

Article

CASCA-YOLOv8: An Enhanced Framework for Accurate Pointer-Type Instrument Reading Recognition

Mingyu Zhao ¹, Shanlin Sun ^{2,*}, Bensheng Xu ² and Zhicheng Tan ²¹ Guangxi Key Laboratory of Low-Altitude Unmanned Autonomous System, Guilin University of Electronic Technology, Guilin 541004, China; kmcqin7113402081@163.com² Guangxi Key Laboratory of Low-Altitude Unmanned Autonomous System, Guilin University of Aerospace Technology, Guilin 541004, China; xubensheng@guat.edu.cn (B.X.); tanzc@guat.edu.cn (Z.T.)

* Correspondence: shy1997@126.com; Tel.: +86-13627832672

Abstract

Pointer-type instruments continue to play an important role in monitoring key operating parameters in industrial environments such as power systems and chemical plants. However, current vision-based reading techniques frequently struggle in terms of accuracy and real-time performance, especially when deployed on edge devices with low processing resources. This study presents an enhanced YOLOv8-based framework that surpasses existing algorithms in terms of accuracy and real-time performance, while maintaining a lightweight design and suitability for deployment on edge devices. A new Context-Aware Spatial-Channel Attention (CASCA) mechanism is suggested, which efficiently captures spatial context and channel dependencies while improving the network's capacity to discern instrument-specific features. To improve computational efficiency, the model uses structural pruning and replaces standard convolutions with Ghost convolutions, resulting in a smaller model size and lower inference complexity while maintaining accuracy. Furthermore, a hybrid approach combining U2Net-based fine segmentation and Hough transform-based line detection is used to accurately recover the pointer's orientation and scale. Comprehensive experiments show that the suggested technique achieves a detection accuracy of 96.1% while boosting inference speed, validating its appropriateness for real-time, high-precision sensor data in industrial settings.

Keywords: pointer-type instrument; YOLOv8; CASCA mechanism; U2Net; gauge detection

1. Introduction

Pointer-type instruments are commonly used in important industrial applications such as power systems, chemical plants, and manufacturing to monitor vital operational parameters such as pressure, temperature, current, and liquid levels. Despite the growing usage of digital gauges and smart sensor technologies, pointer-type instruments remain indispensable in many industries due to their user-friendly displays, high dependability, and ability to operate without an external power supply. However, manual reading remains the most common method, which is not only inefficient and time-consuming, but also prone to human mistake due to difficulties such as angle deviations and subjective assessments [1,2]. With the growing popularity of machine learning applications [3,4], industrial automation, and intelligent detection technologies [5] automating the reading of pointer-type instruments has become crucial for real-time monitoring and boosting operational efficiency [6,7].



Academic Editor: Paolo Spagnolo

Received: 12 January 2026

Revised: 8 February 2026

Accepted: 11 February 2026

Published: 25 February 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

This work addresses the issues related with automatic recognition of pointer-type instrument readings. Existing methods for instrument reading recognition can be roughly classified as classic computer vision techniques or deep learning-based approaches [8]. While traditional methods primarily focus on image processing techniques like edge detection and template matching, deep learning approaches have gained popularity due to their capacity to automatically learn complicated patterns and features from vast datasets. Despite these developments, there are still significant hurdles in applying them to real-world circumstances.

For example, Lu et al. [9] presented MMRNet, a multi-category classification and regression model for pointer localization. While effective, this approach fails to account for scale variation, which can have a substantial impact on reading accuracy in a variety of industrial settings. Similarly, Li et al. [10] created a system for recovering dial areas and recognizing scale lines, but it has difficulty with noisy or occluded images. Huo et al. [11] incorporated the CBAM attention mechanism into YOLOv3, allowing the network to focus on critical areas of the gauge. However, the model's sensitivity to small objects remains limited. Other hybrid approaches, such those by Salaudeen et al. [12] Lu et al. [13] and P. Ni et al. [14], integrate models such as YOLO, ESRGAN and U-Net, or use feature matching and optical character recognition (OCR) algorithms. Li J et al. [15] integrated YOLOv8 and DeepLabv3+ to improve both speed and accuracy.

While promising, these methods are hampered by enormous model sizes, high computing cost, and a limited flexibility to other gauge types.

While these methods have enhanced the accuracy of instrument reading recognition, their real-world applicability is still limited due to many flaws. Many of these techniques rely largely on preprocessing, which may not be robust enough for real-time applications under changing conditions [16]. Furthermore, while these methods perform rather well in controlled situations, they are hindered by enormous model sizes and high computational costs, rendering them unsuitable for real-time applications, particularly on edge devices with limited computational resources. Furthermore, these models frequently lack the ability to generalize across different types of gauges and imaging conditions, limiting their usefulness in a wide range of industrial contexts.

To address these issues, recent advances in lightweight object identification have used approaches including low-rank decomposition [17], network pruning [18,19], quantization [20], and neural architecture search [21]. Among these, network pruning has proven particularly beneficial due to its practical balance of performance and simplicity. Pruning minimizes model size and computational complexity by eliminating unnecessary parameters, making it ideal for real-time deployment [22]. However, despite these overall advances in model efficiency, there is still a gap in their specific application to pointer-type instrument reading identification, which this work seeks to fill.

This study describes an improved YOLOv8-based system that aims to address the shortcomings of existing approaches, particularly in terms of real-time speed, computational economy, and robustness. Through a series of structural optimizations, the suggested approach speeds meter localization while reducing model complexity. By adding the Hough Transform into a U2Net-based segmentation module, we achieve exact scale line and pointer extraction, resulting in crisper boundaries and improved reading accuracy. The primary contributions of this paper are as follows:

1. A lightweight detection network improved with the innovative Context-Aware Spatial-Channel Attention (CASCA) mechanism, which reduces inference time and model size while maintaining excellent accuracy.

2. A robust tilt correction approach that employs ORB feature matching, distance filtering, and RANSAC to successfully correct picture tilt under difficult situations, ensuring accurate instrument reading at diverse viewing angles.
3. A hybrid segmentation technique that combines U2Net-based fine segmentation with the Hough Transform, resulting in exact extraction of scale lines and pointer positions for better reading accuracy.

Through extensive experiments, we show that the proposed framework not only achieves 96.1% detection accuracy but also significantly improves inference speed, making it a viable solution for real-time, high-precision pointer-type instrument reading recognition in industrial settings.

2. Materials and Methods

2.1. Dataset

2.1.1. Data Acquiring

The training dataset included two public datasets supplemented with self-collected photos. Both publicly available datasets are derived from PaddlePaddle AI Studio, whilst the self-collected photographs were taken with a system-provided Android smartphone and 64-megapixel camera. Finally, data augmentation tactics like as blurring and inversion were used to recreate complex real-world conditions. In machine learning, this method improves a model's resilience to complicated external changes while also shortening computational time [23]. The final dataset consisted of roughly 8000 photos, which were divided in an 8:1:1 ratio for training, validation, and testing. Figure 1 below shows partial data processing images.

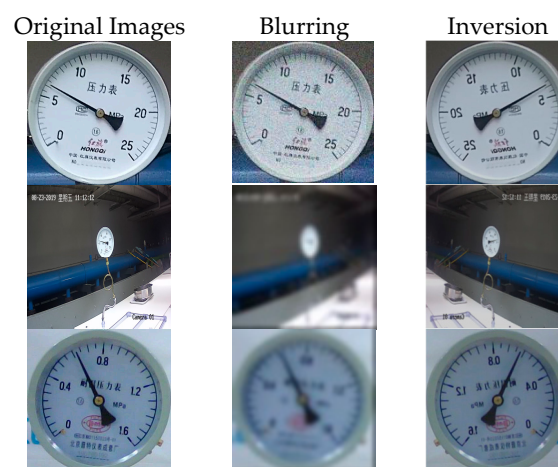


Figure 1. Data-enhanced images.

2.1.2. Perspective Transformation

Non-frontal images are common in actual circumstances, which can reduce the precision of segmenting pointers and scale lines. To mitigate this, the image must undergo a perspective modification. This approach models and corrects projective distortions while preserving geometric consistency, allowing for alignment and 3D reconstruction from many perspectives. Based on this principle, a specialized perspective transformation algorithm was created.

In computer vision, a set of discrete local features (keypoints) can efficiently describe picture content [24]. Leveraging this principle, the proposed method uses feature point detection and matching to estimate the homography matrix between picture pairings, resulting in perspective correction. The procedure starts with ORB feature extraction [25,26],

which identifies and describes keypoints in both the reference and distorted images. ORB combines the efficient FAST keypoint detector with the rotation-invariant BRIEF descriptor to achieve an ideal balance of computational efficiency and matching robustness. The Brute-Force Matcher (BFMatcher) then uses Hamming distance to match features, maintaining only high-quality correspondence pairings. The RANSAC algorithm then uses these matched pairings to estimate a homography matrix that accurately describes the projective transformation between the two viewpoints. Finally, the estimated homography is used to bend the deformed image, thus correcting the perspective and aligning it with the reference view. The entire operation is totally autonomous, requiring no operator intervention, and it may be used in a variety of real-world scenarios, including document rectification, image registration, and augmented reality. Figure 2 shows how the technique improves traditional perspective transformation by incorporating an efficient and robust feature matching mechanism, considerably increasing the algorithm's applicability and reliability.



Figure 2. Comparison of pointer instrument dials before and after perspective transformation.

2.2. The Proposed CASCA-YOLOv8 Algorithm

2.2.1. CASCA Mechanism

The YOLO algorithm initially processes the input image to identify and locate markers, resulting in location information for pointer-type gauges [27]. YOLOv8's architecture consists of four key components: input, backbone, neck, and head [28]. The backbone extracts multi-scale characteristics from input images using modified convolutional structures, which improve feature representation and expand the receptive field. The Neck combines feature maps from many hierarchical levels via upsampling and concatenation techniques, efficiently maintaining both high-level semantic information and low-level spatial features. To improve model accuracy and computational efficiency, this work offers CASCA, a lightweight attention mechanism module. Figure 3 depicts the CASCA branch topologies, Table 1 lists the parameter symbols, and the algorithm is presented in Algorithm 1.

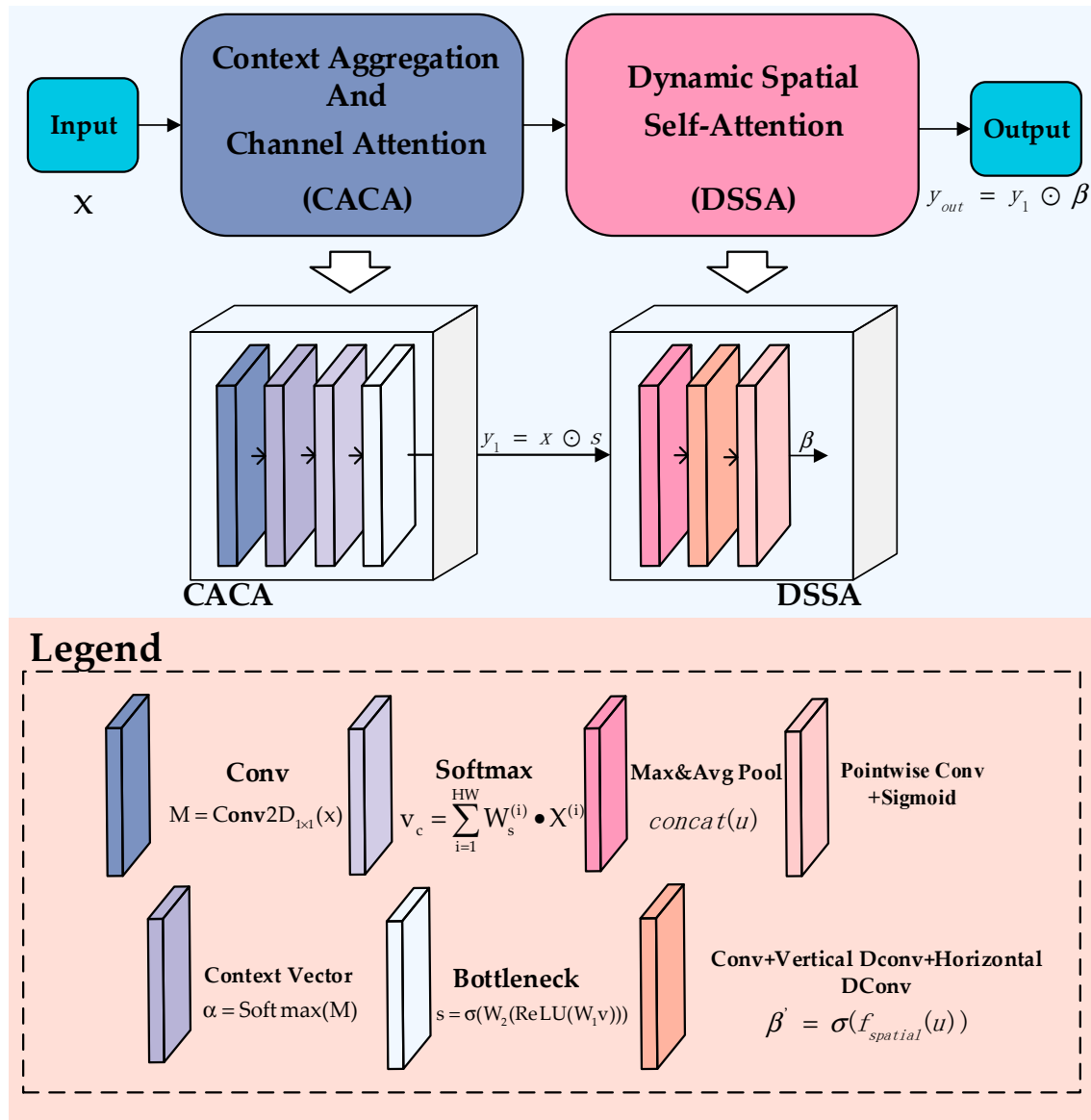


Figure 3. The branch structures of CASCA.

Table 1. Parameter Symbol.

Parameter Symbol	Meaning
B	Batch size
C	Feature map channels
H	Height
W	Width
x	Input feature tensor
M	Spatial Score Map (1 × 1 Convolution Output)
α	Spatial Attention Weights (Softmax Normalization)
v/v _c	Context vector (weighted sum result)
W1, W2	Fully Connected Layer Weight Matrix
s	Channel Attention Coefficient (Sigmoid Output)
y	Channel-weighted features
m	Maximum Pooling Graph
a	Channel-Averaged Pooling Map
u	Pooled features after concatenation (m and a)

Table 1. Cont.

Parameter Symbol	Meaning
f_{spatial}	Spatial Feature Convolution Processing Sequence (1×1 Convolution + Hollow Convolution)
β	Spatial Attention Coefficients (Sigmoid Output)
y_{out}	Final Output Features (Channel Attention + Spatial Attention Results)

Algorithm 1 Context-Aware Spatial-Channel Attention

Require:

x: Input feature map of size [batch_size, channels, height, width]

Ensure:

x_out: Output feature map after applying channel and spatial attention mechanisms

1: Start CombinedAttention

2: Channel Attention:

3: Input: x

4: Apply channel attention mechanism to x

5: Compute and apply channel-wise attention weights

6: Output: $x \times \text{scale}$

7: Spatial Attention:

8: Input: x_out (after channel attention)

9: Apply spatial attention mechanism to x_out

10: Compute and apply spatial attention weights

11: Output: $x_{\text{out}} \times \text{scale}$

12: Return Final Output:

13: x_out

The CASCA module is divided into two complementary branches: Context Aggregation and Channel Attention (CACA) and Dynamic Spatial Self-Attention (DSSA). In this architecture, channel attention is applied before spatial attention, which not only improves computational efficiency but also model correctness. By first refining channel-wise dependencies, the CACA branch offers more detailed contextual information to the DSSA branch, hence enhancing the model's combined capacity to interpret both global structures and local details.

The CASCA models the channel dimension of the input feature map through the context-aware channel attention (CACA) mechanism (as shown in Figure 3). Given an input feature x , a 1×1 convolution is first applied at each spatial position to obtain a score map M . A softmax is then applied over the flattened spatial dimension HW to yield the learned spatial weights. Based on this, x is spatially weighted and aggregated to form the channel description vector v . Subsequently, two fully connected MLP layers generate the channel weights: first, $W1$ reduces the dimension of v and passes it through ReLU; then, $W2$ increases the dimension and passes it through Sigmoid to obtain. Finally, channel-wise broadcast multiplication yields the channel-recalibrated feature $y = x \odot s$, where $\odot$ denotes element-wise broadcast multiplication. This branch replaces fixed global pooling with a learnable mechanism, endowing the channel description vector with content-sensitivity. The dimension-reduction-to-dimension-increase MLP effectively models cross-channel dependencies.

The CASCA generates spatial context weight maps through convolution and performs dynamic normalization on the spatial dimension, replacing traditional global mean pool-

ing. Compared to fixed pooling methods, this dynamic normalization more accurately captures the influence of spatial location on channels, thereby enhancing context awareness. Subsequently, matrix multiplication fuses the spatial weights with channel features to generate dynamic context description vectors, further improving the expressive power of channel features. This module integrates innovative channel attention and spatial attention mechanisms, effectively enhancing feature extraction capabilities.

The spatial attention branch, as illustrated on the right side of Figure 3, operates through the following refined process: the input feature map x first undergoes parallel max pooling and average pooling operations along the channel dimension, producing feature maps M and α , respectively. These maps are then concatenated to form the combined feature representation u . Subsequently, a lightweight convolutional sequence performs directional spatial encoding: initial channel fusion is accomplished through a 1×1 convolution, followed by specialized directional dilated convolutions. Specifically, a vertical dilated convolution (kernel size 3×1 , dilation rate 2, groups 2) processes spatial information along the height dimension, while a horizontal dilated convolution (kernel size 1×3 , dilation rate 2, groups 2) captures features along the width dimension. This decomposed design strategically expands the receptive field in both directional domains while maintaining parameter efficiency. The outputs from both directional convolutions are then integrated through a pointwise 1×1 convolution, producing a unified single-channel spatial response map. This map undergoes sigmoid normalization to generate the final spatial attention weights β . The complete spatial attention output is obtained through element-wise multiplication between the original input x and the spatial attention weights β , resulting in the spatially refined feature map.

The proposed Spatial Attention module effectively captures spatial dependencies by selectively expanding the receptive field in both horizontal and vertical directions through dedicated dilated convolutions. Unlike standard 3×3 convolutions, the employed grouped dilated convolutions enable the network to capture long-range spatial relationships without proportionally increasing computational complexity. This design avoids the excessive parameter growth typically associated with conventional receptive field expansion methods. By strategically incorporating zero-padded dilated kernels, the module enhances cross-scale feature integration while maintaining computational efficiency, thereby significantly improving the model's capacity to recognize both local patterns and global contextual relationships.

The integration of channel and spatial attention enables the CASCA module to achieve a dual enhancement of image features. This integrated process is formulated as follows:

$$y_{\text{out}} = \text{SA}(\text{CA}(x)) = [x \cdot \sigma(W_2(\text{ReLU}(W_1 v)))] \cdot \sigma(f_{\text{spatial}}(u)), \quad (1)$$

The proposed CASCA module accurately identifies task-relevant channel features while simultaneously pinpointing critical spatial regions within the image. In contrast to conventional static attention mechanisms and global pooling techniques, CASCA adopts a dynamic weighting strategy that adapts to varying image content. Traditional approaches, such as global average pooling, rely on fixed statistical measures, which often fail to capture fine-grained local variations and region-specific importance, resulting in inevitable information loss. In comparison, CASCA performs real-time computation of both spatial and channel-wise attention weights, enabling context-aware feature refinement. This dynamic adjustment mechanism substantially enhances feature discriminability and contextual sensitivity, leading to more precise and robust feature representation.

2.2.2. Lightweight Module Ghost Module

To improve computational efficiency, reduce model complexity, and ensure lightweight deployment, while improving feature extraction capabilities, particularly in images with complex backgrounds, the standard convolution layers in the neck section are replaced with lightweight Ghostconv modules [29,30]. This concept, based on the GhostNet architecture, takes advantage of deep networks' inherent feature redundancy. It begins by performing a primary convolution to create a small set of intrinsic feature mappings. The maps are enhanced with cost-effective linear procedures, such as depthwise convolution, to create "ghost" features. The intrinsic and ghost features are then concatenated, keeping representational capacity while significantly reducing computational effort and parameter count.

GhostConv has a much lower computational complexity than typical convolutional layers while maintaining competitive accuracy. This feature makes it especially useful for resource-constrained situations and real-time detection applications. The incorporation of GhostConv into the YOLOv8 architecture enables the creation of a more efficient and lightweight detection system, significantly boosting its deployment practicality in practical applications.

Figure 4 shows that the Ghost module uses a cascaded structure consisting of two GhostConv levels and one SKConv layer to perform effective hierarchical feature processing. The input features are first routed through an initial GhostConv layer for lightweight feature generation, followed by an SKConv layer that adaptively collects multi-scale contextual information using selected kernel techniques. A second GhostConv layer refines the previously processed features. A skip connection combines the original input and altered features, using residual learning to preserve basic knowledge while increasing representational capacity. This approach not only improves gradient flow and training stability, but it also strikes an appropriate balance between feature richness and computational economy.

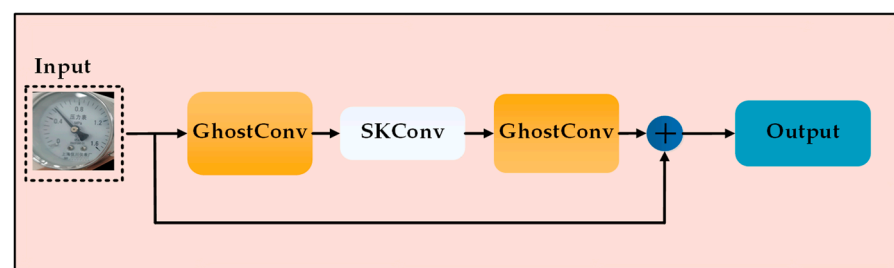


Figure 4. Ghost convolution.

2.2.3. Model Pruning

Following this method, we remove one C2f module from the eighth tier of the backbone network. The C2f module is critical to feature fusion and information flow in the standard YOLOv8 architecture, improving feature diversity and stability via multi-branch convolutions and cross-layer connections [31]. However, under certain application constraints or with limited computational resources, removing the final C2f module from the backbone can drastically reduce parameter count and computational complexity, resulting in faster inference. This structural simplification retains competitive feature extraction capacity while resulting in a smaller network topology, making it ideal for real-time detection applications.

2.2.4. The Algorithm Architecture

Figure 5 shows the general architecture of our CASCA-YOLOv8 system. The diagram depicts the synergistic interaction of its core modules: the CASCA mechanism embedded

in the backbone, the pruned neck structure that uses Ghost convolutions for efficiency, and the integrated U2Net-Hough segmentation module, all orchestrated within a streamlined computational pipeline.

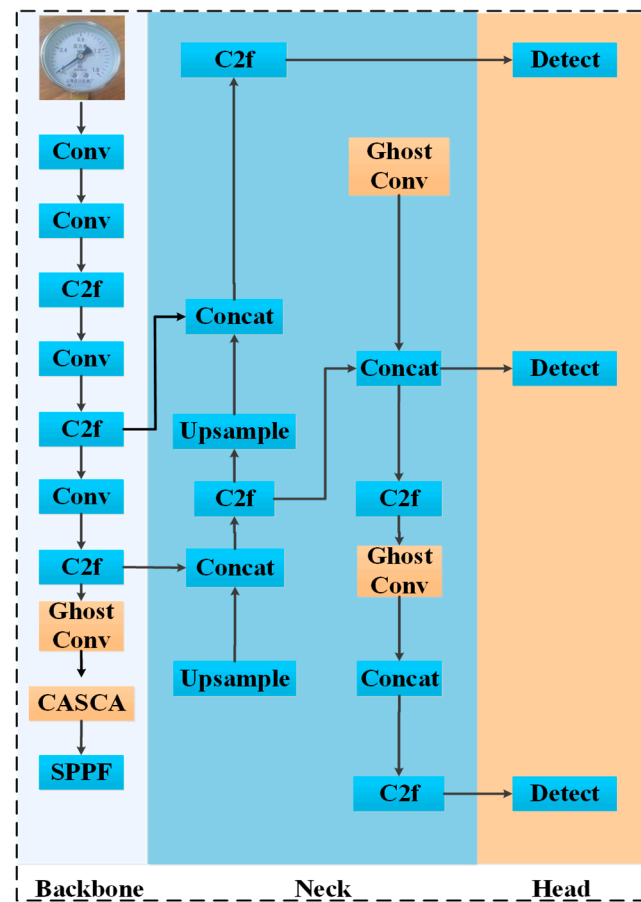


Figure 5. Improved YOLOv8 Algorithm Architecture.

Figure 5 depicts the enhanced YOLOv8 framework, which is based on a hierarchical pipeline with three primary components: the backbone, neck, and head. The Backbone is made up of many C2f modules that act as enhanced residual learning blocks for efficient multi-scale feature extraction. These modules improve gradient flow and feature representation across several network levels. The Neck part uses upsampling and concatenation to accomplish feature fusion. These techniques combine feature maps from many semantic levels, maintaining both high-level context and fine-grained spatial details. During this level, traditional convolution layers are substituted by GhostConv modules. This substitution greatly decreases computational complexity and model parameters while retaining representational capability, adding to the overall lightweight design. The Head consists of numerous detection layers (Detect) that generate the final object localization and classification results. Parts of the convolution processes in the Head, like those in the Neck, are performed using Ghost convolution, which accelerates inference while preserving detection performance. This improved design, which incorporates C2f modules, Ghost convolution, and fast feature fusion pathways, achieves an appropriate mix of detection accuracy and processing efficiency, making it suited for real-time pointer instrument reading applications.

Figure 6 depicts the detailed structure. Examining the feature maps output from various layers, including enhanced layers, reveals that, beginning with layer 0, the feature maps of the GhostConv layer in the backbone have a decreased resolution due to downsampling and convolution processes. The feature map seems hazy and pixelated. The image still

shows the framework of a dial, including parts like the pointer. After conducting Ghost convolution for feature compression and information abstraction, some details are blurred, and the feature map primarily preserves abstract high-level features while losing some low-level detail, resulting in ambiguous object edges.

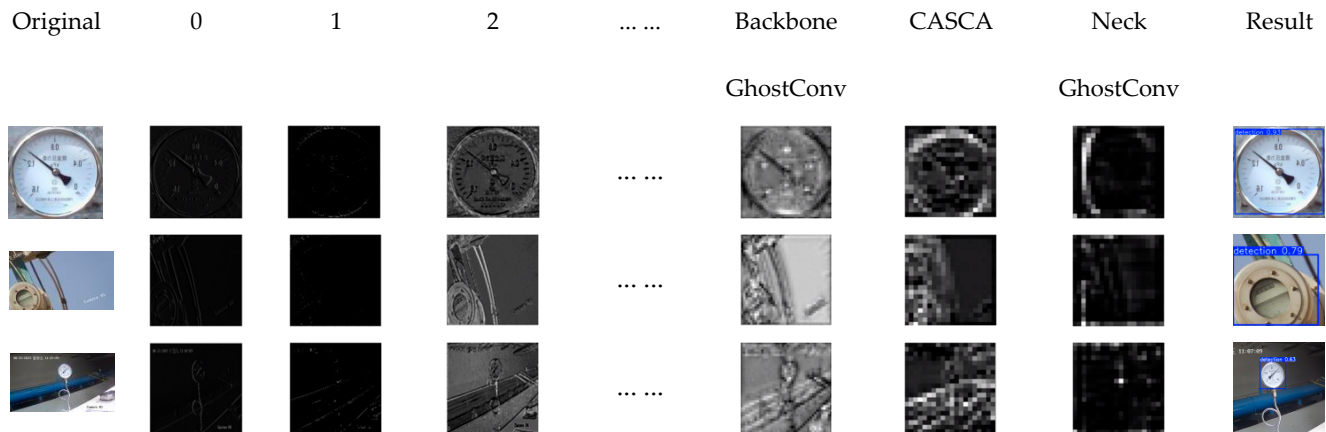


Figure 6. Improved YOLOv8 Algorithm Layer-wise Feature Maps.

The feature map following processing by the CASCA module has a more dramatic pixelation effect than GhostConv, but it has a more defined structure. The image shows that the form of the dial and the position of the pointer have remained unchanged. Although the specifics of the feature map are rather abstract when compared to the GhostConv feature map, the CASCA module preserves more structural information, particularly the contours and center position of the object. This suggests that the CASCA module performs better at preserving crucial spatial information in the image, such as the circular dial and pointer. This feature map could help with subsequent detection and localization tasks because it more clearly reflects the image's essential aspects.

After numerous applications of GhostConv in the neck, the spatial resolution of the feature map continues to diminish due to the removal of redundant information, while the upsampling process recovers the feature map's spatial dimensions. It keeps the image's key information and features while adding abstract information and highlighting the edges and contours.

2.3. Image Segmentation and Optimization

The U-Net model has been frequently used for medical picture segmentation, with satisfactory results [32]. Building on this foundation, U2Net has evolved as a deep neural network architecture optimized for saliency detection and picture segmentation tasks. U2Net is a deep neural network architecture developed primarily for saliency detection and picture segmentation applications. It improves multi-scale feature extraction capabilities and yields richer feature descriptions for downstream detection tasks [33]. Its main innovation is the use of residual U-units (RSUs) to combine receptive fields of various sizes, allowing for the effective capture of multi-scale contextual information. Figure 7 shows how strategically implementing pooling procedures within RSU units increases the network's depth without considerably increasing computational cost [34,35].

U2Net gathers multi-scale contextual information effectively thanks to its deeply layered encoder–decoder structure, which allows for exact segmentation of fine-grained items like instrument pointers and scale lines. To improve segmentation quality, the proposed technique incorporates the Hough transform for line fitting. This traditional technique recognizes conspicuous linear features in a picture by evaluating parameter

space accumulations, with altering the accumulator threshold [36] allowing extraction of a predetermined number of dominating lines. This post-processing phase improves the continuity and structural integrity of linear components, optimizing segmentation borders. The combination of deep learning-based segmentation and traditional image processing not only enhances segmentation accuracy, but also provides a more stable platform for subsequent reading recognition. This hybrid technique offers robust segmentation and exact localization of instrument components, which has significant practical significance in industrial automation applications.

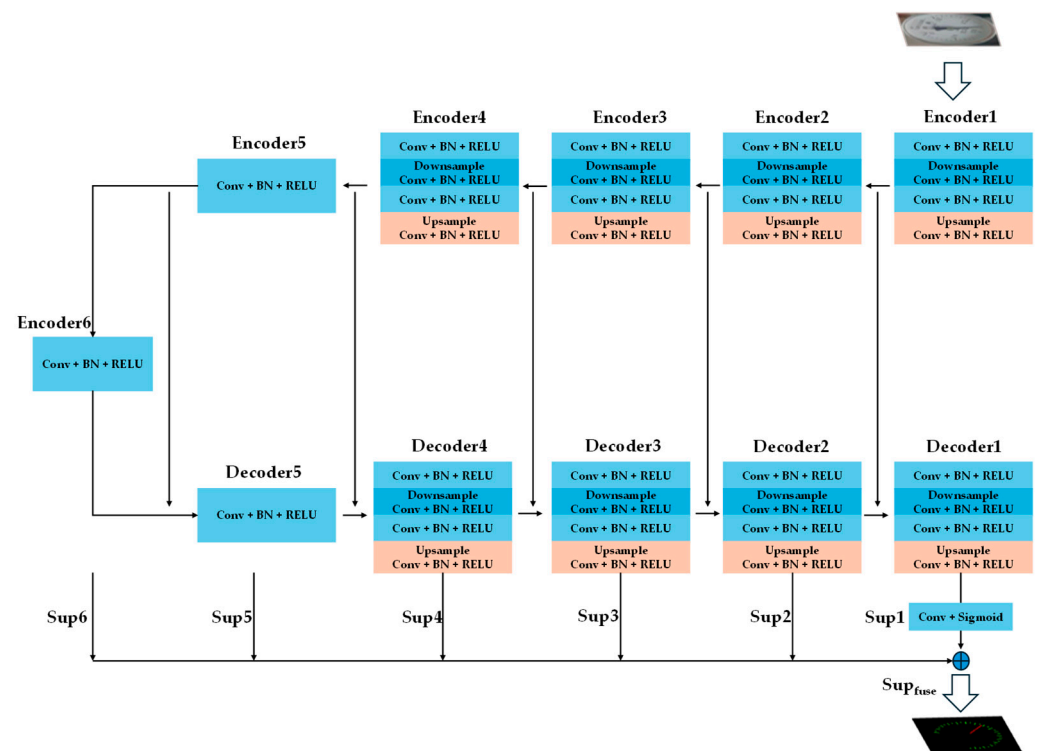


Figure 7. U2Net model.

2.4. Reading

This work achieves accurate localization of scale lines and pointer positions through structural decomposition of the dial. By transforming the dial region into polar coordinates, the spatial relationships between the pointer and scale markings are effectively quantified. Specifically, the center of the dial is defined as the origin O of the polar coordinate system, with the initial scale line serving as the reference axis (typically defined as $\theta = 0$). The position of the pointer tip is subsequently expressed in polar coordinates as (ρ, θ) , where ρ represents the magnitude of the pointer vector (i.e., the distance from origin O to the pointer tip), and θ denotes the angle between the pointer and the reference axis.

Given that the dial's angular range extends from $\theta_{\min}$ to $\theta_{\max}$, corresponding to a scale range of $S_{\min}$ to $S_{\max}$, the pointer reading S can be mathematically derived from the pointer angle θ through the following linear relationship, as follows:

$$S = S_{\min} + \frac{\theta - \theta_{\min}}{\theta_{\max} - \theta_{\min}} \times (S_{\max} - S_{\min}) \quad (2)$$

This method achieves precise conversion from pointer positions in polar coordinate space to physical readings through angular normalization and proportional mapping, enabling automated and high-precision measurement interpretation for pointer-based instruments.

Following the polar coordinate transformation of the image (as illustrated in Figures 8 and 9), the proportional analysis method computes the final reading based on the total number of extracted scale lines and the specific scale position indicated by the pointer. These two algorithms operate as complementary verification mechanisms: the polar coordinate method provides precise spatial localization of the pointer relative to the scale lines, while the proportional analysis method validates this reading using both the total scale count and the pointer's angular position. This dual approach significantly enhances measurement reliability and ensures robust, high-precision reading calculations.

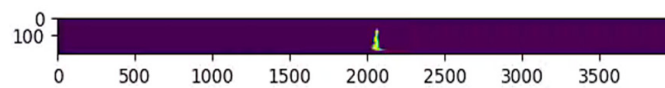


Figure 8. Polar coordinate projection diagram.

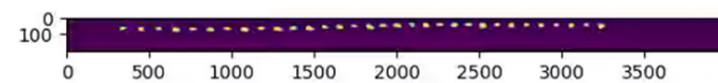


Figure 9. Scale Line Polar Coordinate Unfolding Diagram.

3. Results and Discussion

The trials were carried out using the following software and hardware configurations: an Intel Core i5-9400 CPU @ 2.90 GHz (6 cores), an NVIDIA GeForce RTX 3090 GPU (24 GB VRAM), and 32GB system memory running Windows OS. The development platform was constructed using PyCharm2022.2.5 and OpenCV-Python 4.5.0.80, while the deep learning environment used Python 3.8.0, CUDA 11.0, and PyTorch 2.1.1 + cu110. Data augmentation tactics, such as blurring and inversion, were used to imitate difficult real-world conditions. The final dataset had 8000 images partitioned in an 8:1:1 ratio for training, validation, and testing. The model was trained with batch size = 16, lr = 0.01, image size = 640 × 640, and epochs = 200.

This experiment evaluates the performance of the improved algorithm using multiple metrics. First, the Box metric is calculated based on the Intersection over Union (IoU) between predicted bounding boxes and ground truth bounding boxes. Recall measures the proportion of actual positive samples correctly identified by the model. Higher recall indicates fewer missed detections. Recall is typically evaluated alongside precision to provide a more comprehensive assessment of the model's performance. IoU (Equation (3)) is defined as the ratio of the intersection area between the predicted and ground truth bounding boxes to their union area.

$$IoU = \frac{\text{Area}(B_{pred} \cap B_{true})}{\text{Area}(B_{pred} \cup B_{true})} \quad (3)$$

Equation (4) measures the model's accuracy in target localization, with higher values indicating greater precision. Detection accuracy is reflected through mAP (mean Average Precision), where mAP50 represents the average precision at an IoU threshold of 0.5. The calculation process involves generating a Precision-Recall curve and computing its area under the curve (AUC), then averaging across all categories:

$$mAP = \frac{1}{N} \sum_{i=1}^N AP_i \quad (4)$$

The Equation (5) metric calculates the average AP across IoU thresholds ranging from 0.5 to 0.95 (with a step size of 0.05):

$$mAP_{50:95} = \frac{1}{T} \sum_{t=1}^T AP_{IoU=t} \quad (5)$$

The computational complexity and scale of a model are measured by two metrics: FLOPS (floating-point operations per second, unit: GFLOPs) and the number of parameters P (unit: M). FLOPS calculations traverse all layers of the model, incrementally summing the number of floating-point operations per layer based on the computational characteristics of different operations. Parameter count P quantifies the total number of trainable parameters in the model, encompassing convolutional kernel weights, biases, and other elements. This count is calculated by summing the total number of elements in each layer's weight matrix, reflecting the model's storage requirements and computational complexity.

Inference speed is expressed in FPS (frames per second), Equation (6) calculated as the reciprocal of the time required to process a single image:

$$FPS = \frac{1}{t_{inference}} \quad (6)$$

Higher FPS translates to better real-time performance of the model.

The F1 score is a metric for evaluating the performance of a classification model, combining both precision and recall, and is calculated using the following Formula (7):

$$F1 - score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (7)$$

Precision is accuracy and Recall is recall, which helps improve the model's accuracy and comprehensiveness.

Finally, the number of parameters represents the total count of all trainable parameters in the model, typically measured in millions. This metric reflects the model's scale and complexity; fewer parameters result in a lighter model, facilitating deployment.

In summary, these measures assess the algorithm's strengths and weaknesses across various dimensions, including localization accuracy, detection performance, computational resources, inference speed, and model scale, to ensure a good balance of accuracy and efficiency.

3.1. Comparative Experiments

The performance of the proposed method is assessed using a wide range of criteria. Table 2 shows a complete performance comparison of the proposed enhanced method with numerous mainstream detection techniques.

Table 2. Comparative Test.

	map50 (%)	map50–95 (%)	P (%)	R (%)	F1 (%)	FLOPS (GFLOPs)	FPS (fps)	Params (M)
YOLOv5	0.951	0.736	0.895	0.951	0.920	3.588	72.266	2.509
YOLOv8	0.952	0.735	0.893	0.940	0.916	4.097	83.062	3.011
Faster R-CNN	0.982	0.643	0.685	0.742	0.710	249.670	33.524	136.691
YOLOv11	0.952	0.739	0.896	0.971	0.931	3.219	74.821	2.587
YOLOv12	0.958	0.743	0.898	0.972	0.933	3.240	53.751	2.568
CASCA-YOLOv8	0.961	0.745	0.903	0.976	0.936	3.798	84.242	2.325

The mAP50 and mAP50-95 scores are critical metrics for evaluating the performance of object identification models since they compare the model's accuracy across different Intersection over Union (IoU) thresholds. CASCA-YOLOv8 beats all other models, with an exceptional mAP50 of 0.961 and mAP50-95 of 0.745, indicating better overall detection ability. This outstanding performance is even more evident when compared to YOLOv8 (mAP50: 0.952, mAP50-95: 0.735), YOLOv11 (mAP50: 0.952, mAP50-95: 0.739), and YOLOv12 (mAP50: 0.958, mAP50-95: 0.743). These results demonstrate CASCA-YOLOv8's consistent and reliable detection capacity, particularly when recognizing objects with varied levels of overlap.

In addition to mAP scores, Precision (P) and Recall (R) metrics offer a more complete picture of the model's capacity to properly identify objects and discover all relevant targets. CASCA-YOLOv8 outperforms with the greatest Precision (0.903) and Recall (0.976) scores in this comparison, demonstrating a remarkable balance between limiting false positives and increasing object detection. These results not only outperform YOLOv8 (P: 0.893, R: 0.940) and YOLOv12 (P: 0.898, R: 0.972), but also highlight CASCA-YOLOv8's strong ability to recognize items with high accuracy and thoroughness.

From a computational complexity standpoint, FLOPS (floating-point operations per second) and parameter count are important markers of model efficiency. CASCA-YOLOv8 uses 3.798 GFLOPs, which is much less than Faster R-CNN (249.670 GFLOPs) and YOLOv12 (3.240 GFLOPs), indicating a more efficient model. Furthermore, CASCA-YOLOv8 has a lower parameter count of 2.325 million than Faster R-CNN (136.691 million) and YOLOv8 (3.01 million). This makes CASCA-YOLOv8 a lighter model while maintaining good performance, demonstrating a successful balance between model size and computational capability.

In conclusion, CASCA-YOLOv8 achieves an ideal blend of high accuracy, computational efficiency, and quick inference speed, making it a good candidate for real-time object detection applications. The model outperforms earlier models, including the YOLOv5, Faster C-NN, YOLOv8, YOLOv11, and YOLOv12, in numerous key areas such as mAP, Precision, and FPS, while also providing a more compact and efficient design. CASCA-YOLOv8 retains good detection performance even with fewer parameters, demonstrating its suitability for deployment in resource-constrained situations such as edge devices and mobile apps that require computing economy and speed.

Finally, the constancy of Precision and Recall rates, notably the high Recall performance, illustrates CASCA-YOLOv8's supremacy in comprehensive object detection. The model's durability and training repeatability are further validated by the constant convergence curves (Figure 10), which reflect the efficacy of its design and optimization. These characteristics—consistent performance improvement paired with a lightweight design—make CASCA-YOLOv8 ideal for practical deployment, assuring the best combination of detection accuracy and operational economy.

Furthermore, the consistent convergence curves in Figure 10 demonstrate the training repeatability and robustness of all evaluated models.

The confusion matrix of the model is shown in Figure 11. The CASCA-YOLOv8 model's performance is thoroughly tested and compared to numerous existing models, including Faster R-CNN, YOLOv5, YOLOv8, YOLOv11, and YOLOv12. Confusion matrices are used. As demonstrated by the data, CASCA-YOLOv8 exceeds all other models in terms of detection accuracy, with a stunning detection rate of 0.97. Furthermore, it dramatically reduces the background detection rate to 0.03, the lowest of the models tested. This illustrates that CASCA-YOLOv8 improves object detection precision while also reducing false positives, making it a very efficient and reliable technique for real-world applications. These enhancements show CASCA-YOLOv8's supremacy in handling complicated detection tasks, indicating a substantial advancement in object recognition technology.

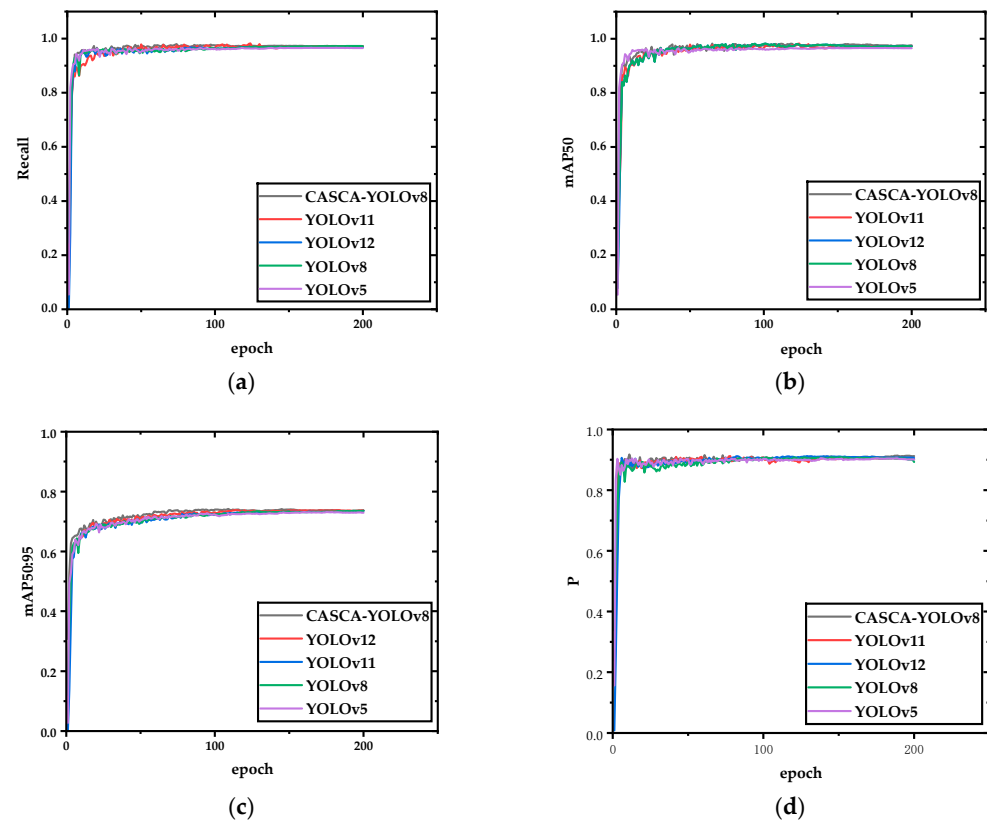


Figure 10. Curve Comparison (a) Recall (b) map50 (c) map50-95 (d) Precision.

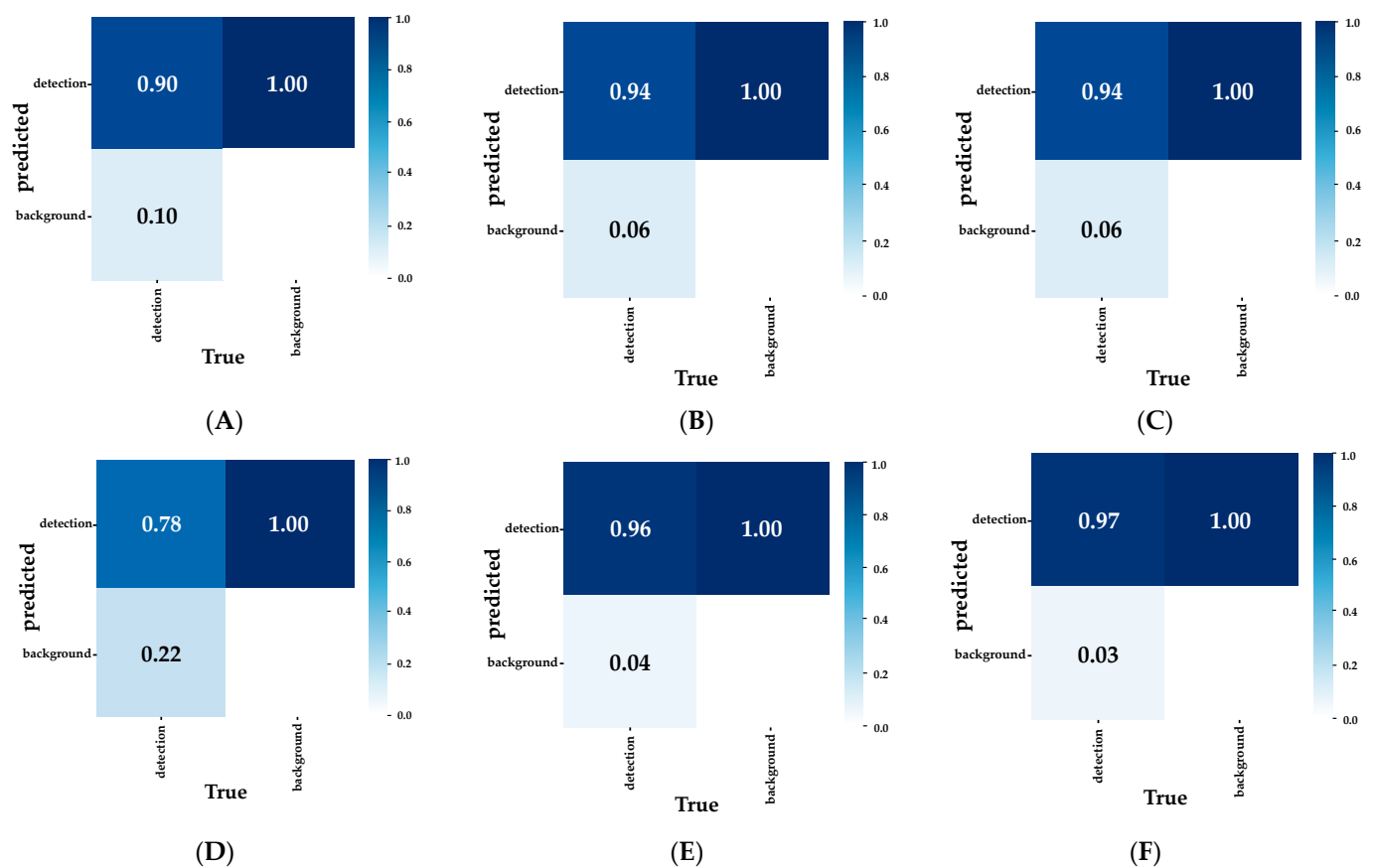


Figure 11. Confusion Matrix (A) YOLOv5; (B) YOLOv8; (C) YOLOv11; (D) Faster R-CNN; (E) YOLOv12; (F) CASAC-YOLOv8.

In summary, CASCA-YOLOv8 achieves an ideal balance between detection accuracy, inference speed, and model compactness, surpassing YOLOv5 and Faster R-CNN, and even outperforming previous YOLOv8 variations, YOLOv11, and YOLOv12. This makes it ideal for applications that require both high real-time performance and precise detection capabilities.

3.2. Ablation Studies

Table 3 summarizes ablation studies assessing proposed improvements. Component A relates to CASCA mechanism integration, component B to network structure pruning, and component C to modified Ghost neural network convolutions.

Table 3. Ablation Experiment.

	map50 (%)	map50-95 (%)	P (%)	R (%)	F1 (%)	FLOPS (GFLOPs)	FPS (fps)	P (M)
YOLOv8	0.952	0.735	0.893	0.940	0.916	4.097	83.062	3.011
YOLOv8 + A	0.961	0.742	0.900	0.968	0.931	4.097	70.941	3.020
YOLOv8 + A + B	0.965	0.746	0.902	0.956	0.928	3.912	83.450	2.559
YOLOv8 + A + B + C	0.961	0.745	0.903	0.976	0.936	3.798	84.242	2.325

Ablation investigations demonstrated independent and cumulative contributions to each improvement. Module A (the CASCA method) improves feature discriminability, increasing mAP50 from 0.952 to 0.961 and mAP50-95 from 0.735 to 0.742. However, it increases computational burden, lowering inference speed from 83.062 to 70.941 FPS. Module B (Network Pruning) significantly eliminates structural redundancy, raising mAP50 to 0.965 and mAP50-95 to 0.746 while increasing inference speed to 83.450 FPS, exceeding the initial baseline. Module C (GhostConv) added efficient feature transformations. The lightweight convolutions sacrificed accuracy to reduce computational load, allowing the algorithm to attain the fastest inference speed of 84.242 FPS while maintaining high precision (mAP50: 0.961, mAP50-95: 0.745). The synergistic integration of these components allows CASCA-YOLOv8 to attain an ideal combination of detection capability and computing economy, as demonstrated by the consistent training curve in Figure 12.

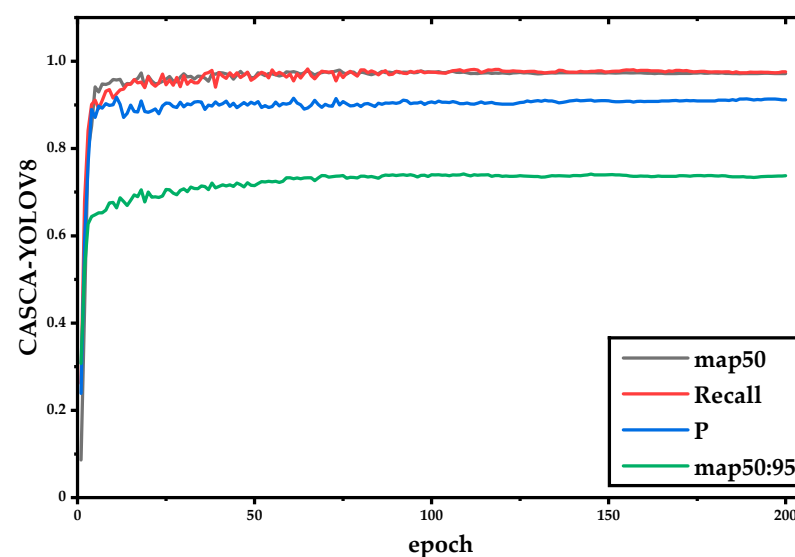


Figure 12. Training Parameter Curve.

3.3. Visualization

Figure 13 shows the results of object detection with our modified CASCA-YOLOv8 method. Each figure depicts the detection of pointer-type instruments, with the bounding boxes and confidence ratings highlighted in blue. The system performs well across a variety of tough settings, with detection confidence ratings ranging from 0.7 to 0.9. The results reveal that the model can efficiently identify and location the instruments even in a variety of situations and orientations. These visual outputs demonstrate the model's capacity to reliably detect items in real-world scenarios, demonstrating its dependability and efficiency for practical use.

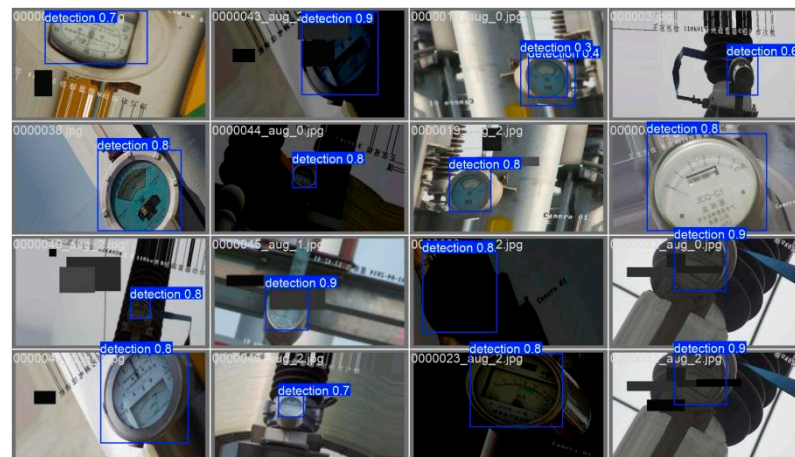


Figure 13. Object detection results.

When no Hough transform-based line fitting is used, the initial segmentation results (Figure 14a) show discontinuous and incomplete scale lines. These flaws impair the correct counting of scale lines, jeopardizing the dependability of final reading estimates. In comparison, the optimized findings (Figure 14b) show much enhanced continuity and completeness of scale lines after incorporating the Hough transform. Figure 15 also compares segmentation results with and without the instrument panel background, demonstrating the robustness of the suggested technique under different visual situations.

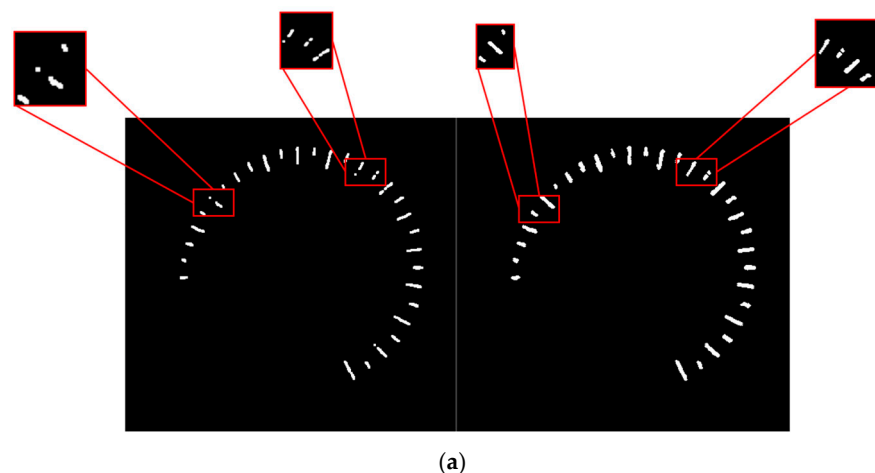


Figure 14. Cont.

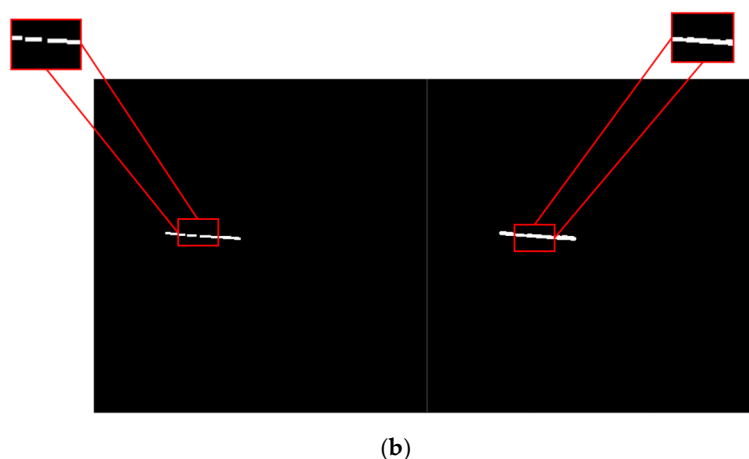


Figure 14. Comparison of scale and pointer images before and after U2Net-Hough transform processing. (a) The original scale and new scale; (b) the original pointer and pointer image.

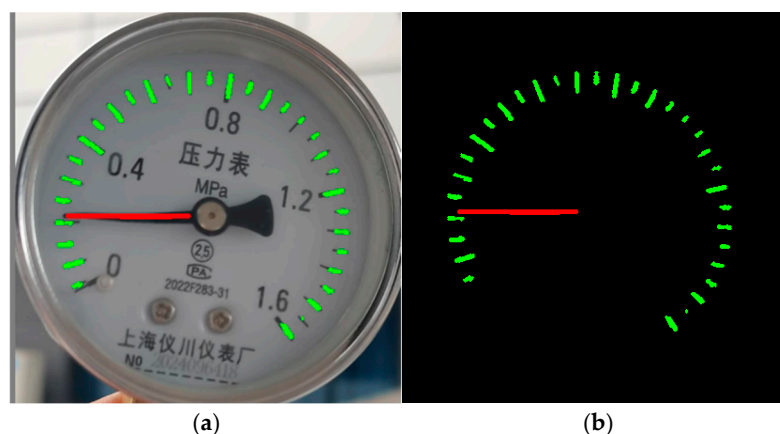


Figure 15. Final result. (a) With the instrument panel background; (b) without the instrument panel background.

3.4. Readings

As shown in Figure 16, the experimental findings confirm that the proposed semantic segmentation model is stable and resilient in practical application scenarios.



Figure 16. Reading results.

4. Conclusions

This research presents an efficient recognition framework for automatic reading of pointer-type instruments using an upgraded YOLOv8 architecture. The proposed CASCA technique, which combines context-aware channel and spatial attention, considerably enhances feature discriminability and detection accuracy by allowing for a more exact localization of pointer positions. Through reasonable modification of the backbone network,

including the removal of one C2f module, the model achieves a significant boost in inference speed while keeping competitive detection performance, significantly improving its suitability for real-time applications. Furthermore, by combining Hough transform-based line fitting into the U2Net segmentation process, the approach achieves more detailed extraction of scale lines and pointer bounds, ensuring accurate numerical reading calculations. The integrated pipeline offers a comprehensive end-to-end solution, from instrument detection to reading output, that successfully balances computing efficiency and measurement precision.

Author Contributions: M.Z. conceived the methodology, conducted experiments, and drafted the original manuscript. B.X. contributed to methodological design, manuscript review and editing. S.S. Supervised the project, acquired funding, provided resources and critical review. Z.T. established the experimental environment and platform infrastructure. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Key Research and Development Program of Guangxi, (Grant No. AB24010110).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Li, X.; Zhao, J.; Zeng, C.; Yao, Y.; Zhang, S.; Yang, S. Deep Learning-Based Pointer Meter Reading Recognition for Advancing Manufacturing Digital Transformation Research. *Sensors* **2025**, *25*, 244. [\[CrossRef\]](#) [\[PubMed\]](#)
2. Li, Z.; Zhou, Y.; Sheng, Q.; Chen, K.; Huang, J. A High-Robust Automatic Reading Algorithm of Pointer Meters Based on Text Detection. *Sensors* **2020**, *20*, 5946. [\[CrossRef\]](#)
3. Chang, C.-Y.; Shie, H.-C. Application of machine learning to classify surface marker of screw. In Proceedings of the 2016 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW), Nantou, Taiwan, 27–29 May 2016; pp. 1–2. [\[CrossRef\]](#)
4. Saeid, S.; Jonathon, A.C. Machine Learning for EEG Analysis. In *EEG Signal Processing and Machine Learning*; Wiley: Hoboken, NJ, USA, 2021; pp. 177–233. [\[CrossRef\]](#)
5. Zhu, Z.; Gao, Z.; Zhuang, J.; Huang, D.; Huang, G.; Wang, H.; Pei, J.; Zheng, J.; Liu, C. MSMT-RTDETR: A Multi-Scale Model for Detecting Maize Tassels in UAV Images with Complex Field Backgrounds. *Agriculture* **2025**, *15*, 1653. [\[CrossRef\]](#)
6. Xu, X.; Wang, L.; Deng, C.; He, B. Pointer Meter Reading Recognition Based on YOLOv11-OBb Rotated Object Detection. *Appl. Sci.* **2025**, *15*, 7460. [\[CrossRef\]](#)
7. Wu, X.; Shi, X.; Jiang, Y.; Gong, J. A High-Precision Automatic Pointer Meter Reading System in Low-Light Environment. *Sensors* **2021**, *21*, 4891. [\[CrossRef\]](#)
8. Alexeev, A.; Kukharev, G.; Matveev, Y.; Matveev, A. A Highly Efficient Neural Network Solution for Automated Detection of Pointer Meters with Different Analog Scales Operating in Different Conditions. *Mathematics* **2020**, *8*, 1104. [\[CrossRef\]](#)
9. Lu, F.; Guo, N.; Xue, M.; Ni, H.Y.; Huang, Y.J. Automatic reading recognition algorithm for multi-category pointer meters based on lightweight network. *Eng. Res. Express* **2025**, *7*, 035270. [\[CrossRef\]](#)
10. Li, F.; Zhang, W.; He, B. Pointer instrument positioning and indication recognition algorithm based on YOLOV5s. *J. Phys. Conf. Ser.* **2024**, *2724*, 012054. [\[CrossRef\]](#)
11. Huo, Y.; Bai, H.; Wu, X.; Wang, J. A Detection and Recognition Method for Pointer-Type Instruments Based on CBAM-YOLOv3. *J. Nat. Sci. Heilongjiang Univ.* **2023**, *40*, 621–630. [\[CrossRef\]](#)
12. Salaudeen, H.; Çelebi, E. Pothole Detection Using Image Enhancement GAN and Object Detection Network. *Electronics* **2022**, *11*, 1882. [\[CrossRef\]](#)
13. Lu, T. Research on Correction and Recognition of Analog Gauges Based on SIFT Feature Point Matching. *Ind. Control Comput.* **2025**, *38*, 7–10.

14. Ni, P.; Mao, P. Reading recognition method of pointer instrument based on YOLOv8+U2-Net. In Proceedings of the 2024 6th International Conference on Internet of Things, Automation and Artificial Intelligence (IoTAAI), Guangzhou, China, 26–28 July 2024; pp. 664–668. [\[CrossRef\]](#)
15. Li, J.; Amin, A.M.; Shao, Z. Recognition of Pointer Meter Readings Based on YOLOv8 and DeepLabv3+. *J. Comput. Commun.* **2025**, *13*, 15–25. [\[CrossRef\]](#)
16. Lü, X.R.; Lai, B.; Zhou, J. Design of Intelligent Recognition Method for Pointer Meters Based on Improved DeepLabV3+. *Electron. Des. Eng.* **2024**, *32*, 145–149+154. [\[CrossRef\]](#)
17. Sui, Y.; Yin, M.; Gong, Y.; Yuan, B. Co-Exploring Structured Sparsification and Low-Rank Tensor Decomposition for Compact DNNs. *IEEE Trans. Neural Netw. Learn. Syst.* **2025**, *36*, 6642–6654. [\[CrossRef\]](#) [\[PubMed\]](#)
18. Enhao, C.; Hao, W.; Zhang, L.; Wei, Z. Channel pruning for convolutional neural networks using l0-norm constraints. *Neurocomputing* **2025**, *636*, 129925. [\[CrossRef\]](#)
19. Wang, X.Y.; Hu, Y.H.; Yang, Z.W. Neural network quantization: Separate scaling of rows and columns in weight matrix. *Neural Comput. Appl.* **2025**, *37*, 1417–1428. [\[CrossRef\]](#)
20. Liu, J.; Huang, L.; Feng, M.; Guo, A.; Yin, L.; Zhang, J. IESSP: Information Extraction-Based Sparse Stripe Pruning Method for Deep Neural Networks. *Sensors* **2025**, *25*, 2261. [\[CrossRef\]](#)
21. Liu, W.B.; Qiao, X.Y.; Zhao, C.Y.; Deng, T.; Yan, F. A new pipeline with ultimate search efficiency for neural architecture search. *Neural Netw.* **2025**, *185*, 107163. [\[CrossRef\]](#)
22. Hang, L.; Peng, Y.F.; Zhang, Y.B.; Lin, B.; Zhao, X.B.; Gao, Y. Filter Pruning by High-Order Spectral Clustering. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*; IEEE: New York, NY, USA, 2025; Volume 47, pp. 2402–2415.
23. Huang, D.; Zhu, C. Strong Wind Data Augmentation Based on F-DBSCAN and C-SMOTE for Wind Power Forecasting. In Proceedings of the 2021 IEEE 5th Conference on Energy Internet and Energy System Integration (EI2), Taiyuan, China, 22–24 October 2021; pp. 2992–2998. [\[CrossRef\]](#)
24. Jakubović, A.; Velagić, J. Image Feature Matching and Object Detection Using Brute-Force Matchers. In Proceedings of the 2018 International Symposium ELMAR, Zadar, Croatia, 16–19 September 2018; pp. 83–86.
25. Xiu, J.; Hou, F.; Li, D.; Guo, S.; Bai, Y.; Zhang, J. Feature Point Identification of Hydropower Plant Based on Improved ORB Algorithm. In Proceedings of the 2023 5th International Conference on Intelligent Control, Measurement and Signal Processing (ICMSP), Chengdu, China, 19–21 May 2023; pp. 1108–1112. [\[CrossRef\]](#)
26. Rublee, E.; Rabaud, V.; Konolige, K.; Bradski, G. ORB: An efficient alternative to SIFT or SURF. In Proceedings of the 2011 International Conference on Computer Vision, Barcelona, Spain, 6–13 November 2011; pp. 2564–2571. [\[CrossRef\]](#)
27. Chang, C.Y.; Khanum, A.; Sun, C.Y.; Chen, Y.T.; Tsai, Y.C.; Hsu, C.C. Generating Parathyroid Reports Using YOLO-Based Large Language Models. *IEEE Access* **2025**, *13*, 137579–137591. [\[CrossRef\]](#)
28. Zheng, J.; Ning, J.X.; Liu, Z.Y. Object Detection Algorithm for Remote Sensing Images Based on Improved YOLOv8. *J. Shenyang Inst. Technol.* **2025**, *44*, 52–58. [\[CrossRef\]](#)
29. Qi, P.C.; Yuan, J.; Rasol, J.; Song, C.; Zhang, N.N.; Zhu, L.L. Design and Implementation of a Lightweight Real-Time Detection Model for Apple Ripeness (GCA-YOLOv8n). *J. S. China Agric. Univ.* **2026**, *47*, 128–138. [\[CrossRef\]](#)
30. Chang, C.Y.; Khanum, A.; Su, S.G.; Lebaku Moses, M.; Liu, K.X. Vertical-Line Mura Defect Detection for TFT-LCDs. *IEEE Access* **2024**, *12*, 158927–158938. [\[CrossRef\]](#)
31. Chen, X.B.; Liu, C.; Wang, S.; Deng, X. LSI-YOLOv8: An Improved Rapid and High Accuracy Landslide Identification Model Based on YOLOv8 From Remote Sensing Images. *IEEE Access* **2024**, *12*, 97739–97751. [\[CrossRef\]](#)
32. Mei, X.; Zhu, Z.; Sun, M.; Ma, Y. Automatic Channel Pruning of U-Net Applied to Retinal Blood Vessels Segmentation. In Proceedings of the 2024 43rd Chinese Control Conference (CCC), Kunming, China, 28–31 July 2024; pp. 7681–7686. [\[CrossRef\]](#)
33. Huang, D.; Chen, Z.; Zhuang, J.; Song, G.; Huang, H.; Li, F.; Huang, G.; Liu, C. DRPU-YOLO11: A Multi-Scale Model for Detecting Rice Panicles in UAV Images with Complex Infield Background. *Agriculture* **2026**, *16*, 234. [\[CrossRef\]](#)
34. Qin, X.B.; Zhang, Z.C.; Huang, C.Y.; Dehghan, M.; Osmar, R.Z.; Jagersand, M. U2-Net: Going Deeper with Nested U-Structure for Salient Object Detection. *Pattern Recognit.* **2020**, *106*, 107404. [\[CrossRef\]](#)
35. Sunil, B.; Meenakashi, S.; Amod, K. Diagnosis of contrast improved common maize leaf diseases using U2Net based deep learning model. *Procedia Comput. Sci.* **2025**, *258*, 679–688. [\[CrossRef\]](#)
36. Ding, C.; Zhang, J.; Hu, Z.H.; Zhu, H.D.; Luo, Z.D. Hough transform guided subpixel edge regression for high precision relocalization of nut geometric parameters. *Precis. Eng.* **2026**, *97*, 527–546. [\[CrossRef\]](#)

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.