water

MDPI

Article

The Use of Hierarchical Temporal Memory and Temporal Sequence Encoder for Online Anomaly Detection in Industrial Cyber-Physical Systems †

Roman Malits * and Avi Mendelson

Department of Computer Science, Technion Institute of Technology, Haifa 32000, Israel; mendlson@technion.ac.il
* Correspondence: romanma@campus.technion.ac.il
† This article is a revised and expanded version of a paper entitled "The use of HTM and TSSE Encoder for Online Anomaly Detection in Industrial Cyber-Physical Systems" which was presented at "Design of Cyber-Secure Water Plants" (DCS-Water'24) Conference, Buford, GA, USA, 23–24 April 2024.

Abstract: This study introduces a novel, practical approach for designing a hierarchical online anomaly detection system for industrial cyber-physical systems. The proposed method utilizes the Hierarchical Temporal Memory (HTM) unsupervised learning algorithm, which requires data to be encoded as sparse binary distributed representations (SDRs). A new SDR encoding method termed the temporal sequence encoder (TSSE) is presented to convert system outputs into SDRs. This approach enables HTM to retain high memory capacity and robust performance when processing data streams of slowly varying physical measurements, typical of many industrial processes. The effectiveness of the proposed system is demonstrated on the Secure Water Treatment (SWaT) dataset, which comprises data collected from a fully operational, scaled-down water treatment plant. The system achieves a recall of 0.906, a precision of 0.935, and an F1 score of 0.92 on SWaT. Compared to previous methods, our approach achieves state-of-the-art recall (~5.3% improvement), along with competitive precision and F1 score, by learning in an online manner without the need for expensive dataset collection, labeling, or retraining phases. These findings suggest that the proposed online anomaly detection method can be effectively applied to a broad range of water treatment and large-scale industrial cyber-physical systems.

Keywords: Hierarchical Temporal Memory; SWaT; water treatment; temporal sequence encoder (TSSE); sparse distributed representation; online anomaly detection; online cyber-attack detection; cyber-physical systems

Academic Editor: Aditya Mathur

Received: 7 December 2024
Revised: 9 January 2025
Accepted: 21 January 2025
Published: 23 January 2025

Citation: Malits, R.; Mendelson, A. The Use of Hierarchical Temporal Memory and Temporal Sequence Encoder for Online Anomaly Detection in Industrial Cyber-Physical Systems. *Water* **2025**, *17*, 321. https://doi.org/10.3390/w17030321

1. Introduction

This study addresses the problem of online anomaly and cyber-attack detection in large-scale industrial cyber-physical systems (CPS), with a focus on water treatment systems. Those systems are considered essential to critical public infrastructure [1] and, due to their importance, are prime targets for cyber-attacks. For instance, in 2016, power distribution companies in western Ukraine were compromised, causing a power outage for more than 230,000 residents [2]. In 2017, malicious software attacked a safety system intended to prevent disasters at Saudi Arabian Oil Company Aramco [3]. Moreover, Stuxnet, the first known cyber-warfare weapon, targeted Iranian nuclear program infrastructure in 2010 [4].

Modern water treatment systems incorporate numerous subsystems with complex behavior, generating large volumes of high-speed, multivariate output data in real-time. These systems are also subject to concept drift, which may occur due to factors such as

changes in demand resulting from weather conditions. Designing an online anomaly detection system for such a setting is challenging. In the online approach, monitoring and analysis of system behavior are performed in real-time. Ideally, decisions should be made with minimum latency, enabling rapid operator response and potentially mitigating critical damage if the system becomes compromised. Due to its significance, online anomaly detection in cyber-physical systems is an active area of research, as reflected in numerous publications [5] and dedicated conferences [6].

Major strategies for online anomaly detection include signature-based, threshold-based, and behavior-based approaches. This work proposes a behavior-based anomaly detection system, as it has the potential to offer enhanced detection of previously unknown threats. In principle, an anomaly detection system should be capable of monitoring all aspects of system operation in real-time, encompassing both network traffic and the behavior of various system components.

The present study focuses on real-time monitoring of the physical system state. It is assumed that if a cyber-attack is engineered to alter the physical behavior of the system, the attacker is unlikely to be able to fully falsify all physical-level process information in real time. Consequently, monitoring the physical system state has the potential to detect intrusion and limit the extent of damage caused by such threats.

To achieve these objectives, a novel, practical approach for designing an online anomaly detection system is proposed, leveraging the Hierarchical Temporal Memory (HTM) unsupervised learning algorithm, originally inspired by J. Hawkins's theory of the neocortex [7]. The proposed system adopts a hierarchical, layered structure that enables tuning of system-wide anomaly detection performance according to the required sensitivity. This design also allows for potential integration into a comprehensive defense framework monitoring all aspects of water plant operation.

The effectiveness of the proposed anomaly detection system is demonstrated using the Secure Water Treatment (SWaT) dataset [8,9], which represents a fully operational, scaled-down water treatment plant.

Deep learning (DL) methods have recently shown promising performance in both offline and online anomaly detection for cyber-physical systems, including several studies reporting strong results on SWaT (see Section 2.1). The findings we present in this work suggest that the HTM-based approach is equally promising, offering multiple benefits in comparison to DL-based systems for water treatment applications. These advantages include HTM's ability to learn and predict multiple temporal sequences in real time, its capacity for continuous processing, and rapid adaptation to changing input statistics without the need for extensive retraining, its inherent robustness to noise through the use of sparse distributed representations (SDR), and its capacity for strong performance with minimal task-specific hyperparameter tuning.

The contributions of this work are as follows:

- A practical unsupervised online anomaly detection approach for large-scale industrial cyber-physical systems based on Hierarchical Temporal Memory (HTM) is introduced.
- A novel temporal SDR sequence encoder, referred to as TSSE, is proposed. This encoder enables the content- and order-sensitive encoding of an SDR temporal sequence into a single SDR representation.
- The proposed method is implemented and evaluated on the Secure Water Treatment (SWaT) dataset [8,9], which captures data from a fully operational, scaled-down water treatment plant.
- Experimental results indicate that the proposed approach achieves excellent detection performance compared to deep learning-based algorithms while offering multiple benefits in comparison to DL-based systems for water treatment applications.

- The method achieves a recall of 0.906, a precision of 0.935, and an F1 score of 0.92. Compared to previous methods, it demonstrates a state-of-the-art recall (~5.3% improvement) along with highly competitive precision and F1.

2. Materials and Methods

2.1. Related Works

Researchers have proposed a wide range of techniques and algorithms for detecting anomalies in large-scale industrial systems. One category of approaches referred to as model-based techniques employs complete system simulations and predefined rules to construct a model of nominal system behavior [10]. Although these methods are often computationally efficient and offer high interpretability, creating an accurate model of complex industrial systems is time-consuming and prohibitively expensive to build and maintain.

Statistics-based methods, such as autoregressive integrated moving averages (ARIMA) [11], rely on probabilistic distribution to identify low-probability events. The primary limitation of these approaches is the difficulty in selecting suitable models for high-dimensional, multivariate data that often exhibit non-stationary characteristics. In recent years, machine learning methods have gained considerable attention for anomaly detection [12,13]. Typically, these approaches utilize a combination of algorithms, including clustering [14], dimensionality reduction (for example, PCA [15]), and decision trees (for example, Isolation Forest [16]). Notably, deep learning (DL)-based methods have demonstrated promising results, employing models such as autoencoders [17], convolution neural networks (CNN) [18], and recurrent neural networks (RNN) [18,19], which learn a high-dimensional feature space boundary that distinguishes normal from abnormal behavior.

In the present study, the proposed method is evaluated using the SWaT dataset [8,9], which represents a full-scale industrial water treatment testbed. Within the literature, a variety of machine learning-based approaches have been proposed for anomaly detection on SWaT [18–31], with deep learning methods often yielding the highest performance [18–21,28,31]. However, training DL models typically necessitates large, representative datasets of system operations, which can be challenging to obtain when only limited examples of anomalies exist. Moreover, in real water treatment systems, statistics that characterize "normal" behavior may shift rapidly due to variations in user demand.

Consequently, DL-based models often need periodic retraining to adapt to new data distributions, a process that is both data-intensive and computationally expensive. We believe this is one of the significant challenges with DL-based anomaly detection for real water treatment systems. To mitigate the impact of non-stationary signals, some prior works (for example, [20,21]) have employed the Kolmogorov–Smirnov test to eliminate channels with non-stationary distributions. Although channel filtering can improve precision, it may introduce blind spots in the monitoring process by excluding valuable indicators of abnormal behavior.

To address these challenges, this study proposes a novel hierarchical anomaly detection approach based on the Hierarchical Temporal Memory (HTM) unsupervised learning algorithm [32]. The proposed HTM-based system does not require test data for channel selection or repeated training phases and allows monitoring of channels that exhibit non-stationary dynamics. Past research has shown that HTM can be a good fit for the task of online anomaly detection in real water treatment systems due to properties such as the ability to learn online and adapt to changes in input data statistics, robustness to noise, handling of multivariate data using sparse encoding, and good performance without the need for task-specific hyperparameter tuning.

Previous investigations [33] have reported that HTM achieves accuracy levels comparable to other high-performing methods in online sequence learning tasks. For univariate

real-time streaming, an HTM-based anomaly detection method has been shown to attain state-of-the-art performance on the NAB benchmark [34,35]. Other studies have demonstrated successful applications of HTM in resource consumption monitoring [36], vehicular controller area network security [37], and high-throughput implementations using field-programmable gate arrays [38]. Moreover, combining HTM with ensemble learning techniques has proven effective for detecting anomalies in multivariate data [39], while employing it as a one-class classifier has been shown to detect video-based abnormal trajectories [40]. The online and unsupervised capabilities of HTM have also been validated in real-time anomaly detection for smart grids, where the model can adapt to fluctuating data distributions with minimal training overhead while maintaining competitive performance [41].

Building on these findings, the present work introduces and evaluates a novel hierarchical anomaly detection system based on HTM. Experimental results on the SWaT dataset indicate that this approach achieves excellent detection performance compared to deep learning-based algorithms.

2.2. Background

2.2.1. SWAT Dataset

In this work, we test our method using the SWaT dataset. SWaT was created using a Secure Water Treatment testbed, a fully operational six-stage (P1. . .P6) scaled-down water treatment plant resembling real plants found in cities (see Figure 1). The facility is at the Singapore University of Technology and Design (SUTD). Its primary purpose is to enable experimentally validated research in the design of security and safety of cyber-physical systems [8,9]. The anomaly detection system proposed in our work is designed to detect intrusion through online monitoring of the physical system sensors and actuators' state. It is not monitoring network traffic. Therefore, we chose the "SWAT.A1 & A2_Dec 2015" dataset version because it contains the most significant amount of physical state data relative to other SWaT dataset versions. The dataset contains 964,722 labeled records collected at a frequency of 1 Hz during eleven days of continuous operation. The records represent data collected from 25 sensors and 26 actuators. The first seven days represent regular plant operation. During the last four days, 36 attacks were carried out, with 32 succeeding and four having no observable impact on the output data.

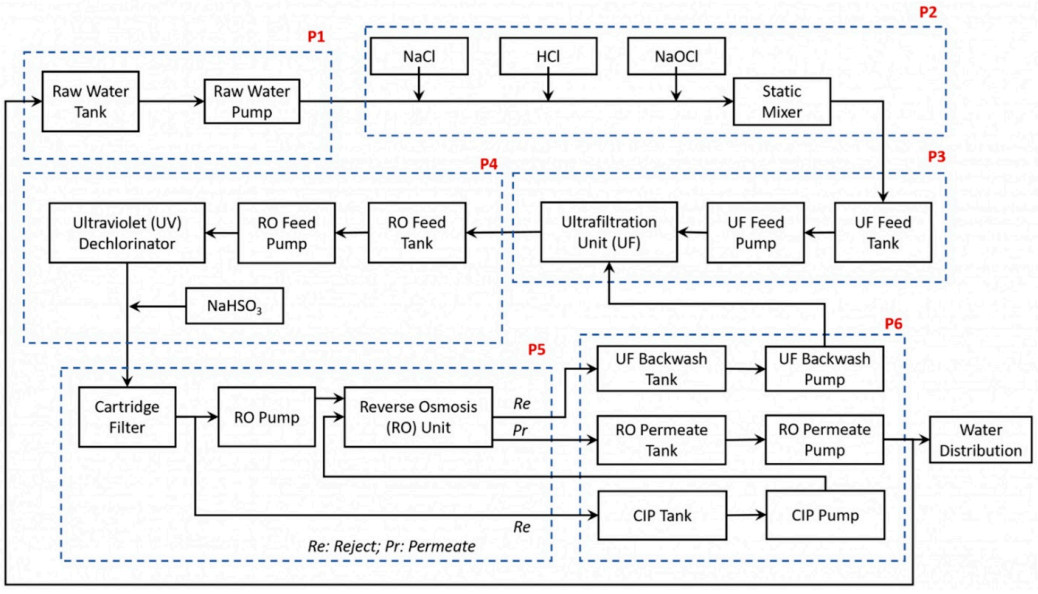


Figure 1. SWaT water treatment block diagram [9].

2.2.2. Hierarchical Temporal Memory (HTM)

HTM [32] is an unsupervised machine learning algorithm based on several known properties of the neocortex described in J. Hawkins's theory [7]. Figure 2 shows the essential components needed for HTM. Each time step t, output streaming data X_t is produced by the observed system for each physical/logical component. These data are transformed by the encoder into a sparse binary distributed representation (SDR) and used as input to temporal memory (TM). SDRs are large binary vectors with small sparsity, meaning that only a small percentage of bits are '1'. This allows SDRs to have high representation capacity. For a typical vector size of 2048 and sparsity of 2%, SDR supports ~2.37×10^{84} different encodings. SDR representation allows high noise robustness. The probability of a false positive partial match of 50% with a random SDR of the same size and sparsity is less than 4×10^{26} [42].

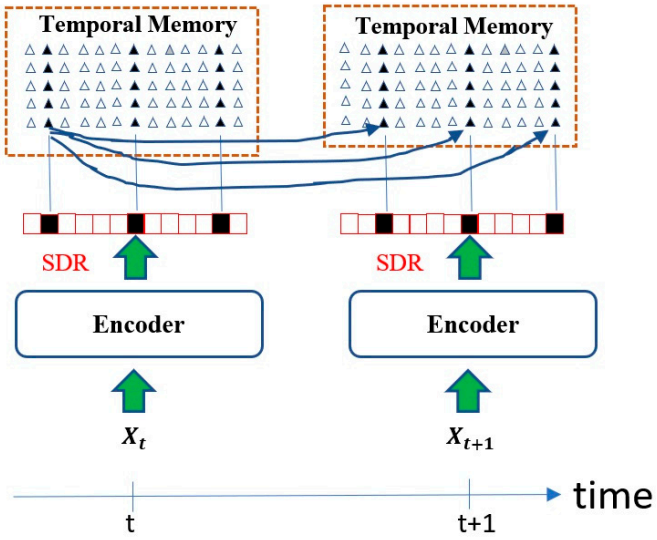


Figure 2. Each timestep encoder transforms system output data to SDR as input for TM.

TM is a neural network organized as a collection of cell columns. Cells in a single column have the same forward input represented by the 'i' bit in the input SDR. When the input SDR bit 'i' is '1', the feed forward-input signal to column 'i' is above the threshold, and the column is active. TM cells differ from the neuron model used in deep learning. In TM, in addition to feed-forward input, each cell output can potentially connect to the lateral input segment of other cells in the network. In each step, the cell is in one of three states: active, off, or predicted. A cell is predicted if the input to one of the cell lateral segments is above a threshold. When a column is active, only predicted cells become active. If no cells are predicted, all column cells will become active (we call this column bursting). This allows the network to predict the following SDR input based on the current input SDR and the learned temporal context. The instant anomaly score is computed based on the prediction error by calculating the ratio of bursting columns to active columns. Lateral connections topology constitutes the temporal memory of HTM. With each change in input SDR, the algorithm uses Hebbian learning to minimize the error of predictions by continuously updating the lateral connection strength between cells. The Hebbian learning algorithm reinforces synapses of correctly predicted cells and weakens synapses of mispredicted cells. This method allows HTM to learn a high-order temporal representation of statistically significant input patterns without overfitting to noise. Please see [33] for a complete description of HTM logic and learning rules. As the basis for our work, we used open-source HTM implementation [43].

2.3. SDR Encoder

2.3.1. Basic SDR Encoder

The task of the encoder is to convert observed system output data streams into SDRs used as input to the TM neural network. Formally, let X be an input space and let $S(n,s)$ be the set of SDRs of size 'n' and sparsity 's'. Encoder is a function f: $X \rightarrow S(n,s)$. We define the number of active bits in the SDR as $w_{(n,s)} = \lfloor n * s \rfloor$. The encoder should determine which SDR bits are active in a way that captures the essential semantic properties of the data. In general, different data types require different encoding schemes. According to [44], when encoding data to SDR, the following properties should be kept:

- The same input should produce the same output SDR encoding.
- As the similarity between inputs increases, the similarity between encoded SDRs should also increase.
- Encoded SDR vector should have enough active bits to handle noise in the input data.

In this work, we use two basic encoders described in [44] to encode data received in a single time step: a category encoder for discrete data and a scalar encoder for continuous data.

A category encoder is useful for discrete data with unrelated categories. The easiest way to encode this type of data are to use a predefined number of active bits for each possible value without overlap between different values. For SDR of size 'n' and sparsity 's', this method allows encoding $\left\lfloor \frac{n}{w_{(n,s)}} \right\rfloor$ separate categories. To use this encoder, the anomaly system designer should be able to define the number of states for each discrete channel.

A scalar encoder is suitable for encoding continuous numbers with a known range. Values outside the range will be truncated and encoded as min/max range values. To use this encoder, the anomaly system designer should be able to define the min/max values for each continuous channel the system will not exceed under normal operating conditions.

In this method, the range of values is split into bins, and then the bins are mapped to overlapping sets of active SDR bits.

Formally, for SDR of size 'n' and sparsity 's', the number of different bins is $bins_{(n,s)} = n - w_{(n,s)} + 1$ and the resolution of each bin is $resolution_{(n,s)} = (\max_{value} - \min_{value})/bin_{(n,s)}$. As a toy example, for a data channel with values range of 0–10.0 encoded using SDR of size n = 100 and s = 2%, we will have $100 - 2 + 1 = 99$ bins and two active ('1') bits for each data value. Each bin represents $10/99$ = ~0.1 resolution. Value 0.0 will be encoded by [110. . .0], the value of 0.1 will be encoded by [0110. . .0], etc.

2.3.2. Temporal SDR Sequence Encoder (TSSE)

SwaT contains multiple streams of data that describe physical processes. According to [44], the Basic Scalar Encoder is the appropriate encoder for this type of value.

In practice, when we used this encoder in our system, we saw poor anomaly detection performance. Closely examining the system's behavior, we have noticed that values in those streams change slowly, sometimes less than 1% on the scale of minutes and sometimes hours, while sampled at 1 Hz resolution. Encoding this type of data using basic SDR scalar encoders results in TM receiving almost the same input SDR in consecutive time steps since channel values were encoded using the same bin (see Section 2.3.1). This results in two problems. The first problem is related to HTM capacity. In theory, using multiple cells in each column allows TM to "remember" many SDR sequences of various context lengths. This is true if input sequences are represented by SDR with low overlap. In practice, since the data are sampled at a very high rate relative to the value change rate, TM is introduced with multiple slowly changing SDR sequences, resulting in fast memory capacity exhaustion for the active columns. The second problem deals with prediction.

When the input values change a statistically significant number of times from A→B→C and, in another instance, from C→B→A, then given input A→B, we expect the TM to predict the following input to be C. The problem arises when A, B, and C are encoded to similar SDRs that differ by a few bits, and this is the case for slowly changing data streams representing physical processes.

Since the TM prediction threshold requires only partial SDRs overlap (for noise robustness), the prediction will be for both C and A. Thus, sequences A→B→A and C→B→C will not be considered an anomaly. To deal with both problems, we propose a novel encoder called the temporal SDR sequence encoder (TSSE). This encoder enables order- and content-dependent encoding of temporal and contextual patterns using SDRs. The encoder is designed to allow TM to learn the semantic properties of slowly changing signals in a way that will enable efficient memory use and good prediction and anomaly detection performance.

The basic idea is to feed TM with SDR representing a temporal pattern instead of a single value. To encode a temporal pattern of N values as SDR, we combine the ideas of context-dependent thinning (CDT) [45] and algebra of computing with vectors by Kanerva [46]. Figure 3a shows that the Substructive CDT procedure allows binding multiple SDRs to a single SDR while preserving the desired size and sparseness. In addition, the resulting SDR is context-dependent such that, on average, each of the input sequence SDRs contributes the same number of active bits to the final result.

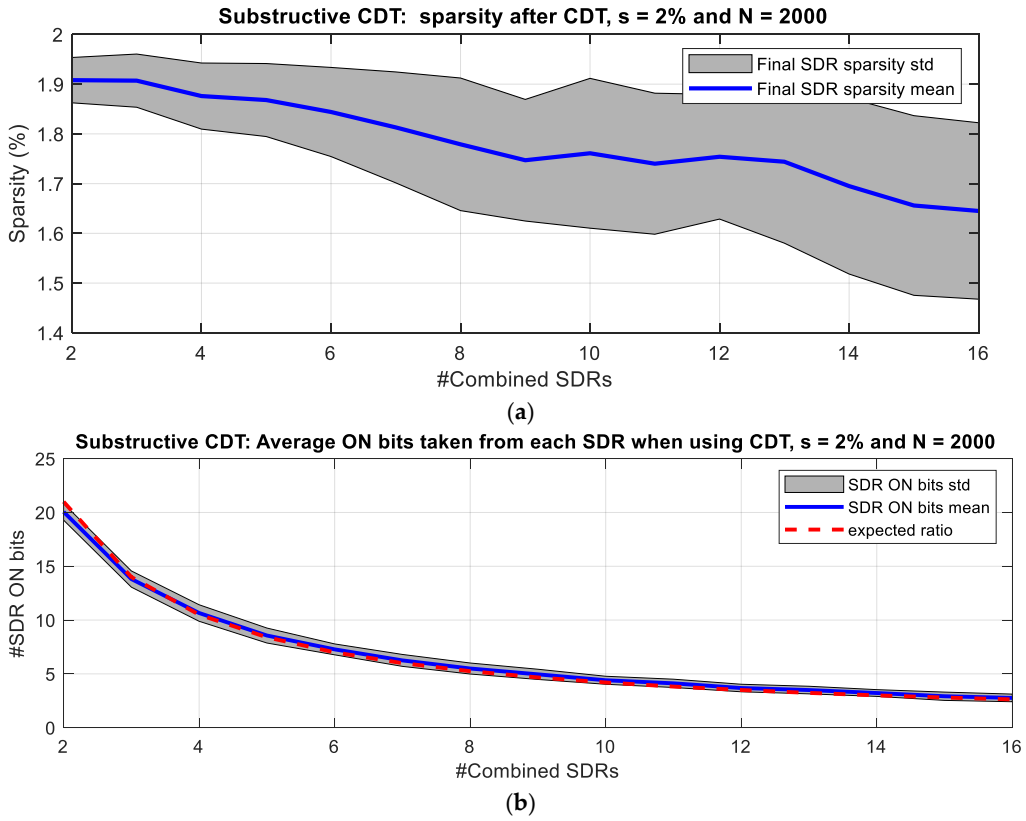


Figure 3. (**a**) Sparsity after substructive CDT procedure. (**b**) Average number of active bits in the final SDR taken from input SDRs.

TSSE consists of the following steps:

1. Given N input values (X^1_{input},. . ., X^N_{input}), encode each value using a basic scalar/category SDR encoder.

2. Given (X^1_{SDR},..., X^N_{SDR}), encode X^k_{SDR} position by performing permutation k times (p^k) [46]. This ensures that value encoding depends on both value and position, and similar consecutive values are represented by SDRs with low overlap.
3. Given ($p^1X^1_{SDR}$,..., $p^NX^N_{SDR}$), apply the Subtractive CDT procedure [45] to convert the temporal SDR sequence to a single SDR representing the pattern.

2.4. Online Anomaly Detection System Description

2.4.1. High-Level Architecture

In this work, we focus on anomalies that affect the physical behavior of the system. In the case of a cyber-attack, we assume that even if the system is compromised, the attacker can falsify only a subset of the monitored physical-level information. Complex water treatment systems require sophisticated anomaly detection methods that allow fine-tuning the detection sensitivity in a way that provides high precision and recall rates. To achieve this goal, we propose a hierarchical approach. The first level in the hierarchy aims to detect anomalies at a single-channel level. Higher levels are intended to detect cross-channel contextual anomalies and tune system-wide anomaly detection performance to meet the desired anomaly detection sensitivity. We show that using two monitoring levels (L1 and L2) is enough to achieve high performance for SWaT. Additional levels should be used to monitor subsystems in large-scale industrial systems. Figure 4 shows a high-level schematic view of the proposed anomaly detection system.

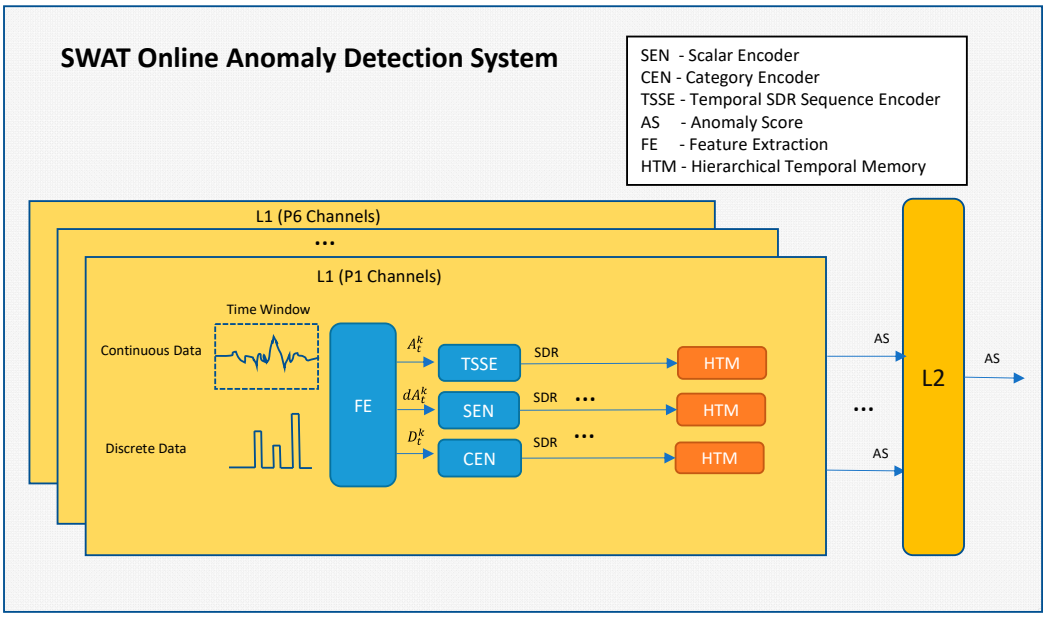


Figure 4. Schematic view of the proposed hierarchical anomaly detection system.

2.4.2. Dataset Preprocessing

An important step when working with a dataset is to visualize the various channels to understand the nature of the data. Figure 5 shows the values of several data channels for the first five days of recording. It is easy to see that the dataset includes both the testbed startup phase and the steady operation phase, for example, by looking at channels AIT202, AIT203, and P205. This work focuses on the steady operation phase starting roughly after 15:00 on 23 December, marked as a red dotted line in Figure 5. In addition, we discard the following channels since they are constant all the time: P202, P301, P401, P404, P502, P601. Channels P201, P204, and P206 are constant during the timeframe marked as normal operation. Any change in those channels in the test set should be considered an anomaly by the algorithm. Channels P204 and P206 exhibit a single change that coincides with

attack #24, and channel P201 shows many changes without correlation to attack labels. We decided to discard channels P201 and P205 and leave channel P204 to demonstrate the algorithm's ability to find trivial anomalies.

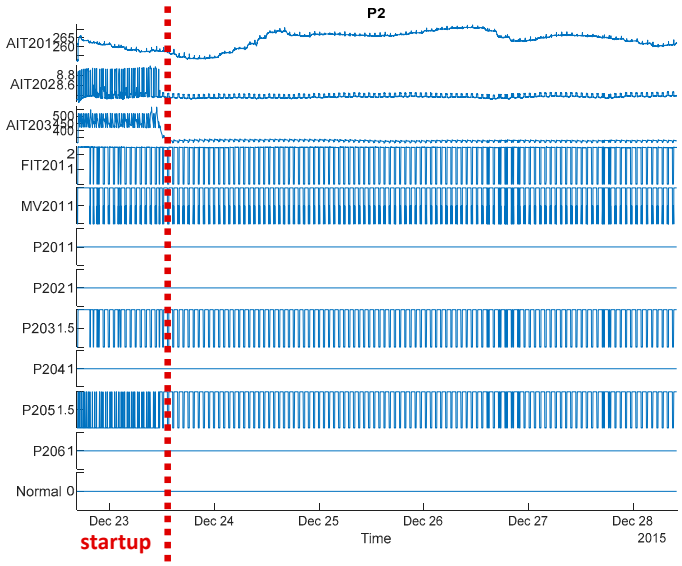


Figure 5. Discarded startup stage in the SwaT dataset, duration marked by the dotted red line.

2.4.3. L1 Logic

We define the monitored system output for channel k and time period T as $X_T^k = (x_1, x_2, \ldots, x_T)$. This is the input to L1. In this work, we used the following features to be monitored by the anomaly detection system:

- D_t^k—Discrete channel values.
- A_t^k—Continuous channel values.
- dA_t^k—First derivative of A_t^k.

For discrete channel k, feature extraction will produce a new value $D_t^k = x_t$ at time t if the current value differs from the previous value $(x_t \neq x_{t-1})$ or x_t did not change for a period of time Th_k separately configured for each discrete channel during training.

For continuous channel k, feature extraction will produce a new value A_t^k and dA_t^k each time step after applying a rolling window to smooth temporal noise. The rolling window size is separately configured for each continuous channel during training.

We define the last w measurements for channel k at time t as $X_{(t,w)}^k = (x_t, x_{t-1}, \ldots, x_{t-w+1})$. Then:

$$A_t^k = \overline{X_{(t,w)}^k} = \frac{\sum_{t-w+1}^{t} x_i}{w} \tag{1}$$

$$dA_t^k = \frac{\overline{X_{(t,w)}^k}}{w} - \frac{\overline{X_{(t-1,w)}^k}}{w} = \frac{x_t - x_{t-w}}{w} \tag{2}$$

L1 has a separate encoder and HTM network for each channel and feature type. For D_t^k, we use a basic category encoder. For A_t^k, we use a TSSE. For dA_t^k we use a basic scalar encoder. Figure 6 illustrates the encoding procedure in L1 for discrete channels. The output of the encoders is used as input to the HTM network. Finally, the instantaneous anomaly score for each channel feature type is calculated using the HTMs prediction error.

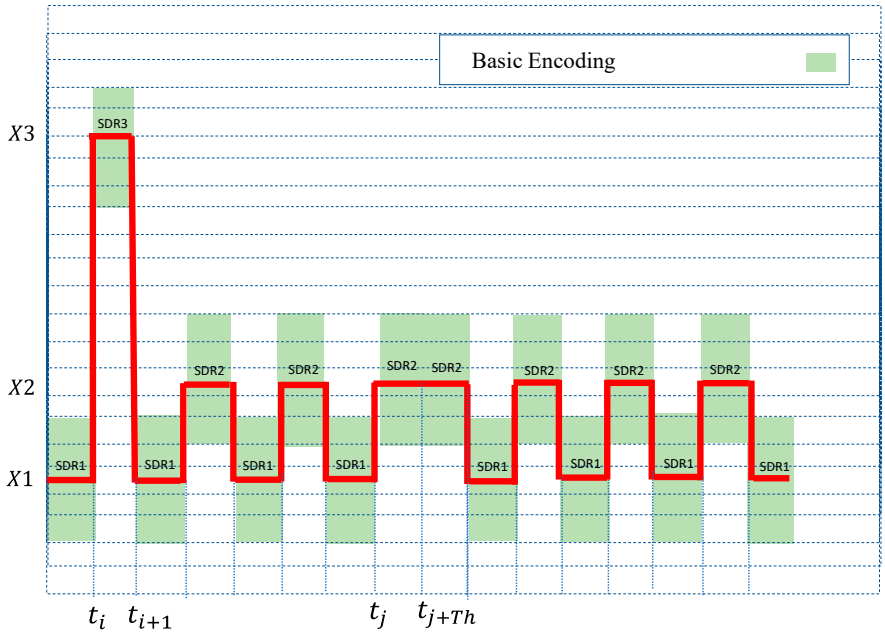


Figure 6. Encoding of discrete channels in the L1 layer, red line represents the channel value.

2.4.4. L2 Logic

The input to L2 is the instantaneous anomaly score calculated in L1 for each channel feature type. L2 uses this information to produce an anomaly score for the monitored system. For that purpose, we implemented a simple policy where an anomaly is declared if at least two simultaneous anomaly detections are above the threshold in different L1 channel features.

2.4.5. Anomaly Score Calculation

With HTM, the instantaneous anomaly score is computed as the ratio of bursting columns relative to the number of active columns (see Section 2.2.2). Anomalies in continuous channels describing physical processes, such as pressure build-up, sometimes manifest as a slow shift from normal behavior. In this case, HTM will produce several instantaneous low anomaly scores over a short time period.

To deal with this phenomenon, we use the sum anomaly score AS^t_{sum} in L2:

$$AS^t_{sum} = \max\left(\sum_{i=t-w_{AS}}^{t} AS^i_{instant}, 1\right) \tag{3}$$

where $AS^t_{instant}$ is the instant anomaly score at timestep t and w_{AS} is a time window.

Setting w_{AS} to 2 min and anomaly detection threshold to 0.7, which allows good performance for all channels. Figure 7 shows the difference between the instant and sum anomaly scores for the LIT101 channel. We can see that AS^t_{sum} gives a high score after the sensor value is out of the 'normal' range. $AS^t_{instant}$ score is high only when there is an abrupt change in the sensor value. During dataset examination, we noticed that SwaT physical channels affected by an attack need time to return to normal behavior. In some cases, the anomaly lasts longer than the attack label. In other cases, the anomaly starts shortly before/after the end of the label. To deal with this phenomenon, authors of [20] removed 3 min of normal behavior data after the end of each label. We deal with that situation using a grace period of T_{grace} = 20 min. If the anomaly score is above the detection threshold during the attack label, this detection is considered True Positive (TP). If there is another anomaly within T_{grace} window, we consider it a True Positive. Similarly, all False

Positive detections occurring in the same time window of T_{grace} are considered single False Positive (FP).

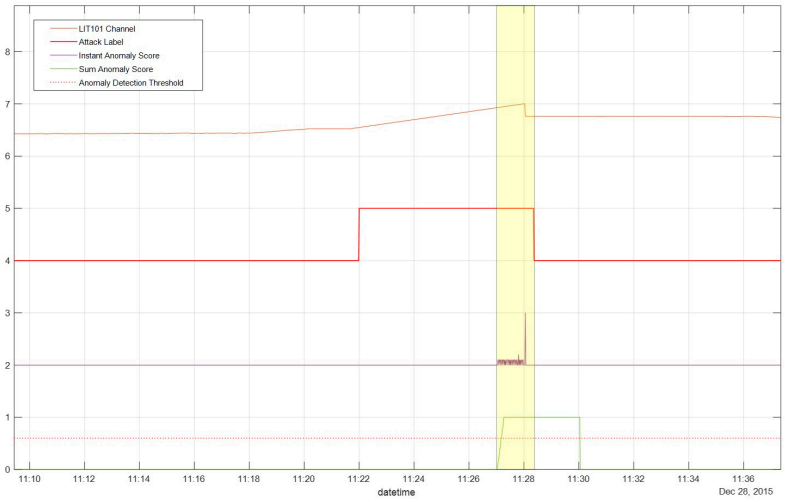
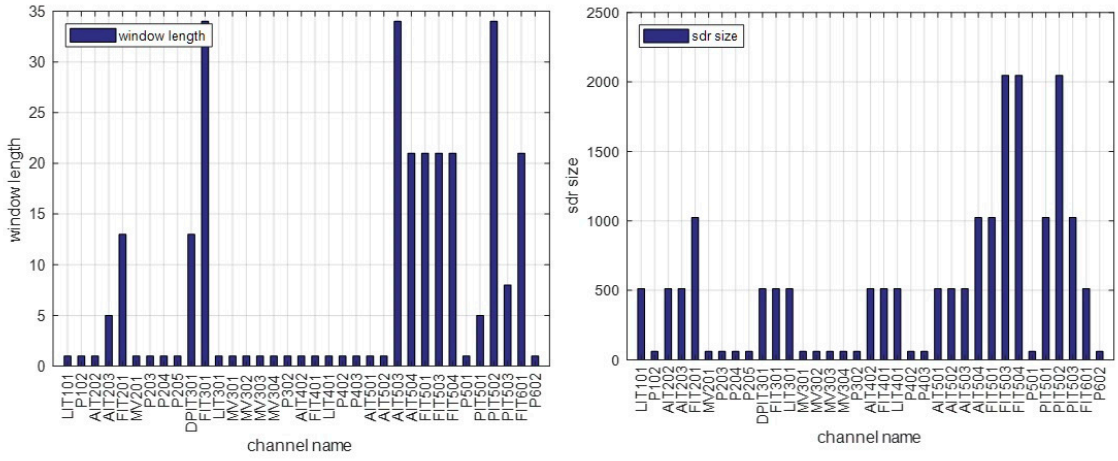


Figure 7. Example of Instant vs. Sum anomaly score for LIT101 channel. The difference can be seen during the time window marked by yellow color.

2.4.6. Hyperparameters Selection

To achieve good performance, we tune hyperparameters for encoders and HTM networks of each feature channel. To do this, we train HTM independently for different combinations of parameters using grid search (At this stage, we set TSSE sequence length to 1). For rolling window size, we test the following Fibonacci values [1, 2, 3, 5, 8, 13, 21, 34]. Figure 8a shows the rolling window values found for different A_t^k channels. In general, we found that a higher rolling window size is needed for noisy channels. Using a lower value than the one found during the search will usually result in a higher FP rate for that channel. For SDR size, we tested the following values [256, 512, 1024, 2048]. SDR size controls the resolution of encoding. Figure 8b shows the SDR size found for different A_t^k channels. For most A_t^k channels SDR length of 512 was found to be sufficient.



(**a**) (**b**)

Figure 8. (**a**) Example of best rolling window length for A_t^k (**b**) Best SDR length size for A_t^k.

Using a lower-than-optimum size will usually result in a lower TP rate, and using a higher value will usually result in a higher FP rate. For Th_k, we test the following Fibonacci values [5, 8, 13, 21, 34, 55, 89, 144, 233, 377]. This parameter allows TM to take into account the duration of a state for D_t^k channels, in addition to the state value transition. In general,

for channels that tend to change slowly, this parameter should have higher values. For each feature channel, we select hyperparameter values that minimize False Positives on training data.

The rest of the HTM parameters are set to recommended values [44], shown in Table 1. Our experiments found that the HTM configuration using those values performed generally well. In an operational system, we recommend tuning all parameters for each TM to optimize performance.

Table 1. Default HTM parameters.

HTM Parameter Name	Value
Cells per column	5
Max segments per cell	32
Max synapses per segment	256
Initial permanence	0.21
Connected permanence threshold	0.14
Permanence decrement	0.001
Permanence increment	0.1

The parameter 'Cells per column' value controls the TM network's memory context length. The parameter 'Max synapses per segment' limits the maximum number of cell connections. We found that lowering those values might cause higher FP, particularly in channels with higher SDR length. Optimizing those values allows us to reduce TM's memory footprint without harming performance. Other parameters influence TM's learning and adaptation speed. Our tests indicate that default values are adequate to achieve good anomaly detection performance on SWaT. Please note that we did not try to optimize memory usage in this work.

2.4.7. Feature Channels Selection

Often, researchers use statistical tests to select the channels they use. For example, the Kolmogorov–Smirnov test (K-S) [47] was used by several researchers for channel selection to test anomaly detection algorithms' performance for SWaT [18,20,21]. K-S is a non-parametric test that allows checking whether training and test data belong to the same statistical distribution without making assumptions regarding the target probability distributions.

The challenge with this method is to select the appropriate K-S threshold. We believe the assumption should be that the test set is unknown, and channel selection should rely only on training data.

Anomaly detection systems should monitor as much available information as possible while keeping the False Positive rate below an acceptable level. Therefore, instead of using statistical tests, we use the following procedure: After selecting the best hyperparameters for the feature channel, we rerun the algorithm on training data to decide if this feature channel should be used. The channel is discarded if a high False Positives rate occurs during the last 80% of the training data. Following this process, we discard the following channels: AIT201, AIT401, FIT502 for A_t^k, and AIT201, AIT401 for dA_t^k.

At the end of this process, the following channels are monitored by our anomaly detection system: LIT101, P102, AIT202, AIT203, FIT201, MV201, P203, P204, P205, DPIT301, FIT301, LIT301, MV301, MV302, MV303, MV304, P302, AIT402, FIT401, LIT401, P402, P403, AIT501, AIT502, AIT503, AIT504, FIT501, FIT502, FIT503, FIT504, P501, PIT501, PIT502, PIT503, FIT601, P602.

3. Results

We implemented and tested the anomaly detection system described in previous chapters. Similarly to previous works [18–20,27,28], we adopt the recall (RE), precision (PR), and harmonic mean of precision and recall (F1 score) as measures for anomaly detection performance evaluation.

Figure 9 shows the combined TP detection matrix for L1 layer features D_t^k and $A_t^k(TSSE)$. The x-axis represents the 41 attempted attack indexes, and the y-axis represents SWaT channel names. To keep the original attacks enumeration, the matrix includes nine black columns for attacks marked as failed (no effect) by dataset creators. Colored squares represent TP detections for various SDR sequence length combinations used as input to TSSE. The matrix contains results for combinations of SDR lengths one, two, and three. We can see that most attacks can be detected even if TSSE input is a single SDR (here, TSSE acts as a basic scalar encoder). We found that the TSSE SDR input sequence length of three is the general sweet spot for anomaly detection in SWaT.

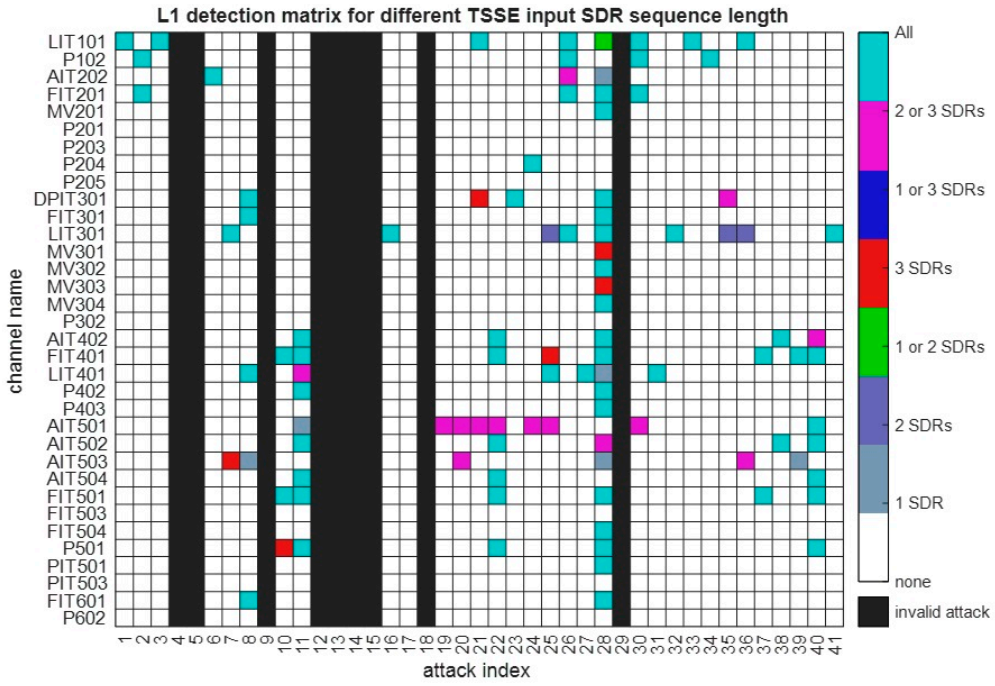


Figure 9. Combined TP detection matrix for L1 layer features D_t^k and A_t^k as a function of TSSE SDR input sequence lengths.

We have also tested TSSE SDR input sequence lengths of four and five, resulting in a lower TP and higher FP rate. We can see from Figure 9 that encoding a sequence of three SDRs with TSSE is generally good practice. Theoretically, this value can be adjusted on a per/channel basis. In the following figures, we present results for the TSSE SDR sequence lengths of three for all channels encoded by TSSE.

Figure 10 shows the combined detection matrix for all L1 channel features D_t^k, A_t^k(TSSE SDR input length of three), and dA_t^k. It is easy to see that the features complement each other.

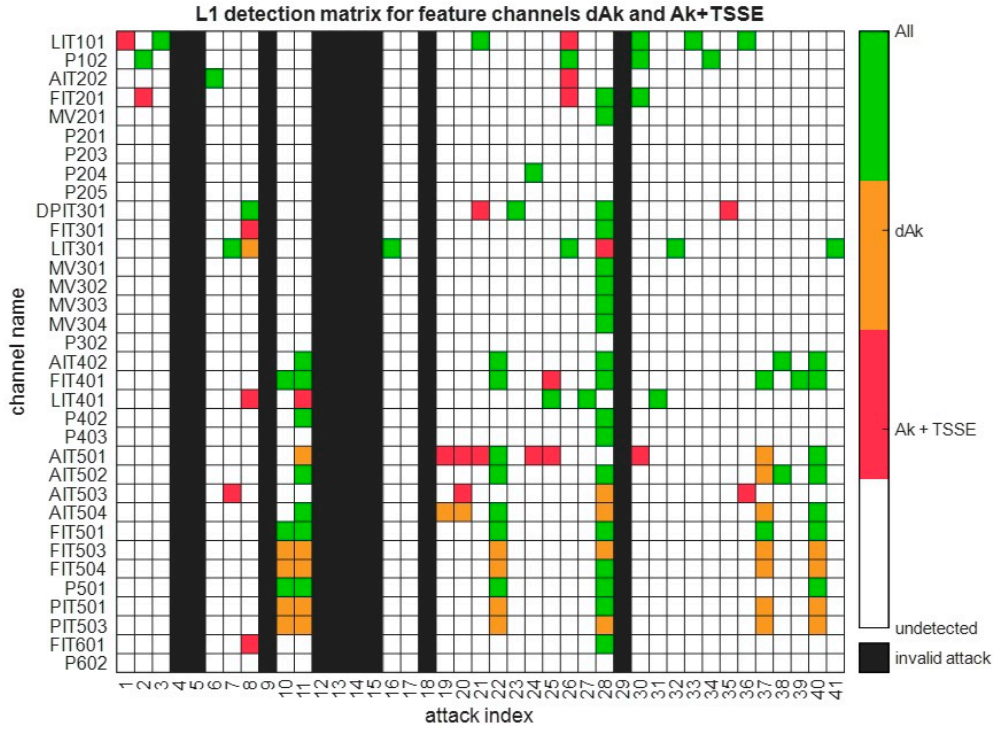


Figure 10. Combined TP detection matrix for all L1 channel features D_t^k, A_t^k and dA_t^k for TSSE length of 3.

Most attacks are detected by multiple features in the same channel (green color) or by the same/different features in multiple channels. As explained in previous chapters, an anomaly is declared by L2 if the anomaly score is higher than the anomaly detection threshold in at least two different L1 features/channels at the same T_{grace} period. Using Figure 10, we can see that 29 out of 32 attacks are TP detections by L2 using this logic. Attacks #1, #17, and #35 are not declared as anomalies by L2 because #1 is detected only by A_t^{LIT101} and #35 is detected only by $A_t^{DPIT301}$. This gives a total Recall of TP/(TP + FN) = 29/32 = 0.906. Figure 11 shows the TP and FP scores per L1 feature channel.

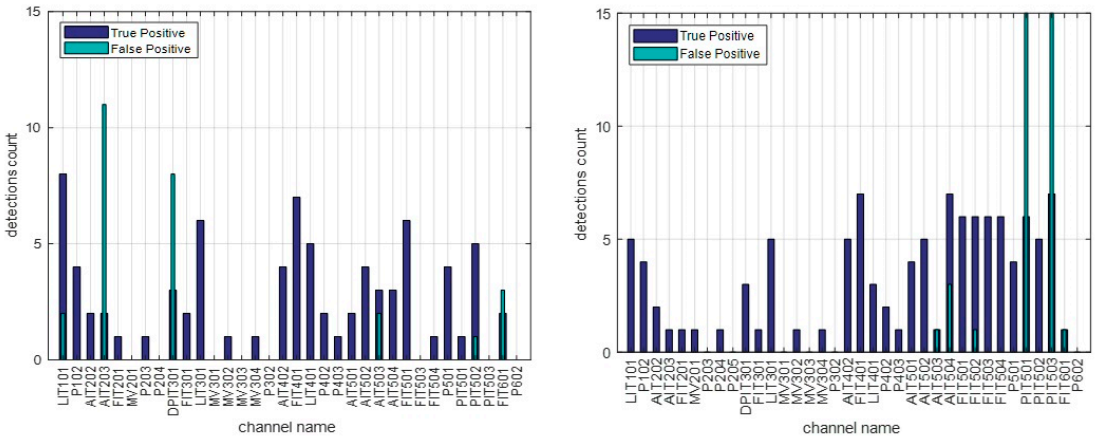


Figure 11. TP/FP scores per L1 channel for (**a**) A_t^k features (**b**) dA_t^k features.

For A_t^k, most False Positives are in channels DPIT301 and AIT203. For dA_t^k, most False Positives are in channels PIT501 and PIT503. Only two out of all the FPs occur in more than one feature channel within the same T_{grace} period and thus are declared as an anomaly by

L2 and considered as FP detections overall: $dA_t^{AIT503}/A_t^{AIT503}$ and $dA_t^{FIT601}/A_t^{FIT601}$. This gives a total Precision of TP/(TP + FP) = 29/31 = 0.935.

Table 2 compares the anomaly detection performance of our method with previous works. The best recall rates achieved by previous works are 0.83–0.86 [18,19,28] using deep neural networks after extensive DNN architectures search, using best hyperparameters and an average of 10 runs to account for random weight initialization variance.

Table 2. Performance comparison to previous works.

Methods	Recall	Precision	F1 Score
LTSM [20]	0.75	0.98	0.851
TABOR [27]	0.79	0.86	0.82
UAE [19]	0.78	0.97	0.86
VAE [19]	0.79	0.94	0.86
PCA [19]	0.84	0.92	0.88
UAE (frequency domain) [19]	0.86	0.91	0.88
LSTM-VAE-M (STD) [21]	0.8	0.898	0.84
1D CNN (combined attacks) [18]	0.79	0.95	0.86
1D CNN (ensembled records) [18]	0.85	0.87	0.86
1D CNN (ensembled attacks) [18]	0.85	1.0	0.92
1D-CNN + GRU [28]	0.83	0.995	0.91
HTM + TSSE	0.906	0.935	0.92

Our method based on HTM + TSSE achieved a state-of-the-art recall of 0.906 while achieving a precision of 0.935 and an F1 Score of 0.92.

4. Discussion

As shown in Table 2, numerous deep learning (DL)-based methods achieve high precision but do not consistently attain high recall. The best overall F1 score of 0.92 is jointly achieved by the DL-based method in [18] and the approach proposed in this study. Specifically, the proposed method achieves a recall of 0.906 and a precision of 0.935, whereas the DL-based method reported in [18] attains a precision of 1.0 but a recall of 0.85. Although both precision and recall are critical metrics for anomaly detection, achieving a higher recall is often more advantageous, provided the precision remains acceptable. This ensures that fewer anomalies are missed, thereby enhancing the likelihood of timely intervention.

In general, effective performance of DL methods requires that training and testing data originate from the same statistical distribution. However, the SWaT dataset contains several nonstationary channels, particularly in stages P2 and P6.

To address this issue, many DL-based methods employ strategies for "discarding" these channels; for instance, references [20,21] utilize the Kolmogorov–Smirnov test for channel selection, using both training and test data in this process and removing all P2 channels. Similarly, the authors in [18] evaluated anomaly detection performance separately for each stage (P1–P6) and excluded the results for P2 and P6 due to poor DL-based performance on those stages. Consequently, the reported performance reflects monitoring only four of the six water treatment stages, potentially creating blind spots that may fail to capture certain anomalies. In contrast, the Hierarchical Temporal Memory (HTM)-based anomaly detection system presented here monitors all stages. For example, attack #24 was not discovered in [18]. We can see in Figure 10 that our system detected anomaly #24 using both A_t^k and dA_t^k features on channel P204 (stage P2).

In real-world water treatment systems, the statistical properties of normal behavior often shift rapidly due to changing user demand. While DL-based methods show promising results [48,49] in cyber-physical systems, we believe this characteristic of real water

treatment systems poses a significant challenge for DL-based anomaly detection. DL-based solutions must be periodically retrained to address both newly discovered anomalies and evolving environmental conditions. This retraining process, which typically involves large datasets and extensive computational resources, can be prohibitively expensive. By contrast, the HTM-based anomaly detection system offers several potential advantages for water treatment applications. Specifically, HTM can learn and predict multiple temporal sequences in real time and can rapidly adapt to changing input distributions without expensive retraining. Furthermore, HTM employs sparse distributed representations (SDR), which are inherently robust to noise, and, as demonstrated in this work, maintains strong performance without extensive task-specific hyperparameter tuning.

The computational complexity of the HTM-based system is linear in the number of feature channels/TM networks. Within each TM network, the complexity scales with the number of network cells and the number of potential synapses per cell. TM learning involves strengthening or weakening the permanence of synapses on active cells (or cells that were incorrectly predictive), requiring only local activity updates rather than the global gradient descent processes typical of deep learning. Moreover, TM does not rely on large batch sizes or multiple passes over the same data, allowing it to operate in a purely streaming, online mode. However, because TM continuously learns at inference time, it can be more computationally expensive than deep learning, particularly if the network size (columns/cells and synapses) is large. Although computations can be potentially parallelized at the cell level, TM's sparse, fine-grained updates can be more memory-intensive or less compatible with GPU-optimized dense matrix operations compared to deep learning networks.

In our simulations, we have seen that memory-compute optimization is possible by performing extensive per-channel hyperparameter tuning. Please note that the training and hyperparameter optimization described in this work were performed solely to maximize anomaly detection accuracy. Using a standard personal computer with a Ryzen 7 3700 CPU and 32 GB of memory, processing the 11-day SWaT dataset required approximately 30 h in a sequential run.

By design, both the TM networks and the architecture of the proposed anomaly detection system support streaming, online operation. The results reported in this study therefore represent the expected performance under real-time conditions, wherein all L1 stages would simultaneously process sensor outputs, and subsequent L2/L3 decisions would be made in real-time.

5. Conclusions

This study introduces a novel, practical approach for designing a hierarchical online anomaly detection system for large-scale cyber-physical systems using sparse binary distributed representations (SDRs) and a Hierarchical Temporal Memory algorithm (HTM).

The first level of the proposed hierarchy detects anomalies at the individual channel level, while subsequent levels are responsible for detecting cross-channel contextual anomalies and adjusting the overall system sensitivity to ensure suitable anomaly detection performance.

A key contribution of this work is the introduction of a new SDR temporal sequence encoder, referred to as TSSE. In contrast to single-value encoding, TSSE produces SDRs that encapsulate ordered temporal patterns of sensor data. This way of encoding system sensor data allows HTM to learn the semantic properties of slowly changing signals in a way that enables efficient memory use and good prediction and anomaly detection performance.

Another contribution is the sum anomaly score calculation method that allows the discovery of slowly developing anomalies and minimizes algorithm performance evaluation errors due to situations where the anomaly starts shortly before/after the end of the label.

The proposed approach is evaluated using the Secure Water Treatment (SWaT) dataset, which replicates a fully operational, scaled-down water treatment plant. By employing two levels of monitoring (L1 and L2), the system demonstrates high anomaly detection performance on SWaT, outperforming various deep learning (DL)-based methods. Specifically, it achieves a recall of 0.906, a precision of 0.935, and an F1 score of 0.92, thus improving the state-of-the-art recall on SWaT by approximately 5.3% while maintaining competitive precision and F1 values.

While maintaining a low false alarm rate is essential, we believe that in critical infrastructure settings, anomaly detection capability is the most important metric while retaining an acceptable false positive rate.

Another important quality of our approach is the relatively low cost needed for deployment and maintenance. Compared to Deep Learning methods, our method does not require expensive dataset collection, labeling, and retraining phases. To start system monitoring, our system requires a supervised evaluation and stabilization period to allow tuning of hyperparameters and online unsupervised learning of the Pareto normal system behavior. During deployment, when the algorithm signals an anomaly, an analysis should be performed to decide if new normal behavior should be learned by the anomaly detection system. This "delayed" learning can be implemented using HTM with a replay buffer.

HTM-based systems offer several potential advantages in water treatment systems compared to DL-based anomaly detection systems. In particular, HTM-based system can learn and predict multiple temporal sequences in real-time. HTM continuously processes streaming data and can quickly adapt to changes in input statistics without the need for expensive retraining. HTM uses sparse distributed representations (SDR) for information processing and representation, which makes it inherently robust to noise. In addition, as we have shown, HTM allows good performance without extensive task-specific hyperparameter tuning.

These findings suggest that the proposed HTM-based online anomaly detection approach can be effectively applied to a broad range of water treatment and large-scale industrial cyber-physical systems.

In future work, we intend to test our method's performance on additional datasets to prove its generalizability. We hope the results presented in this work will encourage other researchers to test our method as well.

The architecture of our HTM-based online anomaly detection system is intended for streaming, online operation. As discussed in the previous chapter, TM network learning can be expensive. Additional research is needed to find more efficient ways for parallelized software and hardware implementation of our approach, in general, and the TM learning algorithm in particular.

This work shows that achieving high anomaly detection performance is possible while monitoring only the system's physical state. In future work, we intend to investigate the benefit of combining online monitoring of physical system state and network traffic. In addition, we intend to explore the use of TSSE for encoding multivariate contextual patterns by combining SDRs from different channels. We also plan to examine the effectiveness of more sophisticated logic for L2 that assigns a dynamic confidence level to each feature channel based on past false positive rates.

Author Contributions: Conceptualization, R.M. and A.M.; methodology, R.M. and A.M.; software, R.M.; validation, R.M.; formal analysis, R.M.; investigation, R.M.; resources, R.M. and A.M.; data curation, R.M.; writing—R.M. and A.M. original draft preparation R.M. and A.M.; visualization, R.M.;

supervision, A.M.; project administration, A.M.; funding acquisition, A.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported in part by a grant "Zero-Trust" working group, TII (Technology Innovation Institute), Emirate of Abu Dhabi, United Arab Emirates.

Data Availability Statement: The SWaT dataset used in this work is available by request from iTrust, Centre for Research in Cyber Security, Singapore University of Technology and Design: https://itrust.sutd.edu.sg/itrust-labs_datasets/dataset_info/ (accessed on 8 November 2021). The anomaly detection algorithm code used to create results for this paper is available from the following github repository under the MIT license: https://github.com/romanma9999/AD_SWAT_WATER (accessed on 8 November 2021).

Acknowledgments: This article is a revised and expanded version of a paper entitled "The use of HTM and TSSE Encoder for Online Anomaly Detection in Industrial Cyber-Physical Systems" which was presented at "Design of Cyber-Secure Water Plants" (DCS-Water'24) Conference 23–24 April 2024, Buford, GA, USA [50]. We thank iTrust, the Centre for Research in Cyber Security, and the Singapore University of Technology and Design for providing the SWaT dataset used in this study. We thank the creators of the htm.core project for providing the HTM algorithm code for this research [43]. The code for htm.core can be obtained from https://github.com/htm-community/htm.core (accessed on 8 November 2021) under the AGPL-3.0 license. We have accessed the code for the first time in July 2022.

Conflicts of Interest: The authors declare no conflicts of interest.

List of Abbreviations

HTM	Hierarchical Temporal Memory Algorithm
TM	Temporal Memory Network
SDR	Sparse Binary Distributed Representation
TSSE	Temporal SDR Sequence Encoder
CPS	Cyber Physical System
SWaT	Secure Water Treatment Dataset
CDT	Context Dependent Thinning
DL	Deep Learning
CNN	Convolution Neural Network
RNN	Recurrent Neural Network
L1	Monitoring Level 1
L2	Monitoring Level 2
D_t^k	Discrete channel k values
A_t^k	Continuous channel k values
dA_t^k	First derivative of channel k values
K-S	Kolmogorov–Smirnov test

References

1. Baheti, R.; Gill, H. Cyber-Physical Systems. In *The Impact of Control Technology*; IEEE Control Systems Society: Gainesville, FL, USA, 2011; pp. 161–166.
2. CFR. Compromise of a Power Grid in Eastern Ukraine. Council of Foreign Relations. 2015. Available online: https://www.cfr.org/cyber-operations/compromise-power-grid-eastern-ukraine (accessed on 15 July 2023).
3. Groll, E. Cyberattack Targets Safety System at Saudi Aramco. 2017. Available online: https://foreignpolicy.com/2017/12/21/cyber-attack-targets-safety-system-at-saudi-aramco/ (accessed on 15 July 2023).
4. Nourian, A.; Madnick, S. A Systems Theoretic Approach to the Security Threats in Cyber-Physical Systems Applied to Stuxnet. *IEEE Trans. Dependable Secur. Comput.* **2018**, *15*, 2–13. [CrossRef]
5. Khraisat, A.; Gondal, I.; Vamplew, P.; Kamruzzaman, J. Survey of intrusion detection systems: Techniques, datasets and challenges. *Cybersecurity* **2019**, *2*, 20. [CrossRef]

6. Hasani, Z. Robust anomaly detection algorithms for real-time big data: Comparison of algorithms. In Proceedings of the 2017 6th Mediterranean Conference on Embedded Computing, Bar, Montenegro, 11–15 June 2017. [CrossRef]
7. Hawkins, J.; Ahmad, S. Why Neurons Have Thousands of Synapses, a Theory of Sequence Memory in Neocortex. *Front. Neural Circuits* **2016**, *10*, 23. [CrossRef]
8. Goh, J.; Adepu, S.; Junejo, K.N.; Mathur, A. A dataset to support research in the design of secure water treatment systems. In *Critical Information Infrastructures Security*; Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Bioinformatics); Springer: Berlin/Heidelberg, Germany, 2017; Volume 10242, pp. 88–99. [CrossRef]
9. Secure Water Treatment (SWAT) Testbed. Technical Report. ITrust, Centre for Research in Cyber Security, Singapore University of Technology and Design. 2020. Available online: https://itrust.sutd.edu.sg/itrust-labs-home/itrust-labs_swat/ (accessed on 15 September 2021).
10. Mishra, V.K.; Palleti, V.R.; Mathur, A. A modeling framework for critical infrastructure and its application in detecting cyber-attacks on a water distribution system. *Int. J. Crit. Infrastruct. Prot.* **2019**, *26*, 100298. [CrossRef]
11. Kozitsin, V.; Katser, I.; Lakontsev, D. Online forecasting and anomaly detection based on the ARIMA model. *Appl. Sci.* **2021**, *11*, 3194. [CrossRef]
12. Buczak, A.L.; Guven, E. A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection. *IEEE Commun. Surv. Tutor.* **2016**, *18*, 1153–1176. [CrossRef]
13. Inoue, J.; Yamagata, Y.; Chen, Y.; Poskitt, C.M.; Sun, J. Anomaly detection for a water treatment system using unsupervised machine learning. In Proceedings of the IEEE International Conference on Data Mining Workshops, ICDMW, New Orleans, LA, USA, 18–21 November 2017; pp. 1058–1065. [CrossRef]
14. Zhang, H.; Nian, K.; Coleman, T.F.; Li, Y. Spectral ranking and unsupervised feature selection for point, collective, and contextual anomaly detection. *Int. J. Data Sci. Anal.* **2020**, *9*, 57–75. Available online: https://link.springer.com/article/10.1007/s41060-018-0161-7#citeas (accessed on 8 April 2021). [CrossRef]
15. Abokifa, A.A.; Haddad, K.; Lo, C.S.; Biswas, P. Detection of Cyber-Physical Attacks on Water Distribution Systems via Principal Component Analysis and Artificial Neural Networks. In Proceedings of the World Environmental and Water Resources Congress, Sacramento, CA, USA, 21–25 May 2017; American Society of Civil Engineers (ASCE): Reston, VA, USA, 2017; p. 489.
16. Heigl, M.; Anand, K.A.; Urmann, A.; Fiala, D.; Schramm, M.; Hable, R. On the improvement of the isolation forest algorithm for outlier detection with streaming data. *Electronics* **2021**, *10*, 1534. [CrossRef]
17. Mirsky, Y.; Doitshman, T.; Elovici, Y.; Shabtai, A. Kitsune: An Ensemble of Autoencoders for Online Network Intrusion Detection. *arXiv* **2018**, arXiv:1802.09089.
18. Kravchik, M.; Shabtai, A. Detecting cyber attacks in industrial control systems using convolutional neural networks. In Proceedings of the 2018 Workshop on Cyber-Physical Systems Security and PrivaCy, Toronto, ON, Canada, 15–19 October 2018; pp. 72–83. [CrossRef]
19. Kravchik, M.; Shabtai, A. Efficient Cyber Attack Detection in Industrial Control Systems Using Lightweight Neural Networks and PCA. *IEEE Trans. Dependable Secur. Comput.* **2021**, *19*, 2179–2197. [CrossRef]
20. Gómez, Á.L.P.; Maimó, L.F.; Celdrán, A.H.; Clemente, F.J.G. MADICS: A methodology for anomaly detection in industrial control systems. *Symmetry* **2020**, *12*, 1583. [CrossRef]
21. Fährmann, D.; Damer, N.; Kirchbuchner, F.; Kuijper, A. Lightweight Long Short-Term Memory Variational Auto-Encoder for Multivariate Time Series Anomaly Detection in Industrial Control Systems. *Sensors* **2022**, *22*, 2886. [CrossRef] [PubMed]
22. Cui, Y.; Ahmad, S.; Hawkins, J. The HTM spatial pooler—A neocortical algorithm for online sparse distributed coding. *Front. Comput. Neurosci.* **2017**, *11*, 111. [CrossRef] [PubMed]
23. Li, D.; Chen, D.; Jin, B.; Shi, L.; Goh, J.; Ng, S.K. MAD-GAN: Multivariate Anomaly Detection for Time Series Data with Generative Adversarial Networks. In *Lecture Notes in Computer Science*; Including Subseries Lecture Notes in Artificial Intelligence and Bioinformatics; Springer Science and Business Media LLC: Berlin/Heidelberg, Germany, 2019; Volume 11730, pp. 703–716. [CrossRef]
24. Li, D.; Chen, D.; Goh, J.; Ng, S. Anomaly Detection with Generative Adversarial Networks for Multivariate Time Series. *arXiv* **2018**, arXiv:1809.04758.
25. Faber, K.; Pietron, M.; Zurek, D. Ensemble neuroevolution-based approach for multivariate time series anomaly detection. *Entropy* **2021**, *23*, 1466. [CrossRef]
26. Deng, A.; Hooi, B. Graph Neural Network-Based Anomaly Detection in Multivariate Time Series. In Proceedings of the 35th AAAI Conference on Artificial Intelligence, AAAI 2021, Virtually, 2–9 February 2021; Volume 5A, pp. 4027–4035. [CrossRef]
27. Lin, Q.; Verwer, S.; Adepu, S.; Mathur, A. TABOR: A graphical model-based approach for anomaly detection in industrial control systems. In Proceedings of the ASIA CCS '18: ACM Asia Conference on Computer and Communications Security, Incheon, Republic of Korea, 4 June 2018; pp. 525–536. [CrossRef]

28. Audibert, J.; Michiardi, P.; Guyard, F.; Marti, S.; Zuluaga, M.A. USAD: Unsupervised Anomaly Detection on Multivariate Time Series. In Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Virtually, 6–10 July 2020; pp. 3395–3404. [CrossRef]
29. Macas, M.; Wu, C.; Fuertes, W. An Attention-Based Deep Generative Model for Anomaly Detection in Industrial Control Systems. *arXiv* **2024**, arXiv:2405.05277. [CrossRef]
30. Shalyga, D.; Filonov, P.; Lavrentyev, A. Anomaly Detection for Water Treatment System Based on Neural Network with Automatic Architecture Optimization. *arXiv* **2018**, arXiv:1807.07282.
31. Adepu, S.; Mathur, A. Assessing the Effectiveness of Attack Detection at a Hackfest on Industrial Control Systems. *IEEE Trans. Sustain. Comput.* **2021**, *6*, 231–244. [CrossRef]
32. Hawkins, J.; Ahmad, S.; Lavin, A. Biological and Machine Intelligence. Release 0.4. 2020. Available online: https://Numenta.Com/Resources/Biological-and-Machine-Intelligence/ (accessed on 8 April 2022).
33. Cui, Y.; Ahmad, S.; Hawkins, J. Continuous online sequence learning with an unsupervised neural network model. *Neural Comput.* **2016**, *28*, 2474–2504. [CrossRef] [PubMed]
34. Ahmad, S.; Lavin, A.; Purdy, S.; Agha, Z. Unsupervised real-time anomaly detection for streaming data. *Neurocomputing* **2017**, *262*, 134–147. [CrossRef]
35. Sorhue, U.G.; Mmereole, F.U.C.; Irikefe-Ekeke, E.P.; Nwakwo, S.U. Evaluating Real-time Anomaly Detection Algorithms—The Numenta Anomaly Benchmark. *Niger. J. Genet.* **2014**, *28*, 34–37. [CrossRef]
36. Rodriguez, M.A.; Kotagiri, R.; Buyya, R. Detecting performance anomalies in scientific workflows using hierarchical temporal memory. *Future Gener. Comput. Syst.* **2018**, *88*, 624–635. [CrossRef]
37. Wang, C.; Zhao, Z.; Gong, L.; Zhu, L.; Liu, Z.; Cheng, X. A Distributed Anomaly Detection System for In-Vehicle Network Using HTM. *IEEE Access* **2018**, *6*, 9091–9098. [CrossRef]
38. Zyarah, A.M.; Kudithipudi, D. Neuromorphic Architecture for the Hierarchical Temporal Memory. *IEEE Trans. Emerg. Top. Comput. Intell.* **2019**, *3*, 4–14. [CrossRef]
39. Shoeleh, F.; Erfani, M.; Le, D.P.; Ghorbani, A.A. Ensemble of hierarchical temporal memory for anomaly detection. In Proceedings of the 2020 IEEE 7th International Conference on Data Science and Advanced Analytics, DSAA 2020, Sydney, NSW, Australia, 6–9 October 2020; pp. 50–59. [CrossRef]
40. Sekh, A.A.; Dogra, D.P.; Kar, S.; Roy, P.P.; Prasad, D.K. ELM-HTM guided bio-inspired unsupervised learning for anomalous trajectory classification. *Cogn. Syst. Res.* **2020**, *63*, 30–41. [CrossRef]
41. Barua, A.; Muthirayan, D.; PKhargonekar, P.; Al Faruque, M.A. Hierarchical Temporal Memory based One-pass Learning for Real-Time Anomaly Detection and Simultaneous Data Prediction in Smart Grids. *IEEE Trans. Dependable Secur. Comput.* **2020**, *19*, 1770–1782. [CrossRef]
42. Lavin, S.; Hawkins, A.J. Sparse Distributed Representations. Available online: https://www.numenta.com/assets/pdf/biological-and-machine-intelligence/BaMI-SDR.pdf (accessed on 8 April 2022).
43. HTM.core Implementation of Hierarchical Temporal Memory. 2019. Available online: https://github.com/htm-community/htm.core/ (accessed on 15 July 2022).
44. Purdy, S. Encoding Data for HTM Systems. *arXiv* **2016**, arXiv:1602.05925.
45. Rachkovskij, D.A.; Kussul, E.M. Binding and Normalization of Binary Sparse Distributed Representations by Context-Dependent Thinning. *Neural Comput.* **2001**, *13*, 411–452. [CrossRef]
46. Laiho, M.; Poikonen, J.H.; Kanerva, P.; Lehtonen, E. High-dimensional computing with sparse vectors. In Proceedings of the 2015 IEEE Biomedical Circuits and Systems Conference (BioCAS), Atlanta, GA, USA, 22–24 October 2015; pp. 1–4. [CrossRef]
47. Natrella, M. Nist/sematech e-handbook of statistical methods. In Proceedings of the World Environmental and Water Resources Congress 2017, Sacramento, CA, USA, 21–25 May 2017; Volume 49, pp. 676–691.
48. Luo, Y.; Xiao, Y.; Cheng, L.; Peng, G.; Yao, D.D. Deep Learning-based Anomaly Detection in Cyber-physical Systems: Progress and Opportunities. *ACM Comput. Surv.* **2021**, *54*, 106. [CrossRef]
49. Pinto, A.; Herrera, L.C.; Donoso, Y.; Gutierrez, J.A. Survey on Intrusion Detection Systems Based on Machine Learning Techniques for the Protection of Critical Infrastructure. *Sensors* **2023**, *23*, 2415. [CrossRef]
50. Roman, M.; Avi, M. The use of HTM and TSSE Encoder for Online Anomaly Detection in Industrial Cyber-Physical Systems. Presented at the Design of Cyber-Secure Water Plants (DCS-Water'24), Buford, GA, USA, 23–24 April 2024.