Drift Sentinel: System & Model Technical Report

1. Dataset Details (Source & Nature)

Since **Drift Sentinel** operates as an unsupervised reliability layer, it treats data differently than a standard supervised model.

- **Training Source (Baseline):**
  - The core object detection logic is based on the **COCO Dataset** and industry-standard **Hard Hat & Safety Vest datasets** (e.g., Roboflow PPE Universe).
  - **Nature:** The training data consists of high-quality, well-lit, and clear images of industrial workers. This represents the "Ideal Operating Condition" (Ground Truth).
- **Inference Stream (Live Input):**
  - **Source:** Real-time video feed via webcam (or uploaded MP4 file).
  - **Nature:** The system processes **Unlabeled, Out-of-Distribution (OOD)** data. It does not look for "workers"; it looks for *deviations* from the baseline image statistics (e.g., shifts in texture, brightness, and sharpness) that indicate environmental degradation.
- **Calibration Data:**
  - The system uses the first **N-frames** of operation or a manual "Re-Calibrate" action to establish a statistical baseline (t_0) for the current environment.

2. Model(s) Used

The system employs a **Hybrid Architecture** combining Deep Learning concepts with Deterministic Computer Vision algorithms.

- **Primary Vision Model (Simulated/integrated):**
  - **Architecture: YOLOv8 (You Only Look Once, v8)**.
  - **Role:** Responsible for the actual detection of PPE (Helmets, Vests).
  - **Metric Monitored:** We do not monitor its *Accuracy* (which requires labels); we monitor its **Confidence Entropy**. When the model becomes uncertain (e.g., confidence drops from 90% to 45%), it signals reliability failure.
- **Drift Detection Engine (The Core Innovation):**
  - **Architecture:** Statistical Computer Vision (OpenCV-based).
  - **Algorithm 1: Laplacian Variance:** Used to calculate the "Blur Score" of the incoming frame. A drop in variance indicates loss of high-frequency edge details (Fog/Smoke).
  - **Algorithm 2: Mean Pixel Intensity:** Used to detect sensor occlusion (e.g., covered lens or blackout conditions).

    - **Algorithm 3: Risk Scoring Ensemble:** A weighted linear equation that combines the Slider Input (Simulation) with Real-Time Vision Metrics.

3. Known Limitations & Failure Cases

While robust, the system operates under specific constraints:

- **1. Sensor Noise vs. Sharpness (The "ISO Grain" Issue):**
    - *Failure Case:* In extremely low light, cameras increase ISO, creating grainy noise. The Laplacian algorithm can mistake this high-frequency noise for "sharp edges," potentially reporting a clear image when it is actually dark and noisy.
    - *Mitigation:* We apply a **Gaussian Blur preprocessing step** to smooth out grain before calculating the drift score.
- **2. Adversarial Brightness (Whiteout):**
    - *Failure Case:* If a flashlight or laser is shone directly into the lens, the "Darkness Detector" will see high brightness and assume the camera is working, even though the image is washed out.
- **3. Semantic Drift:**
    - *Limitation:* The system detects *Visual* drift (fog, blur). It cannot currently detect *Semantic* drift—for example, if workers start wearing a new color of vest that the model was never trained on, but the camera image itself is crystal clear.
- **4. Computational Latency:**
    - *Limitation:* Running pixel-level analysis on 1080p video causes lag. We limit the analysis stream to **320x240p** to ensure <100ms response time.

4. How the System is Working (Workflow Explanation)

The system functions as a **"Man-in-the-Middle" Safety Layer** between the camera and the Operator.

Step 1: Input Acquisition

- The system ingests a raw video frame from the webcam or file source.
- **Optimization:** The frame is immediately downscaled to 320x240 resolution for high-speed processing.

Step 2: Dual-Path Analysis (The Hybrid Logic)

- **Path A: Reality Check (OpenCV)**
    - The system calculates the **Laplacian Variance** (Sharpness) and **Mean Brightness**.
    - IF *Brightness* < *Threshold (110)* OR *Blur* < *Threshold (40)* $\rightarrow$ **Trigger Real Risk**.
- **Path B: Simulation Check (The Slider)**
    - The system reads the **"Drift Simulation"** slider value set by the supervisor (simulating forecasted weather conditions).

    - IF *Slider < 100%* $\rightarrow$ **Trigger Simulated Risk**.

Step 3: Risk Aggregation & Scoring

- The system calculates the **Global Drift Score** using a "Worst-Case" logic:
  $$Drift = Max(RealRisk, SimulatedRisk)$$
- This score is smoothed over time using a weighted moving average to prevent flickering.

Step 4: The Decision Layer (Risk Budget)

- The system possesses an **Operational Risk Budget (Fuel)** starting at 100%.
- As long as the Drift Score is high, the "Fuel" drains.
- **Logic:**
    - Risk < 30: **Low** (Fuel Regenerates).
    - 30 < Risk < 60: **Warning** (Fuel drains slowly).
    - Risk > 60: **CRITICAL** (Fuel drains fast).

Step 5: Intervention (Lockdown)

- When **Risk Level = CRITICAL**, the frontend triggers the **"SYSTEM LOCKDOWN"** overlay.
- This effectively blinds the operator/machine from acting on unreliable video data, forcing a Fail-Safe state.
- The event is logged to the **Black Box Audit Log** with a timestamp and root cause.