# Learning Latent Representation for IoT Anomaly Detection

Ly Vu, Van Loi Cao, Quang Uy Nguyen, Diep N. Nguyen, Dinh Thai Hoang, and Eryk Dutkiewicz

***Abstract*—Internet-of-Things (IoT) has emerged as a cutting-edge technology that is changing human life. The rapid and widespread applications of IoT, however, make cyberspace more vulnerable, especially to IoT-based attacks in which IoT devices are used to launch attack on cyber-physical systems. Given a massive number of IoT devices (in order of billions), detecting and preventing these IoT-based attacks are critical. However, this task is very challenging due to the limited energy and computing capabilities of IoT devices and the continuous and fast evolving of attackers. Among IoT-based attacks, unknown ones are far more devastating as these attacks could surpass most of the current security systems and it takes time to detect them and "cure" the systems. To effectively detect new/unknown attacks, in this paper, we propose a novel representation learning method to better predictively "describe" unknown attacks, facilitating supervised learning-based anomaly detection methods. Specifically, we develop three regularized versions of AutoEncoders (AEs) to learn a latent representation from the input data. The bottleneck layers of these regularized AEs trained in a supervised manner using normal data and known IoT attacks will then be used as the new input features for classification algorithms. We carry out intensive experiments on nine recent IoT datasets to evaluate the performance of the proposed models. The experimental results demonstrate that the new latent representation can significantly enhance the performance of supervised learning methods in detecting unknown IoT attacks. We also conduct experiments to investigate the characteristics of the proposed models and the influence of hyperparameters on its performance. The running time of these models is about 1.3 milliseconds that is pragmatic for most applications.**

***Index Terms*—Supervised learning, IoT anomaly detection, unknown attacks, latent representation, autoencoders.**

## I. Introduction

With the rapid development of Internet-of-Thing (IoT) devices, IoT networks have been providing enormous benefits to many aspects of our life, such as healthcare, transportation, and manufacturing. IoT devices in such networks are able to transfer and collect data with minimal human-machine interactions [1]. The data transmitted from/to these IoT devices often contains sensitive information of users, such as passwords, phone numbers, photos and locations, which usually attracts hackers [2]. Moreover, the rapid growth in the number of diverse IoT devices can lead to a dramatic increase in the number of emerging IoT-based attacks (also referred to as *IoT anomalies*) [3], [4]. Identifying IoT anomalies in networks with a massive number of IoT devices would be a very challenging task, especially with the fast and continuous evolution of IoT anomalies [2]–[8]. Among many IoT attacks, unknown ones are the most dangerous but difficult to detect since these attacks could surpass most of the advanced security techniques and cause serious devastation to network systems [9], [10]. In this paper, we develop a novel learning representation method to better predictively "characterize" new/unknown anomalies. The resulting representation can facilitate supervised learning-based IoT anomaly detection methods, such as Support Vector Machine (SVM), Perceptron (PCT), Nearest Centroid (NCT), and Linear Regression (LR).

Amongst the most popular anomaly detection methods, machine learning-based techniques have proven their great potential in computer and IoT networks [4], [11]–[15]. Depending on the availability of data label, machine learning-based anomaly detection can be categorized into three main approaches: supervised learning, semi-supervised learning and unsupervised learning [16]. Supervised learning methods assume that both labeled normal and abnormal data are available for constructing predictive models. They attempt to model the distinction between normal behaviors and anomalous activities. The downside of supervised learning methods is that they require a sufficient number of normal and abnormal instances to achieve good performance. Moreover, these methods are often ineffective to detect unknown attacks, i.e., the attacks which are not contained in the training data. Semi-supervised learning methods assume that only labeled normal data is available for training models. These methods construct generative models representing normal behaviors and measure how well an unseen sample fits the models. Unsupervised methods require no labeled training data and rely on the assumption that the number of anomalous samples is far fewer than that of the normal samples [16]. Since, unsupervised and semi-supervised approaches do not require abnormal data to construct predictive models, they are often more robust to unknown attacks and hence widely applied to anomaly detection [10], [17], [18]. The limitation of semi-supervised and unsupervised approaches is, however, that they might not be as effective as supervised approaches in identifying the previously known attacks. In this paper, we develop a novel approach that performs effectively on both known and unknown attacks.

Recently, deep learning-based anomaly detection has received a greater attention from researchers and industry [4], [10], [18]–[23]. Among deep learning-based techniques, AutoEncoders (AEs) are widely used for anomaly detection [10]. An AE is a neural network consisting of two parts, an encoder and a decoder [24]. The encoder attempts to compress the input data into a latent feature space at its hidden layer, while the decoder tries to reconstruct the original input data from the latent space at the output layer [25]. An AE is trained to minimize the

Manuscript received January 6, 2020; revised May 15, 2020; accepted July 25, 2020. This research is funded by Vietnam National Foundation for Science and Technology Development (NAFOSTED) under grant number 102.05-2019.05.

L. Vu, V. L. Cao, and Q. U. Nguyen are with Le Quy Don Technical University, Hanoi, Vietnam (e-mail: ly.vu@lqdtu.edu.vn, loi.cao@lqdtu.edu.vn, and quanguyhn@gmail.com).

D. N. Nguyen, D. T. Hoang, and E. Dutkiewicz are with School of Electrical and Data Engineering, University of Technology Sydney, Australia (e-mail: Diep.Nguyen@uts.edu.au, hoang.dinh@uts.edu.au, and Eryk.Dutkiewicz@uts.edu.au).

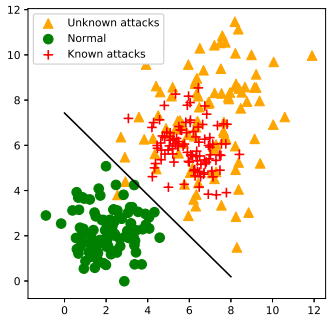



Fig. 1: Visualization of our proposed ideas: Known and unknown abnormal samples are separated from normal samples in the latent representation space.

difference between the input data and its reconstruction at the output layer, called reconstruction error (RE). When being trained on normal data, the AE can be used for detecting abnormal samples by observing its RE [26]. The trained AE often produces small values of RE on the normal data and much higher RE on the abnormal data. Alternatively, the hidden layer of AE can be used as a new feature representation (the latent representation or the latent feature space) for other anomaly detection techniques [9], [10].

In this paper, we propose a novel learning approach that can inherit the strength of supervised learning methods in detecting known IoT attacks and the ability to identify unknown attacks of unsupervised methods. In other words, we aim to learn a new feature representation that enhances the ability in identifying unknown IoT anomalies of supervised learning techniques. In the new feature space, normal data and known IoT anomalies will be forced into two tightly separated regions, called normal region (green circle points in Fig. 1) and anomalous region (red plus points in Fig. 1), respectively. We hypothesize that unknown attacks will appear closer to the anomalous region (yellow triangle points in Fig. 1) as they may share some common characteristics with known ones. Hence, they can be easily detected. In order to obtain the feature representation, we develop two new regularized AEs, namely Multi-distribution AE (MAE) and Multi-distribution Denoising AE (MDAE). These AEs will learn to construct the desired feature representation at their bottleneck layers (also called "latent feature space"). The latent feature space will be then used for facilitating traditional supervised learning techniques. More details of our proposed approach will be described in Section IV. Our major contributions are as follows:

- We introduce a new latent feature representation to enhance the ability in detecting unknown IoT anomalies of supervised learning-based anomaly detection methods.
- We propose two novel regularized AEs to learn the new latent representation. A new regularizer term is added to the loss function of these AEs to separate normal samples from abnormal samples in the latent space. This latent representation is then used as the input to classifiers to identify abnormal samples.
- We perform extensive experiments using nine latest IoT botnet datasets to evaluate our models. The experimental results show that our learning representation models help simple classifiers perform much better when comparing to learning from the original features or using latent representations produced by other AEs.
- We conduct thorough analysis on the characteristics of the latent representation in detecting unknown attacks, performing on cross-dataset test, and its robustness with various values of hyper-parameters. This analysis sheds light on the practical applications of the proposed models.

The rest of paper is organized as follows. In the next section, we highlight recent research on IoT anomaly detection. Section III briefly describes the fundamental background of the methods. The proposed models are then presented in Section IV. Section V presents the experimental settings. Section VI discusses and analyzes results obtained from the proposed models. Finally, in Section VII, we draw conclusions and suggest potential future work.

## II. Related work

IoT anomaly detection methods can be categorized into signature-based and semantic-based methods [11], [27], [28]. The signature-based methods are operated by observing frequently occurring strings or token sequences from anomalous traffic [5]–[8]. The Distributed Denial of Service (DDoS) attacks are usually mounted by IoT botnets[1]. Zhang et al. [5] proposed a lightweight and low-complexity algorithm to prevent DDoS attacks. This work aims to enable IoT devices (working nodes) to intelligently detect and avoid DDoS attacks. To detect anomaly traffic, the work in [5] analyzes the difference between a benign and a malicious request. Then, each node carries out a deep packet inspection to find anomaly signatures. In [6], Dietz et al. aimed to proactively block the spreading of IoT anomalies by automatically scanning vulnerable IoT devices and isolate them from the system. The authors of [7] proposed a host-based intrusion detection and mitigation (IoT-IDM) method using Software Defined Network (SDN) with OpenFlow protocols to address malicious behaviours and block intruders from accessing the IoT devices. When an attack occurs at the network-level, IoT-TDM will generate policies to block and isolate infected hosts. Nevertheless, this technique is not scalable as adding IoT devices would require one to manually tailor protocols. Ceron et al. [8] introduced a solution to handle the traffic generated by IoT malwares. This solution uses malware's actions to modify the traffic at the network layer. Generally, the signature-based approaches require a prior knowledge about the behaviours of known IoT anomalies. Consequently, these approaches are unable to detect unknown attacks which usually cause more serious consequences than known attacks [10].

The semantic-based approach relies on heuristic methods to analyze the anomalous behaviours. Specifically, features of traffic, e.g., the range of packet lengths, inter-packet arrival times, flow size, duration size are used to classify the benign and attack traffic. To that end, machine learning based methods prove themselves as effective solutions for IoT anomaly detection [4], [12]–[15]. Bahsi et al. [12] used the decision tree and K-nearest neighbor algorithms to identify the Mirai botnet family and the Gafgyt botnet family. Chawathe [13] applied

[1]The botnets manipulate through secured channels to launch attacks to targets [5].

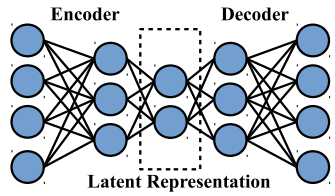
a number of machine learning algorithms including ZeroR and OneR, rule-based, and tree-based classifiers, e.g., J48 and Random Forest (RF) to detect IoT anomalies. However, these approaches are ineffective in detecting new types of botnets. Nomm et al. [14] proposed a solution for identifying anomalies from IoT datasets. First, the authors re-sample normal samples from the IoT datasets to enlarge the normal training set. Then, the Local Outlier Factor (LOF) and One-class Support Vector Machine (OCSVM) are trained on the normal samples of the training set. The trained LOF and OCSVM models can be used for detecting malicious samples. However, the sampling technique may change the distribution of the original data, thereby reducing the effectiveness of LOF and OCSVM. Omar et al. [15] proposed a framework combining feature selection methods (feature ranking and clustering-based data partitioning techniques) and classification for botnet intrusion detection systems. Although this framework and other previous approaches showed a great potential in identifying known botnets, using supervised learning classifiers makes them less effective in detecting unknown botnets.

Recently, deep learning has attracted paramount interest in detecting anomaly in cyber security, e.g., [4], [10], [19]–[23], in which AEs play pivotal roles, e.g., [4], [10], [18], [27], [29]. Meidan et al. [4] proposed an AE model to train with the normal data samples. It then sets a RE threshold to classify an unseen data sample to be a normal or malicious one. Juliette et al. [27] presented an online and real-time unsupervised network anomaly detection algorithm which uses a discrete time-sliding window to continuously update the feature space and an incremental grid clustering. Ibidunmoye et al. [29] estimated an underlying temporal property of the stream via adaptive learning, and then used statistically robust control charts to recognize deviations. However, this approach requires frequent adjustment of the threshold value.

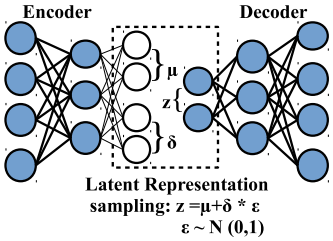
More recently Cao et al. [10] proposed two AE-based models, namely Shrink AE (SAE) and Dirac Delta VAE (DVAE), to learn a latent representation. This latent representation aims to facilitate one-class anomaly detection methods in dealing with high-dimension data. Specifically, the regularizer in [10] helps SAE and DVAE learn (in a semi-supervised manner) to project normal class in a small region at the origin. This is based on an assumption that only normal samples are available for training. They did not use any information of the anomalous class to train the representation models. In many scenarios, however, a certain type of IoT attacks can be collected and labelled. In this case, supervised learning-based methods are usually better than semi-supervised methods in detecting known anomalies. Therefore, our work aims to develop a novel latent representation that facilitates supervised learning-based anomaly detection methods in detecting unknown/new anomalies. Moreover, our proposed regularizers in this work can also be expanded to multiclassification problems that the models in [10] cannot do.

## III. Background

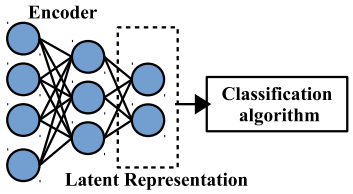
This section describes the structure and the loss functions of the AutoEncoder (AE) [24], Denoising AutoEncoder (DAE) [30] and Variational AutoEncoder (VAE) [31]. They are the core components of the proposed models in Section IV.




(a) AutoEncoder (AE).




(b) Variational AutoEncoder (VAE).




(c) Classification algorithms are applied on the latent representation of an AE

Fig. 2: (a) AE, (b) VAE, and (c) using latent representation of an AE for anomaly detection.

### A. AutoEncoder

An AE is a neural network trained to copy network's input to its output [24]. This network has two parts, i.e., encoder and decoder (as shown in Fig. 2 (a)). Let $\mathbf{W}$, $\mathbf{W}'$, $\mathbf{b}$, and $\mathbf{b}'$ be the weight matrices and the bias vectors of the encoder and the decoder, respectively, and $\mathbf{x} = \{x^1, x^2, \ldots, x^n\}$ be a training dataset. Let $\phi = (\mathbf{W}, \mathbf{b})$ and $\theta = (\mathbf{W}', \mathbf{b}')$ be parameter sets for training the encoder and the decoder, respectively. Let $q_\phi$ denote the encoder, $z^i$ be the representation of the input sample $x^i$. The encoder maps the input $x^i$ to the latent representation $z^i$ (in (1)). The latent representation of the encoder is typically referred to as a "bottleneck". The decoder $p_\theta$ attempts to map the latent representation $z^i$ back into the input space, i.e, $\hat{x}^i$ (in (2)).

$$z^i = q_\phi(x^i) = a_e(\mathbf{W}x^i + \mathbf{b}), \tag{1}$$

$$\hat{x}^i = p_\theta(z^i) = a_d(\mathbf{W}'z^i + \mathbf{b}'), \tag{2}$$

where $a_e$ and $a_d$ are the activation functions of the encoder and the decoder, respectively.

For a single sample $x^i$, the loss function of an AE is the difference between $x^i$ and the output $\hat{x}^i$. The loss function of an AE for a dataset is often calculated as the mean squared error (MSE) over all data samples [10] as in (3).

$$\ell_{AE}(\mathbf{x}, \phi, \theta) = \frac{1}{n}\sum_{i=0}^{n}\left(x^i - \hat{x}^i\right)^2, \tag{3}$$

$n$ is the number of data samples in a dataset.

DAE is a regularized AE [30] that aims to reconstruct the original input from a noised version of the input. This can

help DAE capture the true distribution of the input instead of learning the identity [25], [32]. There are several methods adding noise to the input data, and the additive isotropic Gaussian noise is the most common one. Let define an additive isotropic Gaussian noise $C(\tilde{\mathbf{x}}|\mathbf{x})$ to be a conditional distribution over a corrupted sample $\tilde{\mathbf{x}}$, given a data sample $\mathbf{x}$. Let define $\mathbf{x_{noise}}$ to be the noise component drawn from the normal distribution with the mean is 0 and the standard deviation is $\sigma_{noise}$, i.e., $\mathbf{x_{noise}} \sim \mathcal{N}(0, \sigma_{noise})$. The denoising criterion with the Gaussian corruption is presented as follows:

$$C(\tilde{\mathbf{x}}|\mathbf{x}) = \mathbf{x} + \mathbf{x_{noise}}. \tag{4}$$

Let define $\tilde{x}^i$ to be the corrupted version of the input data $x^i$ obtained from $C(\tilde{\mathbf{x}}|\mathbf{x})$. Note that the corruption process is performed stochastically on the original input each time a point $x^i$ is considered. Based on the loss function of AE, the loss function of DAE can be written as follows:

$$\ell_{DAE}\left(\mathbf{x}, \tilde{\mathbf{x}}, \phi, \theta\right) = \frac{1}{n}\sum_{i=0}^{n}\left(x^i - p_\theta(q_\phi(\tilde{x}^i))\right)^2, \tag{5}$$

where $\tilde{x}^i$ is the corrupted version of $x^i$ drawn from $C(\tilde{\mathbf{x}}|\mathbf{x})$. $q_\phi$ and $p_\theta$ are the encoder and decoder parts of DAE, respectively. $n$ is the number of data samples in a dataset.

### B. Variational AutoEncoder

A VAE [31] is a variant of an AE that also consists of two parts: encoder and decoder (Fig. 2 (b)). The difference between a VAE and an AE is that the bottleneck of the VAE is a Gaussian probability density ($q_\phi(\mathbf{z}|\mathbf{x})$). We can sample from this distribution to get noisy values of the representation $\mathbf{z}$. The decoder inputs a latent vector $\mathbf{z}$ and attempts to reconstruct the input. The decoder is denoted by $p_\theta(\mathbf{x}|\mathbf{z})$.

The loss function of a VAE $\ell_{VAE}(x^i, \theta, \phi)$ for a datapoint $x^i$ includes two terms as follows:

$$\begin{aligned}\ell_{VAE}(x^i, \theta, \phi) = &- \mathbf{E}_{q_\phi(z^i|x^i)}\left[log\, p_\theta(x^i|z^i)\right] \\ &+ D_{KL}(q_\phi(z^i|x^i)||p(z^i)).\end{aligned} \tag{6}$$

The first term is the expected negative log-likelihood of the *i*-th data point. This term is also called the reconstruction error (RE) of VAE since it forces the decoder to learn to reconstruct the input data. The second term is the Kullback-Leibler (KL) divergence between the encoder's distribution $q_\phi(\mathbf{z}|\mathbf{x})$ and the expected distribution $p(\mathbf{z})$. This divergence measures how close $q$ is to $p$ [31]. In the VAE, $p(\mathbf{z})$ is specified as a standard Normal distribution with mean zero and standard deviation one, denoted as $\mathcal{N}(0, 1)$. If the encoder outputs representations $\mathbf{z}$ that are different from those of a standard normal distribution, it will receive a penalty in the loss. Since the gradient descent algorithm is not suitable to train a VAE with a random variable $\mathbf{z}$ sampled from $p(\mathbf{z})$, the loss function of the VAE is re-parameterized as a deterministic function as follows [19]:

$$\begin{aligned}\ell_{VAE}(x^i, \theta, \phi) = &- \frac{1}{K}\sum_{k=1}^{K}\log p_\theta(x^i|z^{i,k}) \\ &+ D_{KL}(q_\phi(z^i|x^i)||p(z^i)).\end{aligned} \tag{7}$$

where $z^{i,k} = g_\phi(\epsilon^{i,k}, x^i)$. $g$ is a deterministic function, $\epsilon^k$ denotes $\mathcal{N}(0, 1)$. $K$ is the number of samples that is used to reparameterize $z^i$ for the sample $x^i$.

After training, the latent layer of AEs (AE, DAE, and VAE) can be used for a classification task (Fig. 2 (c)). The original data is passed through the encoder part of AEs to generate the latent representation. A classification algorithm is then applied on the latent representation instead of the original input. In Section IV we will present three novel models based on AEs for learning latent representation to detect unknown IoT attacks.

## IV. Proposed models

In this section, we describe our proposed latent representation that facilitates supervised learning-based anomaly detection methods in identifying anomalies, especially unknown attacks. We then present two novel regularized AEs that can learn to construct the new latent representation of data.

In the latent representation, normal samples and known anomalous samples are forced to distribute into two tightly separated regions, the normal region and the anomalous region, respectively. Unknown attacks that may share some common attributes with known attacks can be then identified as being closer to the anomaly region than the normal region. To achieve the feature representation, our approach is to develop the AE-based models that can learn to construct the new feature representation at the bottleneck layer. We introduce new regularized terms to the loss functions of AEs. Data labels are incorporated into the regularizers to compress normal and known anomalous data into two tiny separated regions associated with each class of data in the latent representation. The latent representation is then used as the input of binary classifiers, such as SVM and LR. The output of these classifiers is the final score to determine the abnormality of the input data sample.

We have done a preliminary work on learning to represent normal data and anomalous data into two different regions in the bottleneck layer of VAE, called Multi-distribution VAE (MVAE) [19]. VAEs originally learn to map input data into one region following the standard Gaussian shape $\mathcal{N}(0, 1)$ in its bottleneck layer. In MVAE, we incorporated the class labels into the loss function of VAE. This allows MVAE to force normal data and known anomalous data to reside in two different regions in its bottleneck layer. These regions follow the same Gaussian distribution shape ($\sigma = 1$), but reside in different areas determined by the mean values of the Gaussian distributions. The mean values are associated with class labels (normal class or anomalous class). We evaluated MVAE on two publicly network security datasets, and achieved promising results. However, the MVAE latent representation allows known anomalies to "freely" distribute in a "large room", thus MVAE may not reveal robust features which capture the common characteristics of known anomalies and unknown/new ones. Due to the "long tails" of these Gaussian distributions, some samples of normal and anomalous data in the "tails" may be overlapped [19, see Fig. 6]. Another drawback is that we used "large regions" to represent normal data and known anomalous data, thus unknown anomalies can have more chance to appear in the region of the normal class.

The new latent representation introduced in this paper aims to overcome the limitations of MVAE [19]. In particular, the proposed latent representation is able to force the normal and

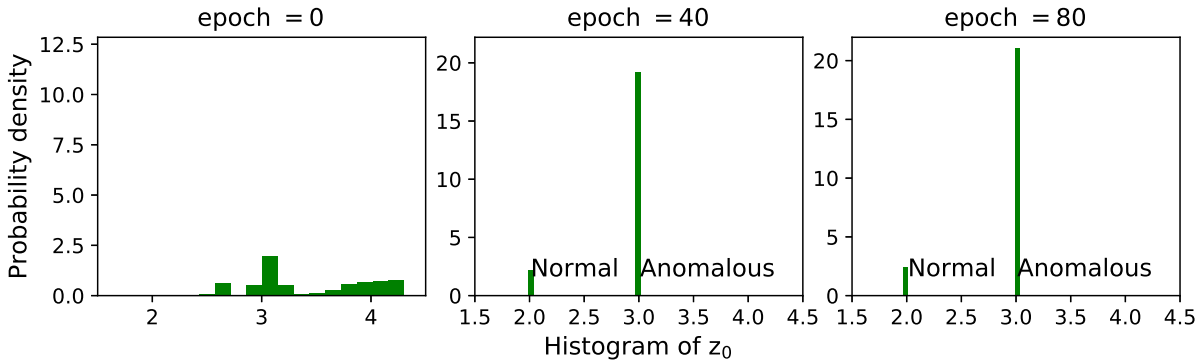



Fig. 3: The probability distribution of the latent data ($\mathbf{z}_0$) of MAE at epoch 0, 40 and 80 in the training process.

anomalous regions to be narrow (as illustrated in Fig. 3). Thus, the new feature representation possesses some outstanding advantages: the distributions of normal class and known anomaly class have very short tails (even no tails); large areas can be reserved for unknown anomalies; and common attributes of samples within each class can be better explored. As mentioned in Section I, known anomalies and unknown anomalies may share some common characteristics, and thus unknown ones should be towards the known anomalous region in this latent feature space. As such for effective learning the latent representation, we introduce new regularizers to a classical AE and a DAE to form two regularized AEs. These AEs are named as Multi-distribution AE (MAE) and Multi-distribution DAE (MDAE). The regularized AEs are simpler and easier to be trained than those of MVAE. Our proposed models are also very different from the regularized AEs presented in [10]. Specifically, the regularized AEs in [10] can learn to represent only normal class into a small region at the origin in a semi-supervised manner.

In this paper, we also introduce a new trade-off parameter to the loss function of MVAE in order to make MVAE more generalized to various anomaly detection problems. Therefore, we first describe our previous work, i.e., MVAE with the new trade-off parameter, in Sub-section IV-A. In Sub-sections IV-B and IV-C, we present our proposed models, i.e., MAE and MDAE.

### *A. Muti-distribution Variational AutoEncoder*

Muti-distribution Variational AutoEncoder (MVAE) [19] is a regularized version of VAE, aiming to learn the probability distributions representing the input data. To that end, we incorporate the label information into the loss function of VAE to represent data into two Gaussian distributions with different mean values. Given a data sample $x^i$ with its associated label $y^i$, $\mu_{y^i}$ is the distribution centroid for the class $y^i$. The loss function of MVAE on $x^i$ can be calculated as follows:

$$\begin{aligned}\ell_{MVAE}(x^i, y^i, \theta, \phi) = -\frac{1}{K}\sum_{k=1}^{K}\log p_\theta(x^i|z^{i,k}, y^i) \\ + D_{KL}(q_\phi(z^i|x^i, y^i)||p(z^i|y^i)),\end{aligned} \tag{8}$$

where $z^{i,k}$ is reparameterized as $z^{i,k} = \mu_{y^i} + \sigma^i\epsilon^k$ (more details presented in Section III) and $\epsilon^k \sim \mathcal{N}(0, 1)$; $K$ and $y^i$ are the number of samples used to reparameterize $x^i$ and the label of the sample $x^i$, respectively.

The loss function of MVAE consists of two terms. The first term is RE, or the expected negative log-likelihood of the *i*-th data point to reconstruct the original data at its output layer. The second term is created by incorporating the label information to the posterior distribution $q_\phi(z^i|x^i)$ and the prior distribution $p(z^i)$ of VAE in (7). Therefore, the second term is the KL divergence between the approximate distribution $q_\phi(z^i|x^i, y^i)$ and the conditional distribution $p(z^i|y^i)$. The objective of adding the label information to the second term is to force the samples from each class data to reside in each Gaussian distribution conditioned on the label $y^i$. Moreover, $p(z^i|y^i)$ follows the normal distribution with the mean $\mu_{y^i}$ and the standard deviation 1.0, $p(z^i|y^i) = \mathcal{N}(\mu_{y^i}, 1)^2$. The posterior distribution $q_\phi(z^i|x^i, y^i)$ is the multi-variate Gaussian with a diagonal covariance structure. In other words, $q_\phi(z^i|x^i, y^i) = \mathcal{N}(\mu^i, (\sigma^i)^2)$, where $\mu^i$ and $\sigma^i$ are the mean and standard deviation, respectively, are sampled from the sample $x^i$. Thus, the Multi-KL term in (8) is rewritten as follows:

$$\begin{aligned}D_{KL}(q_\phi(z^i|x^i, y^i)||p_\theta(z^i|y^i)) \\ = D_{KL}(\mathcal{N}(\mu^i, (\sigma^i)^2)||\mathcal{N}(\mu_{y^i}, 1)).\end{aligned} \tag{9}$$

Let $D$, $\mu^i_j$ and $\sigma^i_j$ denote the dimension of $z^i$, the $j$-th element of $\mu^i$ and $\sigma^i$, respectively; $\mu_{y^i_j}$ is the $j$-th element of $\mu_{y^i}$. Then, applying the computation of the KL divergence in [19], the Multi-KL term is rewritten as follows:

$$\begin{aligned}D_{KL}\left(q_\phi(z|x^i, y^i)\|p_\theta(z|y^i)\right) \\ = \frac{1}{2}\sum_{j=1}^{D}\left((\sigma^i_j)^2 + (\mu^i_j - \mu_{y^i_j})^2 - 1 - \log((\sigma^i_j)^2)\right).\end{aligned} \tag{10}$$

[2]We have tested several small values ($10^{-3}, 10^{-2}$ and $10^{-1}$) for the covariance of the Gaussian distributions of the two classes in order to shorten "tails" of these distributions. At the early iterations of the MVAE training process, the Multi-KL term of MVAE is extreme large in comparison to the RE term, which makes MVAE difficult to reconstruct the input data. In the later iterations, the Multi-KL term is small, but both of the two terms fluctuate substantially [10].

Taking Multi-KL term in (10), the loss function of MVAE in (8) finally is rewritten as follows:

$$\ell_{MVAE}(x^i, y^i, \theta, \phi) = -\frac{1}{K}\sum_{k=1}^{K}\log p_\theta(x^i|z^{i,k}, y^i) + \lambda\frac{1}{2}\sum_{j=1}^{D}((\sigma_j^i)^2 + (\mu_j^i - \mu_{y_j^i})^2 - 1 - \log((\sigma_j^i)^2)), \tag{11}$$

where $\lambda$ is a parameter to control the trade-off between two terms in (11) as discussed in [10]. The trade-off parameter $\lambda$ is approximated by the ratio of two loss terms, i.e., the RE and Multi-KL terms, in the loss function of MVAE.

The mean values for the distributions of the normal class and anomalous class are chosen in order to make these distributions to locate far enough from each others. In our experiments, the mean values are 4 and 12 for the normal class and anomalous class, respectively. These values are calibrated from the experiments for the good performance of MVAE. In this paper, the distribution centroid $\mu_{y^i}$ for the class $y^i$ and the trade-off parameter $\lambda$ are determined in advance. The hyper-parameter $\mu_{y^i}$ can receive two values associated with the normal class and the anomalous class.

### B. Multi-distribution AutoEncoder

This subsection describes how to integrate a regularizer to an AE to create Multi-distribution AutoEncoder (MAE). The regularizer is a multi-distribution penalty, called $\Omega(\mathbf{z})$, on the latent representation $\mathbf{z}$. The penalty $\Omega(\mathbf{z})$ encourages the MAE to construct a new latent feature space in which each class of data is projected into a small region. Specifically, we incorporate class labels into $\Omega(\mathbf{z})$ in order to restrict the data samples of each class to lie closely together centered at a pre-determined value. The new regularizer is presented in (12).

$$\Omega(\mathbf{z}) = \big|\big|\mathbf{z} - \mu_{y^i}\big|\big|^2, \tag{12}$$

where $\mathbf{z}$ is the latent data at the bottleneck layer of MAE, and $\mu_{y^i}$ is a distribution centroid of class $y^i$ in the latent space. The label $y^i$ used in $\Omega(\mathbf{z})$ maps the input data into its corresponding region defined by $\mu_{y^i}$ in the latent representation. The latent feature space is represented by multiple distributions based on the number of classes. Thus, we name the new regularized AE to be Multi-distribution AE.

In the MAE loss function, we also use a parameter $\lambda$ to control the trade-off between the reconstruction error (RE) and $\Omega(\mathbf{z})$ terms as discussed in Sub-section IV-A. Thus, the loss function of MAE can be defined as follows:

$$\ell_{MAE}(\theta, \phi, \mathbf{x}, \mathbf{z}) = \frac{1}{n}\sum_{i=0}^{n}\left(x^i - \hat{x}^i\right)^2 + \lambda\frac{1}{n}\sum_{i=1}^{n}\big|\big|z^i - \mu_{y^i}\big|\big|^2, \tag{13}$$

where $x^i$, $z^i$ and $\hat{x}^i$ are the *i*-th element of the input samples, its corresponding latent data and reconstruction data, respectively. $y^i$ and $\mu_{y^i}$ are the label of the sample $x^i$ and the centroid of class $y^i$, respectively. $n$ is the number of training samples. The first term in (13) is the RE that measures the difference between the input data and its reconstruction. The second term is the regularizer used to compress the input data to the separated regions in the latent space.

To visualize the probability distribution of the latent representation of MAE, i.e., $\mathbf{z}$, we calculate the histogram of one feature of the latent data $\mathbf{z_0}$. Fig. 3 presents the probability distribution of $\mathbf{z_0}$ of normal class and known anomalies during the training process of MAE on the IoT-1 dataset. After some epochs, the latent data is constrained into two tight regions in the latent representation of MAE.

### C. Multi-distribution Denoising AutoEncoder

In this subsection, we discuss the details of the multi-distribution Denoising AutoEncoder (MDAE). In this paper, we employ DAE proposed in [30] to develop MDAE. For each data sample $x^i$, we can draw its corrupted version $\tilde{x}^i$ using (4). MDAE learns to reconstruct the original input $x^i$ from a corrupted data $\tilde{x}^i$, and also penalizes the corresponding latent vector $z^i$ to be close to $\mu_{y^i}$. The loss function of MDAE can be presented in (14).

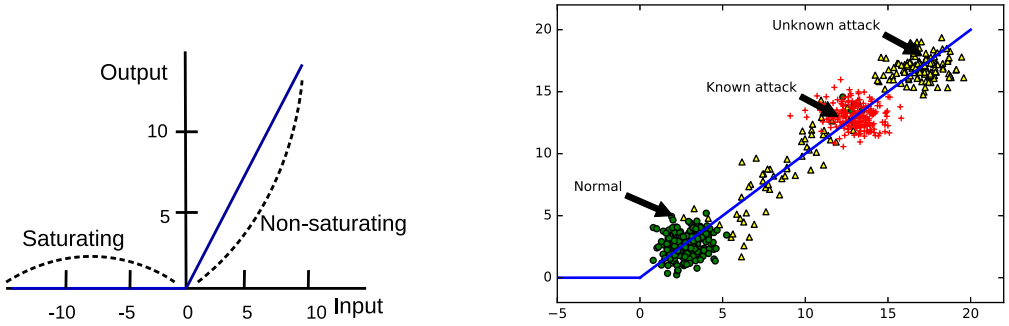
$$\ell_{MDAE}(\mathbf{x}, \tilde{\mathbf{x}}, \mathbf{z}, \phi, \theta) = \frac{1}{n}\sum_{i=0}^{n}\left(x^i - p_\theta(q_\phi(\tilde{x}^i))\right)^2 + \lambda\frac{1}{n}\sum_{i=1}^{n}\big|\big|z^i - \mu_{y^i}\big|\big|^2, \tag{14}$$

where $z^i$ is the latent vector of the data sample $x^i$. $\mu_{y^i}$ is the predefined distribution centroid of the class $y^i$ in the latent feature space of MDAE. $q_\phi$ and $p_\theta$ are the encoder and decoder parts as in DAE, respectively. $n$ is the number of training samples. The hyper-parameter $\lambda$ controls the trade-off between two terms in (14).

To be better separate the normal data and known anomalies and to encourage unknown anomalies moving toward the anomaly region, $\mu_{y^i}$ of MAE and MDAE is selected in the non-saturated area of activation. The non-saturated area is the steep slope area of the graph of the activation function (as shown in Fig. 4 (a)). Thus, $\mu_{y^i}$ needs to be assigned to a positive value under the ReLu activation function. In our experiments, we set $\mu_{y^i} = 2$ for a normal class and $\mu_{y^i} = 3$ for an abnormal class[3]. These values of $\mu_y^i$ are used in all the IoT-based datasets in our experiments.

Given the assigned values of $\mu_{y^i}$, the regularized term will attempt to project the two classes (normal data and known anomalies) into two tightly separated regions on the non-saturated area. If an anomaly is different from normal data points in the input feature space, it tends to differ greatly in the latent representation space [10]. The unknown anomalies are predicted to project to different regions towards the known anomaly region on the non-saturated area of the activation function. Fig. 4 (b) illustrates our idea of learning the latent representation using the ReLU activation function. In this representation, normal data and known attacks are represented in two separated regions, and unknown attacks are predicted to appear in regions closer to known anomalies.

[3]These values are calibrated from the experiments for the good performance of the proposed model. The data points within each data class are forced to be very close to the distribution centroid of each class. Thus, it is sufficient to separate two normal and anomalous regions with the pair of centroids values 2 and 3. The pair of centroid values also keep almost latent vectors being not much larger than the input and the output of MAE and MDAE resulting an easy training process.




(a) Description of saturating and non-saturating areas of the ReLu activation function.

(b) Illustration of the output of ReLU: two separated regions for normal data and known attacks; unknown attacks are hypothesized to appear in regions toward known attacks.

Fig. 4: Using non-saturating area of activation function to separate known and unknown attacks from normal data.

## V. Experimental Settings

This section presents the evaluation of three models MVAE, MAE, and MDAE for learning the latent representation. Four linear classification algorithms including Support Vector Machine (SVM) [33], Peceptron (PCT) [34], Nearest Centroid (NCT) [35] and Linear Regression (LR) [36], are applied to the latent representation produced from MVAE, MAE, and MDAE. We choose these linear classifiers as they are simple and hence can be performed very fast. Thus, these algorithms are appropriate to using in IoTs networks where the device's computing resource is often constrained. The experiments were conducted on nine IoT datasets (specified below). The source code of all tested methods are available for download[4]. All techniques were implemented in python using Tensorflow and Scikit learn frameworks [37]. Moreover, the same computing platform (Operating system: Ubuntu 16.04 (64 bit), Intel(R) Core(TM) i5-5200U CPU, 2 cores and 4GB RAM memory) was used in every experiment in this paper.

To highlight the strength of the proposed models, the performances of the four classifiers trained on the latent representation of MVAE, MAE, and MDAE are compared with those from: (1) stand-alone classifiers (without using latent representation) including a very effective and widely use for network anomaly detection, RF [19], [38], [39]; (2) classifiers using the latent representations of AE, DAE, VAE and Deep Belief Network (DBN) [40]. We carried out four experiments to investigate the properties of the latent representations obtained by MVAE, MAE, and MDAE.

- *Ability to detect unknown attacks:* Evaluate the accuracy of the four classifiers that are trained on the latent representation of the proposed models in comparison to those working with AE, DAE, VAE and DBN, and on the original input data with RF.
- *Cross-datasets evaluation:* Investigate the influence of the various attack types used for training models on the accuracy classifiers in detecting unknown attacks.
- *Influence of parameters*
  - *Influence of the noise factor*: Measure the influence of the noise level on the latent representation of MDAE.
  - *Influence of the hyper-parameters of classifiers:* Investigate the effects of hyper-parameters on the accuracy of the classifiers working on different latent representations.

The experiment settings, IoT attack datasets and metrics used for evaluating our proposed models are presented in the next subsections.

### A. IoT Attack Datasets

We used nine IoT attack-related datasets introduced by Y. Meidan et al. [4] for evaluating our proposed models. These data samples were collected from nine commercial IoT devices in their lab with two most well-known IoT-based botnet families, i.e., Mirai and BASHLITE (Gafgyt). Each of the botnet family contains five different IoT attacks. Among these IoT attack datasets, there are three datasets, namely Ennio_Doorbell (IoT-3), Provision_PT_838_Security_Camera (IoT-6), Samsung_SNH_1011_N_Webcam (IoT-7) containing only one IoT botnet family (five types of botnet attacks). The rest of these datasets consist of both Mirai and BASHLITE (ten types of DDoS attacks). Each data sample has 115 attributes which are categorized into three groups: stream aggregation, time-frame, and the statistics attributes. The details of the datasets are presented in Table I.

We split each of these datasets into a training set and a testing set based on the scenarios presented in Section VI. We randomly select $10\%$ of the training data to create validation sets for model selection [41].

### B. Parameter Settings

The configuration of AE-based models including AE, MAE, MDAE and MVAE is as follows. The parameter for balancing between the RE term and the regularized term $\lambda$ is set at 1 for MAE and MDAE and at 1000 for MVAE[5]. The number of hidden layers is 5, and the size of the bottleneck layer $m$ is calculated using the rule $m = \lceil 1 + \sqrt{n} \rceil$ in [10], where $n$

[4]https://github.com/vuthily/multi-distribution-representation-learning.

[5]The reason for setting much higher value of $\lambda$ for MVAE than MAE and MDAE is that the RE value of MVAE is often much higher than the regulazier value of MVAE while the RE value of MAE and MDAE is mostly equal to the their regularizer.

TABLE I: The nine IoT datasets.

| Dataset | Device Name | Training Attacks | Training size | Testing size |
|---|---|---|---|---|
| IoT-1 | Danmini_Doorbell | combo, ack | 239488 | 778810 |
| IoT-2 | Ecobee_Thermostat | combo, ack | 59568 | 245406 |
| IoT-3 | Ennio_Doorbell | combo, tcp | 174100 | 181400 |
| IoT-4 | Philips_B120N10_Baby_Monitor | tcp, syn | 298329 | 800348 |
| IoT-5 | Provision_PT_737E_Security_Camera | combo, ack | 153011 | 675249 |
| IoT-6 | Provision_PT_838_Security_Camera | ack, udp | 265862 | 261989 |
| IoT-7 | Samsung_SNH_1011_N_Webcam | combo, tcp | 182527 | 192695 |
| IoT-8 | SimpleHome_XCS7_1002_WHT_Security_Camera | combo, ack | 189055 | 674001 |
| IoT-9 | SimpleHome_XCS7_1003_WHT_Security_Camera | combo, ack | 176349 | 674477 |

is the number of the input features. The batch size is $100$, and the learning rate is set at $10^{-4}$. The weights of these AEs are initialized using the methods proposed by Glorot et al. [42] to facilitate the convergence. We employ the ADAM optimization algorithm [43] for training these networks. In these AEs, the Identity and Sigmoid activation functions are used in the bottleneck layers and the output layers, respectively. The rest of layers use the ReLu activation function.

We use the validation sets to evaluate our proposed models at every $20$ epochs for early-stopping. If the average of the Area Under the Curve (AUC) scores (AUC metric will be described in the next subsection) produced from the four classifiers, SVM, PCT, NCT, and LR decreases for a certain amount consecutively over a number of epochs, the training process will be stopped. The hyper-parameters of these classifiers are set by default values as in [44]. The DBN-based model has three layers as in [40] and is implemented by [45] where the number of neurons in each layer is similar to the AEs based models in our experiments.

### C. Evaluation Metrics

To evaluate the effectiveness the of the proposed models, we used three performance metrics to measure the accuracy of classifiers trained on the latent representation of AEs. The first two metrics are False Alarm Rate (*FAR*) and Miss Detection Rate (*MDR*). *FAR* is defined as the number of false alarms of negative samples per the total number of real negative data samples as in (15). *MDR* is defined as the number of miss detection of positive samples per total of real positive samples as in (16).

$$FAR = \frac{FP}{FP + TN}, \tag{15}$$

$$MDR = \frac{FN}{FN + TP}, \tag{16}$$

where *TP* and *FP* are the numbers of correct and incorrect predicted samples for one class respectively, and *TN* and *FN* are the numbers of corrected and incorrect predicted samples of the rest of classes, respectively.

The last metric is Area Under the Curve (AUC). AUC [46] is created by plotting the graph of true positive rate $TPR = \frac{TP}{TP+FN}$ against the false positive rate $FPR = \frac{FP}{FP+TN}$ at various threshold settings. The area of the region under this graph is defined as AUC. Since, the AUC score is one of the most important metrics for evaluating any classification model's performance, we use it as the main metric for comparing between various tested models in this paper. *FAR* and *MDR* are used to add more information about the model's performance in some experiments.

## VI. Results and Analysis

This section describes in details the main experiments and the investigation on the proposed latent representation models. More importantly, we try to give the explanation for the experimental results.

### A. Ability to Detect Unknown Attacks

This section presents the main experimental results of our paper. We evaluate the proposed models based on the ability to detect unknown attacks of the four classifiers training on the latent representation. As mentioned above, each of the nine IoT datasets has five or ten specific types of botnet attacks. For each IoT dataset, we randomly select two types of IoT attacks, and $70\%$ of normal traffic for training, and the rest of IoT attacks and normal data are used for evaluating our models. The training attacks in this experiment are shown in Table I. As seen in this table, we only use two types of DDoS attacks for training, the rest is for testing. This guarantees that there are some types of IoT attacks used for evaluating models that have not been seen in the training process. These types of attacks are considered as unknown attacks. The results produced from the four classifiers working with our proposed models are also compared with those working with the original input space and the latent feature space of AE, DAE, VAE, and DBN. We also compare the results from all linear classifiers with one non-linear classifier (i.e., RF) that is trained on the original feature. The main experimental results (AUC scores) are shown in Table II.

In Table II, we can observe that the classifiers are unable to detect unseen IoT attacks (the AUC scores approximates $0.5$) on the representation resulting from the VAE model. The reason is that the VAE model aims to generate data samples from the normal distribution instead of building a robust representation for classification task. It can be also seen from Table II that the performances of the four classifiers working with all latent representation models on the IoT-9 dataset are not consistent as those on other datasets. When observing the latent representation of AE and DAE, LR and SVM can perform very well on the IoT-9 dataset while PCT and NCT can not. On the contrary, LR and SVM perform less efficiently than PCT and NCT when working on the latent representation of our proposed models.

It can be seen from Table II that, the latent representations resulting from MVAE, MAE, and MDAE help four classifiers achieve higher classification accuracy in comparison to those using the original data. For example, the AUC scores of SVM, PCT, NCT, and LR working on the latent representation of MAE

TABLE II: AUC scores produced from the four classifiers SVM, PCT, NCT and LR when working with standalone (STA), our models, DBN, AE, VAE, and DAE on the nine IoT datasets. In each classifier, we highlight top three highest AUC scores where the higher AUC is highlighted by the darker gray. Particularly, RF is chosen to compare STA with a non-linear classifier and deep learning representation with linear classifiers.

| Class-ifiers | Models | Datasets | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | IoT-1 | IoT-2 | IoT-3 | IoT-4 | IoT-5 | IoT-6 | IoT-7 | IoT-8 | IoT-9 |
| RF | STA | 0.979 | 0.963 | 0.962 | 0.670 | 0.978 | 0.916 | 0.999 | 0.968 | 0.838 |
| SVM | STA | 0.839 | 0.793 | 0.842 | 0.831 | 0.809 | 0.934 | 0.999 | 0.787 | 0.799 |
| | DBN | 0.775 | 0.798 | 0.950 | 0.941 | 0.977 | 0.822 | 0.960 | 0.772 | 0.757 |
| | AE | 0.845 | 0.899 | 0.548 | 0.959 | 0.977 | 0.766 | 0.976 | 0.820 | **0.997** |
| | VAE | 0.500 | 0.500 | 0.500 | 0.500 | 0.500 | 0.500 | 0.500 | 0.500 | 0.500 |
| | DAE | 0.849 | 0.990 | 0.569 | 0.968 | 0.980 | 0.803 | 0.982 | 0.818 | 0.996 |
| | MVAE | 0.914 | 0.948 | 0.978 | 0.985 | 0.932 | 0.950 | **0.998** | 0.826 | 0.858 |
| | MAE | **0.999** | 0.997 | **0.999** | 0.987 | **0.982** | **0.999** | **0.999** | 0.846 | 0.842 |
| | MDAE | **0.999** | **0.998** | 0.999 | **0.992** | **0.982** | **0.999** | **0.999** | **0.892** | 0.902 |
| PCT | STA | 0.768 | 0.834 | 0.568 | 0.835 | 0.809 | 0.933 | 0.998 | 0.753 | 0.802 |
| | DBN | 0.995 | 0.786 | 0.973 | 0.954 | 0.697 | 0.847 | 0.957 | 0.783 | 0.755 |
| | AE | 0.849 | 0.892 | 0.498 | 0.965 | 0.977 | 0.813 | 0.977 | 0.814 | 0.815 |
| | VAE | 0.503 | 0.501 | 0.499 | 0.501 | 0.507 | 0.497 | 0.500 | 0.500 | 0.499 |
| | DAE | 0.882 | 0.903 | 0.534 | 0.969 | 0.982 | 0.862 | 0.984 | 0.857 | 0.849 |
| | MVAE | 0.954 | 0.947 | 0.972 | 0.986 | 0.923 | 0.923 | 0.997 | 0.823 | 0.849 |
| | MAE | **0.996** | 0.996 | **0.999** | **0.998** | **0.989** | **0.999** | **0.999** | 0.833 | **0.991** |
| | MDAE | **0.996** | **0.997** | **0.999** | **0.998** | **0.989** | **0.999** | **0.999** | **0.889** | **0.991** |
| NCT | STA | 0.743 | 0.747 | 0.498 | 0.785 | 0.692 | 0.570 | 0.993 | 0.770 | 0.748 |
| | DBN | 0.994 | 0.786 | 0.954 | 0.938 | 0.961 | 0.927 | 0.859 | 0.781 | **0.964** |
| | AE | 0.985 | 0.767 | 0.498 | 0.834 | 0.835 | 0.997 | 0.945 | 0.746 | 0.767 |
| | VAE | 0.501 | 0.506 | 0.511 | 0.487 | 0.499 | 0.505 | 0.500 | 0.488 | 0.479 |
| | DAE | 0.989 | 0.770 | 0.580 | 0.882 | 0.863 | 0.997 | 0.966 | 0.806 | 0.788 |
| | MVAE | 0.846 | 0.939 | 0.973 | 0.984 | 0.927 | 0.937 | 0.998 | 0.822 | 0.796 |
| | MAE | **0.998** | 0.996 | **0.999** | 0.987 | 0.982 | **0.999** | **0.999** | 0.828 | 0.799 |
| | MDAE | 0.996 | **0.998** | 0.998 | **0.992** | **0.985** | **0.999** | **0.999** | **0.887** | 0.889 |
| LR | STA | 0.862 | 0.837 | 0.565 | 0.829 | 0.802 | 0.932 | 0.998 | 0.791 | 0.800 |
| | DBN | 0.776 | 0.939 | 0.960 | 0.955 | 0.961 | 0.837 | 0.962 | 0.779 | 0.755 |
| | AE | 0.850 | 0.894 | 0.498 | 0.958 | **0.987** | 0.743 | 0.996 | 0.795 | **0.998** |
| | VAE | 0.500 | 0.499 | 0.500 | 0.500 | 0.500 | 0.500 | 0.500 | 0.500 | 0.500 |
| | DAE | 0.871 | 0.902 | 0.587 | 0.966 | 0.982 | 0.801 | 0.996 | 0.810 | 0.988 |
| | MVAE | 0.921 | 0.989 | 0.981 | 0.985 | 0.933 | 0.955 | **0.999** | 0.828 | 0.858 |
| | MAE | **0.999** | 0.997 | **0.999** | 0.988 | 0.984 | **0.999** | **0.999** | 0.835 | 0.840 |
| | MDAE | 0.996 | **0.998** | 0.998 | **0.992** | 0.985 | **0.999** | **0.999** | **0.887** | 0.889 |

are increased from $0.839$, $0.768$, $0.743$, and $0.862$ to $0.999$, $0.996$, $0.998$, and $0.999$ with those working on the original data on the IoT-1 dataset, respectively. The increase in the classification accuracy can be also observed from MDAE and MVAE. Moreover, our proposed models also help the linear classifiers achieve higher AUC scores (the fifth and sixth rows) than those using the latent representations of the AE and DBN (the third and fourth rows). Among the linear classifiers, PCT working with the latent representation of MVAE, MAE and MDAE enhance the accuracy on all the IoT datasets including IoT-9. Finally, four classifiers trained on the latent representations of MAE and MDAE tend to produce more consistent results than the previous one (MVAE) in [19].

Comparing the accuracy of linear classifiers with non-linear classifier (i.e., RF), the table shows that RF is often much better than all linear classifiers when they are trained on the original features. This evidences that these datasets are not linearly separable in the original space. However, by training on the latent representation of MVAE, MAE and MDAE, the accuracy of all linear classifiers are considerably improved and they are often much greater than that of RF. The exception only occurs in IoT-8 where none of the linear classifiers can outperform RF. This result verifies that the proposed models help to project the non-linear datasets in the original space into a linearly separable data in the latent space.

We also carried out an experiment to explain why our models can support conventional classifiers to detect unknown attacks efficiently. In this experiment, we train the AE and MAE on normal data and TCP attacks. Moreover, the size of the hidden layer of AE and MAE is set at 2 to facilitate for the visualization. After training, we test these models on the testing data containing normal samples, the TCP attacks (known attacks) and the UDP attacks (unknown attacks). In Fig. 5, we plot $1000$ random samples of the training and the testing data in the hidden space of AE and MAE. Fig. 5 (a) and Fig. 5 (b) show that the representation of AE still can distinguish the normal samples, known attack samples and unknown attack samples. This is the main reason for the high performance of classifiers on the AE's representation presented in Table II. However, while MAE can compress normal and known attack samples into two very compact areas on both the training and testing data, AE does not obtain this result. The normal and known attacks in the training data of AE spread significantly wider than the samples of MAE. More interestingly, the samples of unknown attack in the testing data of MAE are mapped closely to the region of known attacks, and hence they can be distinguished from normal samples easier. By contrast, the samples of unknown attacks in AE are very close to the normal data, and hence they are difficult to separate from the normal samples (benign samples). This result evidences that our proposed model, i.e., MAE, achieves its objective in constraining the normal data and known attack data into two compact areas at the hidden space.

Moreover, the unknown attacks are also projected closely to the region of known attacks. Subsequently, both attacks (known and unknown) can be effectively identified using some simple classifiers applying on the latent features of MAE.

### B. Cross-datasets Evaluation

Among the two tested botnet families, the Gafgyt botnet family is a lightweight version of Internet Relay Chat model. Thus, the DDoS attacks in Gafgyt are often the traditional SYN, UDP, and ACK Flooding attacks [47]. However, the Mirai botnet is usually a more dangerous IoT malware. It can exploit devices based on several architectures, and it is capable of perpetrating a wide range of DDoS attacks based on different protocols (e.g., TCP, UDP, and HTTP) [47], [48]. As described in Section I, each botnet family, Gafgyt or Mirai, can generate several DDoS attacks. Different botnets can create different network traffic transmitted from *bots* to infected devices, which results in different feature values of attack data.

This experiment aims to exam the stability of the latent representation produced by MVAE, MAE and MDAE when training on one botnet family and evaluating on the other. We consider two scenarios: (1) training data is Mirai, and testing data is Gafgyt, and (2) Gafgyt is chosen for training, and Mirai is used for testing. These scenarios guarantee that the testing attack family has not been seen in the training phase. We use the NCT classifier for investigating our models, and the experimental results of NCT trained on IoT-2[6] are shown in Table III.

TABLE III: Results of the NCT classifier in the cross-datasets experiment. The second column represents the models trained on Gafgyt botnets and evaluated on **Mirai botnets**. In the third column, Mirai is used for training and Gafgyt is used for testing.

| Models | **Mirai botnet** | | | **Gafgyt botnet** | | |
|---|---|---|---|---|---|---|
| | AUC | FAR | MDR | AUC | FAR | MDR |
| STA | 0.747 | 0.002 | 0.504 | 0.747 | 0.002 | 0.504 |
| DBN | 0.732 | 0.003 | 0.433 | 0.720 | 0 | 0.671 |
| AE | 0.717 | 0.004 | 0.562 | 0.628 | 0 | 0.743 |
| **MVAE** | **0.943** | **0.006** | **0.107** | **0.999** | **0.002** | **0.002** |
| **MAE** | **0.974** | **0.010** | **0.042** | **0.999** | **0.001** | **0.001** |
| **MDAE** | **0.988** | **0.010** | **0.006** | **0.999** | **0.001** | **0.001** |

This table shows that when the training data and testing data come from different botnet families, it is difficult for the NCT classifier to detect unknown botnet attacks. Both the stand-alone NCT and NCT with the representation of AE and DBN, tend to produce a poor performance in both scenarios. The reason is that the AE and DBN can only capture useful information of the input data once they gather sufficient data information. In this case, the training attacks and testing attacks come from totally different botnet families. The trained AE and DBN may be unable to represent the attacks that have not been seen in the training phase, which results in a poor performance for the NCT classifier (as shown in the first three rows of Table III). On the other hand, the latent representations of MVAE, MAE and MDAE are designed to reserve some regions being close to the anomaly region for unknown IoT attacks. Thus, these AEs help NCT to identity unknown attacks more effectively, and perform well on both scenarios (as observed in the last three rows of Table III). For example, the AUC scores of NCT to predict the Mirai botnet increase from $0.747$ with the original data to $0.943$, $0.974$ and $0.988$ with representations of MVAE, MAE and MDAE, respectively. These results confirm that our learning representation models can enhance the ability to detect unknown IoT attacks of simple classifiers.

[6]Due to the space limitation, we only present the results of the NCT classifer on one dataset, i.e., IoT-2. The results of the other classifiers on the rest datasets are similar to the results in this subsection.

### C. Influence of Parameters

This subsection analyzes the impact of several important parameters to the performance of the proposed models. The analyzed parameters include the noise factors in MDAE, the hyper-parameters in SVM and NCT classifiers.

*1) Influence of the noise factor:* This experiment examines the impact of the noise factor on the MDAE's performance. In this paper, the Gaussian noise function in (4) is employed to add noise to the input of MDAE. We will analyze the characteristics of MDAE when the standard deviation $\sigma_{noise}$ of Gaussian function is varied in the range of $[0.0, 0.1]$ on the IoT-1 dataset. The value of the standard deviation $\sigma_{noise}$ presents the amount of information of the input data which is noised.

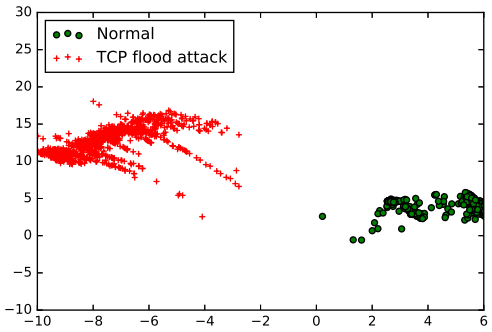
Fig. 6 presents the influence of the noise factor on MDAE observed by measuring the classification accuracy average of the four classifiers. This figure shows that the mean of AUC tends to be stable with $\sigma_{noise} \leq 0.01$, reaches a peak at $\sigma_{noise} = 0.01$, and then decreases gradually when $\sigma_{noise} > 0.01$. At the same time, a major part of the *FAR* and *MDR* curves go in the opposite direction. These results imply that MDAE achieves the best performance when the value of $\sigma_{noise}$ is 0.01.

*2) Influence of the Hyper-parameters of Classifiers:* This experiment investigates the influence of the hyper-parameters on the performance of classifiers when they are trained on the original feature space and the latent representation of five deep learning models including AE, DBN, MVAE, MAE and MDAE. We conduct experiments on two well-known classification algorithms, i.e., SVM and NCT[7]. The IoT-2 dataset is chosen for this experiment.

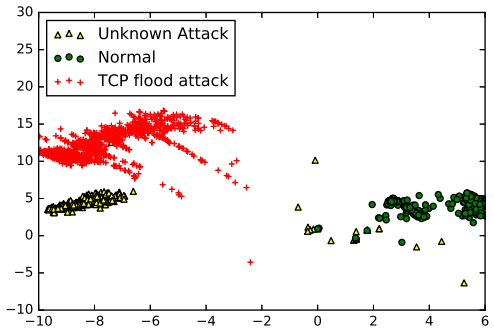
The first parameter is analyzed as the hyper-parameter $C$ of SVM. The hyper-parameter $C$ is a regularizer that controls the trade-off between complexity of decision plane and the frequency of error in the SVM algorithm [49]. This hyper-parameter presents the generalization ability to detect unseen anomalies of the SVM classifier. Fig. 7 (a) presents the influence of $C$ on the performance of SVM (AUC scores). It can be seen that the AUC scores of SVM training on the original feature space and the latent feature spaces of the AE and DBN vary considerably when $C$ is varied in the range from $10^{-4}$ to $10^{0}$. By contrast, the MVAE, MAE and MDAE models support the SVM to produce very high and consistent AUC scores over a wide range of $C$ values. It suggests that the proposed models generate more robust latent representations. It makes the SVM training process consistent/insensitive on a wide range of hyper-parameter settings.

The second parameter is the distance $metric$ used in the NCT classifier. The five distance metrics including $cosine$, $euclidean$, $manhattan$, $mahalanobis$ and $chebyshev$ are used to measure distances in NCT. The $metric$ hyper-parameter is

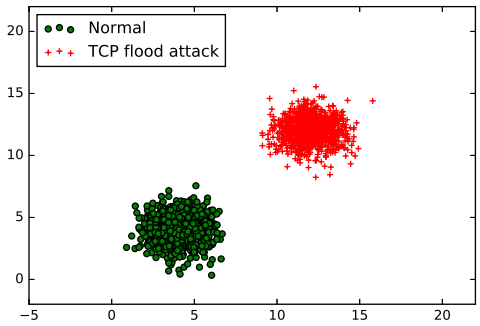
[7]The performance of SVM and NCT is often strongly influenced by some important hyper-parameters.



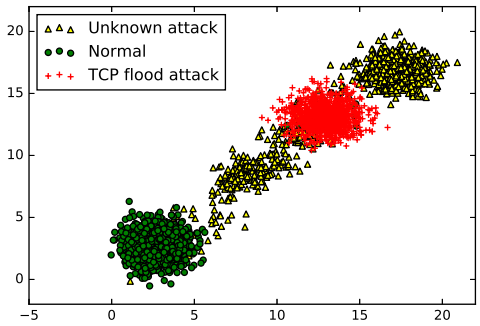

(a) AE representation of training samples.




(b) AE representation of testing samples.




(c) MAE representation of training samples.




(d) MAE representation of testing samples.

Fig. 5: Latent representation resulting from AE model (a,b) and MAE model (c,d).

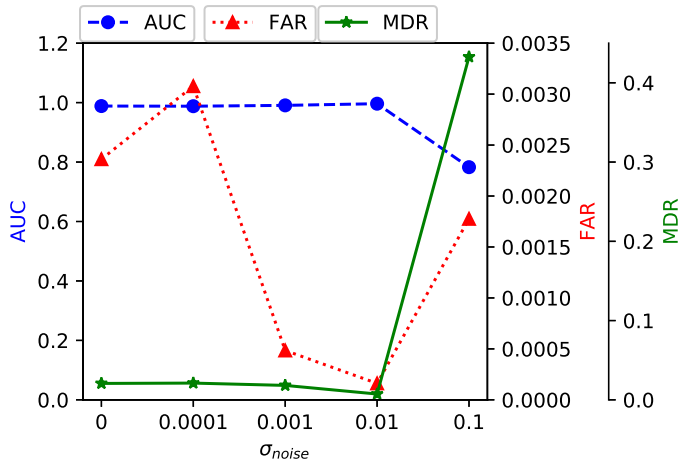


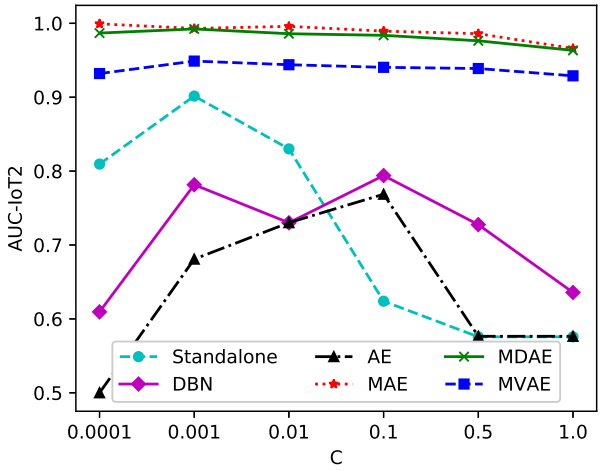

Fig. 6: Influence of noise factor on the performance of MDAE measuring by the average of AUC scores, FAR scores and MDR scores produced from SVM, PCT, NCT and LR on the IoT-1 dataset. The noise standard deviation value at $\sigma_{noise} = 0.01$ results in the highest AUC, and lowest FAR and MDR.

used to calculate distances between data samples in a feature array [34]. Fig. 7 (b) shows that the NCT classifier working with MVAE, MAE and MDAE tends to yield high and stable AUC scores over the five different values of the *metric* parameter. On the other hand, the AUC scores of NCT training on the original feature space and the feature spaces of AE and DBN are much lower and unpredictably changed with different values of *metric*.

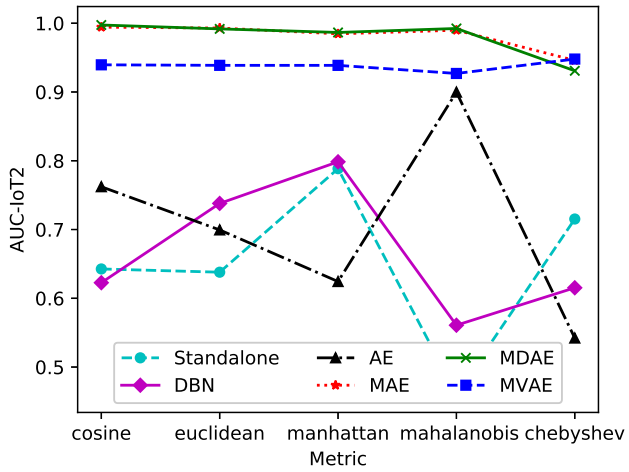
The experiments in this subsection clearly show that our proposed models (i.e., MVAE, MAE and MDAE) can support classifiers to perform consistently on a wide range of hyper-parameter settings. Perhaps, the reason for these results is that the latent representations of our models can map normal data and attacks into two separated regions. Thus, linear classifiers can easily distinguish attacks from normal data, and its performance is less sensitive to hyper-parameters.

### D. Assumptions and Limitations

Although our proposed models possess many advantages to learn a new representation of the IoTs datasets, they are subject to few limitations. First, we assume that there is sufficient data to train good models for learning representation. In this paper, the number of training samples for both normal and attack data are more than ten thousands samples. If we do not have enough data for training (only few hundreds), it will be difficult to train a good model for mapping input features to a new feature space. In the future, we will study techniques to generate synthesized data [50], [51] to train the models proposed in this paper. Second, the advantages of representation learning models come with the cost in running time. Since, a neural network is used to




(a) SVM.




(b) NCT.

Fig. 7: AUC scores of (a) the SVM classifier and (b) the NCT classifier with different parameters on the IoT-2 dataset.

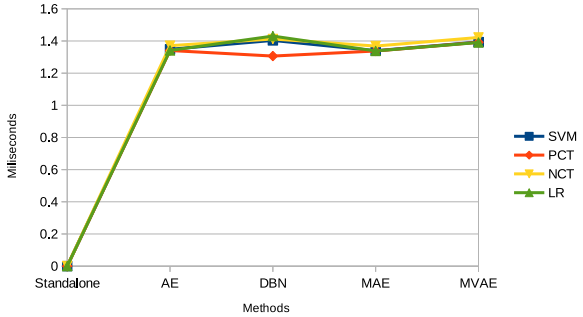



Fig. 8: Average testing time for one data sample of four classifiers with different representations on IoT-9.

project the input data into a new presentation, the executing time of these models is often much longer than using classifiers on the original feature spaces. Fig. 8 presents the average of time for processing one data sample of all tested methods. It can be seen that the processing time of all deep learning models are roughly equal and they are much longer than that of the standalone method.

IoT applications appear in many aspects of our life, such as smart city, home automation, data security [52], [53]. Specifically, alert systems in the data security applications require usage and pattern analysis for all data, across all systems, in real-time. Moreover, only stream processing is able to filter, aggregate and analyze continuous collection of data in milliseconds [53]. As a result, no behaviours of IoT traffic data get overseen or outdated. Thus, the running time of these models is still acceptable (about 1.3 milliseconds) for most applications in the real world.

## VII. Summary

In this paper, we have designed three novel AE based models for learning a new latent representation to enhance the accuracy in anomaly detection. As far as we known, our work is the first research that attempts to design regularized versions of AE to learn a latent representation in a supervised learning manner. In our models, normal data and known attacks are projected into two narrow separated regions in the latent feature space. In order to obtain such a latent representation, we have added new regularized terms to three AE versions resulting in three regularized models namely the MVAE, MAE and MDAE. These regularized AEs are trained on the normal data and known IoT attacks, and the bottleneck layer of the trained AEs was then used as the new feature space for linear classifiers.

We have carried out extensive experiments to evaluate the strength and examine different aspects of our proposed models. The experimental results demonstrate that our proposed models can map the non-linear separable normal and attacks data in the original space into linear and isolated data in their latent feature space. Moreover, the results also showed that unknown attacks tend to appear closely to known attacks in the latent space. The distribution of the data in the latent space makes the classification tasks much easier than executing them on the original feature space. Specifically, linear classifiers working with the latent feature produced from our models often significantly outperform those working with the original features and the features created by AE and DBN on the nine IoT attack datasets. The new data representation also helps the classifiers perform consistently when training on different datasets and varying a wide range of hyper-parameter settings.

In the future, one can extend our current work in several directions. First, the proposed models in this paper are only examined on two-class classification problems. In the future, it is interesting to extend these models and test them on multi-class classification problems. Second, the distribution centroid ($\mu_{y^i}$ in (8)) are currently determined through trial and error. It is desirable to find an approach to automatically select good values for each dataset. Last but not least, the regularized AE models are only tested on a number of IoT datasets. It is also more comprehensive to experiment them on a wider range of problems.

## REFERENCES


[1] N. C. Luong, D. T. Hoang, P. Wang, D. Niyato, D. I. Kim, and Z. Han, "Data collection and wireless communication in internet of things (IoT) using economic analysis and pricing models: A survey," *IEEE Communications Surveys Tutorials*, vol. 18, no. 4, pp. 2546–2590, Fourthquarter 2016.

[2] I. Ahmed, A. P. Saleel, B. Beheshti, Z. A. Khan, and I. Ahmad, "Security in the internet of things (IoT)," in *2017 Fourth HCT Information Technology Trends (ITT)*, Oct 2017, pp. 84–90.

[3] Y. Meidan, M. Bohadana, A. Shabtai, M. Ochoa, N. O. Tippenhauer, J. D. Guarnizo, and Y. Elovici, "Detection of unauthorized IoT devices using machine learning techniques," *arXiv preprint arXiv:1709.04647*, 2017.

[4] Y. Meidan, M. Bohadana, Y. Mathov, Y. Mirsky, A. Shabtai, D. Breitenbacher, and Y. Elovici, "N-baiot—network-based detection of IoT botnet attacks using deep autoencoders," *IEEE Pervasive Computing*, vol. 17, no. 3, pp. 12–22, Jul 2018.

[5] C. Zhang and R. Green, "Communication security in internet of thing: Preventive measure and avoid ddos attack over IoT network," in *Proceedings of the 18th Symposium on Communications & Networking*, ser. CNS '15. San Diego, CA, USA: Society for Computer Simulation International, 2015, pp. 8–15.

[6] C. Dietz, R. L. Castro, J. Steinberger, C. Wilczak, M. Antzek, A. Sperotto, and A. Pras, "IoT-botnet detection and isolation by access routers," in *2018 9th International Conference on the Network of the Future (NOF)*, Nov 2018, pp. 88–95.

[7] M. Nobakht, V. Sivaraman, and R. Boreli, "A host-based intrusion detection and mitigation framework for smart home IoT using openflow," in *2016 11th International Conference on Availability, Reliability and Security (ARES)*, Aug 2016, pp. 147–156.

[8] J. M. Ceron, K. Steding-Jessen, C. Hoepers, L. Z. Granville, and C. B. Margi, "Improving IoT botnet investigation using an adaptive network layer," *Sensors (Basel)*, vol. 19, no. 3, p. 727, 2019.

[9] V. L. Cao, M. Nicolau, and J. McDermott, "A hybrid autoencoder and density estimation model for anomaly detection," in *International Conference on Parallel Problem Solving from Nature*. Springer, 2016, pp. 717–726.

[10] V. L. Cao, M. Nicolau, and J. McDermott, "Learning neural representations for network anomaly detection," *IEEE Transactions on Cybernetics*, vol. 49, no. 8, pp. 3074–3087, Aug 2019.

[11] S. Khattak, N. R. Ramay, K. R. Khan, A. A. Syed, and S. A. Khayam, "A taxonomy of botnet behavior, detection, and defense," *IEEE Communications Surveys Tutorials*, vol. 16, no. 2, pp. 898–924, Second 2014.

[12] H. Bahşi, S. Nõmm, and F. B. La Torre, "Dimensionality reduction for machine learning based IoT botnet detection," in *2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, Nov 2018, pp. 1857–1862.

[13] S. S. Chawathe, "Monitoring IoT networks for botnet activity," in *2018 IEEE 17th International Symposium on Network Computing and Applications (NCA)*, Nov 2018, pp. 1–8.

[14] S. Nomm and H. Bahsi, "Unsupervised anomaly based botnet detection in IoT networks," *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pp. 1048–1053, 2018.

[15] O. Y. Al-Jarrah, O. Alhussein, P. D. Yoo, S. Muhaidat, K. Taha, and K. Kim, "Data randomization and cluster-based partitioning for botnet intrusion detection," *IEEE Transactions on Cybernetics*, vol. 46, no. 8, pp. 1796–1806, Aug 2016.

[16] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, no. 3, pp. 15:1–15:58, Jul. 2009. [Online]. Available: http://doi.acm.org/10.1145/1541880.1541882

[17] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "Network anomaly detection: Methods, systems and tools," *IEEE Communications Surveys Tutorials*, vol. 16, no. 1, pp. 303–336, First 2014.

[18] R. Chalapathy and S. Chawla, "Deep learning for anomaly detection: A survey," *arXiv preprint arXiv:1901.03407*, 2019.

[19] L. Vu, V. L. Cao, Q. U. Nguyen, D. N. Nguyen, D. T. Hoang, and E. Dutkiewicz, "Learning latent distribution for distinguishing network traffic in intrusion detection system," in *2019 IEEE International Conference on Communications (ICC)*, May 2019, pp. 1–6.

[20] Z. Li, Z. Qin, K. Huang, X. Yang, and S. Ye, "Intrusion detection using convolutional neural networks for representation learning," in *International Conference on Neural Information Processing*. Springer, 2017, pp. 858–866.

[21] Wei Wang, Ming Zhu, Xuewen Zeng, Xiaozhou Ye, and Yiqiang Sheng, "Malware traffic classification using convolutional neural network for representation learning," in *2017 International Conference on Information Networking (ICOIN)*, Jan 2017, pp. 712–717.

[22] M. Lotfollahi, M. J. Siavoshani, R. S. H. Zade, and M. Saberian, "Deep packet: A novel approach for encrypted traffic classification using deep learning," *Soft Computing*, pp. 1–14, 2019.

[23] S. E. Chandy, A. Rasekh, Z. A. Barker, and M. E. Shafiee, "Cyberattack detection using deep generative models with variational inference," *Journal of Water Resources Planning and Management*, vol. 145, no. 2, p. 04018093, 2018.

[24] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in *Advances in neural information processing systems*, 2007, pp. 153–160.

[25] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[26] N. Japkowicz, C. Myers, and M. Gluck, "A novelty detection approach to classification," in *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 1*, ser. IJCAI'95, 1995, pp. 518–523.

[27] J. Dromard, G. Roudière, and P. Owezarski, "Online and scalable unsupervised network anomaly detection method," *IEEE Transactions on Network and Service Management*, vol. 14, no. 1, pp. 34–47, March 2017.

[28] H. Bahşi, S. Nõmm, and F. B. La Torre, "Dimensionality reduction for machine learning based IoT botnet detection," in *2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, Nov 2018, pp. 1857–1862.

[29] O. Ibidunmoye, A. Rezaie, and E. Elmroth, "Adaptive anomaly detection in performance metric streams," *IEEE Transactions on Network and Service Management*, vol. 15, no. 1, pp. 217–231, March 2018.

[30] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *Journal of Machine Learning Research*, vol. 11, no. Dec, pp. 3371–3408, 2010.

[31] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[32] B. Du, W. Xiong, J. Wu, L. Zhang, L. Zhang, and D. Tao, "Stacked convolutional denoising auto-encoders for feature representation," *IEEE Transactions on Cybernetics*, vol. 47, no. 4, pp. 1017–1027, April 2017.

[33] "Support Vector Machine," https://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html#sklearn.svm.LinearSVC.

[34] "Percetron," https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Perceptron.html.

[35] "Nearest Centroid," https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.NearestCentroid.html.

[36] "Linear Regression," https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html.

[37] "Scikit-learn," https://scikit-learn.org/stable/.

[38] S. D. D. Anton, S. Sinha, and H. Dieter Schotten, "Anomaly-based intrusion detection in industrial data with svm and random forests," in *2019 International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, 2019, pp. 1–6.

[39] J. Zhang, M. Zulkernine, and A. Haque, "Random-forests-based network intrusion detection systems," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 38, no. 5, pp. 649–659, Sept. 2008.

[40] R. Salakhutdinov and H. Larochelle, "Efficient learning of deep boltzmann machines," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 2010, pp. 693–700.

[41] Y. Kim, "Convolutional neural networks for sentence classification," *arXiv preprint arXiv:1408.5882*, 2014.

[42] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 2010, pp. 249–256.

[43] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[44] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in python," *Journal of machine learning research*, vol. 12, no. Oct, pp. 2825–2830, 2011.

[45] "Implementation of deep belief network," https://github.com/JosephGatto/Deep-Belief-Networks-Tensorflow.

[46] Jin Huang and C. X. Ling, "Using auc and accuracy in evaluating learning algorithms," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 3, pp. 299–310, March 2005.

[47] M. De Donno, N. Dragoni, A. Giaretta, and A. Spognardi, "Ddos-capable IoT malwares: Comparative analysis and mirai investigation," *Security and Communication Networks*, vol. 2018, 2018.

[48] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. A. Halderman, L. Invernizzi, M. Kallitsis, D. Kumar, C. Lever, Z. Ma, J. Mason, D. Menscher, C. Seaman, N. Sullivan, K. Thomas, and Y. Zhou, "Understanding the mirai botnet," in *26th USENIX Security Symposium (USENIX Security 17)*. USENIX Association, Aug. 2017, pp. 1093–1110.

[49] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[50] V. Raj, S. Magg, and S. Wermter, "Towards effective classification of imbalanced data with convolutional neural networks," in *IAPR Workshop on Artificial Neural Networks in Pattern Recognition*. Springer, 2016, pp. 150–162.
[51] L. Vu, C. T. Bui, and Q. U. Nguyen, "A deep learning based method for handling imbalanced problem in network traffic classification," in *Proceedings of the Eighth International Symposium on Information and Communication Technology*. ACM, 2017, pp. 333–339.
[52] K. Yasumoto, H. Yamaguchi, and H. Shigeno, "Survey of real-time processing technologies of iot data streams," *Journal of Information Processing*, vol. 24, no. 2, pp. 195–202, 2016.
[53] "Real-time stream processing for internet of things," https://medium.com/@exastax/real-time-stream-processing-for-internet-of-things-24ac529f75a3.

**Ly Vu** received her B.Sc. and M.Sc. degrees in computer science from Le Quy Don Technical University (LQDTU), Vietnam and Inha University, Korea, respectively. She is currently pursuing the Ph.D. degree with LQDTU. She was a Lecturer with Le Quy Don Technical University. Her research interests include data mining, machine learning, deep learning, network security.

**Van Loi Cao** Van Loi Cao received the B.Sc. and M.Sc. degree in computer science from Le Quy Don Technical University, Hanoi, Vietnam, and the Ph.D degree from University College Dublin, Dublin, Ireland. He was an Assistant Lecturer with the Le Quy Don Technical University, where he is currently a Lecturer of the Information Security Department, the Faculty of Information Technology. His current research interests include deep learning, machine learning, anomaly detection, and information security.

**Quang Uy Nguyen** received received B.Sc. and M.Sc. degree in computer science from Le Quy Don Technical University (LQDTU)), Vietnam and the PhD degree at University College Dublin, Ireland. Currently, he is a senior lecturer at LQDTU and the director of Machine Learning and Applications research group at LQDTU. His research interest includes Machine Learning, Computer Vision, Information Security, Evolutionary Algorithms and Genetic Programming.

**Diep N. Nguyen** is a faculty member of the Faculty of Engineering and Information Technology, University of Technology Sydney (UTS). He received M.E. and Ph.D. in Electrical and Computer Engineering from the University of California San Diego (UCSD) and The University of Arizona (UA), respectively. Before joining UTS, he was a DECRA Research Fellow at Macquarie University, a member of technical staff at Broadcom (California), ARCON Corporation (Boston), consulting the Federal Administration of Aviation on turning detection of UAVs and aircraft, US Air Force Research Lab on anti-jamming. He has received several awards from LG Electronics, University of California, San Diego, The University of Arizona, US National Science Foundation, Australian Research Council. His recent research interests are in the areas of computer networking, wireless communications, and machine learning application, with emphasis on systems' performance and security/privacy. He is an Associate Editor of IEEE Transactions on Mobile Computing, IEEE Access (special issue), and a Senior Member of IEEE.

**Dinh Thai Hoang** (M'16) is currently a faculty member at the School of Electrical and Data Engineering, University of Technology Sydney, Australia. He received his Ph.D. in Computer Science and Engineering from the Nanyang Technological University, Singapore, in 2016. His research interests include emerging topics in wireless communications and networking such as ambient backscatter communications, vehicular communications, cybersecurity, IoT, and 5G networks. He is an Exemplary Reviewer of IEEE Transactions on Communications in 2018 and an Exemplary Reviewer of IEEE Transactions on Wireless Communications in 2017 and 2018. Currently, he is an Editor of IEEE Wireless Communications Letters and IEEE Transactions on Cognitive Communications and Networking.

**Eryk Dutkiewicz** received his B.E. degree in Electrical and Electronic Engineering from the University of Adelaide in 1988, his M.Sc. degree in Applied Mathematics from the University of Adelaide in 1992 and his PhD in Telecommunications from the University of Wollongong in 1996. His industry experience includes management of the Wireless Research Laboratory at Motorola in early 2000's. Prof. Dutkiewicz is currently the Head of School of Electrical and Data Engineering at the University of Technology Sydney, Australia. He is a Senior Member of IEEE. He also holds a professorial appointment at Hokkaido University in Japan. His current research interests cover 5G and IoT networks.