Unsafe Events Detection in Smart Water Meter Infrastructure via Noise-Resilient Learning

*Ayanfeoluwa Oluyomi[†], Sahar Abedzadeh[‡], Shameek Bhattacharjee[‡], and Sajal K. Das[†]
[†]Dept. of Computer Science, Missouri University of Science and Technology, Rolla, MO 69401, USA
[‡]Dept. of Computer Science, Western Michigan University, Kalamazoo, MI 49008, USA
E-mail: {aoonzb, sdas}@mst.edu, {sahar.abedzadeh, shameek.bhattacharjee}@wmich.edu

***Abstract*—Residential smart water meters (SWMs) collect real-time water consumption data, enabling automated billing and peak period forecasting. The presence of unsafe events is typically detected via deviations from the benign profile of water usage. However, profiling the benign behavior is non-trivial for large-scale SWM networks because once deployed, the collected data already contain those events, biasing the benign profile. To address this challenge, we propose a real-time data-driven unsafe event detection framework for city-scale SWM networks that automatically learns the profile of benign behavior of water usage. Specifically, we first propose an optimal clustering of SWMs based on the recognition of residential similarity water usage to divide the SWM network infrastructure into clusters. Then we propose a mathematical invariant based on the absolute difference between two generalized means – one with positive and the other with negative order. Next, we propose a robust threshold learning approach utilizing a modified Hampel loss function that learns the robust detection thresholds even in the presence of unsafe events. Finally, we validated our proposed framework using a dataset of 1,099 SWMs over 2.5 years. Results show that our model detects unsafe events in the test set, even while learning from the training data with unlabeled unsafe events.**

***Index Terms*—Resilient Machine Learning, Anomaly Detection, Smart Water Distribution, Smart Living CPS**

I. Introduction

A water distribution network is a complex network of pipes and joints designed to ensure safe and dependable treated water delivery from the source tanks to the end consumers. However, the reliability can be jeopardized due to various unsafe events, affecting the safety and consistency of water supply. Installing smart water meters (SWMs) on individual households enables visibility of the entire water distribution network and supports key functions, such as automated billing and usage tracking. Specifically, the SWMs measure hourly water consumption for every house, and relay the readings to a data concentrator receiving the measurements from SWMs via wireless connection. Subsequently, such data are forwarded to the utility cloud (see Figure 1).

Motivation: Unsafe events in the context of this paper include SWM malfunctions and/or consequent negative values, large-scale radio frequency (RF) connectivity issues due to network device malfunctions, SWM software issues, water theft, or actual negative flow (i.e., water flowing to the opposite direction). Automated and real-time detection of such unsafe events in a city-scale SWM network is an open problem. This motivates our innovative work.

Since a typical city has hundreds of thousands of SWMs, running a per-meter level anomaly detection technique at the utility head-end will require significant computational and memory resources, which may not be scalable for large cities. Moreover, a human operator cannot possibly monitor the health of each SWM. This necessitates a lightweight wide-area anomaly detection technique at a coarse-grained resolution (cluster or zone-level) in real-time without manual supervision.

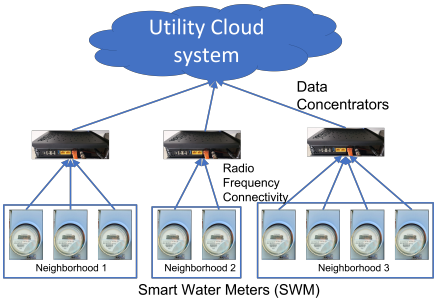


Fig. 1: Water usage data flow

A. Unsafe Events and Impacts in Consumer SWM Networks

Let us first enumerate possible unsafe events in the SWM networks and their impacts on the utilities and customers.

Water Leaks: This event shows up as higher water consumption readings from a group of smart meters. While high water usage can also be due to legitimate events (e.g., irrigation, gardening, or use of multiple water outlets), the ease of detecting leaks depends on the severity of the damage and the underlying causes, such as aging household piping, sabotage of pipes connecting a group of households, and natural disasters. Therefore, real-time detection of water leaks is highly important to the public water utilities. This is because undetected water leaks not only pose a financial burden on the consumers, resulting in higher bills, but also a reputation loss for the utility. Moreover, unaccounted water usage (due to water theft or leaks) leads to revenue loss and hence financial sustainability. The infrastructure (mainly pipes) will also be progressively damaged if the leak is not detected or handled early enough.

Backward Flow of Water: Backward flow or backflow occurs when the water pressure drops suddenly or completely fails, thus allowing contaminated water, liquids, or gases to

*A major part of the work was done when A. Oluyomi visited S. Bhattacharjee at Western Michigan University in 2023.

flow back into the clean water supply. Negative water pressure can be caused by a sudden change in the flow velocity, which can be due to the closing or opening of valves, starting or stopping a pump, or changes in the pipe geometry. Such an event shows up as negative water consumption from various smart water meters. Heavy rainfall can cause flooding, which can form puddles around the sprinkler devices on lawns/farms. During a pressure reversal, the contaminated water containing fertilizer, pesticides, etc. can be drawn back into the clean water supply line [8], leading to a public health concern. The dataset documentation in [4] provides further explanations for negative readings (e.g., organized water theft and SWM faults), which need to be detected promptly.

Sustained Missing Data: Our analysis of the SWM dataset revealed that many SWMs did not send water consumption data for several hours; then a single large consumption reading was sent corresponding to the total consumption during the outage. Such events can be due to SWM malfunctions, intermittent RF connectivity, and third-party SWM software issues, among others. Long periods of missing data impact the water utility operations by impeding the accuracy of data-driven processes (e.g., forecasting, billing, water conservation techniques), the service quality, and operational efficiency.

B. Aims, Objectives, and Challenges

To detect unsafe events in SWMs, we used hourly-recorded water consumption data from 1,099 houses [3]. Since different houses exhibit distinct water consumption patterns, our first objective is to design a clustering algorithm that can precisely identify and group houses based on the inter-dependence of such patterns. Water consumption in a household depends on multiple factors, such as the number of occupants, the nature of their employment, and individual preferences, which make the clustering method non-trivial. However, once such clusters are identified, we run an instance of the anomaly detection technique on the aggregate raw data from the entire cluster, ensuring the scalability of unsafe event detection. Once the relevant "cluster" originating the anomaly is identified, the utility can opportunistically invoke a more costly per-meter anomaly detection technique or a manual inspection only for those houses belonging to that identified cluster.

The most important technical challenge is the presence of unsafe events in the training dataset collected from actual deployments as this complicates the accurate characterization of the benign profile of any smart living cyber-physical system (CPS) such as the SWM networks. Hence, the objective is to automatically learn the benign profile of the operation while learning the thresholds of benign operation characterizing the event detection criterion, even with the inclusion of unsafe events in the training dataset. Our research contributions aim to address these unique objectives and challenges.

C. Contributions of This Paper

This paper proposes a novel time-series event detection framework for large-scale water distribution networks. The major contributions are summarized below.

(1) We design a clustering algorithm that partitions the residential smart water meters into smaller clusters using similarities in their consumption behavior. Creating such clusters, where the data in a cluster are highly positively correlated with each other, maximizes the invariant of the anomaly detection metric characterizing the benign behavior. Our approach uses mutual information that measures how much water meters are connected to each other owing to their complex, non-linear relationships. A graph-theoretic technique is used to partition the houses into clusters with strong correlations. The clustering approach promotes decentralized cluster-wise implementation, enabling straightforward identification of the cluster where an unsafe event has occurred. Subsequently, each smart water meter data within that cluster can be inspected individually.

(2) We propose an unsafe event detection model that leverages the absolute difference between the negative and positive orders of the generalized mean. Our findings demonstrate that this invariant provides optimal sensitivity to changes induced by unsafe events while remaining insensitive to benign fluctuations. The cyclo-stationarity of the human behavior in water consumption ensures the effectiveness of this approach.

(3) We develop a robust method to learn the thresholds of benign (normal) operations even when unsafe events occur during the training process. This method uses the Hampel three-part estimator to automatically learn unbiased thresholds that characterize normal behavior, even when unsafe events affect the invariant's structure during training. This contributes to the model's ability to distinguish normal behaviors from unsafe events in the test and deployment stage.

(4) We validate the proposed framework using the SWM dataset from Alicante, Spain. Experimental results demonstrate that our framework effectively detects unsafe events. The performance is evaluated by comparing its decisions with manually labeled water usage data, serving as ground truth.

D. Design Rationale and Theoretical Intuition

Before proceeding further, we outline the design rationale and theoretical intuition to shed light on the essential properties required for our framework to be effective in any CPS.

(a) The framework is relevant to smart living CPS applications, where the data from the physical world is readily affected by human behaviors, not just by the laws of physics. The clustering method combined with the invariant helps abstract this randomness into a time-invariant latent space.

(b) Our invariant design is such that it ameliorates the need to manually remove the negative and zero values before learning the benign profile essential in the existing time series invariant using Pythagorean Means [12]. Additionally, our proposed invariant contains properties that cause deviations under unsafe events due to the asymmetry of convexity properties in the generalized means with a negative order, as opposed to the generalized means computed with a positive order.

(c) The robust threshold learning involves a 'modified' Hampel loss function (a composite loss function) rather than the common practice of a regular loss function, which gives completely different fitness values depending on the error. This

explains why the learnt threshold is accurate in presence of unlabeled, unsafe events. Typically the inputs to a loss function in a regression-type learning problem are directly the regression errors. However, since the reduction of false alarms is the principal design challenge in time-series anomaly detection, we add more weights to the regression errors on moderately outlying points rather than in-lying points before passing it to the Hampel loss function that allows the threshold learner to be not too conservative. This explains good performance in false alarms, unlike the usual outcome where the use of robust loss functions comes at the cost of increased false positives.

(d) Our proposed framework will have less success if a positive covariance structure is absent (affecting the invariant stability) under feasible spatio-temporal granularities, and if a wide sense cyclo-stationarity is not observed in the extracted time-series invariant (affecting the safe margin generation).

The rest of the paper is organized as follows. The clustering algorithm is proposed in Section II along with the model of water consumption behavior. Section III describes the anomaly detection framework while IV presents the experimental results. Section V offers conclusions.

II. Proposed Clustering Algorithm

This section explores the need for clustering and detail its design, implementation, approach, and execution process. The clustering is done on a subset of data that are hand cleaned, and this process is done only once. Once the clusters are established, our invariant detection metric is calculated separately per cluster *using raw uncleaned data* as collected from the actual SWM from the wild. This reflects the reality of the problem of training to learn the profile of benign behavior in the presence of unlabeled unsafe events without human supervision. Then the thresholds of benign behavior are learned separately for each cluster. Next, in the test set, the same invariant is applied and compared with the thresholds learned during the training stage that specifies benign operation. If the invariant in the test violates the learned threshold, the proposed framework declares an unsafe event, else it infers benign.

A. Understanding Water Consumption Patterns

We used the Hellenic Data Service [3] containing hourly water usage data from 1,099 residential houses over 2 years and 5 months (January 2015 – May 2017). We used a 60-20-20 split for the training, cross-validation, and testing sets.

Fig. 2 shows that the mean water usage for corresponding hours of the entire training set is highly unpredictable. Hence, popular methods such as the cumulative sum (CUSUM), which requires prior knowledge of an in-control mean serving as a benign baseline, fail to solve this problem (see the proof in Sec. IV-D). As illustrated in Fig. 2, each line represents the average daily usage trend for an individual house (only 30 houses are displayed for brevity) over the hours of a day. Furthermore, it is important to highlight that the actual mean usage within a specific hour of different days exhibits substantial variations. Consequently, these fluctuations significantly influence the traditional metrics like the moving averages and simple means. Hence, the need for anomaly detection metrics that consider these variations, while profiling the benign behavior and establishing a detection criterion.

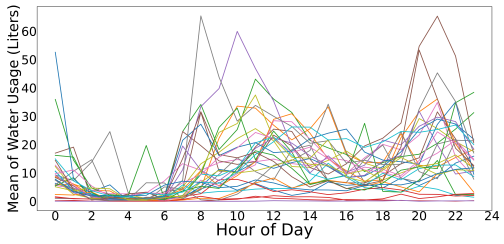


Fig. 2: Mean of water usage for corresponding hours

B. Residency Similarity Recognition (RSR) Algorithm

We observed a notable correlated increase in water usage, particularly during the early hours of the day across most houses, indicating the period when most household occupants get ready for the day. Similarly, the correlated drops in usage during the night time shows correlated behaviors in general.

1) Need for Clustering: Even under benign conditions, human behaviors are complex and non-linear. We need to identify subgroups where the correlation is high and treat them as single points of observation for visibility of the entire SWM network. This is why the clustering makes sense. The reason behind considering correlation as the basis for clustering is that this is a required property to provide stability to the invariant metric inspired by our work [1]. We showed that the ratio of Pythagorean means (e.g., harmonic mean to arithmetic mean ratio) gives a stationary time series if the data comes from a network of highly correlated random variables.

The strength of the positive correlation between water usage of houses becomes low when considering a randomly selected section of the city or the entire city. Low correlation adversely affects the effectiveness of the anomaly detection model because the derived invariant is not stable under benign conditions. Hence, clustering smart meters based on the temporal information led to an intelligent clustering algorithm, called the *Residency Similarity Recognition* (RSR) algorithm.

The primary objective of the RSR approach is to partition the houses into highly correlated/similar clusters. We use mutual information (MI) to capture similarity between houses due to non-linear dependence between the data coming from different meters. The RSR also enhances the robustness of the detection model by improving the invariant's stationarity (see Section III-A) and facilitates decentralized monitoring and localization of unsafe events to specific clusters.

2) Measuring Dependence between Meters: We use mutual information [6] to determine the dependencies among SWM meters. The MI quantifies the knowledge gained about one variable given another variable's value. Unlike popular correlation techniques such as Pearson correlation [17], the MI captures both non-linear and linear dependencies between variables, enabling a better understanding of smart living CPS, such as SWM networks. We calculate the MI as:

$$I(W_i; W_j) = \int\int p(w_i, w_j) \log \frac{p(w_i, w_j)}{p(w_i)p(w_j)} dw_i dw_j \quad (1)$$

where w_i and w_j are water usage for houses i and j over an hourly time window and $I(W_i; W_j)$ is the measure of MI between two house water usage w_i and w_j. The MI between any two variables is symmetric. After measuring the MI between every pair of houses, we use a graph-theoretic approach to obtain the correlated clusters. An MI close to zero indicates independence between variables while higher MI values indicate stronger mutual dependency.

3) Formulation as a Graph Problem: Assuming (theoretically) every house may have some dependencies with other houses, the initial formulation is an undirected complete graph, $G = (V, E)$, where $v \in V$ is the set of vertices (i.e., SWMs) and the $e \in E$ is the set of edges denoting the dependency between two vertices. The MI value, i.e., $w_e = I(W_i; W_j)$ between any pair of vertices is the edge weight denoting the strength of that dependency.

The goal is to find a set of correlated clusters such that the members within the cluster high dependency. Therefore, we assign a minimum acceptable MI value w_{cut} such that only edges with $w_e > w_{cut}$ are feasible, and the rest of the edges are deleted. Note that w_{cut} is a crucial hyperparameter, since a low w_{cut} results in a graph with more edges with low weights. Thus houses having low dependencies/correlations with other houses will be clustered together, which will weaken the invariant of the detection metric for that cluster. A weak invariant under benign conditions is not conducive to the detection of unsafe events or successful profiling of the benign behavior of a cluster. Conversely, a high w_{cut} will remove a lot of edges with considerable dependency strengths. This will cause a significant number of SWMs to be not included in any of the clusters. Section II-C discusses the method employed to obtain the optimal w_{cut} along with N, which is the maximum allowable number of houses for a single cluster. Next, we define some key terms and then describe the clustering algorithm.

A *Cluster* c_k is the k-th cluster containing a set of vertices with high similarity of water usage. The clusters are formed by incrementally adding vertices one at a time so as to maximize the edge weight w_e of a cluster. While forming a cluster, the current collection of vertices is the candidate cluster c. We need the following definitions inspired from [5] to understand the clustering.

Volume is defined as the sum of all edge weights w_e inside a candidate cluster c, where each edge $e = (v_i, v_j) \in E$ has both v_i and v_j in c, Thus,

$$vol(c) = \sum_{v_i, v_j \in c} w_e \tag{2}$$

Sumcut is the sum of weights of the edges that connect c to the rest of the graph. Each edge $e = (v_i, v_{i'}) \in E$ has one vertex v_i inside c and the other vertex $v_{i'}$ outside c. Thus,

$$sum_{cut}(c) = \sum_{v_i, v_{i'} | v_i \in c \wedge v'_i \notin c} w_{e_{i,i'}} \tag{3}$$

4) Description of Clustering Algorithm: Algorithm 1 returns a set C of clusters. The algorithm starts by verifying that

Algorithm 1: Algorithm for Clustering Houses

Data: Edge-list ($e_{i,j} = \{V_i, V_j, w_{e_{i,j}}\}$)
Result: Clusters $C = c_1, \ldots, c_k$
1 **while** *edgelist* $\neq \emptyset$ **do**
2 Select random $e_{i,j}$ such that $e_{i,j} \in edgelist$;
3 $c \leftarrow v_i, v_j$;
4 $vol(c) = w_{e_{i,j}}$;
5 $sum_{cut}(c) = \sum_{v_i, v'_i | v_i \in c \wedge v'_i \notin c} w_{e_{i,i'}}$;
6 $edgelist \setminus = \{(v_i, v_j)\}$;
7 **while** $vol(c) \leq sum_{cut}(c)$ *and* $count(c) \leq N$ **do**
8 Neighbors$(c) \leftarrow \{e_{i,i'} \mid (v_i, v'_i)\}$;
9 // $v_i \in c$, and $v'_i \notin c$
10 $sum_{cut}(c) = \sum_{e_{i,i'} \in \text{Neighbors}(c)} w_{e_{i,i'}}$;
11 // Sum of edge weights of vertices that connect c to the rest of the graph
12 $v_{i'+1} = \max[w_{e_{i,i'}}\ suchthat\ e_{i,i'} \in \text{Neighbors}(c)]$;
13 $c \leftarrow v_{i'+1}$;
14 $vol(c) += w_{e_{i,i'+1}}$;
15 $edgelist \setminus = v_{i'+1}$;
  end
16 **return** c_k;
17 remove all $(v, e) \in c$ from $edgelist$;
  end
18 **return** $C = c_1, \ldots, c_k$;

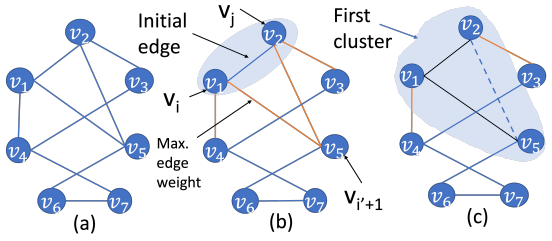


Fig. 3: A cluster formulation with RSR

the edgelist is not empty (line 1). Using Fig. 3 as an example, all edges in Fig. 3a have $w_e > w_{cut}$. The initial edge v_1, v_2 is randomly selected from the edge list to establish the process of building the first cluster that contains vertices v_1 and v_2 (lines 2 and 3). Then the algorithm calculates the candidate cluster's $vol(c)$ and $sum_{cut}(c)$ (lines 4 and 5). In Fig. 3b, $vol(c)$ is w_e of edge v_1v_2, while $sum_{cut}(c)$ is the sum of w_e for edges v_1v_4, v_1v_5, v_2v_5, and v_2v_3. Vertices v_1, v_2 are removed from the edgelist to prevent duplication (line 6).

The algorithm proceeds iteratively from lines 7 to 15 while $vol(c) \leq sum_{cut}(c)$ and the number of vertices, $count(c)$ within the candidate cluster c does not exceed N. Here, $Neighbors(c)$ is the set of all vertices that connect the candidate cluster to the rest of the graph (line 8). Then, we calculate the sum of w_e of all edges that connect the candidate cluster to the rest of the graph, i.e., the sum of all w_e in $Neighbors(c)$ (line 10).

Next, the algorithm selects the edge with the highest w_e from $Neighbors(c)$ (line 12), and includes the adjacent vertex $v_{i'+1}$ (currently outside the candidate cluster) into that candidate cluster (line 13). Then, w_e of edge $(v_i, v_{i'+1})$ is added to $vol(c)$ (line 14). This is illustrated in Fig. 3b and Fig. 3c,

where vertex v_5 is included in the cluster since the edge v_1v_5 has the highest w_e. Then $vol(c)$ is calculated as the sum of w_e for the edges v_1v_5 and v_1v_2.

The $sum_{cut}(c)$ is updated as the sum of w_e for edges v_1v_4, v_2v_3, and v_5v_6. Subsequently, all rows where vertex $v_{i'+1}$ and all its adjacent vertices are removed from the edgelist to prevent duplication (line 15). Once any of the two conditions on line 7 is violated, the algorithm returns cluster c_k along with the vertices formed in the process. For the example in Fig. 3c, vertices v_1, v_2, v_5 form the first cluster since the sum of w_e of v_1v_2 and v_1v_5 is greater than the sum of v_1v_4, v_2v_3, and v_5v_6. The vertices and adjacent vertices are removed from the edgelist (line 17) and the iterative process continues. Once the edge list becomes empty, the set C consisting of all $c_1, ..., c_k$ clusters formed is returned (line 18) and the algorithm terminates.

5) Using the RSR Algorithm with Cleaned Data: The RSR algorithm aims to create clusters that contain houses with highly correlated water usage patterns. We employ a hand-cleaned version of the dataset and apply the RSR algorithm to create a one-time cluster formation. In a real-life scenario, the clustering only has to be done only once in a few years to consider water patterns that might have changed over the years. After the clusters are formed, the raw uncleaned data from the corresponding SWMs within each cluster is used in the rest of this paper. The use of an uncleaned version of the data preserves all the challenges in the problem of 'automatically' learning the profile of benign behavior in the presence of unsafe events from a real smart living CPS. The benign profile and its thresholds, if learned accurately, can be used for detecting unsafe events in the deployment/test set. The process of cleaning the data is explained in Appendix VI-A.

C. Hyperparameter Learning for the RSR Algorithm

We derive the optimal w_{cut} and N by finding which combination of the candidate w^*_{cut} and N^* maximizes the strength of stationarity in the time-series data for all the clusters. To quantify the strength of stationarity, we use an Augmented Dickey-Fuller (ADF) [11] test with the time-series data $ad(t_\phi)^{c_k} = |HM(t_\phi)^{c_k} - AM(t_\phi)^{c_k}|$ (see Appendix VI-B for details). Here $ad(t_\phi)^{c_k}$ is obtained using an invariant proposed in our earlier work [2], which however fails under uncleaned SWM data containing unsafe events, thereby reducing the detection accuracy.

Now, let $\Delta y_t^{c_k}$ be the ADF test statistics for a cluster c_k and $\Delta_{avg}(C)$ the mean of $\Delta y_t^{c_k}$ produced for a given set C. Furthermore, let $\Upsilon(C')$ be the number of houses that do not belong to any cluster c_k (after C has been obtained) because of their low edge weight to adjacent houses.

By jointly minimizing a weighted sum, RSRmin $= w_\Delta \Delta_{avg}(C) + w_\Upsilon \Upsilon(C')$, we obtain the optimal values of w_{cut} and N (see Algorithm 2). Here, it is crucial to note that $w_\Delta > w_\Upsilon$, since we want more importance to be placed on $\Delta_{avg}(C)$. This choice reflects the importance of stationarity over the number of houses covered in our optimization process.

Algorithm 2: RSR Hyperparameter Learning

```
Result: N, w_cut
for N*, w*_cut do
    for w*_Δ, w*_Υ do
        Obtain set C with Algorithm 1;
        Δ_avg(C) = mean(Δy_t^{c_k});
        Υ(C') = number of houses not clustered;
        RSRmin ← w_Δ.Δ_avg(C) + w_Υ.Υ(C')
    end
end
N, w_cut, w_Δ, w_Υ = arg min_{N*, w*_cut, w*_Δ, w*_Υ} (RSRmin);
```

For each combination of w^*_{cut} and N^*, a unique set of clusters $C = c_1, ..., c_k$, is produced, and we calculate the RSRmin for that set C. Finally, we identify the global minima of RSRmin to determine the optimal values of $w_{cut} = 0.061$, $N = 80$, $w_\Delta = 0.99$, and $w_\Upsilon = 0.01$. The optimization process involves varying w^*_{cut} from 0.001 to 0.205 with a step size of 0.005, and N^* from 50 to 100 with a step size of 5 to obtain the optimal values for w_{cut} and N.

Clustering with Optimal Hyperparameters: Using the optimal values, we found 14 clusters, out of which the last 3 clusters had only two houses, making the event detection trivial. Hence, we report the performance on the remaining 11 clusters. We provide cluster-wise test statistics results in Appendix VI-C. We also presented a comparison of the performance of RSR algorithm with other conventional algorithms in Appendix VI-D.

III. Unsafe Event Detection Method

In this section, we first present our novel invariant design, followed by a method that learns a robust detection threshold that can successfully detect unsafe events in the test set, and finally the details of optimizing the hyperparameters of the invariant design and the detection threshold learning method.

A. Characterizing Benign Invariant

As outlined in section II, the clustering process produces clusters that maximize the correlation between houses included within the cluster. In this section, the uncleaned raw data collected from the wild are used for each house.

Consider a cluster c_k comprising $n(c_k)$ houses. Next, we compute an invariant metric $AD^{c_k}(t_\phi)$ for each cluster c_k at every time slot t_ϕ, where ϕ represents the hour of the day such that $\phi \in \{1, 24\}$. For the rest of this section, we remove the symbol c_k from all notations/equations, since the detection model runs per cluster c_k in the same way.

Let $w_i(t_\phi)$ be the water usage recorded by the i-th SWM at the time slot t corresponding to the ϕ hour of the day. Then the Generalized Mean (GM) value from all $w_i(t_\phi)$ from n meters in the cluster can be calculated as:

$$GM^{(p)}(t_\phi) = \left(\frac{1}{n} \sum_{i=1}^{n} \Big(w_i(t_\phi) \Big)^p \right)^{\frac{1}{p}} \tag{4}$$

where the parameter $p \in \mathbb{R}$ denotes the order of the GM statistic. Instead of GM, we propose a new $AD(t_\phi)$ invariant,

defined as the absolute difference between the two generalized mean values $GM^{p^-}(t_\phi)$ and $GM^{p^+}(t_\phi)$, written as:

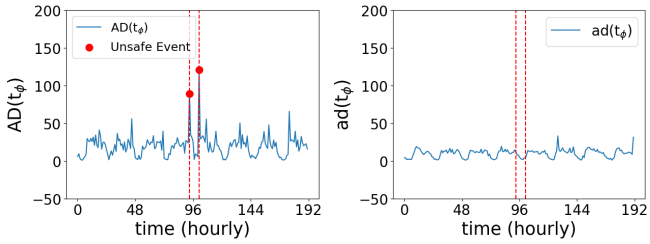
$$AD(t_\phi) = |GM^{p^-}(t_\phi) - GM^{p^+}(t_\phi)| \tag{5}$$

such that the first term $GM^{p^-}(t_\phi)$ has a strictly negative order parameter such that $p^- \in \mathbb{R}^- \setminus -1$, while the other term $GM^{p^+}(t_\phi)$ strictly has a positive order of p, where $p^+ \in \mathbb{R}^+$. This equation can be expanded into the following:

$$AD(t_\phi) = \left| \left(\frac{1}{n} \sum_{i=1}^{n} \left(w_i(t_\phi) \right)^{p^-} \right)^{\frac{1}{p^-}} - \left(\frac{1}{n} \sum_{i=1}^{n} \left(w_i(t_\phi) \right)^{p^+} \right)^{\frac{1}{p^+}} \right| \tag{6}$$

The orders p^+ and p^- is chosen by hyperparameter tuning such that it magnifies deviations during unsafe events. The $GM^{p^{-1}}$ i.e., HM, do not exist under zero and negative values, and hence is not a feasible value of p^-. An explanation is given in the appendix VI-E, validating the observed deviation in the invariant whenever an unsafe event occurs.

The optimization approach employed to determine p^+ and p^- is outlined in section III-E. To emphasize the importance of the clustering algorithm and demonstrate the robustness of the invariant under benign conditions, we illustrate a scenario in Fig. 4a displaying the behavior of $AD(t_\phi)$ for a cluster where an unsafe event has taken place in 93rd and 101st hour. It is clear from Fig. 4a, that our invariant shows a deviation matching the 93rd and 101st hour. In contrast, in the absence of clustering and previously proposed invariants (See fig. 4b), no considerable deviation is present.



(a) $AD(t_\phi)$ with our clustering (b) $ad(t_\phi)$ without clustering

Fig. 4: Effect of cluster level correlation on the invariance.

However, the $AD(t_\phi)$ is also vulnerable to occasional spikes under benign conditions; which reduces detection confidence. Hence, we implement a two-tier approach that smoothens the invariant into a stateless residual space, that provides the ability to improve detection accuracy while simultaneously reducing the false alarms. We generate safe margins around $AD(t_\phi)$ (Sec. III-B), resulting in stateless residuals (Sec. III-C), which are used to learn the appropriate robust threshold in Sec. III-D.

B. Safe Margin Generation

Stateless residuals measure the difference between the expected range of the invariant value at a time window t (called safe margin) and the actual invariant value at the same time window. We devise a method for determining the safe margin through temporal reasoning. This entails calculating certain indicators of central tendency and variability of the invariant within a specific temporal context. This idea is based on obtaining a range that wraps around each hour as seen in Fig. 5.

1) Expected Value of Invariant per Slot: Given t_ϕ as a unique hour slot in a day, the expected value for the invariant $E[AD(t_\phi)]$ is defined as the average of the AD_{t_ϕ} values at each corresponding hour t_ϕ (i.e. same ϕ) over the entire historical training data, mathematically written as:

$$E[AD(t_\phi)] = \frac{\sum_{h(\phi)} AD(t_\phi)}{h(\phi)} \tag{7}$$

where $h(\phi)$ is the total number of data entries corresponding to the ϕ-th hour of any given day over the historical training set. In our training dataset $h(\phi)$ is 522, and since ϕ corresponds to each hour of the day, we end up with 24 entries of $E[AD(t_\phi)]$, one corresponding to each hour.

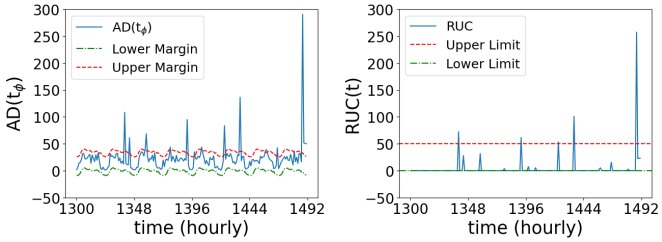
2) Safe Range of the Invariant: Now, we determine a safe margin around each $E[AD(t_\phi)]$ (location parameter) by multiplying a scalar factor ϵ to the median absolute deviation of all samples in $AD(t_\phi)$ (denoted as $MAD(AD(t))$ in the training set regardless of the hour ϕ. We use MAD as a measure of dispersion around the $E[AD(t_\phi)]$ given its robustness to outliers when compared to the standard deviations. The ϵ provisions the trade-off that allows for benign fluctuations in the metric to be tolerated. The resulting safe margin consists of both the upper margin $\Gamma_{high}(t_\phi)$ and lower margin $\Gamma_{low}(t_\phi)$ corresponding to any given hour ϕ of the day:

$$\Gamma_{high}(t_\phi) = E[AD(t_\phi)] + \epsilon.MAD(AD(t)) \tag{8}$$

$$\Gamma_{low}(t_\phi) = E[AD(t_\phi)] - \epsilon.MAD(AD(t)) \tag{9}$$

Finding the optimal ϵ is critical for minimizing false alarms (FA) while concurrently controlling missed detection (MD). This hyperparameter is determined in Section III-E. A large ϵ results in a wider range of safe margins, leading to more missed detections (MDs). Conversely, a smaller ϵ yields a narrower safe margin, resulting in an increase in false alarms.

Figure 5a shows the $AD(t_\phi)$ and the safe margins for a test set cluster after obtaining the safe margins from the training set. We only show t for 8 days (192 time slots).



(a) $AD(t_\phi)$ samples shown continuously for 8 days with the upper and lower safe margins (b) Stateless Residual shown continuously for 8 days with the optimal learnt detection standard limits

Fig. 5: Snapshot of the Detection model.

C. Formation of Stateless Residuals

As defined in Eqn. 10, we compute a signed residual distance $RUC(t)$ (Residual Under Curve (RUC)) by subtracting the observed sample $AD(t_\phi)$ at time t for a specific hour ϕ from the corresponding safe margins for that hour i.e. the

$\Gamma_{high}(t_\phi)$ and $\Gamma_{low}(t_\phi)$. The ϕ was omitted from subsequent equations because we are not particular about the specific hour of the day once the $RUC(t)$ has been obtained (See Fig. 5(b)). Mathematically,

$$RUC(t) = \begin{cases} AD(t_\phi) - \Gamma_{high}(t_\phi), & \text{if } AD(t_\phi) > \Gamma_{high}(t_\phi); \\ AD(t_\phi) - \Gamma_{low}(t_\phi), & \text{if } AD(t_\phi) < \Gamma_{low}(t_\phi); \\ 0, & \text{otherwise} \end{cases} \tag{10}$$

Specifically, Eq. 10 can theoretically produce zero (0), positive or negative values. Formally, the set of positive, negative, and zero stateless residuals is denoted as $RUC^{+}(t)$, $RUC^{-}(t)$, and $RUC^{0}(t)$ respectively.

D. Learning Resilient Thresholds of Stateless Residuals

The $RUC(t)$ is obtained from raw uncleaned data that already contains the unsafe events. Unless painstaking manual labeling is done to identify and remove $RUC(t)$ samples corresponding to the unsafe events/errors, the thresholds identified by typical regression or even max(residual) [13], will create biased thresholds that are unable to detect most events.

Therefore, we need to design a robust learning mechanism that automatically learns a correct event detection threshold without manual cleaning of abnormal events. Our method combines weighted regression with Hampel's three-part re-descending M-estimator. Formally, to learn the upper threshold - the inputs to the regression problem are the $RUC^{+}(t)$ values, the candidate upper threshold hypothesis space is parameterized by τ^{+}, and the optimal output learned threshold is τ_{max}. Similarly, to learn the lower threshold, the inputs are $RUC^{-}(t)$, the candidate threshold hypothesis space is τ^{-}, and the optimal lower threshold is τ_{min}. Since the method is the same for learning both thresholds, henceforth we use just $RUC(t)$ and τ notations for brevity.

The generic regression error is denoted as $r(t) = RUC(t) - \tau$. Now we make a design change by not providing $r(t)$ as direct inputs to the Hampel loss function. Instead, we define $s(t) = r(t).\lambda$, where $s(t)$ is a weighted regression error such that, $\lambda = w_2 \quad \forall \quad r(t) \geq 0$ and $\lambda = w_1 \quad \forall \quad r(t) < 0$ and $w_1 < w_2$. As seen in Algorithm 3, $r(t) \geq 0$ is checked, then $r(t).w_2$ is checked with the different a, b, and c values such that less importance are placed on the bigger outliers, while the weights provide the trade-off that reduces false alarms.

Hampel's Three-part Loss Function (HTF): is one of the robust estimators for outlier resilient learning in the broad class of M-estimators. The outliers in this context are the presence of $RUC(t)$ samples triggered by unsafe events already present in the training data that is used to learn the profile of benign operation. For weighted regression error $s(t)$, the HTF loss function is defined as:

$$\rho(s(t)) = \begin{cases} \frac{(s(t))^2}{2} & \text{if } |s(t)| \leq a \\ a|s(t)| - \frac{a^2}{2} & \text{if } a < |s(t)| \leq b \\ ab - \frac{a^2}{2} + \frac{a(c-b)}{2}\left(1 - \left(\frac{c-|s(t)|}{c-b}\right)^2\right), & \text{if } b < |s(t)| \leq c \\ ab - \frac{a^2}{2} + \frac{a(c-b)}{2} & \text{if } |s(t)| > c \end{cases} \tag{11}$$

the a, b, and c are different threshold values of the regression error such that $0 < a \leq b < c < \infty$, such that the corresponding loss function value depends on the magnitude of $s(t)$. The HTF has a special property, such that the first-order partial derivative of the estimator can be customized to gradually descend to a zero. This unique characteristic ensures that moderate and large outliers are entirely disregarded while smaller outliers are not entirely disregarded, thus, significantly enhancing the efficiency of the re-descending M-estimator by balancing legitimate benign fluctuations and large $RUC(t)$ spikes that are due to unsafe events [7].

Why HTF for Training Data: High values of $RUC(t)$ are less frequent (highly unsafe events) compared to moderate high values to low values signifying more benign conditions. Thus, high $s(t)$ should contribute less to the threshold. Thus, the range $|s(t)| \leq a$ is assigned the most importance (first part of Eqn 11). All the other parts of Eqn. 11) are such that they give decreasing importance with an increase in the $s(t)$. The next task is to determine the appropriate a, b, and c values in the HTF function.

Obtaining a, b and c in HTF: The mechanism to find a, b, and c is data-specific unlike the other parts of the paper. The K-means clustering algorithm was employed to categorize the $RUC(t)$ values into distinct clusters. K-means' ability to cluster without preconceived assumptions about the data's distribution made it an ideal choice for this purpose. The optimal number of k-means clusters was determined to be eight, as it effectively separated residual data into 8 clusters. Since the HTF requires four distinct regions, we need to combine the clusters into 4 segments demarcated by 3 values a, b, c. To ensure the robustness of the borders, we computed the MAD of two preceding and two succeeding clusters.

Obtaining Thresholds τ_{max} and τ_{min}: After determining the components for HTF, we use algorithm 3 to calculate the noise resilient upper limit τ_{max} which searches among $RUC^{+} \geq 0$. To obtain the lower limit τ_{min}, we searched among $RUC < 0$, and fig. 5b, shows the learned thresholds.

E. Hyperparameter Optimization

Next, we show how to find the optimal hyperparameters p^{+}, p^{-}, ϵ for the robust threshold τ_{max} and τ_{min}. As discussed in Appendix VI-E, when an unsafe event occurs, a significant deviation in the invariant can be guaranteed with the properties of the Schur concave and Schur convex. Therefore, we varied p^{+} and p^{-} from 1 to 5 and -2 to -6 respectively with a step size of 0.5, omitting $p^{-} = -1$ as it corresponds to Harmonic Mean that does not work with negative readings (a result of backflow events). We also varied the ϵ from 0 to 4 with a step size of 0.05. We generated different sets of $RUC(t)$ and learned the threshold τ_{max} and τ_{min}. To obtain the optimal combination of these hyperparameters, we formulated the optimization problem as an error minimization problem aimed at minimizing the number of False Alarms (FA) and Missed Detection (MD) for each combination of the parameters such that it minimizes $w_{fa}.FA + w_{md}.MD$, such that $w_{fa} = 0.6$ and $w_{md} = 0.4$ to emphasize the base rate prior between benign and unsafe events. We assign more weight to FA as occurrences of unsafe events are infrequent

Algorithm 3: Robust Learning of τ_{max}

Data: $[RUC^{c_k}(t), [\tau^+], , a, b, c, w1, w2]$
Result: τ_{max}
for τ^+ **do**
  if $r(t) \geq 0$ **then**
    if $(|r(t)| \cdot w2) \leq a$ **then**
      $cost \leftarrow \frac{1}{2} \cdot \left((r(t) \cdot w2)^2\right)$
    end
    else if $a < (|r(t)| \cdot w2) \leq b$ **then**
      $cost \leftarrow (a \cdot (|r(t)| \cdot w2)) - (0.5 \cdot (a^2))$
    end
    else if $b < (|r(t)| \cdot w2) \leq c$ **then**
      $cost \leftarrow (a \cdot b) - (0.5 \cdot (a^2)) + (a \cdot \frac{c-b}{2}) \cdot (1 - (\frac{c-(|r(t)| \cdot w2)}{c-b})^2)$
    end
    else if $(|r(t)| \cdot w2) > c$ **then**
      $cost \leftarrow (a \cdot b) - (0.5 \cdot (a^2)) + (a \cdot \frac{c-b}{2})$
    end
  end
  else
    if $(|r(t)| \cdot w1) \leq a$ **then**
      $cost \leftarrow 0.5 \cdot \left((|r(t)| \cdot w1)^2\right)$
    end
    else if $a < (|r(t)| \cdot w1) \leq b$ **then**
      $cost \leftarrow (a \cdot (|r(t)| \cdot w1)) - (0.5 \cdot (a^2))$
    end
    else if $b < (|r(t)| \cdot w1) \leq c$ **then**
      $cost \leftarrow (a \cdot b) - (0.5 \cdot (a^2)) + (a \cdot \frac{c-b}{2}) \cdot (1 - (\frac{c-(|r(t)| \cdot w1)}{c-b})^2)$
    end
    else if $(|r(t)| \cdot w1) > c$ **then**
      $cost \leftarrow (a \cdot b) - (0.5 \cdot (a^2)) + (a \cdot \frac{c-b}{2})$
    end
  end
end
$\tau_{max} = \arg\min_{\tau^+}(cost)$;

compared to benign. The w_{fa} and w_{md} in our framework should be modified accordingly for other systems. Algorithm 4 formalizes the hyperparameter learning.

Algorithm 4: Model Hyperparameter Learning

Data: Data clusters
Result: p^+, p^-, ϵ
for *each cluster* c_k **do**
  for p^+, p^- **do**
    Calculate $AD(t_\phi) = |GM^{p^-}(t_\phi) - GM^{p^+}(t_\phi)|$;
    for ϵ **do**
      Generate $\Gamma_{high}(t_\phi)$ and $\Gamma_{low}(t_\phi)$;
      Generate new RUC(t);
      Robust learning for τ_{max} and τ_{min};
      $error \leftarrow w_{fa}.FA + w_{md}.MD$;
    end
  end
  p^+_*, p^-_*, ϵ_* = $\arg\min_{p^+,p^-,\epsilon}(error)$;
end

F. Detection Criterion in Test Set

This section discusses the process of arriving at the predicted events using the two-tier detection approach.

1) Obtaining RUC(t) in Test Set: Let $AD(t^c(\phi)$ be the instantaneous current value of our invariant obtained from the observed water usage in cluster c_k at time t_ϕ in the test set.

Now we use the 24 distinct values of $\Gamma_{high}(t_\phi)$ (from Eqn. 8) and $\Gamma_{low}(t_\phi)$ (from Eqn. 9) known from the training phase, and compare it with the current $AD(t^c(\phi)$ sample using the Eqn. 10 by replacing $AD(t_\phi)$ with $AD(t^c(\phi)$, which produces the current $RUC(t^c)$ sample.

2) Detection Decision: We flag an unsafe event if the RUC gotten from the safe margins (tier 1) is not within the lower and upper threshold τ_{min} and τ_{max} respectively. This is given in equation 12 and shown with figure 5b.

$$RUC(t^c) : \begin{cases} \in [\tau_{\min},\ \tau_{\max}], & \text{No Events;} \\ \notin [\tau_{\min},\ \tau_{\max}], & \text{Unsafe Event.} \end{cases} \tag{12}$$

3) Predicted Events ($Z(t)$): We obtain the predicted event at time window t, denoted as $Z(t)$ calculated as the difference between $RUC(t^c)$ and the standard limits τ_{max} and τ_{min}.

$$Z(t) = \begin{cases} RUC(t^c) - \tau_{max}, & \text{if } RUC(t^c) > \tau_{max}; \\ RUC(t^c) - \tau_{min}, & \text{if } RUC(t^c) < \tau_{min}; \\ 0, & \text{otherwise} \end{cases} \tag{13}$$

4) Ground Truth Formation: We manually identified the unsafe events described in section I-A by rigorously studying the data and referring to available documentation for both cleaned and uncleaned versions of the data. It should be noted that the different events explained in Section I-A have two effects - which is either a large positive reading (+ve Events) or a negative reading (-ve Events) from one or more SWMs. For sustained missing data, we observed that a huge positive reading always follows the missing data, thus, the effect of this is seen as a moderately large positive number (+ve Events). To generate a ground truth events dataset D_{gt} from the raw data D_{raw} we applied the following rules/conditions to label the ground truth of events:

$$\zeta(t) = \begin{cases} 2 & \text{if } \exists \quad i \quad , w_i(t_\phi) > \theta_h \\ 3 & \text{if } \exists \quad i \quad , w_i(t_\phi) < \theta_l \\ 0 & \text{otherwise} \end{cases} \tag{14}$$

where θ_h and θ_l are a certain high water reading threshold and a certain low water reading threshold respectively, and $\zeta(t)$ is the time-stamped event label. We now discuss how we determined θ_h and θ_l for the SWM application.

Determining θ_l: Practically speaking, water usage should be strictly positive, thus, we postulate that any water usage significantly less than 0 is an unsafe event, hence $\theta_l = -1$.

Determining θ_h: Determining the appropriate water usage per hour is a complex task due to varying water usage per hour, day, week, and season, as well as household occupants and building types. Hence to obtain θ_h, we employ Chebyshev's Inequality on cleaned water usage data (the cleaning method is explained in appendix VI-A). We used a clean dataset because we can obtain the appropriate standard deviation (σ) and mean (μ) of the actual water consumption unlike the uncleaned data that contains large positive numbers (up to 13 million liters), these values influence the σ which in turn leads to taking larger step sizes to obtain θ_h.

Chebyshev's Inequality is represented as $P(|X - \mu| \geq g\sigma) \leq \frac{1}{g^2}$, where, X in our context is the water usage, g is scalar multiplicand of the standard deviation σ. This inequality provides a reliable means of setting the upper bounds on the proportion of data that can deviate significantly from the mean. With this, we calculate $\theta_h = \mu + g\sigma$, where μ and σ are the mean and standard deviation of the entire cleaned dataset.

The θ_h is common for all clusters for the following reason: When thresholds are calculated on a per-cluster basis, the RSR algorithm (section II), cluster-specific θ_h leads to significant impacts on μ and σ, causing the threshold to shift either to the far right or far left of the distribution. High water usage clusters can cause the threshold to shift to the far right, leading to the categorization of unsafe events as benign. Conversely, low water usage clusters can cause the threshold to shift to the left, labeling benign water usage as unsafe events. To address these challenges, the entire dataset is used to determine the threshold, hence introducing noise and ensuring the threshold value, θ_h, is more robust in dealing with these two extremes.

We obtain the value of $g = 30$ while calculating θ_h, which translated into having only 0.11% of the data outside of the bound. From this, we obtained $\theta_h = 740.7$. We validated θ_h with real-world statistics [9] of water usage per hour and we observe that θ_h is higher than the expected water usage per hour. Thus, if the water usage in an hour t from house i is greater than 740.6529 liters, we consider it an unsafe event.

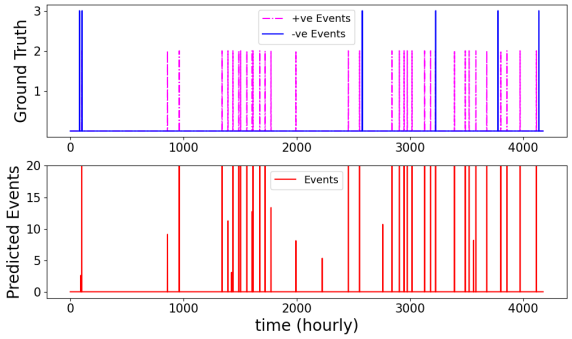


Fig. 6: Ground Truth vs $Z(t)$ for test data

Illustration: Figure 6 shows a comparison of predicted events $Z(t)$ (lower plot) with the ground truth $\zeta(t)$ (upper plot) where levels 2 and 3 correspond to events that lead to positive and negative outliers respectively. Thus, using $Z(t)$ helps us understand what kind of event might have happened.

G. Performance Evaluation Metrics

Apart from false alarm and missed detection counts and rates per cluster, we report a less used but NIST recommended performance metric for cyber-physical systems, called *Expected Time Between False Alarms*, defined as: $E(T_{fa}) = \frac{\sum_{i=1}^{nFA} T_E}{nFA}$, where nFA is the total number of false alarms, and T_E is the duration between two consecutive false alarms. This metric provides valuable insights into the occurrence and spacing between false alarms [13].

IV. Experimental Evaluation

A. Experimental Setup

We used residential household water usage data collected by Hellenic Data Service [3]. The dataset contains water usage from 1,099 houses from Jan 2015 - May 2017. We formed a training set of 521 days (Jan 2015 - June 2016), a cross-validation set of 174 days (May 2016 - Nov 2016) for hyperparameter selection, and a test set of 174 days (Nov. 2016 - May 2017) for performance evaluation, all containing uncleaned raw data.

As per Section II-B, we used the clustering approach to divide the houses into 11 clusters. For each cluster, we identified optimal hyperparameters using the cross-validation set with Algorithm 4. We fit the optimal hyperparameters into our model with the training data to extract optimal the $RUC^{c_k}(t)$ set per cluster, which is then fed as input to Algorithm 3 to obtain optimal thresholds $\tau_{max}^{c_k}$ and $\tau_{min}^{c_k}$ for each cluster c_k. The optimal hyperparameters and the thresholds are then applied to the test set. We calculated the invariant $AD^{c_k}(t_\phi^c)$ for each cluster by using the optimal $p+$ and $p-$ for that cluster (Sec III-A). Then we use the safe margin gotten during the training using the optimal ϵ and compare the invariant $AD^{c_k}(t_\phi^c)$ against the safe margins for each cluster to obtain the $RUC(t^c)$. We compare $RUC(t^c)$ against the robust thresholds using Eqn. 12 to represent the decision taken by our method on unsafe events. For the sake of visualization, we use the Eqn. 13 to compare the predicted unsafe events against the ground truth events.

B. Overall Performance with Learnt Hyperparameters

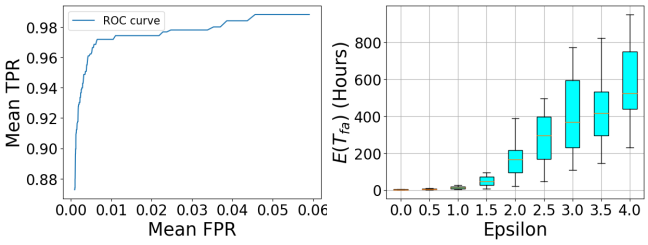
To assess the performance of our anomaly detection approach, we applied the learned $p^-, p^+, \epsilon, \tau_{\max}$, and $\tau_{\min}$ to each of the 11 clusters. The results are presented in table I, where c_k represents the cluster name, $\sum \zeta(t)$ is the total number of ground truth unsafe events gotten from the raw dataset, $\sum Z(t)$ is the total number of unsafe events our model was able to predict, the missed detection and false alarms are denoted as MD and FA respectively.

TABLE I: Experimental Results for the 11 Clusters

c_k	$\sum \zeta(t)$	$\sum Z(t)$	MD	FA	$E(T_{fa})(Hours)$
1	43	38	5	6	443.0
2	45	35	0	5	693.4
3	72	70	2	8	320.75
4	45	43	2	7	231.57
5	37	35	2	2	1307.0
6	83	72	11	6	504.83
7	37	32	5	3	824.67
8	213	213	0	6	543.67
9	45	40	5	4	773.25
10	51	46	5	1	4175.0
11	43	41	2	4	713.5
Total	704	665	39	52	

From Table I, we observe that over a testing time of 4,175 hours across all clusters- 94.5% of the unsafe events were detected, while the overall false alarm rate was 1.4% and the missed detection rate was 5.5%. Table I also provides the cluster-wise results for MD, FA, $\sum Z(t)$ and $\sum \zeta(t)$.

Furthermore, we give the average expected time between consecutive false alarms. The worst $E(T_{fa})$ is 231.57 hours for cluster 4, implying a FA once every 9 days, while cluster 10 has the highest $E(T_{fa})$ with a FA expected only once every 174 days. The average $E(T_{fa})$ across all clusters in 957.3 hours. In summary, our anomaly detection approach exhibits accuracy in the identification of unsafe events in SWM infrastructure based on learned benign behavioral models, while maintaining a low incidence of FA. While the false alarm rate is low, more research is needed to further reduce the frequency of false alarms.



(a) ROC curve (b) Mean $E(T_{fa})$

Fig. 7: Evaluating Sensitivity of Performance

C. Performance Sensitivity Analysis

Here we show sensitivity analysis using ϵ as a free parameter, examining its impact on true positive rate (TPR), false-positive rate (FPR), and $E(T_{fa})$.

Figure 7a presents the average Receiver Operating Characteristic (ROC) curve across the 11 clusters, serving as a robust indicator of our framework's performance. Notably, at a 98% TPR, the FPR remains impressively low, standing at just 0.037. This low FPR is crucial to Cyber-Physical Systems (CPS). This is because of the infrequent occurrences of events, the cost of false alarms is high. Specifically, Figure 7b provides insight into the expected time gap between two successive false alarms across varying values of ϵ.

D. Comparison with Existing Works

Now, we compare our proposed approach with four existing methods for anomaly detection: (i) CUSUM control [14], [15]; (ii) α-Winsorization [16]; (iii) the closest theoretically similar work proposed by us [12] that uses the idea of correlation clustering and extracts a Pythagorean mean ratio invariant (from time series) and Cauchy Lorentz loss function for a robust threshold learning algorithm under noisy training data; (iv) our invariant $AD(t_\phi)$ when combined with simple linear regression (i.e., square loss) to show the design benefit of the modified Hampel loss function.

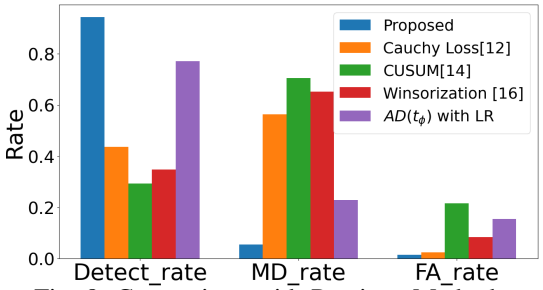


Fig. 8: Comparison with Previous Methods

Fig. 8 demonstrates that our proposed approach (marked in blue bars) outperforms other methods in terms of anomaly detection rate, missed detection, and false alarm rates.

Specifically, in terms of detection rate, our proposed method is two times better than [12] and approximately three times better than CUSUM and Winsorized Statistics. In terms of false alarm rate, our method exhibits only 1.4% while the framework in [12] has 8%. Thus, the proposed method is more than four times better compared to the next best approach.

To further understand the significant improvement provided by Algorithm 3, our results are 15 times better in terms of false alarm rate and 4.5 better in missed detection rate compared to linear regression used with our proposed invariant.

V. Conclusion

We propose a framework for real-time detection of unsafe events in SWM infrastructure using a time-series approach. First, a clustering algorithm groups SWMs with similar water usage and then utilizes the absolute difference between the negative and positive order of the generalized means to create an invariant that takes higher values under unsafe events and lower values under benign conditions. We proposed a modified application of Hampel three three-part loss function as a noise-resilient technique to learn the bounds of normal behavior in the presence of unsafe events in the training set. The results show that the proposed framework allows decentralized detection of unsafe events with high accuracy and low false-positive rates, and outperforms various well-known methods.

Acknowledgements: The work is supported by NSF grants CNS-2030611, CNS-2030624, and DGE-1433659.

References

[1] S. Bhattacharjee and S. K. Das, "Detection and Forensics against Stealthy Data Falsification in Smart Metering Infrastructure," *IEEE Transactions on Dependable and Secure Computing*, 18(1): 356–371, Jan 2021.

[2] S.Bhattacharjee, P. Madhavarapu, S. Silvestri, S.K. Das, "Attack Context Embedded Data Driven Trust Diagnostics in Smart Metering Infrastructure", *ACM Trans. on Security and Privacy*, 2021.

[3] Helix, "Smart Water Meter Consumption Time Series - Datasets - HELIX," Hellenic Data Service, 2020. https://data.hellenicdataservice.gr/dataset/78776f38-a58b-4a2a-a8f9-85b964fe5c95 (accessed Apr. 03, 2022).

[4] Daiad, "Trials Evaluation and Social Experiment Results," *European Commission's 7th Framework Programme*, 2017. daiad.eu/wp-contentuploads/201711D7.3_Trials_Evaluation_v1.0.pdf. (accessed Apr. 03, 2022).

[5] J. Islam et al., "Anomaly based Incident Detection in Large Scale Smart Transportation Systems," *ACM/IEEE 13th International Conference on Cyber-Physical Systems* (ICCPS), Milano, Italy, 2022, pp. 215-224.

[6] A. Kraskov, H. Stögbauer, and P. Grassberger, "Estimating Mutual Information," *Phys Rev E Stat Phys Plasmas Fluids Relat Interdiscip Topics*, 69(6), p. 16, 2004.

[7] D. F. Andrews, P. J. Bickel, F. R. Hampel, P. J. Huber, W. H. Rogers, and J. W. Tukey, *Robust Estimates of Location: Survey and Advances*, Princeton University Press, Princeton, NJ, USA, 1972.

[8] N. X. Truong and T. H. Thiep, "Selection of Hydrological and Hydraulic Models Applied in Urban Drainage," South Asian Research Journal of Engineering and Technology, vol. 3, no. 6, 2021.

[9] A. Gutierrez-Escolar, A. Castillo-Martinez, J. M. Gomez-Pulido, J. M. Gutierrez-Martinez, and E. Garcia-Lopez, "A New System for Households in Spain to Evaluate and Reduce Their Water Consumption," *Water*, 6(1): 181–195, Jan 2014.

[10] B. J. Frey and D. Dueck, "Clustering by Passing Messages between Data Points," *Science*, 315(5814): 972–976, Feb 2007.

[11] R. Mushtaq, "Augmented Dickey Fuller Test," *SSRN Electronic Journal*, Aug 2011.

[12] P. Roy, S. Bhattacharjee, S. Abedzadeh, and S. K. Das, "Noise Resilient Learning for Attack Detection in Smart Grid PMU Infrastructure" *IEEE Trans Dependable and Secure Computing*, to appear, 2024.

[13] D. I. Urbina et al., "Limiting the Impact of Stealthy Attacks on Industrial Control Systems," Proceedings of ACM SIGSAC Conference on Computer and Communications Security, 2016, doi: 10.1145/2976749.

[14] "6.3.2.3. Cusum Control Charts." Accessed: Jan. 31, 2024. [Online]. Available: https://www.itl.nist.gov/div898/handbook/pmc/section3/pmc323.htm

[15] V. R. Palleti, V. K. Mishra, C. M. Ahmed, and A. Mathur, "Can Replay Attacks Designed to Steal Water from Water Distribution Systems Remain Undetected?," *ACM Transactions on Cyber-Physical Systems*, 5(1): 1–19, 2020.

[16] R. Wilcox, "Trimming and Winsorization," Encyclopedia of Biostatistics, Feb. 2005.

[17] P. Schober and L. A. Schwarte, "Correlation coefficients: Appropriate use and interpretation," Anesth Analg, vol. 126, no. 5, pp. 1763–1768, May 2018.

VI. Appendix

A. Data Cleaning for Clustering

Let us discuss the data cleaning process required for forming the cluster with the RSR algorithm in section II-B5.

To clean a household's negative values at a time t, we analyzed adjacent values, selected the positive value closest in magnitude to the absolute value of the negative number, and summed them up. We consider t' of the selected value as the $corrective_{t'}$. If the sum is negative, we attribute it to the original hour and set the $corrective_{t'}$ to zero, else, the original hour becomes zero and the $corrective_{t'}$ retains the sum. This also was able to handle some large positive numbers. The rationale behind this approach lies in the assumption that the occurrence of negative values can be addressed by summing them up with adjacent positive values. This assumption is also backed up by further study into the data where it was observed that some instances of large negative readings were mirrored (or corrected) by approximately equal large instances of positive values, which we attributed to potential meter malfunctions. Summing these values refines the dataset to accurately depict the actual water usage.

If two or more positive values exist within an hour, we obtained and assigned their sum to that specific hour. This is because SWM could potentially capture and send readings more frequently to the data concentrator than what is expected.

For missing data over multiple hours, we identified the potential cause as network malfunction. We addressed this by averaging the next available value over the past t_m+1 missing values and recorded this average as the reading for the current hour and the t_m consecutive missing hours. This was able to handle some large positive numbers. This is because we assume that each hour preceding the spike contains some data rather than being entirely missing.

For applications where location or geographical data is present, the clustering could be made more accurate.

B. Details on Augmented Dickey Fuller (ADF) Test

In this section, we present the step-by-step processes we followed to test for the goodness of the RSR algorithm in forming clusters presented in section II that is crucial to achieving invariance in metrics.

Time-series: [2] proposed an invariant that works well on cleaned data based on the absolute difference between the Harmonic Mean (HM) and the Arithmetic mean (AM) at time t_ϕ. We use this to obtain the time series for the ADF, which is denoted by $ad(t_\phi)^{c_k}$ such that

$$ad(t_\phi)^{c_k} = |HM(t_\phi)^{c_k} - AM(t_\phi)^{c_k}| \tag{15}$$

where Eq. 15 is calculated per cluster c_k. Also, the potency of this combination was confirmed by preliminary results. The $ad(t_\phi)^{c_k}$ is then fed as the time series data into the ADF to assess the stability.

Augmented Dickey-Fuller Test (ADF): The ADF test is a statistical method used to determine if the time series is stationary, indicating that its statistical properties, such as mean and variance, remain constant over time. A more negative (i.e., higher in magnitude) test statistic indicates stronger evidence for stationarity [11]. This is defined as:

$$\Delta y_t = a + \beta t + \gamma y_{t-1} + \delta_1 \Delta y_{t-1} + \delta_2 \Delta y_{t-2} + \delta_3 \Delta y_{t-3} + \ldots + \delta_{q-1} \Delta y_{t-(q-1)} + e_t \tag{16}$$

Where Δy_t is the differenced time series; a is a constant; β is the time trend coefficient; γ is the coefficient of the lagged level of the time series; $\delta_1, \ldots, \delta_{q-1}$ are the lagged differences coefficients of the time series; and e_t is the error term.

In the framework of the ADF test, the null hypothesis (H_0) and alternative hypothesis (H_1) is formulated as follows:

H_0 : The time series data possesses a unit root

H_1 : The time series data is stationary

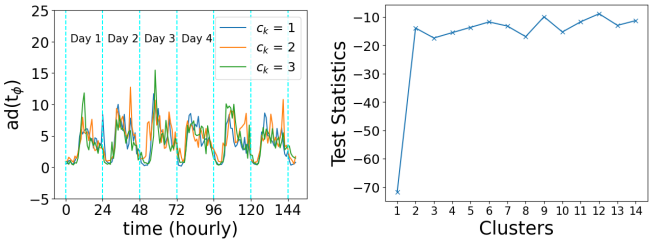
The presence of a unit root in a time series data analysis indicates a persistent stochastic trend, making it non-stationary, its absence however signifies stationarity. We embraced this concept in our study. To use this test, once a cluster is formed, we aggregate the water usage of each house based on equation 15 and then subject $ad(t_\phi)^{c_k}$ to the ADF test (equation 16) to assess the stationarity which would indicate the efficacy of the RSR algorithm. We provided our hypothesis based on this and explained it with two clauses.

H_0 : The time series data possesses a unit root

H_1 : RSR is effective; time series data is stationary

Clause A (H_1): The ADF test can determine if a cluster is stationary given the expected cyclostationary behavior of $ad(t_\phi)^{c_k}$ for corresponding hours of each day, thus, indicating the RSR's effectiveness of clustering houses with similar water usage. The concept is depicted in fig. 9a. The $ad(t_\phi)^{c_k}$ for 3 clusters is shown for 6 days. We observe the stationarity of the $ad(t_\phi)^{c_k}$ indicating the efficacy of the RSR in clustering houses with similar water usage patterns, thus, the visual similarity for each corresponding hours of each day, i.e., the $ad(t_\phi)^{c_k}$ at $t = 00:00$ on day 1 is similar to the $ad(t_\phi)^{c_k}$ at $t = 00:00$ from day 2 to day 6.

Clause B (H_0): Conversely, if the algorithm is ineffective, it will group houses with dissimilar water usage patterns leading to a unit root, thus, diverging the $ad(t_\phi)^{c_k}$ significantly, leading to non-stationarity when subjected to the ADF test.



(a) Stationary $ad(t_\phi)^{c_k}$ with RSR (b) Test statistics for all clusters

Fig. 9: RSR results for $w_{cut} = 0.061$ and $N = 80$

C. Results of Test Statistics for Each Cluster

After obtaining the optimal parameter for w_{cut} and N in section II-C, we inserted these parameters into the RSR algorithm to calculate $ad(t_\phi)^{c_k}$. We present a comprehensive breakdown of the $\Delta y_k^{c_k}$ per cluster in Fig. 9b. We observe that the RSR initially clustered strongly correlated houses; however, as the iterations progressed, these correlations weakened, resulting in higher test statistics. Despite this, all formed clusters passed the stationarity test. The stationarity test is determined by subjecting the test statistics to a critical value, and if the test statistic is greater than the critical value, the null hypothesis cannot be rejected, suggesting that the time series is non-stationary.

D. Comparison of the RSR Algorithm

We compared the RSR algorithm with time series K-means (TSKM), Agglomerative clustering (Agg), and Density-Based Spatial Clustering of Applications with Noise (DBSCAN). We provide the results in Table II where we observe that DBScan and RSR perform better than TSKM and Agg in terms of the percentage of clusters passing the stationarity test. We obtained lower test statistics of -17.48 for the proposed RSR when compared to DBSCAN which is -11.03, indicating that RSR gives more stable invariants compared to DBSCAN even with the same percentage of stationary clusters (those that pass the stationarity test).

TABLE II: Comparison with Other Clustering Algorithms

Algorithm	TSKM	Agg	DBSCAN	**RSR**
Number of Clusters Formed	8	18	1	**11**
Percentage of stationary cluster (%)	62.5	77.8	100	**100**

E. Deviation Reasoning

Here, we give a theoretical explanation of why our proposed $AD(t_\phi)$ in section III deviates under unsafe events.

In mathematics, a $Schur-convex$ function, alternatively referred to as an S-convex or isotonic, is a type of function denoted as $f : \mathbb{R}^d \rightarrow \mathbb{R}$. This mathematical concept has a fundamental property: for any pair of vectors, namely, x and y belonging to the d-dimensional real space $\mathbb{R}^d$, if it holds true that x is majorized by y or x is more spread out than y, then this function satisfies the condition $f(x) \leq f(y)$. This property implies that the function assigns smaller values to vectors that are "dominated" by other vectors.

In contrast, f is $Schur-concave$, which means that any pair of vectors such that x is majorized by y, then this function satisfies the condition $f(x) \geq f(y)$. Let's start interpreting the behavior of the generalized mean,$GM^p(x_1, x_2, \ldots, x_n)$ in Eqn. 4, in terms of Schur concavity and Schur convexity:

If $p < 0$: the generalized mean $GM^p \equiv GM^{p^-}$ is Schur concave. In terms of the power mean formula, since smaller values are given more weight, an unsafe event causing a large decrease (e.g., negative water flow) in one or more numbers in the series $(x_1, x_2, \ldots, x_n)$, causes the resulting GM^{p^-} value to decrease sharply, and the magnitude of the decrease is controlled by the value of p^-.

If $p \geq 1$: the generalized mean $GM^p \equiv GM^{p^+}$ is Schur convex. In terms of the power mean formula, since smaller values are given more weight, an unsafe event causing a large increase (e.g., leakage, network malfunction) in one or more numbers in the series $(x_1, x_2, \ldots, x_n)$, causes the resulting GM^{p^+} value to increase sharply, and the magnitude of the increase is controlled by the value of p^+.

Since our invariant definition contains a combination of both GM^{p^+} and GM^{p^-} such that

$$AD(t_\phi) = |GM^{p^-}(t_\phi) - GM^{p^+}(t_\phi)|$$

regardless of the nature of the unsafe event. Given we are taking the absolute difference between GM^{p^+} and GM^{p^-}, our invariant will increase under unsafe events following properties listed in this work, as long as $p \neq -1$

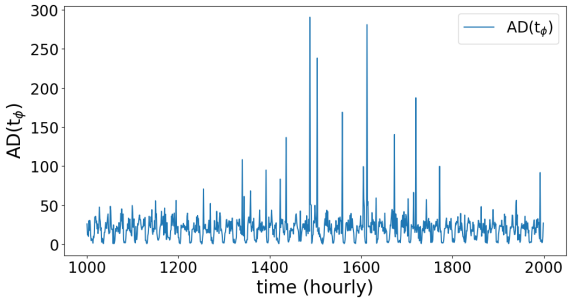


Fig. 10: The Absolute Difference ($AD(t_\phi)$)

As illustrated in Fig. 10, there are discernible spikes within a specific time window. These spikes correspond to exceptionally high values of $AD(t_\phi)$ in Equation 5. This phenomenon arises when there is a substantial separation between GM^{p^-} and GM^{p^+}. In such cases, either GM^{p^-} or GM^{p^+} assumes a significantly elevated value.

When GM^{p^-} exhibits a very large value, it implies that within the given time window, numerous instances of water usage data fall within the range of very small to moderate values, while larger values are relatively less prominent. Conversely, when GM^{p^+} gets a high value, it signifies that within the same time window, very large to moderate water usage values are given priority over smaller values.

Thus, the detection of highly significant small-scale water usage or water drainage events can be accomplished through the analysis of GM^p, where $p < 1$. Similarly, the identification of substantial large-scale water usage events can be achieved by scrutinizing GM^p, where $p > 1$.