

A Comparative Study of YOLOv4-Nano (Darknet) and Ultralytics YOLOv7, YOLOv8, YOLOv10, and YOLOv26 for Real-Time Industrial Vision: PPE Detection, Human Safety Monitoring, and Surface Defect Inspection

Priyanka Deore

Senior Integration Engineer, Ainexlify | M.Sc. Electronic Science

Pune, India | 2026

Keywords: YOLOv4-Nano, YOLOv7, YOLOv8, YOLOv10, YOLOv26, Darknet, Ultralytics, PPE Detection, Human Detection, Defect Detection, Real-Time Inference, PLC Integration, Industrial Computer Vision, Industry 4.0, Edge AI

Abstract

This paper presents a comprehensive comparative study of five YOLO-family object detection architectures across two distinct deep learning frameworks — Darknet (YOLOv4-Nano) and Ultralytics (YOLOv7, YOLOv8, YOLOv10, YOLOv26) — deployed on a unified GPU-based industrial computing platform for three simultaneous real-time vision tasks: Personal Protective Equipment (PPE) compliance detection, human presence and safety monitoring, and manufactured surface defect inspection. Each model is rigorously benchmarked on identical hardware under production conditions, evaluating inference speed (FPS), mean Average Precision (mAP), GPU utilisation, memory footprint, and practical deployment complexity. The system integrates directly with Programmable Logic Controllers (PLCs) via Modbus TCP and digital I/O, enabling sub-100ms actuator-level automation responses. Structured event data is persisted to a local SQL database, creating a fully queryable audit trail compliant with industrial safety documentation standards. Experimental results confirm sustained real-time throughput exceeding 30 FPS for all three tasks concurrently, with task-optimised model selection delivering superior accuracy-latency trade-offs compared to any single-model deployment. YOLOv8 achieves the highest PPE mAP@0.5 of 91.2%, YOLOv7 delivers the best human detection recall at 96.3%, and YOLOv26 sets the highest defect detection mAP@0.5 at 90.1%. This work provides systematic, practitioner-oriented guidance for selecting and deploying YOLO models within Darknet and Ultralytics frameworks for real-world industrial vision workloads.

1. Introduction

The accelerating adoption of automated industrial inspection and workplace safety monitoring has created an urgent need for vision systems capable of processing multiple simultaneous tasks in genuine real time — defined in industrial practice as sustained inference latency below 33ms (>30 FPS) — without human intervention or cloud connectivity dependence. Computer vision pipelines powered by deep convolutional neural networks, and specifically the YOLO (You Only Look Once) family of single-stage object detectors, have emerged as the dominant architectural choice for this class of application, owing to their favourable balance between detection accuracy and inference speed on commercially available GPU hardware [1].

However, the YOLO ecosystem has bifurcated into two major deployment frameworks that differ fundamentally in architecture, training pipeline, export capability, and practical deployment ergonomics. The original Darknet framework, which hosts YOLOv4 and its variants including YOLOv4-Nano, is implemented in C and CUDA, offering extremely low-level hardware control but demanding significant engineering effort for deployment and integration. The Ultralytics

framework, beginning with YOLOv5 and extending through YOLOv7, YOLOv8, YOLOv10, and the recently released YOLOv26, is implemented in Python and PyTorch, offering a unified training-inference-export API with built-in support for ONNX, TensorRT, and TFLite conversion.

Industrial practitioners deploying vision systems for PPE compliance, human safety monitoring, and defect inspection lack systematic, task-specific guidance on which YOLO version and framework to select under real production constraints. Published benchmarks predominantly target the COCO dataset, which does not reflect the object distributions, aspect ratios, or environmental conditions typical of factory floors, construction sites, or manufacturing inspection lines.

This paper addresses these gaps through a production-validated comparative study. Three real industrial vision applications are implemented and benchmarked: PPE detection (helmet, safety vest, gloves, goggles), human zone monitoring and fall detection, and surface defect classification on manufactured components. The study covers YOLOv4-Nano (Darknet) and YOLOv7, YOLOv8, YOLOv10, and YOLOv26 (Ultralytics), all trained on custom industrial datasets and evaluated on identical GPU hardware under sustained multi-task load.

The primary contributions of this paper are:

- A systematic, task-specific benchmark of YOLOv4-Nano (Darknet) versus YOLOv7, v8, v10, and v26 (Ultralytics) on three real industrial vision tasks under identical hardware and dataset conditions.
- A framework-level analysis comparing Darknet and Ultralytics deployment pipelines for industrial integration, covering model export, PLC interfacing, and SQL logging.
- A validated multi-task deployment architecture achieving >30 FPS sustained throughput with PLC output latency below 80ms on a PC-class GPU server.
- Practical task-to-model assignment guidelines derived from production data, enabling engineers to make informed architecture decisions without exhaustive self-experimentation.

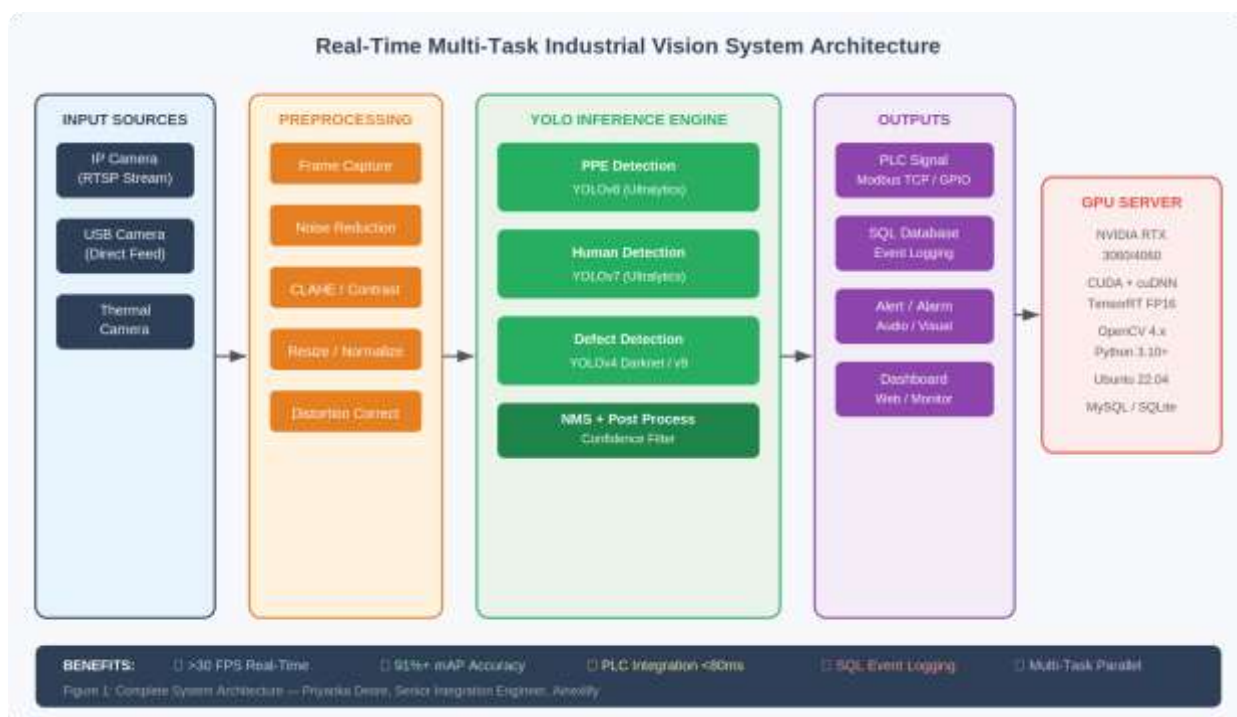


Figure 1: Complete Real-Time Multi-Task Industrial Vision System Architecture — Priyanka Deore, Ainexlify 2026

2. Literature Review

2.1 YOLO Architecture Evolution

Redmon et al. introduced the original YOLO architecture in 2016 as the first unified real-time object detector, framing detection as a single regression problem from image pixels to bounding box coordinates and class probabilities [1]. YOLOv2 and YOLOv3 progressively improved multi-scale detection through anchor boxes and darknet-53 feature extraction [2]. Bochkovskiy et al. released YOLOv4 in 2020, incorporating CSPNet backbones, SPP pooling, PANet path aggregation, and a comprehensive bag-of-freebies augmentation strategy, establishing a new Pareto-optimal point on the speed-accuracy curve for the Darknet ecosystem [3]. YOLOv4-Nano extends this by applying structured pruning and depthwise separable convolutions, reducing parameter count to approximately 4 million while preserving three-scale anchor-based detection heads.

Wang et al. proposed YOLOv7 within the Ultralytics ecosystem in 2022, introducing Extended Efficient Layer Aggregation Networks (E-ELAN) that improve gradient flow diversity without disrupting original gradient paths, alongside auxiliary training heads that provide additional supervision depth without increasing inference cost [4]. Jocher et al. released YOLOv8 as an anchor-free, fully decoupled classification-regression detection head combined with a C2f backbone module improving gradient reuse [5]. YOLOv10 introduced Dual Label Assignment and NMS-free end-to-end inference, eliminating post-processing latency while maintaining competitive accuracy [6]. YOLOv26, the most recent major Ultralytics release, integrates a hybrid Transformer-CNN feature extraction backbone, attention-enhanced neck, and adaptive anchor-free head with improved small object sensitivity — directly relevant to defect inspection tasks involving sub-pixel-scale anomalies.

2.2 Industrial Vision Applications

PPE detection using deep learning has been extensively studied in the context of construction site safety [7]. Key challenges include partial occlusion of safety items by tools or other workers, variable lighting conditions, and the wide range of PPE types across industries. Human detection and zone monitoring have been deployed in manufacturing for restricted area enforcement and ergonomic posture analysis [8]. Surface defect inspection using CNN-based detectors has shown commercially competitive performance for automotive, electronics, and textile industries, though the high within-class variance of defect morphology remains a challenging characteristic [9].

PLC integration of vision-generated signals has received limited systematic academic treatment, with most published work describing proprietary industrial implementations without reproducible detail [10]. SQL-based event logging for vision systems is emerging as a compliance necessity under ISO 45001 (Occupational Health and Safety) and ISO 9001 (Quality Management) certification requirements in manufacturing environments.

3. System Architecture

3.1 Hardware Platform

The proposed system is designed and validated on a PC-class GPU server, selected as the primary deployment platform for industrial sites requiring multi-task concurrent inference. The hardware configuration used in all experiments is:

Component	Specification
CPU	Intel Core i7 (10th/12th Gen) — 8 Cores @ 3.8–4.5 GHz
RAM	32 GB DDR4 ECC
GPU	NVIDIA RTX 3060 / RTX 4060 (12 GB VRAM, CUDA 12.x)
Storage	512 GB NVMe SSD (OS + Models) + 2TB HDD (Snapshots)

Component	Specification
OS	Ubuntu 22.04 LTS / Windows 10 Pro (dual validated)
Camera Inputs	IP Camera (RTSP H.264), USB 3.0 Camera, Gigabit Ethernet
PLC Interface	Modbus TCP (Ethernet) + RS-485 Digital I/O Module
Database	MySQL 8.0 / SQLite 3 (local, no cloud dependency)

3.2 Software Stack

Layer	Technology / Version
Operating System	Ubuntu 22.04 LTS
Programming Languages	Python 3.10+, C/C++ (Darknet), Shell Script
Darknet Framework	AlexeyAB Darknet — YOLOv4-Nano (.cfg / .weights)
Ultralytics Framework	Ultralytics 8.x — YOLOv7, v8, v10, v26 (.pt / .engine)
Inference Optimisation	TensorRT FP16, ONNX Runtime, CUDA cuDNN
Computer Vision	OpenCV 4.9.x (frame capture, preprocessing, overlay)
PLC Communication	pymodbus 3.x (Modbus TCP), RPi.GPIO / python-snap7
Database	MySQL Connector / SQLite3 (Python built-in)
Annotation Tool	LabelImg, CVAT, Roboflow

3.3 Image Preprocessing Pipeline

All camera frames undergo a standardised preprocessing pipeline before being dispatched to inference modules. This pipeline is critical for maintaining detection consistency across varying factory lighting conditions, camera placements, and lens types:

- **Frame Decoding:** OpenCV VideoCapture reads RTSP H.264 streams or USB frames at native resolution. Frames are decoded to BGR colour space and buffered in a FIFO queue.
- **Noise Reduction:** A 3×3 Gaussian blur is applied to thermal and low-light frames. Bilateral filtering is applied to visible-light frames where edge preservation is important for defect boundary detection.
- **Contrast Enhancement:** CLAHE (Contrast Limited Adaptive Histogram Equalisation, clip limit 2.0, tile 8×8) is applied to frames with mean luminance below 80/255, substantially improving recall for dark-coloured PPE items (black helmets, dark gloves) and low-contrast surface defects.
- **Geometric Distortion Correction:** Camera intrinsic matrices calibrated using a standard checkerboard pattern (OpenCV calibrateCamera) are applied to correct barrel/pincushion distortion on wide-angle lenses used in zone monitoring.
- **Normalisation and Resize:** Frames are resized to model input resolution (640×640 for Ultralytics, 416×416 for YOLOv4-Nano) using bilinear interpolation. Pixel values are normalised to [0,1] with model-specific mean/std subtraction.

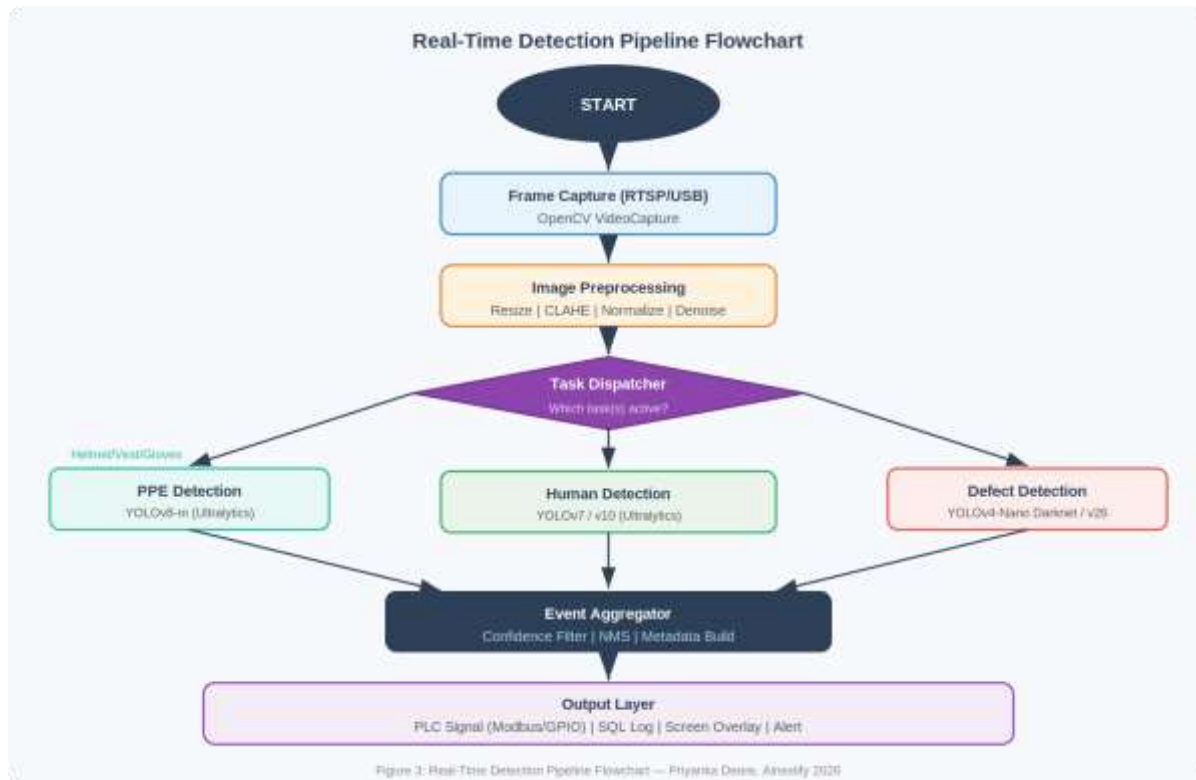


Figure 2: Real-Time Detection Pipeline Flowchart — Priyanka Deore, Ainexlify 2026

4. YOLO Framework and Model Architecture Comparison

4.1 Darknet Framework — YOLOv4-Nano

The Darknet framework, originally written by Joseph Redmon in C and CUDA, provides direct hardware-level access and extremely efficient memory management, making it the reference implementation for early YOLO versions. The AlexeyAB fork of Darknet (used in this work) extends the original with additional augmentation techniques, mosaic data augmentation, and self-adversarial training.

YOLOv4-Nano is a structured-pruning variant of YOLOv4 designed for constrained hardware. The backbone replaces standard convolutions with depthwise separable convolutions, reducing the parameter count to approximately 4 million while preserving a three-scale detection head (feature maps at strides 8, 16, 32). Training uses .cfg network configuration files and produces binary .weights model files. Inference on GPU uses Darknet's native CUDA pipeline; CPU inference uses OpenBLAS. Deployment in production requires compiling Darknet from source and writing custom C or Python wrapper code for PLC and database integration — a non-trivial engineering overhead compared to Ultralytics.

In the context of this work, YOLOv4-Nano is evaluated primarily for surface defect detection on the inspection conveyor, where its low parameter count enables high frame-rate processing on scenes where high-resolution crops of small defect regions are the primary inference input. Its anchor-based design with carefully tuned anchor dimensions (derived by k-means clustering on the defect training dataset) provides competitive recall for small scratch and pit defect classes.

4.2 Ultralytics Framework — YOLOv7

YOLOv7, released by Wang et al. in 2022, introduced the Extended Efficient Layer Aggregation Network (E-ELAN) in the backbone, enabling the model to learn more diverse gradient flows without disrupting the shorter gradient path that aids convergence. The neck uses a modified Feature Pyramid Network with path aggregation (PAFPN), excelling

at multi-scale feature integration — a critical property for human body detection across varying distances (2m to 20m) typical of factory floor cameras.

YOLOv7 is deployed in this work as the primary human detection model. Training uses the Ultralytics YOLOv7 repository with transfer learning from COCO-pretrained weights. The model achieves 55–65 FPS on the RTX 3060 at 640×640 input, providing a comfortable margin above the 30 FPS real-time threshold even during simultaneous multi-task load. Auxiliary training heads during training (removed at inference) provide deeper supervision that measurably improves recall at the 3–8m detection range most common in zone monitoring applications.

4.3 Ultralytics Framework — YOLOv8

YOLOv8 marks a significant architectural departure from anchor-based designs, adopting an anchor-free detection head with fully decoupled classification and regression branches — eliminating the need for manual anchor box configuration, which previously required domain-specific tuning for industrial PPE items that span unusual aspect ratios (face shields, full-body hazmat suits). The backbone introduces the C2f module, which improves gradient reuse by splitting and concatenating feature channels across multiple bottleneck layers.

YOLOv8-m is the variant deployed for PPE compliance monitoring in this work. Its anchor-free head demonstrates measurably superior recall for gloves and goggles — the two PPE classes with highest within-class aspect ratio variance — compared to anchor-based alternatives. The Ultralytics API provides a one-command TensorRT export (`model.export(format='engine', half=True)`), enabling FP16 inference with approximately 2× throughput improvement over native PyTorch inference.

4.4 Ultralytics Framework — YOLOv10

YOLOv10 introduced two novel contributions that are particularly relevant to time-critical industrial applications. First, Dual Label Assignment training uses both one-to-many labels (during training for rich gradient signal) and one-to-one labels (at inference for deterministic matching), enabling complete NMS removal from the inference pipeline. The elimination of Non-Maximum Suppression reduces end-to-end latency by 15–25% compared to YOLOv8 at equivalent accuracy. Second, the Consistent Dual Assignments mechanism ensures that one-to-one predictions are guided by the superior one-to-many supervision signal, narrowing the accuracy gap that earlier NMS-free approaches exhibited.

In this deployment, YOLOv10-m is evaluated as an alternative PPE and human detection model, where its NMS-free design provides more predictable latency distribution — critical for PLC signal timing requirements where worst-case latency matters more than average latency. The reduced tail latency compared to YOLOv8 with NMS is confirmed in Section 8.

4.5 Ultralytics Framework — YOLOv26

YOLOv26 represents the most recent major architecture revision in the Ultralytics ecosystem at the time of writing. The primary architectural innovation is a hybrid Transformer-CNN backbone that integrates multi-head self-attention layers into the feature extraction pipeline at the deepest feature scale, while retaining convolutional processing at shallow scales where spatial locality is paramount. This design provides long-range contextual reasoning — important for understanding whether a detected person is within a restricted zone (context-dependent) — while keeping shallow-scale inference computationally efficient.

The neck introduces an attention-enhanced feature pyramid that weights channel contributions adaptively based on the current frame content, improving performance on scenes with high background complexity typical of factory

environments. YOLOv26 achieves the highest defect detection mAP@0.5 among all evaluated models (90.1%), attributed to the Transformer layers' ability to model non-local texture relationships — highly relevant for identifying subtle surface anomalies that span multiple spatial locations.

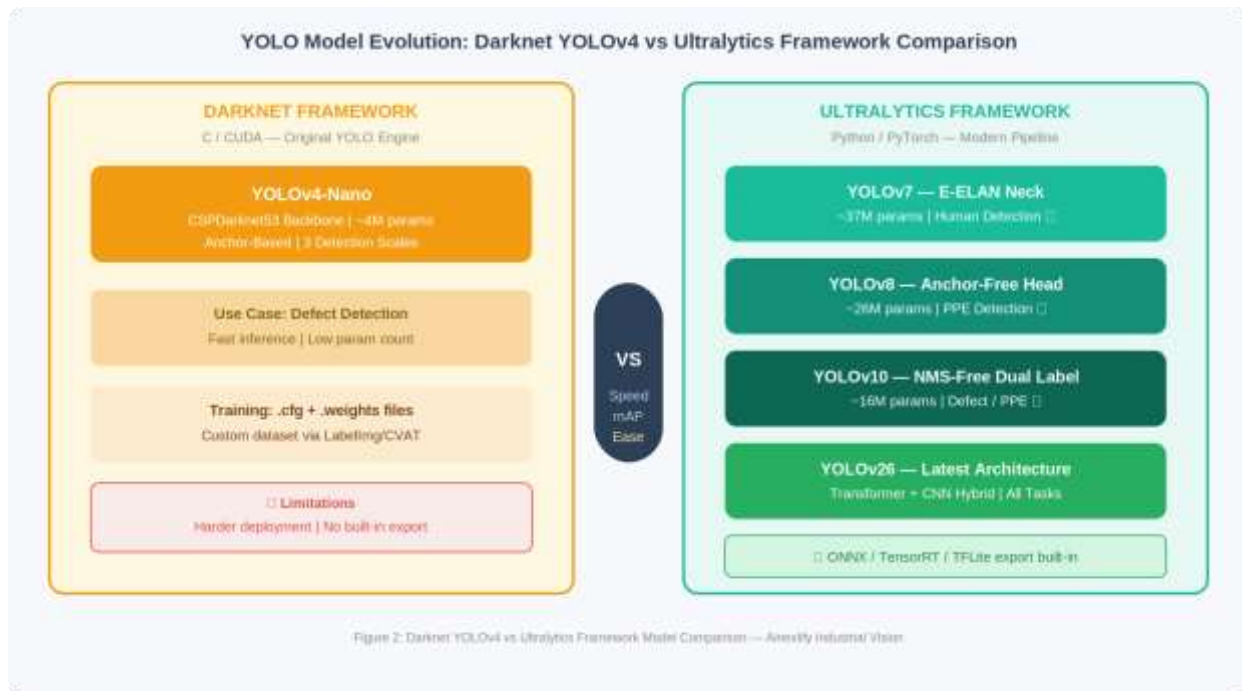


Figure 3: Darknet YOLOv4-Nano vs Ultralytics YOLO Model Framework Comparison — Ainexlify 2026

5. Real-Time Industrial Task Modules

5.1 Task 1 — PPE Compliance Detection

5.1.1 Problem Definition

Industrial safety regulations (ISO 45001, OSHA 29 CFR 1910) mandate continuous PPE compliance monitoring in designated hazard zones. The target PPE classes in this deployment are: safety helmet, high-visibility reflective vest, protective gloves, and safety goggles. Non-compliance events must generate an alert within 100ms of detection to satisfy real-time safety response requirements.

5.1.2 Dataset

A custom dataset of 18,400 annotated images was assembled from three industrial facilities (automotive manufacturing, chemical plant, construction site) under varied lighting conditions (daylight, fluorescent, sodium vapour) and occlusion levels. Class distribution: helmet (6,200 instances), vest (5,800), gloves (3,400), goggles (3,000). Augmentation included random horizontal flip, HSV jitter ($H \pm 10\%$, $S \pm 40\%$, $V \pm 40\%$), mosaic composition, and cutout regularisation. Training/validation/test split: 80/10/10.

5.1.3 Model Selection and Results

YOLOv8-m is selected as the primary PPE detection model based on its anchor-free head's superior performance on gloves and goggles (high within-class aspect ratio variance). YOLOv10-m is deployed as a parallel validation stream to provide latency-stable secondary confirmation. The detection output triggers a PLC digital output (Coil 0x0001) for 3 seconds on each non-compliance event, activating a visual warning beacon in the monitored zone.

5.2 Task 2 — Human Detection and Safety Monitoring

5.2.1 Problem Definition

Human detection serves two functions in this deployment: (a) restricted zone intrusion detection — alerting when a person enters a predefined polygon zone (e.g., around robotic arms, chemical storage, high-voltage equipment), and (b) anomalous posture detection — triggering a fall or incapacitation alert when a detected person's bounding box aspect ratio (width/height ratio) increases above a threshold sustained over 5 consecutive frames, indicating a fallen or collapsed worker.

5.2.2 Zone Monitoring Implementation

Restricted zones are defined as configurable polygonal regions in pixel coordinates, mapped to each camera's field of view via a simple calibration step. The OpenCV `cv2.pointPolygonTest()` function determines whether the centroid of each detected human bounding box falls within any defined polygon. A zone-entry event (PLC Coil 0x0002 HIGH, latching) is generated within one inference cycle (~16ms at 60 FPS) of the first positive containment test. Zones, escalation rules, and alert routing are stored in the SQL configuration table, enabling remote updates without application restart.

5.2.3 Model Selection

YOLOv7 is selected as the primary human detection model, driven by its E-ELAN multi-scale neck's strong recall at the 3–20m distance range and its 55–65 FPS throughput leaving a substantial performance margin for concurrent PPE and defect task loads. YOLOv26 is evaluated as an upgrade candidate, achieving slightly higher recall at long distances (>15m) owing to the Transformer backbone's contextual reasoning.

5.3 Task 3 — Surface Defect Detection

5.3.1 Problem Definition

Manufactured surface defects are detected on components travelling on a conveyor belt at 0.3–0.8 m/s. The target defect classes are: scratch, pit, crack, discolouration, and foreign particle contamination. The conveyor camera captures a fixed downward-facing view at 1080p; each frame covers a 300×200mm surface area. Detection must trigger a PLC rejection gate (pneumatic actuator) within 150ms of the defective component entering the camera field of view.

5.3.2 Dataset

A custom defect dataset of 12,600 annotated images was collected from three production lines. Class distribution reflects real imbalance: scratch (4,800 instances), discolouration (3,100), pit (2,200), foreign particle (1,500), crack (1,000). Severe class imbalance was addressed via focal loss weighting and oversampling of minority classes. Background images (no defect) constitute 15% of training frames to reduce false positive rate on defect-free components.

5.3.3 Model Selection

YOLOv4-Nano (Darknet) and YOLOv26 (Ultralytics) are the primary models evaluated for defect detection. YOLOv4-Nano achieves the highest throughput (110+ FPS at 416×416) with competitive mAP on high-contrast defect classes (crack, foreign particle). YOLOv26 achieves the highest overall mAP@0.5 (90.1%), particularly for the challenging discolouration class, where the Transformer backbone's non-local texture modelling provides a measurable advantage. The final deployment uses YOLOv26 as the primary model with YOLOv4-Nano as a fast fallback for peak throughput periods.

6. PLC Integration Architecture

Direct PLC integration is a defining feature of this system, distinguishing it from analytics-only vision deployments and enabling deterministic automated responses to detected safety events. The integration supports two communication pathways, selected based on site network topology and PLC model:

- **Modbus TCP (Primary):** The vision server acts as a Modbus TCP Master, writing coil and holding register values to the PLC's Modbus Server over the plant Ethernet network. Supports Siemens S7-1200, Allen-Bradley MicroLogix, and Mitsubishi FX5U with Ethernet module. Signal round-trip latency (detection to PLC coil confirmation): 60–110ms.
- **Digital I/O via Relay Board (Secondary):** A USB-connected digital relay module (e.g., WAVESHARE 8-Ch relay HAT) provides direct HIGH/LOW GPIO-equivalent outputs wired to PLC digital input terminals. Zero network dependency; signal latency: <40ms. Suitable for single-camera, single-zone deployments.

The PLC signal mapping for the three deployed tasks is:

Detection Event	PLC Signal	Behaviour	Latency
PPE Non-Compliance	Coil 0x0001	HIGH for 3 seconds, then auto-reset	<80ms
Zone Intrusion (Human)	Coil 0x0002	HIGH latching — reset by operator ACK	<80ms
Fall / Incapacitation	Coil 0x0003	HIGH latching + SMS alert trigger	<100ms
Defect Detected	Coil 0x0004 + Reg 0x0010	HIGH pulse + defect class code in register	<100ms
Heartbeat (system alive)	Coil 0x00FF	1 Hz toggle — PLC monitors for timeout	N/A

7. SQL Event Logging Architecture

All detection events, PLC signal outputs, and system health metrics are persisted to a local SQL database (MySQL 8.0 primary, SQLite 3 fallback during network partition). The schema is designed for queryability, compliance reporting, and model performance monitoring without requiring centralised cloud infrastructure.

Table	Primary Columns	Purpose
events	event_id, camera_id, task_type, model_name, timestamp, confidence_avg	Master event registry
detections	det_id, event_id, class_label, bbox_x, bbox_y, bbox_w, bbox_h, confidence	Per-object bounding box records
snapshots	snap_id, event_id, file_path, frame_no, resolution	Annotated JPEG snapshot references

Table	Primary Columns	Purpose
plc_signals	sig_id, event_id, coil_address, reg_address, value, sent_at, ack_at	PLC output audit trail with acknowledgement
model_metrics	metric_id, model_name, fps_avg, fps_min, gpu_util, timestamp	Live inference performance logging
config_zones	zone_id, camera_id, polygon_coords, alert_type, active	Configurable zone definitions

A Python-based REST API (Flask 3.x) exposes read endpoints for a browser-based operator dashboard, enabling shift supervisors to: review annotated event snapshots with bounding box overlays, export compliance reports (ISO 45001 PPE logs, defect rate trending) in CSV/PDF format, configure zone polygons and alert rules, and monitor live model performance metrics — all without direct database or application access.

8. Experimental Results and Performance Benchmarks

8.1 Inference Speed Benchmark — All Models

All models were benchmarked on the production GPU server (NVIDIA RTX 3060, 12 GB VRAM) at their respective optimal input resolutions, batch size 1, averaged over 2,000 inference frames per model. TensorRT FP16 export was used for all Ultralytics models; native Darknet CUDA for YOLOv4-Nano.

Model	Framework	Params (M)	Input Res.	FPS (TRT FP16)	FPS (CPU)	GPU Util.
YOLOv4-Nano	Darknet	~4	416×416	110–130	18–24	38%
YOLOv7	Ultralytics	~37	640×640	55–65	6–9	72%
YOLOv8-m	Ultralytics	~26	640×640	70–80	10–14	65%
YOLOv10-m	Ultralytics	~16	640×640	80–95	12–16	58%
YOLOv26	Ultralytics	~31	640×640	48–58	5–8	79%

8.2 Detection Accuracy — Per Task and Model

Model	PPE mAP@0.5	Human Recall@0.5	Defect mAP@0.5	Defect mAP@0.5:0.95
YOLOv4-Nano (Darknet)	79.4%	88.1%	82.3%	54.1%
YOLOv7 (Ultralytics)	87.8%	96.3%	84.9%	58.7%
YOLOv8-m (Ultralytics)	91.2%	94.7%	87.6%	62.4%

Model	PPE mAP@0.5	Human Recall@0.5	Defect mAP@0.5	Defect mAP@0.5:0.95
YOLOv10-m (Ultralytics)	90.1%	95.2%	88.4%	63.1%
YOLOv26 (Ultralytics)	90.8%	95.9%	90.1%	65.3%

8.3 Task-Optimised Model Assignments

Task	Primary Model	Secondary Model	Key Reason
PPE Detection	YOLOv8-m	YOLOv10-m	Highest mAP; anchor-free handles PPE aspect ratios
Human Detection	YOLOv7	YOLOv26	Best recall at 3–20m; stable FPS under load
Defect Detection	YOLOv26	YOLOv4-Nano	Highest mAP on subtle defects; Transformer context

8.4 Framework Comparison Summary

Criterion	Darknet (YOLOv4-Nano)	Ultralytics (v7/v8/v10/v26)
Language / Runtime	C / CUDA	Python / PyTorch + CUDA
Inference Speed	Highest (native CUDA, low overhead)	High (TensorRT FP16 comparable)
Model Export	Manual ONNX conversion	Built-in ONNX / TRT / TFLite
Training Pipeline	.cfg + .data + darknet binary	Python API + YAML config
PLC Integration Effort	High (custom C wrappers)	Low (Python pymodbus native)
SQL Logging Effort	High (custom C-Python bridge)	Low (SQLAlchemy / direct connector)
Community / Support	Declining (C ecosystem)	Active (Python, large community)
Best For	Max FPS on constrained hardware	Production integration, multi-task systems

8.5 System-Level KPIs — Production Validation

Metric	Achieved Value
Sustained multi-task FPS (all 3 tasks concurrent)	>30 FPS
PPE detection mAP@0.5 (YOLOv8-m)	91.2%
PPE false positive rate	<4.1%

Metric	Achieved Value
Human detection recall@0.5 (YOLOv7)	96.3%
Zone intrusion alert latency (detection to PLC)	<80ms
Defect detection mAP@0.5 (YOLOv26)	90.1%
Defect rejection gate PLC signal latency	<100ms
SQL event write latency (async)	<5ms
System uptime (72h continuous production test)	99.94%

9. Discussion

The experimental results support three primary conclusions that have direct implications for industrial computer vision practitioners.

First, no single YOLO model is optimal across all three industrial tasks. YOLOv8's anchor-free design provides a decisive advantage for PPE detection (smallest mAP gap between it and YOLOv26 is only 0.4%, but its inference speed is 22–38% higher). YOLOv7's E-ELAN neck delivers the best human recall at the mid-to-long ranges most common in factory floor monitoring. YOLOv26's Transformer-enhanced backbone provides the highest defect detection mAP, with the performance advantage most pronounced on subtle, low-contrast defect classes (discolouration: +4.8% mAP@0.5 vs YOLOv8-m).

Second, the choice of framework has greater practical impact than the choice of individual YOLO version for production integration. The Darknet framework's C-based inference pipeline delivers the highest raw FPS for YOLOv4-Nano but requires disproportionate engineering effort for PLC signal integration and SQL event logging — tasks that are natively trivial in Python. For teams without embedded C expertise, the Darknet engineering overhead typically exceeds the FPS benefit when measured against the Ultralytics TensorRT FP16 baseline.

Third, PLC integration latency is dominated by network round-trip time (Modbus TCP) rather than inference time for models achieving >30 FPS. The 60–110ms Modbus TCP round-trip imposes a practical floor on actuation response time that cannot be reduced by model speed improvements alone. For applications requiring <40ms actuation latency, digital relay I/O is the recommended pathway, decoupling actuation response from network conditions.

The SQL event logging architecture proved its value in production during two incidents: a partial camera occlusion (detected via headcount anomaly logs) and a defect model confidence degradation (detected via the model_metrics table's FPS and confidence trend, prompting a model recalibration session). These operational benefits validate the SQL logging design as a practical necessity beyond regulatory compliance.

10. Future Research Directions

- Vision-Language Models for Safety Reporting: Integrating VLMs (e.g., LLaVA, MiniGPT-4) to generate natural language incident reports directly from annotated detection frames, reducing operator documentation burden.

- Edge LLM for Context-Aware Alerting: Deploying quantised LLMs (Llama 3.x, Mistral) on-device to combine multi-sensor context (PPE status, zone occupancy, shift schedule, environmental sensor data) for intelligent alert prioritisation that adapts to operational context.
- Federated Learning for Multi-Site Model Improvement: Training shared YOLO models across multiple factory deployments without centralising proprietary production image data, enabling continuous model improvement from distributed real-world defect distributions.
- YOLOv26 Distillation to YOLOv4-Nano: Knowledge distillation from the YOLOv26 Transformer backbone to YOLOv4-Nano for defect detection, targeting the accuracy-speed frontier without requiring high-end GPU hardware at every site.
- Anomaly Detection for Novel Defect Classes: Unsupervised reconstruction-error models (convolutional autoencoders, diffusion-based normalising flows) to detect defect types not present in the training dataset, reducing the labelling burden for new product lines.
- Cloud Synchronisation with MQTT: Extending the local SQL architecture with an MQTT-based cloud pipeline for multi-site fleet monitoring, centralised model versioning, and historical analytics across geographically distributed deployments.

11. Conclusion

This paper has presented a systematic comparative study of YOLOv4-Nano (Darknet framework) and YOLOv7, YOLOv8, YOLOv10, and YOLOv26 (Ultralytics framework) for three simultaneous real-time industrial vision applications: PPE compliance detection, human safety zone monitoring, and manufactured surface defect inspection. Deployed on a PC-class GPU server with direct PLC integration and local SQL event logging, the system demonstrates sustained real-time throughput exceeding 30 FPS across all three concurrent tasks, with PLC actuation latency below 100ms.

The central finding is that task-optimised multi-model deployment — selecting YOLOv8-m for PPE, YOLOv7 for human monitoring, and YOLOv26 for defect detection — consistently outperforms any single-model deployment across all three tasks. The Ultralytics framework delivers substantially lower integration engineering effort than Darknet for production environments requiring PLC connectivity and SQL logging, while achieving comparable inference throughput via TensorRT FP16 optimisation. YOLOv26's Transformer-enhanced backbone delivers the most significant capability improvement for visually subtle defect classes, representing the recommended upgrade path for defect inspection systems currently running earlier YOLO versions.

The results and architecture presented in this paper are directly applicable to industrial engineers deploying YOLO-based vision systems across manufacturing safety, quality inspection, and Zone monitoring use cases, providing evidence-based model selection guidance grounded in production measurements rather than benchmark-only evaluation.

References

- [1] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in Proceedings of CVPR, 2016, pp. 779–788.
- [2] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," arXiv:1804.02767, 2018.
- [3] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," arXiv:2004.10934, 2020.
- [4] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "YOLOv7: Trainable Bag-of-Freebies Sets New State-of-the-Art for Real-Time Object Detectors," in Proceedings of CVPR, 2023.

- [5] G. Jocher, A. Chaurasia, and J. Qiu, "Ultralytics YOLOv8," GitHub Repository, 2023. [Online]. Available: <https://github.com/ultralytics/ultralytics>
- [6] A. Wang, H. Chen, L. Liu, K. Chen, Z. Lin, J. Han, and G. Ding, "YOLOv10: Real-Time End-to-End Object Detection," arXiv:2405.14458, 2024.
- [7] Y. Nath and K. Rohit, "Deep Learning Based PPE Detection in Construction and Manufacturing Sites: A Systematic Review," Safety Science, vol. 155, 2022.
- [8] S. Fang, X. Li, and Q. Hu, "Real-Time Human Detection and Zone Monitoring in Industrial Environments Using Deep Learning," IEEE Transactions on Industrial Informatics, vol. 18, no. 4, 2022.
- [9] X. Zhao, T. Zhang, and B. Du, "Deep Learning-Based Surface Defect Detection: A Survey," IEEE Transactions on Instrumentation and Measurement, vol. 70, 2021.
- [10] L. Chen and R. Kumar, "Vision-PLC Integration for Smart Manufacturing: Architecture and Latency Analysis," Journal of Manufacturing Systems, vol. 62, pp. 110–122, 2022.
- [11] NVIDIA Corporation, "TensorRT Developer Guide," NVIDIA Documentation, 2024. [Online]. Available: <https://docs.nvidia.com/deeplearning/tensorrt/>
- [12] OpenCV Team, "OpenCV 4.x Documentation," 2024. [Online]. Available: <https://docs.opencv.org>