EDGEGUARD: Real-Time Industrial Safety Monitoring using NPU Acceleration on NXP i.MX 8M Plus

Autonomous PPE Detection at the Edge

Authors:

Robert Rășcanu

Alexandra Bucătaru

Why Edge AI for Industrial Safety?

Problem Definition & Motivation

High Risk: Non-compliance with PPE regulations leads to severe injuries in industrial environments.
Cloud Limitations: Cloud-based analytics suffer from latency and high bandwidth costs.
Privacy Concerns: Streaming employee footage to external servers raises GDPR and privacy issues

Our Solution:

- **EdgeGuard:** An autonomous vision system that processes video feeds locally on the device.
- **Goal:** Immediate detection of "Hard Hats" vs. "No Helmet" without internet dependency.

DATASET & DATA ENGINEERING

Collection: Curated a custom dataset specifically for PPE (Personal Protective Equipment).
Classes: Defined two distinct classes: Helmet and Head (violation).
Augmentation: Applied techniques (noise, blur, rotation) to simulate harsh industrial lighting and conditions.
Annotation: Implemented "tight-fit" bounding box annotations to improve detection accuracy.

Model Training

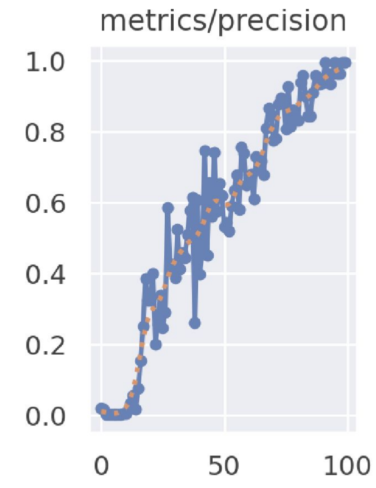
Transfer Learning with YOLOv5

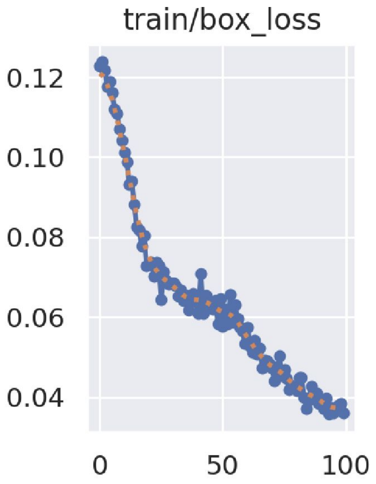
Architecture:

- Selected **YOLOv5 Nano** for its balance between speed and accuracy on embedded devices.
- Training Environment: Google Colab (for the ease of use GPUs).

Training Results:

- **mAP@0.5 (Mean Average Precision):** 87.4%
- **Precision:** 99.5% (Ensuring minimal false alarms).
- **Recall:** 79.5%





Optimization for Edge
The Challenge - Quantization

The Bottleneck:

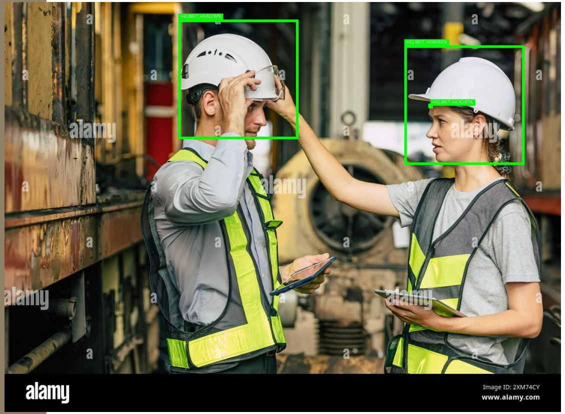
Standard deep learning models (***Float32***) are too large and slow for edge processors.

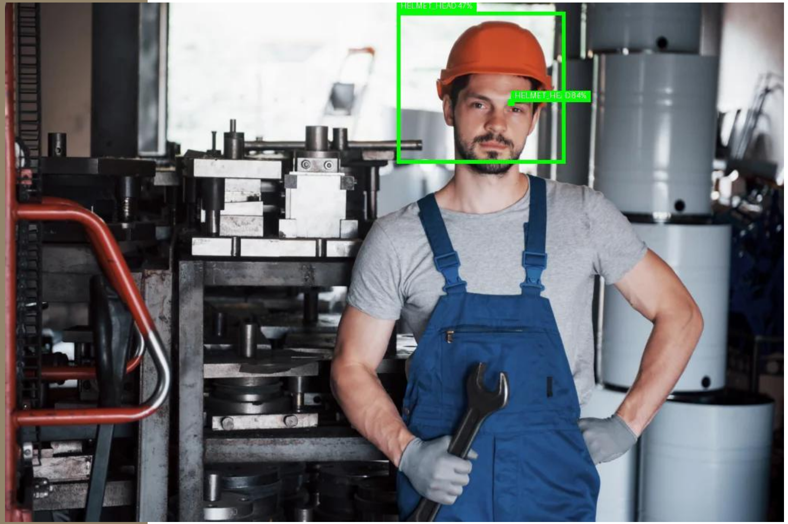
The Solution: Post-Training Quantization (PTQ).

- Converted weights from 32-bit floating-point to **8-bit integers (*INT8*)**.
- Reduced model size by **4x** (approx. 1.9 MB) with negligible accuracy loss.
- Ensured compatibility with the **VeriSilicon NPU** on the i.MX 8M Plus

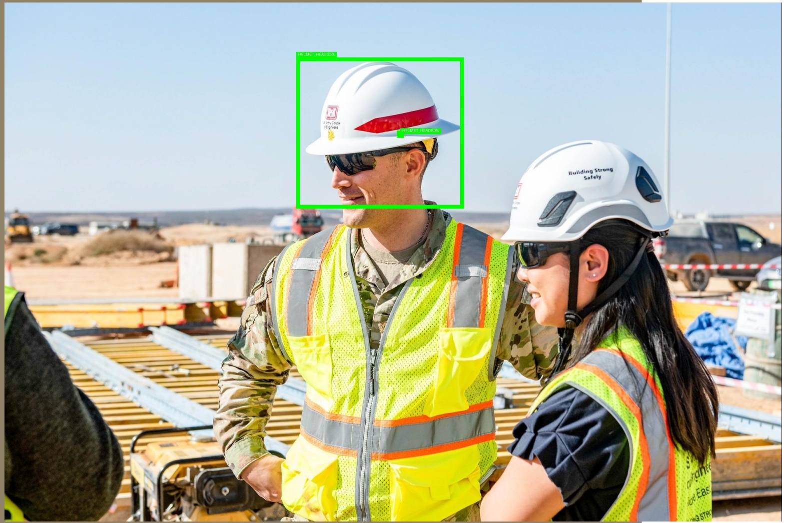
Video showcasing the optimization of running a model on the TPU comparing to the GPU

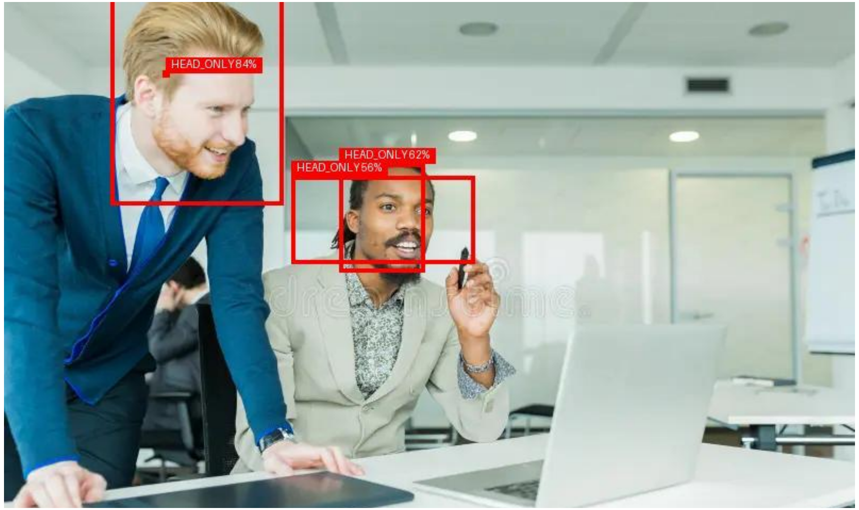
Model used: MobileNetV1





Model used: YOLOv5





<u>Model used</u>: YOLOv5

Performance Evolution: CPU vs. NPU

Comparative Analysis:

- **CPU Inference (Unoptimized):** ~4761 ms per frame (Failed attempt, non-real-time).
- **NPU Inference (MobileNet V1 Test): 2.96 ms** per frame.
- **NPU Inference (YOLOv5 Final):** ~11–12 ms per frame.

Throughput:

- Theoretical throughput of over **300 FPS** for classification tasks.
- Achieved robust real-time performance (>30 FPS) for object detection.

Metric	CPU (Standard)	NPU (EdgeGuard)	Improve-ment
Inference Time	4761 ms	11.2 ms	425x Faster
Throughput (FPS)	0.2 FPS	89.2 FPS	Real-Time
Model Size	7.5 MB (FP32)	1.9 MB (INT8)	4x Smaller
Power Efficiency	Critical	Instant	976
Safety Check	Delayed	Immediate	301

Conclusions & Future Work

1. Achieved a massive speed improvement using NPU acceleration compared to standard CPU execution.
2. The system is privacy-preserving (data stays on device) and cost-effective.

Future Steps: Integration with live IP cameras and audible alarms for immediate site warnings.