Revitalizing Self-Organizing Map: Anomaly Detection Using Forecasting Error Patterns

Young Geun Kim, Jeong-Han Yun, Siho Han, Hyoung Chun Kim, Simon S. Woo

Revitalizing Self-Organizing Map: Anomaly Detection using Forecasting Error Patterns

Young Geun Kim[1], Jeong-Han Yun[3], Siho Han[2], Hyoung Chun Kim[3], and Simon S. Woo[2]

[1] Department of Statistics, Sungkyunkwan University, South Korea
[2] Department of Applied Data Science, College of Computing and Informatics, Sungkyunkwan University, South Korea
[3] The Affiliated Institute of ETRI, South Korea

Abstract. Detecting rare cases of anomalies in Cyber-Physical Systems (CPSs) is an extremely challenging task. It is especially difficult to accurately model various instances of CPS measurements due to the dearth of anomaly samples and the subtlety of how their patterns appear. Moreover, the detection performance may be severely limited owing to mediocre or inaccurate forecasting by the underlying prediction models. In this work, we focus on improving the anomaly detection performance by leveraging the forecasting error patterns generated from prediction models, such as Sequence-to-Sequence (seq2seq), Mixture Density Networks (MDNs), and Recurrent Neural Networks (RNNs). To this end, we introduce Self-Organizing Map-based Anomaly Detector (SOMAD), an anomaly detection framework based on a novel test statistic, *SomAnomaly*, for Cyber-Physical System (CPS) security. Upon evaluation on two popular CPS datasets, we demonstrate that SOMAD outperforms baseline approaches through online multiple testing, using Time-Series Aware Precision and Recall (TaPR) metrics. Accordingly, we empirically demonstrate that forecasting error patterns of raw CPS data can be useful when detecting anomalies through a fast, statistical multiple testing approach such as ours.

Keywords: Anomaly Detection · Self-Organizing Map · CPS

1 Introduction

Cyber-Physical Systems (CPSs) are susceptible to various types of anomalies, such as attacks on controllers, networks, or cyber-physical elements, as well as hardware failures, operator errors, and software misconfigurations, which can cause different types of anomalies. Accurately distinguishing the latter kinds of anomalies, which may simply be glitches, from actual anomalies is an important, yet challenging task that may lead to high false positives. Such errors are costly, especially for CPSs, since the main infrastructures might have to stop running for inspection purposes. In particular, the detection of anomalous instances in water pumps, power grids, or nuclear power plants [19], is crucial for the prevention

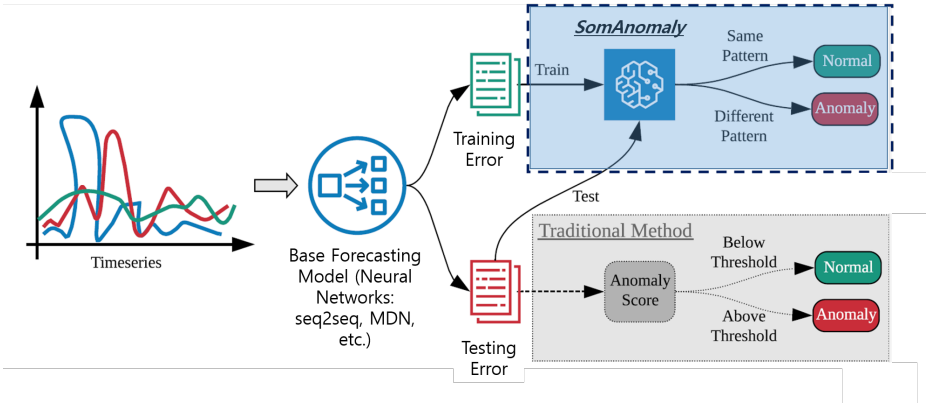


Fig. 1: SOMAD (area colored in blue) vs. traditional anomaly detection approach (area colored in gray). SOMAD leverages FEs to detect anomalies, while the traditional approach directly detects anomalies based on the output of base forecasting models, such as seq2seq and MDNs.

of huge economic losses as well as environmental catastrophes. The first step of anomaly detection [12,8,20] typically involves building a rule-based or statistical machine learning-based (e.g., neural networks) forecasting model from the given training data as shown in Fig. 1. Next, the anomaly score is computed from the forecasting error (FE), and an observation is considered anomalous if that score exceeds the out-of-limit (OOL) threshold. Similarly, statistical machine learning-based approaches typically fit a model to the data and consider an observation as anomalous if the prediction error is exceeds a predefined threshold. Meanwhile, there are several challenges related to the detection of anomalies in CPSs. For example, in many practical situations, there exist normal instances of which the measurements exceed the threshold and contextual anomalies of which the measurements do not exceed the threshold; the former should not be considered anomalous, while the latter should. Another challenge is that it is realistically difficult to collect anomalous data beforehand, which complicates the process of building an accurate anomaly predictor trained only on data corresponding to normal instances. Moreover, an attacker can perform sophisticated manipulations on the FEs, such that they are observed to be less than the pre-defined threshold, leading to a high false positive rate. Accurate forecasting is usually attainable for models trained and optimized on abundant data of both normal and abnormal classes, but this is rarely the case in real-world scenarios regarding CPSs, since only a few cases of anomalies are readily available. Therefore, improving the anomaly detection performance by simply optimizing conventional forecasting models themselves has its limitations.

In this work, we propose Self-Organizing Map-based Anomaly Detector (SOMAD), an anomaly detection framework based on a novel test statistic, *SomAnomaly*, to detect anomalies in CPS sensor and actuator data based on their

FE patterns. The overview of our approach is depicted in Fig. 2. The main objective of our approach is to amplify the differences in the respective FE patterns of normal and anomalous events, with SOM [21] used to learn and characterize the FE patterns of normal data. We construct 3D tensors and discretize the patterns into a small number of grids (SOM grids) to train only on the normal FE patterns to experiment under realistic scenarios where anomalies are extremely scarce. We conduct hypothesis testing based on *SomAnomaly* to identify anomalies using our online, multiple testing algorithm. Note that the output of SOM is codebook matrices or a collection of normal error patterns from which we can measure the distance with the input test error patterns. For our experiments, we use two benchmark CPS datasets, Secure Water Treatment (SWaT) [13] and HIL-based Augmented ICS (HAI) Security [29], We configure the training and test sets such that the former only contains data corresponding to normal observations to reproduce a challenging, realistic anomaly detection scenario, where unseen anomalous events are encountered only during the test phase. During evaluation, we assess our model based on contextual anomalies, i.e., anomalies that do not exceed a pre-defined threshold, using the latest time series-aware precision and recall metrics [17], since conventional metrics fail to serve as an accurate evaluation method for anomaly detection in time series [17]. and demonstrate that our proposed approach significantly outperforms baseline methods. Our contributions are summarized as follows:

1. We propose a novel SOM-based anomaly detection framework and publicly release our code for reproducibility[4].
2. We demonstrate that our proposed method can effectively detect unseen anomalies in high-dimensional CPS data through an online, multiple hypothesis testing algorithm under a realistic scenario where anomalous cases are extremely rare.
3. We conduct experiments on benchmark CPS datasets – SWaT [13] and HAI [29] – and achieve on average 36% increase in the time series-aware F_1 score compared to those of baseline approaches.

2 Related Work

Anomaly detection in multivariate time series data is a challenging task, and numerous approaches have been proposed in the past few years to tackle this problem. When identifying anomalies in Cyber-Physical Systems (CPS), the first-order approach can be implemented by building a knowledge base, when comprehensive and accurate domain knowledge is available [28,23,24,26]. However, in modern CPS, developing a knowledge base from a large number of variables in a complex CPS is challenging. Recently, data-driven approaches such as deep-learning or unsupervised clustering-based methods, which do not require broad and specific knowledge of the domain, have been developed [18,16,6,15,11]. These methods can learn and derive information and patterns from data, not

[4] `https://github.com/ygeunkim/somanomaly`

using much expert domain knowledge. However, the development of a thorough knowledge base from complex CPS data in the modern era is challenging, requiring extensive domain knowledge. To cope with this limitation, recent approaches include data-driven deep learning-based unsupervised clustering methods, which can automatically extract relevant information and derive characteristic patterns from the data [18,16,6]. Another popular approach is to use OOL thresholds. Given some time series data, the p-norm is often used to calculate the anomaly score. Note that when $p = 1$, it represents the sum of the absolute values. [9,8] used the 2-norm, mean-squared error (MSE), while [20] used the 4-norm. For convenience, we denote the method using the p-norm criterion as static thresholding, which is the simplest, yet one of the most popular methods. Also, the cumulative sum (CUSUM) method [14] is widely used; it divides a time series into fixed time window intervals and computes the sum of the p-norms for each window. The resulting sum serves as the anomaly score for each window: if the sum is larger than the threshold, the window is considered anomalous. We use static thresholding with $p = 4$ and CUSUM as our baselines for performance comparisons.

On the other hand, Aloudi et al. [2] present PASAD, an efficient model-free Process-Aware Stealthy-Attack Detection mechanism that monitors sensors and raises an alarm in the occurrence of a structural change in the behavior of physical processes. The authors conduct Singular Spectrum Analysis (SSA), a non-parametric spectral estimation approach, to separate the deterministic part of a dynamical system from its chaotic part. However, in this work, we focus on forecasting error based approach unlike directly exploiting the sensor values. Recently, also neural networks have shown their effectiveness in time series modeling and anomaly detection. Malhotra et al. [25] proposed a Long Short-Term Memory (LSTM) Network with an Encoder-Decoder scheme, which detects anomalies based on reconstruction errors between the input and its reconstructed output; with the latter approach, the authors were able to achieve better generalization compared to when using a simple distance-based approach. Zhai et al. [31] developed Deep Structured Energy-Based Models (DSEBMs), which connect Energy-Based Models (EBMs) to a regularized autoencoder to eliminate the need for complicated sampling. The latter model uses energy scores as well as reconstruction errors to detect anomalies. However, the aforementioned methods cannot jointly analyze temporal dependencies, noise resistance, and the severity of anomalies [31]. Also, Zhang et al. [32] developed Multi-Scale Convolutional Recurrent Endcoder Decoder (MSCRED), which constructs multi-scale signature matrices to characterize multiple levels of the system status over the time. However, most deep learning-based methods resort to some sort of thresholding, such as OOL, to distinguish anomalies from normal instances.

Also, Sequence-to-Sequence (seq2seq), Mixture Density Networks (MDNs), and Recurrent Neural Networks (RNNs) [27] have also been popular choices. Typically, a seq2seq model is composed of an encoder and a decoder, where the task of the former is to learn from the input data and create a fixed feature dimension while that of the latter is to decode it to reconstruct the input for

further use cases, e.g., anomaly detection. Bishop [5] proposed MDNs, which can rapidly learn from a training set and produce a probability distribution function for the expected signal as a function of time, by combining a conventional neural network with a Mixture Density model. Lastly, RNNs have also been use for time series-related tasks. In this work, we use these three models as our underlying anomaly detection classifier to generate forecasting error patterns from the SWaT and HAI datasets.

3 Mathematical Preliminaries for Self-Organizing Maps

In this section, we present the mathematical background and preliminary information regarding SOM, as well as the Lindeberg-Feller theorem, necessary for the comprehension of our proposed method.

3.1 Self-Organizing Maps (SOM)

Kohonen [21] developed a novel Artificial Neural Network (ANN) structure called SOM, which maps observations to topological maps with finite number of prototypes called Kohonen neurons (SOM grids). Each grid has its own vector called the codebook in the input space and SOM updates it for each training sample to approximate the training data. In particular, the codebook matrices are the output of SOM, which are a collection of the normal error patterns in our case. We use the distance between the codebook matrices and the corresponding test error patterns of a window to predict anomalies as shown in Fig. 2. Another advantage of SOM is that it does not use a back-propagation algorithm, but implements competitive learning to compute the distance between the observation and the codebook matrices and update the relevant codebook vectors for each training sample (additional details are provided in the following paragraphs).

We extend Kohonen's SOM algorithm [21] by proposing a novel test statistic called *SomAnomaly*, such that we can investigate the similarity in the input patterns to effectively determine anomalies. First, we formally describe the original SOM process based on Kohonen's research [21]. We consider an $n \times p$ array, where n is the number of observations and p is the number of predictors. Then, the observational unit is a p-dimensional vector, defined as the input vector, which can be written as follows: $x_i = (\xi_{i1}, \xi_{i2}, \ldots, \xi_{ip})^T \in \mathbb{R}^p,\ i = 1, \ldots, n$.

For SOM, we have the following hyperparameters: $\alpha(0)$ (initial learning rate), $\sigma(0)$ (initial radius), N_x (number of SOM grid in the x-dimension), N_y (number of SOM grid in the y-dimension), h (neighborhood function), and g (decay function). Let $N = N_x N_y$ be the total number of nodes, then in each node, a codebook vector with the same dimension as the input vector is assigned. The codebook vector if defined as follows: $m_i = (m_{i1}, m_{i2}, \ldots, m_{ip})^T \in \mathbb{R}^p,\ i = 1, \ldots, N$.

Next, we define each $m_{ij}, j = 1, \ldots, p$, as the weight in a codebook vector. For each iteration, the learning algorithm randomly chooses one input vector and its closest codebook vector, defined as the Best Matching Unit (BMU). Then, the neighboring node of BMU whose distance to the cluster, known as

the prototype distance, is less than the radius is updated as follows: $m_j(t+1) = m_j(t) + \alpha(t)h(r_c - r_j)[x(t) - m_j(t)]$. Following the update, SOM maps each input vector to two-dimensional nodes or neurons. By searching for its BMU, every input vector finds the closest codebook vector, which is then sent to its corresponding node.

3.2 Lindeberg-Feller Theorem for Setting Threshold

A number of anomaly detection methods require setting a threshold to detect anomalies. The limitation, however, is that most approaches select a threshold empirically without strong mathematical or statistical foundations. To that end, we apply the Central Limit Theorem (CLT) to provide a statistical foundation for determining the threshold set for our approach. For our proposed approach, we employ a generalization of the CLT called Lindeberg-Feller theorem [22,7] for non-identically distributed cases in multiple statistical testing. First, let us define a triangular array of random variables $\{X_{nj}\}_{j=1}^{n}$ for $n = 1, 2, \ldots$ for simplicity. Also, let us assume that X_{nj} is independent for each n, such that $E[X_{nj}] = 0$ and $Var[X_{nj}] = \sigma_{nj}^2 < \infty$. Then, let $Z_n = \sum_{j=1}^{n} X_{nj}$ and $B_n^2 = \sum_{j=1}^{n} \sigma_{nj}^2$. Lastly, for every $\epsilon > 0$, the Lindeberg-Feller theorem can be stated as follows:

$$\frac{1}{B_n^2} \sum_{j=1}^{n} E\left[X_{nj}^2 I\left(|X_{nj}| \geq \epsilon B_n\right)\right] \to 0, \tag{1}$$

as $n \to \infty$, then

$$\frac{Z_n}{B_n} \xrightarrow{\mathcal{D}} \mathcal{N}(0,1). \tag{2}$$

Eq. 1 is called the Lindeberg condition. We employ Lindeberg-Feller theorem to statistically determine an optimal threshold for anomaly detection.

4 Our Approach

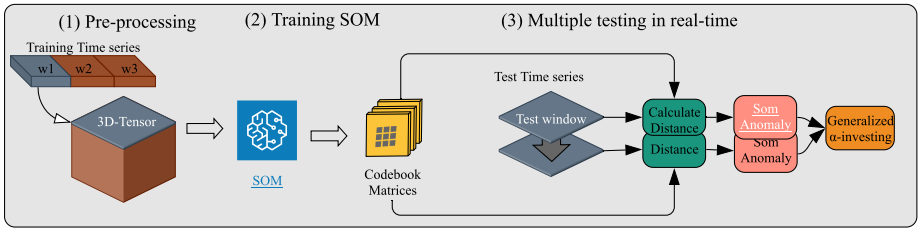


Fig. 2: Overall pipeline of SOMAD based on the *SomAnomaly* statistic

The overall process of our approach, SOMAD, is presented in Fig. 2. To characterize the error patterns of normal data, we first construct 3D tensors and discretize the normal patterns. Then, we use SOM to train only on normal FE patterns and conduct a hypothesis testing to measure the distance between the input and normal error patterns. Lastly, we identify anomalies using the *SomAnomaly* statistic and an online multiple hypothesis testing algorithm. We explain the details of each step in the following paragraphs.

1. Pre-processing. We consider p-dimensional time series with sample size n to represent multivariate time series by converting the data into a 3D tensor, as shown in Fig. 2. When training the SOM, we treat each window as a single observation. More specifically, the pre-processing step to obtain a 3D tensor can be summarized as follows:

1. Slide the window of size w with a shift size s.
2. Combine the windows into a 3D tensor of size $m \times w \times p$, where $m = \frac{n-w}{s} + 1$.

2. Training the SOM. Following its conversion to 3D tensor, the error pattern data is changed from a vector to a matrix. For matrix computation, we consider the Frobenius norm and use the following distance function between A and $B = (\beta_{jk}) \in \mathbb{R}^{w \times p}$ to determine the discrepancy of FE patterns between normal and anomalous data:

$$d(A, B) = \left(\sum_{j=1}^{w} \sum_{k=1}^{p} (\alpha_{jk} - \beta_{jk})^2 \right)^{\frac{1}{2}}. \tag{3}$$

To train the SOM, we propose an incremental SOM training algorithm, shown in Algorithm 1, using the distance function in Eq. 3. This training process maps each normal pattern window onto the SOM grids by finding the closest corresponding codebook matrix. Since the number of grids is finite, we can assume that the pattern is discretized, and every training error window maps onto finite prototypes, each of which has its own codebook matrix. This means that normal error patterns are discretized by the patterns represented by the codebook matrices.

3. SomAnomaly for Hypothesis Testing. From the output of SOM, i.e., the collection of normal error patterns defined as codebook matrics, the input test error patterns of windows that deviate significantly from the codebook matrices are considered anomalous. Accordingly, we use the distances between codebook matrices and the test error patterns to determine anomalies, as shown in Fig. 2. In order to facilitate the threshold selection, we aim to choose the maximum distance through hypothesis testing.

Hypothesis Testing and *SomAnomaly* statistic. We construct a window whenever a new set of samples of size w is available (*streaming windows*). Let $D_{ti},\ t = 1, \ldots,\ i = 1, \ldots, N$, be the distance between the t-th streaming window and the lastly updated codebook matrix of i-th node, where N is the number of Kohonen neurons, SOM grids, and each D_{ti} is a random variable. We assume that $\{D_{ti}\}_{i=1}^{N}$ are mutually independent for each t. In order to detect

Algorithm 1: Incremental SOM training algorithm using 3D tensor

Data: 3D tensor $[X_1, \ldots, X_m] \in \mathbb{R}^{m \times w \times p}$
Input: SOM parameters
Initialize learning rate and radius
Initialize codebook matrices
Compute the distance $r_c - r_i$ between nodes c and i in the SOM space;
for $j \leftarrow 1$ **to** N **do**
  Randomly choose an input observation;
  for $j \leftarrow 1$ **to** N **do**
    if $r_c - r_j \leq \sigma(t)$ **then**
      Update the neighboring node of BMU by
      $W_j(t+1) = W_j(t) + \alpha(t)h(r_c - r_j)[X(t) - W_j(t)]$
    end
    Decay $\alpha(t)$ and $\sigma(t)$
  end
end
Output: $W_j(u), j = 1, 2, \ldots, N$

whether the j-th window significantly deviates from the trained SOM grid, we first determine μ_i and σ_i^2. Since, in our work, the training set consists only of normal observations, we treat the training set as a pseudo-population. Using this training set, we obtain alternative values to the true mean and variance and $\tilde{\mu}_i$ and $\tilde{\sigma}_i^2$ can be defined as follows:

$$\tilde{\mu}_i = \frac{1}{m}\sum_{j=1}^{m} \tilde{d}_{ji} \quad , \quad \tilde{\sigma}_i^2 = \frac{1}{m-1}\sum_{j=1}^{m}(\tilde{d}_{ji} - \tilde{\mu}_i)^2, \tag{4}$$

where $\tilde{d}_{ji}$ is the distance between the i-th node codebook and the j-th observation of the training set. Also, we define the pseudo-mean and variance constants as follows:

$$\tilde{\mu} = \frac{1}{N}\sum_{i=1}^{N} \tilde{\mu}_i, \qquad \tilde{\sigma}^2 = \frac{1}{N}\sum_{i=1}^{N} \tilde{\sigma}_i^2. \tag{5}$$

For each window, we conduct hypothesis testing to identify large codebook distances. For $t = 1, 2, \ldots$, we compare the average of the mean with the pseudo-mean constant by a right tailed test as follows:

$$H_{0t} : \frac{1}{N}\sum_{i=1}^{N} \mu_i = \tilde{\mu} \quad vs. \quad H_{1t} : \frac{1}{N}\sum_{i=1}^{N} \mu_i > \tilde{\mu}. \tag{6}$$

Rejecting the t-th null hypothesis corresponds to marking the t-th window as anomalous, because it indicates that the average distance of the window is larger than that of normal samples. To apply Eq. 6 for a fixed window t, we define the sample mean $\overline{D}_t = \frac{1}{N}\sum_{i=1}^{N} D_{ti}$. Based on the mutual independence

assumption of $\{D_{ti}\}_{i=1}^{N}$, we can employ the Lindeberg-Feller CLT [22] to define the following test statistic:

Definition 1 (SomAnomaly Statistic). *Consider a t-th test, shown in Eq. 6. Then, we define the* ***SomAnomaly statistic*** *for each $t = 1, 2, \ldots,$ as follows:*

$$S_t = \frac{1}{B_N} \sum_{i=1}^{N} (D_{ti} - \tilde{\mu}_i) = \frac{N(\overline{D}_t - \tilde{\mu})}{B_N}, \tag{7}$$

where $B_N^2 = \sum_{i=1}^{N} \sigma_i^2$.

To test each H_{0t}, we explore the null distribution of S_t. First, suppose that for each $t \in \mathbb{N}, i \in \{1, \ldots, N\}$,

$$E[D_{ti}] = \mu_i < \infty,\ Var[D_{ti}] = \sigma_i^2 < \infty. \tag{8}$$

Let $Y_{ti} = D_{ti} - \mu_i$ and let $B_N^2 = \sum_{i=1}^{N} \sigma_i^2$. Now we assume the Lindeberg condition.

Assumption 1 (Lindeberg Condition). Consider $Y_{ti},\ t = 1, \ldots,\ i = 1, \ldots, N$. For every $\epsilon > 0$,

$$\frac{1}{B_N^2} \sum_{i=1}^{N} E\left[Y_{ti}^2 I\left(|Y_{ti}| \geq \epsilon B_N\right)\right] \to 0 \quad \text{as} \quad N \to \infty.$$

According to the Lindeberg condition, the *SomAnomaly* statistic weakly converges to the standard normal distribution under the corresponding null hypothesis.

Furthermore, for *SomAnomaly*, we can compute the p-value P_t for each t-th test as follows:

$$P_t = Pr(Z \geq s_t), \quad Z \sim \mathcal{N}(0, 1), \tag{9}$$

where s_t is the observed *SomAnomaly* statistic. We reject the test in Eq. 6 if P_t is smaller than the significance level α. However, note that we are conducting a sequence of tests; if we compare P_t with α for every t, the type I error or false discovery rate [4] may increase. Therefore, as shown in Fig. 2, we apply online multiple testing that can control these types of errors. To that end, we employ Generalized α-investing (GAI) [1], which controls the marginal false discovery rate (mFDR) under the significance level α [10]. Algorithm 2 presents the application of GAI to our problem. Also, further optimization can be achieved to remove empty grids that are not used in Defn. 1.

Based on this, we propose Optimized SomAnomaly Statistic, defined as follows:

Algorithm 2: Generalized α-investing (GAI) using SomAnomaly

Data: Trained SOM on the normal tensor input data
Input: Window size, shift size, α, η, ρ

Initialize $W(0) = \alpha\eta$
for $t = 1, 2, \ldots$ **do**
  Compute *SomAnomaly* and its p-value P_t for the streaming window
  $$\phi_t = \frac{1}{10} W(t-1)$$
  Set α_t such that
  $$\frac{\phi_t}{\rho} = \frac{\phi_t}{\alpha_t} - 1$$
  Test t-th hypothesis as follows:
  $$R_t = \begin{cases} 1 & P_t \leq \alpha_t \\ 0 & \text{otherwise} \end{cases}$$
  $$\psi_t = \min\left(\frac{\phi_t}{\rho} + \alpha, \frac{\phi_t}{\alpha_t} + \alpha - 1\right)$$
  $$W(t+1) = W(t) - \phi_t + R_t\psi_t$$
end
Output: Results of the tests $\{R_1, R_2, \ldots\}$

Definition 2 (Optimized SomAnomaly Statistic). *Let v be the index of mapped nodes and $B_v^2 = \sum_{i \in s} \sigma_i^2$. Then, the* ***Optimized SomAnomaly Statistic*** *is defined as:*

$$S_t^* = \frac{1}{B_v} \sum_{i \in v} (D_{it} - \tilde{\mu}_i),\ t = 1, 2, \ldots \tag{10}$$

Our experimental results show that S_t^* in Defn. 2 can detect anomalies more effectively than S_t presented in Defn. 1; hereafter, we refer to *SomAnomaly* as S_t^* presented in Defn. 2.

5 Experiment

We use two benchmark CPS datasets, SWaT and HAI, as well as commonly used neural network models, such as Seq2Seq, MDN, and RNNs, to generate FE patterns, as base predictors. We use the same Seq2Seq, MDN, and RNN as proposed by Kim et al. [20] and Bishop [5]. For the SWaT dataset, we use the first 7 days worth of data, containing only normal instances, for training and the next 4 days worth of data, comprising 36 attacks, for testing. For the HAI dataset, we use the first 7 days worth of data for training and the next 7 days worth

Table 1: Description of the base forecasting error models (underlying neural network classifiers) and CPS datasets

Dataset/NN Classifier	Forecasting model and CPS dataset
SWaT/seq2seq	seq2seq for each station in SWaT [20]
SWaT/MDN	MDN for each station in SWaT
SWaT/RNN	RNN for 14 correlation groups in SWaT
HAI/RNN	RNN for 14 correlation groups in HAI

of data, comprising 38 attacks, for testing. We only train with the normal data for both SWaT and HAI, formulating a more challenging, yet realistic anomaly detection scenario, where anomalous events rarely occur. For comparison, we use four variants of the datasets and prediction model combinations, as shown in Table 1.

In addition, we use the time series-aware performance evaluation metric TaPR[5] [17] and TSAD[6] [30], since the conventional precision, recall, and F_1 metrics cannot effectively capture the detection performance on highly imbalanced data [17,30] such as in our case, where the datasets are mostly composed of normal data. Specifically, we set the detection scoring parameter, weight for the detection score, and the subsequent scoring parameter as 0.001, 0.8, and 60, respectively. A detection scoring parameter of 0.001 indicates that when the anomaly detection algorithm succeeds at detecting at least 0.001 of an attack range, it generates a score; a subsequent scoring parameter of 60 indicates that it would admit a minute-after-attack detection. We put more weight to the detection score than the range detection, such that 0.8(detection score) + 0.2(overlap score) [17]. For TSAD, we use the default settings suggested in the original GitHub repository by Intel. We consider the following three approaches based on seq2seq, MDN, and RNN: 1) static threshold, 2) CUSUM, and 3) SOMAD, of which the first and second are our baselines. As in the work by Kim et al. [20], we use the 4-norm, which yields the best performance for static thresholding and CUSUM. We conduct our experiments under the following settings: MacOS Catalina machine —2.3 GHz 8-core Intel Core^{TM} i9-9880H processor, 32GB 2667 MHz DDR4 onboard memory, Intel ® UHD Graphics 630 1536 MB, and AMD Radeon^{TM} Pro 5500M 4 GB.

We also measure the average running time of SOMAD using Python. The whole process from training to detection takes less than 14 minutes in the worst case. SOMAD thus enables real-time detection of anomalies, making it more computationally efficient and practically applicable compared to deep learning-based approaches that take much longer to train.

[5] https://github.com/saurf4ng/TaPR
[6] https://github.com/IntelLabs/TSAD-Evaluator

6 Results

Table 2: Results using TaPR-based recall (Re), precision (Pr), and F_1 score.

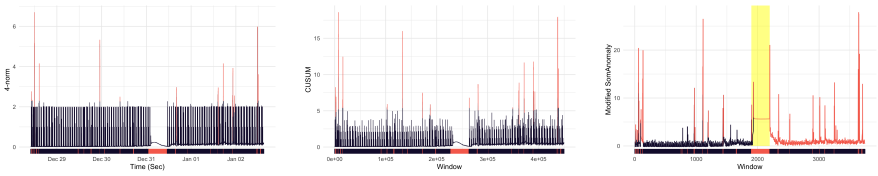
Method	SWaT/seq2seq			SWaT/MDN			SWaT/RNN			HAI/RNN		
	Re	Pr	F_1	Re	Pr	F_1	Re	Pr	F_1	Re	Pr	F_1
Static	0.44	0.45	0.45	0.63	0.40	0.49	0.78	0.64	0.70	0.87	0.76	0.81
CUSUM	0.58	0.70	0.63	0.64	0.56	0.59	**0.79**	0.59	0.67	0.71	0.52	0.60
SOMAD	**0.65**	**0.94**	**0.77**	**0.94**	**0.81**	**0.87**	0.76	**0.93**	**0.84**	**0.88**	**0.79**	**0.83**

Table 3: Results using TSAD-based recall (Re), precision (Pr), and F_1 score.

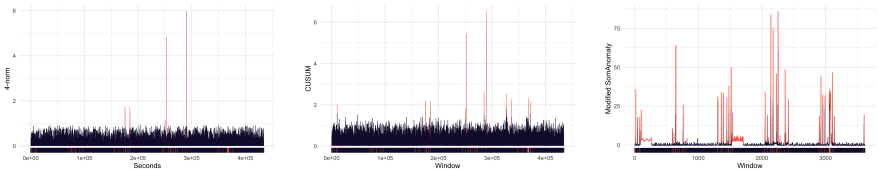
Method	SWaT/seq2seq			SWaT/MDN			SWaT/RNN			HAI/RNN		
	Re	Pr	F_1	Re	Pr	F_1	Re	Pr	F_1	Re	Pr	F_1
Static	0.25	0.41	0.31	0.34	0.35	0.35	0.33	**0.55**	0.42	0.20	0.71	0.31
CUSUM	0.30	**0.62**	0.40	0.38	0.39	0.38	0.37	0.45	0.41	0.36	0.44	0.39
SOMAD	**0.61**	0.60	**0.61**	**0.92**	**0.58**	**0.71**	**0.59**	0.54	**0.57**	**0.65**	**0.79**	**0.71**

Overall performance. We present our results for TaPR [17] and TSAD [30] in Table 2 and Table 3, respectively, for each combination of datasets. We also visualize the detection results of each method for SWaT/RNN and HAI/RNN in Fig. 3, where the X and Y-axes represent time and anomaly score, respectively. As shown in Table 2, SOMAD outperforms the baseline methods in terms of the time series-aware F_1 score in all cases, achieving 0.77, 0.87, 0.84, and 0.83 for the datasets SWaT/seq2seq, SWaT/MDN, SWaT/RNN, and HAI/RNN, respectively. Although the overall precision, recall, and F_1 score using TSAD are generally lower than those using TaPR, SOMAD still outperforms all baseline approaches in terms of the TSAD F_1 score. Similar to the results for TaPR shown in Table 2, SOMAD consistently achieves high recall compared to baseline methods. Therefore, even though the absolute performance for TSAD is lower for each dataset combination, the relative performance remains more or less the same, demonstrating the effectiveness of SOMAD.

Comparison with baseline approaches. On the other hand, the baseline methods, static thresholding and CUSUM, achieve mediocre performance.



(a) Static - first group of SWaT/RNN (b) CUSUM - first group of SWaT/RNN (c) SOMAD - SWaT/RNN



(d) Static - first group of HAI/RNN (e) CUSUM - first group of HAI/RNN (f) SOMAD - HAI/RNN

Fig. 3: Plots of anomaly detection results for SWaT/RNN and HAI/RNN. We plot the results of data corresponding to the first stations of SWaT and HAI for static thresholding and CUSUM, and those of data corresponding to all stations for SOMAD. The predicted anomalies are colored in red, the predicted normal instances are colored in black, and the strip along the X-axis is the ground truth.

Although static thresholding achieves a reasonable performance for RNN-based datasets (F_1 scores of 0.70 for SWaT/RNN and 0.81 for HAI/RNN), it does not perform well on the other datasets (F_1 scores of 0.45 for SWaT/Seq2Seq and 0.49 for SWaT/MDN). For CUSUM, it performs better for Seq2Seq and MDN-based datasets, but worse for RNN-based datasets. As illustrated in Fig. 3, the values used to distinguish normal data from anomalies are higher for static thresholding and CUSUM than for SOMAD. Moreover, both baseline methods produce high false positives; using TSAD, they achieve recall below 40% in all cases. In contrast, SOMAD achieves both high recall and precision in all cases, outperforming the baseline methods in terms of time series-aware recall except for one case and TSAD precision in two cases; this indicates that our proposed method can accurately detect a larger amount of anomalies.

Discussion. The performance difference between baseline approaches and SOMAD is mainly attributed to whether the method is capable of detecting clustered anomalies. Given that both SWaT and HAI datasets contain multiple clusters of consecutive anomaly samples over time, SOMAD successfully detects most of them unlike the baseline methods, as shown in Fig. 3. A time series prediction method based on SOM, which is characterized by its locality [3], exerts a clustering effect, leading to reduced false alarm rates and consequently to enhanced detection power compared to the baseline approaches for SWaT and HAI. As shown in the highlighted region in Fig. 3c, SOM's locality property is readily reflected in our anomaly detection task as well. On the other hand,

SOMAD incorrectly classifies normal samples as anomalies, hence the false positives, especially for samples corresponding to timestamps far apart from those of training data. This performance loss is mainly due to the long-term dependency issue persisting in popular deep learning-based forecasting models. Since they are used to generate our input data, i.e., the forecasting errors, SOMAD naturally suffers from worse-quality data originating from inaccurate forecasting when detecting anomalies.

7 Conclusion

Our proposed anomaly detection framework SOMAD inflates the differences between the respective FE patterns of normal and abnormal events. As demonstrated through numerous experiments under a realistic scenario where anomalies are only present in the test data, SOMAD outperforms conventional methods, achieving a high detection rate without compromising precision. While most of the prior work focused on improving the base forecasting model itself, our proposed approach shows great promise in detecting anomalies even with FE values that are either very small or similar to those of normal instances.

Acknowledgments

The work was supported by the affiliated institute of ETRI [2019-065]. Also, this work was partly supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by Korea government Ministry of Science, ICT (MSIT) (No. 2019-0-01343, Regional strategic industry convergence security core talent training business) and the Basic Science Research Program through National Research Foundation of Korea grant funded by Korea government MSIT (No. 2020R1C1C1006004). Additionally this research was partly supported by IITP grant funded by the Korea government MSIT (No. 2021-0-00017, Original Technology Development of Artificial Intelligence Industry) and was partly supported by the Korea government MSIT, under the High-Potential Individuals Global Training Program) (2019-0-01579) supervised by the IITP. Finally, this work was partly supported by Institute of Information & communications Technology Planning & Evaluation(IITP) grant funded by the Korea government (MSIT) (No.2019-0-00421, AI Graduate School Support Program (Sungkyunkwan University)).

References

1. Aharoni, E., Rosset, S.: Generalized α-investing: definitions, optimality results and application to public databases. Journal of the Royal Statistical Society: Series B (Statistical Methodology) **76**(4), 771–794 (2014). https://doi.org/10.1111/rssb.12048, `https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/rssb.12048`

2. Aoudi, W., Iturbe, M., Almgren, M.: Truth will out: Departure-based process-level detection of stealthy attacks on control systems. In: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security. pp. 817–831 (2018)
3. Barreto, G.A.: Time series prediction with the self-organizing map: A review. Perspectives of neural-symbolic integration pp. 135–158 (2007)
4. Benjamini, Y., Hochberg, Y.: Controlling the false discovery rate: a practical and powerful approach to multiple testing. Journal of the Royal statistical society: series B (Methodological) **57**(1), 289–300 (1995)
5. Bishop, C.M.: Mixture density networks (1994), `http://publications.aston.ac.uk/id/eprint/373/`
6. Campello, R.J., Moulavi, D., Zimek, A., Sander, J.: Hierarchical density estimates for data clustering, visualization, and outlier detection. ACM Transactions on Knowledge Discovery from Data (TKDD) **10**(1), 5 (2015)
7. Ferguson, T.S.: A Course in Large Sample Theory. Kogan Page Publishers (Jul 1996)
8. Filonov, P., Kitashov, F., Lavrentyev, A.: RNN-based Early Cyber-Attack Detection for the Tennessee Eastman Process. CoRR (Sep 2017)
9. Filonov, P., Lavrentyev, A., Vorontsov, A.: Multivariate industrial time series with cyber-attack simulation: Fault detection using an lstm-based predictive data model. ArXiv **abs/1612.06676** (2016)
10. Foster, D.P., Stine, R.A.: α-investing: a procedure for sequential control of expected false discoveries. Journal of the Royal Statistical Society Series B-Statistical Methodology **70**(2), 429–444 (Apr 2008)
11. Fuertes, S., Picart, G., Tourneret, J.Y., Chaari, L., Ferrari, A., Richard, C.: Improving spacecraft health monitoring with automatic anomaly detection techniques. In: 14th International Conference on Space Operations. p. 2430 (2016)
12. Giraldo, J., Urbina, D., Cardenas, A., Valente, J., Faisal, M., Ruths, J., Tippenhauer, N.O., Sandberg, H., Candell, R.: A survey of physics-based attack detection in cyber-physical systems. ACM Computing Surveys (CSUR) **51**(4), 76 (2018)
13. Goh, J., Adepu, S., Junejo, K.N., Mathur, A.: A Dataset to Support Research in the Design of Secure Water Treatment Systems. In: CRITIS. pp. 88–99. Springer International Publishing, Cham (2016)
14. Goh, J., Adepu, S., Tan, M., Lee, Z.S.: Anomaly detection in cyber physical systems using recurrent neural networks. In: 2017 IEEE 18th International Symposium on High Assurance Systems Engineering (HASE). pp. 140–145. IEEE (2017)
15. Hautamaki, V., Karkkainen, I., Franti, P.: Outlier detection using k-nearest neighbour graph. In: Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004. vol. 3, pp. 430–433. IEEE (2004)
16. He, Z., Xu, X., Deng, S.: Discovering cluster-based local outliers. Pattern Recognition Letters **24**(9-10), 1641–1650 (2003)
17. Hwang, W.S., Yun, J.H., Kim, J., Kim, H.C.: Time-series aware precision and recall for anomaly detection: Considering variety of detection result and addressing ambiguous labeling. In: Proceedings of the 28th ACM International Conference on Information and Knowledge Management. pp. 2241–2244. ACM (2019)
18. Idé, T., Papadimitriou, S., Vlachos, M.: Computing correlation anomaly scores using stochastic nearest neighbors. In: Seventh IEEE international conference on data mining (ICDM 2007). pp. 523–528. IEEE (2007)
19. Khaitan, S.K., McCalley, J.D.: Design techniques and applications of cyberphysical systems: A survey. IEEE Systems Journal **9**(2), 350–365 (2014)

20. Kim, J., Yun, J.H., Kim, H.C.: Anomaly detection for industrial control systems using sequence-to-sequence neural networks (2019)
21. Kohonen, T.: Self-organized formation of topologically correct feature maps. Biological cybernetics **43**(1), 59–69 (1982)
22. Lindeberg, J.W.: Eine neue herleitung des exponentialgesetzes in der wahrscheinlichkeitsrechnung. Mathematische Zeitschrift **15**(1), 211–225 (1922)
23. Lunt, T.F., Jagannathan, R., Lee, R., Listgarten, S., Edwards, D.L., Neumann, P.G., Javitz, H.S., Valdes, A.: Ides: The enhanced prototype-a real-time intrusion-detection expert system. In: SRI International, 333 Ravenswood Avenue, Menlo Park. Citeseer (1988)
24. Lunt, T.F., Tamaru, A., Gillham, F.: A real-time intrusion-detection expert system (IDES). SRI International. Computer Science Laboratory (1992)
25. Malhotra, P., Ramakrishnan, A., Anand, G., Vig, L., Agarwal, P., Shroff, G.: Lstm-based encoder-decoder for multi-sensor anomaly detection. arXiv preprint arXiv:1607.00148 (2016)
26. Rolincik, M., Lauriente, M., Koons, H.C., Gorney, D.: An expert system for diagnosing environmentally induced spacecraft anomalies. In: Proc. of 5th Annual Space Operations and Applications Research Symposium. pp. 36–44 (1992)
27. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning representations by back-propagating errors. nature **323**(6088), 533–536 (1986)
28. Sebring, M.M.: Expert systems in intrusion detection: A case study. In: Proc. 11th National Computer Security Conference, Baltimore, Maryland, Oct. 1988. pp. 74–81 (1988)
29. Shin, H.K., Lee, W., Yun, J.H., Kim, H.: HAI 1.0: Hil-based augmented ICS security dataset. In: 13th USENIX Workshop on Cyber Security Experimentation and Test (CSET 20). USENIX Association (Aug 2020), `https://www.usenix.org/conference/cset20/presentation/shin`
30. Tatbul, N., Lee, T.J., Zdonik, S., Alam, M., Gottschlich, J.: Precision and recall for time series. In: Proceedings of the 32Nd International Conference on Neural Information Processing Systems. pp. 1924–1934. NIPS'18, Curran Associates Inc., USA (2018), `http://dl.acm.org/citation.cfm?id=3326943.3327120`
31. Zhai, S., Cheng, Y., Lu, W., Zhang, Z.: Deep structured energy based models for anomaly detection. arXiv preprint arXiv:1605.07717 (2016)
32. Zhang, C., Song, D., Chen, Y., Feng, X., Lumezanu, C., Cheng, W., Ni, J., Zong, B., Chen, H., Chawla, N.V.: A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data. arXiv preprint arXiv:1811.08055 (2018)