IoT Theft Detection System

Aaryans Nepal

CSC 494 - IoT (Spring 2026)

Problem and Goal

Retail theft is a persistent problem that results in billions of dollars in losses for businesses each year. While large retailers can afford advanced monitoring systems, small and mid-size stores often rely on basic CCTV setups and staff observation. Current solutions fall short:

- **CCTV is reactive**: Footage is reviewed *after* theft, not during.
- **Staff monitoring is inconsistent**: Employees can't watch every customer 24/7.
- **Commercial AI is expensive**: Enterprise systems cost thousands per month, leaving small/mid-size retailers vulnerable.

Project Description

A privacy-first IoT theft detection system that uses "Edge AI" to identify suspicious concealment behaviors in real-time. By processing video locally on a **Raspberry Pi 5**, the system flags events without expensive cloud subscriptions or invasive facial recognition.

Features and Requirements

1. **Real-time Pose Estimation**: Uses MediaPipe to track 33 human body landmarks at **8-12 FPS**.
2. **Concealment Logic**: Detects suspicious hand-to-pocket/hip movements based on configurable geometric thresholds.
3. **Instant Alerting**: Sends a "Proof of Theft" notification (screenshot + timestamp) to a **Telegram Bot** within 5 seconds.

4. **Local Event Logging**: All detections are saved to a local **SQLite** database with confidence scores and timestamps.
5. **Web Dashboard**: A **Flask-based** interface for store managers to review event history and tune system sensitivity.
6. **Privacy-by-Design**: Only processes skeletal coordinates locally; no raw video is ever uploaded to the cloud.

System Architecture

Success Metrics

1. **Performance**: MediaPipe Pose running at **8+ FPS** on Pi 5 without external accelerators.
2. **Accuracy**: Concealment events detected with a configurable confidence threshold to minimize false positives.
3. **Latency**: **<5 second delay** from the physical act of concealment to the mobile Telegram alert.
4. **Reliability**: System runs continuously for **8+ hours** (a full retail shift) without crashing.

AI Learning Goals

- **Software Domain**: Use AI to learn and optimize **Python Computer Vision** (OpenCV & MediaPipe) specifically for ARM-based processors.
- **Hardware Domain**: Use AI to learn efficient **Raspberry Pi Camera & Resource Management** to prevent overheating during long-duration monitoring.

Sprint 1 Goals (Weeks 1-6)

- **Core Detection MVP**: Get MediaPipe running on Pi 5 at 8+ FPS.
- **Heuristic Development**: Implement hand-to-hip distance tracking logic.
- **Local Storage**: Establish the SQLite database for basic event logging.

Sprint 2 Goals (Weeks 7-10)

- **Alert Integration**: Connect the detection engine to the Telegram Bot API for real-time notifications.
- **UI Development**: Build the Flask web dashboard for history review and system health monitoring.
- **Tuning**: Use collected data to reduce false positives (e.g., distinguishing between a phone and merchandise).

Current Risks and Unknowns

- **Thermal Management**: Will the Pi 5 throttle its CPU during 8+ hours of continuous AI inference?
- **Lighting Variability**: How will the Pi Camera Module 3's HDR handle harsh retail lighting or shadows?
- **Logic Refinement**: What is the "sweet spot" distance threshold to trigger an alert without annoying the user with false alarms?