On the Role of Re-Descending M-estimators in Resilient Anomaly Detection for Smart Living CPS

Sahar Abedzadeh, Shameek Bhattacharjee
Department of Computer Science, Western Michigan University, Kalamazoo, USA
E-mail: sahar.abedzadeh@wmich.edu, shameek.bhattacharjee@wmich.edu

***Abstract*—Anomaly-based attack detection methods that rely on learning the benign profile of operation are commonly used for identifying data falsification attacks and faults in cyber-physical systems. However, most works do not assume the presence of attacks while training the anomaly detectors- and their impact on eventual anomaly detection performance during the test set. Some robust learning methods *overcompensate mitigation which leads to increased false positives in the absence of attacks/threats during training*. To achieve this balance, this paper proposes a framework to enhance the robustness of previous anomaly detection frameworks in smart living applications, by introducing three profound design changes for threshold learning of time series anomaly detectors:(1) Tukey biweight loss function instead of square loss function (2) adding quantile weights to regression errors of Tukey (3) modifying the definition of empirical cost function from MSE to the harmonic mean of quantile weighted Tukey losses. We show that these changes mitigate performance degradation in anomaly detectors caused by untargeted poisoning attacks during training- while is simultaneously able to prevent false alarms in the absence of such training set attacks. We evaluate our work using a proof of concept that uses state-of-the-art anomaly detection in smart living CPS that detects false data injection in smart metering.**

***Index Terms*—Explainable Machine Learning, Smart Living CPS, Resilient AI/ML, Data Poisoning attacks.**

I. INTRODUCTION

Data poisoning attacks [1] are recognized as a significant threat to all machine learning (ML) tasks - where an attacker injects false data as inputs while training an ML model. This degrades the accuracy of predictive outcomes made by the ML model during the test set.

On the other hand, many smart living CPS applications use anomaly based attack detection frameworks to detect false data injection attacks. These methods operate by learning the profile of benign operation via an anomaly detection metric calculated from CPS sensor data. Then it learns boundaries of the anomaly metric. Such learnt boundaries act detection thresholds and hence form the attack detection criterion that is used in test set. Presence of any attack in the test set is expected to cause the anomaly detection metric to violate the learned thresholds of the anomaly detection metric. Naturally, it is clear that unknown attacks already present while training such anomaly detectors, will skew the anomaly detection metric and consequently the learnt detection thresholds.

Current research efforts on data poisoning have predominantly focused on advanced data poisoning attack strategies that bypass popular deep learning methods and standard learning methods [6, 7] and industrial CPS [15]. Research in mitigation of even simple data poisoning attacks in the context of smart living CPS, has been limited. In this paper, we investigate how to mitigate impact of simple poisoning attacks on the accuracy of anomaly based attack detectors in smart living CPS, while not sacrificing false alarm performance, without making any parametric assumptions on the data/attacks.

A. Motivation and Research Gaps in Smart Living CPS

A series of previous works [2, 4, 5] had laid a foundation for a unified science of anomaly based detection of attacks and emergency events in smart living CPS. Specifically, in [2] the CPS sensor data is collected from smart meters and the goal was to detect presence of organized data falsification (e.g. electricity theft). Similarly, in [5], the same overall approach is applied to detect emergency events in a smart transportation network by learning the profile of benign behavior derived from speed data collected from road side sensors.

All the above works in anomaly detection for smart living CPS follows the following unified recipe: (1) First, a latent space embedding method transforms the raw CPS sensor data into a time series latent space (in a lower dimension) that serves as an anomaly detection metric; (2) Second, regression based learning is used that takes the latent space as its training data inputs and learns the best fit boundaries of the latent space that specify the thresholds of benign operation. (3) Third, the latent space contains mathematical properties that show deviations beyond the learnt thresholds under attacks [4, 2] or unsafe events [5] in the test set, hence detecting the threats. The particular challenge in smart living CPS, unlike industrial CPS, is the reduction of false alarms since benign fluctuations in the aggregate data patterns are more common due to human behaviors which are more random.

Additionally, fundamentals of machine learning assume that training and test set data come from the same distribution. Thus learning the thresholds of the latent space accurately requires training on data gathered from the live deployment of CPS sensors. However, attacks may be already present in the live deployment of CPS sensors from which anomaly detectors learn benign profile of operation. Hence, the latent space will be altered and the inputs to the threshold learner (which are points in the latent space) are altered; which in turn alters the thresholds of the anomaly detector.

Due to the altered thresholds, there is a degradation in the missed detection (actual attacks in the test set go undetected) or the false alarms (benign samples in the test set are erroneously inferred as attacks) depending on the direction of change. We justify the above claim later in Sec. III-B with a case study on a state of the art anomaly detection framework applied to smart metering infrastructure. We find that simple data falsification attacks during training that does not require any advanced knowledge by an adversary ensure that data falsification attacks go undetected in the test set, due to the biasing of learnt anomaly detection thresholds. Hence, we need to introduce modifications to the basic design of time series anomaly detection frameworks for smart living CPS.

One method in [3] offers one of the first mitigation techniques against data poisoning attacks using two robust M-estimators (Huber and Cauchy Loss) that replaces the traditional L2 or L1 norm used in regression techniques that learn a best fit threshold from the points in the latent space. However, as we prove later (See Sec. III-D), we observed that while the Cauchy loss function mitigates the impact of undetected attacks, it also leads to increased false alarms, when there was no poisoning attack (which is the most probable scenario). In contrast, Huber loss function has a lower false alarms in the event of no poisoning during training, but was poor in mitigating impact of undetected attacks caused by poisoning attacks. Thus, we need a framework that mitigates impact of data poisoning attacks without increasing the base rate false alarms (i.e, false alarms that happen if there was no poisoning).

B. Contributions

To mitigate the impact of poisoning on anomaly detection performance, we propose an integrated approach of learning the thresholds of the time series latent space by three design changes (1) Tukey biweight loss function instead of square loss function (2) adding quantile weights to regression errors of Tukey (3) modifying the definition of empirical cost function from the traditional MSE/MAE to the harmonic mean of quantile weighted Tukey losses. We also explain the role of each of three design changes in the mitigation that gives explain-ability to our methods' success compared to existing approaches. We show that this combined approach operates effectively under varying strength, scale, and duration levels of training data contamination caused by data poisoning attacks. We assess the security performance using two NIST recommended CPS metrics: (a) Undetected Attack's Impact, (b) Expected Time between False alarms. We prove that our method outperforms existing methods in both metrics and provide ablation studies to show contribution of each design change on the performance.

II. Background and Preliminaries

Although our work is generic, we show one application in the context of an anomaly based data falsification attack detector for smart metering infrastructure to bride the gap between theory and practice. We use real datasets [8] from 200 meters from a Texas Microgrid and the anomaly detection model under attack is proposed in [2].

Smart meters deployed on houses collect power consumption data, which corresponds to raw CPS sensor data. For t-th hour, and for the i-th house, the power consumption reported by the i-th smart meter is denoted as $P^i(t)$. Now, we give a background of the anomaly detection method being poisoned.

A. Anomaly Detection Problem Specification

The poisoned model under question is a start of the art time series anomaly detection method for smart living CPS proposed in a series of previous works [2, 4].

Our choice of this method is justified by the following: (1) most recent theoretical advance in time series level anomaly detection technique that can detect very low data falsification margins and also emergency events; (2) the method generalizes to different CPS/IoT domains such as and Smart Transportation Systems [5], Phasor Measurement Units [4]; (3) follows the latest National Institute of Standards and Technology (NIST) specified guidelines on CPS security [13];

For enabling a unified understanding we discuss poisoned framework in 3 phases: (1) Time Series Representation Learning that produces a time series latent space from raw time series data during training; (2) Learning of Thresholds from Latent Space; (3) Anomaly Detection Criterion. Dividing the framework in this way, will allow researchers to apply this work for other methods which might use a different time series anomaly detection metric.

B. Time Series Representation Learning

The works [2, 4], proved that the time series of harmonic to the arithmetic mean ratios from multiple smart meters (in general distributed CPS sensors [5, 4]) in a microgrid is invariant under benign conditions. Specifically, the ratio is to be calculated over strategic spatial and temporal granularity from the distributed CPS sensing devices such co-variance of data from individual meters (and in general CPS sensors) is maximized. Mathematically, the invariant is defined as:

$$Q^r(T) = \frac{\sum_{t=1}^{W} HM(t)}{\sum_{t=1}^{W} AM(t)} \tag{1}$$

where $HM(t)$ and $AM(t)$ are harmonic and arithmetic mean of power consumption $P^i(t)$ from all smart meters in a microgrid at time slot t with a box-cox transformation. The temporal granularity is a time window T of length W slots (hours).

In the case of AMI, the previous work found $W = 24$ hours and the granularity included the whole solar village of 200 houses due to its small size. Hence for a typical year, the range of $T \in \{1, 365\}$.

1) Safe Margins: Safe margins quantify an expected upper and lower bound of the anomaly detection metric at any time window T. The $\mu^{Q^r}(T)$ is defined as the cumulative weighted time average of ratios observed on the T-th window across multiple years. The σ_r denote the standard deviation of the probability distribution of $Q^r(T)$ samples in the training set. The upper and lower safe margin is a neighborhood around the expected value $\mu^{Q^r}(T)$, controlled by a scalar factor κ of the standard deviation (σ_r)- mathematically written as:

$$\Gamma_{high}(T) = \mu^{Q^r}(T) + \kappa\sigma_r \tag{2}$$

$$\Gamma_{low}(T) = \mu^{Q^r}(T) - \kappa\sigma_r \tag{3}$$

where $\Gamma_{high}(T)$ is the upper safe margin and the $\Gamma_{low}(T)$ is the lower safe margin.

2) Stateless and Stateful Residuals: Stateless residuals $\nabla(T)$ keep track of the difference between the safe margin and the ratio sample $Q^r(T)$. Mathematically, this is represented as:

$$\nabla(T) : \begin{cases} = Q^r(T) - \Gamma_{high}(T), & \text{if} \quad Q^r(T) > \Gamma_{high}(T); \\ = Q^r(T) - \Gamma_{low}(T), & \text{if} \quad Q^r(T) < \Gamma_{low}(T); \\ = 0, & \text{otherwise}; \end{cases} \tag{4}$$

The corresponding stateful residual are obtained from the stateless residuals by keeping the cumulative sum of the stateless residuals $\nabla(T)$ over a sliding window of length F. At any T, the stateful residual is given by:

$$RUC(T) = \sum_{j=T-FS}^{T} \nabla(j) \tag{5}$$

Thus, the vector of $RUC(T)$ values for a year will have 365 entries which can be either zero, positive or negative depending on the patterns on how stateless residuals evolve over time during the training set.

3) Threshold Learning from regression errors: The positive $RUC(T)$ values (denoted by the set $\boldsymbol{RUC^+(T)}$) form the training data input points from which the optimal upper threshold (τ_{max}) is learnt. Similarly, the negative $RUC(T)$ values $\boldsymbol{RUC^-(T)}$ form the input training data points from which the best lower threshold (τ_{min}) is learnt.

Let each training data point in each of the above sets be represented as $r_{i^+} \in \boldsymbol{RUC^+(T)}$ and $r_{i^-} \in \boldsymbol{RUC^-(T)}$.

The threshold learning problem is a regression problem with only the bias parameter as the unknown, because we need to fit a straight line to the sets $\boldsymbol{RUC^+(T)}$ and $\boldsymbol{RUC^-(T)}$ to find a time invariant threshold that will be used to raise a threat alarm in the deployment/test set.

In this regression problem, the unknown parameter search space (hypothesis space) are denoted by $\boldsymbol{\tau^+}$ and $\boldsymbol{\tau^-}$, for the upper and lower thresholds respectively. For example, $\boldsymbol{\tau^+}$ denote the set of all the candidate upper thresholds for learning τ_{max}. Now we discuss terminologies of regression analysis:

Regression Error s_i: The regression error is the difference between a training data point and a parameter candidate (candidate threshold)($s_i = r_i - \hat{\tau}$). For each candidate, a regression error value can be calculated for all training data points. In linear regression, positive and negative error values do not have unequal weights.

Loss Function: The loss function is a mathematical transformation that represents *how* each error value contributes to the goodness of the candidate model parameter. Therefore, w.r.t. one candidate model parameter and one training data point, there is one loss value. For example, linear regression uses squares of the regression errors ($L2$ norm) as loss function.

Empirical Risk (Cost) Function It is an equation that maps the goodness of a candidate parameter across all training data points (observations) in the training data. Typically, it is the average/mean of the loss function values over all training data points. In linear regression, the empirical risk is called *mean squared error*. The candidate parameter that has the minimum empirical risk is the optimal learning answer. Hence, this type of regression is known as ordinary least squares regression.

However, in [2] the authors incorporated unequal weighting of positive and negative regression errors into the loss function. Furthermore, it used an $L1$ norm (absolute errors) instead of an $L2$ norm (squared errors) in its definition of the empirical loss function. Algorithm 1, summarizes the learning of thresholds as proposed in [2]. In Algorithm 1, the regression error is the $\tau - RUC(T)$. Notice, that, unlike linear regression, the regression errors get unequal weights and the weighted regression errors are stored separately in $\mathbb{C}$ and $\mathbb{P}$.

Algorithm 1 Calculate τ_{max}

```
for $RUC^+(T), \tau^+$ do
    if $(RUC^+(T) < \tau^+)$ then
        $cost^+ : \frac{|\tau^+ - RUC^+(T)|}{2}$
        $\mathbb{C} \leftarrow cost^+$
    else
        $penalty^+ = |RUC^+(T) - \tau^+|2$
        $\mathbb{P} \leftarrow penalty^+$
    end if
end for
$\tau_{max} = \arg\min_{\tau^+} \frac{1}{N^+} \left| \sum_{\mathbb{C}} cost^+ - \sum_{\mathbb{P}} penalty^+ \right|$
```

4) Detection Criterion: The $RUC(T^{test})$ is the invariant in the test set. If it is violates the upper and lower thresholds, it indicates an attack.

$$RUC(T^{test}) : \begin{cases} \in [\tau_{min}, \tau_{max}] & \text{No Attack;} \\ \notin [\tau_{min}, \tau_{max}], & \text{Attack Inferred;} \end{cases} \tag{6}$$

Using the above detection criterion, [2] showed that under different attacks and faults, the $RUC(T^{test})$ metric violates the learnt thresholds.

III. Threat Model and Challenges

Poisoning attacks in this context aim to bias the learning of the latent space thresholds. This enables false data injection attacks to evade detection during testing (known as integrity violation in adversarial machine learning). False data injection from smart meters are well documented [2] and can happen due to vulnerability in hall sensors, magnetic, or optical ports or vulnerabilities in PLC communications.

Naturally, this can only happen when the attacks during training cause *widening of the true thresholds* (upper threshold increases and lower threshold decreases). Secondly, the attack goal dictates the need for two threat models; one for the training to alter the learning outputs- and another for the test set- to escape detection of attacks launched in the test set.

A. Training Phase Threat Model

Poisoning Attack Strength $\delta_{avg}^{(p)}$ is the average margin of data falsified from one CPS sensing endpoint (e.g. smart meter) during training. The $\delta_{avg}^{(p)}$ is a variable in our attack simulation to accommodate varying intents/capabilities of attackers.

Poisoning Attack Scale $\rho_{mal}^{(p)}$ is the total fraction of CPS sensing devices from which data is falsified during the training.

In our simulation, the variable $\rho_{mal}^{(p)}$ represents different attack budgets and capabilities of adversaries. Powerful adversaries who can afford higher $\rho_{mal}^{(p)}$ and vice-versa.

Poisoning Attack Type specifies the manner in which data is falsified from its original value P_t^i. The attack type determines the direction of change in the $RUC(T)$. The paper [2], reports several possibilities of data falsification in a smart metering infrastructure; including additive, deductive, camouflage, etc. Most attack types create a decrease in the ratios and hence the $RUC(T)$ samples decrease, which will cause the lower thresholds to widen significantly as shown in Sec. III-B.

Poisoning Duration: specifies the duration of the poisoning attack- which influences the number of poisoned/biased points given to the learning model. This is kept as a variable to emulate different possibilities beyond defender's control.

B. Impact of Training Phase Data Falsification Attacks

We select a deductive poisoning attack type which falsify data by reducing the original readings; where $P_t^i - \delta_t^{(p)}$, where $\delta_t^{(p)} \in [\delta_{min}, \delta_{max}]$ and the expected value of $\delta_t^{(p)}$ samples converge to the strategic training attack strength $\delta_{avg}^{(p)}$ and this falsification occurs from a $\rho_{mal}^{(p)}$ fraction of total smart meters. Study cited in [2], showed evidence that the most common attack on smart meters is the deductive attack which lead to organized electricity theft. Certain faults and network errors causes data drops [2], which behaves mathematically similar to deductive attacks, where the subset of meters show zero values. The above proves that this threat model is credible for analyzing bias caused in training anomaly detectors.

Figs. 1 shows the effect of a simple deductive attack with $\delta_{avg}^p = 200$, $\rho_{mal}^p = 0.3$ for 1 month while training a model from a live deployment of smart meters on the $RUC(T)$. Since the ratio and the $RUC(T)$ drops, more negative RUCs are triggered in the $RUC^-(T)$. This biases the learning of the τ_{min} (See). More negative RUCs trigger a lower τ_{min}. In general, for most anomaly detection metrics (including $RUC(T)$), the higher the $\rho_{mal}^{(p)}$ and $\delta_{avg}^{(p)}$, the larger the deviations in the metric and hence more serious the strength of poisoning. However, if $\delta_{avg}^{(p)}$ is very high then there can be obvious changes in the $RUC(T)$ space. Furthermore, the benign changes (see 50-100) makes threshold learning essential. Counter-intuitively, the *learnt upper threshold also increases* after the poisoning attack. This surprise is caused due to the design of the safe margins $\Gamma_{high}(T)$ and $\Gamma_{low}(T)$, which was done to lower false alarms in the original method. Hence, we need robust learning for both thresholds. Furthermore, note the clustering based approaches to detect possible points are not unified because the number of clusters that will be formed a-priori is unknown and entirely depends on the values of $\rho_{mal}^{(p)}$ and $\delta_{avg}^{(p)}$ and there could be multiple such combinations used by an adversary.

From the above, we conclude that the learning of thresholds are poisoned in both directions. This is an interesting development because a single poisoning attack type (i.e. deductive) during the training can influence both the upper threshold and lower thresholds as shown in Fig. 1. This allows for evasion of attacks during the test set.

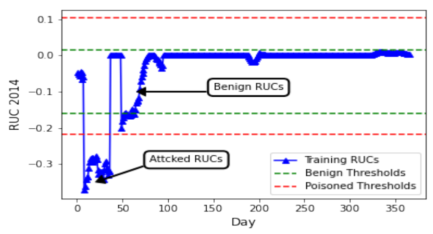


Fig. 1. Latent Space $RUC(T)$ with Benign and Poisoned

C. Test Phase Threat Model

The test set's threat model mirrors similar perturbation from smart meters because the adversary goals is to escape the detection of data falsification in the test set. We distinguish between training and test set attacks by using two terms: $\delta_{avg}^{(e)}$ (indicating attack strength for evasion) and $\rho_{mal}^{(e)}$ (indicating attack scale for evasion) in the test set, which correspond the attack that just bypasses the threshold and remains undetected.

D. Challenges of Resilient ML in Smart Living

The paper [3], proposed two robust regression algorithms (Cauchy loss, Huber loss) for learning the anomaly detection threshold to mitigate poisoning attacks. However, we observed some limitations in these works and methods which follow a similar approach. The expected time between consecutive false alarms $E_{T_{FA}}$ is a NIST recommended metric [13, 12] for evaluating time series anomaly detection in CPS. A larger $E_{T_{FA}}$ is better indicating less frequent false alarms.

One disadvantage of employing Cauchy loss is that if poisoning attacks are not present, then the false alarm performance becomes very poor. Table I, shows the $E_{T_{FA}}$ in units of days for the previous work [3] with the same dataset. We can see that compared to all other loss functions such as Huber, L1, and L2 losses, the Cauchy loss has low $E_{T_{FA}}$ even though it gives the most mitigation in the presence of poisoning. Because benign data in smart living CPS is unpredictable due to human behavioral randomness (Fig. 1), false alarm cannot be sacrificed in fear of poisoning attacks.

TABLE I
EXPECTED TIME BETWEEN FALSE ALARMS FOR PREVIOUS WORKS

Empirical Loss	$E_{T_{FA}}$ in Days
L1	364.0
L2	364.0
Cauchy	**115.5**
Huber	313.84

Note that the absence of attacks are much more likely than presence of attacks (base rate fallacy), which makes this problem even bigger. However, in cases where poisoning attacks do occur, the Cauchy loss function is most effective at mitigating the attack [3]. This highlights the first engineering challenge: we need a compromise between the mitigating the attack when present without increasing false alarm frequency- which existing works do not handle. The second engineering challenge which is not handled in [3] is it does not take into account the *poisoning duration* as a parameter into its poisoning threat model. As we show later, poisoning duration negatively affects the detection performance in test set regardless of loss function choice; and therefore require additional safeguards.

IV. Mitigation Under Poisoning Attacks

Here we give the details of our method, starting with theoretical intuition followed by the different design changes we propose resilient learning of time series anomaly detection thresholds.

M-estimation In robust statistical learning, M-estimation is the practice of replacing the well known L2 and L1 loss, with a special class of functions known as M-estimators. Formally, let regression errors for each of the i-th training data point be denoted as $s_i = r_i - \hat{\tau}$, the loss function of $\sum_i s_i^2$ is replaced by $\sum_i M(s_i)$, where $M(s_i)$ belongs to class of M-estimators [14], and the optimal estimate of τ (τ_{opt}) is:

$$\tau_{opt} = \arg\min_{\hat{\tau}} \Big[\frac{1}{N}\sum_{i=1}^{N} M(s_i, \hat{\tau})\Big] \tag{7}$$

where $\hat{\tau}$ is a candidate parameter or the hypothesis space.

Influence Function (IF) [14] corresponds to the first order partial derivative of an M-estimator function with respect to the regression errors (which is a function of the data input). Therefore by chain rule, IF quantifies the extent of variation in the estimator when the sample points during the training are perturbed. Formally, IF is defined as:

$$\psi(s_i) = \sum_i \frac{\partial(M(s_i))}{\partial s_i} \tag{8}$$

where $s_i = r_i - \hat{\tau}$ is directly affected by the perturbation of r_i. To test how sensitive the learning is to poisoning attacks, one can compare how much the loss function changes when the input data is altered during training. This is where the Influence function comes in — it helps us compare the rate of change in the influence function of various M-estimators.

Redescending M-estimators: Redescending M-estimators have influence functions $\psi(s_i)$ which are non-decreasing for lower values of regression errors s_i, but start to decrease as the regression errors increase. It satisfies the following *key property*: $\psi(s_i) = 0 \quad \forall \quad s_i, \quad |s_i| > \beta_t$, and $0 < \beta_t < \infty$ is referred to as the minimum rejection point. Formally, the $\psi_{redescending}(s_i) < \psi_{non-redescending}(s_i)$ for larger s_i, which happen due to outlying points in the inputs to the learner (the effect of data poisoning).

Among the three re-descending M-estimators (Hampel, Tukey, Andrew's sine), we hypothesize to use the Tukey-Lambda Bi-Weight estimator because of the following: Hampel's three-part function features three distinct rejection points but the poisoning scales, strengths, and durations are beyond defender's control and can take any value. Therefore, the idea of 3 distinct rejection points is in-appropriate and applicable only under abrupt short lived changes like those due to noise. In contrast, Tukey's biweight function offers a *smoother transition* between penalizing inliers and outliers. This means that while outliers receive higher penalties than inliers, the shift from less penalized inliers to more penalized outliers is gradual, unlike Hampel's function, which has less smooth transitions. Andrews' sine function [19], exhibits high efficiency only with Gaussian distribution which violate the nature of the both the raw data and latent space in smart living CPS. Considering these factors, qualitatively, we choose Tukey's biweight function as the best among the others. Next section, we give quantitative evidence of Tukey's benefit.

A. Tukey Lambda Loss as Resilient Estimator

Tukey Lambda biweight or bisquare loss function [17] for regression is defined by the following:

$$M_{Tukey}(s_i) = \begin{cases} \frac{\beta_t^2}{6}\Big(1-(1-(\frac{s_i}{\beta_t})^2)^3\Big) & \text{if } |s_i| <= \beta_t; \\ \frac{\beta_t^2}{6} & \text{otherwise.} \end{cases} \tag{9}$$

where s_i is the regression error between i-th data point and the candidate parameter (i.e., τ^+ or τ^-) and the β_t is a scaling parameter corresponding to Tukey Loss that controls the degree of descent in IF and minimum rejection point. For each $|s_i| \leq \beta_t$, the loss function is a truncated quadratic function, starting from zero at s = 0 and reaching a maximum value of $\frac{\beta_t^2}{6}$ at $s = \beta_t$. Any $s_i > \beta_t$, the constant does not change $\frac{\beta_t^2}{6}$ ensuring that $M_{Tukey}(s_i)$ remains bounded as s_i increases due to poisoning attackers. By taking the derivative of Eqn 9, the IF of Tukey loss is given by:

$$IF_{Tukey}(s_i) = \begin{cases} s_i\Big(1-\frac{s_i^2}{\beta_t^2}\Big)^2 & \text{if } |s_i| <= \beta_t; \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

Fig. 2, shows that IF_{Huber} is non-redescending, while IF_{Cauchy} is weakly redescending, while Tukey is strongly redescending. For small regression errors, $IF_{Tukey} \approx IF_{Huber}$. Additionally, for large regression errors, $IF_{Tukey} < IF_{Cauchy} < IF_{Huber}$. To conclude, while IF_{Cauchy} begins to diminish the influence of large regression errors (caused by poisoning or faults), the $IF_{Tukey} < IF_{Cauchy} < IF_{Huber}$ property makes Tukey the best estimator under poisoning attacks followed by Cauchy and Huber respectively.

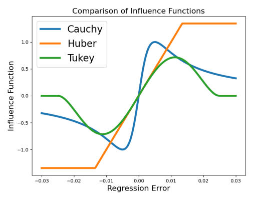


Fig. 2. Influence Function: Huber, Cauchy and Tukey

However, medium regression errors are typically caused by benign fluctuations in the latent space common in smart living applications. However, for some medium regression errors, we $IF_{Tukey} \geq IF_{Cauchy}$, which indicates that the Tukey loss allows the estimator to be influenced; which should lead to false alarm reduction in the absence of poisoning attacks. The false alarm benefit can be further enhanced by replacing s_i in Eqn. 9 with λs_i where λ is a weight (line 5 and line 12 in Algorithm 2), which we discuss in Sec. IV-C that describes how the design changes are merged.

B. Resilient Empirical Risk (Cost) Function

Note that as poisoning duration increases, the optimal estimate is going to get biased regardless of loss function choice; since loss functions only specify goodness of fit per training input point not across all training points. As the number of poisoned points increases with increasing poisoning duration, small drifts in losses incurred per point (even for the most robust loss functions) add up, creating an inevitable change in the optimal estimate.

To enhance resilience against longer duration poisoning attacks, we propose a modification: using the harmonic mean of quantile Tukey losses instead of the usual mean of losses across all training points. Formally, the new ERF is:

$$ERF_{HM} = \Big(\frac{N}{\sum_{RUC(T)} \frac{1}{L_s}} \Big) \tag{11}$$

where L_s denotes the quantile Tukey loss incurred per training point (this appears in Line-18 in Algorithm 2), N is the number of training input points in the latent space. For example, N^- represents cardinality of the $RUC^-(T)$ and the corresponding loss are denoted as L_{s^-}. The process of learning is identical for the τ_{max} and hence we do not re-write Algorithm 2 separately for learning τ_{max}.

Now we explain why this modification will improve mitigation under increasing poisoning duration. Note that, arithmetic mean of the loss function values:

$$ERF_{AM} = \Big(\frac{1}{N} \sum_{RUC(T)} L_s \Big) \tag{12}$$

is a Schur convex function (increases exponentially). In contrast, the harmonic mean of loss function values (i.e. Eqn. 11 is a Schur concave function (increases approximately logarithmically) [18]. Since Schur concave function grows at a lesser rate with additive drifts (caused by larger loss values of outlying points) for the same loss function $ERF_{HM} < ERF_{AM}$. This results in lower rate of change in the ERF for each parameter choice τ in Eqn. 11. Hence, as poisoning duration increases, the harmonic mean of quantile Tukey losses must be better.

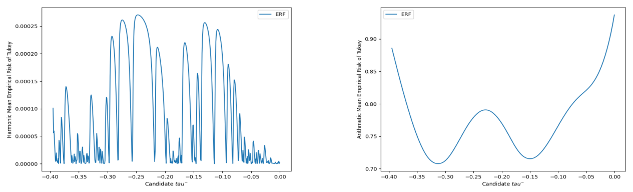
Note, due to Schur concave nature of our modified empirical risk, our objective changes from seeking the typical global minima to identifying the global maxima (line 18 in Algorithm 2); hence we need to replace the last line from the typical argmin to argmax, to find the optimal estimate.

Trade-offs: While using Eqn. 11 as ERF in Algorithm 2 offers robustness, the Eqn. 11 is not smooth and differentiable (See Fig. 3(a)), making efficient gradient descent/ascent technique not applicable. Hence, one needs to either use simulated annealing or a brute force approach to find the maxima.

In contrast, the usual arithmetic mean of losses (See Fig. 3(b)) as an empirical risk function, gradient descent applies well. To conclude, the mitigation comes at the cost of performance. However, for this framework there is only one parameter to learn, the search space is over a reduced latent space, and parameter needs to be learnt only once. Hence, a brute force approach to find the exact global maxima in Algorithm 2 is implemented. In future, we will implement simulated annealing to report any performance difference.

C. Final Learning Algorithm with Quantile Weighted Tukey

Here we merge the tukey biweight, quantile weighted regression errors and robust ERF_{HM} into one (Algorithm 2). We introduce two weights (i.e., λ_{high} and λ_{low}) to the s_i before it is passed to the Tukey loss, such that $\lambda_{high} > \lambda_{low}$. This can be verified from line 5 and line 12 in Algorithm 2. The Lines 2-9, tackle the outer points, while 10-16 tackle inner points relative a candidate choice τ^-. The λ_{low} weight corresponds to regression errors of inner points relative to τ^-, and λ_{high} corresponds to regression errors of outer points. This further adds to the balance between false alarms and limiting impacts of poisoning attacks. Finally, line 18 implements our ERF_{HM} in Eqn. 11. Note however, that for learning of the upper threshold τ^{hmtu}_{max}, the definition of outer and inner points get switched due to the sign change in $RUC^+(T)$. We need to be careful about how the quantile weights are described compared to previous works on this problem [3]. Therefore, for $RUC^+ < \tau_+$, the line 5, of Algorithm 2 will be replaced by λ_{low} instead, and the line 12 will be replaced by λ_{high}, unlike Algorithm 2.



(a) (b)

Fig. 3. Empirical Risk with Quantile Tukey: (a) ERF_{HM} (b) ERF_{AM}. Both figures correspond to $\rho^{(p)}_{mal} = 0.40$, $\delta^{(p)}_{avg} = 200$, poisoning duration 6 months, poisoning attack type deductive, hypothesis space is τ^-. Note that the maxima of ERF_{HM} is achieved for $\tau^- = -0.23$, while the minima of traditional ERF_{AM} is achieved at $\tau^- = -0.32$. This proves that ERF_{AM} is less robust due more negative threshold under attacks

Algorithm 2 Empirical Risk Function as Harmonic Mean of Quantile Tukey losses($\tau^{(hmtu)}_{min}$)

```
1:  for $RUC^-(T), \tau^-$ do
2:      if $(RUC^-(T) < \tau^-)$ then
3:          $S^- : |\tau^- - RUC^-(T)|$
4:          if $(S^- \leq \beta_t)$ then
5:              $L_{s^-} : \frac{\beta_t^2}{6}\Big[1 - \Big[1 - \Big(\frac{s^- \lambda_{\mathbf{high}}}{\beta_t}\Big)^2\Big]^3\Big]$
6:          else
7:              $L_{s^-} : \frac{\beta_t^2}{6}$
8:          end if
9:      else
10:         $S^- : |RUC^-(T) - \tau^-|$
11:         if $(S^- \leq \beta_t)$ then
12:             $L_{s^-} : \frac{\beta_t^2}{6}\Big[1 - \Big[1 - \Big(\frac{s^- \lambda_{\mathbf{low}}}{\beta_t}\Big)^2\Big]^3\Big]$
13:         else
14:             $L_{s^-} : \frac{\beta_t^2}{6}$
15:         end if
16:     end if
17: end for
18: $\tau^{(hmtu)}_{min} = \arg\max_{\tau^-} \Big( \frac{N^-}{\sum_{RUC^-(T)} \frac{1}{L_{s^-}}} \Big)$
```

D. Learning Hyperparameters

Now we describe the hyperparameters selection β_t, λ_{low} λ_{high} over a cross validation dataset. We split the dataset from the 2016 dataset into partition of four month of cross validation. The rest of the 8 months of 2016 was set aside for testing. Among the 4 months, we have 2 months of benign and 2 months of various attacks. Hence, we have different pairs of benign and attack realizations in the cross validation set. We jointly measure the false alarms and the missed detection over the benign and attack portions respectively, by trying different values of the hyper-parameters. We pick the hyperparameters according to the following:

$$\arg\min_{\beta_{t^+}, \beta_{t^-}} (w_{md} MD + w_{fa} FA) \tag{13}$$

We maintained a fixed weight assignment of $w_{fa} = 0.7$ and $w_{md} = 0.3$. This choice is substantiated that the reduction of false alarms at the very least, twice as important.

The β_t^+ and β_t^- is 0.0009, $\lambda_{high} = 0.8$ and $\lambda_{low} = 0.2$. The hyperparameters, β_c, β_h, in Cauchy and Huber Losses were evaluated in [3], we adopt those values directly for comparison with our work.

E. Security Evaluation Metrics

Here we elaborate the two main security evaluation metrics. ***Impact of Undetected Attack:*** This metric quantifies mitigation by measuring the impact of an undetected attack on the utility after implementing both our attack and defense strategies [13]. To determine the daily impact of an undetected attack, we quantify the lost revenue in relation to the unit price of electricity, denoted as RR. This calculation is expressed as:

$$RR = (\delta_{avg} * M * \eta * E)/1000 \tag{14}$$

here, $\eta = 24$ represents the number of daily reports, $E =$ \$0.12 stands for the average cost of electricity per unit (KW-Hour) in the USA, δ_{avg} signifies the margin of false data in test set, M denotes the number of compromised smart meters in the test set. The impact of an undetected attack is:

$$Impact = RR * T_{detect} \tag{15}$$

where T_{detect} refers to the time duration in days between the start and end of the undetected attack within the test set. A smaller impact signifies superior performance.

For mitigation performance, we report the I in the test set in the presence of optimal evasion attacks measured over a 90-day period. Optimal evasion attack is a data falsification that just evades the poisoned threshold and hence is the stealthiest possible attack. *The lesser the I, the better is the mitigation.* ***Expected Time between Consecutive False Alarms:*** The expected time between two false alarms, denoted as $E_{T_{FA}}$ is a metric recommended by recent NIST [13] recommended metric for time series anomaly detection:

$$E_{T_{FA}} = \frac{\sum_{\eta_{FA}} T_{BFA}}{\eta_{FA} - 1} \tag{16}$$

where η_{FA} represents the number of false alarms and the T_{BFA} is the time interval between any two consecutive false alarms. In the case of a single false alarm, this equation does not apply, and instead, $E_{T_{FA}}$ is 364, reflecting the expectation of another false alarm occurring within a year. Note, *the higher the $E_{T_{fa}}$ the better it is*, (i.e. less frequent false alarms).

For false alarm performance, we report $E_{T_{fa}}$ for 9 months of 2016 (i.e., the test set), when there no poisoning attacks during the training set and no evasion attacks in the test set.

V. Experimental Results

We validate our work using the data collected from the Pecan Street Project with 200 houses [8] from a solar micro-grid. Amongst the different datasets available, the texas dataset shows most variations and data errors [2], which motivated us to pick this dataset. Unlike in [2, 3], we *did not do any data cleaning for this paper*, so we can test mitigation of both attacks and any gross data errors. Further, experimental details on extraction of $RUC(T)$ can be found in [2].

The smart meter data are partitioned into three segments for training, cross-validation, and testing purposes. The training dataset comprises all records from 2014 and 2015, following the approach outlined in prior research [3] to ensure fair performance evaluation. The remaining records from 2016 are split into two subsets for cross-validation and testing. For cross-validation, all records from the first three months of 2016 are utilized, while the data from the subsequent nine months of year 2016, constitute the testing dataset. The learned hyperparameters for proposed method are $\beta_t^+ = 0.0009$ and $\beta_t^- = 0.0009$. To extend poisoning duration, we introduce variations in the duration of deductive attack types in the training dataset, categorizing them into three distinct periods: 2 months (8.33% poisoned samples), 4 months, and 6 months (25% poisoned samples). We also vary the poisoning strength per smart meter $\delta_{avg}^{(p)}$ from $50W$ to $400W$ (above 400W most methods can detect attacks) and the poisoning scale $\rho_{mal^{(p)}}$ varying from 20% to 70% of the meters compromised. We report performance averaged over all possibilites.

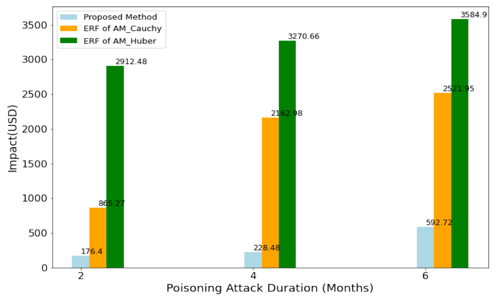


Fig. 4. Average Impact of undetected attacks over Varying Poisoning Durations Averaged over all $(\delta_{avg}^{(p)}, \rho_{mal}^{(p)})$ pairs: Proposed vs. Previous Works

A. Impact Mitigation Performance Under Poisoning

The Fig.4 illustrates the average impact of undetected attack in the test set, (averaged over all $\delta_{avg}^{(p)}$ and $\rho_{mal}^{(p)}$) of our proposed approach; as compared to existing works that use quantile weighted versions of cauchy and huber losses and the ERF is arithmetic mean of such losses. Note that attack is undetected means we implemented an attack that will not exceed the learnt threshold - assumes adversary with complete knowledge. From Fig. 4 we can easily infer that proposed method has the lowest impact of undetected attack regardless of the increasing poisoning duration.

B. Base Rate False Alarm Performance

Table II, gives the expected time between false alarms for different design choices in the event that *there are no poisoning attacks in the training; no data falsification in the test set*; and utility uses our design as compared to others. We see that our proposed approach gives $E_{T_{fa}}$ of 241 days. In contrast, we see that regular Tukey *that does not use quantile regression errors* give a poorer false alarm performance with 201.34 days between any two false alarms. This shows specifically the *benefit of incorporating quantile regression errors* rather than regular errors. Additionally, the quantile Cauchy is the worst giving on average 116.52 days gap between two false alarms, and quantile Huber gives 233.65 days which is the worse than our proposed approach - but huber is least robust under poisoning. This proves that our method mitigates impact of undetected attack without sacrificing base rate false alarm performance. Note the difference in the numbers compared to [3] is due to the use of uncleaned data in our paper.

TABLE II
BASE RATE EXPECTED TIME BETWEEN FALSE ALARMS: COMPARISON

Empirical Loss	$E_{T_{fa}}$
Harmonic Mean of Quantile Tukey Loss - **Proposed**	241.0
Arithmetic Mean of Quantile Tukey Loss	232.16
Arithmetic Mean of Regular Tukey Losses [17]	201.34
Arithmetic Mean of Quantile Cauchy Loss [3]	116.52
Arithmetic Mean of Quantile L_1 Loss [3]	239.01
Arithmetic Mean of Quantile Huber Loss [3]	233.65

C. Mitigation Sensitivity Analysis with Ablation Studies

Here, we show how mitigation performance is affected by varying poisoning strength and scales keeping the same poisoning duration of 4 months. Fig. 5(a) shows impact of three different loss function for each poisoning strength value, (averaged over all corresponding $\rho_{mal}^{(p)}$ varying between 20% and 70%). As the poisoning strength increases, Cauchy's and Huber's performance in terms of undetected attack impact increases while for the proposed approach. This is true for Fig. 5(b) which shows sensitivity over varying $\rho_{mal}^{(p)}$ (each averaged over all possible $\delta_{avg}^{(p)}$).

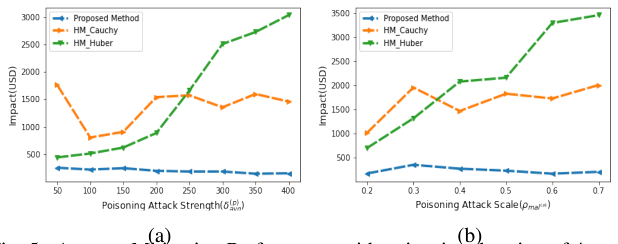


Fig. 5. Average Mitigation Performance with poisoning duration of 4 months: (a) varying poisoning strength (averaged over all possible $\rho_{mal}^{(p)}$); (b) varying poisoning scale $\rho_{mal}^{(p)}$ (averaged over all possible $\delta_{avg}^{(p)}$). **The Figures are generated by using proposed ERF_{HM} same for varying loss functions to show the individual benefit of loss function choice.**

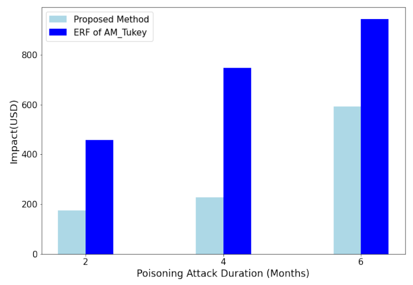


Fig. 6. Average Impact Mitigation over varying poisoning durations. **generated by keeping the same loss function (i.e. Tukey) to show the individual benefit of the proposed ERF_{HM} into our design**

Fig. 6 shows only the difference made by the difficult to optimize empirical risk function, while keeping everything else in Algorithm 2 same. We observe that our modified empirical risk (light blue bars) gives a much lower impact of undetected attack, compared to the use of arithmetic mean of quantile weighted Tukey losses (dark blue bars). This proves that the performance trade-off for using our proposed robust empirical risk ERF_{HM} is worth it; as the poisoning duration increases, the benefit of ERF_{HM} over ERF_{AM} become more apparent.

D. Conclusion

In this paper, we enhanced the robustness of anomaly based attack detection methods in smart living CPS against training set (simple poisoning) attacks of various scales, strengths, and durations. Our framework contained three design changes compared to the usual practice of learning the thresholds - (i) quantile regression errors, (ii) Tukey Bi-weight as loss function, (iii) harmonic mean of Tukey loss as empirical risk function. We found that combining the three proposed design changes in the learning the threshold of time series anomaly detectors, it is possible to mitigate both impact of poisoning attacks as well as base rate false alarms (in the absence of poisoning) without a priori assumptions on attack capabilities. Furthermore, also offered explanations of why our design choices improved performance and also did not make any assumptions about optimal attacks.

Acknowledgements: Research was supported by US National Science Foundation CNS-2030611.

REFERENCES

[1] A. E. Cinà, K. Grosse, A. Demontis, B. Biggio, F. Roli, M. Pelillo, "Machine Learning Security against Data Poisoning: Are We There Yet?", arXiv preprint arXiv:2204.05986,2022.
[2] S. Bhattacharjee, S.K. Das, "Detection and Forensics against Stealthy Data Falsification in Smart Metering Infrastructure", *IEEE Trans. of Dependable and Secure Computing*, Vol. 18, Jan. 2021.
[3] S., Bhattacharjee, M. J., Islam, S., Abedzadeh. "Robust anomaly based attack detection in smart grids under data poisoning attacks", *ACM AsiaCCS Workshop on Cyber-Physical System Security*, pp. 3-14. May. 2022.
[4] P. Roy, S. Bhattacharjee, S. Abedzadeh and S. K. Das, "Noise Resilient Learning for Attack Detection in Smart Grid PMU Infrastructure," *IEEE Transactions on Dependable and Secure Computing*, 2022.
[5] J. Islam et al., "Anomaly based Incident Detection in Large Scale Smart Transportation Systems," 2022 ACM/IEEE 13th International Conference on Cyber-Physical Systems (ICCPS), 2022, pp. 215-224.
[6] J. Steinhardt, P. W. Koh, P. Liang. "Certified Defenses for Data Poisoning Attacks", . Advances in neural information processing systems, 30, 2017
[7] M. A. Ramirez , S.K. Kim , H. A. Hamadi , E. Damiani , Young-Ji Byon , T. Y. Kim , C. S. Cho and C. Y. Yeun, "Poisoning attacks and defenses on artificial intelligence: A survey.", arXiv preprint arXiv:2202.10276, 2022
[8] [Online] Pecan Street Project Dataset https://www.pecanstreet.org/
[9] G. Shevlyakova, S. Morgenthalerb, A. Shurygin., "Redescending M-estimators", *Journal of Statistical Planning and Inference*, pp.2906-2917. No. 10, 2008.
[10] D. M. Khan, M. Ali, Z. Ahmad, S. Manzoor, and S. Hussain., "A New Efficient Redescending M-Estimator for Robust Fitting of Linear Regression Models in the Presence of Outliers", Mathematical Problems in Engineering, pp. 1-11, 2021
[11] P. Roy, S. Bhattacharjee, S.K. Das, "Real Time Stream Mining based Attack Detection in Distribution Level PMUs", *IEEE GLOBECOM*, Dec. 2020.
[12] D. Urbina, J. Giraldo, A. A. Cardenas, N. Tippenhauer, J. Valente, M. Faisal, R. Candell, H. Sandberg, "Limiting the Impact of Stealthy Attacks on Industrial Control Systems" *ACM CCS*, 2016.
[13] D. Urbina, J. Giraldo, A. A. Cardenas, N. Tippenhauer, J. Valente, M. Faisal, J. Ruths, R. Candell, H. Sandberg, "Survey and New Directions for Physics-Based Attack Detection in Cyber-Physical Systems", *ACM Computing Surveys*, Vol. 51(10), 2018.
[14] P. Huber, E. Ronchetti, "Robust Statistics", *Wiley Probability and Statistics*, ISBN 978-0470129906, Feb. 2009.
[15] A. Ghafouri, Y. Vorobeychik, X. Koutsoukos, "Adversarial Regression for Detecting Attacks in Cyber-Physical Systems" *IJCAI*, 2018.
[16] X. Li, Q. Lu, Y. Dong and D. Tao, "Robust Subspace Clustering by Cauchy Loss Function," *IEEE Transactions on Neural Networks and Learning Systems*, Vol. 30(7), pp. 2067-2078, 2019.
[17] S. Beatriz, S. Aelst. "Advantages of M-estimators of location for fuzzy numbers based on Tukey's biweight loss function." Intl. Journal of Approximate Reasoning. Vol.93. pp.219-237. 2018.
[18] P. S. Bullen: Handbook of Means and Their Inequalities. Dordrecht, Netherlands: Kluwer, 2003, pp. 175-177.
[19] D. F. Andrews: A robust method for multiple linear regression. Technometrics, 16(4), pp.523-531