

# WORKPLACE SAFETY PPE DETECTION SYSTEM

## Midterm Project Proposal - ITAI 1378: Comp Vision-Artificial Intel

### *Team Members:*

- *DeMarcus Crump*
- *Chloe Tu*

*Project Tier: Tier 2*

*Date: October 30, 2025*





# THE PROBLEM

## What problem are you solving?

Failure to wear required Personal Protective Equipment (PPE) on construction sites leads to preventable injuries, deaths, and costly OSHA penalties. Manual monitoring is inconsistent and error-prone.

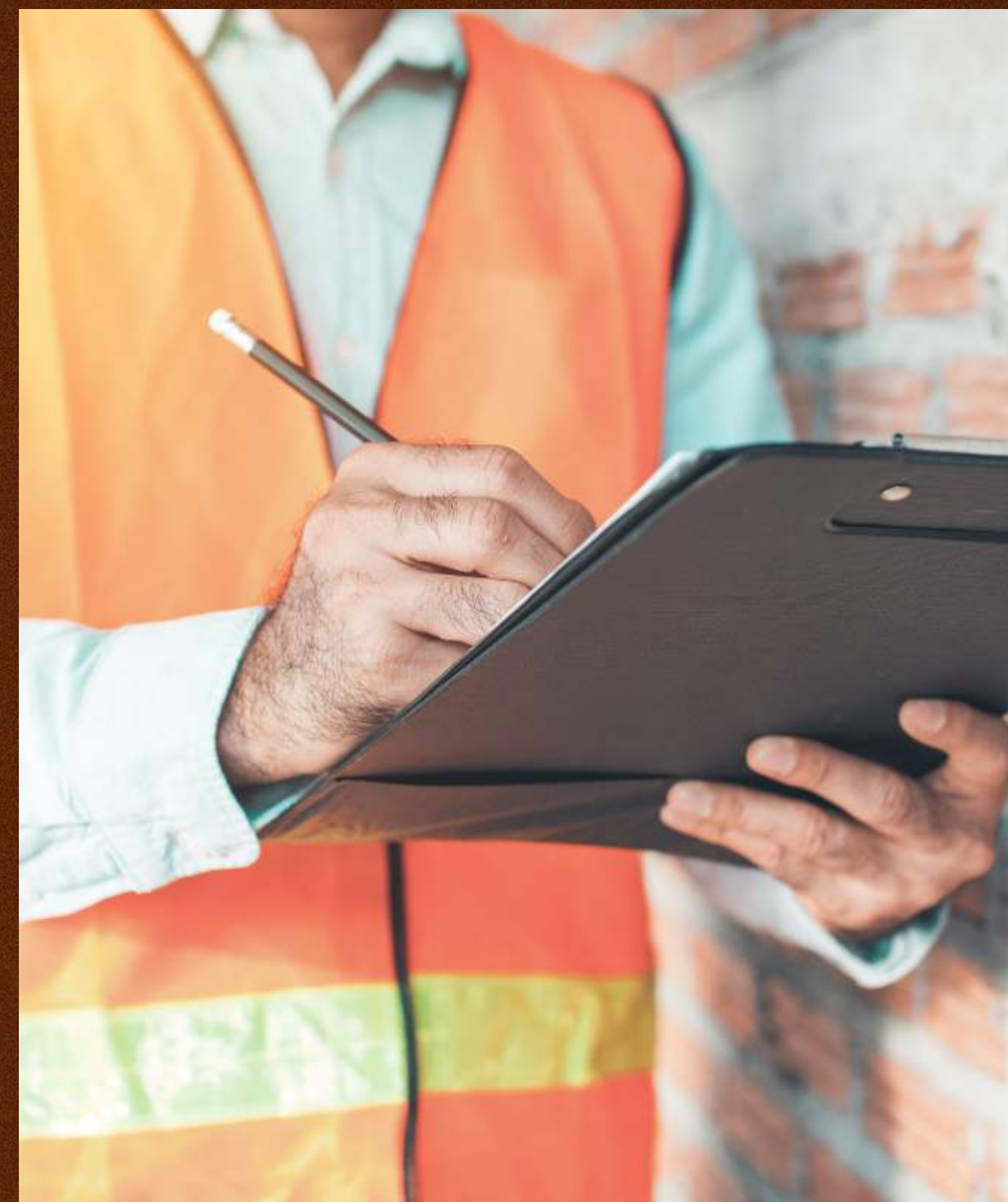
## Who cares about this problem?

- **Construction/safety managers** - enforce compliance
- **OSHA/government agencies** - regulatory compliance
- **Workers** - personal safety
- **Insurance companies** - reduce injury claims
- **Employers** - avoid penalties and costs

## Why is it important?

### Key Statistics:

- **4,764 workers died on the job in 2020 (US)**; nearly half in construction/material-moving [OSHA, BLS]
- **\$1 billion per week** in direct worker compensation costs for serious injuries
- **70% of fall incidents** involve lack of PPE
- **Only 60% of workers** consistently wear required PPE
- **Average injury cost:** ~\$42,000
- **Death cost:** >\$1.3 million
- **OSHA penalties:** \$15,625–\$156,259 per violation



- **Business Impact:** Automated detection reduces injuries, cuts insurance costs, ensures OSHA compliance, and saves millions annually.



# YOUR SOLUTION (OVERVIEW)



## What will your system do?

**One-Sentence Summary:** A real-time computer vision system that detects PPE on workers, verifies compliance, and generates alerts/reports for violations.

### Benefits:

- Automated continuous monitoring
- Real-time violation detection
- Data-driven safety reports
- Reduced risk of accidents



## How will it solve the problem?

Simple Workflow:

1. Camera Feed (Image/Video)



2. YOLOv8 Object Detection  
(detect helmets, vests, masks,  
etc.)



3. Compliance Rule Engine  
(check: helmet ✓, vest ✓, mask ✓)



4. Annotated Image + Compliance  
Report



5. Real-Time Alerts for Violations



# TECHNICAL APPROACH

## Technique

**Object Detection:** Multi-class detection of PPE items and workers

## Model

**YOLOv8** (You Only Look Once, version 8)

### Why YOLOv8:

- State-of-the-art accuracy (90%+ mAP on benchmarks)
- Real-time performance (30+ FPS on GPU; 10+ FPS on CPU)
- Handles 5–10 PPE classes easily
- Pre-trained weights available
- Well-documented and actively maintained [Ultralytics, 2025]

## Framework

### Why PyTorch + Ultralytics YOLO Library:

- PyTorch is industry-standard for deep learning
- Ultralytics provides simple API (`model.train()`)
- Google Colab support (free GPU)
- Minimal configuration needed



- **Justification:** Real-time PPE detection requires speed, accuracy, and multi-class capability. YOLOv8 balances all three, making it ideal for workplace safety applications.



# DATA PLAN

## Source

Dataset (Kaggle) : [Construction Site Safety Image Dataset](#)

## Size

~2,800 images (train/valid/test pre-split)



## Labels (Classes)

### Key PPE Classes:

- Hardhat, NO-Hardhat
- Safety Vest, NO-Safety Vest
- Mask, NO-Mask
- Person
- Additional: Gloves, machinery, vehicles (~25 total classes)

**Format:** YOLO annotation format (.txt files with bounding boxes)

## Preparation

### What's Already Done:

- ✓ Fully labeled with bounding boxes
- ✓ Pre-split into train/valid/test directories
- ✓ Pre-processed to standard resolution

### What We'll Do:

- Download from Kaggle
- Verify annotation format
- Apply augmentation during training (rotation, brightness, flips)
- YOLOv8 does this automatically

**License:** CC BY 4.0 (open for research and demo use)





# SYSTEM DIAGRAM



## Major Steps:

1. **Acquisition:** Capture image/video from workplace cameras.
2. **Preprocessing:** Resize and normalize input.
3. **Model Inference:** YOLOv8 detects PPE items and workers.
4. **Compliance Logic:** Rule engine checks for required PPE.
5. **Output:** Visual overlays + compliance reports

## Pipeline Flow

**Input: Camera Feed**  
(Image or Video)



### Preprocessing

- Resize to 640x640
- Normalize



### YOLOv8 Detection

- Detect PPE items
- Localize worker



### Compliance Check

- Helmet present?
- Vest present?
- Mask present?



### Output:

- Annotated Image
- Compliance Report
- Real-Time Alerts





# SUCCESS METRICS

Metric Type	Metric	Target	Rationale
Primary	mAP@50 (Detection Accuracy)	$\geq 80\%$	Acceptable for workplace safety
Primary	Precision	$\geq 88\%$	Minimize false positives (avoid false alarms)
Primary	Recall	$\geq 75\%$	Catch most actual violations
Secondary	Inference Speed (CPU)	<100ms per image	Real-time feasibility
Secondary	Per-Class F1 Score	$\geq 0.80$ per PPE type	Balanced performance across classes




### Why These Metrics:

- **mAP@50:** Standard object detection metric (mean average precision at 0.5 IoU)
- **Precision:** High precision avoids annoying workers with false alarms
- **Recall:** High recall ensures we catch most safety violations
- **Speed:** Real-time monitoring requires fast inference

**Benchmarks:** YOLOv8 achieves 90%+ mAP on standard benchmarks; our target of 80%+ is realistic for this dataset size [Ultralytics Performance Guide, 2025]



# WEEK-BY-WEEK PLAN

Week	Date	Task	Milestone
10	Oct 30	Download dataset, set up environment, initial exploration	Dataset ready; Midterm presentation submitted
11	Nov 6	Train YOLOv8 model on PPE dataset	Model working; achieves $\geq 75\%$ mAP
12	Nov 13	Test model, improve accuracy with augmentation	Good accuracy ( $\geq 80\%$ mAP)
13	Nov 20	Build compliance logic, create demo with visual overlays	Demo ready; system functional
14	Nov 27	Record demo video, final testing, documentation	Everything done; GitHub finalized
15	Dec 4	Present project in class	 Presentation day

## Key Milestones Detail:

- **Week 10:** Download Kaggle dataset, set up Google Colab with PyTorch/YOLOv8, create GitHub repo, explore data
- **Week 11:** Train YOLOv8-small for 50–100 epochs, monitor metrics, save best checkpoint
- **Week 12:** Evaluate on test set, apply augmentation, retrain, improve to 80%+ mAP
- **Week 13:** Write compliance rule engine, create inference script, annotate output images
- **Week 14:** Record 3–5 min demo video, finalize code/docs, prepare presentation
- **Week 15:** Live or recorded demo during final presentation





# CHALLENGES & BACKUP PLANS

## Potential Challenges & Solutions

Challenge	Probability	Plan B / Mitigation
Accuracy < 80%	Medium	Apply aggressive augmentation; try YOLOv8-medium; fine-tune on hard examples
Class imbalance (some PPE rare)	Medium	Use weighted loss functions; class-specific augmentation
Occluded PPE (workers blocked)	Medium	Train with occlusion patterns; adjust confidence thresholds
Lighting variability	High	YOLOv8 handles well with augmentation; test various lighting
Slow inference on CPU	Low	Use YOLOv8-nano (faster); optimize with quantization; ensure GPU available
Insufficient training data	Low	Combine with SH17 dataset (8,099 images); aggressive augmentation
Demo technical failure	Low	Record backup video in advance; prepare static image demo

### Risk Management:

- **If model underperforms:** Fine-tune hyperparameters, try larger model variant, add more data
- **If time runs short:** Focus on core features first (detection + basic compliance), add extras if time allows
- **If deployment issues:** Prepare demo on static images instead of live video processing





# RESOURCES NEEDED

Resource	Options / Notes	Cost
Compute	Google Colab (Free or Pro), Kaggle Notebooks	\$0–\$10/month
GPU Training	Colab free tier (12–16GB GPU)	\$0
CPU Inference	Any modern laptop/desktop	\$0
Frameworks	PyTorch, Ultralytics YOLO, OpenCV	\$0 (open source)
Dataset	Kaggle (direct download)	\$0
APIs	Kaggle API (optional), Ultralytics HUB (optional)	\$0 (free tiers)
Storage	Google Drive (15GB free), Colab storage	\$0
Code Repository	GitHub (public repo)	\$0

## Software Stack:

- Python 3.7+
- PyTorch (deep learning)
- Ultralytics YOLO (model training/inference)
- OpenCV (image processing & visualization)
- NumPy, Pandas, Matplotlib (data science tools)

**Total Estimated Cost:** \$0 (fully achievable with free resources)

**Optional:** Google Colab Pro (\$10/month) if free tier GPU is insufficient – but not required for this project





# REFERENCES & SOURCES

## Workplace Safety Statistics:

1. U.S. Bureau of Labor Statistics (2024). "Occupational Injuries and Illnesses." BLS.gov
2. OSHA, U.S. Department of Labor (2025). "Construction Industry Safety Resources." OSHA.gov

## Dataset:

3. Kaggle: [Construction Site Safety Image Dataset](#) Roboflow

## Technical References:

- 4 Ultralytics (2025). "YOLOv8 Documentation." : <https://docs.ultralytics.com>

3. Ultralytics (2025). "YOLO Performance Metrics Guide." : <https://docs.ultralytics.com/guides/yolo-performance-metrics/>

**Presentation Prepared:** October 28, 2025

**Submission Deadline:** October 30, 2025

**Final Project Due:** December 4, 2025

