Cluster Comput (2019) 22:S949–S961
https://doi.org/10.1007/s10586-017-1117-8

A survey of deep learning-based network anomaly detection

Donghwoon Kwon[1] · Hyunjoo Kim[2] · Jinoh Kim[1] · Sang C. Suh[1] · Ikkyun Kim[2] · Kuinam J. Kim[3]

Received: 30 April 2017 / Revised: 27 July 2017 / Accepted: 10 August 2017 / Published online: 27 September 2017

Abstract A great deal of attention has been given to deep learning over the past several years, and new deep learning techniques are emerging with improved functionality. Many computer and network applications actively utilize such deep learning algorithms and report enhanced performance through them. In this study, we present an overview of deep learning methodologies, including restricted Bolzmann machine-based deep belief network, deep neural network, and recurrent neural network, as well as the machine learning techniques relevant to network anomaly detection. In addition, this article introduces the latest work that employed deep learning techniques with the focus on network anomaly detection through the extensive literature survey. We also discuss our local experiments showing the feasibility of the deep learning approach to network traffic analysis.

✉ Hyunjoo Kim
hjookim@etri.re.kr

✉ Kuinam J. Kim
kuinamj@gmail.com

Donghwoon Kwon
donghwoon.kwon@tamuc.edu

Jinoh Kim
jinoh.kim@tamuc.edu

Sang C. Suh
sang.suh@tamuc.edu

Ikkyun Kim
ikkim21@etri.re.kr

1 Department of Computer Science and Information Systems, Texas A&M University, Commerce, Commerce, USA

2 Information Security Research Division, Electronics & Telecommunications Research Institute, Daejeon, Korea

3 Department of Convergence Security, Kyonggi University, Suwon, Korea

Keywords Network anomaly detection · Deep learning · Network traffic analysis · Intrusion detection · Network security

1 Introduction

According to the Internet Security Threat Report, abbreviated as the ISTR, approximately 430 million new malware variants, 362 Crypto-ransomware, and other Internet threats were found in 2015 [1]. Cybercrime and threat activities have become a critical part of our daily life, and the importance of network security has continuously emerged as a major concern. A recent ransomware "WannaCry" that hit over 10,000 organizations in over 150 countries on May 12th, 2017 [2], shows the increasing severity of the emerging cyber-attacks.

Intrusion detection (ID) is the core element for network security [3]. The main objective of ID is to identify abnormal behaviors and attempts caused by intruders in the network and computer system, and it is a big challenge to design and implement an intrusion detection system (IDS) meeting the objective [3,4]. Depending on *where* the intrusion detection performs, there exists two different types of IDS: host-based IDS (HIDS) and network-based IDS (NIDS). Depending on *how* the intrusion detection takes place, an IDS can implement *misuse detection* (based on signatures) and/or *anomaly detection* [3–6]. Basically, misuse detection is driven by known attacks, which are used to define patterns of malicious network activities, while anomaly detection is more suitable for detecting unknown attacks based on the profiles defining normal (and anomalous) behaviors. Needless to say, the important goal of these techniques is to recognize the intrusions for better preparation against future attacks [7].

A great deal of attention has been given to deep learning over the past several years for many applications in diverse areas, such as image processing, natural language processing, computer vision, and so forth [8,9]. In this work, we focus on investigating deep learning techniques employed for anomaly-based network intrusion detection. We survey the latest studies that utilize deep learning methods for network anomaly detection. In particular, this survey is more interested in the deep networks for unsupervised or generative learning (than deep networks for supervised learning and hybrid deep networks). In addition to the extensive literature survey, we introduce our local experiments with an fully connected network (FCN) model, which show the effectiveness of deep learning models to network traffic analysis.

This survey is organized as follows. Section 2 provides an overview of anomaly detection methodologies and a group of deep learning techniques with the KDDCup 1999-class datasets [10,11] widely used for the network anomaly detection study. Section 3 provides literature survey with the concentration on defining existing methodologies with the dataset used for evaluation. In Sect. 4, we discuss the effectiveness of deep learning for network anomaly detection with the introduction of our initial experiments using an FCN model. We conclude our presentation with a summary of this survey in Sect. 5.

2 Background

The definition of an Intrusion Detection System (IDS) is a device, process, and application that monitors system and network activity for unauthorized and malicious activity [12], and a variety of classification for anomaly detection in the IDS is possible as per the different criteria shown in Fig. 1.

Based on the classification schema shown in Fig. 1, it is an obvious fact that deep learning consisting of multiple algorithms refers to a machine learning technique. As just described, the most important term here is Classification, and defining classification needs to be examined.

The first step is to acquire raw data which will be a form of datasets, and these datasets will be reduced by one of the three important strategies, i.e. dimensionality reduction, clustering, and sampling. An algorithm known as a classifier is then used to implement classification. For the classifier, two important datasets such as a training dataset and a testing dataset are required. The purpose of the training dataset is to train the classifier, and the testing dataset is used for testing the performance of classifier. There is also an important dataset called a validation dataset for the purpose of tuning the parameters. One thing to keep in mind is that both training and validation datasets are the only datasets that should be used for training the classifier. This concept is depicted in Fig. 2.

2.1 Data reduction

Data reduction is one of the important concepts in the data mining field, and there are three well-known data reduction strategies:

2.1.1 Dimensionality reduction [13,14]

Dimensionality reduction or dimension reduction is aimed at reducing the number of random variables. There are two well-known methodologies such as feature selection and feature extraction. These two methodologies will be explained in detail later.

2.1.2 Clustering [14,15]

The fundamental concept of clustering is to organize objects into similar groups. A main task of clustering is exploratory data mining and is widely used in a variety of areas such as machine learning, pattern recognition, image analysis, information retrieval, and bioinformatics. Note that there are many choices of clustering definitions and clustering algorithms.

2.1.3 Sampling [14,16]

Sampling is mostly used in a statistic area to decide if a small set of samples represents the entire dataset.

Among those three data reduction methodologies, we realized that dimensionality reduction and clustering have been mostly adopted for data reduction. In addition, there are two representative methodologies, i.e. feature selection and feature extraction, for dimensionality reduction as described earlier. The most significant difference between those two is data transformation [14,17]. For instance, if a subset of the existing features (attributes) is not transformed, this means feature selection. In other words, if the original set of N features is given, it is a process to choose a subset of M features from the original set of N features $M < N$. The major roles of feature selection in machine learning are as follows: (1) To reduce the dimensionality of feature space, (2) To speed up a learning algorithm, (3) To improve the predictive accuracy of a classification algorithm and, (4) To improve the comprehensibility of the learning results.

On the other hand, the concept of feature extraction is to extract a set of new features from the original feature set through functional mapping. If n features (attributes), i.e. $A1, A2, \ldots, An$, are given, new features, $B1, B2, \ldots, Bn$ and $Bi = Fi(A1, A2, \ldots, An)$, where Fi means a mapping function, can be acquired. The biggest benefit of this methodology is to expect the minimum set of new features through a transformation.

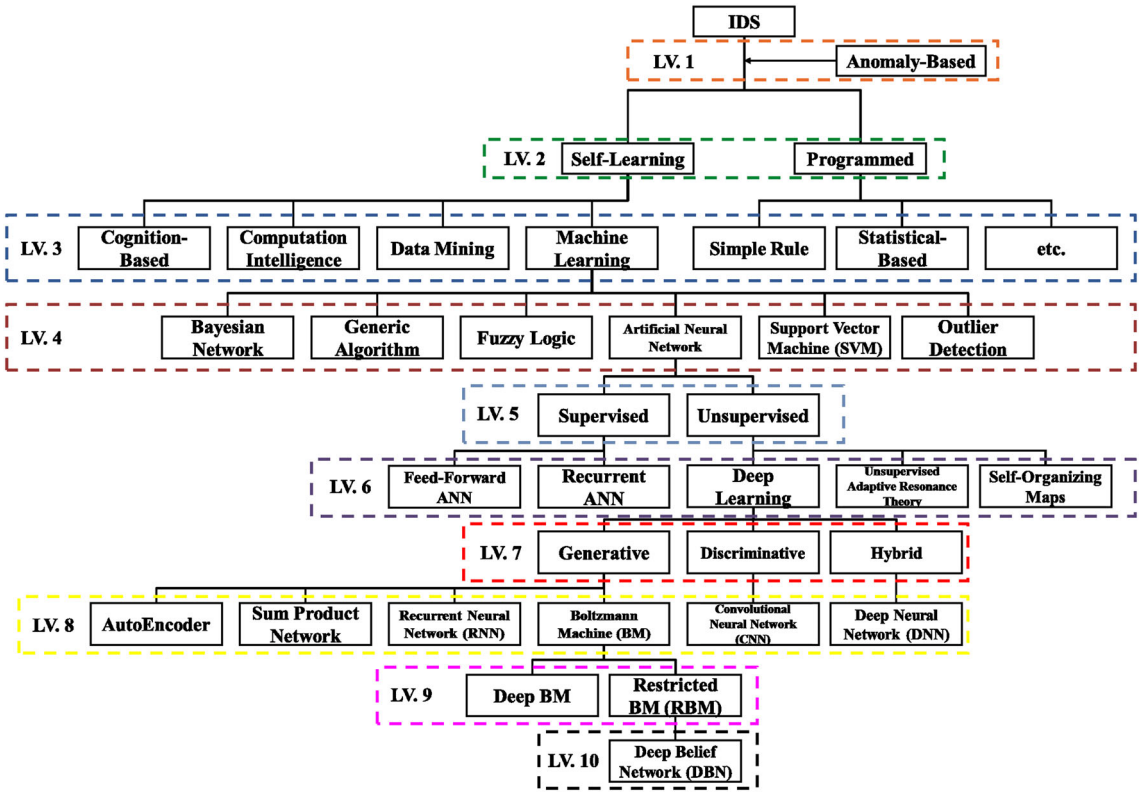

Fig. 1 Classification of anomaly detection

Fig. 2 Classification procedures

2.2 Techniques for dimensionality reduction

Seven state-of-the-art techniques to reduce input dimensionality are widely used, and the followings show such techniques: (1) principal component analysis (PCA) [18], (2) linear discriminant analysis (LDA), [19] (3) information gain attribute evaluation [14,20], (4) Gain Ratio Attribute Evaluation [14,20], etc. Keep in mind that both first and second techniques are used for dimensionality reduction as feature extraction, and other techniques are used for dimensionality reduction as feature selection.

2.2.1 PCA [21]

PCA refers to a statistical technique that aims at reducing the dimensions of a given unlabeled high-dimensional dataset. This technique is especially suitable for datasets in multiple dimensions such as image compression, pattern recognition, gene expression, network intrusion detection, etc.

2.2.2 LDA [19]

LDA is a popular method to reduce dimensionality. The main idea of this method is to provide a linear transformation of n-dimensional samples into m-dimensional space, so their relationship is $m < n$. For this reason, samples that belong to the same class are close together, however, samples from different classes are far from each other.

2.2.3 Information gain attribute evaluation [14,20]

One of the important tasks in machine learning and deep learning is to determine which attribute in a given dataset is the most important. This point refers to selection of the best

attribute, also known as feature ranking, and such selection is performed by the concept of entropy. The main formula of this method is:

$$\text{Info Gain}\,(Class, Attribute) = H\,(Class) - H\,(Class) - H\,(Class|Attribute) \quad (1)$$

where, H is the information entropy. This method is widely used for standard feature selection as one of the dimensionality reduction methods. Additionally, the fundamental concept is to evaluate the worth of an attribute by measuring the information gain regarding the class.

2.2.4 Gain ratio attribute evaluation [14,20]

This method is very similar with the previous one. It evaluates the worth of an attribute by measuring the gain ratio about the class using the:

$$\text{Gain R}\,(Class, Attribute) = \frac{(H\,(Class - H(Class|Attribute))}{H\,(Attribute)} \quad (2)$$

Note that there are other techniques of dimensionality reduction such as CfsSubset Attribute Evaluator (CFS), wrapper subset, and OneR attribute [14,22].

2.3 Classification

As described earlier, an algorithm known as a classifier is used to implement classification. Yet, the most important fact here is that classification refers to learning, also known as classification techniques [23]. One of the representative learning methodologies is Machine Learning, that is related to changes in the systems that perform tasks associated with artificial intelligence such as recognition, diagnosis, prediction, etc. [23]. There are three main types of learning, i.e. supervised learning, unsupervised learning, and semi-supervised Learning [23,24].

2.3.1 Supervised learning [24]

Supervised learning is known as predictive or directed classification so that a set of possible classes can be identified in advanced. In this learning, correctly classified data instances are given as a training set, and one of the supervised learning algorithms is employed for prediction in the next stage. Thus, this is appropriate when there is a specific target value. Furthermore, this learning type needs to have a subset of training datasets, which means both the input and desired results. New data is then classified based on the training set. The well-known supervised learning algorithms are support vector machines (SVMs), artificial neural network (ANN), logistic regression, nave Bayes (NB), K-nearest neighbors (KNN), random forests (RF), decision trees (DT), etc. [25].

2.3.2 Unsupervised learning [24]

Unsupervised learning is known as descriptive or undirected classification. If there are data without the desired output, it is called unsupervised. The well-known unsupervised learning algorithms are clustering, self-organizing map (SOM), deep learning, etc.

2.3.3 Semi-supervised learning [24,26]

Semi-supervised learning is a hybrid version of supervised learning and unsupervised learning so that both labeled and unlabeled data should be used for learning.

2.4 Deep learning

Deep learning has a variety of definitions, but the core and key words with respect to deep learning are as follows: unsupervised machine learning, artificial intelligence, learning multiple layers, etc. [27]. All these key words are closely related to neural network, pattern recognition, etc., and they refer to a wide class of machine learning techniques. In addition, depending on how the techniques can be used, deep learning can be categorized into one of the three classes such as: (1) deep networks for unsupervised or generative learning, (2) deep networks for supervised learning and, (3) hybrid deep networks. Yet, only the first class is considered in this survey.

Since labeled data are not available in the unsupervised learning process, its main focus is to generate labeled data by applying unsupervised learning algorithms such as restricted Boltzmann machine (RBM), deep Boltzmann machine (DBM), deep belief network (DBN), deep neural network (DNN), generalized denoising AutoEncoders, recurrent neural network (RNN), etc. The following describes basic deep learning terminologies and concepts in this survey.

2.4.1 RBM [28]

A RBM emerged due to the complexity of a general BM; in other words, units in each layer (visible and hidden) are connected so that it results in slow learning speed, as well as, long learning time [28]. For this reason, the fundamental idea of the RBM is to restrict connections between units in the same layer [29]. The difference between both models is shown in Fig. 3.

The RBM is an undirected, energy-based, and probabilistic generative model with two main objectives: (pre) training

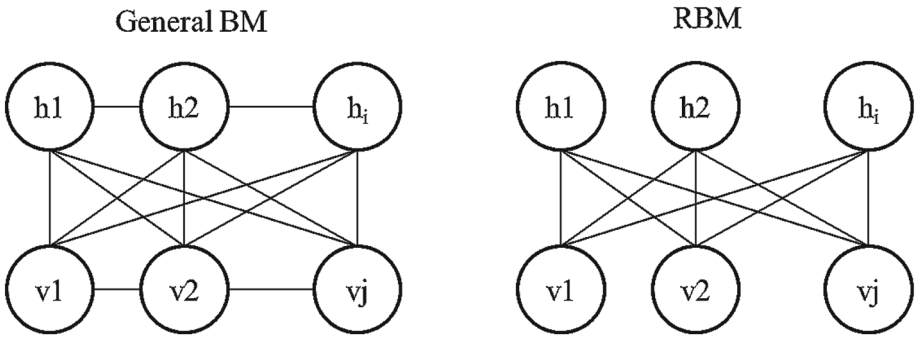

Fig. 3 Difference between general BM and RBM

and reconstruction [4,30,31]. The main idea of deep learning for classification is that it is an architecture-based which consists of hierarchical layers of non-linear processing steps, and its architecture can be augmented with n hidden layers [32]. Yet, the issue is that since the augmented neural networks cannot be well trained by back-propagation that is a common algorithm of neural network learning, the generative pre-training scheme should be used for solving such issue [32]. Thus, the proposed pre-training scheme is the RBM.

Based on Fig. 3, two layers such as visible and hidden have to be given, and units in each layer are connected each other [27,29,31,33]. Yet, there are no connections between units in a same layer. If there are v_i units in a visible layer, h_j units in a hidden layer, w_{ij} weights (or the strength of the connection) between the ith visible unit and the jth hidden unit, and biases (a, b), an energy function of the visible and hidden units is defined as: [3,27,30,33]

$$E(v,h) = -\sum_{i=1,i\in V}^{n} a_i v_i - \sum_{j=1,j\in V}^{m} b_j h_j - \sum_{i=1}^{n} v_i h_j w_{ij} \quad (3)$$

where v_i and h_j represent the binary states of the ith visible unit and jth hidden unit, and n and m are the number of visible and hidden units.

Then, a joint distribution, $p(v,h)$ over the visible and hidden layers can be defined as [17,18,30]:

$$p(v,h) = \frac{1}{Z} e^{-E(v,h)} \quad (4)$$

where Z is a normalization factor or partition function [18,30].

Z is computed by summing all possible pairs of a visible and a hidden vector, and conditional probabilities can be computed by the given energy function and joint distribution shown in Eq. 5 and 6 [17,18]:

$$p(h_j = 1|v) = \sigma\left(\sum_{i=1}^{n} w_{ij} v_i + a_i\right) \quad (5)$$

$$p(v_i = 1|h) = \sigma\left(\sum_{j=1}^{m} w_{ij} h_j + b_i\right) \quad (6)$$

where σ is a sigmoid function. Keep in mind that the energy formula and conditional probabilities above for a Bernoulli (visible)–Bernoulli (hidden) RBM.

One important thing here is how to adjust weights and get an unbiased sample. Studies introduced the simple formula of adjusting weights shown in Eq. 7 [3,30].

$$w_{ij} = \varepsilon\left(\langle v_i h_j\rangle data - \langle v_i h_j\rangle model\right) \quad (7)$$

where ε is a learning rate. An unbiased sample of $\langle v_i h_j\rangle data$ can be easily obtained based on Eq. 5 and 6. In contrast, Gibbs sampling based on iterations is used for getting an unbiased sample of $\langle v_i h_j\rangle model$, and updating all of the visible and hidden units in parallel based on Eqs. 5 and 6 is very slow [30]. For this reason, the research proposed a faster learning procedure, known as Contrastive Divergence learning (CD), shown in Eq. 8 [33]:

$$\Delta w_{ij} = \varepsilon\left(\langle v_i h_j\rangle data - \langle v_i h_j\rangle reconstruction\right) \quad (8)$$

2.4.2 DBN [27]

DBN is a probabilistic generative model consisting of multiple layers of stochastic and hidden variables. The relationship between RBM and DBN is interrelated because composing and stacking a number of RBMs enable many hidden layers to train data efficiently through activations of one RBM for further training stages [27]. This concept is a foundation of DBN shown in Fig. 4. Note that stacking RBM procedures is based on the layer-by-layer greedy learning strategy resulted from learning by either Gaussian-Bernoulli RBM or Bernoulli–Bernoulli RBM. As described in Sect. 2.4.1, the Bernoulli–Bernoulli RBM is for binary variables, but the Gaussian-Bernoulli RBM (GBRBM) is for continuous value data. If visible units v_i and hidden units h_j are given, the GBRBM is defined as [34]:

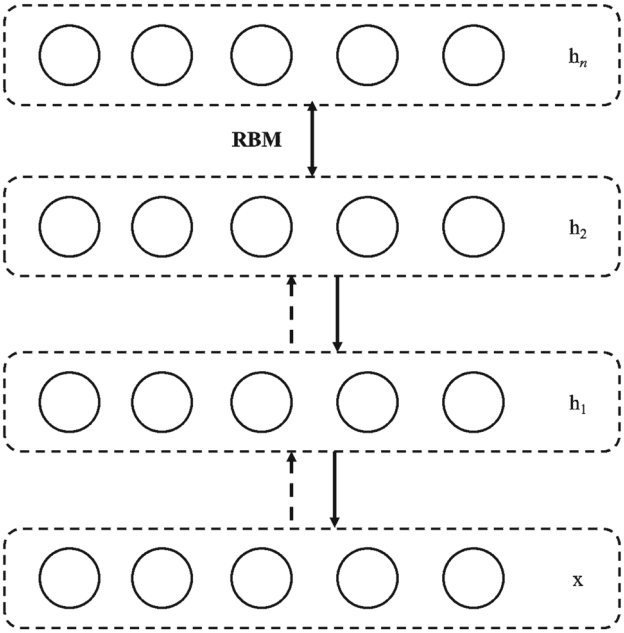

Fig. 4 The concept of DBN

$$E(v,h) = -\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij} h_j \frac{v_i}{\sigma_i} - \sum_{i=1}^{n} \frac{(v_i - a_i)^2}{2\sigma_i^2} - \sum_{j=1}^{m} b_j h_j \tag{9}$$

where σ_i is the standard deviation associated with v_i. Note that other mathematical symbols are same as the Bernoulli–Bernoulli RBM. Conditional probabilities for visible and hidden units can be also defined as Eqs. 10 and 11, respectively:

$$p(v_i = v|h) = N\left(v|a_i + \sum_j h_j w_{ij}, \sigma_i^2\right) \tag{10}$$

$$p(h_j = 1|v) = f\left(b_j + \sum_i w_{ij} \frac{v_i}{\sigma_i^2}\right) \tag{11}$$

2.4.3 DNN [27,35]

Many artificial intelligence (AI) fields adopt the DNN due to the superior performance. This performance results from an ability to extract high level features from raw data for a good representation of input data based on statistical learning. The concept of the DNN is based on a multilayer perceptron (MLP). The weights of the MLP are fully connected, and they are initialized by either the unsupervised or supervised pre-training technique. Yet, one of the biggest disadvantages in this model is computational complexity.

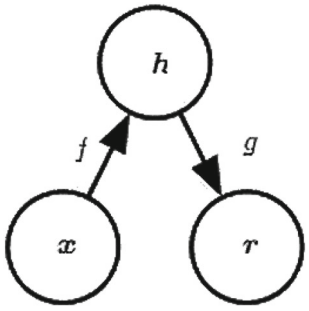

Fig. 5 The general structure of the AutoEncoder

2.4.4 AutoEncoder [36]

An AutoEncoder has been traditionally used for dimensionality reduction and feature learning. The fundamental idea of the AutoEncoder is that this is composed of a hidden layer h which refers to the input and two main parts, i.e. encoder function $h = f(x)$ and decoder, also known as reconstruction: $r = g(h)$. The main purpose of this concept is that both encoder and decoder are trained together, and the discrepancy between the original data and its reconstruction can be minimized. The general structure of the AutoEncoder is depicted in Fig. 5.

Additionally, Deep AutoEncoder is a part of an unsupervised model [27]. This is also considered as a generative model if a denoising criterion is used for training.

2.4.5 RNN [27]

The major issue of the standard neural networks is that all the data points are dependent. For this reason, if data points are time or space-related, the chance of losing the state of the network is high. An RNN, however, is a sequence-based so that it can model an input or output which is composed of independent sequence-based elements. The RNN can be used for either unsupervised or supervised. When this is used in the unsupervised learning mode, prediction of the data sequence from previous data samples is possible, but this concept sometimes causes the training difficulty.

2.5 Dataset for network anomaly detection

To the best of our knowledge, two datasets such as KDD-Cup 1999 dataset and NSL-KDD dataset have been popularly employed as training and testing datasets. In this section, both datasets will be briefly described.

2.5.1 KDDCup 1999 [10]

KDDCup 1999 dataset has been widely used for anomaly detection methods. Its training dataset is composed of

4,900,000 single connection vectors that contain 41 features and they are labeled as either normal or an attack. Attack labels (types) fall in one of the following four categories:

- *Denial of Service Attack* (*DoS*) This is an attack that intends for making computing or memory resources too busy.
- *User to Root Attach* (*U2R*) This attack type starts with access to a normal user account on the system, and then tries to exploit vulnerability to get local access.
- *Remote to Local Attack* (*R2L*) This attack occurs when an attacker, who does not have an account on the system, sends packets to the system over a network.
- *Probing* This is an attempt to gather information about computer networks for the purpose of circumventing its security controls.

2.5.2 NSL-KDD [11,37]

NSL-KDD dataset is not new, but it was emerged due to the disadvantage of the KDDCup 1999 dataset. Since the KDDCup 1999 dataset contains the huge amount of redundant records, approximately 75 and 78% are duplicated in the testing and training dataset, it makes the learning algorithm biased. To solve such issue, NSL-KDD, a new version of KDDCup 1999 dataset, is widely adopted for anomaly detection.

The NSL-KDD dataset contains four files, two files for training ("KDDTrain+" and "KDDTrain_20%") and the other two for testing ("KDDTest+" and "KDDTest-21"). Table 1 provides a summary of the four files in the dataset.

3 Literature review

Based on the background section above, this section mainly focuses on providing information about anomaly detection technologies in conjunction with machine learning techniques, particularly in the deep learning.

Alom et al. [30] pointed out the problems of the intrusion detection system in terms of S/W and H/W. S/W approaches like SNORT, BRO, etc. have millions lines of programming code to detect the malware entering in to user device, and H/W approaches involve usage of field programmable gate arrays (FPGA) and content address memory for speed classification of incoming packets. Even the application-specific integrated circuit (ASIC) are being designed for special application that could detect the malicious content using static or dynamic string matching techniques with the help of deterministic finite automata (DFA). However, the authors mentioned that S/W implementation provides high detection accuracy at the same high false positive, but human interference is required to distinguish between the actual intrusion and a false positive. The low flexibility, low scalability, and high cost of FPGA are the main H/W problems. Moreover, FPGA and ASIC-based approaches utilize the complementary metal oxide semiconductor (CMOS) logic. Due to such problems, the authors applied RBM-based DBN to intrusion detection using the NSL-KDD dataset. As a result of this approach, they achieved 97.5% accuracy for 40% of the training data of the entire dataset.

Li et al. [3] introduced various methods for malicious code detection such as NB, DT, SVM, and so on through literature research. However, the major problem was that feature extraction was not appropriately applied to malicious detection so that it caused low detection rate and accuracy. For this reason, this research proposed a hybrid malicious code detection model in conjunction with Auto-Encoder and DBN. In this research, AutoEncoder that consists of three steps such as pre-training, unrolling, and find-tuning was used for dimensionality reduction, and RBM was also used in this AutoEncoder methodology as a pre-training step. One important concept was that the RBM algorithm was driven by the energy formula and the weight updating formula as described earlier. After performing these two formulas, the pre-training was done by generating the RBM output which would be combined with the next RBM input data as the independent layer. The unrolling step was to connect these two independent RBM (the RBM output and the next RBM input) into a multi-layered AutoEncoder consisting of an encoder and decoder layer. The fine-tuning process did further adjustments to the initial weights which were used in the pre-training step. In this paper, the multiclass cross-entropy error function for weight evaluation was applied, and 10% of the samples of KDDCup 1999 as a dataset were accepted for experiment. The experimental results showed that with the increase in the number of pre-training and fine-

Table 1 NSL-KDD training and testing data set files [38]

File	Description	# data points	# normal	% anomaly (%)
Train+	Full NSL-KDD training set	125,973	67,343	46.5
Train20	A 20% of subset of the NSL-KDD training set	25,192	13,449	46.6
Test+	Full NSL-KDD testing set	22,544	9711	56.9
Test−	A subset of NSL-KDD testing set (more difficult)	11,850	2152	81.8

tuning iterations, in the respect of detection accuracy, the proposed method was superior to the method of single DBN. Apparently, using AutoEncoder to achieve data dimension reduction was effective and it could improve the detection accuracy.

Tao et al. [39] conducted research focusing on how to build a suitable data fusion algorithm in network security situation awareness (NSSA). Data fusion in NSSA aimed to eliminate the redundancy of big network traffic data by feature extraction, classification, and integration. The key component of the data fusion algorithm was feature extraction which was a preprocessing method to overcome dimension complexity, and its fundamental concept was that a few features could represent the original big data by analyzing internal characteristics. To do this, three well-known methods such as PCA, LDA, and Fisher score were used in this research. Additionally, this paper combined Fisher score with the unsupervised learning advantages of the Deep AutoEncoder algorithm. The experiment revealed two main results. The first experiment result was that the classification accuracy was improved once increasing the number of dimensionalities. Plus, all the algorithms such as J48, BPNN, and SVM performed well under more than eight dimensionalities. Therefore, the result indicated that the Deep AutoEncoder algorithm improved the generalization ability of the three algorithms by reducing data dimensionality. The second experiment result was about classification efficiency. The authors applied the Deep AutoEncoder algorithm to big network traffic data classification and compared it to three different algorithms such as J48, BPNN, and SVM in terms of classification efficiency. Data fusion based on this Deep AutoEncoder algorithm showed that the classification times of three algorithms were sharply declined. The Deep AutoEncoder algorithm remarkably improved the efficiency of big network traffic classification.

Niyaz et al. [5] introduced two categories of the NIDS: signature-based (SNIDS) and anomaly detection-based (ADNIDS). Focusing on ADNIDS characteristics, the authors pointed out that there were two issues to develop an effective and flexible NIDS for unknown future attacks. The first issue was that it was difficult to extract features from the network traffic dataset for anomaly detection. The second issue was that the labeled (or trained) traffic dataset from real network traffic was not available to develop such ADNIDS. For this reason, this research listed a variety of machine learning techniques for ADNIDS (i.e. ANN, SVM, NB, RF, Self-Organized Maps SOM, etc.), but the deep learning techniques used in this research were sparse AutoEncoder and soft-max regression (SMR). Based on those techniques, this research introduced two main steps in terms of feature extraction and supervised classification. The first step was to collect unlabeled network traffic data from different sources due to the objective that good feature extraction could be achieved by deep learning. Then, the next step was to apply the extracted features to, a labeled traffic dataset, which could be collected in a confined, isolated, and private network environment for supervised classification. The primary goal of this research was to evaluate the performance of deep learning based on metrics such as accuracy, precision, recall, and f-measure. To evaluate classification accuracy, this research evaluated the NSL-KDD training dataset using 2-class (normal and attack), 5-class (normal and four different attach categories), and 23-class (normal and twenty two different attacks). According to their evaluation, deep learning showed better accuracy performance for 2-class compared to SMR; however, there was no significant accuracy performance for 5-class and 23-class. The accuracy for 2-class was 88.39%, which was much higher than SMR (78.06%). It should be noted that the best accuracy using the NB-Tree methodology was 82%. Based on 2-class again, while deep learning achieved lesser precision compared to SMR, it achieved better recall values than SMR. Due to a higher recall value, deep learning outperformed SMR for the f-measure value. Deep learning achieved 90.4% f-measure value whereas SMR achieved only 76.8%. Similar observations were made for the 5-class as in the case of 2-class previously mentioned. The f-measure values for SMR and deep learning were 72.14 and 75.76%.

Salama et al. [4] conducted research for the anomaly intrusion detection scheme using RBM-based DBN. In this research, SVM classifier was followed by DBN which was used for feature reduction so that a hybrid scheme of DBN and SVM was the primary research methodology. This methodology consisted of three main phases: pre-processing, DBN feature reduction, and classification.

- *Pre-processing phase* Preprocessing of NSL-KDD dataset contains three processes: (1) Mapping symbolic features to numeric value, (2) Data scaling. It means that since the data have significantly varying resolution and ranges, the attribute data should be scaled to fall within the range 0 and 1 and, (3) Assign attack names to one of the five classes. It means that 0 for normal, 1 for DoS, 2 for U2R, 3 for R2L, and 4 for Probe.
- *DBN feature reduction (extraction)* The main idea of this process is to reduce data dimensionality with back-propagation that with intention with enhancing the reduced data output. So the BP-DBN structure consisting of two RBM layers is efficient to reduce data. The first RBM layer is used to reduce the data from 41 features to 13 features, and the second layer is used to reduce data from 13 features to 5 output features.
- *Classification* The five features output from DBN where passes to the SVM classifier are to be classified. The SVM technique is based on the Statistical Learning Theory (SLT).

One thing to keep in mind is that DBN can be used in two different ways, either as a dimensionality reduction method before SVM is applied or as a classifier by itself. As a result of evaluating the NSL-KDD dataset, the proposed methodology achieved classification accuracy improvement. Furthermore, the testing speed of this methodology was improved that was important for real time network applications. The second significant result was that the performance accuracy of DBN compared to different feature reduction methods such as PCA, Gain-Ratio, and Chi-Square was improved. The testing performance accuracy of the reduced data (from forty one features to thirteen features) using the SVM classifier showed better performance than the other reduction methods.

Kim et al. [40] pointed out that long short term memory (LSTM) was applied to RNN, and they used this concept for modeling the IDS. The major problem of the conventional RNN was that back propagation training time (BPTT) was used to handle with a variable-length sequence input, and the proposed model was initially trained with the training dataset in BPTT. Yet, the problem was that although the output error gradient was saved for each step, the RNN with BPTT algorithm made this error gradient vanish. Due to such problems, the authors mentioned that LSTM should be applied. 10% of the KDDCup 1999 training dataset and 10% of the KDDCup 1999 testing dataset were used in this research. Before performing experiments, the authors mentioned about finding a hyper-parameter value. The definition of the hyper-parameters is that they are parameters for model initiation. Depending on the hyper-parameter value, the performance changed, and the learning rate and hidden layer size had great effect on the performance. Based on this concept, the authors took experiments by changing the values of the learning rate and hidden layer size so that they found out that the detection rate and false alarm rate showed a growing trend as the learning rate was increased. More precisely, when they set 0.01 and 80 for the learning rate and hidden layer size, it showed the best efficiency in terms of the detection rate (0.877) and false alarm rate (0.133). The authors also measured the performance, and for evaluating this, they generated 10 test datasets from KDD, which had 5,000 instances in each dataset. According to their experiments, the average detection rate was 98.8% among the total attack instances, and the average false alarm rate was approximately 10%. This means that about 10% normal instances were misclassified. Furthermore, DoS and normal instances were well detected, but U2R instances were never detected due to not enough U2R instances (only 30 U2R instances were used).

Tank et al. [41] applied the DNN methodology to build up an IDS in a Software Defined Networking (SDN) environment using the NSL-KDD dataset. Only six features out of 41 features in the NSL-KDD dataset: duration, protocol_type, src_bytes, dst_bytes, count, and srv_count were chosen for experiments. To perform experiments, those six features were classified into two classes such as normal and attack, and then, authors tuned the hyper-parameter to optimize the research model. Note that the hyper-parameter in this research was a learning rate, and a different value of the learning rate in the range of 0.1–0.0001 was given to find out the best loss and accuracy; that is, minimize the loss and maximize the accuracy. In the training phase, the loss decreases and the accuracy increases as long as the learning rate decreases. Their experiments showed that the loss was 7.4% and the accuracy was 91.7% when the learning rate was 0.0001. However, experimental results in the testing phase were not good enough when the learning rate decreases. The loss was 20.3% and the accuracy was 74.6% when the learning rate was 0.0001. In addition, their accuracy was measured by the F-measure (also known as F1 score). In this case, the best learning rate was 0.001 with 75% of F1 score.

To summarize the literature review, it is possible to figure out common points, differences, and/or enhanced points. One of the common points is that the six research papers above used the same or similar dataset as the KDDCup 1999 and NSL-KDD datasets. Based on those datasets, it is possible to identify and group attack types into five categories such as DoS, U2R, R2L, Probe, and Normal. The most important point is that they mainly discussed methodologies in terms of data dimension reduction and classification focusing on unlabeled data. RBM itself, RBM-based DBN, AutoEncoder, Deep AutoEncoder, and LSTM-based RNN were used for data dimensionality reduction, and SVM and DBN were used as a classifier. Yet, the research using the DNN methodology manually chose the features in use. Thus, the extracted major idea from the literature review is that it is required to reduce data dimensionality using one of the methodologies as mentioned above and then, needs to apply it to one of the classification methodologies such as SVM, DBN, etc. By doing so, it is possible to evaluate data accuracy, detection rate, etc.

4 Discussion

In this section, we discuss the effectiveness of deep learning for network anomaly detection by introducing our initial experiments using a Fully Connected Network (FCN) model. We then briefly discuss the next step possible to explore for deep learning-based network anomaly detection.

In our initial experiments, we constructed a deep learning model based on FCN as shown in Fig. 6 using Python tensorflow in the Google cloud platform. In this model, the first step is the data preprocessing and transformation. Since the NSL-KDD dataset is composed of numerical and categorical values, we normalized numerical values by z-score and encoded categorical values as numerical values. The normalized and transformed data is then passed to the fully

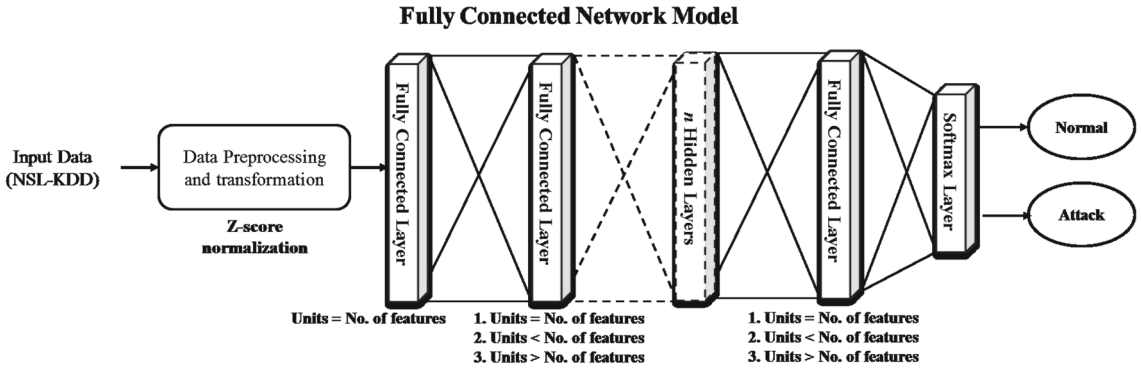

Fig. 6 Fully connected neural network model constructed in our experiment

Table 2 Experimental results with the NSL-KDD dataset using our FCN model

Combination	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)
Train+/Test+	89.4	87.7	94.6	91.0
Train20/Test+	90.4	88.6	95.3	91.8
Train+/Test−	83.0	87.2	92.9	90.0
Train20/Test−	84.2	87.7	93.8	90.7

connected network. In the fully connected layers, training is performed by the following hyper-parameter configurations: manipulation of (1) the number of units, (2) the number of hidden layers, (3) epochs, and (4) the learning rate. In our experiments, we set hyper-parameter configurations as follows: units = {1 unit, 10% of the entire features, 20% of the entire features, 40% of the entire features, and 100% entire features}; hidden layers = {1 and 3}; epochs = {40}; and learning rate = {1e−4 and 1e−5}. Based on those configurations settings, the FCN model performs training to find the best optimization, and outputs are produced through the Softmax layer.

With the introduced model and configuration settings, a set of experiments with four combinations of the NSL-KDD datasets (i.e., Train+, Train 20, Test+, and Test−) were conducted. Training+ is a full NSL-KDD training set, and Train 20 is a 20% of subset of the NSL-KDD dataset. Test+ is a full NSK-KDD test testing set, and Test− is a subset of the NSL-KDD test set (the detailed information of the files can be found from Table 1). Table 2 provides the experimental results with a set of measures. Our initial experiments using the FCN model show highly promising results with over 90% F1-score all across the combinations. Note that the conventional machine learning techniques, such as SVM, random forest, and Adaboosting, show the accuracy of 50.3–82.5% for the NSL-KDD data set [42]. Compared to this, we can see that a simple FCN model can improve performance significantly.

Besides FCN, there would be many other alternatives to form a deep learning network. Moreover, new deep learning techniques are being introduced armed with improved functionality. For instance, generative adversarial network (GAN) has been actively employed to detect anomalies in image data [43,44], but it has not been studied for network anomaly detection in great detail.

The basic idea of the GAN methodology is based on generative models which can be used in two different ways, i.e. to create a probability distribution (pmodel) using a training dataset or to generate samples from pmodel [45]. This methodology has several advantages. An interesting advantage is that GAN shows outstanding testing with training and sampling datasets from generative models. In addition, it is possible to train any missing data through prediction with the reinforced learning algorithms in the GAN model. Another advantage would be that GAN can work with machine learning to generate multi-modal outputs. Employing such new emerging deep learning technologies would improve the overall performance for network anomaly detection with greater accuracy.

5 Conclusion

In this survey, we investigated deep learning techniques employed for anomaly-based network intrusion detection, with the growing attention to deep learning nowadays in many areas. An overview of anomaly detection methodologies have been introduced with the topics of data reduction, dimensionality reduction, classification, as well as a group of deep learning techniques. A significant body of past work in network anomaly detection using deep learning methods has been discussed with their specific interest and potential limitations. This survey also investigated the effectiveness of deep learning models to network traffic analysis by introduc-

ing our local experiments with an FCN model, which show highly promising results with improved accuracy to detect anomalies compared to the conventional machine learning techniques such as SVM, random forest, and Adaboosting.

Acknowledgements The authors are grateful to Ritesh Malaiya for his assistance for experimenting. This work was supported in part by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIP) (No. 2016-0-00078, Cloud-based Security Intelligence Technology Development for the Customized Security Service Provisioning).

References

1. Semente: 2016 Internet Security Threat Report (ISTR), vol. 21, p. 8, April 2016
2. Gartner Provides Three Immediate Actions to Take as WannaCry Ransomware Spreads. http://www.gartner.com/newsroom/id/3715918
3. Li, Y., Ma, R., Jiao, R.: Hybrid malicious code detection method based on deep learning. Int. J. Secur. Appl. **9**(5), 205–216 (2014)
4. Salama, M.A., Eid, H.F., Ramadan, R.A., Darwish, A., Hassanien, A.E.: Hybrid intelligent intrusion detection scheme. Soft Comput. Ind. Appl. **96**, 293–303 (2011)
5. Niyaz, Q., Sun, W., Javaid, A.Y., Alam, M.: A deep learning approach for network intrusion detection system. In: 9th EAI International Conference on Bio-Inspired Information and Communications Technologies, pp. 1–11, May 2016
6. Ahmed, A.: Signature-based network inrusion detection system using JESS(SNIDJ). Graduate Project Technical Report, TAMUCC, pp. 2–6 (2004)
7. Ning, P., Jajodia, S.: Intrusion detection techniques. The Internet Encyclopedia. doi:10.1002/047148296X.tie097
8. Najafabadi, M.M., Villanustre, F., Khoshgoftaar, T.M., Seliya, N., Wald, R., Muharemagic, E.: Deep learning applications and challenges in big data analytics. J. Big Data **2**(1), 1 (2015)
9. Deng, L., Yu, D.: Deep learning: methods and applications. Found. Trends Signal Process. **7**(3–4), 197–387 (2014)
10. Tavallaee, M., Bagheri, E., Lu, W., Ghorbani, A.A.: A detailed analysis of the KDD CUP 99 dataset. In: Proceedings of the 2009 IEEE Symposium on Computational Intelligence in Security and Defense Applications (CISDA 2009), pp. 53–58 (2009)
11. Revathi, S., Malathi, A.: A detailed analysis on NSL-KDD dataset using various machine learning techniques for intrusion detection. Int. J. Eng. Res. Technol. **2**(12), 1848–1853 (2013)
12. Vinchurkar, D.P., Reshamwala, A.: A review of intrusion detectiom system using neural network and machine learning technique. Int. J. Eng. Sci. Innov. Technol. **1**(2), 54–63 (2012)
13. Das, S., Kalita, H.K.: Advanced dimensionality reduction method for big data. In: Research advances in the integration of big data and smart computing, information science reference (an imprint of IGI global), p. 200 (2016)
14. Panwar, S.S., Raiwani, Y.P.: Data reduction techniques to analyze NSL-KDD Dataset. Int. J. Comput. Eng. Technol. **5**(10), 21–31 (2014)
15. Jain, A.K.: Data clustering: 50 years beyond K-means. J. Pattern Recognit. Lett. **31**(8), 651–666 (2010)
16. John, G.H., Langley, P.: Static versus dynamic sampling for data mining, KDD 96. In: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, pp. 367–370 (1996)
17. Motoda, H., Liu, H.: Feature selection, extraction, and construction. Commun. Inst. Inf. Comput. Mach. Taiwan **5**(2), 67–72 (2002)
18. Elrawy, M.F., Abdelhamid, T.K., Mohamed, A.M.: IDS in telecommunication network using PCA. Int. J. Comput. Netw. Commun. **5**(4), 147–157 (2013)
19. Datti, R., Lakhina, S.: Performance comparison of features reduction techniques for intrusion detection system. Int. J. Comput. Sci. Technol. **3**(1), 332–335 (2012)
20. Bajaj, K., Arora, A.: Dimension reduction in intrusion detection features using discriminative machine learning approach. Int. J. Comput. Sci. Issues **10**(4), 324–328 (2013)
21. Ibraheem, N.B., Jawhar, M.M.T., Osman, H.M.: Principle components analysis and multi-layer perceptron based intrusion detection system. In: Fifth Scientific Conference Information Technology, vol. 10(1), pp. 127–135 (2013)
22. Chae, H., Jo, B., Choi, S., Park, T.: Feature selection for intrusion detection using NSL-KDD. In: Proceedings of the 12th WSEAS International Conference on Information Security and Privacy, pp. 184–187, November 2013
23. Namratha, M., Prajwala, T.R.: A comprehensive overview of clustering algorithms in pattern recognition. IOSR J. Comput. Eng. **4**(6), 23–30 (2012)
24. Koturwar, P., Girase, S., Mukhopadhyay, D.: A survey of classification techniques in the area of big data. Int. J. Adv. Found. Res. Comput. **1**(11), 1–7 (2014)
25. Caruana, R., Niculescu-Mizil, A.: An empirical comparison of supervised learning algorithms. In: Proceedings of the 23rd International Conference on Machine Learning, pp. 161–168, June 2006
26. Lin, F., Cohen, W.W.: Semi-supervised classification of network data using very few labels. In: Proceedings of the 2010 International Conference on Advances in Social Networks and Mining, pp. 192–198, August 2010
27. Deng, L., Yu, D.: Deep learning methods and applications. Found. Trends Signal Process., **7**(3–4), 199–201, 217 (2014)
28. Hinton, G.E.: Boltzmann machine. Scholarpedia **2**(5), 1668 (2007)
29. Fischer, A., Igel, C.: Training restricted Boltzmann machines: an introduction. Pattern Recognit. **47**, 25–39 (2014)
30. Alom, M.Z., Bontupalli, V., Taha, T.M.: Intrusion detection using deep belief networks. Int. J. Monit. Surveill. Technol. Res. **3**(2), 35–56 (2015)
31. Kim, S.K., McMahon, P.L., Olulotun, K.: A large-scale architecture for restricted Boltzmann machines. In: Proceedings of the 2010 18th IEEE Annual International Symposium on Field-Programmable Custom Computing Machines, pp. 201–208, May 2010
32. Kang, M., Kang, J.: Intrusion detection system using deep neural network for in-vehicle network security. PLoS ONE **11**(6), e0155781 (2016). doi:10.1371/journal.pone.0155781e0155781
33. Hinton, G.E.: A practical guide to training restricted Boltzmann machines. UTML Technical Report 2010-003, University of Toronto, August 2010
34. Yamashita, T., Tanaka, M., Yoshida, E., Yamauchi, Y., Fujiyoshii, H.: To be Bernoulli or to be Gaussian, for a restricted boltzmann machine. In: 2014 22nd International Conference on Pattern Recognition (ICPR), pp. 1520–1525. IEEE (2014)
35. Sze, V., Chen, Y.-H., Yang, T.-J., Emer, J.: Efficient processing of deep neural networks: a tutorial and survey. arXiv preprint, arXiv:1703.09039 (2017)
36. Hinton, G.E., Salakhutdinov, R.: Reducing the dimensionality of data with neural networks. Science **313**, 504–507 (2006)
37. Kayack, H.G., Zincir-Heywood, A.N., Heywood, M.I.: Selecting features for intrusion detection: a feature relevance analysis on KDD 99 intrusion detection datasets. In: Proceedings of the 3rd Annual Conference on Privacy Security and Trust, October 2005
38. Tavallaee, M., Bagheri, E., Lu, W., Ghorbani, A.A.: A detailed analysis of the kdd cup 99 data set. In: CISDA 2009. IEEE Sym-

posium on Computational Intelligence for Security and Defense Applications, 2009, pp. 1–6. IEEE (2009)
39. Tao, X., Kong, D., Wei, Y., Wang, Y.: A big network traffic data fusion approach based on fisher and deep auto-encoder. Information **7**(2), 20 (2016)
40. Kim, J., Kim, J., Thu, H.L.T., Kim, H.: Long short term memory recurrent neural network classifier for intrusion detection. In: 2016 International Conference on Platform Technology and Service (PlatCon), pp. 1–5, Feb 2016
41. Tang, T.A., Mhamdi, L., McLernon, D., Zaidi, S.A.R., Ghogho, M.: Deep learning approach for network intrusion detection in software defined networking. In: 2016 International Conference on Wireless Networks and Mobile Communications (WINCOM), pp. 258–263. IEEE (2016)
42. Baek, S., Kwon, D., Kim, J., Suh, S., Kim, H., Kim, I.: Unsupervised labeling for supervised anomaly detection in enterprise and cloud networks. In: The 4th IEEE International Conference on Cyber Security and Cloud Computing (IEEE CSCloud 2017), July 2017
43. Schlegl, T., Seeböck, P., Waldstein, S.M., Schmidt-Erfurth, U., Langs, G.: Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. arXiv preprint, arXiv:1703.05921 (2017)
44. Xue, Y., Xu, T., Zhang, H., Long, R., Huang, X.: Segan: adversarial network with multi-scale *l*_1 loss for medical image segmentation. arXiv preprint, arXiv:1706.01805 (2017)
45. Goodfellow, I.: Nips 2016 tutorial: generative adversarial networks. arXiv preprint, arXiv:1701.00160 (2016)

Donghwoon Kwon received his D.Sc. degree in Applied Information Technology from Towson University. He has been working as an ad-interim assistant professor and adjunct faculty of Computer Science Department at Texas A&M University-Commerce, USA since January 2016. The areas of research interests are Internet of Things (IoT), Cloud Computing, Big Data with Machine Learning, and Mobile Application, as well as, IT / IS Project Management in conjunction with Software Engineering and Database Management. He has been participating in the smart connected car research project as a Principal Investigator (PI) in collaboration with Gyeongbuk Institute of IT Convergence Industry Technology (GITC) and Rovitek Inc. since August 2016 and developed the deep learning-based blind spot detection system using Python Tensorflow.

Hyunjoo Kim received her M.S. degree and Ph.D. degree in Computer Engineering from Sungkyunkwan University, Korea, in 2002 and 2016, respectively. She joined the Electronics and Telecommunications Research Institute (ETRI) in 2002. She is currently working as a senior member of the engineering staff of the Intelligent Security Research Group. Her research interests include the malware analysis, network security, cloud security and BigData analytics.

Jinoh Kim received his Ph.D. degree in Computer Science from University of Minnesota, Twin Cities. He is currently an Assistant Professor of Computer Science at Texas A&M University-Commerce. The areas of research interests span from systems and networks, including large-scale distributed systems, big-data computing and analytics, and network traffic measurement and analysis. Prior to that, he was a researcher at the Lawrence Berkeley National Laboratory for 2010-2011 and an Assistant Professor of Computer Science at Lock Haven University of Pennsylvania for 2011-2012. From 1991 to 2005, he was a researcher and a senior researcher at ETRI (a national lab in Korea) participating in various research projects in system/network management and security.

Sang C. Suh is currently a Professor and Head of Computer Science Department at Texas A&M University-Commerce, USA. He founded a transdisciplinary research center called Intelligent Cyberspace Engineering Lab (ICEL) to launch and carry out many transdisciplinary systems R&D projects. His research theme and major research thrusts of his ICEL spans around many interdisciplinary research topics including bio-informatics and biological sciences, knowledge discovery and engineering, human computer interaction, data mining, big data, visual analytics, and adaptive search engines. He has authored and published over a hundred peer-reviewed scientific articles, several book chapters and books in the areas of data mining, bioinformatics, cyber-physical systems, knowledge and data engineering, visual analytics, and adaptive search engines. He has chaired numerous technical sessions in many international conferences and served in the program committees of over twenty international conferences. He currently serves as the President of the Society for Design and Process

Science which is an international professional organization with focus on transdisciplinary research.

Ikkyun Kim received his M.S. and Ph.D. degrees in Computer Engineering from the Kyoung-Pook National in 1996 and 2009 respectively. He joined the Electronics and Telecommunications Research Institute in 1996. He was an invited researcher for the Purdue University from 2004 to 2005. He is currently working as a director of the Network Security Research Laboratory of the ETRI. His research interests are on the Network Security, Computer Network, Cloud Security and BigData Analytics.

Kuinam J. Kim is a professor of Information Security Department in the Kyonggi University, Korea. He received his Ph.D. and M.S. in Industrial Engineering from Colorado State University in 1994. His B.S. in Mathematics from the University of Kansas. He is Executive General Chair of the Institute of Creative and Advanced Technology, Science, and Engineering. His research interests include cloud computing, wireless and mobile computing, digital forensics, video surveillance, and information security. He is a senior member of IEEE.