Applied Soft Computing 72 (2018) 79–89

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Semi-supervised learning based distributed attack detection framework for IoT

Shailendra Rathore, Jong Hyuk Park*

Department of Computer Science and Engineering, Seoul National University of Science and Technology, (SeoulTech), Seoul 01811, Republic of Korea

ARTICLE INFO

Article history:
Received 20 November 2017
Received in revised form 11 May 2018
Accepted 22 May 2018
Available online 1 July 2018

Keywords:
Internet of Things
Fog computing
Machine learning
Cyber security
Security attack detection

ABSTRACT

Alongside the development of Internet of Things (IoT), security attacks are also increasing day by day. A number of centralized attack detection mechanisms have been proposed to detect attacks in IoT, wherein an attack detection system is deployed at the central point in the network that collects data from the network and classifies it as "attack" or "normal" using a supervised machine learning algorithm. Note, however, that these mechanisms have failed to achieve significant results due to the distinct requirements of IoT devices, such as scalability, distribution, resource limitations, and low latency. Moreover, the application of supervised machine learning for classification needs a significant amount of labeled data. In this paper, we introduce a fog-based attack detection framework that relies on the fog computing paradigm and a newly proposed ELM-based Semi-supervised Fuzzy C-Means (ESFCM) method. As an extension of cloud computing, fog computing enables attack detection at the network edge and supports distributed attack detection. The ESFCM method uses a semi-supervised fuzzy c-means algorithm to handle the labeled data issue and an Extreme Learning Machine (ELM) algorithm to provide good generalization performance at a faster detection rate. The evaluation was performed on the NSL-KDD dataset, demonstrating that the proposed framework achieved better performance than the centralized attack detection framework. More specifically, it recorded a lower detection time of 11 milliseconds and an accuracy rate of 86.53%.

1. Introduction

The Internet of Things (IoT) is now widely used in various domains including smart buildings, power grids, entertainment, transportation, and health care. It is forecast to play a significant role in future technical revolutions, and its utilization is likely to increase exponentially over the coming years [1,2]. Due to the increasing number of IoT devices and the massive amount of data associated with them, the issue of security in the IoT has been raised. Security in the IoT implies the need for the defense of IoT applications and the IoT infrastructure [3]. Many IoT devices can be easily targeted by intrusion because they are connected with outside resources at the network layer, and they do not have proper security defense. As such, an attacker can compromise the network layer and obtain control over an IoT device, which can then be used maliciously, or it can compromise other nearby devices connected to it. According to a report from Statista [4], the number of devices connected through the Internet is expected to reach 75.44 billion worldwide by 2025. The increasing number of IoT devices will provide many opportunities for attackers to compromise them through malicious emails, collusion attacks, and denial of service attacks among many other types of attack. The HP report [5] stated that 70% of IoT devices are subject to various network attacks that exploit various vulnerabilities, such as encryption and password security. Thus, network attacks are now emerging as a key barrier to the more widespread adoption of IoT services, and they can be detected at the network layers of the IoT framework [6]. In the KDD99 dataset and its recent version NSL-KDD [7], attacks at the network layer are classified into four major types: (a) Denial of Service (DoS), wherein an attacker tries to make access to a service or machine unavailable to the legitimate user; (b) Probe, wherein an attacker attempts to obtain information of the targeted network by scanning the network and host activities; (c) User to Root (U2R), wherein attackers exploit various traditional attack techniques such as stolen credentials and malware infection to escalate their privilege from limited privilege to root or super user access, and; (d) Remote to Local (R2L), wherein an attacker can perform imitation of local users and obtain access to the targeted machine.

The existing mechanism for attack detection on wireless network works on the centralized cloud, which cannot satisfy the

* Corresponding author.
E-mail address: jhpark1@seoultech.ac.kr (J.H. Park).

https://doi.org/10.1016/j.asoc.2018.05.049
1568-4946/

distinct requirements of the IoT, such as scalability, distribution, resource limitations, and low latency, to name but a few [8]. In the IoT, a number of communication and control operations are performed among a diverse set of devices. It supports the intelligent processing and analysis of data and decision making in an autonomous manner by linking such devices as data processing links communication devices and sensors. On the other hand, cloud computing works as a sort of front end in the IoT that allows end users to use the entire range of services supported over the internet to carry out their ordinary computing operations. It also provides high-performance, reliability, and ubiquity to the IoT. Nevertheless, due to communication implications and its geographically centralized architecture, cloud computing in the IoT inefficient to support task that needs costly computation, storage and very low latency. Such tasks involve cloud-based attack detection in the IoT, where a centralized attack detection system performs poorly due to the overheads incurred by storing and processing data from an ever larger number of devices. Moreover, it delivers a higher detection time due to the long distance of an IoT device from a centralized attack detection system. This implies that the existing cloud-based attack detection mechanism cannot solve the attack detection problem in the IoT. Nowadays, fog computing is being investigated as a type of distributed intelligence computing that could overcome the limitations of cloud computing. As an extension of cloud computing, fog computing enables cloud-things service at the network edge; this means that communication and data processing should be performed nearer to the data sources [9]. It supports an encouraging methodology to handle the geographical distribution and low-latency desired in IoT. It addresses the issue of resource limitation in IoT such as high computation, storage, control, and network bandwidth by offloading the fog nodes near the network. This further escalates the efficiency and efficacy of IoT applications. Similar to other services [10], attack detection in IoT could be carried out at the fog layer, where each fog node is responsible for detecting attacks on IoT devices associated with it. Thus, costly computation and storage from IoT devices are offloaded using distributed attack detection at the fog layer. Attack detection in IoT can be classified into two major categories: signature and anomaly-based detection. Each category has its own benefits and limitations. In signature-based detection, the attack on an IoT device is identified by collecting certain data from the device and comparing these data with a set of patterns or rules called signature. On the other hand, anomaly-based detection builds a model containing samples of normal behaviors and considers deviation from the model to identify suspicious behavior or attack on a device [11]. Note, however, that these approaches do not support the detection of zero day attacks. The major problem is how to detect an attack whose signature is not available in the set of predefined patterns or rules.

The KDDCUP'99 [12] dataset was generated from the DARPA98 network traffic in 1999 and a well-known baseline dataset employed in the International Knowledge Discovery in Databases (KDD) competition. In literature, this dataset has been widely employed in the area of IoT security to evaluate the performance of anomaly-based attack detection in IoT [13]. Furthermore, several machine learning approaches that may be unsupervised or supervised have been applied to provide better accuracy for IoT attack detection. The supervised machine learning approach uses labeled data wherein each instance has a label belonging to a particular class of attack. Many supervised machine learning methods – such as support vector machine, neural network, k-nearest neighbor, and deep learning – have been widely applied to attack detection in IoT [13–15]. These methods generate a trained model using labeled data that is further used to predict the correct label of a new unseen instance. Many researchers have reported several benefits and difficulties related to supervised machine learning. As the key difficulties of using supervised machine learning, it requires a huge amount of labeled instances. For the evaluation of IoT attack detection, only KDDCUP'99 is the available labeled dataset. Note, however, that several new types of attacks have been identified and developed. Thus, for new types of attacks, this dataset provides less accurate results, and it is considered obsolete. The KDDCUP'99 dataset is widely used by many researchers because it is publicly available for attack detection and it enables the extraction of useful information. Its drawbacks notwithstanding, supervised machine learning provides the benefits of achieving better accuracy to detect existing attacks whose sample is available in the labeled dataset. The unsupervised machine learning approach employs unlabeled data for classification. Clustering [16] is the most widely used unsupervised machine learning approach wherein instances in the unlabeled dataset are grouped into different clusters based on the similarity among them. Instances in the same cluster are considered to have similar properties or characteristics and are classified under the same class. As the drawback of unsupervised learning, it provides lower accuracy results for classification due to the manual assignments of cluster number. Nevertheless, it has better performance than supervised machine learning in terms of detecting new attack instances, and it is also considered to be more effective for attack detection in IoT. On the other hand, a semi-supervised machine learning approach that combines supervised and unsupervised machine learning approaches has also been proposed. It uses both labeled and unlabeled data for training and extracts the intra-structure of the dataset. Semi-supervised Fuzzy C-Means [17], currently the most widely used semi-supervised learning approach, enhances the performance of Fuzzy C-Means by using different types of prior knowledge in the form of constraints or labels. Considering the recent development in the area of fog computing and machine learning, the main objective of our research is to introduce an innovative fog-based attack detection framework that can perform attack detection in distributed manner and capable of detecting new attacks in IoT. To sum up, this paper offers the following contributions:

- This paper provides design and implementation of fog-based attack detection framework for IoT, which satisfies the distinct requirements of IoT such as distribution, resource limitations, and lower detection time.
- We propose an ELM-based Semi-Supervised Fuzzy C-Means (ESFCM) method that integrates a Semi-supervised Fuzzy C-Means with the Extreme Learning Machine (ELM) classifier to support efficient attack detection in IoT.
- We evaluate the performance of the proposed framework by comparing it with the centralized cloud-based framework.
- To demonstrate the effectiveness of the proposed ESFCM method for attack detection in IoT, this paper compares it with traditional machine learning methods in terms of standard measures.

The rest of the paper is organized as follows: in Section 2, we discuss the various existing approaches to detecting security attacks in IoT; in Section 3, we propose a Fog-based attack detection framework that introduces a fog computing paradigm and a novel ESFCM method for attack detection in IoT; in Section 4, we evaluate the performance of the proposed framework on the NSL-KDD dataset; finally, we conclude our paper in Section 5.

2. Related works

Alongside the development of IoT, many researchers have proposed their approaches to detecting security attacks in IoT. The existing approaches are mainly based on the study of various components of IoT to investigate suitable attack detection methods.

Hamed Haddad Pajouh et al. [18] proposed an activity detection model that relies on the dimension reduction and classification module to identify malicious activities, such as Remote to Local (R2L) and User to Root (U2R) attacks in IoT. They applied linear discriminate analysis and component analysis to reduce the dimension of dataset with lower features. The proposed model was evaluated using the two-tier classification module of K-Nearest Neighbor and Naïve Bayes algorithms over NSL-KDD dataset. Yair Meidan et al. [19] applied the supervised machine learning algorithm to identify unauthorized IoT devices. Xianghui Cao et al. [20] provided several security responses for the ghost attack, which relies on ZigBee networks on IoT devices. Kuan Zhang et al. [21] presented some defense mechanism for the Sybil attack, including mobile Sybil defense, behavior classification-based Sybil defense, and social graph-based Sybil defense. Pheeha Machaka et al. [22] used the Cumulative Sum algorithm to detect distributed denial of service attacks in IoT. Qian Chen et al. [23] proposed an automated approach to cyber security management in the IoT system. This approach provided an autonomous way of cyber-attack detection and reaction at the early stage. The authors demonstrate that the proposed approach can protect against unknown attacks in a Web-based application with little or no manual involvement. Qazi Mamoon Ashraf et al. [24] provided an analysis of various techniques to mitigate security attacks in IoT. Huansheng Ning et al. [25] introduced a hierarchical authentication approach to support the transmission of anonymous data in layered IoT networks. Lianbing Deng et al. [26] introduced a transfer learning-based intrusion detection approach for the IoT, wherein transfer learning was supported by principle component analysis and Fuzzy C-Means. The proposed approach relies on a centralized cloud architecture, and the experimental results show that this approach lowers the false positive rate and improves the detection rate for attack detection in the IoT. Rupinder Singh et al. [27] proposed an Advanced Hybrid Intrusion Detection System that combined a multilayer perceptron neural network with fuzzy logic to detect Sybil, wormhole, and hello flooding attacks in wireless sensor networks. The application of fuzzy logic supported the labeling of unlabeled datasets, which further reduced the efforts for dataset labeling and provided efficient attack detection. R. Vijayanand et al. [28] integrated multiple support vector machine classifiers to detect attacks in the advanced metering infrastructure of the smart grid.

In summary, existing approaches to attack detection in the IoT focus on the design and development of an intrusion detection system. However, such an intrusion detection system relies on a centralized cloud, which is less capable of supporting the unique requisites of the IoT environment such as, distribution, scalability, resource limitations, and low latency. Moreover, the existing approaches use a traditional supervised machine-learning algorithm, which requires a significant amount of labeled data for attack detection and is unable to detect zero day attacks. As such, it is necessary to design and develop a new approach to detecting attacks in the IoT that considers all aspects of present and future challenges.

3. Proposed Fog-based attack detection framework

Considering the limitation of existing approaches to attack detection in IoT, in this section, we propose a robust framework for performing attack detection in the IoT environment in a sophisticated manner. Among several security attacks in IoT, we focused on network security attacks as a use case to develop our framework. Typically, for network security attack detection in IoT, existing researches were carried out on intrusion detection data containing information on "legitimate" and "malware" connection types logged in to an IoT network. In a general wireless network, the attack detection system detects an attack by employing an "Anomaly-based scheme" or a "Signature-based scheme" on network data. The attack detection system is deployed at a certain point in the network, where it collects data from the network and makes a classification of "attack" or "normal". Other than the conventional scheme, machine learning methods are employed on the network data, and classification is carried out using the supervised machine learning algorithm. Nevertheless, this conventional scheme may not be appropriate for smart IoT systems, including those of smart cities [29], due to such issues as heterogeneity, distribution of IoT applications/services, and the unavailability of a significant amount of labeled data for the supervised learning algorithm. Thus, we propose a fog-based attack detection framework that uses a fog computing paradigm to support distributed attack detection and a newly proposed ESFCM method to handle the problem of labeled data in IoT.

3.1. Overview of the framework

In the IoT system, several devices are positioned at different localities with long distances among them. Compared to a regular wired or wireless system, the IoT system consists of a larger number of devices. Thus, a single attack detection system must have storage capability to store and process data from all the network devices and should provide quick response in a short amount of time. In this situation, the detection system will have poor performance in the IoT network due to the larger number of devices and long distance between them. As such, in the proposed framework shown in Fig. 1, attack detection is supported at the fog layer wherein each fog node is responsible for the detection of attacks occurring at all TCP/IP layers. Each TCP/IP layer at a fog node contains specific IoT devices connected to the fog node through the base station – for example, the network layer has routers, whereas the transport layer consists of switches. Each fog node detects attacks by processing data obtained from the IoT devices connected to that node only. Thus, the detection load is shared to several fog nodes at the fog layer, as a result of which the detection performance improves. Moreover, detection of attacks is performed close to the IoT device, thereby decreasing the detection time.

3.2. ELM-based semi-supervised fuzzy c-means method

In this section, we first discuss the Semi-supervised Fuzzy C-Means (SFCM) algorithm and introduce a fast learning mechanism for a single hidden layer feedforward neural network (SLFN) called ELM, and then propose an ESFCM method for attack detection in IoT.

Overview of the SFCM algorithm Clustering analysis is a prevalent unsupervised method of grouping unlabeled data [30]. In this method, a given dataset is partitioned into clusters so that, according to some measure, the data points that are most similar belong to the same cluster, and those that are different belong to different clusters. Fuzzy C-Means (FCM) is one of the widely used techniques for unsupervised learning. FCM allows the assignment of a data point to different classes with several degrees of membership and offers additional conceptual improvement in clustering. Thus, it provides a more reasonable way to treat patterns, and it can also identify eventual outliers [31].

Let $A = \{a_1, a_2, \ldots, a_N\}$, where $a_i \in R^d$ is a dataset of dimension d and size N. In FCM, the patterns are partitioned by minimizing an objective function. J.C.Bezdek et al. [32] described the objective function as follows:

$$\min J_M = \sum_{i=1}^{P}\sum_{j=1}^{N} u_{ij}^{M} \|a_j - v_i\|^2 \quad (1)$$

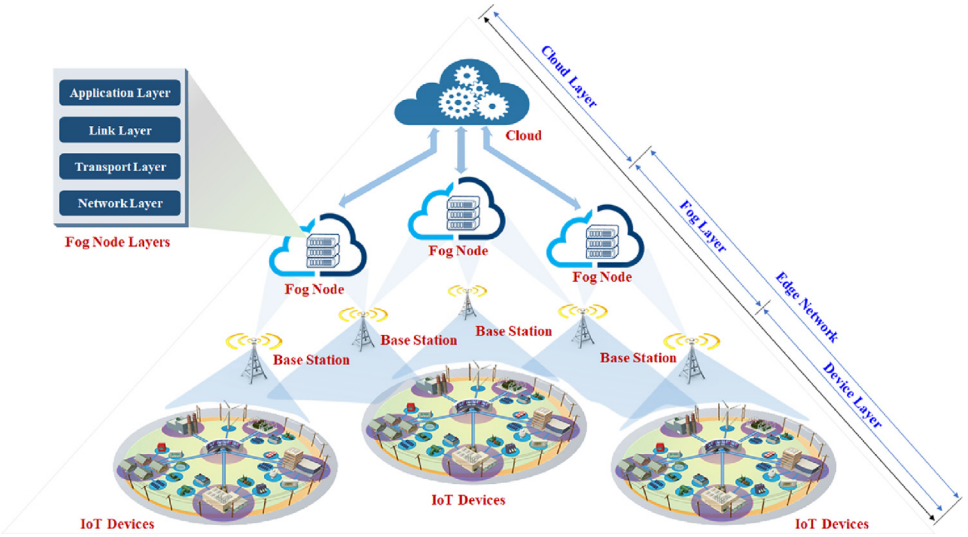

Fig. 1. Architecture of proposed framework.

where P represents the number of classes $\{1, 2, \ldots, P\}$. Each class in P is associated with a prototype $\{v_i\}$, and the set of prototypes associated with all classes is represented by $V = \{v_1, v_2, \ldots, v_P\}$. u_{ij} represents the degree of membership for data point a_j to class i and works under two conditions: $0 \le u_{ij} \le 1$ and $\sum_{i=1}^{P} u_{ij} = 1$. The degree of membership for all data points in dataset A to classes $\{1, 2, \ldots, P\}$ can be denoted as partition matrix $U = \{u_{ij}\}_{P \times N}$, and M is the fuzziness degree of matrix U. $d_{ij} = \|a_j - v_i\|$ denotes the distance between data point a_j and prototype v_i.

The semi-supervised clustering method is based on the concept of using different types of prior knowledge to enhance clustering performance. Such prior knowledge has been considered in the form of constraints or labels by earlier researches [33,34]. In this paper, we consider prior knowledge in the form of labels to design a semi-supervised model. We use the SFCM method introduced in [17], where the objective function of the standard FCM method is extended to capture the visible and hidden data space structure. The extended objective function involves two terms: the first term signifies the original objective function to discover the hidden space structure, and the second term considers labeled data to define the visible data structure. Formally, let the given labeled dataset $A' = \{a'_1, a'_2, \ldots, a'_{N'}\}$, where $a'_i \in R^d$, and $L(a'_i) \in \{1, 2, \ldots, P\}$ denote the label of data; the extended objective function of SFCM can be defined as below [17]:

$$\min J_M = \sum_{i=1}^{P}\sum_{j=1}^{N} u_{ij}^{M} \|a_j - v_i\|^2 + \sum_{i=1}^{P}\sum_{j=1}^{N'} u'^{M}_{ij} \|a'_j - v_i\|^2 \tag{2}$$

where u'_{ij} and u_{ij} refer to the degrees of membership for labeled data point a'_j and unlabeled data point a_j to class i, respectively, satisfying $0 \le u'_{ij} \le 1$, $\sum_{i=1}^{P} u'_{ij} = 1$, and $0 \le u_{ij} \le 1$ and $\sum_{i=1}^{P} u_{ij} = 1$. For labeled data, a supplementary condition must be satisfied.

$$u'_{ij} \ge u'_{kj}, ThickSpace \forall k \in \{1, 2, \ldots, P\}/\{i\}, ThickSpace j = 1, 2, \ldots, N' \quad \text{if } L(a'_j = i) \tag{3}$$

For optimization, the iteration equations can be obtained by using Lagrange multiplier method as follows: For data prototype

$$v_i = \frac{\sum_{j=1}^{N} u_{ij}^{M} a_j + \sum_{j=1}^{N'} u'^{M}_{ij} a'_j}{\sum_{j=1}^{N} u_{ij}^{M} + \sum_{j=1}^{N'} u' M_{ij}} \tag{4}$$

For an unlabeled data point a_j

$$u_{ij} = \frac{1}{\sum_{i=1}^{P} \left(\frac{d_{ij}}{d_{lj}}\right)^{2/(M-1)}} \tag{5}$$

For labeled data point a'_j

$$u'_{ij} = \frac{1}{\sum_{i=1}^{P} \left(\frac{d'_{ij}}{d'_{lj}}\right)^{2/(M-1)}} ThickSpace \text{if } L(a'_i) = i \tag{6}$$

$$u'_{kj} = \min \left\{ u'_{ij} \cdot \frac{1 - u'_{ij}}{\sum_{i=1, l \ne i}^{P} \left(\frac{d'_{kj}}{d'_{lj}}\right)^{2/(M-1)}} \right\}, \quad \forall k \in \{1, 2, \ldots\ldots, P\}/\{i\}, \quad \text{if } L(a'_i) = i \tag{7}$$

ELM algorithm ELM [35] is originally introduced to perform training of single hidden-layer feedforward neural networks (SLFNs). It relies on empirical risk minimization wherein the weight at the output layer of SLFNs is determined by randomly assigning the weights and biases to the input layer and hidden layer, respectively. In comparison to other tradition algorithms, such as Support Vector Machine, it provides better generalization performance at faster learning speed [36].

For M arbitrary instances (x_i, y_i), where $x_i = [x_{i1}, x_{i2}, ldots, x_{ip}]^T \in R^p$ and $y_i = [y_{i1}, y_{i2}, ldots, y_{iq}]^T \in R^q$, the trained model of standard SLFNs with activation function $q(x)$ and H hidden layer can be mathematically represented as follows:

$$\sum_{j=1}^{H} \alpha_j q(w_j \cdot x_i + b_j) = z_i \quad (i = 1, 2, \ldots\ldots, M) \tag{8}$$

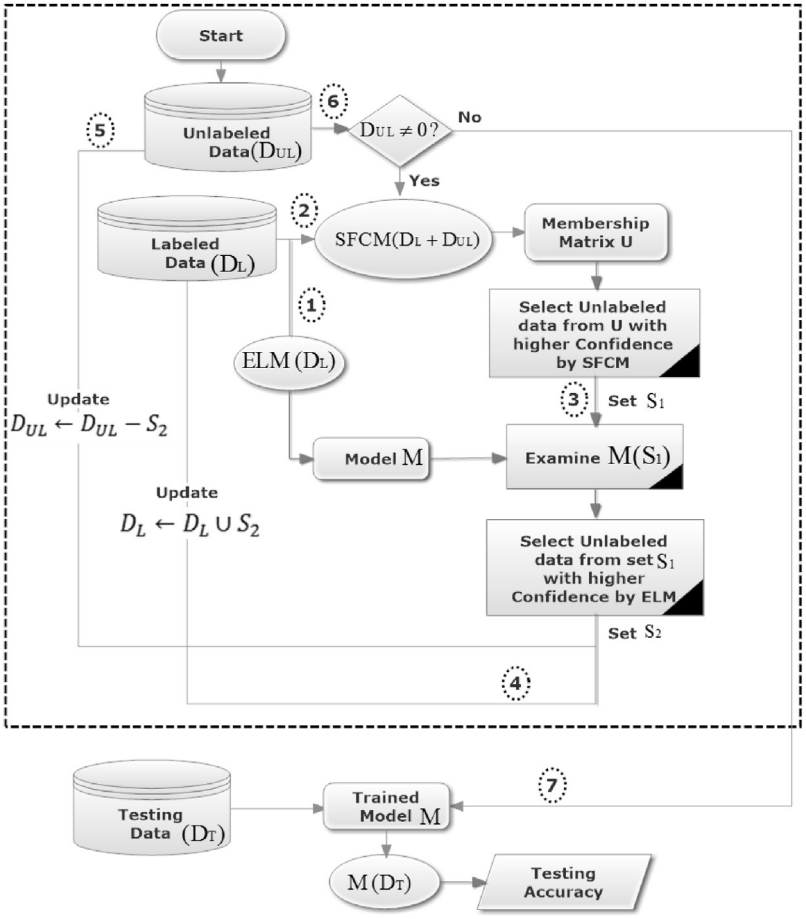

Fig. 2. Proposed EFCM method.

where $w_j = [w_{j1}, w_{j2}, ldots, w_{jp}]^T$ is the weight vector that links input nodes to the jth hidden node and $\alpha_j = [\alpha_{j1}, \alpha_{j2}, ldots, \alpha_{jq}]$ is the weight vector that links output nodes to the jth hidden nodes; $z_i = [z_{i1}, z_{i2}, ldots, z_{iq}]^T$ refers to the ith output vector, and b_j denotes the value of threshold for the jth hidden node.

The M instances can be approximated by SLFNs with zero error, meaning $\sum_{i=1}^{M} \|z_i - y_i\| = 0$ if α_j, w_j, and b_j exist such that

$$\sum_{j=1}^{H} \alpha_j q(w_j \cdot x_i + b_j) = y_i \quad (i = 1, 2, ldots, M) \tag{9}$$

The equation above can be represented compactly as below

$$Q\alpha = Y \tag{10}$$

where

$$Q = \begin{Bmatrix} q(w_1 x_1 + b_1) & \dots & q(w_H x_1 + b_H) \\ \vdots & \dots & \vdots \\ q(w_1 x_M + b_1) & \dots & q(w_H x_M + b_H) \end{Bmatrix}_{M \times H} \tag{11}$$

$$\alpha = \begin{Bmatrix} \alpha_1 \\ \vdots \\ \alpha_H \end{Bmatrix}_{H \times P} \quad and \quad Y = \begin{Bmatrix} y_1 \\ \vdots \\ y_M \end{Bmatrix}_{M \times P} \tag{12}$$

Q represents the hidden layer output matrix of SLFNs. The smallest least square solution for Eq. (10) is:

$$\alpha = Q^{+}Y \tag{13}$$

where Q^{+} is the Moore–Penrose generalized inverse of matrix Q. Then the output function of ELM can be modeled as follows:

$$F(x) = h(x)\alpha = h(x)Q^{+}Y \tag{14}$$

Proposed method In this subsection, we propose an ESFCM classification method wherein the SFCM method is integrated with the ELM classifier. The proposed method is described in Algorithm 1. The Confidence thresholds of SFCM (γ_1) and ELM (γ_2) are used as input parameters to define the integration of both methods. For the given unlabeled dataset D_{UL} and the labeled dataset D_L, the integration provides the labeling of unlabeled datasets to labeled datasets, which further helps prepare a trained model to classify the unknown instances in the testing dataset D_T. The overall workflow of the algorithm is shown in Fig. 2 and described as follows: **Step 1** The algorithm first trains the ELM classifier on labeled dataset D_L to generate the trained model (M). The ELM classifier uses the hidden nodes L and the sigmoid activation function $q(y) = 1/1 + e^{-y}$ as a hidden node output function.

Step 2 The SFCM (as described earlier) is employed over both the unlabeled data D_{UL} and the labeled data D_L to learn the underlying data space structures, and returns the matrix U of membership degree for the unlabeled dataset D_{UL}, where each entry u_{ij} defines the degree of membership of jth unlabeled instance to the ith class.

Step 3 The unlabeled instances in U that show higher certainty of belonging to one class (i.e., have one value of degree of membership u_{ij} greater than the γ_1 confidence thresholds of SFCM) are further classified using the trained model M.

Step 4 Furthermore, each unlabeled data x most confidently classified by the trained model M (i.e., has output the value $F(x)$

Table 1
Attack categories of IoT ecosystem.

Denial of Service (DoS)	Probing (PROBE)	User to Root (U2R)	Remote to Local (R2L)
Teardrop	Satan	Rootkit	Warezmaster
Land	Port sweep	Load module	Wareclient
Smurf	NMAP	Buffer overflow	SPY
Neptune	IP sweep	Perl	Phf
Ping of death			Multi HOP
Back			IMAP
			Guess password
			FTP write

greater than the confidence threshold of ELM (γ_2)) are added to the labeled data together with their predictive label.

Step 5 The remaining unlabeled data are re-clustered using the SFCM, and re-training is carried out using the ELM.

Step 6 This process is repeatedly performed until the label is assigned to all the unlabeled data (D_{UL} is empty). (Steps 1–5 are repeated).

Step 7 The algorithm returns a trained model by employing the label assigned to all the unlabeled data and the labeled dataset, which is further used over the testing instances to classify them.

Thus, our algorithm combines the advantages of the ELM and SFCM algorithms together, provides superior performance at a faster learning rate, and handles the issue of labeled data by assigning labels to all unlabeled data.

Algorithm 1. ESFCM method

Input: D_{UL}: Unlabeled dataset, D_L: Labeled dataset, D_T: Test dataset, Classifier: ELM, Hidden-node output function: $q(y) = 1/1 + e^{-y}$, L=Number of hidden nodes, γ_1: Confidence thresholds of SFCM, γ_2: Confidence thresholds of ELM
Output $Acc_{(D_T)}$: Testing Accuracy

Process:

While ($D_{UL} \neq 0$) **do**
  Initialize: set $S_1 \longleftarrow \phi$, $S_2 \longleftarrow \phi$
  $M = ELM(D_L)$ /* Train ELM using D_L */
  $U = SFCM(D_L + D_{UL})$ /* obtain the matrix of membership degree for D_{UL}*/
  for all instance a_j in the D_{UL} **do**
    if($u_{ij} \geq \gamma_1$) **then**
      $S_1 \longleftarrow a_j$
    end if
  end for
    /* Examine the set S_1 using M and compute the output $F(x)$ for each instance x in the set S_1 */
  Examine $M(S_1)$
  for all instance x in the S_1 **do**
    if($F(x) \geq \gamma_2$) **then**
      $S_2 \longleftarrow x$
    end if
  end for
  Update $D_L \longleftarrow D_L \cup S_2$
  Update $D_{UL} \longleftarrow D_{UL} - S_2$
  if ($S_2 \neq \phi$) **then**
    Reduce (γ_1)
  end if
end while
Examine $M(D_T)$ /* Examine the dataset D_T using M */

4. Performance evaluation

4.1. Data specification

To evaluate the efficacy of our proposed framework and algorithm for attack detection in IoT, we utilized the intrusion detection dataset. The most widely used intrusion detection datasets are NSL-KDD [7] and KDDCUP'99 [12]. In our performance evaluation, we used the NSL-KDD dataset consisting of sample instances related to four categories of attacks as shown in Table 1 and recognized as key attacks in the IoT environment [37].

Each instance in the dataset consists of 41 input attributes and 1 class label. The attributes are largely categorized into three major types: (1) basic attributes examined using connection at the TCP/IP layer without considering the payload; (2) content attributes extracted by examining the TCP/IP payload and required to detect malicious actions inside the payload segment, and; (3) traffic attributes involving host and time-based traffic attributes and which are extracted from traffic by utilizing the historical window and temporal window, respectively. Table 2 depicts all the input attributes to describe the NSL-KDD dataset. The basic attributes are represented by attributes 1–10; content attributes and traffic attributes are illumined by attributes 11–22 and 23–41, respectively. A class label that can be either attack or normal defines the status of each instance in the dataset. The original KDDCUP'99 dataset further categorizes the attack class into different classes of attack. The details of each attack are provided in Table 1.

4.2. Data preprocessing

The attributes in the dataset have two types of values: symbolic and numeric. The attributes with numeric data types can be processed by ELM. Note, however, that the symbolic attributes cannot be processed by ELM. Thus, it is essential to convert symbolic data into numeric data to process them. Table 3 lists the symbolic attributes and the symbolic values associated with them. In Table 3, some attributes, such as is_guest_login, is_host_login, logged_in, and land can be processed in the same way as numeric attributes because these attributes have values 0 or 1. Note, however, that other attributes such as flag, service, and protocol_type are defined using more than two different values. The flag attribute has 11 distinct values. Similarly, the service and protocol_type attributes are defined by 66 and 3 distinct values, respectively. Many approaches have been proposed to process symbolic attributes. Such approaches involve Conditional Probability [38] and Dummy/Indicator Variables [39]. We employed the Indicator/Dummy Variables approach in our experiment to process service and flag attributes. Since this approach will escalate dataset dimensionality, however, clustering approach introduced by Hernndez-Pereira et al. [40] was also employed to cluster similar types for symbolic attributes. Thus, in our experiment, we first reduced the dimensionality of service and flag attributes by clustering them into different classes as shown in Tables 4 and reftab:Table 5. Then, we converted these classes into indicator variables.

ELM needs normalized data for the classification task. To normalize the data, scaling of data was performed. In our experiment, the KDDTest^{-21} and KDDTest^{+} datasets were used for testing purposes. The training was performed over the KDDTrain^{+}_20Percent dataset, which was divided into two subsets of D_{UL} and D_L where D_{UL} is the set of unlabeled instances and D_L is the set of labeled instances. To test the efficacy of the proposed algorithm properly, the size of D_{UL} was set to be much higher than that of D_L during the simulation process. The ratio of D_{UL} and D_L was 90:10, where 90% is unlabeled data D_{UL} and the remaining 10% is labeled data D_L. Note, however, that testing datasets KDDTest^{-21} and KDDTest^{+}

Table 2
Input attributes description.

Type	Attributes	Data type
Basic attributes	(1) Duration, (2) hot, (3) urgent, (4) wrong_fragment, (5) dst_bytes, (6) src_bytes.	Numeric
	(7) Flag, (8) Service, (9) Protocol_type.	Nominal
	(10 Land)	Binary
Content attributes	(11) num_outbound_cmds, (12) num_access_files, (13) num_shells, (14) num_file_creations, (15) num_root, (16) su_attempted, (17) root_shell, (18) num_compromised, (19) num_failed_logins.	Numeric
	(20) is_guest_login, (21) is_host_login, (22) logged_in.	Binary
Time-based traffic attributes	(23) srv_diff_host_rate, (24) diff_srv_rate, (25) same_srv_rate, (26) srv_rerror_rate, (27) rerror_rate, (28) srv_error_rate, (29) serror_rate, (30) srv_count, (31) Count.	Numeric
Host-based traffic attributes	(32)dst_host_srv_rerror_rate, (33) dst_host_rerror_rate, (34) dst_host_srv_serror_rate, (35) dst_host_serror_rate, (36) dst_host_srv_diff_host_rate, (37) dst_host_same_src_port_rate, (38) dst_host_diff_srv_rate, (39) dst_host_same_srv_rate, (40) dst_host_srv_count, (41) dst_host_count	Numeric

Table 3
Symbolic attributes with values.

Symbolic attributes	Symbolic values	Number of distinct values
Flag	REJ, S0, S1, RSTOS0, S2, OTH, SH, RSTO, SF, S3, RSTR	11
Service	netbios_dgm, auth, klogin, exec, pm_dump, ftp, http, dap, ftp_data, red_i, nntp, finger, ctf, courier, http_8001, supdup, login, efs, hostnames, systat, netbios_ns, time, ssh, iso_tsap, echo, tim_i, nnsp, uucp, mtp, ftp_u, netbios_ssn, imap4, urh_i, kshell, shell, smtp, ntp_u,....	66
Protocol_type	icmp, udp, tcp	3
Land	1 and 0	2
is_guest_login	1 and 0	2
is_host_login	1 and 0	2
logged_in	1 and 0	2

Table 4
Clustering of service attribute.

Symbolic attributes	Service type	Description
SC_1	domain, host name,....	Service belongs to names server
SC_2	nnetstat, svstat,....	Service that is responsible to obtain system statistics and parameters
SC_3	http,....	Service belongs to web applications
SC_4	imap4, smtp	Service for mail transfer
SC_5	tftp, ftp,....	Service for file transfer
SC_6	ssh, telnet,....	Service for remotely access other machines
SC_7	urp_i, ecr_i, tim_i, eco_i,....	Service for ICMP protocol
SC_8	Remaining services	All other services

Table 5
Clustering of flag attribute.

Flag class	Flag type	Description
FC_1	REJ	Connection attempt rejected
	S0	Connection attempt seen, no reply
FC_2	SF	Normal establishment and termination
	OTH	No SYN seen, just midstream traffic
	S1	Connection established but not terminated
FC_3	RSTO	Connection established, Sender aborted
	S2	Connection established and close attempt seen by Sender
FC_4	RSTR	Connection established, Receiver aborted
	S3	Connection established and close attempt seen by Receiver
FC_5	SH	Sender sends a SYN followed by a FIN and Receiver does not see SYN ACK
	RSTOS0	Sender sends a SYN followed by a RST and Receiver does not see SYN ACK
FC_6	SHR	Receiver sends a SYN ACK followed by a FIN and Sender does not see SYN
	RSTRH	Receiver sends a SYN ACK followed by a RST, and Sender does not see SYN

were employed as a whole during the performance evaluation of the proposed framework.

4.3. Evaluation methodology

The experiment was conducted after scaling and preprocessing of the required data to evaluate the performance of the proposed framework. In the experimental evaluation, a fog node having training and testing data was obtained in the data preprocessing stage. The fog node runs data training and testing on the proposed method (as described in Section 3.2). In the proposed method, ELM source code [41] for ELM and FuzzyCMeans [42] for SFCM were used and tested over the both KDDTest^{-21} and KDDTest^{+} dataset. For the ELM classifier, the sigmoidal function $q(y) = 1/1 + e^{-y}$ was used as a hidden-node output function, and the optimized value of the parameter L (number of hidden nodes) was selected by employing 50 simulations of the ELM with a 10-fold cross validation on the training data using a different value of L. Similarly, the Confidence thresholds of the ELM and SFCM were selected from the interval [0,1] by employing 10 simulations of both the ELM and the SFCM over the labeled and unlabeled datasets. The evaluation results on attack detection are reported in the form of confusion matrix as shown in Table 6.

Table 6
Confusion matrix.

Actual	Detected	
	Attack	Normal
Attack	T_{post}	F_{neg}
Normal	F_{post}	T_{neg}

Where T_{post}: true positive indicating the quantity of attack samples correctly detected, T_{neg}: true negative showing the quantity of normal samples correctly detected, F_{neg}: false negative showing the quantity of attack samples incorrectly detected, and F_{post}: false positive denoting the quantity of normal samples incorrectly detected.

The following standard measures can be calculated based on the confusion matrix to evaluate the performance of the proposed framework:

(a) Accuracy (ACC): the fraction of correctly detected samples in total detected samples.

$$ACC = (T_{post} + T_{neg})/(T_{post} + F_{neg} + F_{post} + T_{neg}) \quad (15)$$

(b) Positive Predictive Value (PPV): the proportion of correctly detected attack samples in the total samples detected as attack.

$$PPV = T_{post}/(T_{post} + F_{post}) \quad (16)$$

(c) Sensitivity (detection rate): the proportion of correctly detected attack samples in the total number of actual attack samples.

$$Sensitivity = T_{post}/(T_{post} + F_{neg}) \quad (17)$$

(d) F-score: the harmonic mean of PPV and sensitivity

$$F - score = 2T_{post}/(2T_{post} + F_{post} + F_{neg}) \quad (18)$$

(e) Mathew Correlation Coefficient (MCC): The coefficient to estimate the correlation between actual and detected results.

$$MCC = (T_{post} * T_{neg} - F_{post} * F_{neg}) / \sqrt{(T_{post} + F_{neg})(T_{post} + F_{post})(T_{neg} + F_{post})(T_{neg} + F_{neg})} \quad (19)$$

(f) AUC represents the area under the Receiver Operating Characteristic (ROC) curve defining the ratio of true positive samples and false positive samples.

4.4. Performance evaluation of the proposed framework

In this section, we present a thorough evaluation of our framework using the evaluation methodology presented above. In conducting our evaluation, we employed both KDDTest^{-21} and KDDTest^{+} datasets. In our experiment, we performed two types of comparative analysis. The first one is framework-level comparative analysis wherein the performance result of our distributed framework is compared with a centralized cloud-based framework. The second one is the evaluation of the proposed ESFCM method against other traditional machine learning classifiers for attack detection in IoT.

For a framework-level comparative analysis, the experiment was conducted by deploying the ESFCM algorithm on the cloud layer of the centralized framework and the fog node at the fog layer of our distributed framework. Table 7 summarizes the performance results of the centralized and distributed framework on both test datasets in terms of the standard evaluation measures as described previously. The performance results demonstrate that the distributed framework obtained a better performance compared with the centralized framework, which could be because, unlike in the centralized framework, each fog node in the distributed framework performs attack detection by processing the data obtained from the IoT devices connected to that node only. Thus, the detection load is shared among several fog nodes in the fog layer, as a result of which the detection performance improves. ACC in Table 7 shows that the proposed ESFCM method is better for both frameworks. Furthermore, the performances of both centralized and distributed frameworks were tested by varying the total quantity of data traffic and measuring the attack detection time against this variation. Fig. 3 shows a comparison of the attack detection time for the centralized and distributed frameworks. In Fig. 3, the distributed framework always achieved a lower detection time than the centralized framework for both the KDDTest^{-21} and KDDTest^{+} datasets. This could be because, in the distributed framework, the detection of attacks is performed at the fog layer, which is close to the IoT device, thereby shortening the detection time. In summary, the distributed framework outperforms the traditional centralized framework in detecting attacks in IoT networks, suggesting that distributed attack detection at the fog layer is an effective method of detecting attacks in smart IoT systems such as smart cities [29], where real-time detection is required.

In order to validate the efficacy of our proposed method for attack detection in the IoT, we employed six different traditional machine learning classifiers in our framework – Support Vector Machine (SVM) [43], Logistic Regression (LR) [44], k-Nearest Neighbors (k-NN) [45], Decision Tree (J48) [46], Random Forest (RF) [47], and Bayesian Network (BN) [48] – and evaluated its performance. Fig. 4a and b show the comparison of the proposed method against the traditional classifiers in case of both KDDTest^{-21} and KDDTest^{+} dataset for IoT attack detection. The following observation can be made using Fig. 4a and b: (1) Our proposed method has significantly higher performance than all of the traditional classifiers; (2) k-NN achieved higher performance among all the traditional classifiers.

To validate the significance of our research, we compared our research work with the other existing state-of-the-art research works based on various aspects. Table 8 summarizes existing works and our own research work based on the following six aspects: application type, security architecture, proposal of an innovative classification approach, conventional machine learning classifier, dataset labeling required, and methodology. In Table 8, 'application type' denotes the platform on which the proposed security method is applied, while 'security architecture' represents whether the architecture used by the research work is centralized or distributed. Table 8 also depicts whether the research work proposes an innovative classification approach or employs a conventional machine learning classifier. It can be easily seen in Table 8 that most of the research works deploy the centralized architecture

Table 7
Performance of proposed framework on KDDTest^{-21} and KDDTest^{+} dataset.

Framework	Datasets	Evaluation measure						
		Sensitivity (%)	PPV (%)	*F*-score (%)	ACC (%)	Test time (ms)	MCC (%)	AUC (%)
Distributed	KDDTest^{-21}	75.14	75.82	75.48	75.77	13	74.55	75.81
	KDDTest^{+}	86.11	86.59	86.35	86.53	11	85.79	85.95
Centralized	KDDTest^{-21}	70.10	70.55	70.32	70.54	24	68.85	68.91
	KDDTest^{+}	80.72	80.85	80.45	80.69	23	79.10	79.32

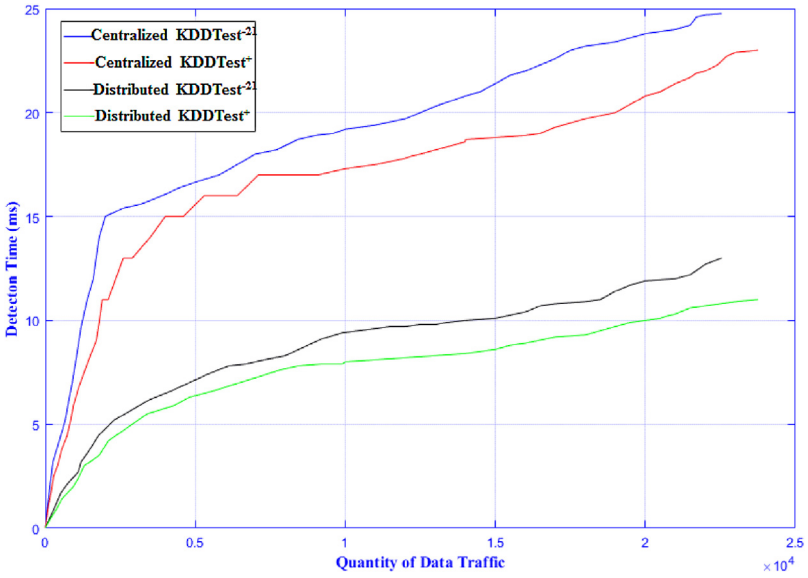

Fig. 3. Attack detection time comparison of centralized and distributed framework.

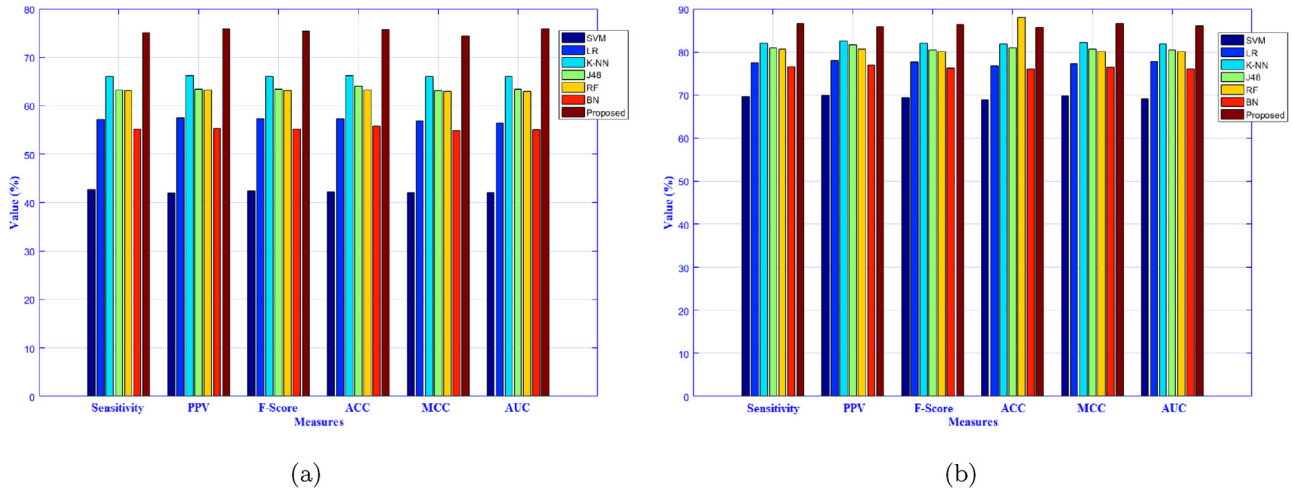

Fig. 4. Performance comparison of the proposed method with different traditional machine learning classifiers on (a) KDDTest^{-21}, (b) KDDTest^{+}.

Table 8
Significance of our research work over other state-of-the-art research works based on existing metrics.

Research work	Application type	Security architecture	Proposal of an innovative classification approach	Conventional machine learning classifier	Dataset labeling required	Methodology
Hamed Haddad Pajouh et al. [18]	IoT	Centralized	Yes	No	High	Two-layer Dimension Reduction and Two-tier Classification
Yair Meidan et al. [19]	IoT	Centralized	No	Yes	High	supervised machine learning algorithm
Qian Chen et al. [23]	IoT	Centralized	No	Yes	High	Naive Bayesian classifier
Lianbing Deng et al. [26]	IoT	Centralized	No	Yes	Less	Principal component analysis and FCM
Rupinder Singh el al. [27]	Wireless sensor networks	Cluster-based architecture	Yes	No	Less	A Multilayer Perceptron Neural Network with the fuzzy logic
R. Vijay Anand et al. [28]	Advanced metering infrastructure of smart grid	Centralized	No	Yes	High	Integration of multiple support vector machine classifiers
Proposed work	IoT	Distributed	Yes	No	Less	ELM-based Semi-Supervised Fuzzy C-Means (ESFCM) method

for security and employ the traditional machine learning classifier. Some research works employ a newly proposed classification approach, however they have concerns, such as high dataset labeling required, applicable for different types of applications instead of the IoT, and the deployment of a centralized security architecture. Note, however, that our proposed work supports the design and implementation of a distributed security architecture that is applicable to the IoT, and relies on a newly proposed ESFCM method that requires less labeled data to facilitate attack detection in the IoT, while offering superior performance over a shorter detection time.

5. Conclusion

In this paper, we studied security in IoT. Based on the study, we proposed a fog-based attack detection framework for IoT, which relies on a fog computing paradigm and a newly proposed ESFCM method to facilitate real-time distributed attack detection in IoT. Fog computing provides distributive capability wherein the detection load is shared to several fog nodes at the fog layer and detection performance improves. Moreover, the proposed ESFCM method solves the issue of labeled data unavailability and real-time detection by combining the ELM algorithm with the SFCM algorithm. The experimental evaluation on the NSL KDD dataset shows that the framework achieved superior performance, with an attack detection time of 11 ms and an accuracy rate of 86.53%. The comparison of the proposed ESFCM method with the traditional machine learning classifiers demonstrates that it can handle the issue of labeled data and achieve better performance. Our findings suggest that a fog computing paradigm can efficiently support distributed attack detection and that the newly proposed ESFCM method can handle the problem of labeled data in the IoT. The framework can be deployed with smart IoT systems – such as smart homes and smart cities – as a recognition unit that detects attacks by accumulating and analyzing information from a diverse set of IoT devices. However, the ELM applied in the proposed framework can further result in lower performance due to the random assignment of input bias and weights. Such random input bias and weights may cause an ill-posed problem in which the classification returns more than one solution. To address the ill-posed problem, our proposed framework can be enhanced by using a deep learning algorithm to eliminate the requirement of manual feature engineering and provide self-learning and compression capabilities, thus resulting in greater accuracy with faster processing.

Acknowledgement

We give a special thanks to Pradip Kumar Sharma for his suggestions, valuable comments, and review of the manuscript. This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) (No 2016R1A2B4011069).

References

[1] P.K. Sharma, S. Singh, Y.-S. Jeong, J.H. Park, Distblocknet: a distributed blockchains-based secure sdn architecture for iot networks, IEEE Commun. Mag. 55 (9) (2017) 78–85.
[2] P.K. Sharma, S.Y. Moon, J.H. Park, Block-vn: a distributed blockchain based vehicular network architecture in smart city, J. Inf. Process. Syst. 13 (1) (2017) 84.
[3] J. Jang, I.Y. Jung, J.H. Park, An effective handling of secure data stream in iot, Appl. Soft Comput. (2017).
[4] Statista, Internet of Things (iot) connected devices installed base worldwide from 2015 to 2025 (in billions), https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/, (accessed 30.10.17). 2018.
[5] Hp study reveals 70 percent of internet of things devices vulnerable to attack, http://www8.hp.com/sg/en/hp-news/press-release.html?id=1744676#.WgVBQ1uCzcs, (accessed 01.10.17). 2018.
[6] H. Suo, J. Wan, C. Zou, J. Liu, Security in the internet of things: a review, in: International Conference on Computer Science and Electronics Engineering (ICCSEE), vol. 3, IEEE, 2012, pp. 648–651.
[7] M. Tavallaee, E. Bagheri, W. Lu, A.A. Ghorbani, A detailed analysis of the kdd cup 99 data set, in: IEEE Symposium on Computational Intelligence for Security and Defense Applications, CISDA 2009, IEEE, 2009, pp. 1–6.
[8] A. Alrawais, A. Alhothaily, C. Hu, X. Cheng, Fog computing for the Internet of things: security and privacy issues, IEEE Internet Comput. 21 (2) (2017) 34–42.
[9] A.S. Sohal, R. Sandhu, S.K. Sood, V. Chang, A cybersecurity framework to identify malicious edge device in fog computing and cloud-of-things environments, Comput. Secur. (2017).
[10] A.M. Rahmani, T.N. Gia, B. Negash, A. Anzanpour, I. Azimi, M. Jiang, P. Liljeberg, Exploiting smart e-health gateways at the edge of healthcare internet-of-things: a fog computing approach, Future Gener. Comput. Syst. 78 (2018) 641–658.
[11] D.M. Mendez, I. Papapanagiotou, B. Yang, Internet of things: Survey on security and privacy, arXiv preprint arXiv:1707.01879.
[12] Kddcup 1999 data, http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html, (accessed 30.10.17).
[13] R.K. Gunupudi, M. Nimmala, N. Gugulothu, S.R. Gali, Future Gener. Comput. Syst. 74 (2017) 417–429.
[14] C.D. McDermott, A. Petrovski, Investigation of Computational Intelligence Techniques for Intrusion Detection in Wireless Sensor Networks, 2017.
[15] M. Lopez-Martin, B. Carro, A. Sanchez-Esguevillas, J. Lloret, Conditional variational autoencoder for prediction and feature recovery applied to intrusion detection in iot, Sensors 17 (9) (2017) 1967.
[16] F. Hosseinpour, P. Vahdani Amoli, J. Plosila, T. Hämälä inen, H. Tenhunen, An intrusion detection system for fog computing and iot based logistic systems using a smart data approach, Int. J. Digital Content Technol. Appl. 10 (2016).
[17] S. Zeng, X. Tong, N. Sang, R. Huang, A study on semi-supervised fcm algorithm, Knowl. Inf. Syst. 35 (3) (2013) 585–612.
[18] H.H. Pajouh, R. Javidan, R. Khayami, D. Ali, K.-K.R. Choo, A two-layer dimension reduction and two-tier classification model for anomaly-based intrusion detection in iot backbone networks, IEEE Trans. Emerg. Top. Comput. (2016).
[19] Y. Meidan, M. Bohadana, A. Shabtai, M. Ochoa, N. O. Tippenhauer, J. D. Guarnizo, Y. Elovici, Detection of unauthorized iot devices using machine learning techniques, arXiv preprint arXiv:1709.04647.
[20] X. Cao, D.M. Shila, Y. Cheng, Z. Yang, Y. Zhou, J. Chen, Ghost-in-zigbee: energy depletion attack on zigbee-based wireless networks, IEEE Internet Things J. 3 (5) (2016) 816–829.
[21] K. Zhang, X. Liang, R. Lu, X. Shen, Sybil attacks and their defenses in the Internet of things, IEEE Internet Things J. 1 (5) (2014) 372–383.
[22] P. Machaka, A. McDonald, F. Nelwamondo, A. Bagula, Using the cumulative sum algorithm against distributed denial of service attacks in Internet of things, in: International Conference on Context-Aware Systems and Applications, Springer, 2015, pp. 62–72.
[23] Q. Chen, S. Abdelwahed, A. Erradi, A model-based validated autonomic approach to self-protect computing systems, IEEE Internet Things J. 1 (5) (2014) 446–460.
[24] Q.M. Ashraf, M.H. Habaebi, Autonomic schemes for threat mitigation in Internet of things, J. Netw. Comput. Appl. 49 (2015) 112–127.
[25] H. Ning, H. Liu, L.T. Yang, Aggregated-proof based hierarchical authentication scheme for the Internet of things, IEEE Trans. Parallel Distrib. Syst. 26 (3) (2015) 657–667.
[26] L. Deng, D. Li, X. Yao, D. Cox, H. Wang, Mobile network intrusion detection for iot system based on transfer learning algorithm, Cluster Comput. (2018) 1–16.
[27] R. Singh, J. Singh, R. Singh, Fuzzy based advanced hybrid intrusion detection system to detect malicious nodes in wireless sensor networks, in: Wireless Communications and Mobile Computing, 2017.
[28] R. Vijayanand, D. Devaraj, B. Kannapiran, Support vector machine based intrusion detection system with reduced input features for advanced metering infrastructure of smart grid, in: 4th International Conference on Advanced Computing and Communication Systems (ICACCS), IEEE, 2017, pp. 1–7.
[29] C. De Maio, G. Fenza, V. Loia, F. Orciuoli, Distributed online temporal fuzzy concept analysis for stream processing in smart cities, J. Parallel Distrib. Comput. 110 (2017) 31–41.
[30] A.K. Jain, Data clustering: 50 years beyond k-means, Pattern Recogn. Lett. 31 (8) (2010) 651–666.
[31] N.R. Pal, K. Pal, J.M. Keller, J.C. Bezdek, A possibilistic fuzzy c-means clustering algorithm, IEEE Trans. Fuzzy Syst. 13 (4) (2005) 517–530.
[32] J. Bezdek, Pattern Recognition with Fuzzy Objective Function Algorithms, Springer Science & Business Media, New York, 1973.
[33] S. Basu, A. Banerjee, R. Mooney, Semi-supervised clustering by seeding, Proceedings of 19th International Conference on Machine Learning (ICML-2002, Citeseer) (2002).
[34] A. Bar-Hillel, T. Hertz, N. Shental, D. Weinshall, Learning distance functions using equivalence relations, Proceedings of the 20th International Conference on Machine Learning (ICML-03) (2003) 11–18.
[35] G.-B. Huang, Q.-Y. Zhu, C.-K. Siew, Extreme learning machine: theory and applications, Neurocomputing 70 (1–3) (2006) 489–501.
[36] G.-B. Huang, H. Zhou, X. Ding, R. Zhang, Extreme learning machine for regression and multiclass classification, IEEE Trans. Syst. Man Cybern. Part B (Cybern.) 42 (2) (2012) 513–529.

[37] Securing the internet of things: A proposed framework, http://www.cisco.com/c/en/us/about/security-center/secure-iot-proposed-framework.html (accessed 05.11.17).
[38] D.W. Aha, D. Kibler, M.K. Albert, Instance-based learning algorithms, Mach. Learn. 6 (1) (1991) 37–66.
[39] J. Neter, M.H. Kutner, C.J. Nachtsheim, W. Wasserman, Irwin Chicago Applied Linear Statistical Models, Vol. 4, 1996.
[40] E. Hernández-Pereira, J.A. Suárez-Romero, O. Fontenla-Romero, A. Alonso-Betanzos, Conversion methods for symbolic features: a comparison applied to an intrusion detection problem, Expert Syst. Appl. 36 (7) (2009) 10612–10617.
[41] Basic elm algorithms, http://www3.ntu.edu.sg/home/egbhuang/elm_codes.html (accessed 06.11.17).
[42] Mathworks, fcm, https://kr.mathworks.com/help/fuzzy/fcm.html (accessed 06.11.17). 2018.
[43] C.-C. Chang, C.-J. Lin, Libsvm: a library for support vector machines, ACM Trans. Intell. Syst. Technol. 2 (3) (2011) 27.
[44] S. Le Cessie, J.C. Van Houwelingen, Ridge estimators in logistic regression, Appl. Stat. (1992) 191–201.
[45] L.E. Peterson, K-nearest neighbor, Scholarpedia 4 (2) (2009) 1883.
[46] Class jrip, http://weka.sourceforge.net/doc.stable/weka/classifiers/rules/JRip.html.
[47] A. Liaw, M. Wiener, et al., Classification and regression by randomforest, R News 2 (3) (2002) 18–22.
[48] I. Rish, An empirical study of the naive bayes classifier, in: IJCAI 2001 Workshop on Empirical Methods in Artificial Intelligence, vol. 3, IBM New York, 2001, pp. 41–46.