Neural Computing and Applications
https://doi.org/10.1007/s00521-019-04363-x

EMERGING TRENDS OF APPLIED NEURAL COMPUTATION - E_TRAINCO

# Gryphon: a semi-supervised anomaly detection system based on one-class evolving spiking neural network

Konstantinos Demertzis[1] · Lazaros Iliadis[1] · Ilias Bougoudis[2]

Received: 17 September 2018 / Accepted: 19 July 2019


**Abstract**
The backbone of the economy, security and sustainability of a state is inseparably linked to the security of its critical infrastructure. Critical infrastructures define goods, systems or subsystems that are essential to maintain the vital functions of society, health, physical protection, security plus economic and social well-being of citizens. The digital security of critical infrastructures is a very important priority for the well-being of every country, especially nowadays, because of the direct threats dictated by the current international conjuncture and due to the emerging interactions or interconnections developed between the National Critical Infrastructures, internationally. The aim of this research is the development and testing of an *Anomaly Detection intelligent* algorithm that has the advantage to run very fast with a small portion of the available data and to perform equally well with the existing approaches. Such a system must be characterized by high efficiency and very fast execution. Thus, we present the Gryphon advanced intelligence system. Gryphon is a Semi-*Supervised Unary Anomaly Detection System* for big industrial data which is employing an *evolving Spiking Neural Network* (eSNN) *One-Class Classifier* (eSNN-OCC). This machine learning algorithm corresponds to a model capable of detecting very fast and efficiently, divergent behaviors and abnormalities associated with cyberattacks, which are known as *Advanced Persistent Threat* (APT). The training process is performed on data related to the normal function of a critical infrastructure.

**Keywords** Critical infrastructure · Industrial control systems · SCADA · Advanced persistent threat · Evolving spiking neural network · One-class classification · Anomaly detection · Semi-supervised learning

## 1 Introduction

### 1.1 Critical infrastructure protection

Protecting the critical infrastructures (CRIN) of a country is of the utmost importance. Any kind of potential CRIN failure caused either by terrorist attack or due to systems' weaknesses can cause complex and dynamic interdependencies, and it can have domino effects with incalculable consequences [1]. The most significant CRIN are related to the following sectors: Energy, Information Technology (IT), Transportation, National Defense, Government Infrastructure and Industry [1, 2]. Automation and remote control are today the most important methods by which critical infrastructures can improve their productivity and the quality of provided services [1–3]. From this point of view, automation and efficient management of industrial IT systems, require the adoption of secure, reliable and sophisticated network control devices that operate with precision. Supervisory Control and Data Acquisition (SCADA) systems and sensors are typical automated control devices, used in control loops [4]. These interconnected systems are active devices operating on real-time industrial networks, and they allow remote monitoring and

✉ Konstantinos Demertzis
kdemertz@fmenr.duth.gr

Lazaros Iliadis
liliadis@civil.duth.gr

Ilias Bougoudis
ibougoudis@iup.physik.uni-bremen.de

[1] School of Engineering, Department of Civil Engineering, Faculty of Mathematics Programming and General Courses, Democritus University of Thrace, Kimmeria, Xanthi, Greece

[2] Institute of Environmental Physics, DOAS Group, University of Bremen, Otto-Hahn Allee 1, 28359 Bremen, Germany

Published online: 27 July 2019

process control, even in cases where devices are remotely distributed. Critical infrastructures are exposed to new risks due to vulnerabilities in communications and information technologies (*malware*, *spyware*, *ransomware*), which is also greatly enhanced by the heterogeneity of these systems [5].

Safety of the CRIN systems requires a very fast anomaly detection process, where every second counts. The fact that such systems have to run complex real-time algorithms in order to analyze (in real time) huge amounts of data (big data) makes them relatively slow. Thus, we do not only need algorithms with high percentages of correct classifications, but we need security algorithms to run as fast as possible. The basic aim of this research is the development of an intelligent model/system that can achieve the optimal compromise between accuracy and speed. Thus, we have developed a system that performs equally well by using a small portion of the data. This makes it equally reliable with well-established methods, but much faster, thus more efficient.

It is important to emphasize that attacks against CRIN are typically identified as APT. Cyber criminals are fully familiar with specialized methods and tools to exploit unknown vulnerabilities to the public "*zero-days*" [5]. A zero-day vulnerability, at its core, is a flaw. It is an unknown exploit in the wild that exposes a vulnerability in software or hardware and can create complicated problems well before anyone realizes something is wrong. In fact, a zero-day exploit leaves no opportunity for detection [6]. The cyber criminals try not to be perceived without considering time, and in most of the cases they are highly capable, organized, funded and they have significant incentives [6].

Given the growing complexity of threats, the ever-changing environment and the need for critical infrastructures, such attacks could, in the worst-case scenario, cause massive economic damages through data leakage or misuse, even the loss of lives of innocent, directly or indirectly. This is another motivation for the adoption of intelligent solutions to prevent, detect and deal with threats or anomalies under the conditions and operating parameters of CRIN [7].

Also, given the passive operation of traditional security systems, which in most cases are unable to detect serious threats such as APT attacks, alternative, more active and more effective security methods, is needed. Our team is trying to solve such complex digital security problems and has previously proposed many innovative artificial intelligence applications [8–15].

### 1.2 Unary classification and anomaly detection

The unary classification method (UCC), also known as one-class classification (OCC), implements intelligent categorization of cases belonging to a specific class, among an existing set of records [16]. OCC is learning from a training set that contains only records of one specific class. Typically, these methods aim to implement classification models in which the negative class is absent, either because the missing class is not sampled, or due to the fact that it is difficult to do so. This mode of operation, in which classifiers are required to determine effectively and reliably the boundaries of the class separation only based on the knowledge of the positive class, is a particularly complex problem of machine learning. When only data from the target class is available, the classifier is trained to receive target objects and to reject the ones that deviate significantly. Finally, it should be noted that the basic concept in OCC problem solving is the reverse of the generalization that is being pursued in other machine learning problems [17]. Particularly, it is intended that the parameter setting is fully defined, even if this exponentially increases the complexity of the classifier, provided it is able to correctly classify the target data.

In this sense, the UCC is the most appropriate approach for detecting abnormalities and identifying patterns or trends, in a set of data that display divergent behaviors than expected. UCC achieves high levels of successful detection, while maintaining low false error rates (false alarm) [18].

### 1.3 Semi-supervised unary anomaly detection

In supervised detection of abnormalities, the applied models are binary [17, 18] (normal versus anomalous behavior). On the other hand, in unsupervised anomaly detection, training data are not required, and predictions are mainly based on the fact that normal snapshots are more than the extreme ones in the dataset. If this reasoning does not apply, then the techniques have a large error rate [17, 18].

In the case of semi-supervised (SESU) detection of anomalies, the techniques assume that a set of training data has been declared only as normal. The usual approach in this case is to construct a model for the set to respond to normal behavior and then to apply it in order to determine the anomalies in the testing data. More generally, it should be stressed that the success of the SESU learning depends on some key assumptions imposed by each model or algorithm. Thus, each case is dependent on these initial assumptions or peculiarities. This fact makes the method

one of the most rational machine learning approaches that can be applied in real-world cases [19].

## 2 Literature review

Several approaches have been studied in [19–22] which use different OCC methods [23]. In [24], Huang and his team used one-class extreme learning machines (ELM) for liver tumor detection. Zhu et al. [25] have introduced the data and feature ensemble ELM (DFEN-ELM). They have extended Huang's initial work by using one-class kernel-based ELM. Juszczak [26] has defined OCC as class descriptors, capable to learn restricted domains in a multidimensional pattern space, using primarily just a positive set of examples. Luo et al. [27] have proposed a cost-sensitive one-class classification support vector machine (*OCC-SVM*) algorithm for intrusion detection. Their experiments have suggested that giving different cost or importance to system users than to processes results in higher performance in intrusion detection. Tax and Laskov [28] have developed a SVM-based one-class classifier, for incremental and online learning. Manevitz and Yousef [29, 30] have employed *OC-SVM* and *Auto-Associative Neural Networks* for retrieving the interested document from the internet in the presence of positive examples only. Shieh and Kamm [31] have proposed a kernel density estimation method that assigns weights to the training data objects, such that the outliers get the least weights, whereas the positive-class members get higher weights, for creating bootstrap samples.

Qian and Sherif [32] have applied autonomous computing technology, to monitor SCADA system performance. Their approach proactively estimates upcoming attacks for a given system model of a physical infrastructure. Soupionis et al. [33] have proposed a combinatorial method for automatic detection and classification of faults and cyberattacks occurring on the power grid system when there is limited data from the power grid nodes due to cyber implications.

In addition, Tao et al. [34] have described the network attack knowledge, based on the theory of the factor expression of knowledge. They have studied the formal knowledge theory of SCADA network from the factor state space and equivalence partitioning. This approach utilizes the factor neural network (FNN) theory which contains high-level knowledge and quantitative reasoning, used to establish a predictive model including analytic FNN and analogous FNN. This model abstracts and builds an equivalent and corresponding network attack and a defense knowledge factors system. Finally, [35] has introduced a new *European Framework-7* project *Cockpit CI* (*Critical Infrastructure*) and roles of intelligent machine learning methods to prevent SCADA systems from cyberattacks.

## 3 Proposed framework

### 3.1 The Gryphon

APT attacks can undertake mechanical control, dynamic rearrangement of centrifugation or reprogramming of devices in order to accelerate or slow down their operations, leading the overall industrial equipment to destruction or permanent damage [36]. Industrial control systems (ICS), which are mostly used in critical infrastructures, use SCADA and distributed control systems (DCS) based on the programmable logic controller (PLC) [37]. These systems are active devices of industrial networks deployed in critical infrastructures. Successful completion of specialized activities, such as remote control or data recording, requires that all devices used be accurately and reliably controlled [37].

This research effort, proposes the development of an innovative computational intelligence algorithm, which significantly enhances critical infrastructure security mechanisms with minimal resource consumption. Specifically, it proposes an advanced anomaly detection framework, the Gryphon: a semi-supervised unary anomaly detection system for big industrial data. It is based on the evolving spiking neural network (eSNN-OCC) approach. It implements a special form of SESU learning, to categorize the obtained anomalies in SCADA critical infrastructure devices, which are the result of APT attacks.

The implementation of the Gryphon is an evolution of the spiking one-class anomaly detection framework (SOCCADF) approach that has been proposed by our research team in [38]. Additionally, the algorithmic approach has also evolved from the SESU learning system that we have previously proposed in [39]. In general, it is based on the literature known methods, such as those outlined below, which are optimally combined in a hybrid manner to create a comprehensive intelligent learning system. We have proven that this system optimally implements a decision rule that properly assigns labels (categorizes) for new unlabeled data.

### 3.2 Fuzzy c-means clustering

According to Zadeh [40–42] every element "$x$" of the Universe of discourse "$X$" belongs to a fuzzy set (FS) with a degree of membership in the closed interval [0,1]. Thus, the following function 1 is the mathematical foundation of a FS.

$$S = \{(x, \mu s(x)/\mu s : X\{[0,1] : x\}\mu s(x)\} \quad (1)$$

The *Fuzzy c-means* clustering algorithm initially gives random values to the cluster centers, and then it assigns all data points to all clusters with varying degrees of membership (DOM) by measuring the Euclidean distance. Then, the DOM of each data point to each cluster is estimated based on Eq. (2)

$$\mu_j(x_i) = \frac{\left(\frac{1}{d_{ji}}\right)^{\frac{1}{m-1}}}{\sum_{k=1}^{p}\left(\frac{1}{d_{ki}}\right)^{\frac{1}{m-1}}} \quad (2)$$

where $m$ is the *fuzzification parameter* with values in the interval [1.25, 2, 40]. The values of $m$ specify the degree of overlapping between the clusters. Then, the centers are estimated again. Equation (3) is used for the re-estimation of the values of new cluster centers [42].

$$c_j = \frac{\sum_i \left[\mu_j(x_i)\right]^m x_i}{\Sigma_i \left[\mu_j(x_i)\right]^m} \quad (3)$$

where $c_j$ is the center of the $j$th cluster with ($j$ = 1, 2…$p$) and $x_i$ is the $i$th point [42]. This is an iterative algorithm and the process is repeated till the centers are stabilized.

### 3.3 The evolving spiking neural network one-class classifier

The eSNNs are modular connectionist-based systems that evolve their structure and functionality in a continuous, self-organized, online, adaptive and interactive way based on incoming information [43]. Their topology is strictly feedforward, organized in several layers, and weight modification occurs on the connections between neurons of the existing layers.

The winner takes that all approaches are employed. According to this algorithm, only the weight of the first postsynaptic neuron to fire is updated. This kind of plasticity constitutes an underlying learning and information storage mechanism, and it possibly contributes to the development of neuronal circuits during brain development [44]. The synaptic activity plasticity rule (SAPR) is a temporally symmetric form of Konorski/Hebbian learning. The synaptic connection strength in SAPR is modified using an update function that takes advantage of the membrane potential of the postsynaptic neuron [45].

The proposed eSNNs intensify the importance of the spikes taking place in an earlier moment, whereas the neural plasticity is used to monitor the learning algorithm by using one-pass learning. In order to classify real-valued datasets, each data sample is mapped into a sequence of spikes using the rank-order population encoding (ROPE) technique. The topology of the developed eSNN is strictly feedforward, organized in several layers, and weight modification occurs on the connections between the neurons of the existing layers.

The ROPE method is an extension of the rank-order encoding and is alternative to the conventional rate coding scheme. It uses the order of firing neuron's inputs to encode information. It allows the mapping of vectors of real-valued elements into a sequence of spikes. Neurons are organized into neuronal maps which share the same synaptic weights. Whenever the synaptic weight of a neuron is modified, the same modification is applied to the entire population of neurons within the map. Inhibition is also present between each neuronal map. If a neuron spikes, it inhibits all the neurons in the other maps with neighboring positions. This prevents all the neurons from learning the same pattern. When propagating new information, neuronal activity is initially reset to zero. Then, as the propagation goes on, each time one of their inputs fire and neurons are progressively desensitized. This is making neuronal responses dependent upon the relative order of firing of the neuron's afferents [43]. The general architecture of an evolving spiking neural network is shown in Fig. 1. Real-valued vector elements are mapped into the time domain using rank-order population encoding based on Gaussian receptive fields. As a consequence of this transformation input, neurons emit spikes at pre-defined firing times, invoking the one-pass learning algorithm of the eSNN. The learning iteratively creates repositories of output neurons, one repository for each class. Here, a two-class problem is presented. Due to the evolving nature of the network, it is possible to accumulate knowledge as it becomes available, without the requirement of retraining with already learnt samples, which is one of the eSNN principles.

The topology of the proposed eSNN is strictly feedforward, organized in three layers (input layer, hidden

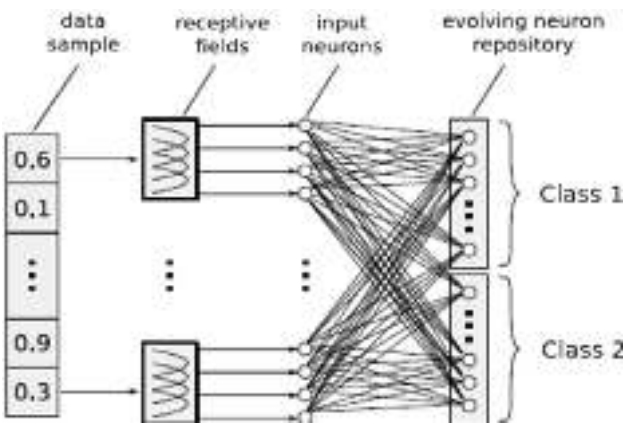


**Fig. 1** Evolving spiking neural network (eSNN) architecture [43]

evolving layer and output layer). The encoding is performed by employing the ROPE technique which uses 20 *Gaussian Receptive Fields* (GRF) per variable. The data are normalized to the interval [− 1, 1], and so the coverage of the Gaussians is determined by using $i_{\min} = -1$ and $i_{\max} = 1$. Each input variable is encoded independently by a group of one-dimensional GRF. The GRF of neuron $i$ is given by its center $\mu_i$ by Eq. (4) and width $\sigma$ by Eq. (5) [43].

$$\mu_i = I^n_{\min} + \frac{2i-3}{2}\frac{I^n_{\max} - I^n_{\min}}{M-2} \tag{4}$$

$$\sigma = \frac{1}{\beta}\frac{I^n_{\max} - I^n_{\min}}{M-2} \tag{5}$$

where $1 \leq \beta \leq 2$ and the parameter $\beta$ directly control the width of each GRF (see Fig. 2).

When a neuron reaches its threshold, it spikes and inhibits neurons at equivalent positions in the other maps so that only one neuron will respond at any location. Every spike triggers a time-based Hebbian-like learning rule that adjusts the synaptic weights. The eSNN uses one-pass learning (OPAL) method in the training process. The aim of the OPAL is to create a repository of trained output neurons during the presentation of training samples. After presenting a certain input sample to the network, the corresponding spike train is propagated through the eSNN which may result in the firing of certain output neurons. It is possible that no output neuron is activated and the network remains silent, which results in an undetermined classification result. If one or more output neurons have emitted a spike, the neuron with the shortest response time among all activated ones is determined. The label of this neuron is the classification result for the presented input.

For each training sample $i$ with class label $l$, a new output neuron is created which is fully connected to the previous layer, resulting in a real-valued weight vector $w^{(i)}$ with $w^{(i)}_j \in R$ denoting the connection between the presynaptic neuron $j$ and the created neuron $i$. In the next step, the input spikes are propagated through the network, and the value of weight $w^{(i)}_j$ is computed according to the order of spike transmission through a synapse [43]

$$j: \; w^{(i)}_j = (m_l)^{\text{order}(j)} \tag{6}$$

where $j$ is the *p*resynaptic neuron of $i$. Parameter $0 < m_l < 1$ is the modulation factor of the Thorpe neural model. Differently labeled output neurons may have different modulation factors $m_l$. Function order($j$) represents the rank of the spike emitted by neuron $j$. For example, a rank order($j$) would be assigned the 0 value, if neuron $j$ is the first among all presynaptic neurons of $i$ that emits a spike. In a similar fashion, the spikes of all presynaptic neurons are ranked and then used in the computation of the weights. The firing threshold $\theta^{(i)}$ of the created neuron $i$ is defined as the fraction $c_l \in R$, $0 < c_l < 1$, of the maximal possible potential [43]

$$u^{(i)}_{\max}: \; \theta^{(i)} \leftarrow c_l u^{(i)}_{\max} \tag{7}$$

$$u^{(i)}_{\max} \leftarrow \sum_j w^{(i)}_j (m_l)^{\text{order}(j)} \tag{8}$$

The weight vector of the trained neuron is compared to the weights corresponding to neurons already stored in the repository. Two neurons are considered too "similar" if the minimal *Euclidean* distance between their weight vectors is smaller than a specified similarity threshold $s_l$. It should be mentioned that the eSNN object uses an optimal similarity

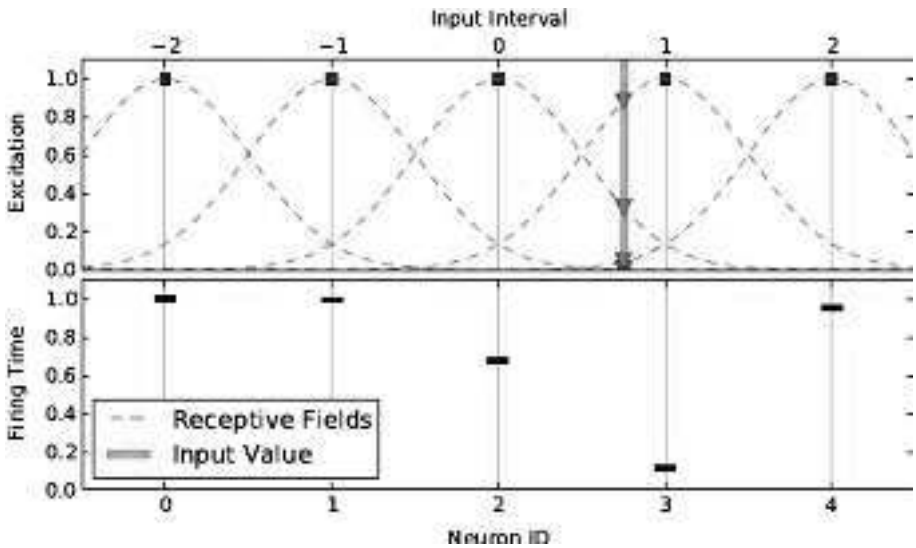


**Fig. 2** Population encoding based on Gaussian receptive fields [43]

threshold $s = 0.6$. Both the firing thresholds and the weight vectors were merged according to Eqs. (9) and (10) [43]

$$w_j^{(k)} \leftarrow \frac{w_j^{(i)} + N w_j^{(k)}}{1+N} \tag{9}$$

$$\theta^{(k)} \leftarrow \frac{\theta^{(i)} + N\theta^{(k)}}{1+N} \tag{10}$$

The integer $N$ denotes the number of samples previously used to update neuron $k$. The merging is implemented as the average of the connection weights, and of the two firing thresholds. After merging, the trained neuron $i$ is discarded and the next sample is processed. If no other neuron in the repository is similar to the trained neuron $i$, the neuron $i$ is added to the repository as a new output. The above procedure is described in detail in Algorithm 1 [43].

**Algorithm 1.** Training an evolving Spiking Neural Network (eSNN)

**Require:** $m_l$, $s_l$, $c_l$ **for a class label** l ∈ L
1: initialize neuron repository $R_l = \{\}$
2: **for all** samples $X^{(i)}$ belonging to class l **do**
3: $w_j^{(i)} \leftarrow (m_l)^{\text{order(j)}}$, ∀ $j$ | $j$ pre-synaptic neuron of i
4: $u_{max}^{(i)} \leftarrow \sum_j w_j^{(i)} (m_l)^{\text{order(j)}}$
5: $\theta^{(i)} \leftarrow c_l u_{max}^{(i)}$
6: **if** $\min(d(w^{(i)}, w^{(k)})) < s_l$, $w^{(k)} \in R_l$ **then**
7: $w^{(k)} \leftarrow$ merge $w^{(i)}$ and $w^{(k)}$ according to Equation 9
8: $\theta^{(k)} \leftarrow$ merge $\theta^{(i)}$ and $\theta^{(k)}$ according to Equation 10
9: **else**
10: $R_l \leftarrow R_l \cup \{w^{(i)}\}$
11: **end if**
12: **end for**

All of the eSNN parameters included in this search space are optimized according to the versatile quantum-inspired evolutionary algorithm (vQEA) [43].

### 3.4 Collective classification

Collective classification [46] is a combinatorial optimization problem, in which we are given a set of nodes, $V = \{V_1, \ldots, V_n\}$ and a neighborhood set of node $N = \{N_1, \ldots, N_n\}$, where $N_i \subseteq V \backslash \{Vi\}$, which describes the underlying network structure. Each node in $V$ is a random variable that can take a value from an appropriate domain. $V$ is further divided into two sets of nodes: $X$, the nodes for which we know the correct values (observed variables) and $Y$ the nodes whose values need to be determined. Our task is to label the nodes $Y_i \in Y$ with one value belonging to a small labels' dataset $L = \{L_1, \ldots, L_q\}$; In this case, we will use the shorthand $y_i$ to denote the label of node $Y_i$ [46].

### 3.5 The proposed Gryphon algorithm

An important innovation of the proposed semi-supervised method is the easy validation of the classification, performed on an unknown set of values. This is achieved with the employment of actual measurable factors.

Let us consider a supervised learning problem with a training set comprising of $N$ samples of the form $\{X, Y\} = \{x_i, y_i\}_{i=1}^{N}$, where $x_i \in R^{n_i}$, $y_i$ is a $n_o$-dimensional binary vector with only one input (corresponding to the class where $x_i$ belongs) equal to a multidimensional classification process, where $n_i$ and $n_o$ are the dimensions of input and output, respectively. Generally, unclassified data provide useful information for exploring the structure of the general dataset, whereas the respective classified data contribute to the learning process.

The proposed Gryphon algorithm initially performs SESU clustering using a small number of labeled data, in order to categorize a number of records in clusters. The actual process is based on a measure of similarity (MESI). Typically, every cluster is assigned a characteristic center of gravity value (CGV). During iterations, the values of the centers are adjusted and when the CGV stabilize, the iterations are terminated. Then, the "*classes to clusters*" evaluation method that employs a SESU approach is used to verify the final clustering result. Initially a minimum data sample related to the obtained clusters (labeled data) is used. The remaining unlabeled data which ignore the class attribute are used to provide useful information related to the structure of the overall dataset, as they dynamically modulate and adjust the classes based on the values that belong to each cluster.

The test set uses the labeled data to test the performance of the algorithm by calculating the classification error, based on known assignments. This option evaluates whether the selected clusters match the specified class of data. The overall process is presented in Algorithm 2.

**Algorithm 2.** The Gryphon Algorithm

**Inputs**: Input labeled data $\mathbf{D_l}$, clusters of the labeled data $\mathbf{L_l}$ and a set of unlabeled data $\mathbf{D_u}$, let $\mathbf{N} = |D_l|$ and $\mathbf{M} = |D_u|$, $\mathbf{F} \in R$

**Step 1**: *% Train the classifier*
Train the classifier eSNN-OCC using $\mathbf{D_l}$ to produce the model $\mathbf{M_l}$
Use the model $\mathbf{M_l}$ to "pre-label" all the examples from $\mathbf{D_u}$
Assign weights of 1.0 to every example in $\mathbf{D_l}$
and of $\mathbf{F} \times \mathbf{N/M}$ to all the examples in $\mathbf{D_u}$
Merge the two sets $\mathbf{D_l}$ and $\mathbf{D_u}$ into $\mathbf{D}$

**Step 2**: *% Make a prediction*
**for a class label** $l \in L_d$
initialize neuron repository $R_l = \{\}$
**for all** samples $X^{(i)}$ belonging to class l **do**
$w_j^{(i)} \leftarrow (m_l)^{order(j)}$, $\forall$ j | j pre-synaptic neuron of i
$u_{max}^{(i)} \leftarrow \sum_j w_j^{(i)} (m_l)^{order(j)}$
$\theta^{(i)} \leftarrow c_l u_{max}^{(i)}$
**if** $\min(d(w^{(i)}, w^{(k)})) < s_l$, $w^{(k)} \in R_l$ **then**
$w^{(k)} \leftarrow$ merge $w^{(i)}$ and $w^{(k)}$ according to $u_{max}^{(i)} : \theta^{(i)} \leftarrow c_l u_{max}^{(i)}$
$\theta^{(k)} \leftarrow$ merge $\theta^{(i)}$ and $\theta^{(k)}$ according to $u_{max}^{(i)} \leftarrow \sum_j w_j^{(i)} (m_l)^{order(j)}$
**else**
$R_l \leftarrow R_l \cup \{w^{(i)}\}$
**end if**
**end for**

**Output**: Classified data vectors

**Step 3**: *% Initialization of clusters*
Identify the discrete number of clusters based on $\mathbf{L_l}$
For every cluster, create matrices with the mean and standard deviation of all $\mathbf{D_l}$

**Step 4**: *% Calculate the new centers of the clusters*
For every cluster, recreate these matrices, based on the testing data $\mathbf{D_u}$
Calculate a variable, based on the formula below:
x =(1./(2*pi*ns.^2)).*exp(-((test-nm).^2)./(2.*sn.^2))
where ns is the new standard deviation matrix, nm is the new mean matrix and test $\mathbf{D_u}$
Sum all these variables for each cluster

**Step 5**: *% Calculate the winner cluster for each record*
For every testing data $\mathbf{D_u}$, find the minimum value of the summary calculated before.
*% Calculate the fuzzy membership values for every cluster for every record*
For every testing data $\mathbf{D_u}$ and for every class, divide the mean matrix with the sum of the values calculated before (normalization probability – membership value)

**Outputs**: Winner cluster for each testing data $\mathbf{D_u}$, $\mathbf{C_u}$ and fuzzy membership values for every cluster for every testing data $\mathbf{D_u}$, $\mathbf{F\_M\_V_{u,j}}$ (j the number of clusters)

**Step 6**: *% Validation of the clustering process*
Repeat Steps 1 – 3 from the previous part, only this time from $\mathbf{D_u} \rightarrow \mathbf{D_l}$, using $\mathbf{C_u}$ as labels

**Output**: Winner cluster for each testing data $\mathbf{D_l}$, $\mathbf{L2_l}$

**Step 7**: *% Comparison process*
For every initially labeled data $\mathbf{D_l}$:
Compare the initial label $\mathbf{L_l}$ with $\mathbf{L2_l}$
Create confusion matrix based on these comparisons

**Step 8**: *% Iteration process*
Repeat Steps 5 - 6 for every $\mathbf{D_w}$ of $\mathbf{D_u}$

*% Generalization of the amount of the extreme cases, based on the fuzzy membership values*

**Inputs:** The winner class for every record ($\mathbf{C_u}$) and the fuzzy membership values for each record ($\mathbf{F\_M\_V_{u,j}}$)

**Step 9**: *% Order process*
For every record:
*If max(F_M_V$_{u,j}$) = A AND F_M_V$_{u,A}$ – max2(F_M_V$_{u,j}$) <= threshold,* then
*% max2(F_M_V$_{u,k}$) = k, the second biggest membership value*
Change the winner class for this record to k ($\mathbf{C_u = k}$)

**Outputs**: Updated winner cluster for each record $\mathbf{C_u}$

## 3.6 Threshold criteria

The OCC process is much more difficult than a traditional binary or multiclass classifier, as it is trained to accept target objects and to reject the ones that have significant deviation [18]. Minimizing the errors is also a difficult process, because in this type of categorization, cross-validation is unavailable since there is no data from the other classes [17]. Finally, it should be stressed that one-class problem-solving technique is inverse to the generalization approaches that are pursued in other machine learning problems, as it tends to provide a fully defined configuration of parameters. This can exponentially increase the complexity of the classifier, trying to correctly

classify target data. The more complex the model, the smaller the rank range in the target data range, and the less likely it is for the outliers to be categorized correctly. In practice, one can create a complex model by setting all its possible parameters without being at risk from overfitting [17].

As it has been pointed out, the threshold for separating the classes is the most important and critical factor for the success of the OCC method [17, 18]. This paper proposes a *heuristic* algorithm (based on a well-established scientific approach) that determines the threshold, considering that the training set contains only positive samples. The training phase uses a distance function $d$, between the objects and the target category. (When calculating a value/class for a new instance, they compute *Euclidean* distances between this instance and the training instances to make a decision.) The Euclidean metric (and distance magnitude) is that which corresponds to the distance between any two points in space corresponding to the length of a straight line drawn between them. The determination of the threshold $\theta$ for the separation of the classes (normal or outlier) is determined by discarding a set of training samples (TRSA), most of which diverge from the target class. In this way, the classifier is strengthened. Even when all samples are correctly labeled, the rejection of a small but representative portion of TRSA helps the classifier to learn the most representative one [17, 18].

The proposed heuristic algorithm implements conditionally the interquartile range (IQR) method [47] with some serious assumptions.

In the introduced method, quadrants do not divide the data into four equal parts (quarters), but the interquartile range of the median value, which is considered to be the second quadrant denoted as $Q_2$, is implemented to include the intermediate 80% of the observations. This is clearly shown in Fig. 3.

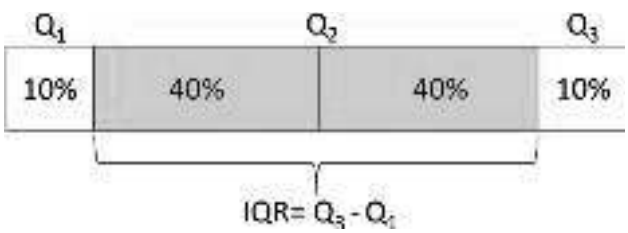


**Fig. 3** Graphical display of IQR method

The pseudocode of the optimal threshold determination is presented in Algorithm 3.

Algorithm 3: Optimal Threshold Determination

***Optimal Threshold***:
*1: Calculate the error using Euclidean distance between actual and predicted on each training data;*
*2: Arrange the error in decreasing order;*
*3: Set the threshold at rejection of 10% most erroneous data and 10% of fittest data;*

This method ensures that the small sample of data used in training is the characteristic of the normal SCADA operation on critical infrastructure systems. The proposed algorithm that determines the classification threshold is presented below in a natural language form.

## 4 Datasets

Appropriate datasets were chosen that closely simulate ICS communication and transaction data. They were used for the development and evaluation of the proposed model. Contained preprocessed network transaction data and preprocessed to strip lower layer transmission data were also considered (e.g., TCP, MAC) namely [48]:

- The *water_tower_dataset* includes 23 independent parameters and 236,179 instances, from which 172,415 normal and 63,764 outliers. Totally, 86,315 normal instances were used in the training phase (*water_train_dataset*), whereas the rest 86,100 normal instances and 63,764 outliers comprised the *water_test_dataset*. The introduced Gryphon algorithm used only the 10% of the *water_train_dataset* for the training process (8500 normal instances).
- The *gas_dataset* includes 26 independent features and 97,019 instances, from which 61,156 normal and 35,863 outliers. The training of the algorithm was performed on the *gas_train_dataset* that contains 30,499 normal instances, whereas the rest 30,657 normal instances and 35,863 outliers belong to the *gas_test_dataset*. The proposed Gryphon considered only 10% of the *gas_train_dataset* in the training process (3000 normal instances).
- Finally, the *electric_dataset* includes 128 independent variables with 146,519 instances, from which 90,856 are normal and 55,663 are outliers. The training was performed based on the *electric_train_dataset* comprising 45,402 normal instances, whereas the rest 45,454

normal and the 55,663 outliers belong to the *electric_test_dataset*. The proposed Gryphon used only 10% of the *electric_train_dataset* in the training process (4500 normal instances).

These sets contain data logs from a *Gas Pipeline*, a *Lab Scale Water Tower* and a *Lab Scale Electric Transmission* system. The logs include flagged network transactions during the normal operation of specific ICSs, as well as transactions during 35 different cyberattacks. In addition to the logs, our data include normal behavior measurements, as well as abnormalities detected during attacks that were simulated in a virtual ICS environment. Details regarding the dataset, their choice and assessment can be found in [48].

## 5 Results

In multiclass classification, all of the indices presented below [17] should be calculated in a one versus all approach [49].

The magnitude of misclassifications is indicated by the false positive (FP) and false negative (FN) indices appearing in the confusion matrix. A FP is the number of cases where we wrongfully receive a positive result, and the FN is exactly the opposite. On the other hand, the true positive (TP) is the number of records where we correctly receive a positive result. The true negative (TN) is defined, respectively.

The *True Positive rate* (TPR) also known as *Sensitivity*, the *True Negative rate* also known as *Specificity* (TNR), and the *Total Accuracy* (TA) are defined by using Eqs. (11), (12) and (13), respectively [17, 49]:

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{11}$$

$$\mathrm{TNR} = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FP}} \tag{12}$$

$$\mathrm{TA} = \frac{\mathrm{TP} + \mathrm{TN}}{N} \tag{13}$$

The precision (PRE), the recall (REC) and the F-Score indices are defined as in Eqs. (14), (15) and (16), respectively [17, 49]:

$$\mathrm{PRE} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \tag{14}$$

$$\mathrm{REC} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{15}$$

$$\text{F-Score} = 2 \times \frac{\mathrm{PRE} \times \mathrm{REC}}{\mathrm{PRE} + \mathrm{REC}} \tag{16}$$

More specifically, since it is a unary classification, the true positives and the false positives are output by the classification execution. The true negatives are indirectly estimated by us by counting the number of cases that were not classified to belong to the one and only existing class. The number of cases that were correctly not assigned to the one existing class determines the true negative index, whereas the number of records that were wrongfully not assigned to this class determines the false negative index.

In the OCC cases, the probability density of the positive order is known, which means that during training, only the number of positive-class objects that are not categorized correctly (FN) can be minimized. In the absence of examples and distribution of samples belonging to other categories, it is not possible to estimate the number of objects of other classes that were poorly categorized as false positive by the OCC classifier. Given that TP + FN = 1 and FP + TN = 1, an OCC algorithm includes information only for TP and FN and no information about FP and TN [17, 49]. Table 1 presents an extensive comparison (for of all three datasets) with the SOCCADF approach [38] that was developed and proposed by our research team in a previous survey on the same subject, with the OCC support vector machines (OCC-SVM) and OCC *Combining Density and Class Probability Estimation* (OCC-CD/CPE) learning.

As shown in the above table, other algorithms [38] appear to have a slightly better performance across all datasets, compared to the Gryphon which is proposed herein. This fact does not detract in any case from the value of the proposed framework. On the contrary, it is a strong demonstration of the Gryphon's potential, considering the objective difficulties raised in this research. Specifically, it is obvious that Gryphon implements the anomalies recognition as successfully and rationally as possible, considering the fact that it is trained only with 10% of the others approaches data vectors [38]. It is important to recall that the Gryphon's training was based on a totally specific subset of the initial training set (which originated from the 80% of the actual data) thus limiting the learning algorithm to a very specific vector distribution.

Thus, in the proposed Gryphon algorithm, we face a very appealing compromise where we are using much less data, which means faster response and less complexity, whereas the efficiency and accuracy are practically the same.

## 6 Discussion and conclusions

This research paper discusses an innovative, reliable, low-demand and highly effective anomaly detection system, employing computational intelligence principles. The Gryphon: a semi-supervised unary anomaly detection system is an evolution of the spiking one-class anomaly

**Table 1** Comparison between algorithms

| Classifier | Classification accuracy and performance metrics | | | | |
|---|---|---|---|---|---|
| | Total accuracy | Precision | Recall | F-score | ROC area |
| *water_tower_dataset* | | | | | |
| GRYPHON | 98.03% | 0.980 | 0.980 | 0.980 | 0.980 |
| SOCCADF | 98.08% | 0.981 | 0.981 | 0.981 | 0.994 |
| OCC-SVM | 98.01% | 0.980 | 0.980 | 0.980 | 0.995 |
| OCC-CD/CPE | 96.75% | 0.975 | 0.975 | 0.975 | 0.980 |
| *gas_dataset* | | | | | |
| GRYPHON | 97.68% | 0.975 | 0.975 | 0.970 | 0.980 |
| SOCCADF | 98.82% | 0.988 | 0.988 | 0.988 | 0.995 |
| OCC-SVM | 97.98% | 0.980 | 0.980 | 0.980 | 0.990 |
| OCC-CD/CPE | 95.67% | 0.960 | 0.960 | 0.960 | 0.975 |
| *electric_dataset* | | | | | |
| GRYPHON | 96.92% | 0.970 | 0.969 | 0.969 | 0.985 |
| SOCCADF | 98.30% | 0.983 | 0.983 | 0.983 | 0.999 |
| OCC-SVM | 97.63% | 0.978 | 0.978 | 0.978 | 0.990 |
| OCC-CD/CPE | 97.02% | 0.970 | 0.970 | 0.970 | 0.985 |

detection framework algorithm (SOCCADF) that was proposed in a previous research effort of our team [38]. A comparative in-depth analysis between these two approaches has been carried out.

Gryphon uses the sophisticated eSNN classification algorithm to identify deviations in the way ICS works, which in most cases stem from cyberattacks.

The implementation of the Gryphon was based on the unary classification philosophy. Its intelligent algorithm was trained with a dataset related to normal ICS behaviors and it was tested in defining the abnormalities of these systems. An important innovation of Gryphon is the use of eSNN which have proven to be capable, of solving a multidimensional and complex IT security problem. The eSNN simulate the functioning of biological brain cells in a most realistic mode. They manage to rationally model spatiotemporal cases, as the signals are transported with time pulses. Apart from the duration, the frequency of time pulses between neurons is an important parameter. This creates potential for a fully defined configuration of the model, based on target data, resulting in high-accuracy classification.

Another important aspect is the use of SESU as it is the most realistic approach of timely systems for which it is impossible to parameterize all possible classes of operation. Additionally, the use of artificial intelligence in real-time analysis of industrial equipment greatly enhances the active defense mechanisms of critical infrastructures. Another key issue is the development of the *class separation threshold* used in training. This threshold has emerged after extensive research into the way ICS work and after comparisons and tests of their inherent behavior boundaries, in order to determine their classification in the potential states (normal or outliers).

This system, and more generally the active security philosophy, greatly enhances the control of critical infrastructures, which are the main objective of advanced attacks.

The performance of the proposed system was tested in three multidimensional datasets of high complexity, which resulted from extensive research into the operation of ICS (SCADA, DCS and PLC). The high precision results that have emerged greatly enhance the general methodology followed, although the degree of difficulty and realism that has been added has created extremely multifactorial issues of thorough investigation and reflection. It is obvious that the proposed framework, which simplifies the process and minimizes the cost and running time, is an important prerequisite for the establishment of an effective risk reduction and CRIN protection system. Proposals for the development and future improvements of this system should focus on further optimizing the parameters of the eSNN algorithm, aiming to achieve an even more efficient, accurate and quicker classification process.

It would be essential to extend this approach to be used for the classification of data streams with online learning methods. Finally, an additional element that could be studied in the direction of future extension is related to the employment of self-improvement methods with redefinition of the system's parameters in a meta-learning way, so that it can fully automate the localization process of APT attacks.

## Compliance with ethical standards

**Conflict of interest** The authors certify that they have NO affiliations with or involvement in any organization or entity with any financial interest (such as honoraria; educational grants; participation in speakers' bureaus; membership, employment, consultancies, stock ownership, or other equity interest; and expert testimony or patent-licensing arrangements), or non-financial interest (such as personal or professional relationships, affiliations, knowledge or beliefs) in the subject matter or materials discussed in this manuscript.

## References


1. Hurst W, Merabti M, Fergus P (2014) A survey of critical infrastructure security. In: Butts J, Shenoi S (eds) Critical infrastructure protection VIII. ICCIP 2014. IFIP Advances in information and communication technology, vol 441. Springer, Berlin
2. Yusufovna F, Alisherovich F, Choi M, Cho E, Abdurashidovich F, Kim T (2009) Research on critical infrastructures and critical information infrastructures. In: Proceedings of the symposium on bio-inspired learning and intelligent systems for security, pp 97–101
3. Hurst W, Merabti M, Fergus P (2013) Behavioral observation for critical infrastructure security support. In: Proceedings of the seventh IEEE European modeling symposium, pp 36–41
4. Wang C, Fang L, Dai Y (2010) A simulation environment for SCADA security analysis and assessment. In: Proceedings of the international conference on measuring technology and mechatronics automation, vol 1, pp 342–347
5. Walker J, Williams B, Skelton G (2010) Cyber security for emergency management. In: Proceedings of the IEEE international conference on technologies for homeland security, pp 476–480
6. Jeun I, Lee Y, Won D (2012) A practical study on advanced persistent threats. In: Kim T et al (eds) Computer applications for security, control and system engineering. Communications in computer and information science, vol 339. Springer, Berlin
7. Demertzis K, Iliadis LS, Anezakis V-D (2018) An innovative soft computing system for smart energy grids cybersecurity. In: Santamouris M (ed) Advances in building energy research. Taylor & Francis, London, pp 1–22
8. Demertzis K, Iliadis L (2014) A hybrid network anomaly and intrusion detection approach based on evolving spiking neural network classification. In: Sideridis A, Kardasiadou Z, Yialouris C, Zorkadis V (eds) E-democracy, security, privacy and trust in a digital world. e-Democracy 2013. Communications in computer and information science, vol 441. Springer, Cham
9. Demertzis K, Iliadis L (2014) Evolving computational intelligence system for malware detection. In: Iliadis L, Papazoglou M, Pohl K (eds) Advanced information systems engineering workshops. CAiSE 2014. Lecture notes in business information processing, vol 178. Springer, Cham. https://doi.org/10.1007/978-3-319-07869-4_30
10. Demertzis K, Iliadis L (2014) Bio-inspired hybrid artificial intelligence framework for cyber security. In: Daras N, Rassias M (eds) Computation, cryptography, and network security. Springer, Cham
11. Demertzis K, Iliadis L (2014d) Bio-inspired hybrid intelligent method for detecting android malware. In: Iliadis L, Papazoglou M, Pohl K (eds) Advanced information systems engineering workshops. CAiSE 2014. Lecture notes in business information processing, vol 178. Springer, Cham
12. Demertzis K, Iliadis L (2015a) Evolving smart URL filter in a zone-based policy firewall for detecting algorithmically generated malicious domains. In: Gammerman A, Vovk V, Papadopoulos H (eds) Statistical learning and data sciences. SLDS 2015. Lecture notes in computer science, vol 9047. Springer, Cham
13. Demertzis K, Iliadis L (2015b) SAME: an intelligent anti-malware extension for android ART virtual machine. In: Núñez M, Nguyen N, Camacho D, Trawiński B (eds) Computational collective intelligence. Lecture notes in computer science, vol 9330. Springer, Cham
14. Demertzis K, Iliadis L (2017) Computational intelligence anti-malware framework for android OS. Vietnam J Comput Sci 4:245. https://doi.org/10.1007/s40595-017-0095-3
15. Demertzis K, Iliadis L (2016) Ladon: a cyber-threat bio-inspired intelligence management system. J Appl Math Bioinform 6(3):45–64
16. Shehroz SK, Madden MG (2014) One-class classification: taxonomy of study and review of techniques. Knowl Eng Rev. https://doi.org/10.1017/S026988891300043X
17. Mao J, Jain AK, Duin PW (2000) Statistical pattern recognition: a review. IEEE Trans Pattern Anal Mach Intell 22(1):4–37
18. Ban T, Abe S (2006) Implementing multi-class classifiers by one-class classification methods. In: International joint conference on neural networks, pp 327–332
19. Munoz-Mari J, Bovolo F, Gomez-Chova L, Bruzzone L, Camp-Valls G (2010) Semisupervised one-class support vector machines for classification of remote sensing data. IEEE Trans Geosci Remote Sens 48(8):3188–3197. https://doi.org/10.1109/TGRS.2010.2045764
20. Roth V (2006) Kernel fisher discriminants for outlier detection. Neural Comput 18(4):942–960
21. Abe N, Zadrozny B, Langford J (2006) Outlier detection by active learning. In: Proceedings of the 12th ACM SIGKDD international conference on knowledge discovery and data mining, pp 767–772. ACM Press, New York
22. Tax DMJ, Muller KR (2004) A consistency-based model selection for one-class classification. In: Proceedings of the 17th international conference on pattern recognition (ICPR 2004), vol 3, pp 363–366
23. Wilk T, Wozniak M (2012) Soft computing methods applied to combination of one-class classifiers. Neurocomputing 75:185–193
24. Huang W, Li N, Lin Z, Huang GB, Zong W, Zhou J, Duan Y (2013) Liver tumor detection and segmentation using kernel based extreme learning machine. In: IEEE conference on Engineering in Medicine and Biology Society (EMBC), pp 3662–3665, 3–7 July
25. Zhu WZ (2015) Data and feature mixed ensemble based extreme learning machine for medical object detection and segmentation. Multimed Tools Appl 75:2815–2837
26. Juszczak P (2006) Learning to recognize. A study on one-class classification and active learning. Ph.D. thesis, Delft University of Technology
27. Luo J, Ding L, Pan Z, Ni G, Hu G (2007) Research on cost-sensitive learning in one-class anomaly detection algorithms. In: Xiao B, Yang LT, Ma J, Muller-Schloer C, Hua Y (eds) Autonomic and trusted computing, vol 4610. Lecture notes in computer science. Springer, Berlin, pp 259–268
28. Tax DMJ, Laskov P (2003) Online SVM learning: from classification to data description and back. In: IEEE 13th workshop on neural networks for signal processing, (NNSP'03), pp 499–508. IEEE
29. Manevitz L, Yousef M (2001) One-class SVM for document classification. J Mach Learn Res 2:139–154
30. Manevitz L, Yousef M (2007) One-class document classification via neural networks. Neurocomputing 70:1466–1481

31. Shieh AD, Kamm DF (2009) Ensembles of one class support vector machines, vol 5519. Lecture notes in computer science. Springer, Berlin, pp 181–190
32. Chen Q, Abdelwahed S (2013) A model-based approach to self-protection in computing system. In: Proceeding CAC '13 of the ACM cloud and autonomic computing conference, article No. 16
33. Soupionis Y, Ntalampiras S, Giannopoulos G (2016) Vol 8985 of the book series Lecture notes in computer science. https://doi.org/10.1007/978-3-319-31664-2_29
34. Tao X, Renmu H, Peng W, Dongjie X (2004) Applications of data mining technique for power system transient stability prediction. Proc IEEE Electr Util Deregul Restruct Power Technol 1:389–392
35. Yasakethu SLP, Jiang J (2013) Intrusion detection via machine learning for SCADA system protection, learning and development ltd. In: Proceedings of the 1st international symposium for ICS and SCADA cyber security research
36. Weiss J (2003) Current status of cybersecurity of control systems. In: Presentation to Georgia Tech protective relay conference
37. Boyer SA (2010) SCADA: supervisory control and data acquisition, 4th edn. International Society of Automation, Research Triangle Park
38. Demertzis K, Iliadis L, Spartalis S (2017) A spiking one-class anomaly detection framework for cyber-security on industrial control systems. In: Boracchi G, Iliadis L, Jayne C, Likas A (eds) Engineering applications of neural networks. EANN 2017. Communications in computer and information science, vol 744. Springer, Cham
39. Bougoudis I, Demertzis K, Iliadis L, Anezakis VD, Papaleonidas A (2016) Semi-supervised hybrid modeling of atmospheric pollution in urban centers. In: Proceedings engineering applications of neural networks. EANN 2016. Communications in computer and information science, vol 629. Springer
40. Kecman V (2001) Learning and soft computing. MIT Press, Cambridge
41. Iliadis L (2007) Intelligent systems and application in risk estimation. In: Stamoulis A (eds) Thessaloniki, Greece. ISBN: 978-960-6741-33-3
42. Iliadis L, Papaleonidas A (2016) Computational intelligence an intelligent agents. In: Tziolas A (eds) Thessaloniki, Greece. ISBN: 978-960-418-601-3
43. Schliebs S, Kasabov N (2013) Evolving spiking neural network—a survey. Evol Syst 4:87. https://doi.org/10.1007/s12530-013-9074-9
44. Sjostrom J, Gerstner W (2010) Spike-timing dependent plasticity. In: Scholarpedia 5.2. Revision 142314, p 1362
45. Swiercz W, Swiercz W, Cios KJ, Staley K, Kurgan L, Accurso F, Sagel S (2006) A new synaptic plasticity rule for networks of spiking neurons. IEEE Trans Neural Netw 17(1):94–105
46. Sen P, Namata G, Bilgic M, Getoor L, Galligher B, Rad ET (2008) Collective classification in network data. Adv Artif Intell 29(3):93–106
47. Zwillinger D, Kokoska S (2000) CRC standard probability and statistics tables and formulae. CRC Press, Boca Raton
48. Morris TH, Thornton Z, Turnipseed I (2015) Industrial control system simulation and data logging for intrusion detection system research. Int J Netw Secur (IJNS) 17(2):174–188
49. Fawcett T (2006) An introduction to ROC analysis. Pattern Recognit Lett 27(8):861–874. https://doi.org/10.1016/j.patrec.2005.10.010