Project Update #2 (February 09, 2026)

CS 6678 - Advanced Machine Learning
Instructor: Dr. Leslie Kerby

George Lake

Current Project Focus

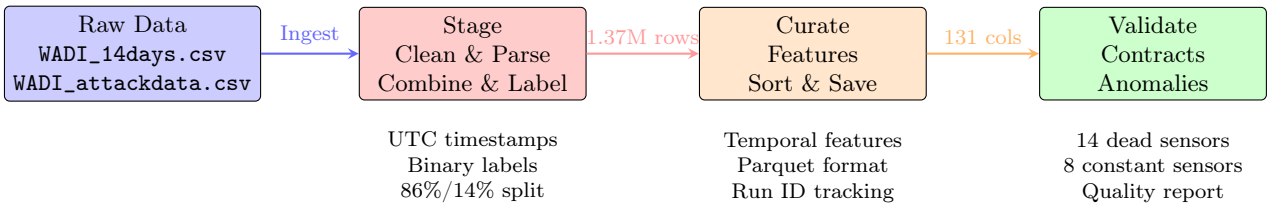
Research Question
Can machine learning distinguish between benign sensor failures and malicious cyber attacks in Industrial Control Systems (ICS)? Existing anomaly detection research treats this as binary classification (normal vs. abnormal), but operational response to failures versus attacks differs fundamentally; misclassification leads to either alarm fatigue or undetected persistence.

The current phase focuses on understanding the WaDi (Water Distribution) testbed dataset and establishing reproducible data preprocessing infrastructure. WaDi provides 1.2M normal operation observations and 173K cyberattack observations across 127 sensors over 16 days. The immediate work involves: (1) exploratory data analysis to understand sensor behavior, attack patterns, and data quality issues, (2) building a validation framework to ensure data integrity, and (3) identifying precise attack time windows to ensure future failure injection occurs only during verified normal operation periods.

Changes since last update
This is the first biweekly update. The core research question remains unchanged from the project sketch. Dataset selection evolved from SWaT to WaDi based on practical considerations; WaDi's download size is 850MB, while SWat's download size is 100GB.

Work Since Last Update

Data Preprocessing Pipeline Implementation Retrieved the dataset from iTrust (WaDi Testbed) and built a production-ready four-stage pipeline following: Ingest → Stage → Curate → Validate architecture:



What Was Learned

Domain Knowledge Required for Effective Validation
Building the validation framework revealed that generic checks ("is null?") miss critical quality issues in ICS data. Effective validation requires understanding physical constraints: valve positions must be 0-100%, flow rates cannot be negative, pump states are binary. This domain-driven approach identified 14 sensors (11%) with no recorded data (100% missing values - likely never connected) and 8 sensors with constant zero values (likely connected but never activated). Both categories provide no discriminative information, reducing effective feature space from 127 to approximately 105 sensors.

Temporal Leakage Risks Identified
Pipeline design revealed three critical requirements for future work: (1) train/test splits must be chronological to avoid leaking future sensor states into training, (2) any rolling window features must use only past observations, (3) failure injection must occur only during verified normal periods to prevent spurious correlations with attack labels.

Attack Metadata Integration Required Before Failure Injection
Pipeline design identified a blocking dependency: the current uniform labeling (entire attack dataset: `label=1`) is insufficient for safe failure injection. The `attack_description.xlsx` file contains precise boundaries for 15 distinct attack scenarios. Parsing this metadata is required to identify safe time windows for failure injection and avoid contaminating attack periods.

Current Challenges or Risks

Defining Realistic Sensor Failure Parameters
The core challenge is defining "realistic" sensor failures without ground truth examples. For each failure type, parameter ranges must be justified:

- **Drift:** What constitutes a plausible drift rate? 0.1% per minute, 1% per minute, etc. Too slow, and failures become undetectable; too fast, and they're trivially separable from attacks.
- **Bias:** What offset magnitude is realistic? ±5% of sensor range, ±20%? At what point does a biased sensor trigger operator intervention?
- **Noise:** How much precision degradation occurs in failing sensors? 2× baseline standard deviations, 5×? What noise characteristics (Gaussian, impulse, colored) are physically motivated?
- **Duration:** How long do failures persist before detection/repair? 5 minutes? 60 minutes?Duration affects both realism and class balance.

Without domain expertise in water distribution systems or access to real failure data, these parameters risk being arbitrary. Literature search on sensor failure characteristics is a critical next step.

Class Imbalance Strategy Requires Careful Design
The current dataset has 1.2M normal observations (86%) versus 173K attack observations (14%). Adding synthetic sensor failures will create a third class, but the final distribution depends critically on failure duration. Sensor failures are not single data points but time periods; a 30-minute drift failure affects 1,800 consecutive rows (assuming per-second sampling). Key design decisions:

- **Number of failure events:** How many distinct failure instances to inject (50? 100? 200?)
- **Duration distribution:** How long failures persist (5-60 minutes? Fixed vs. variable durations?)
- **Resulting class balance:** 100 failures at 30 minutes average = 180K failure rows (6% of dataset), which may be sufficient without aggressive sampling techniques.

Next Steps

Priority 1: Attack Metadata Integration
Parse `attack_description.xlsx` to extract precise temporal boundaries for all 15 attack scenarios. Update dataset labels from uniform `label=1` to time-window-specific labels based on actual attack start/end times. Visualize attack temporal distribution to assess: (1) Are attacks evenly distributed across 16 days or clustered in specific periods? (2) How much normal operation time exists between attacks for failure injection? (3) Does temporal distribution support chronological train/test splitting?
Decision to resolve: Can chronological split provide adequate attack representation in test set, or is alternative validation strategy required?

Priority 2: Failure Parameter Literature Search
Conduct focused literature review on sensor failure characteristics in SCADA/ICS systems, particularly water distribution. Target questions: What drift rates are documented in real sensor degradation studies? What bias magnitudes trigger operator alarms? What noise characteristics appear in failing sensors? How long do failures persist before detection/repair in operational systems?
Expected outcome: Justified parameter ranges for five failure types (drift, bias, noise, stuck-at, intermittent) grounded in domain literature.

Priority 3: Failure Injection Design Document
Document failure synthesis methodology including: parameter specifications for each failure type (drift rate, bias magnitude, noise level, duration), injection strategy (which sensors, when to inject, how to avoid attack windows), and labeling scheme (0=normal, 1=attack, 2=failure with metadata tracking).
Decision to resolve: How many failure instances to inject based on duration analysis and resulting class distribution?

Priority 4: Begin Failure Injection Implementation
Begin implementing failure injection framework including: functions for each failure type (drift, bias, noise, stuck-at, intermittent), logic for selecting injection timestamps and target sensors, constraints to prevent overlap with attack periods, and metadata logging system to track all injection parameters for reproducibility. Test initial implementation on a small data subset to verify methodology before processing full dataset.