Comparative Analysis of Real-Time Detection Models for Intelligent Monitoring of Cattle Condition and Behavior

Oleg Ivashchuk , Zhanat Kenzhebayeva * , Alexey Zhigalov , Moldir Allaniyazova , Gulnara Kaziyeva , Kaiyrbek Makulov , Vyacheslav Fedorov , Olga Ivashchuk

Posted Date: 28 October 2025

doi: 10.20944/preprints202510.2158.v1

Keywords: benchmarking; object detection; agriculture; real-time; RTMDet; D-FINE; YOLOv11; MMDetection

 doi:10.20944/preprints202510.2158.v1

Article

Comparative Analysis of Real-Time Detection Models for Intelligent Monitoring of Cattle Condition and Behavior

Oleg Ivashchuk [1], Zhanat Kenzhebayeva [1,*], Alexey Zhigalov [2], Moldir Allaniyazova [1], Gulnara Kaziyeva [1], Kaiyrbek Makulov [1], Vyacheslav Fedorov [3] and Olga Ivashchuk [3]

[1] Department of Computer Science, Faculty of Science and Technology, Caspian University of Technology and Engineering (KUTI) named after Sh. Yessenov, micro district 32, Aktau, 130000, Republic of Kazakhstan

[2] Individual Entrepreneur Zhigalov A.A. (neural.dev), Agmashenebeli str., 1, Batumi, 6004, Adjara, Georgia

[3] Department of Information and Robotic Systems, Belgorod State National Research University, 85, Pobedy St., Belgorod, 308015, Russia

* Correspondence: zhanat.kenzhebayeva@yu.edu.kz; Tel.:+7(701)125-61-23

Abstract

The present study provides a systematic benchmarking of nine state-of-the-art object detection models adapted to a specialized animal dataset. The main objective was to evaluate their performance in terms of accuracy and inference speed in the context of real-time and commercial applications within the agro-industrial sector. The methodology involved aggregating and thoroughly cleaning several cattle image datasets, followed by fine-tuning models with diverse architectures — two-stage, one-stage, and transformer-based. Performance evaluation was conducted on a unified hardware platform equipped with an NVIDIA RTX 4090 GPU with 24 GB of VRAM, which is critical for the objective assessment of speed-related metrics. Key results show that the D-FINE and Co-DETR models demonstrated the highest accuracy (AP@[IoU=0.50:0.95] of 0.872 and 0.851, respectively), while RTMDet and YOLOv11 proved to be the fastest (15.81 ms/image and 19.14 ms/image in reproducible mode). A general trend was observed toward significant improvement in average precision (AP) metrics on the specialized dataset compared to their known results on general-purpose datasets such as COCO, while relative inference speed rankings remained largely consistent.

Keywords: benchmarking; object detection; agriculture; real-time; RTMDet; D-FINE; YOLOv11; MMDetection

1. Introduction

1.1. Relevance of the Problem

Modern agriculture is undergoing intensive integration of digital technologies into all aspects of business and production processes. In this context, computer vision and deep learning models play a crucial role, enabling highly accurate, non-invasive object detection — an essential component in livestock management. Automated intelligent animal monitoring systems make it possible not only to accurately count livestock but also to track their health, analyze behavior and growth rates, optimize feeding schedules, and detect diseases in time.

For example, computer vision systems can detect early signs of illness or injury — such as nasal discharge or eye problems — allowing farmers to take timely action and prevent disease outbreaks [1]. Monitoring animal movement and interactions helps identify signs of stress or discomfort [2]. Moreover, drones equipped with cameras provide effective aerial surveillance of livestock across large fields, helping track grazing patterns and locate missing animals [3].

 doi:10.20944/preprints202510.2158.v1

For all these applications, detection capability is of critical importance, as it directly affects operational efficiency, economic viability, and animal welfare. However, this domain presents several unique challenges: significant variation in animal sizes (from small objects in drone footage to close-up large animals), frequent occlusions within herds, complex dynamic natural backgrounds, and a strict need for real-time processing for practical farm management decisions.

Despite the widespread use of deep learning models, there is still a noticeable lack of systematic comparative benchmarks focused on real-time commercial performance for domain-specific agricultural datasets. This gap is especially critical under the constraints of real-time operation and commercial deployment, which require balancing accuracy, inference speed, resource efficiency, and licensing considerations.

Thus, conducting a systematic benchmarking of modern open-source object detectors on a specialized animal dataset to evaluate detection performance (accuracy and inference speed) in real-time and commercial contexts is a highly relevant and timely research task.

1.2. Research Problem Statement

Systematic benchmarking plays a fundamental role in computer vision, as it enables objective evaluation of model performance and guides their practical deployment. However, general-purpose benchmarks such as COCO (Common Objects in Context, Microsoft) often prove insufficient for domain-specific applications [4]. This limitation arises from differences in object characteristics (e.g., number of classes, unique shapes), scene complexity, and performance requirements (e.g., strict timing constraints, accuracy thresholds) in specialized domains such as agriculture.

As mentioned above, in agricultural settings, objects may be very small or very large, frequently occluded, and appear against backgrounds that differ substantially from urban or domestic scenes represented in COCO. Therefore, directly applying models trained on general datasets without domain adaptation and specialized benchmarking can lead to inadequate performance in real-world agricultural conditions.

The goal of this study is to conduct a systematic benchmarking of nine modern open-source object detection architectures on a specialized cattle dataset, evaluating performance in terms of accuracy and inference speed under real-time and commercial conditions.

We hypothesize that the accuracy of detectors fine-tuned on our dataset will generally correlate with their known performance on public datasets (e.g., COCO), though deviations may occur due to the domain-specific nature of the cattle dataset. In contrast, inference speed is expected to remain consistent with previously reported values.

The study will analyze the performance of various architectural paradigms — two-stage, one-stage, and transformer-based — on a real, domain-specific dataset. Special attention is given to the trade-off between accuracy and inference speed, which is critical for practical real-time applications. Domain adaptation aspects affecting the generalization capability of models in agricultural environments are also considered.

The expected outcomes — a unique set of empirical benchmarks and practical recommendations — will provide tangible value for stakeholders in the agro-industrial sector, helping them make informed decisions when selecting suitable object detection models for specific needs. Detailed documentation of the experimental setup and methodology underscores the importance of research reproducibility.

2. Overview of Existing Object Detectors

2.1. General Information on Object Detectors

The evolution of object detection has progressed from traditional image processing methods to modern deep learning approaches, which have significantly enhanced model performance. Contemporary object detection systems, known for their high accuracy and promising results, can identify objects of interest in images and videos. The unique capability of convolutional neural

 doi:10.20944/preprints202510.2158.v1

networks (CNNs) to emulate human vision has attracted great attention, establishing CNNs as the foundation of most modern computer vision systems [5].

Modern object detection frameworks are typically divided into three main paradigms: two-stage, one-stage, and transformer-based detectors.

2.1.1. Two-Stage Detectors

Two-stage models, such as those from the R-CNN family, operate in two steps [6]: first, they generate region proposals likely to contain objects, and then classify and refine the bounding boxes. This approach traditionally provides high accuracy but is slower due to its sequential processing.

Cascade R-CNN represents a multi-stage extension of the base R-CNN architecture, designed to improve accuracy for high Intersection over Union (IoU) thresholds used in evaluating detection quality [7]. The model employs a chain of detectors trained sequentially, with each subsequent stage applying stricter criteria to filter out false positives close in IoU to true objects.

The key idea behind Cascade R-CNN is that the output of one stage serves as the input for the next, allowing the model to progressively improve localization accuracy at each step. This cascaded structure enhances robustness and overall detection reliability.

Cascade R-CNN is part of the MMDetection library, which is licensed under Apache-2.0 [8]. This open-source license permits free use, modification, and distribution of the software for personal, scientific, and commercial purposes.

2.1.2. One-Stage Detectors

One-stage detectors simplify the detection pipeline by predicting object locations and categories in a single step, eliminating the need for region proposals typical of two-stage methods. Their single-pass structure makes them faster and more efficient, which is particularly important for real-time applications.

RetinaNet is a one-stage object detection model that effectively addresses the issue of foreground-background imbalance [9]. This problem is common in images containing many densely packed objects, such as aerial photographs or animal groups. RetinaNet's key contribution is the Focal Loss function [10], which reduces the influence of easily classified examples, enabling the model to focus on "hard" negative samples that typically cause false detections. Additionally, RetinaNet employs Feature Pyramid Networks (FPN) [11], a hierarchical architecture that enhances multi-scale object detection — crucial for identifying both small and large objects in close proximity.

RTMDet is a highly efficient detector optimized for real-time tasks, balancing inference speed and recognition accuracy [12]. It utilizes advanced convolutional blocks designed to reduce computational complexity without sacrificing prediction quality. RTMDet demonstrates strong scalability and can be adapted for related tasks such as segmentation and rotated object detection [13].

YOLOv11, one of the latest generations of the YOLO (You Only Look Once) family, focuses on achieving a balance between inference speed and accuracy [14]. Its architecture introduces three major modules:

C3k2 — a lightweight feature extraction block that reduces computational cost with minimal accuracy loss;

SPPF (Spatial Pyramid Pooling – Fast) — an enhanced feature aggregation module that helps the model detect objects at multiple scales;

C2PSA — a spatial attention mechanism that increases sensitivity to key image regions.

These improvements enable YOLOv11 to effectively handle images with dense object arrangements and diverse scales while maintaining high accuracy [15].

2.1.3. Transformer-Based Detectors

 doi:10.20944/preprints202510.2158.v1

This emerging paradigm leverages the attention mechanism from transformer architectures to capture global context across the image [16]. Transformer-based detectors currently define the cutting edge of 2D object detection, achieving state-of-the-art results.

DETR (Detection Transformer) introduced a groundbreaking approach by integrating transformers directly into the object detection pipeline [17]. DETR is an end-to-end deep learning architecture that takes an image as input and outputs a set of bounding boxes and class labels without relying on traditional components like anchor boxes or non-maximum suppression (NMS).

D-FINE is a powerful real-time transformer-based detector that redefines the bounding-box regression task in DETR-like models, significantly improving localization accuracy. It introduces two key components: Fine-grained Distribution Refinement (FDR) and Global Optimal Localization Self-Distillation (GO-LSD), which enhance model precision without requiring prolonged training [18].

DINO (DETR with Improved Denoising Anchor Boxes and Mixed Query Prediction) is a high-performance variant of the DETR family, modernizing the original architecture with improved query initialization, multi-scale feature integration, and iterative refinement [19]. These enhancements lead to superior accuracy. In this study, DINO is considered as a fixed-class detector, not in its open-vocabulary form as seen in Grounding DINO.

RT-DETR (Real-Time Detection Transformer), developed by Baidu, is designed for efficient real-time detection using a transformer-based architecture [20]. It employs a fully end-to-end design without post-processing, simplifying and accelerating inference. Its hybrid encoder combines intra-scale feature interaction and cross-scale feature fusion, enabling effective detection of both small and large objects.

Co-DETR extends the original DETR architecture by adding an auxiliary detection head — a separate prediction module inspired by traditional detectors such as Faster R-CNN and RetinaNet. This auxiliary head improves the handling of overlapping or small objects. Co-DETR also features an optimized loss function for more accurate matching between predictions and ground truth, enhancing convergence and precision in complex conditions such as occlusions, high object density, and visual noise [21].

3. Research Methodology

This section provides a detailed description of the research methodology, ensuring its reproducibility.

3.1. Data Preparation

For this study, several public datasets containing cattle images were collected and aggregated: “Cattle (Cows) Object Detection Dataset,” “Animals Detection Images Dataset,” “Cattle Breeds Dataset,” and “Bovine Cattle Images.” All redundant and irrelevant data (non-cow images) were removed, along with duplicates and mirrored duplicates. Manual label correction was performed to ensure high annotation accuracy.

As a result, 4,367 images containing 14,400 bounding box annotations were obtained.

To ensure dataset representativeness, the data were stratified by scene type (e.g., “small distant cows” and “dense herds”), allowing proportional representation of each group in training and validation sets.

All annotations were converted into COCO JSON format, compatible with MMDetection, with a strict validation of data integrity and file path consistency.

3.2. Hardware and Software Configuration

All experiments were conducted on the following documented hardware:
GPU: NVIDIA RTX 4090 (24 GB VRAM),
CPU: AMD Ryzen 9 9900X,
RAM: 64 GB.

 doi:10.20944/preprints202510.2158.v1

The software environment was configured in an isolated Conda environment running Linux with:

NVIDIA driver version 570.86.10,

CUDA Toolkit 12.8,

PyTorch 2.4.1,

MMDetection 3.3.0.

All dependencies were recorded in a requirements.txt file to ensure reproducibility.

3.3. Model Selection and Adaptation

Nine state-of-the-art object detection models were selected for comparative analysis:

Cascade R-CNN [22], RetinaNet [23], RTMDet [24], D-FINE [25], DETR [26], DINO [27], RT-DETR [28], Co-DETR [29], and YOLOv11 [30].

The main selection criterion was model size (20–70M parameters) — large enough for robust accuracy but lightweight enough for real-time and embedded applications.

MMDetection-based models (Cascade R-CNN, RetinaNet, RTMDet, DETR, DINO, RT-DETR, Co-DETR) were obtained from the official repository under the Apache-2.0 license, allowing free academic and commercial use.

YOLOv11 (from Ultralytics) is distributed under AGPL-3.0, requiring derivative works to remain open-source.

D-FINE and its checkpoints were obtained from the official repository (https://github.com/Peterande/D-FINE), also licensed under Apache-2.0.

All configuration files were modified to match the single-class detection task (“cow”), with:

num_classes = 1,

input resolution fixed at 1280×1280,

unified logging and checkpoint storage (work_dirs).

Unified augmentation strategy:

To ensure fairness, all model-specific augmentations were replaced with a shared minimal set:

Horizontal flip (p = 0.5),

Random rotation (−10° to +10°),

Filtering out objects smaller than 16×16 px.

This minimized variability between models while maintaining comparability, although it may have slightly reduced the potential accuracy of certain models (since aggressive augmentations often improve generalization).

3.4. Training Protocol

Each model underwent fine-tuning using its pretrained weights and adapted configurations.

Training duration (number of epochs) was adjusted individually until convergence.

Learning rates (LR) and LR schedules were based on MMDetection defaults, with occasional manual scaling (e.g., ×0.1 reduction for fine-tuning).

Optimization process:

Different optimizers were tested per model, and the best-performing settings were retained.

Training progress was monitored with Weights & Biases (WandB) integrated into MMDetection, allowing visualization of loss curves and validation metrics.

The final evaluation used the checkpoint with the highest boxAP score.

3.5. Model Performance Evaluation Protocol

Performance evaluation included two key aspects: accuracy and inference speed.

(1) Accuracy (AP) [31]:

Each model’s best checkpoint was evaluated using the COCOeval script on a 569-image test set (input size 1280×1280, no Test-Time Augmentation).

 doi:10.20944/preprints202510.2158.v1

Metrics recorded:
AP@[IoU=0.50:0.95] (main metric),
AP@[IoU=0.50],
AP for small, medium, and large objects,
Average Recall (AR).
(2) Inference Speed (Latency):
A custom benchmarking script measured latency (ms/image) on an NVIDIA RTX 4090 (FP32 precision, batch size = 1).
Warm-up runs were performed to stabilize performance before timing.
Both preprocessing and postprocessing times (resize + normalization) were included to reflect real-world conditions.
Two measurement modes were implemented:
Reproducible mode:
cudnn.benchmark=False, deterministic=True.
Prioritizes consistency for academic comparison.
Production-optimized mode:
cudnn.benchmark=True, deterministic=False.
Reflects realistic deployment performance for industrial systems.
Both latency values are reported in results — the first ensuring reproducibility, the second representing operational speed.

4. Results

This section presents the outcomes of a systematic benchmarking of nine object detection models on a specialized cattle dataset.
The results include both accuracy metrics (Average Precision, AP) and inference speed metrics (latency and FPS) for each model, along with comparisons to their reported performance on the COCO dataset.

4.1. Evaluation of Detection Accuracy and Inference Speed

The final accuracy and inference speed results for each model are summarized in Table 1.

Table 1. Accuracy and Inference Speed of Cattle Detection Models.

Model	AP@.5:.95 (mean±std)	AP@.50 (mean±std)	Speed (ms/img, R)	FPS (R)	Speed (ms/img, P)	FPS (P)	Params (M)
Cascade R-CNN R-50-FPN	0.772	0.926	35.83	27.91	39.81	25.12	69,15
Co-DETR R-50	0.851	0.966	280.94	3.56	283.67	3.53	64,45
DETR R-50	0.777	0.947	24.7	40.49	25.17	39.72	41.55
D-FINE-L	0.872	0.966	21.61	46.28	22.23	44.98	30.66
DINO R-50	0.819	0.949	77.99	12.82	79.11	12.64	47,54
RetinaNet R-50-FPN	0.751	0.941	26.11	38.3	28.38	35.24	36,33
RT-DETR R-50vd	0.722	0.843	30.12	33.2	30.67	32.61	42,78
RTMDet-m	0.773	0.944	15.81	63.24	15.73	63.56	24.66
YOLOv11_L	0.707	0.871	19.14	52.24	19.74	50.65	25.31

Note: The AP values are derived from the COCOeval evaluation results. The inference speed (ms/img) and frames per second (FPS) are reported for both "reproducible" (R) and "production-optimized" (P) modes.

 doi:10.20944/preprints202510.2158.v1

To enable a fair comparison, the total number of parameters for each model was also calculated, providing an additional measure of model complexity.

Accuracy was evaluated using Average Precision (AP) at multiple IoU thresholds, while inference speed was measured as the average time per image (ms/image) and frames per second (FPS) in two operation modes:

Reproducible (R): optimized for experimental reproducibility,

Production (P): optimized for real-world deployment performance.

4.2. Comparative Analysis

The analysis reveals that the D-FINE model achieved the highest accuracy among all tested detectors, reaching AP@[0.5:0.95] = 0.872 and AP@0.50 = 0.966.

This demonstrates its superior localization precision and high reliability in predictions.

Co-DETR also achieved strong results (AP@[0.5:0.95] = 0.851, AP@0.50 = 0.966), ranking second overall in terms of detection accuracy.

Other transformer-based models — DETR and DINO — showed good, though slightly lower, performance (AP@[0.5:0.95] = 0.777 and 0.819, respectively), likely due to their architectural characteristics and domain adaptation specifics.

In terms of inference speed under reproducible conditions, RTMDet demonstrated the lowest latency — 15.81 ms per image (63.24 FPS) — making it the fastest among all models.

YOLOv11 ranked second with 19.14 ms (52.24 FPS).

Both belong to the one-stage family, traditionally optimized for speed, and this is confirmed by the experimental results.

Despite its superior accuracy, D-FINE maintained impressive performance at 21.61 ms (46.28 FPS), making it particularly attractive for real-world applications requiring both precision and responsiveness.

Considering its balance of accuracy and speed, D-FINE emerges as the most well-rounded solution, combining top-tier precision with near real-time inference speed.

For applications where maximum speed is critical, RTMDet or YOLOv11 are preferable, as they provide acceptable accuracy with minimal latency.

Meanwhile, DETR and RetinaNet offer moderate inference times (~25–26 ms per image), suitable for use cases with less stringent real-time constraints.

However, Co-DETR (≈280.9 ms) and DINO (≈78.0 ms) were significantly slower, largely due to their more complex architectures and higher parameter counts, which reduce suitability for time-sensitive deployments.

4.3. Comparison with COCO Dataset

The proposed hypothesis suggested that the accuracy of object detectors on the specialized cattle dataset would correlate with their known performance on widely used benchmarks (e.g., COCO), with possible deviations due to domain specificity, while the inference speed should remain more stable.

When comparing the obtained AP@[0.5:0.95] values on the cattle dataset with the published AP metrics on COCO (for the same model versions and checkpoints), the following pattern is observed (here, COCO AP refers to the AP metric on the COCO dataset, and Cow AP refers to the AP metric on the cattle dataset — the domain-specific dataset):

Cascade R-CNN: COCO AP = 41 compared to Cow AP = 0.772 — significant increase in AP on the domain dataset;

RetinaNet: COCO AP = 39.5 compared to Cow AP = 0.751 — likewise, a substantial improvement;

RTMDet: COCO AP = 49.4 compared to Cow AP = 0.773 — notable increase;

D-FINE: COCO AP = 55.1 compared to Cow AP = 0.872 — the highest absolute improvement;

DETR: COCO AP = 39.9 compared to Cow AP = 0.777 — significant improvement;

 doi:10.20944/preprints202510.2158.v1

DINO: COCO AP = 50.1 compared to Cow AP = 0.819 — considerable improvement;

RT-DETR: COCO AP = 53.1 compared to Cow AP = 0.722 — improvement, though less pronounced compared to other transformer-based models;

Co-DETR: COCO AP = 54.8 compared to Cow AP = 0.851 — strong improvement;

YOLOv11: COCO AP = 53.4 compared to Cow AP = 0.707 — improvement, but relatively modest compared to others.

Overall, a strong positive correlation is observed: models that demonstrated high accuracy on COCO generally achieved high results on the cattle dataset.

However, the absolute AP values on the cattle dataset are significantly higher than on COCO.

This confirms the first part of the hypothesis regarding correlation, while also highlighting noticeable deviations in absolute values.

These deviations can be explained by domain-specific factors:

The cattle dataset contains only one object class, simplifying the classification task compared to the multi-class COCO dataset.

Additionally, cattle images tend to have less diverse backgrounds or more uniform poses, making detection easier than in COCO's complex and varied contexts.

Regarding inference speed, measurements in both the "reproducible" and "production" modes showed consistent similarity, confirming the reliability of the benchmarking protocol.

Comparison with the published COCO inference speeds is difficult due to hardware differences used in the original benchmarks (e.g., D-FINE and RT-DETR were tested on T4 GPUs).

Nevertheless, the relative ranking of model speeds generally remains consistent:

One-stage detectors (RTMDet, YOLOv11) remain the fastest;

Some transformer-based models (DINO, Co-DETR) are the slowest.

4.4. Qualitative Analysis

A qualitative evaluation was performed through visual inspection of detection results across several test images.

This analysis revealed not only quantitative differences but also qualitative aspects such as successful detections, false positives, and missed objects.

Visualizations showed that high-accuracy models like D-FINE and Co-DETR generally exhibited:

More precise bounding box localization,

Fewer false positives,

Better performance in dense herds and complex background scenes.

For example, D-FINE successfully detected even small, distant cattle, a key advantage for aerial imagery or drone-based monitoring.

Conversely, faster models such as YOLOv11 and RT-DETR, while highly efficient, occasionally missed small or partially occluded animals, and sometimes produced false positives on background elements resembling cattle (e.g., rocks, shadows, or machinery parts).

In dense herd scenes, some detectors struggled to distinguish individual animals, leading to overlapping bounding boxes or missed detections.

False detections typically arose from background textures similar to cows, while missed detections occurred under strong lighting, motion blur, or occlusion conditions.

This qualitative analysis underscores that while quantitative metrics (e.g., AP) are crucial, visual assessment provides valuable insight into model robustness under real-world conditions and helps identify the types of domain-specific errors each model tends to make.

5. Discussion

The conducted benchmarking of nine open-source object detection models on a specialized cattle dataset revealed substantial performance differences, highlighting the importance of domain adaptation for real-world commercial applications in the agro-industrial sector.

 doi:10.20944/preprints202510.2158.v1

5.1. Accuracy and Domain Adaptation

The results clearly demonstrate that all models achieved significantly higher accuracy (AP) on the cattle dataset compared to their reported performance on COCO, confirming the first part of the hypothesis — a strong correlation accompanied by noticeable deviations in absolute values.

This improvement is largely attributed to domain-specific characteristics:

The cattle dataset contains only one object class, simplifying classification compared to the multi-class COCO dataset;

The homogeneity of the objects (cows) and the less diverse visual context facilitate more effective fine-tuning.

This finding indicates that for narrowly focused agricultural tasks, even models that perform moderately on general-purpose datasets can achieve very high detection accuracy after appropriate domain adaptation.

5.2. Trade-off Between Accuracy and Speed

The analysis revealed diverse trade-offs between accuracy and inference speed across the evaluated models.

D-FINE, which achieved the highest accuracy, also demonstrated remarkably high speed, making it one of the most balanced models for applications that require both precision and real-time performance.

RTMDet and YOLOv11 confirmed their reputation as high-speed detectors, ideal for systems where throughput is the top priority — such as continuous livestock monitoring or drone-based video analytics.

Transformer-based models like DINO and Co-DETR, while highly accurate, were notably slower, limiting their use in strict real-time scenarios under current hardware conditions.

5.3. Architectural Insights

Two-stage models (Cascade R-CNN):

Demonstrated good accuracy but lagged behind one-stage and some transformer-based architectures in speed;

Their multi-step nature enhances localization precision but comes at a higher computational cost.

One-stage models (RetinaNet, RTMDet, YOLOv11):

Confirmed their advantage in speed;

In particular, RTMDet and YOLOv11 stand out as fast and sufficiently accurate, making them strong candidates for commercial deployment in agricultural monitoring systems.

Transformer-based models (DETR, DINO, RT-DETR, Co-DETR, D-FINE):

Demonstrated the potential for achieving the highest accuracy (notably D-FINE and Co-DETR), though with variable inference speed;

D-FINE and RT-DETR show that transformer architectures can be optimized for real-time performance, whereas DINO and Co-DETR remain relatively slow.

5.4. Stability of Inference Speed Across Experiments

The second part of the hypothesis — the stability of inference speed — was generally confirmed.

The relative speed ranking of models remained consistent across all experiments and both inference modes.

The differences between the “reproducible” and “production” settings were minimal, indicating that model speed is primarily determined by:

Architecture design, and

Hardware configuration,

and is less sensitive to dataset domain characteristics compared to accuracy.

 doi:10.20944/preprints202510.2158.v1

This consistency supports the reliability of the benchmarking methodology and highlights the robustness of the tested implementations.

5.5. Study Limitations

Test Set Size:

Although the dataset was significantly expanded and stratified, the test set size (569 images) may not be sufficient to fully assess result stability, particularly for rare or edge-case scenarios.

Single-Class Detection:

The study focused solely on detecting one class ("cow");

Broader agricultural applications — such as multi-species detection or equipment-based monitoring — would require additional testing and adaptation.

Specific Hardware Configuration:

All benchmarks were performed on a single GPU configuration (NVIDIA RTX 4090);

Model performance may vary across different hardware platforms, necessitating further benchmarking for edge devices and server-based deployments.

6. Conclusion

This study successfully conducted a systematic benchmarking of nine open-source object detection models on a specialized cattle dataset, providing critical empirical data for their potential commercial deployment in the agro-industrial sector.

It has been demonstrated that domain adaptation significantly increases the accuracy of all tested models compared to their reported performance on widely used benchmarks such as COCO.

6.1. Key Findings

D-FINE achieved the highest overall accuracy (AP@[0.5:0.95] = 0.872) and demonstrated strong inference speed (21.61 ms/image), making it the optimal choice for most commercial applications requiring high reliability and real-time detection.

RTMDet and YOLOv11 confirmed their leadership in speed (15.81 ms/image and 19.14 ms/image, respectively), making them preferable for scenarios where minimal image processing latency is critical, though they achieved slightly lower accuracy compared to D-FINE.

Transformer-based architectures, such as D-FINE and Co-DETR, showed potential for achieving very high accuracy, while other models like DINO and Co-DETR remained speed-constrained despite acceptable precision.

Inference speed across all models proved to be stable and predictable, confirming that performance is primarily determined by model architecture and hardware, rather than by dataset domain.

6.2. Directions for Future Work

Investigate data augmentation strategies specifically tailored for agricultural imagery and assess their impact on model performance.

Expand the dataset to include a broader range of scenarios, lighting conditions, and possibly other livestock species.

Evaluate models on diverse hardware platforms, including cost-effective GPUs and edge accelerators, to determine their suitability for different commercial deployment environments.

Integrate object detection with tracking systems to enable behavioral analytics and automated herd management.

Explore model quantization and distillation techniques to further enhance inference speed without substantial loss of accuracy.

6.3. Recommendations for the Agro-Industrial Sector

 doi:10.20944/preprints202510.2158.v1

Prioritize D-FINE for precision-critical applications.

For tasks requiring both high accuracy and real-time performance — such as detailed health monitoring and behavior analysis — D-FINE offers the best balance of precision and speed.

Use RTMDet or YOLOv11 for high-speed monitoring systems.

In scenarios where processing speed is paramount (e.g., large-scale herd counting using drones, where a slight accuracy loss is acceptable), RTMDet or YOLOv11 are the most suitable choices.

Always perform domain adaptation.

Fine-tuning models on domain-specific datasets is essential, as it leads to a substantial increase in detection accuracy compared to directly applying models pre-trained on generic datasets.

Conduct pilot deployments before scaling.

Before large-scale rollout, it is critical to run pilot studies that consider specific hardware and network environments, ensuring that the desired performance is achieved under real-world operational conditions.

Acknowledgments: This research is funded by the Science Committee of the Ministry of Science and Higher Education of the Republic of Kazakhstan (Grant No. AP23489215).

References

1. Reza, M. N., Lee, K.-H., Habineza, E., Samsuzzaman, K., Choi, Y. K., Kim, G., & Chung, S.-O. (2025). RGB-based machine vision for enhanced pig disease symptoms monitoring and health management: A review. Journal of Animal Science and Technology, 67(1), 17–42. https://doi.org/10.5187/jast.2024.e111
2. Tsai, Y.-C., Hsu, J.-T., Ding, S.-T., Rustia, D. J. A., & Lin, T.-T. (2020). Assessment of dairy cow heat stress by monitoring drinking behaviour using an embedded imaging system. Biosystems Engineering, 199, 97–108. https://doi.org/10.1016/j.biosystemseng.2020.03.013
3. Barbedo, J. G. A., Koenigkan, L. V., & Santos, T. T. (2018). Detection of cattle using drones and convolutional neural networks. Sensors, 18(7), 2048. https://doi.org/10.3390/s18072048
4. Lin, T.-Y., Maire, M., Belongie, S., Bourdev, L., Girshick, R., Hays, J., Perona, P., Ramanan, D., Zitnick, C. L., & Dollár, P. (2014). Microsoft COCO: Common Objects in Context. In Computer Vision – ECCV 2014 (pp. 740–755). Springer. https://doi.org/10.1007/978-3-319-10602-1_48
5. Lamichhane, B. R., Srijuntongsiri, G., & Horanont, T. (2025). CNN-based 2D object detection techniques: A review. Frontiers in Computer Science, 7. https://doi.org/10.3389/fcomp.2025.1437664
6. Mohammed, S. Y. (2025). Architecture review: Two-stage and one-stage object detection. Franklin Open, 12, 100322. https://doi.org/10.1016/j.fraope.2025.100322
7. Cai, Z., & Vasconcelos, N. (2021). Cascade R-CNN: High quality object detection and instance segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence, 43, 1483–1498. https://doi.org/10.1109/TPAMI.2019.2956516
8. OpenMMLab. (n.d.). Cascade R-CNN [Software repository]. GitHub. Retrieved from https://github.com/open-mmlab/mmdetection/tree/main/configs/cascade_rcnn
9. Zaidi, S. S. A., Ansari, M. S., Aslam, A., Kanwal, N., Asghar, M., & Lee, B. (2022). A survey of modern deep learning based object detection models. Digital Signal Processing, 126, 103514. https://doi.org/10.1016/j.dsp.2022.103514
10. Lin, T.-Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2020). Focal loss for dense object detection. IEEE Transactions on Pattern Analysis and Machine Intelligence, 42(2), 318–327. https://doi.org/10.1109/TPAMI.2018.2858826
11. Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017). Feature pyramid networks for object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 936–944). https://doi.org/10.1109/CVPR.2017.106
12. Lyu, C., Zhang, W., Huang, H., Zhou, Y., Wang, Y., Liu, Y., Zhang, S., & Chen, K. (2022). RTMDet: An empirical study of designing real-time object detectors. arXiv preprint arXiv:2212.07784. https://doi.org/10.48550/arXiv.2212.07784

 doi:10.20944/preprints202510.2158.v1

13. OpenMMLab. (n.d.). RTMDet [Software repository]. GitHub. Retrieved from https://github.com/open-mmlab/mmdetection/tree/main/configs/rtmdet
14. He, L., Zhou, Y., Liu, L., et al. (2025). Research on object detection and recognition in remote sensing images based on YOLOv11. Scientific Reports, 15, 14032. https://doi.org/10.1038/s41598-025-96314-x
15. Khanam, R., & Hussain, M. (2024). YOLOv11: An overview of the key architectural enhancements. arXiv preprint arXiv:2410.17725. https://doi.org/10.48550/arXiv.2410.17725
16. Shehzadi, T., Hashmi, K. A., Stricker, D., & Afzal, M. Z. (2023). Object detection with transformers: A review. arXiv preprint arXiv:2306.04670. https://doi.org/10.48550/arXiv.2306.04670
17. A review of detection transformer: From basic architecture to advanced developments and visual perception applications. (2025). Sensors, 25(13), 3952. https://doi.org/10.3390/s25133952
18. Peng, Y., Li, H., Wu, P., Zhang, Y., Sun, X., & Wu, F. (2024). D-FINE: Redefine regression task in DETRs as fine-grained distribution refinement. arXiv preprint arXiv:2410.13842. https://doi.org/10.48550/arXiv.2410.13842
19. Zhang, H., Li, F., Liu, S., Zhang, L., Su, H., Zhu, J., Ni, L. M., & Shum, H.-Y. (2022). DINO: DETR with improved denoising anchor boxes for end-to-end object detection. arXiv preprint arXiv:2203.03605. https://doi.org/10.48550/arXiv.2203.03605
20. Zhao, Y., Lv, W., Xu, S., Wei, J., Wang, G., Dang, Q., Liu, Y., & Chen, J. (2024). DETRs beat YOLOs on real-time object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 16965–16974). https://doi.org/10.1109/CVPR52733.2024.01605
21. Zong, Z., Song, G., & Liu, Y. (2023). DETRs with collaborative hybrid assignments training. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (pp. 6748–6758). https://doi.org/10.1109/ICCV51070.2023.00621
22. OpenMMLab. (n.d.). MMDetection: Cascade R-CNN [Software repository]. GitHub. Retrieved from https://github.com/open-mmlab/mmdetection/tree/main/configs/cascade_rcnn
23. OpenMMLab. (n.d.). MMDetection: RetinaNet [Software repository]. GitHub. Retrieved from https://github.com/open-mmlab/mmdetection/tree/main/configs/retinanet
24. OpenMMLab. (n.d.). MMDetection: RTMDet [Software repository]. GitHub. Retrieved from https://github.com/open-mmlab/mmdetection/tree/main/configs/rtmdet
25. Peterande. (n.d.). D-FINE [Software repository]. GitHub. Retrieved from https://github.com/Peterande/D-FINE
26. OpenMMLab. (n.d.). MMDetection: DETR [Software repository]. GitHub. Retrieved from https://github.com/open-mmlab/mmdetection/tree/main/configs/detr
27. OpenMMLab. (n.d.). MMDetection: DINO [Software repository]. GitHub. Retrieved from https://github.com/open-mmlab/mmdetection/tree/main/configs/dino
28. flytocc. (n.d.). MMDetection (branch rtdetr): RT-DETR [Software repository]. GitHub. Retrieved from https://github.com/flytocc/mmdetection/tree/rtdetr/configs/rtdetr
29. OpenMMLab. (n.d.). MMDetection Projects: CO-DETR [Software repository]. GitHub. Retrieved from https://github.com/open-mmlab/mmdetection/tree/main/projects/CO-DETR
30. Ultralytics. (n.d.). YOLOv11 models [Software documentation]. Ultralytics Docs. Retrieved from https://docs.ultralytics.com/models/yolo11/
31. Zhigalov, A. A., Ivashchuk, O. A., Biryukova, T. K., & Fedorov, V. I. (2022). Neural network detection methods for agricultural animals in dense dynamic groups on images. Artificial Intelligence and Decision Making, (4), 95–106.