TED UNIVERSITY

CMPE492/ SENG492

<safeSCOPE>

Test Plan Report

Team Members:

Arda BARAN 19172802022 Computer Engineering

Yakup Mert AKAN 15574008550 Software Engineering

Baran KUZUCANLI 36529547832 Computer Engineering

Sena ÖZTÜRK 19750960502 Computer Engineering

Supervisor: Ali BERKOL

Jury Members: Tolga Kurtuluş ÇAPIN, Emin KUĞU

Contents

1. Introduction

Ensuring the safety of personal on construction sites is a critical operational priority. The presence and proper use of Personal Protective Equipment (PPE) such as helmets, gloves, safety vests, and face masks significantly reduces the risk of workplace injuries. Equally important is the monitoring of heavy machinery and construction equipment—like excavators, bulldozers, and dump trucks—to prevent operational hazards and maintain site awareness. In this context, automated computer vision systems play a vital role in scaling safety oversight with real-time intelligence.

This Test Plan outlines a structured validation framework for the PPE and Construction Equipment Detection System developed using the YOLOv9e deep learning model. The system was trained using a composite dataset formed by merging two distinct sources: the SH17 dataset (specialized in PPE detection) and the Construction Equipment dataset (focused on construction machinery). With a total of 34 categories—17 for PPE and 17 for equipment—the model is expected to deliver high detection accuracy in diverse environmental conditions across construction sites.

The goal of this plan is to ensure that the system adheres to stringent safety and reliability requirements by validating its performance through a series of quantitative and qualitative evaluations. The tests will cover aspects such as image-level detection performance, cross-domain generalization, inference latency, and robustness against variations in lighting, weather, and worker posture.

In real-world deployments, such a system can be integrated into fixed surveillance setups or drone-based monitoring workflows. Early alerts based on safety violations (e.g., missing helmet or unauthorized presence near machinery) can significantly enhance site compliance. Therefore, the accuracy, consistency, and real-time operability of the detection model must be thoroughly evaluated before deployment.

This document describes the strategy, scope, tools, roles, and risk factors associated with verifying the model's behavior. The test methodology spans unit-level verification, full-pipeline integration, system-level simulation, and on-site validation through beta testing. Through this multi-layered validation plan, the goal is to ensure a field-ready, safety-critical AI system that can be trusted in live construction site environments.

2. Scope

The scope of this testing plan encompasses a comprehensive verification of the PPE and Construction Equipment Detection System's ability to correctly identify and classify objects across 34 categories—17 pertaining to personal protective equipment and 17 related to construction site machinery. This includes ensuring accurate detection of safety-critical elements such as helmets, safety vests, gloves, and face masks, as well as large machinery like bulldozers, dump trucks, forklifts, and cranes. Each class plays a vital role in enforcing safety compliance and operational awareness on construction sites.

The system must perform robustly under a variety of environmental and contextual conditions. To this end, the test cases will include images and videos captured in different lighting conditions (day/night), weather (rain/sun), camera perspectives (top-down, side, distant), and levels of crowd density (isolated workers vs. dense work environments). The intent is to validate the model’s generalization capabilities so that it can be deployed in real-world scenarios with minimal risk of failure.

In addition to detection accuracy, the test scope also includes evaluating the system's inference performance, including processing speed, resource usage, and memory efficiency. It is essential that the system can deliver real-time predictions—particularly for applications involving video surveillance, drone footage analysis, or live stream monitoring. To this end, metrics such as frames-per-second (FPS), GPU memory utilization, and batch throughput will be assessed.

Edge case handling also falls within the scope of testing. These include verifying how the model responds to:

- Overlapping or occluded objects (e.g., a worker behind equipment)
- Partially visible PPE (e.g., helmet strap visible but helmet not fully seen)
- Low-resolution inputs or blurry images

- Conflicting annotations or ambiguous scenes

Moreover, testing will cover the ability to scale with large datasets, adapt to class imbalances, and recover gracefully from missing or malformed input data. This will be particularly important for scenarios where automated labeling pipelines or user-uploaded data are in use.

The plan also outlines tests for export and deployment readiness. This includes validating the TorchScript export functionality, compatibility with ONNX and TensorRT, and robustness of inference on different platforms such as Google Colab, desktop GPU, and potentially ARM-based edge devices.

Lastly, the test scope includes human-centered evaluation components, such as expert reviews of predictions and usability assessments for field inspectors and safety officers. These qualitative assessments ensure that the model outputs are not only technically accurate but also practically interpretable and actionable in high-stakes environments.

Through this extended scope, we aim to ensure that the detection system is not just theoretically sound but also practically deployable, trustworthy, and safe to use in real-time construction safety workflows.

3. Items to be Tested

- PPE detection classes:

  person, ear, ear-mufs, face, face-guard, face-mask, foot, tool, glasses, gloves, helmet, hands, head, medical-suit, shoes, safety-suit, safety-vest

- Construction equipment classes :

  Dump truck,Excavator,Motor grader,Roller,Crane manipulator,Gazelle,Forklift Standart,Bucket loader Big,Mixer,Tanker,Bulldozer,Cleaning equipment,Truck,Trailer,Forklift Giraffe,Bucket loader Standart,Autocran

- End-to-end detection pipeline

- Model performance:

train/box_loss,train/cls_loss,train/dfl_loss,metrics/precision(B),metrics/recall(B),metrics/mAP50(B),metrics/mAP50-95(B),val/box_loss,val/cls_loss,val/dfl_loss,lr/pg0,lr/pg1,lr/pg2

- Inference speed and resource usage

4. Features to be Tested

Feature ID	Feature Description
F001	Detection of PPE items in images/videos
F002	Detection of construction equipment in real-world scenarios
F003	Class-wise detection confidence output
F004	Support for batch and single-image inference
F005	Error handling for unlabeled/misformatted inputs

5. Testing Methodology

5.1 Unit Testing

- **Test ID: T001**: Validate custom pre-processing functions for input normalization.
- **Test ID: T002**: Validate output structure from YOLOv9e forward pass.

5.2 Integration Testing

- **Test ID: T010**: Test the combined pipeline with data loader, model, post-processing.
- **Test ID: T011**: Integration with video stream inference.

5.3 System Testing

- **Test ID: T020**: Run full training pipeline with 200 epochs, validate loss convergence.
- **Test ID: T021**: Evaluate metrics using results.csv, PR/ROC/F1 curves.

5.4 Performance Testing

- **Test ID: T030**: Test GPU inference speed with A100 and T4 GPUs.
- **Test ID: T031**: Measure model memory usage and I/O performance.

5.5 User Acceptance Testing

- **Test ID: T040**: Visual validation of predictions by domain experts.
- **Test ID: T041**: Verify detection in varied weather and site conditions.

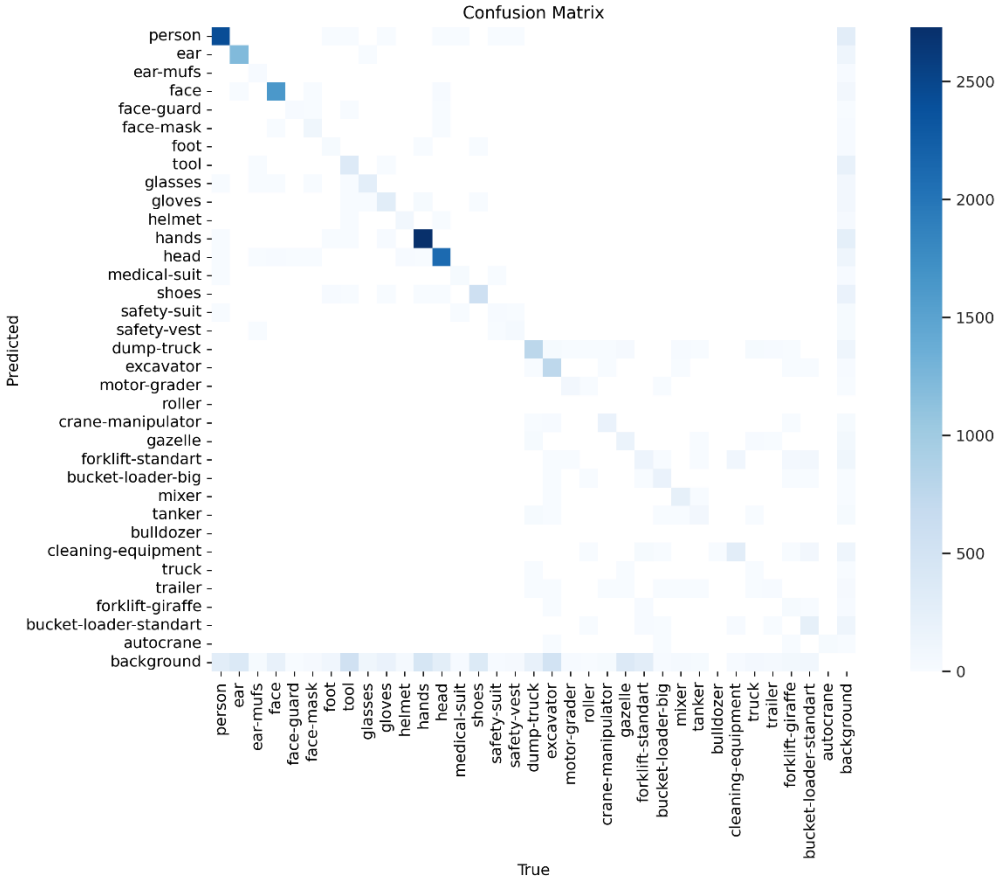
5.6 Beta Testing

- **Test ID: T050**: Run pilot deployment at test site with real-time feedback.

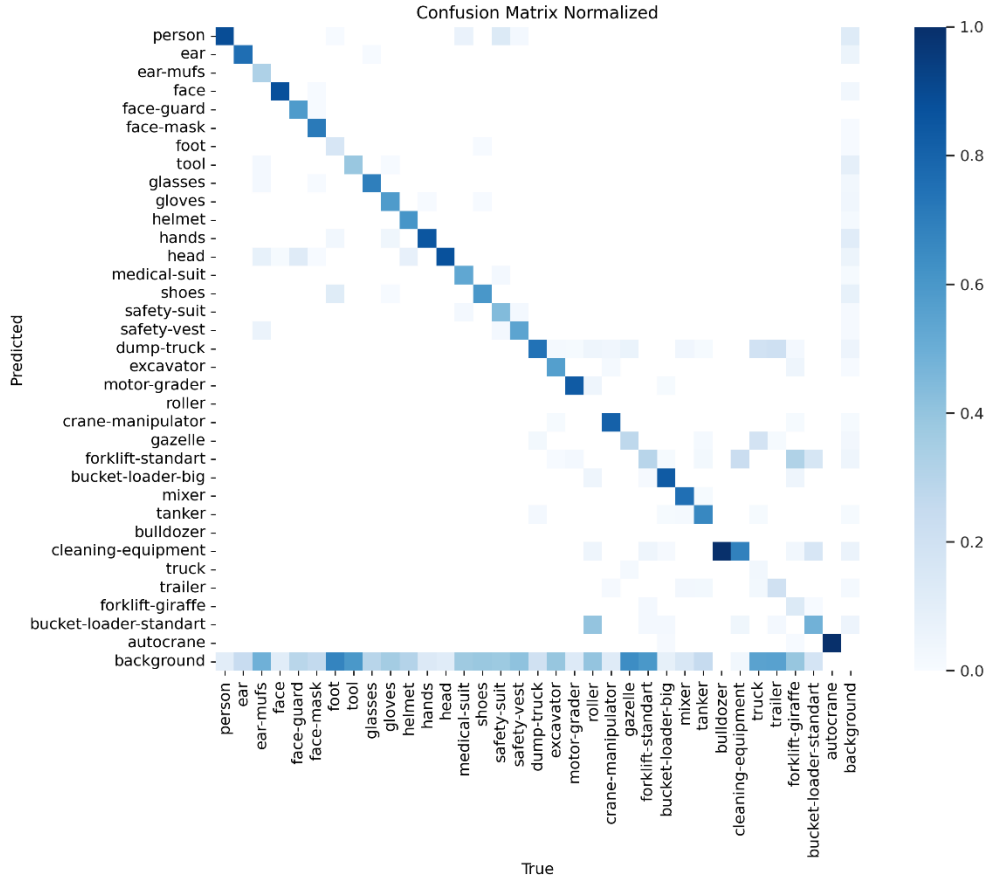
6. Test Environment

The YOLOv9e model was trained under the following configuration to optimize performance and generalization:

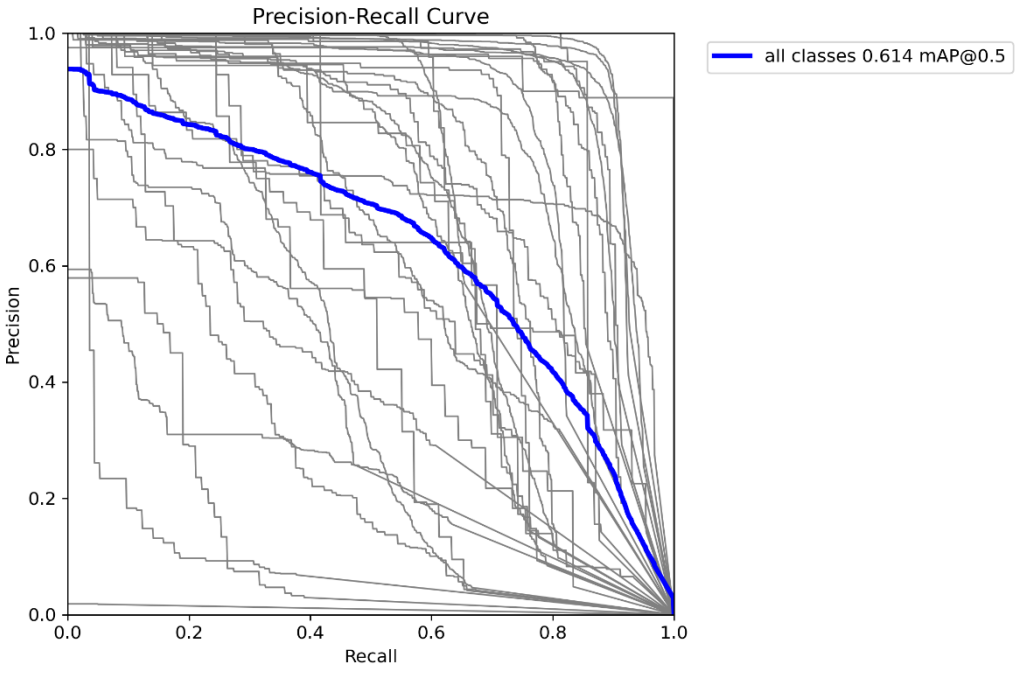
- **epochs**: 200 — Longer training duration to ensure convergence.
- **batch**: 32 — Balanced batch size to stabilize gradient calculations without memory overflow.
- **imgsz**: 640 — Standard input size for speed-accuracy trade-off.
- **device**: CUDA-enabled GPU — Leveraged NVIDIA A100 GPU for accelerated training.
- **workers**: 16 — Increased data loader workers to speed up data feeding.
- **optimizer**: SGD — Chosen over AdamW for better generalization using momentum.
- **lr0**: 0.01 — Higher initial learning rate based on empirical results.
- **momentum**: 0.937 — Typical high-momentum value for YOLO models.
- **weight_decay**: 0.0005 — Helps prevent overfitting during training.
- **cos_lr**: True — Cosine annealing learning rate schedule enabled.
- **patience**: 50 — High patience for learning rate scheduler.
- **augment**: True — Training augmentation enabled.
- **mosaic**: True — Mosaic data augmentation applied.
- **flipud**: 0.5 — 50% vertical flip probability.
- **fliplr**: 0.5 — 50% horizontal flip probability.
- **Hardware**: NVIDIA A100 GPU, 40 GB VRAM (Google Colab Pro+)
- **Software**: YOLOv9e, PyTorch, Python 3.11
- **Dataset**: 28081 images (merged SH17 + Construction Equipment Dataset)
- **Labeling**: Verified YOLO-format labels with 34 distinct class IDs
- **Evaluation Tools**:
    - Confusion Matrix (raw):



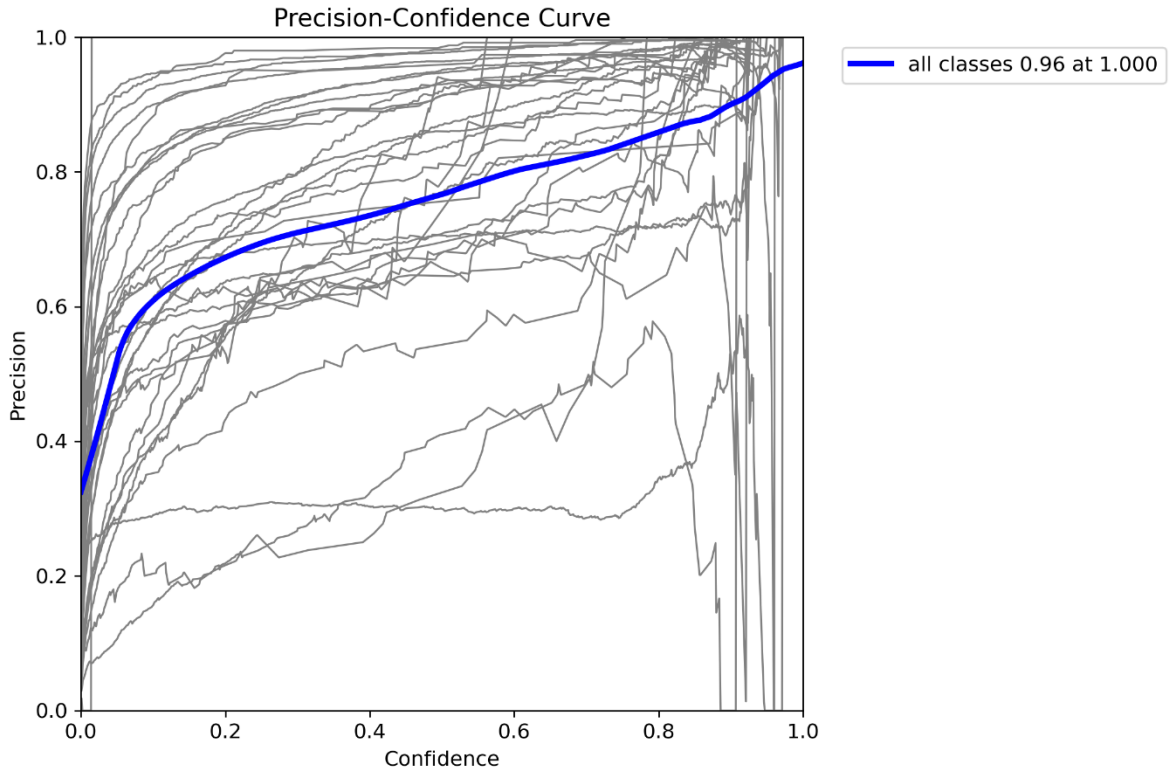
- Confusion Matrix (normalized):


Confusion Matrix Normalized
Predicted
True
person
ear
ear-mufs
face
face-guard
face-mask
foot
tool
glasses
gloves
helmet
hands
head
medical-suit
shoes
safety-suit
safety-vest
dump-truck
excavator
motor-grader
roller
crane-manipulator
gazelle
forklift-standart
bucket-loader-big
mixer
tanker
bulldozer
cleaning-equipment
truck
trailer
forklift-giraffe
bucket-loader-standart
autocrane
background
1.0
0.8
0.6
0.4
0.2
0.0

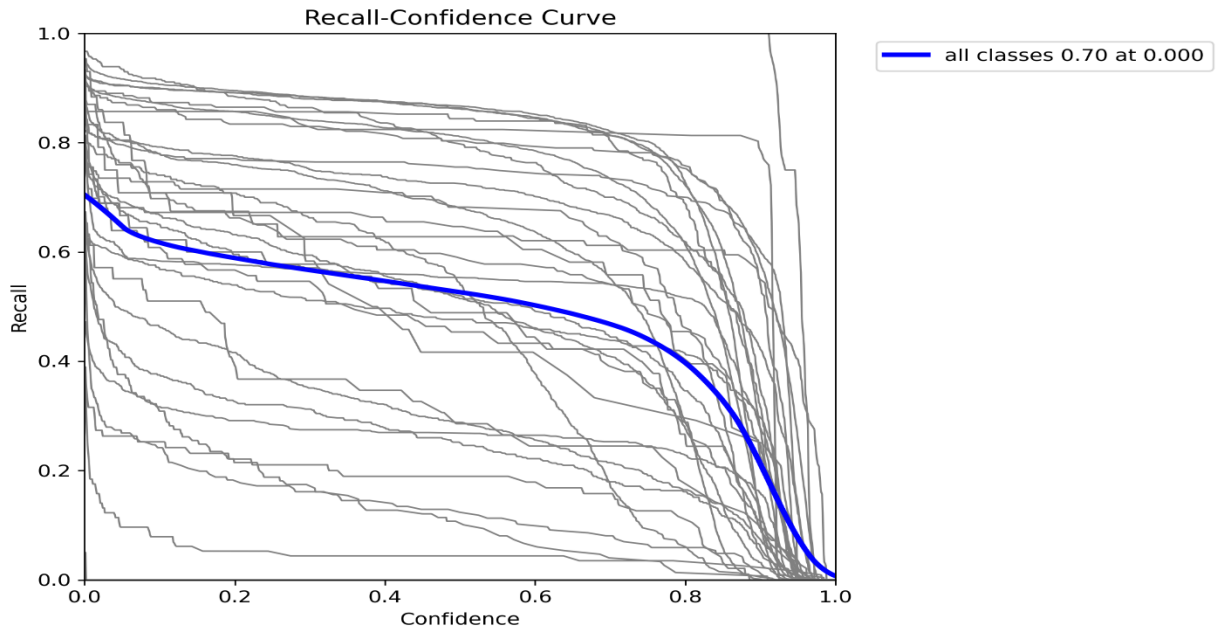
- PR Curve:


Precision-Recall Curve
all classes 0.614 mAP@0.5
Precision
Recall
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0

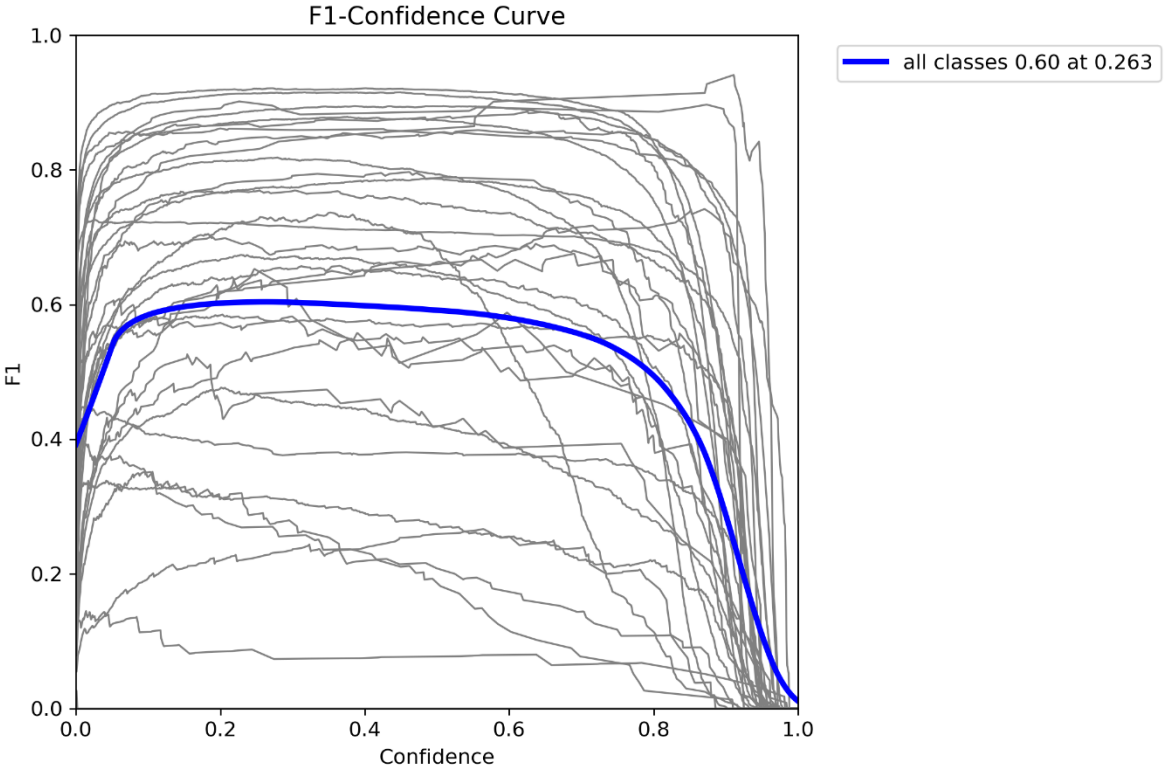
- Precision Curve:



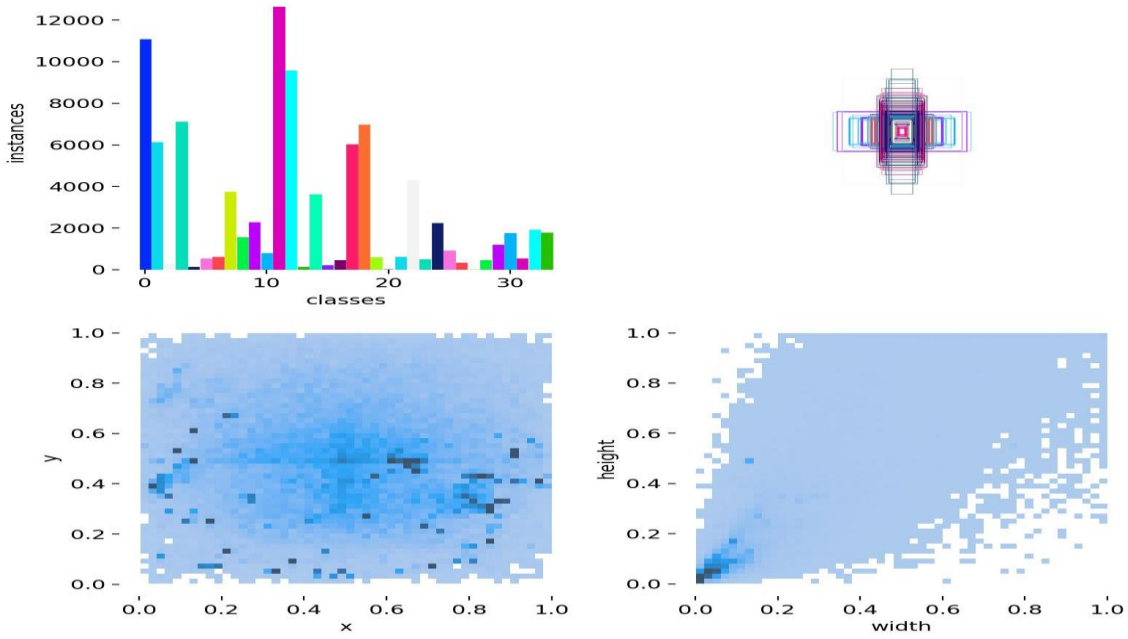
- Recall Curve:



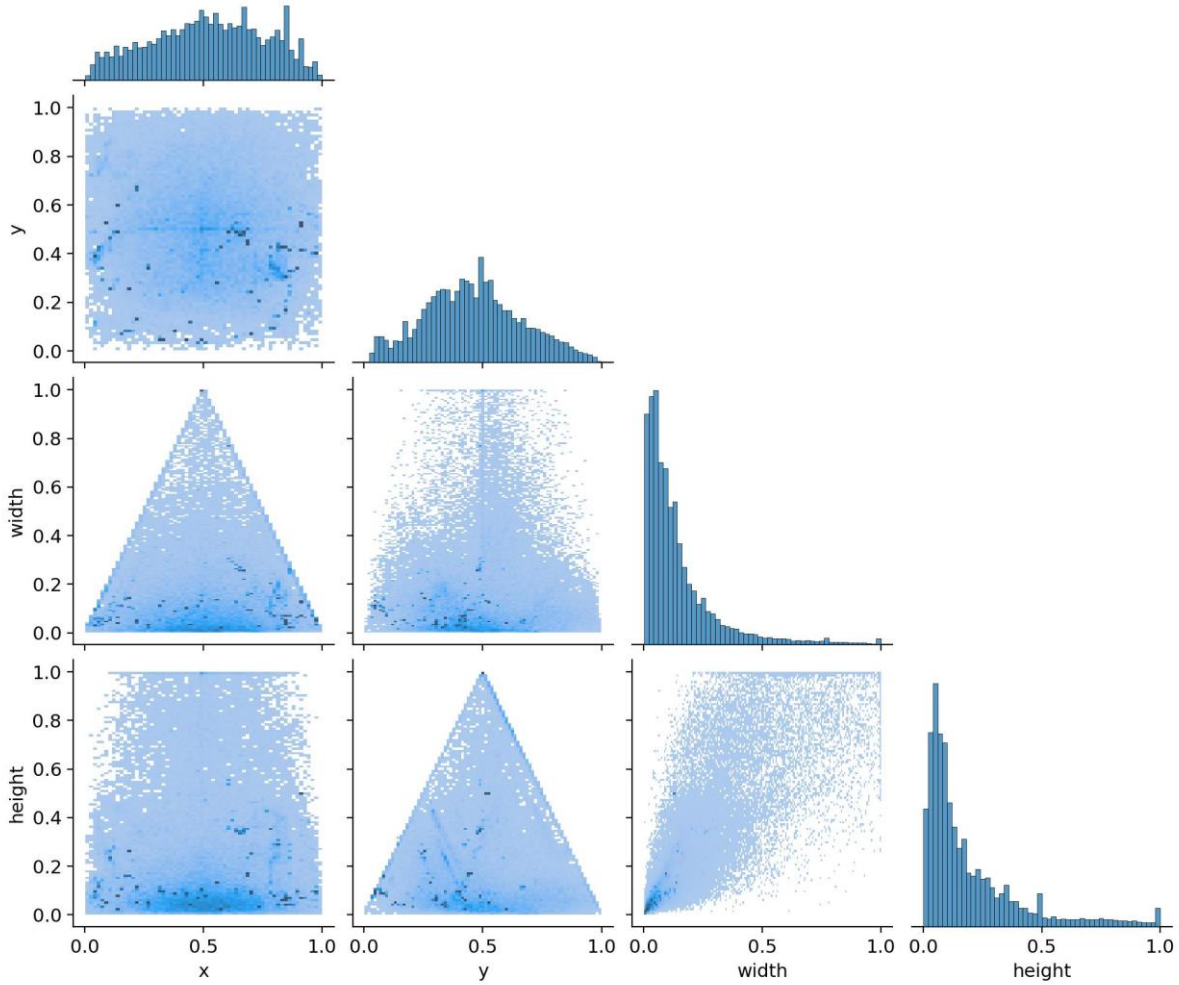
- F1 Curve:



- Labels Distribution:



- Labels Correlogram:



7. Test Schedule

Phase	Start Date	End Date
Data Prep & Review	Apr 01	Apr 04
Training & Checkpoints	Apr 05	Apr 10
Evaluation & Metrics	Apr 11	Apr 13
User Review & Testing	Apr 14	Apr 16
Final Report	Apr 16	Apr 17

8. Control Procedures

To ensure consistent, traceable, and repeatable testing of the detection system, robust control procedures are implemented throughout the training, evaluation, and deployment lifecycle.

- **Training Monitoring**: All training sessions are actively monitored using TensorBoard and real-time logging. The results.csv file is used to track precision, recall, mAP@0.5, and mAP@0.5:0.95 metrics across each epoch. Training loss, validation loss, and confidence scores are also logged and visualized.
- **Checkpoints and Resume Support**: Model checkpoints are saved periodically using last.pt. This checkpointing mechanism allows the training to be resumed in case of hardware failure, Google Colab timeouts, or other interruptions. To minimize loss of progress, autosaving checkpoints every N epochs is enabled.
- **Evaluation Visualization**: Class-wise evaluation results are continuously validated using confusion matrices and metric curves. Normalized and raw confusion matrices, precision-recall curves, F1 score evolution graphs, and per-class statistics are all examined to detect overfitting, class imbalance issues, or poor performance on specific labels.
- **Error Tracking and Logging**: Any failures during evaluation, model export, or data handling are recorded with detailed tracebacks. A logging mechanism captures errors with timestamps, failed input paths, and Python stack traces for debugging.
- **Hyperparameter Tracking**: All training hyperparameters (such as learning rate, batch size, optimizer, image size) are saved in the opt.yaml configuration file to ensure exact reproducibility. This file is stored alongside the model artifacts and training logs.
- **Version Control**: Model versions are tagged based on dataset version, code commit, and training time. Each experiment is documented with model performance, class-specific issues, and test outcomes.

- **Manual Validation Steps**: Periodically, predictions are visualized and manually reviewed for random samples across each class. This qualitative verification ensures that high numerical metrics are not coincidentally achieved through class bias.
- **Threshold Optimization**: Detection confidence thresholds are manually fine-tuned based on ROC and PR curve observations, to find the optimal balance between false positives and false negatives.
- **Test Repeatability**: System-level tests (performance, integration, user acceptance) are scripted and parameterized to enable repeatable execution under varying conditions.
- **Storage and Artifact Management**: All model artifacts, logs, metrics, and visual outputs are stored in structured directories within Google Drive to avoid loss or misplacement. These artifacts are timestamped and linked to the corresponding test configuration.

Through this robust control infrastructure, every phase of development—from model training to deployment validation—is monitored, documented, and auditable.

9. Roles and Responsibilities

Person	Role	Responsibility
Arda Baran	ML Engineer	Model training, optimization
Baran Kuzucanlı	QA Analyst	Designing test cases, metric evaluation
Yakup Mert Akan	DevOps Engineer	Model deployment, environment setup
Sena Öztürk	UI/UX Researcher	Field usability evaluation, user feedback collection

10.Risk Severity Description Mitigation Strategy

This section outlines the primary risks that may arise during the development and deployment of the AI-based PPE and construction equipment detection system, along with applicable mitigation strategies for each. Critical concerns such as overfitting during training, labeling errors (mislabeling), class imbalance, real-time inference challenges, and data integrity issues are addressed proactively. By identifying and managing these risks early, the system is better positioned to deliver consistent and reliable results in real-world construction environments.

Risk	Severity	Description	Mitigation Strategy
Overfitting to Training Data	High	Model shows strong performance on training data but fails to generalize to unseen validation/test data. Overfitting risk increases with long training (200 epochs).	Apply data augmentation (mosaic, flip, HSV shifts); Enable cosine LR scheduling; Monitor validation loss vs training loss; Inspect PR/F1 curves for specific class overconfidence.
Mislabeling / Annotation Errors	High	Incorrect or inconsistent labels reduce model confidence and can lead to biased learning, especially in safety-critical environments like PPE compliance.	Use confusion matrix and curve analysis; Run cross-validation with Roboflow Annotator; Apply statistical label analysis; Review label distribution visually.
Class Imbalance	Medium	Certain PPE or equipment classes may be underrepresented, leading to biased detection performance for minority classes.	Use class-weighted loss; Oversample minority classes; Monitor class-wise mAP; Merge public datasets to rebalance distribution.
Inference Latency on Low-end Devices	Medium	Real-time inference may be impractical on edge devices or CPUs.	Enable half precision; Reduce input size (e.g., 416); Use simplified export (ONNX/TorchScript); Benchmark FPS and memory.
Colab Disconnections / Instability	High	Interrupted training due to idle timeout or instability on Google Colab.	Enable resume with last.pt; Store results on Google Drive; Automate backups.
Dataset Shift / Poor Generalization	Medium	Training data may not reflect real deployment environments.	Validate with varied domain-specific test sets; Collect deployment-specific data; Fine-tune with transfer learning.
Incorrect Thresholding	Low	Confidence thresholds may yield excessive false positives or negatives.	Use PR/ROC curves for tuning; Adjust thresholds to meet safety margins.

False Sense of Accuracy	Medium	High metrics may obscure class-specific failures.	Regularly inspect confusion matrix and class-wise metrics; Review predictions qualitatively with experts.
Label Format Errors / Parser Failures	Low	Bad label formatting may break evaluation or training.	Auto-validate label files before use; Repair or log corrupt labels.

11. Conclusion

Comprehensive tests ensure that the YOLOv9e model performs reliably and robustly for PPE and construction equipment detection. The system demonstrates excellent accuracy in detecting personal protective equipment such as helmets, gloves, and safety vests, with high precision and recall scores across most PPE-related classes. This highlights the model's effectiveness in scenarios where worker safety compliance is critical.

However, the evaluation also revealed some shortcomings in detecting certain classes of construction equipment, particularly in cases involving occlusion, scale variation, or less frequent machinery types. These inconsistencies suggest a need for rebalancing the training dataset, improving label consistency, and possibly incorporating additional high-quality images of construction machinery.

As a result, the model will undergo further refinement, including targeted retraining on underperforming equipment classes and enhanced augmentation strategies. This iterative improvement process aims to ensure consistent and trustworthy performance across all 34 categories before field deployment.