IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Revue d'Intelligence Artificielle
Vol. 34, No. 2, April, 2020, pp. 215-226
Journal homepage: http://iieta.org/journals/ria

Review on Generative Deep Learning Models and Datasets for Intrusion Detection Systems

Gayatri Ketepalli*, Premamayudu Bulla

Department of IT, VFSTR (Deemed to be University), Guntur 522213, Andhra Pradesh, India

Corresponding Author Email: drbpm_it@vignan.ac.in

https://doi.org/10.18280/ria.340213

Received: 11 December 2019
Accepted: 28 February 2020

Keywords:
IDS, ANN, machine learning, deep learning, RNN

ABSTRACT

Intrusion detection systems (IDSs) play an essential role in defense of all networks and information systems around the world. IDS is one way of reducing malicious attacks. When attackers adjust their attack tactics and find alternative attack strategies, IDS must also develop through more advanced methods. Deep learning is a subfield of machine learning (ML) methods focused on learning results. A comprehensive review of various deep learning methods employed in IDSs is discussed first in this paper. Then a deep classification scheme is introduced, and the significant works recorded in the deep learning works are summarized. We performed a taxonomy survey of the deep architectures and algorithms accessible in these works and grouped such algorithms into three groups: hierarchical, composite, and generative. Afterward, a wide range of intrusion detection fields investigates selected deep learning applications. Finally, we address common types of datasets and frameworks.

1. INTRODUCTION

The health of machines and network systems has long been at the forefront of study. All IT departments have been reported to be highly critical and relevant in the area of information security, which cannot be overlooked. The three basic principles on which a secure system depends must be met. The IDS described intrusion detection as "the method to track and evaluate events occurring in a device or network, and for indications of intrusions, identified as attempts to breach the confidentiality, privacy, functionality, or the circumvention of a machine or network security mechanisms" [1, 2].

2. DEEP LEARNING APPROACHES

Deep learning is a subset of Artificial Intelligence ML (AI) networks that can learn from both labeled [3] and unlabeled data [4] in a supervised and unattended manner. Deep Learning is an AI feature that simulates the workings of the human brain to process information and to establish trends for the use of decision-making processes [5]. There is no single definition of deep learning, but most meanings emphasize the following aspects:

- Branch of ML.
- Models are typically nonlinear.
- Uses both supervised and unsupervised approaches to fit models to data.
- Models are neural network structures with numerous layers.

Based on the way structures and methods are designed for utilization, for example, recognition/classification [6] or synthesis/generation, the majority of the study in this field and the applied algorithms in the field of intrusion detection can be broadly categorized to three main categories that are [7]:

(1) Generative (unsupervised) [8].
(2) Discriminative (supervised) [9].
(3) Hybrid deep architecture [10].

The classification of deep learning approaches is illustrated in Figure 1 [11].

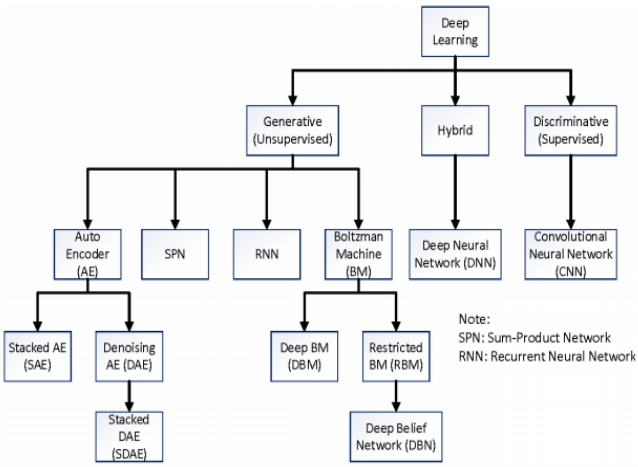

Figure 1. Taxonomy of deep learning methods

3. DEEP NETWORKS FOR UNSUPERVISED OR GENERATIVE LEARNING

Unsupervised learning, also known as generative architecture, uses unlabeled details. The main idea to add generative architectures to pattern recognition is pre-training or unattended knowledge [12]. Due to the difficulty in understanding the NIDS networks, deep generative structures are required. This is why, with a small amount of training data,

it is incredibly important to learn all the lower layers in the layer-by-layer system without relying on all the higher layers [13]. Many methods have been listed as unmonitored instruction in the following way.

3.1 Autoencoder (AE)

Deep AE is a specific form of unattended artificial NN whose output is the actual input of results. The primary purpose of AE is to use a forward-looking method to recreate an input-output. The data was compressed and then sent as an output, mostly identical to the initial input. This is the essence of an AE–to calculate and correlate related inputs and outputs with implementation outcomes. More specifically, the process of removing features is nonlinear and does not include class labels; hence generative [14, 15]. The AE requires three or more layers in the NN:

(1). The input data layer should be appropriately coded (for example, voice or pixel spectrums);

(2). Two or more hidden layers that shape the encoding are significantly smaller.

(3). An output layer in which each neuron has the same meaning as in the input layer.

Figure 2 shows the general structure of an AE, which maps the input x to an output (reconstruction) r via the internal representation or code h. There are two functions in the auto-encoder: the encoder f (maps from x to h) and the decoder g (maps from h to r) [15]. Aminento et al. [16] have implemented the SAE that is a classifier of deep learning algorithms for KDD99 Dataset; a system focused on the AE algorithm / stacked autoencoder (SAA). The solution presented revealed four distinct IDS: the device layer IDS-A, the transportation layer IDS-T, the network layer IDS-N and the data link layer IDS-L. Each IDS category is responsible for a number of network devices spread through computer networks.

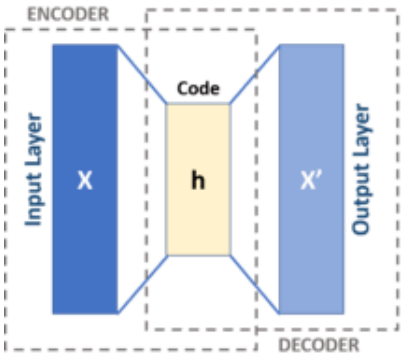

Figure 2. General Structure of Auto Encoder [15]

Every IDS class has its unique data set based on the TCP / IP layer property. E.g., the IDS-A dataset includes instances of standard attacks and layer attacks on applications. Furthermore, the role selection approach was extended to each dataset to pick the essential feature set for each IDS sort. Nevertheless, the IDS-T was only used as proof of concept (POC). Description of IDS-N, IDS-A, and IDS-L was not addressed in their article.

The researchers used ANN [17] as a tool for the collection of features. The ANN model shows a bias attribute for the secret sheet. The researchers used two secret (encoder) layers. They completed their stacked design using the SoftMax Regression Feature approach of supervised learning utilizing labels from training data. This research has shown that the IDS can be rendered by separating IDS into smaller sections and that the dimensionality of features and that the lightweight IDS can reach a comparable detection rate as the ordinary IDS. Nevertheless, it is still a difficult issue to introduce lightweight IDS for a wireless network.

Javaid et al. [18] using Self-taught Learning (STL), a deep learning method, for NSL-KDD data-set tests for intrusion into the network. We used the function learning based on Sparse Auto-Encoder (SAE) because it is easier to implement and is good [19]. STL classification was conducted in two stages: SAE for unmonitored functional learning and SoftMax classification for derived training data. The efficiency can be increased by using methods such as SAE and others. They did not apply their real-time NIDS methodology to real-world networks. We also indicated that, rather than derivative functionality, the function learning on-the-go on raw network traffic headers could be a research area of great importance in the future.

Mirsky et al. [20] introduced Kitsune in the other work: a plug and play NIDS that can learn how to track attacks on the LAN, without control, and in a manner that is adequately online. AE used to differentiate between normal and abnormal traffic patterns in the main algorithm of Kitsune (KitNET).

A function extraction system supports KitNET and determines a network channel template route effectively. The main contribution of this work was: 1) the latest AE-based NIDS (Kitsune), which is plug-in and lightweight for easy network devices.

2) An on-line solution in an unsupervised way for the auto group model (i.e., mapping properties to NN inputs). It has almost been checked on an IoT network, a video monitoring active IP camera network, and many specific attacks [20].

To upgrade IDS, Fahimeh Farahnakian et al. suggested a system for Deep Auto Encoder (DAE). They concluded that AE is the most inspiring paradigm in deep learning to extract features from the high-dimensional results [21]. The Deep Auto Encoder dependent ISD (DAE-IDS) they indicated is composed of four autoencoders that use the AE result at the current layer as an AE entry at the corresponding layer For DAE-IDS research, they have employed a greedy unattended level training methodology that helps improve the efficiency of the deep model. After the four auto-encoders have been equipped, a Soft-Max layer is used to identify and target the inputs. The KDDCUP 1999 data set was used to measure the performance of the DAE-IDS as the data set was primarily used to test the IDSs. The suggested approach achieved an accurate identification of 94.71% from the KDD-CUP 1999 test data collection of 10% [22]. In their future work, they discussed investigating how sparing limits are placed on AE and how SAE can be built to improve the efficiency of intrusion detection further.

Another research by Zhang et al. [17] suggested a broad enough IDS-based approach to learning. The IDS contains a Deep Auto-Encoder (DAE) task selection engine and a multi-layer perceptron (MLP) classification primarily. Another key in the selection of features is the inclusion of loss weights in different instances so that the user can select a small group of features that effectively represent attacks. Only these valuable characteristics are maintained after collection, and high performance is obtained with a rather lightweight grouping.

The utility of the proposed method was tested by experiments conducted with the UNSW-NB data set in which

12/202 properties are selected after the collection of functions, resulting in a selection ratio of 5.9%. Upon classification by using an MLP with two hidden layers, they obtained a high precision 98.80 percent identification. F score that represents attack detection performance has achieved 0.952. The approach has shown exciting potential for use in high-speed networks [17].

Nathan Shone et al. suggested a fundamentally innovative approach to allowing NIDS service in complex networks. The described model incorporates deep and superficial learning that can accurately interpret a broad range of network traffic. In fact, because of the classification capacity of stacked soft-max autoencoders, it is rather weak in comparison to other biased methods, including RF, K-NN, and SVMs. You also entered the power to control precision and time effectiveness (i.e., shallow learning) with the proposed non-symmetrical Deep Auto Encoder (NDAE) and Random Forest (RF). The assessment of the model using the GPU-enabled tensor-flow and the analyzes of KDDCup 1999 and NSLKDD data sets produced good performance. The algorithm has increased the accuracy and pace of the exercise by nearly 5% by up to 98.81% [14]. As future work, they have proposed to increase the ability of the model to handle zero-day attacks and then to expand their practical tests utilizing real-world backbone network traffic to show the properties of the expanded model [14].

3.2 Boltzmann machine (BM)

BM is an asymmetrically connected network of neurons, like modules that make stochastic judgments on whether or not they are on. BM has a basic learning algorithm, which helps them to find unusual characteristics in data sets made up of binary vectors [23, 24]. BM is used to address two very different computer functions. For a search problem, the relation weights were set and used for the cost optimization feature. BM stochastic dynamics then allow it to check binary state vectors that mean right optimization solutions.

For a learning function, the BM shows a group of binary data vectors and needs to find weights on the relations in a way that the data vectors are enough solutions for the optimization question that these weights describe. BM makes numerous small changes in weight to solve a research challenge, and every modification causes them to address several quest tasks. BM is classified mainly into two categories: The Extreme Boltzmann Machinery and the Regional Boltzmann Machinery (RBM) Machinery. When these RBMs are placed upside down, they are known as Deep Belief Networks (DBNs) [25]. The schematic relation of BM, RBM, and DBM is shown in Figure 3.

A BM is entirely related between and within the layers, while an RBM excludes the lateral relations in hidden and apparent layers. The random variables represented by secret units are, therefore, not conditional in terms of the circumstances of the exposed groups, and vice versa [26].

Gao et al. suggested an approach based on the DBN multilayer for the analysis of DoS assaults. DBN is made up of many RBMs. The RBM instruction is provided herein as an advance in the learning process. The learned functions of RBM are then used as input data to acquire RBM from the next layer of the DBN stack. On the KDD CUP 1999 data set, the effectiveness of the DBN approach was checked. The DBN model's identification accuracy was higher than the SVM and ANN methods [27].

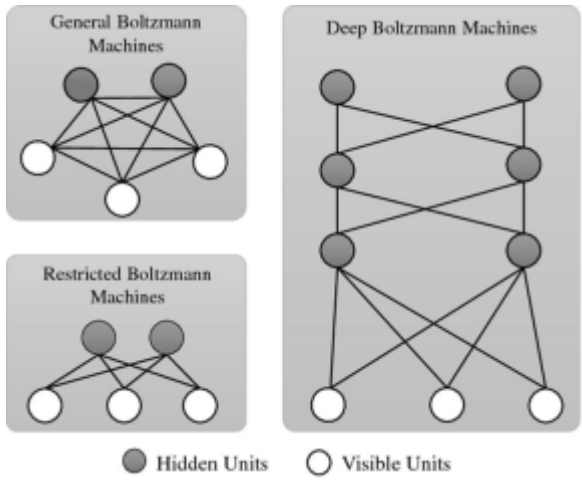

Figure 3. General structure of Boltzmann machines, restricted Boltzmann machines and deep Boltzmann machines

Seo et al. contrasted intrusion detection levels between the NIDS only using a classification model and the NIDS with data educated where the use of RBM excludes noise and outliers. Noise and outliers in the KDD Cup' 99 Information are extracted by RBM information implementation and new data creation. The research suggested a classification model training method to be able to detect network intrusions using the data regenerated based on the RBM features [28].

Zhang [24] used the two SVM, RBM, and DB hybrid algorithms. The algorithms were used for review of the false-positive values, precision, false-negative levels, and testing period, using the data set used for the Third International Competition for Knowledge Discovery and Information Mining Techniques (KDD Cup-99). DBN is more effective than the others, relative to one another, and the traditional hybrid intrusion-detection algorithm, in speed and precision both, owing to the unregulated learning of the RBM networks and the convergence of the NNs at the edges.

By contrasting the traditional model paired with NN and function selection, RBM-DBN has improved dramatically in terms of accuracy and the false definite limit. This is because independent learning has eliminated the disadvantages of the traditional NN and plays the role of feature extraction, which in terms of time costs and, therefore, is less time-consuming in the training paradigm compared to conventional ML. RBM-DBN is beneficial. RBM-DBN was able to solve the possible problem that large data samples carry into the model training and testing period, and that indicates that RBM-DBN is sufficient for significant data intrusion detection [24].

Alrawashdeh et al. [29] called a deep learning system for identifying phenomena using an RBM and a deep network of religions. Their solution was based on a1-hidden RBM layer for unattended function reductions. The resultant weights of the RBM are passed to another RBM that generates a deep trust network. The pre-trained weights are transferred to a fine-tuning layer composed of a soft-max Multiclass Logistic Regression (LR) classifier. Our design worked in terms of accuracy and detection speed better than the previous deep learning methods introduced by Salama et al. [30], and Li et al. [31]. In the complete KDD-CUP data set of 1999, they reached a detection rate equivalent to 97.9 percent. We proposed extending their ML approach as a potential expansion to broader and more demanding data sets, including a more extensive range of attacks.

Imamverdiyev and Abdullayeva [32] presented a comparative study of the precision of their proposed solution to the identification of DoS attacks with Bernoulli-Bernoulli, Gaussian–Bernoulli RBM. The identification precision of the methods in the NSL-KDD data set was checked. The suggested Gaussian–Bernoulli deep RBM multilayer precision was achieved. For the future, they proposed working with LSTM decoders and full and two-way LSTM encoders.

3.3 Recurrent neural network (RNN)

Besides, RNNs are NNs that use recurrence, which uses knowledge from a previous transmission over NN. Mostly, all RNNs can be viewed as recurrence. RNNs are acceptable and have been very useful when used in matters in which the data feedback on which the predictions have to be produced are in a context of a sequence (series of entities of interest in order) [8]. Figure 4 is the general RNN form, where hk indicates the input at phase k, and xk indicates the output [33].

Kim, J. and Kim, H. [34] applied recurrent neural networks to the Hessian free optimization IDS, a thoroughly experienced intrusion detection algorithm. They used the DARPA data set to train and check their intrusion detection model. It was used for the contest data-set KDDCup-99. The experimental results revealed that RNN with Hess-free optimization is a very efficient method for intrusion detection. We suggested more studies for the identification of current malware and attacks as a guide for potential work.

Kim et al. [35] have developed a deep-learning IDS platform. We used Long-Short-Term Memory (LSTM) on an RNN and equipped their IDS using a KDDCup-99 data set. For the training stage, by extracting samples from the KDDCup-99 data set and comparing the results with other IDS classifiers, they have noticed that the assaults are identified efficiently via the LSTM-RNN classifier. Because they have the highest precision and identification, although the false alarm rate is slightly higher than the other. Through the performance tests, the process of deep learning has been verified as adequate for IDS.

Yin et al. [36] presented the design and implementation of the recurrent NN detection system. Researchers have also analyzed the model output in binary and multi-class classifications, the number of neurons, and the different impact on performance on the learning scale. On the other side, they explored in the multi-class classification of the naive Bayes, multi-layer interpretation, random forest, SVM, and different approaches to ML on the 1999 KDD-Cup benchmark. They published a study of RNN-IDS performance and other ML approaches, both in binary and multi-class classifications.

The author's experimental results revealed that RNN-IDS is very appropriate for IDSs. The success of the RNN-IDS is higher than the conventional classification method on the 1999 KDD-Cup data set in each of the binary and multi-class classifications. Each of the IDS precision and the capacity to identify the form of intrusion can be significantly enhanced by the model [36]. Further work requires, on the other hand, to the training times using GPU acceleration to avoid accidents and disappearances and to investigate the utility of LSTM classification, the bidirectional RNN algorithm in the area of intrusion detection.

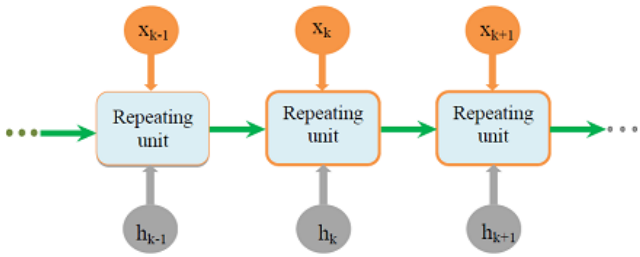

Figure 4. A general structure of RNN

Table 1. Unsupervised (generative) deep learning methods applied to the intrusion detection system

References	Method(s)	Description	Achievement	Dataset
Javaid et al. [18]	Self-Taught Learning (STL), Sparse Auto Encoder (SAE)	Using Self-Taught learning as a classification method and Sparse Autoencoder for Unsupervised Feature Learning	STL achieved a classification accuracy rate of more than 98% for all types of classification.	NSL-KDD
Mirsky et al. [20]	Autoencoder	Use of autoencoders with or without ensembles for online anomaly detection in computer networks.	the algorithm is efficient enough to run on a single core of a Raspberry PI and has even more significant potential on stronger CPUs	Mirai Dataset
Farahnakian et al. [22]	Deep Autoencoder((DAE)	Four auto-encoders in which the output of the autoencoder at the current layer is used as the input of the autoencoder in the next layer.	The proposed approach achieved detection accuracy of 94.71% on the total 10% KDDCUP99 test dataset	KDD CUP'99
Zhang et al. [17]	Deep Autoencoder (DAE)	The IDS mainly consists of a DAE based feature selection engine and an MLP based classifier.	This work achieved a high detection accuracy of 98.80%	UNSW-NB
Shone et al. [14]	Non-symmetric Deep Auto-Encoder (NDAE) and Random Forest (RF)	Combination the power of stacking our proposed Non-symmetric Deep Auto-Encoder (NDAE) (deep- learning) and the accuracy and speed of Random Forest (RF)	5% improvement in accuracy and training time reduction of up to 98.81%	KDD Cup '99 and NSL-KDD
Gao et al. [27]	Multilayer Deep Boltzman Network (DBN)	DBN consists of numerous RBMs. The trained features of RBM are used as input data for learning RBM of the next layer of the DBN stack.	This work showed that DBN could learn a better generative model and perform well on intrusion recognition task.	KDD Cup '99

Althubiti et al. [37] have introduced an intrusion detection model Long Short-Term Memory (LSTM) using the CSIC 2010 HTTP data set. They then assembled the model using Adam optimizer, intending to find the best solution for the problem of binary intrusion classification by using the exact rate as an output predictor.

They used an NN of three levels, input, output, and secret. We also trained the LSTM model, consisting of a nine neuron input layer, which corresponds to the nine properties of a six hidden neuronal layer and a layer output which provided either regular or abnormal neuron output. The number of iterations has been described as 100 epochs, the initialized network weights (0-0.05), and the logarithmic loss function.

We noticed that Adam Optimizer is suitable for the LSTM RNN model for intrusion detection and found that the LSTM RNN model utilizing Adam Optimizer can create an acceptable IDS binary classifier. We suggested that LSTM be extended with more recent intrusion detection data sets and evaluate the efficacy of complex LSTMs with various optimizers in their future work suggestions [37].

Tang et al.'s research [38] proposed a Gated Recurrent Neural Network (GRU-RNN) that allowed SDN IDSs. The system described was evaluated using the KDDCup-99 data set and obtained a precision equivalent to 89 percent, with only six raw characteristics. Our test results have revealed that the GRU-RNN introduced does not reduce the network's efficiency.

Similar to other conventional methods, their solution used the least number of features. And this improves the device performance of the real-time detection process. In comparison, the performance estimation of the network has shown that its approach does not significantly affect the output of the system. This research could be further improved by improving the software and using other tools to improve accuracy. You can also try to implement their system in a centralized way to reduce the burden on the controller [38]. See also these studies [39, 40] for new plays.

3.4 Sum-product networks

The product networks (SPNs) are cyclically directed graphs that comprise variables such as leaves, summations, and goods as weighted edges and internal nodes [32]. The summation nodes have mixing templates, while the multiplication nodes reflect the hierarchy of characteristics [11]. Thus, SPN can be viewed as a mixture of combining models and function regimes, as shown in Figure 5.

Table 1 provides a brief list of all the works that use unattended, deep-learning approaches that have been discussed above, as well as a short description of the techniques and datasets and the outcomes obtained from these works.

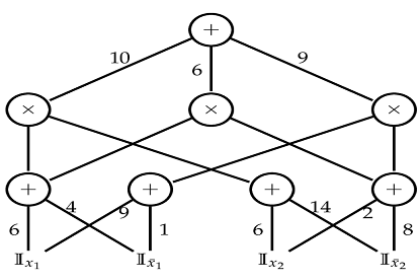

Figure 5. An example of a sum-product network over two Boolean variables X1 and X2

4. DEEP NETWORKS FOR SUPERVISED OR DISCRIMINATIVE LEARNING

We are meant to provide differential power directly for classification of occurrences, typically by characterizing the rear class distributions that are centered on the observable results. The goal label data for this method of controlled learning is usually available directly or indirectly. These are also referred to as the broad networks of discrimination [11]. The Convolutional Neural Network (CNN) is the most famous example of this type of architecture. CNN is used as a unique system that is primarily appropriate for image recognition. The benefit of CNN lies in the fact that the layout of the curriculum requires no time. CNN will train multi-layer networks with gradients to analyze complex, non-linear, high-dimensional, large data sets [41, 42]. CNN employs three key concepts, namely: pooling, central transmission areas, and weights exchanged. One of the comprehensive researches that successfully utilized CNN is Google's AlphaGo [43]. The design of CNN is shown in Figure 6.

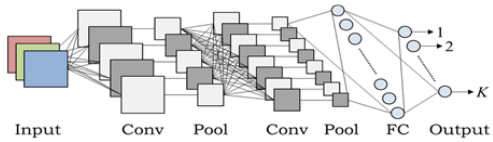

Figure 6. An example of CNN architecture

Yao et al. [44] has proposed a framework for the analysis of anomaly interference focused on Hybrid MLP / CNN (Multilayer Perceptron / Chaotic NN). To improve the detection rate of delayed attacks, the hybrid MLP / CNN NN is developed. The modeling tests were performed using the DARPA 98 data set. The hybrid MLP / CNN model NN results as a chaotic neuron input from the MLP to match the number of chaotic neurons in the MLP output. When an input classification outcome is evaluated by MLP, the CNN attached to the MLP output node can be redirected and maintained. We also identified occurrence occurrences with the use of the MLP system and the CNN memorial features. Because of the hybrid NN's flexible time-delay requirements and capacity, robust intrusion detection and low false alarm levels can be achieved. The system has a strong scalability capacity and the ability to detect new patterns of attack by detecting BSM strings.

Wu et al. [45] proposed a model of NIDS utilizing CNNs. They have used CNN for the automated traffic sorting of raw data and have specified the coefficient of cost function weight of each class to solve the problem of imbalanced data set based on their numbers. The model does not only that the False Alarm Ratio (FAR); it also improves class precision by small amounts. To further minimize computing prices, the original traffic vector format has been transformed into the image format. The initial KDDCup-99 data set has been used for the performance estimation of the new CNN model. The test results revealed that the accurate, FAR, and machine expense of the presented model was better than conventional norm algorithms. Further improvements can be made to the precision of the identification of this work. The CNN concept configuration can be changed to achieve the target. Furthermore, because the detection period is also the key to intrusion detection, it is essential to ensure that the model can satisfy the time requirements of the IDS in improving the detection accuracy [46]. Please refer to the research [47] for further details.

Table 2 gives a brief overview of all research utilizing supervised deep learning approaches described above and offers a short description of the techniques and datasets that are used in these comparisons, as well as the outcomes of such study.

Table 2. Supervised (discriminative) Deep Learning Methods applied to an intrusion detection system

References	Method(s)	Description	Achievement	Dataset
Yao et al. [45]	MLP/CNN	A hybrid MLP/CNN neural network was built to enhance the detection rate of time-delayed attacks.	High intrusion detection rates and low false alarm rates	DARPA 1998
Wu et al. [46]	Convolution Neural Networks (CNNs).	Used CNN to select traffic features from raw datasets automatically, and they set the cost function weight coefficient of each class based on its numbers to solve the imbalanced dataset problem.	Accuracy, FAR and calculation cost of the proposed model performs better than traditional standard algorithms	NSL-KDD

Table 3. Hybrid deep learning methods applied to the intrusion detection system

References	Method(s)	Description	Achievement	Dataset
Kim et al. [35]	Deep Neural Network (DNN)	There are four hidden layers and 100 hidden nodes in the DNN model, and used the ReLU activation function, and used the Adam optimizer for DNN learning.	High accuracy and detection rate averaging 99%. FAR achieved 0.08%.	KDD Cup '99
Tang et al. [48]	Deep Neural Network (DNN)	They have constructed a simple DNN with an input layer, three hidden layers, and an output layer. The input dimension is six, and the output dimension is two. The hidden layers contain twelve, six and three neurons respectively	Performance with an accuracy of 75.75% for just using six basic network features.	NSL-KDD
Potluri et al. [49]	Deep Neural Network (DNN)	Forty-one features are used as input to the DNN. 1st hidden layer is AE is used to select 20 features out of the 41 features. 2nd hidden layer is another AE (with ten neurons) are used to select the ten features out of 20. The (1st and 2nd) hidden layers are fed to the pre-training process of the DNN. 3rd hidden layer (SoftMax) is used to select five features out of 10 and also used as a fine tuner with supervised learning.	The detection accuracies were reliable on the NSL-KDD dataset by generalizing the attack classes to fewer types.	NSL-KDD

5. HYBRID DEEP NETWORKS

Hybrid deep architectures are the synthesis of each generative and discriminative design. This framework attempts to differentiate data from a biased method. On the other side, the effects of generative architectures were hugely beneficial at the early stage [11].

The DNN is an example of deep hybrid networks, is a multi-layer network that has entirely linked hidden layers and is typically used to plan layered RBM. Many other generative structures that are called synthetic or biased as labeling functions are applied to class marks. Figure 7 displays the underlying DNN architecture [48].

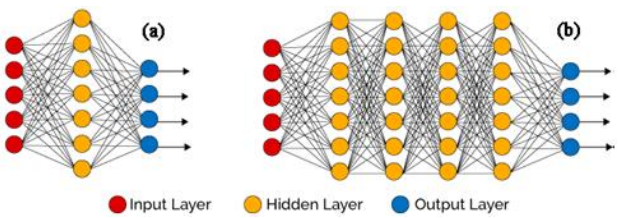

Figure 7. (a) Simple neural network architecture; (b) Simple architecture of deep neural network (DNN)

Kim et al. [35] suggested a smart IDS study using the DNN model to identify attacks effectively. For testing and training, they have used the famous 1999 KDDCup data set for intrusion detection. The test data was generated with the intention of data pre-processing and sample extraction. A DNN model consisting of 4 hidden layers and 100 hidden units was used as the classification algorithm for the proposed IDS and used the ReLU function for the activation function of the hidden layers. Furthermore, the analysis used the adaptive moment (Adam) optimizer, a stochastic solution to DNN learning optimization. The results showed a significantly high precision and identification rate of about 99 percent. In comparison, the FAR has hit about 0.08% [49].

A flow-based anomaly detection system was developed by Tuan A Tang et al., with the use of an IDS DNN model and trained in the use of an NSLKDD data set. In the analysis they suggested, they used only six main features of the NSLKDD data set (which can be easily obtained in an SDN environment). The experimental work has found an optimal DNN hyper-parameter and verified the detection rates and false alarms. The model received an output with a precision of around 75.75%, which is quite fair for six leading network apps. For the future, they suggested the deployment of this approach in a right SDN context with actual network traffic and tested latency and performance effectiveness of the whole network [50].

Potluri et al. [49] have built an optimized DNN model to classify network data abnormalities. NSLKDD data set is used to calculate the training time and to evaluate the detection method's performance. Every 41 attribute inserted into the DNN is the input sheet. The hidden layer 1 is the first autoencoder to pick the 20/40 features of the input data. Therefore, AE has 20 neurons. The hidden layer2 is a new AE of 10 neurons and 10 of 20 features from the hidden layer1. The first two secret layers are included in the DNN pre-training process. The hidden layer 3 is the Soft-Max tier, which reduces the number of functions to 5 and also does fine-tuning for controlled instruction.

The emphasis was on measuring the performance of the DNN training related to different types of processors and the number of cores. Similar to sequential processing, speeding the training process by using the Multi-Core CPU became quicker. The GPU was, however, unable to attain the expected performance because of the type of data used. The analysis can be extended by evaluating the efficacy of the accelerated systems (each Multi-Core CPU and GPU) with very complicated intrusion detection results. Therefore, the collection of different characteristics from all 41 can be regarded to improve the precision of the identification of DNN-based IDS [51].

Table 3 offers a brief overview of all experiments utilizing hybrid deep learning approaches, which are listed above, and of the methods and datasets used in those works, as well as a brief description of the methodology and the findings of those studies.

6. ACCESSIBLE INTRUSION DETECTION DATASETS FOR DEEP LEARNING

Several type of data is now gathered by several research groups, both for their study purposes and to provide data to cooperative databases. Below are the most common data sets used for intrusion detection in DL analysis.

6.1 DARPA, KDD99, and NSL-KDD datasets

DARPA 1998 has collected and managed the first reference data from MIT Lincoln Lab, for IDS assessment, under the patronage of' Defense Advanced Research Projects Agency' (DARPA) and' Air Force Research Lab' (AFRL). Because the DARPA data collection contains raw files, scholars will receive features for using them in ml algorithms from those files [52].

The 1999 KDDCup data set was used in the IDS assessment system of DARPA [53]. The data composed of 4 GB of compressed TCP dump data originating from network traffic over seven weeks. The loading is roughly 5 m. Link codes, each with almost 100 bits. It consists of approximately 4,900,000 single link vectors, each with 41 attributes. These include simple (for example, packet size and protocol type), domain knowledge (for example, amount of failed logins), and synchronized measurement functionality (for example, percentage of connections with SYN errors). Each vector has either a standard mark or an attack (22 specified attack types) [14].

Tavallaee et al. have generated the newly developed NSLKDD data set to address the fundamental issues discussed in the KDD 1999 dataset [52]. It is an improvedKDD-99 dataset version [54]. Three main issues have been discussed in the NSL-KDD dataset. Secondly, overlapping reports of the instruction and evaluation sets have been minimized to remove the most obsolete documents from partly classifying programs. Second, the training and test sets were generated by collecting several various documents from different parts of the standard KDD-99 dataset, to obtain accurate results concurrently with the implementation of classification systems. Eventually, the unbalanced dilemma between the number of tests and instruction to be minimized was resolved. The new version of the data set also suffers from several issues addressed by Mc-Hugh [55] and which may not be an ideal reflection of the actual networks available. The new NIDS study also utilizes this data set, and so scientists are still persuaded that it is a necessary standard that allows them to compare different strategies.

The NSLKDD data set is the configuration of the 1999 KDDCup dataset (i.e., 22 assault or regular traffic trends and 41 patented fields). The general description of the associated data sets (DARPA, KDD-99, and NSLKDD) is given in Figure 8. DARPA is a static software platform [56]. KDD-99 is the DARPA data-set extracted function. NSLKDD decreases scale and removes duplicates of the KDD-99 data set.

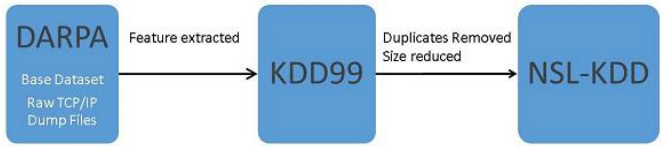

Figure 8. The correlation between the main and the extracted datasets

6.2 ECML-PKDD 2007 dataset

For the European Conference on ML and Information Exploration 2007, the ECML-PKDD 2007 data set was developed in 2007. The ECML / PKDD Exploration Competition was a data mining contest conducted in tandem with the 18th European Machine Learning Conference (ECML). The dataset is presented in the extensible markup (XML) format. The whole sample consists of a single I d, and the three main parts are meaning, class, and question [57, 58].

6.3 ISOT (information security and object technology) dataset

The ISOT dataset consists of an openly available mix of several botnets and standard data sets with a total traffic volume of 1,675,424. For malicious ISOT flow, Storm and Waledac botnets have been taken from the French chapter of the honeynet network. Non-malicious traffic was collected from the Hungarian Traffic Lab Ericson Analysis. This traffic was then merged with another L-generated dataset. National Laboratory of Berkeley (LBNL). In addition to HTTP search, World Warcraft traffic, and Azureus Bit-torent network traffic, this list covers general transportation from several programs. For this cause, this traffic is an extensive data set for the Ericson Laboratory [58].

6.4 HTTP CSIC 2010 dataset

The HTTP CSIC2010 data set includes several thousands of site requests that are generated and created internally at the CSIC Information Security Institute. The dataset can be used to check network attack protection systems. Such statistics comprise 6,000 daily requests, and more than 25,000 irregular requests and HTTP requests are either usual or abnormal [50].

6.5 CTU-13 (Czech technical university) dataset

CTU-13 dataset is a compilation of seizures in a non-fictional network environment with 13 specific malwares. This data set aims to catch real mixed botnets traffic. Botnets traffic generated by compromised hosts and regular traffic produced by confirmed hosts. Eventually, traffic in the past is a legacy of traffic that we don't say for sure [51]. The UNSW-NB15

dataset is now available. This list covers nine specific current assault styles and many other natural operations. This dataset includes 49 class-label attributes, which include the traffic characteristics of the network using a flux between hosts (i.e., server-to-client or client-to-server) and packet headers.

6.6 The ADFA dataset

In 2013, Australian Defense Force Academy Linux Data Set was distributed in New South Wale University by the Defense Force Academy in Australia. ADFA (UNIX dataset) was developed on Ubuntu Linux 11.04 host OS with Apache 2.2.17 running PHP 5.3.5 to test server-based IDS. It began with FTP, SSH, MySQL 14.14, and TikiWiki. Its dataset contains standard and attack device call traces dependent on Linux. The goal of the ADFA dataset is to replace the current test data sets because they have struggled to represent the characteristics of existing computer systems [55].

6.7 ISCX IDS 2012

The data set used to check the classifiers is the Sh data set from the Information Security Center of Excellence (ISCX 2012) [54]. The data set has mainly been developed to design, check, and analyze network intrusion and anomaly detection algorithms. The data collection comprises 17 properties, and the tag meaning shows whether the flow is normal or abnormal. The complete ISCX-labeled data set contains about 1512000 packets with 19 features obtained over a week (regular and attack) network activity. The comparison of datasets is shown in Table 4.

Table 4. Comparison of performance of datasets

Dataset	Network Traffic	Labeled data	IoT traces	Zero-day attacks	Full packet capture	Year
DARPA 98	Yes	Yes	No	No	Yes	1998
KDDCUP 99	Yes	Yes	No	No	Yes	1999
NSL-KDD	Yes	Yes	No	No	Yes	2009
ECML-PKDD	Yes	No	No	No	No	2007
ISCX 2012	Yes	Yes	No	No	Yes	2012
ADFA	Yes	Yes	No	Yes	Yes	2014
ISOT	Yes	Yes	No	Yes	Yes	2014
HTTP CSIC	Yes	Yes	No	Yes	Yes	2010
Bot-IoT	Yes	Yes	Yes	Yes	Yes	2018
CTU-13	Yes	Yes	Yes	No	Yes	2013

7. FRAMEWORKS FOR DEEP LEARNING IMPLEMENTATION

The design of deep learning incorporates the application of modularized deep learning techniques, methods of optimization, distribution methods, and infrastructure support. In this section, the most common frameworks used for depth learning algorithms are briefly presented.

7.1 NVIDIA cuDNN

The NVIDIA CUDA DNN (cuDNN) software is a GPU-accelerated basic version of the DNN class. The cuDNN maintains highly tuned regular procedures such as retroactive and forward convolutions, standardization, pooling, and trigger levels. Several frameworks, such as TensorFlow, Theano, Caffe, and Torch, rely on high-performance GPU acceleration [58]. NVidia-1 is now a driving force in the creation of hardware technologies, such as the Graphical Processing Unit (GPU), and other processors that can speed up and improve the performance of DL approaches [8].

7.2 Tensor-flow

Tensor-Flow (TF), the Dist-Belief successor, is the distributed NN training system used by Google since 2011. TF is an open-source computation library built by Google's brain team [57]. TF was designed to work faster with a Python API using a C / C++ processor. TF has funding for CUDA. Almost any form of networks can be developed using TensorFlow, although it does not enable the hyper-parameter configuration of deep networks. Tensor-Flow also offers a C++ GUI.

Tensor-Flow has released TF-Slim, a set with high standards for the description of complicated Tensor-Flow models. The library TF-Slim offers specific abstracts, which enable users to describe models easily and concisely while retaining the transparency of the model design and keeping its hyper-parameters transparent. TF has as much developer support as possible to incorporate deep learning frameworks. TF is very common in deep learning research as it is scalable for many specific algorithms. It also facilitates low-level, and high-level testing of networks with multiple GPUs is reliable and provides parameter changes consistent [56].

7.3 Theano

Theano was developed by the ML team at the University of Montreal. It is an open-source cross-platform python library. Theano is a Python module used to describe, refine, and test the mathematical expression in a multidimensional sequence. Theano offers high potential for network modeling, dynamic application development, and speed with various GPU supports. Nonetheless, Theano provides low-level APIs and contains several complex compilations that usually take a little time. In the same period, Theano has many different learning opportunities and is still being used by a significant number of researchers and developers [58].

7.4 Keras

To implement deep learning in Theano and TF wrote in Python, Keras was created. This provides the ability to incorporate high-level NN APIs easily for deep learning algorithms. Keras's key point is that it follows Theano and TF and can operate on top of either Theano or TF, a broader and extensible, flexible, and user-friendly application platform that

uses Python. Thanks to the Theano and TF architecture, high-level libraries such as Keras can be written, which can operate on any backend. TF and Theano systems are generally larger than Keras' comparable programs [56]. The Keras model is shown in Figure 9.

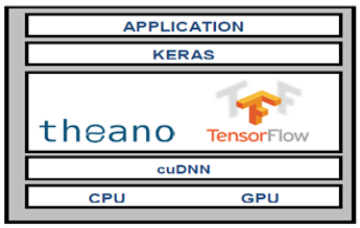

Figure 9. The architecture of Keras

7.5 Caffe

Caffe is an open-source insightful coding toolkit built with group collaborators by Berkeley Center for Vision and Coding. It has a flexible, articulate design and tempo. Caffe is a platform used to describe dynamic algorithms. It promises C++ core language and linking MATLAB and Python support. This toolkit provides a comprehensive framework for teaching, evaluating, and implementing the deep learning platform. NVidiaGPU assists Caffe in speeding deep learning.

7.6 Deeplearning-4j

Deeplearning-4j is a deep learning open-source platform built with Java, CUDA, Python, Scala, and C++. It was published under the 2.0 license for Apache. It was created by an ML team and funded by a start-up company called Skymind. It deals with OSs like UNIX, macOS, Ios, and OS X [50]. Dl4j is an open-source, distributed, and commercial ML toolkit developed by Sky's mind to incorporate deep learning. Spark and Hadoop are combined with the CPU and GPU powered platform for easy and quick prototyping of the DNN application [51].

7.7 Torch/PyTorch

The torch is an open-source deep learning platform based on the simple, fast, and portable scripting language of Lua. Its architecture is a theoretical computation system that follows ML frameworks extensively. Py-Torch has recently experienced a high level of acceptance in the deep learning community and is known as an adversary of Tensor-Flow. In general, Py-Torch is a Torch frame port used for deep NNs development, and the large tensor calculation execution is strong in difficulty.

Py-Torch, recently built on Facebook, is a front-end torch integration with considerable GPU support for adequate performance deep learning growth. It ensures the Python front end that allows complex NN development. On the other side, the toolkit has been launched, and there has been no community support, learning opportunities, and assessment.

7.8 Cognitive network toolkit (CNTK)

It was developed to provide a cohesive platform for common deep learning frameworks by Microsoft Research. This gives multi-GPU comparisons to computing methods and incorporates predictive separation and stochastic gradient descent. This toolkit was released in 2015 and defined as ML's Visual Studio (VS). For those who have used VS to programming, it could be a more straightforward and more fitting way to learn deeply. Generally, performance is quite good. It is a relatively new addition to the toolkits that are offered in the public domain and less use than many others [52].

7.9 MX-net

MX-Net is a deep learning platform built using C++ with many language bindings, which provides distributed computer support that involves multi-GPUs. In comparison to the higher / symbolic level API, it allows access to both lower-level frameworks. Efficiency is taken into account for other efficient systems, namely Tensor-Flow, Caffe, and others.

7.10 DIGITS

NVIDIA has developed DIGITS, and it is a web-based platform for deep networking creation. It is similar in many ways to Caffe, and it uses a text file to define the parameters and the network instead of a programming language. It has a network visualization tool, which allows text errors easier to detect. It also has learning method simulation software and has several GPU supports.

8. CONCLUSION

In this document, we have provided an outline of deep learning and the focus on the most relevant concepts. We studied new articles on fundamental knowledge in the area of intrusion detection. Selected frameworks for intrusion detection are explored, and some commonly adopted architectures for deep learning are illuminated. The approach is discussed in detail in three groups of deep learning structures, namely Generative (unsupervised), Discriminatory (supervised), and Hybrid deep architecture. All three grades offer a great deal of stability and find them valuable and useful in a variety of problems over the decades. The unattended infrastructure, for example, can be split into AE, the Sum-Product Network (SPN), BM, and RNN. We have seen the relevant works and methods implemented in the intrusion detection domain for every class listed above. Instead, we listed the most common data sets for intrusion detection for deep learning and the most successful application frameworks for more in-depth knowledge. To the comparative findings of the related works, the supervised learning algorithms manage labeled data since it is difficult for labeled data to be obtained when dealing with big data, it cannot provide sufficient efficiency in these situations, so the unchecked learning algorithms are used for processing unlabeled data.

Intrusion detection data sets are extremely important for instruction and test systems. Dataset always has several functions that are mostly obsolete or meaningless. Deep learning approaches are typically used to isolate attributes or raising dynamic characteristics. If we have no information about the relationship between raw input data and guided classification performance, we may use deep learning methods. Based on earlier research, AE and RNN are used in the grouping more than CNN, and the efficiency of AE and RNN is higher than CNN, though CNN is quicker than AE and RNN. It should be noted here if researchers need to use the CNN

process, before using this technique, they can first translate the raw data into a picture format. This is because the CNN algorithm is perfect for managing image files; for example, Facebook uses CNN to tag instantly, Amazon to produce product recommendations and Google to scan for user pictures.

In brief, most of the methods mentioned demonstrated a capacity to achieve high levels of accuracy more automatically. AE and the RNN approach for future work may be incorporated into templates for precision enhancements, and the use of object extraction and feature selection as a hybrid approach to maximize intrusion detection performance is suggested.

REFERENCES

[1] Bace, R., Mell, P. (2001). NIST special publication on intrusion detection systems.

[2] Lazarevic, A., Kumar, V., Srivastava, J. (2005). Intrusion detection: A survey. In: Kumar, V., Srivastava, J., Lazarevic, A. (eds) Managing Cyber Threats. Massive Computing, Springer, Boston, MA, pp. 19-78. https://doi.org/10.1007/0-387-24230-9_2

[3] Wagh, S.K., Pachghare, V.K., Kolhe, S.R. (2013). Survey on intrusion detection system using Machine learning techniques. International Journal of Computer Applications, 78(16): 30-37.

[4] Stallings, W. (1998). Cryptography and Network Security: Principles and Practice. Pearson.

[5] Aghdam, M.H., Kabiri, P. (2016). Feature selection for intrusion detection system using ant colony optimization. International Journal of Network Security, 18(3): 420-432.

[6] Tsai, C.F., Hsu, Y.F., Lin, C.Y., Lin, W.Y. (2009). Intrusion detection by machine learning: A review. Expert Systems with Applications, 36(10): 11994-12000. https://doi.org/10.1016/j.eswa.2009.05.029

[7] Durakovic, B. (2017). Design of experiments application, concepts, examples: State of the art. Periodicals of Engineering and Natural Sciences, 5(3): 421-439. https://doi.org/10.21533/pen.v5i3.145

[8] Pouyanfar, S., Sadiq, S., Yan, Y.L., Tian, H.M., Tao, Y.D., Reyes, M.P., Shyu, M.L., Chen, S.C., Iyengar, S.S. (2018). A survey on deep learning: Algorithms, techniques, and applications. ACM Computer Surveys, 51(5): 92. https://doi.org/10.1145/3234150

[9] LeCun, Y., Bengio, Y., Hinton, G. (2015). Deep learning. Nature, 521: 436-444. https://doi.org/10.1038/nature14539

[10] Li, D., Li, X. (2013). Machine learning paradigms for speech recognition: An overview. IEEE Transactions on Audio, Speech, and Language Processing, 21(5): 1060-1089. https://doi.org/10.1109/TASL.2013.2244083

[11] Aminanto, E., Kim, K. (2016). Deep learning in intrusion detection system: An overview. 2016 International Research Conference on Engineering and Technology (IRCET).

[12] Li, D., Yu, D. (2014). Deep learning: Methods and applications. Foundations and Trends® in Signal Processing, 7(3-4): 197-387. http://dx.doi.org/10.1561/2000000039

[13] Bengio, Y., Boulanger-Lewandowski, N., Pascanu, R. (2012). Advances in optimizing recurrent networks. 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouver, BC, pp. 8624-8628. https://doi.org/10.1109/ICASSP.2013.6639349

[14] Shone, N., Ngoc, T.N., Phai, V.D., Shi, Q. (2015). A deep learning approach to network intrusion detection. EEE Transactions on Emerging Topics in Computational Intelligence, 2(1): 41-50. https://doi.org/10.1109/TETCI.2017.2772792

[15] Goodfellow, I., Bengio, Y., Courville, A., Bengio, Y. (2016). Deep Learning. MIT Press Cambridge.

[16] Aminanto, M.E., Kim, K. (2016). Deep learning-based feature selection for intrusion detection system in the transport layer. Computer Science, 26(1): 535-538.

[17] Zhang, H., Wu, C.Q., Gao, S., Wang, Z., Xu, Y., Liu, Y. (2018). An effective deep learning-based scheme for network intrusion detection. 2018 24th International Conference on Pattern Recognition (ICPR), Beijing, pp. 682-687. https://doi.org/10.1109/ICPR.2018.8546162

[18] Javaid, A., Niyaz, Q., Sun, W., Alam, M. (2016). A deep learning approach for network intrusion detection system. Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS), pp. 21-26. https://doi.org/10.4108/eai.3-12-2015.2262516

[19] Raina, R., Battle, A., Lee, H., Packer, B., Ng, A.Y. (2007). Self-taught learning: Transfer learning from unlabeled data. Proceedings of the 24th International Conference on Machine Learning, pp. 759-766. https://doi.org/10.1145/1273496.1273592

[20] Mirsky, Y., Doitshman, T., Elovici, Y., Shabtai, A. (2018). Kitsune: An ensemble of autoencoders for online network intrusion detection. arXiv Prepr. arXiv1802.09089.

[21] Hinton, G.E., Osindero, S., Teh, Y.W. (2006). A fast learning algorithm for deep belief nets. Neural Computation, 18(7): 1527-1554. https://doi.org/10.1162/neco.2006.18.7.1527

[22] Farahnakian, F., Heikkinen, J. (2018). A deep auto-encoder based approach for an intrusion detection system. 2018 20th International Conference on Advanced Communication Technology (ICACT), Chuncheon-si Gangwon-do, Korea (South), pp. 178-183. https://doi.org/10.23919/ICACT.2018.8323688

[23] Deng, L., Hinton, G., Kingsbury, B. (2013). New types of deep neural network learning for speech recognition and related applications: An overview. 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouver, BC, pp. 8599-8603. https://doi.org/10.1109/ICASSP.2013.6639344

[24] Zhang, X., Chen, J. (2017). Deep learning-based intelligent intrusion detection. 2017 IEEE 9th International Conference on Communication Software and Networks (ICCSN), Guangzhou, pp. 1133-1137. https://doi.org/10.1109/ICCSN.2017.8230287

[25] Salakhutdinov, R., Mnih, A., Hinton, G. (2007). Restricted Boltzmann machines for collaborative filtering. Proceedings of the 24th International Conference on Machine Learning, pp. 791-798. https://doi.org/10.1145/1273496.1273596

[26] Bozcan, I., Oymak, Y., Alemdar, I.Z., Kalkan, S. (2018). What is (missing or wrong) in the scene? A hybrid deep Boltzmann machine for contextualized scene modeling.

2018 IEEE International Conference on Robotics and Automation (ICRA), pp. 1-6. arXiv:1710.05664

[27] Gao, N., Gao, L., Gao, Q., Wang, H. (2014). An intrusion detection model based on deep belief networks. 2014 Second International Conference on Advanced Cloud and Big Data, Huangshan, pp. 247-252. https://doi.org/10.1109/CBD.2014.4

[28] Seo, S., Park, S., Kim, J. (2016). Improvement of network intrusion detection accuracy by using restricted Boltzmann machine. Computational Intelligence and Communication Networks (CICN), 2016 8th International Conference on, pp. 413-417.

[29] Alrawashdeh, K., Purdy, C. (2016). Toward an online Anomaly intrusion detection system based on deep learning. 2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA), Anaheim, CA, pp. 195-200. https://doi.org/10.1109/ICMLA.2016.0040

[30] Salama, M.A., Eid, H.F., Ramadan, R.A., Darwish, A., Hassanien, A.E. (2011). Hybrid Intelligent Intrusion Detection Scheme. In: Gaspar-Cunha, A., Takahashi, R., Schaefer, G., Costa, L. (eds) Soft Computing in Industrial Applications. Advances in Intelligent and Soft Computing, Springer, Berlin, Heidelberg, pp. 293-303. https://doi.org/10.1007/978-3-642-20505-7_26

[31] Li, Y., Ma, R., Jiao, R. (2018). A hybrid malicious code detection method based on deep learning. International Journal of Software Engineering and its Applications, 9(5): 205-216. https://doi.org/10.14257/ijseia.2015.9.5.21

[32] Imamverdiyev, Y., Abdullayeva, F. (2018). Deep learning method for denial of service attack detection based on restricted Boltzmann machine. Big Data, 6(2): 159-169. https://doi.org/10.1089/big.2018.0023

[33] Khan, A., Zhang, F. (2017). Using recurrent neural networks (RNNs) as planners for bio-inspirrobotic motion. 2017 IEEE Conference on Control Technology and Applications (CCTA), Mauna Lani, HI, pp. 1025-1030. https://doi.org/10.1109/CCTA.2017.8062594

[34] Kim, J., Kim, H. (2015). Applying recurrent neural network to intrusion detection with hessian free optimization. International Workshop on Information Security Applications, pp. 357-369. https://doi.org/10.1007/978-3-319-31875-2_30

[35] Kim, J., Kim, J., Thu, H.L.T., Kim, H. (2016). Long short term memory recurrent neural network classifier for intrusion detection. 2016 International Conference on Platform Technology and Service (PlatCon), Jeju, pp. 1-5. https://doi.org/10.1109/PlatCon.2016.7456805

[36] Yin, C., Zhu, Y., Fei, J., He, X. (2017). A deep learning approach for intrusion detection using recurrent neural networks. IEEE Access, 5: 21954-21961. https://doi.org/10.1109/ACCESS.2017.2762418

[37] Althubiti, S., Nick, W., Mason, J., Yuan, X., Esterline, A. (2018). Applying long short-term memory recurrent neural network for intrusion detection. SoutheastCon 2018, St. Petersburg, FL, pp. 1-5. https://doi.org/10.1109/SECON.2018.8478898

[38] Tang, T.A., Ali, S., Zaidi, R., Mclernon, D., Mhamdi, L., Ghogho, M. (2018). Deep recurrent neural network for intrusion detection in SDN-based networks. 2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft), pp. 25-29. https://doi.org/10.1109/NETSOFT.2018.8460090

[39] Durakovic, B., Basic, H. (2013). Continuous quality improvement in textile processing by statistical process control tools: A case study of medium-sized company. Periodicals of Engineering and Natural Sciences, 1(1): 39-46. http://dx.doi.org/10.21533/pen.v1i1.15.g157

[40] Poon, H., Domingos, P. (2011). Sum-product networks: Anew deep architecture. 2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops), Barcelona, pp. 689-690. http://dx.doi.org/10.1109/ICCVW.2011.6130310

[41] LeCun, Y., Bottou, L., Bengio, Y., Haffner, P. (1998). Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11): 2278-2324. https://doi.org/10.1109/5.726791

[42] Silver, D., Huang, A., Maddison, C.J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., Hassabis, D. (2016). Mastering the game of Go with deep neural networks and tree search. Nature, 529: 484-489. https://doi.org/10.1038/nature16961

[43] Hidaka, A., Kurita, T. (2017). Consecutive dimensionality reduction by canonical correlation analysis for visualization of convolutional neural networks. Proceedings of the ISCIE International Symposium on Stochastic Systems Theory and its Applications, pp. 160-167. https://doi.org/10.5687/sss.2017.160

[44] Yao, Y., Wei, Y., Gao, F., Yu, G. (2006). Anomaly Intrusion detection approach using hybrid MLP/CNN neural network. Sixth International Conference on Intelligent Systems Design and Applications, Jinan, 2006, pp. 1095-1102. https://doi.org/10.1109/ISDA.2006.253765

[45] Wu, K., Chen, Z., Li, W. (2018). A novel intrusion detection model for a massive network using convolutional neural networks. IEEE Access, 6: 50850-50859. https://doi.org/10.1109/ACCESS.2018.2868993

[46] Thanaki, J. (2017). Python Natural Language Processing. 2017. Packt Publishing Ltd.

[47] Kim, J., Shin, N., Jo, S.Y., Kim, S.H. (2017). Method of intrusion detection using deep neural network. 2017 IEEE International Conference on Big Data and Smart Computing (BigComp), Jeju, pp. 313-316. https://doi.org/10.1109/BIGCOMP.2017.7881684

[48] Tang, T.A., Mhamdi, L., McLernon, D., Zaidi, S.A.R., Ghogho, M. (2016). Deep learning approach for network intrusion detection in software-defined networking. 016 International Conference on Wireless Networks and Mobile Communications (WINCOM), Fez, pp. 258-263. https://doi.org/10.1109/WINCOM.2016.7777224

[49] Potluri, S., Diedrich, C. (2016). Accelerated deep neural networks for enhanced intrusion detection system. 2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA), Berlin, pp. 1-8. https://doi.org/10.1109/ETFA.2016.7733515

[50] Kendall, K.K.R. (1999). A database of computer attacks for the evaluation of intrusion detection systems. Massachusetts Institute of Technology.

[51] Stolfo, S.J., Fan, W., Lee, W., Prodromidis, A., Chan,

P.K. (2000). Cost-based modeling for fraud and intrusion detection: Results from the JAM project. Proceedings DARPA Information Survivability Conference and Exposition. DISCEX'00, Hilton Head, SC, USA, pp. 130-144. https://doi.org/10.1109/DISCEX.2000.821515

[52] Tavallaee, M., Bagheri, E., Lu, W., Ghorbani, A.A. (2009). A detailed analysis of the KDD CUP 99 data set. 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, Ottawa, ON, pp. 1-6. https://doi.org/10.1109/CISDA.2009.5356528.

[53] Dhanabal, L., Shantharajah, S.P. (2015). A study on NSL-KDD dataset for intrusion detection system based on Classification algorithms. Int. J. Adv. Res. Comput. Commun. Eng., 4(6): 446-452.

[54] Özgür, A., Erdem, H. (2016). A review of KDD99 dataset usage in intrusion detection and machine learning between 2010 and 2015. PeerJ Prepr., 4: e1954v1. https://doi.org/10.7287/peerj.preprints.1954v1

[55] McHugh, J. (2000). Testing intrusion detection systems: a critique of the 1998 and 1999 DARPA intrusion detection system evaluations as performed by the Lincoln laboratory. ACM Transactions on Information and System Security, 3(4): 262-294. https://doi.org/10.1145/382912.382923

[56] Srikanth Yadav, M., Kalpana., R. (2019). Data preprocessing for intrusion detection system using encoding and normalization approaches. 2019 11th International Conference on Advanced Computing (ICoAC), Chennai, India, pp. 265-269. https://doi.org/10.1109/ICoAC48765.2019.246851

[57] Patil, A., Yada, S. (2018). Performance analysis of anomaly detection of KDD cup dataset in R environment. International Journal of Applied Engineering Research, 13(6): 4576-4582.

[58] Patil, A., Srikanth Yadav, M. (2018). Performance analysis of misuse attack data using data mining classifiers. International Journal of Engineering & Technology, 7(4): 261-263. https://doi.org/10.14419/ijet.v7i4.36.23782