognition system for seven segment display images of measuring instruments," in Proc. IEEE Region 10 Conf. (TENCON), 2009, pp. 1-6.
[7] Y. Huang et al., "A robust seven-segment display recognition method based on deep neural networks," IEEE Access, vol. 8, pp. 205-216, 2020.
[8] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, real-time object detection," in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), 2016, pp. 779-788.
[9] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal speed and accuracy of object detection," arXiv:2004.10934, 2020.

[10] G. Jocher, A. Chaurasia, and J. Qiu, "Ultralytics YOLOv8," tech. rep., Ultralytics, 2023.
[11] L. Liu et al., "Automatic reading of digital and analogue instrument displays based on deep learning," Sensors, vol. 21, no. 3, art. 979, 2021.
[12] G. Salomon et al., "Image-based automatic dial meter reading in unconstrained scenarios," Measurement, vol. 204, art. 112025, 2022.
[13] A. Paszke et al., "PyTorch: An imperative style, high-performance deep learning library," in Advances in Neural Information Processing Systems (NeurIPS), 2019, pp. 8024-8035.
[14] G. Bradski, "The OpenCV Library," Dr. Dobb's Journal of Software Tools, 2000.