International Journal of Innovative Research in Science, Engineering and Technology (IJIRSET)

| e-ISSN: 2319-8753, p-ISSN: 2320-6710| www.ijirset.com | Impact Factor: 7.512|

||Volume 9, Issue 7, July 2020||

Intrusion Detection System using Machine Learning Algorithm

Naganath Buvasaheb Khade, S. M. Shinde, Vinayak Macchindra Sale

PG Student, Department of CSE, SVERI's COE Pandharpur, Maharashtra, India

Assistant Professor, Department of CSE, SVERI's COE Pandharpur, Maharashtra, India

Assistant Professor, Department of CSE, SVERI's COE Pandharpur, Maharashtra, India

ABSTRACT: A system Network intrusion discovery framework (NIDS) helps the system admin to identify network security breaks in their own association.Nonetheless, numerous difficulties emerge while building up an intelligent and effective NIDS for unexpected and capricious attacks. In recent years, one of the foremost focuses inside NIDS studies has been the application of machine learning knowledge of techniques. Proposed work present a novel deep learning model to enable NIDS operation within modern networks. The model shows a combination of deep learning, capable of correctly analyzing a wide-range of network traffic. Moreover, additionally proposes novel deep learning classification display built utilizing feature extraction techniques. The performance evaluated network intrusion detection analysis dataset, particularly KDDCUP dataset.

KEYWORDS: Deep and machine learning, intrusion detection, Auto-encoders, KDD, Network security.

I. INTRODUCTION

One of the major challenges in network security is the provision of a robust and effective Network Intrusion Detection System (NIDS). Despite the considerable advances in NIDS system, the majority of solutions still operate using less-successful signature-based techniques, rather than anomaly detection strategies. The current issues are the existing techniques leads to ineffective and inaccurate detection of attacks. There are three main limitations like, volume of network data, in-depth monitoring and granularity required to improve effectiveness and accuracy and finally the number of different protocols and diversity of data traversing. The main focus of NIDS research has been the application of machine learning and shallow learning techniques. The initial deep learning research has demonstrated that its superior layer-wise feature learning can better or at least match the performance of deep learning techniques. It is able to facilitating a deeper evaluation of network data and faster identification of any anomalies. In this paper, proposes a novel deep learning version to enable NIDS operation inside modern networks.

Despite increasing awareness of network security, the existing solutions remain incapable of fully protecting inter- net applications and computer networks opposite the threats from ever-advancing cyber-attack method like as DoS attack and computer malware. Developing effective and adaptive security approaches, therefore, has become more critical than ever before. The traditional security techniques, as the first line of security defense, such as user authentication, firewall and data encryption, are insufficient to fully cover the hole landscape of network security while facing challenge from ever-evolving intrusion skills and method [1]. Hence, other line of security defense is more recommended, like Intrusion Detection System (IDS). Currently, an IDS alongside with anti-virus software has become an important complement to the security infrastructure of most organizations. The combination of these two lines provides a more comprehensive defense against those threats and enhances network security. A significant amount of research has been conducted to develop intelligent intrusion detection techniques, which help achieve better network security. Bagged boosting-based on C5 decision trees [2] and Kernel Miner [3] are two of the earliest attempts to build intrusion detection schemes. Methods proposed in [4] and [5] have successfully applied machine learning techniques to classify network traffic patterns that do not match normal network traffic. Both systems were equipped with five distinct classifiers to detect normal traffic and four different types of attacks (i.e., DoS, probing, U2R and R2L).

However, current network traffic data, which are often huge in size, present a major challenge to IDSs [9]. These big data slow down the entire detection process and may lead to unsatisfactory classification accuracy due to the computational difficulties in handling such data. Classifying a huge amount of data usually causes many mathematical difficulties which then lead to higher computational complexity. As a well-known intrusion calculation dataset, KDD

| e-ISSN: 2319-8753, p-ISSN: 2320-6710| www.ijirset.com | Impact Factor: 7.512|

||Volume 9, Issue 7, July 2020||

Cup 99 dataset is a typical example of more-scale datasets. This dataset contains of more than five million of training samples and two million of testing samples respectively. Such a large scale dataset check the building and testing procedure of a classifier, or form the classifier unable to do due to framework failures caused by low memory. Furthermore, large-scale datasets usually contain noisy, redundant, or uninformative features which present critical challenges to knowledge discovery and information modelling.

II. RELATED WORK

The paper [1] focuses on deep learning methods which are inspired by the structure depth of human brain learn from lower level characteristic to higher levels concept. It is because of abstraction from multiple levels, the Deep Belief Network (DBN) helps to learn functions which are mapping from input to the output. The process of learning does not dependent on human-crafted features. DBN uses an unsupervised learning algorithm, a Restricted Boltzmann Machine (RBM) for each layer. Advantages are: Deep coding is its ability to adapt to changing contexts concerning data that ensures the technique conducts exhaustive data analysis. Detects abnormalities in the system that includes anomaly detection, traffic identification. Disadvantages are: Demand for faster and efficient data assessment.

The main purpose of [2] paper is to review and summarize the work of deep learning on machine health monitoring. The applications of deep learning in machine health monitoring systems are reviewed mainly from the following aspects: Autoencoder (AE) and its variants, Restricted Boltzmann Machines and its variants including Deep Belief Network (DBN) and Deep Boltzmann Machines (DBM), Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN).Advantages are:DL-based MHMS do not require extensive human labor and expert knowledge.The applications of deep learning models are not restricted to specific kinds of machines. Disadvantages are: The performance of DL-based MHMS heavily depends on the scale and quality of datasets.

Proposes the use of a stacked denoisingautoencoder (SdA), which is a deep learning algorithm, to establish an FDC model for simultaneous feature extraction and classification. The SdA model [3] can identify global and invariant features in the sensor signals for fault monitoring and is robust against measurement noise. An SdA is consisting of denoising auto encoders that are stacked layer by layer. This multilayered architecture is capable of learning global features from complex input data, such as multivariate time-series datasets and high-resolution images. Advantagesare:SdA model is useful in real applications. The SdA model proposes effectively learn normal and fault-related features from sensor signals without preprocessing. Disadvantages are: Need to investigate a trained SdA to identify the process parameters that most significantly impact the classification results.

Proposes a novel deep learning-based recurrent neural networks (RNNs)model [4] for automatic security audit of short messages from prisons, which can classify short messages(secure and non-insecure).In this paper, the feature of short messages is extracted by word2vec which captures word order information, and each sentence is mapped to a feature vector. In particular, words with similar meaning are mapped to a similar position in the vector space, and then classified by RNNs.Advantagesare:The RNNs model achieves an average 92.7% accuracy which is higher than SVM.Taking advantage of ensemble frameworks for integrating different feature extraction and classification algorithms to boost the overall performance.Disadvantagesare:It is apply on only short messages not large-scale messages.

Signature-based features technique as a deep convolutional neural network [5] in a cloud platform is proposed for plate localization, character detection and segmentation. Extracting significant features makes the LPRS to adequately recognize the license plate in a challenging situation such as i) congested traffic with multiple plates in the image ii) plate orientation towards brightness, iii) extra information on the plate, iv) distortion due to wear and tear and v) distortion about captured images in bad weather like as hazy images. Advantages are: The superiority of the proposed algorithm in the accuracy of recognizing LP rather than other traditional LPRS. Disadvantages are: There are some unrecognized or miss-detection images.

In [6] paper, a deep learning approach for anomaly detection using a Restricted Boltzmann Machine (RBM) and a deep belief network are implemented. This method uses a one-hidden layer RBM to perform unsupervised feature reduction.The resultant weights from this RBM are passed to another RBM producing a deep belief network. The pre-trained weights are passed into a fine tuning layer consisting of a Logistic Regression (LR) classifier with multi-class soft-max.Advantagesare:Achieves 97.9% accuracy.It produces a low false negative rate of 2.47%.Disadvantages are:Needto improve the method to maximize the feature reduction process in the deep learning network and to improve the dataset.

The paper [7] proposes a deep learning based approach for developing an efficient and flexible NIDS. A sparse autoencoder and soft-max regression based NIDS was implemented. Uses Self-taught Learning (STL), a deep learning based technique, on NSL-KDD - a benchmark dataset for network intrusion. Advantages are: STL achieved a classification accuracy rate more than 98% for all types of classification. Disadvantages are: Need to implement a real-time NIDS for actual networks using deep learning technique.

In [8] paper choose multi-core CPU's as well as GPU's to evaluate the performance of the DNN based IDS to handle huge network data.The parallel computing capabilities of the neural network make the Deep Neural Network (DNN) to effectively look through the network traffic with an accelerated performance.Advantagesare:The DNN based IDS is reliable and efficient in intrusion detection for identifying the specific attack classes with required number of samples for training.The multicore CPU's was faster than the serial training mechanism.Disadvantagesare:Need to improve the detection accuracies of DNN based IDS.

In [9] paper, proposes a mechanism for detecting large scale network-wide attacks using Replicator Neural Networks (RNNs) for creating anomaly detection models.Our approach is unsupervised and requires no labeled data. It also accurately detects network-wide anomalies without presuming that the training data is completely free of attacks.Advantagesare:The proposed methodology is able to successfully discover all prominent DoS attacks and *SYN Port* scans injected. Proposed methodology is resilient against learning in the presence of attacks, something that related work lacks.Disadvantagesare:Need to improve proposed methodology by using stacked autoencoder deep learning techniques.

Based on the flow-based nature of SDN, we propose a flow-based anomaly detection system using deep learning. In [10] paper, apply a deep learning approach for flow-based anomaly detection in an SDN environment.Advantagesare:It finds an optimal hyper-parameter for DNN and confirms the detection rate and false alarm rate.The model gets the performance with accuracy of 75.75% which is quite reasonable from just using six basic network features.Disadvantagesare:It will not work on real SDN environment.

III. SYSTEM OVERVIEW

In this paper, propose a novel deep learning model to enable NIDS operation within modern networks. The model proposes is a combination of deep and shallow learning, capable of correctly analyzing a wide-range of network traffic. More specifically, we combine the power of stacking our proposed Non-symmetric Deep Auto-Encoder (NDAE) (deep learning) and the accuracy and speed of Random Forest (RF) (shallow learning). This paper introduces our NDAE, which is an auto-encoder featuring non-symmetrical multiple hidden layers. NDAE can be used as a hierarchical unsupervised feature extractor that scales well to accommodate high-dimensional inputs. It learns non-trivial features using a similar training strategy to that of a typical auto-encoder. Stacking the NDAEs offers a layer-wise unsupervised representation learning algorithm, which will allow our model to learn the complex relationships between different features. It also has feature extraction capabilities, so it is able to refine the model by prioritizing the most descriptive features.

Advantages are:

1. Due to deep learning technique, it improves accuracy of intrusion detection system.
2. The network or computer is constantly monitored for any invasion or attack.
3. The system can be modified and changed according to needs of specific client and can help outside as well as inner threats to the system and network.
4. It effectively prevents any damage to the network.
5. It provides user friendly interface which allows easy security management systems.
6. Any alterations to files and directories on the system can be easily detected and reported.

B. Mathematical Model

1. Preprocessing:

In this step, training data source (T) is normalized to be equipped for processing by using following steps:

$$T_{norm} = \{\frac{T-\mu_T}{\sigma_T}, \sigma_T \neq 0 \; and \; T - \mu_T, \sigma_T = 0 \qquad (1)$$

Where,

$$T = \{x_{i,j} | i = 1,2, \dots, m \; and \; j = 1,2,3, \dots, n\}$$

| e-ISSN: 2319-8753, p-ISSN: 2320-6710| www.ijirset.com | Impact Factor: 7.512|

||Volume 9, Issue 7, July 2020||

$$\mu_T = \{\mu_j | j = 1,2,3, \dots, n\}$$
$$\sigma_T = \{\sigma_j | j = 1,2,3, \dots, n\}$$

T is m samples with n column attributes; x_{ij} is the jth column attribute in ith sample, μ_T and μ_T are 1*n matrix which are the training data mean and standard deviation respectively for each of the n attributes. Test dataset (TS) which is used to measure detection accuracy is normalized using the same μ_T and μ_T as follows:

$$TS_{norm} = \frac{(TS-\mu_T)}{\sigma_T}, \sigma_T \neq 0 \; and \; TS - \mu_T, \sigma_T = 0 \qquad (2)$$

2. Feature Selection:

NDAE is an auto-encoder featuring *non-symmetrical* multiple hidden layers. The proposed NDAE [1] takes an input vector $x \in R^d$and step-by-step maps it to the latent representations $h_i \in R^d$(here d represents the dimension of the vector) using a deterministic function shown in (3) below:

$$h_i = \sigma(W_i . h_{i-1} + b_i); \; i = \overline{1,n}, \qquad (3)$$

Here, $h_0 = x$, $\sigma$is an activation function (in this work use sigmoid function $\sigma(t) = 1/(1 + e^{-t})$) and n is the number of hidden layers. Unlike a conventional auto-encoder and deep auto-encoder, the proposed NDAE does not contain a decoder and its output vector is calculated by a similar formula to (4) as the latent representation.

$$y = \sigma(W_{n+1} . h_n + b_{n+1}) \qquad (4)$$

The estimator of the model $\theta = (W_i, b_i)$can be obtained by minimizing the square reconstruction error over m training samples $(x^{(i)}, y^{(i)})_{i=1}^m$, as shown in (5).

$$E(\theta) = \sum_{i=1}^m (x^{(i)}, y^{(i)})^2 \qquad (5)$$

C. Algorithms

1. Restricted Boltzamine Machine Algorithm

$x_1$is a sample from the training distribution for the RBM
$\in$ is a learning rate for the stochastic gradient descent in Contrastive Divergence
W is the RBM weight matrix, of dimension (number of hidden units, numbr of inputs)
b is the RBM offset vector for input units
c is the RBM offset vector for hidden units
Notation: $Q(h_{2_i} = 1 | x_2)$is the vector with elements $Q(h_{2_i} = 1 | x_2)$
Step 1: **for** all hidden units i **do**
Step 2: compute $Q(h_{1_i} = 1 | x_1)$(for binomial units, $sigm(c_i + \sum_j W_{ij} x_{1j})$)
Step 3: sample $h_{1_i} \in \{0,1\}$from $Q(h_{1_i} | x_1)$
Step 4: **end for**
Step 5: **for** all visible units j **do**
Step 6: compute $P\left(x_{2_j} = 1 \middle| h_1\right)$(for binomial units, $sigm(b_j + \sum_i W_{ij} h_{1i})$)
Step 7: sample$x_{2_j} \in \{0,1\}$from $P\left(x_{2_j} = 1 \middle| h_1\right)$
Step 8: **end for**
Step 9: **for** all hidden units j **do**
Step 10: compute $Q(h_{2_i} = 1 | x_2)$(for binomial units, $sigm(c_i + \sum_j W_{ij} x_{2j})$)
Step 11: **end for**
Step12: $W \longleftarrow W + \epsilon(h_1 x_1' - Q(h_{2_i} = 1 | x_2) x_2')$
Step 13: $b \longleftarrow b + \epsilon(x_1 - x_2)$
Step 14: $c \longleftarrow c + \epsilon(h_1 - Q(h_{2_i} = 1 | x_2))$

2. Deep Belief Network Algorithm

Train a DBN in a purely unsupervised way, with the greedy layer-wise procedure in which each added layer is trained as an RBM (e.g., by Contrastive Divergence).
$\hat{P}$is the input training distribution for the network
$\in$ is a learning rate for the RBM training
ℓ is the number of layers to train
$W^k$is the weight matrix for level k, for k from 1 to ℓ
$b^k$is the visible units offset vector for RBM at level k, for k from 1 to ℓ

$c^k$is the hidden units offset vector for RBM at level k, for k from 1 to ℓ

Mean_field_computation is a Boolean that is true iff training data at each additional level is obtained by a mean-field approximation instead of stochastic sampling

Step 1: **for** k = 1 to ℓ**do**
Step 2: initialize $W^k = 0, b^k = 0, c^k = 0$
Step 3: **while** not stopping criterion **do**
Step 4: sample $h^0 = x$ from $\hat{P}$
Step 5: **for**$i = 1$ to $k - 1$**do**
Step 6: **if** mean_field_computation**then**
Step 7: assign h_j^i to $Q(h_j^i = 1|h^{i-1}$, for all elements j of h^i
Step 8: **else**
Step 9: assign h_j^i to $Q(h_j^i|h^{i-1}$, for all elements j of h^i
Step 10:**end if**
Step 11:**end for**
Step 12: RBMupdate($h^{k-1}, \in, W^k, b^k, c^k$) {thus providing $Q(h^k|h^{k-1}$)for future use}
Step 13: **end while**
Step 14: **end for**

3. Random Forest Classifier

1. Let the number of training cases be *N*, and the number of variables in the classifier be *M*.
2. The number *m* of input variables to be used to determine the decision at a node of the tree; *m* should be much less than *M*.
3. Choose a training set for this tree by choosing *n* times with replacement from all *N* available training cases (i.e. take a bootstrap sample). Use the rest of the cases to estimate the error of the tree, by predicting their classes.
4. For each node of the tree, randomly choose *m* variables on which to base the decision at that node. Calculate the best split based on these *m* variables in the training set.
5. Each tree is fully grown and not pruned (as may be done in constructing a normal tree classifier).

For prediction a new sample is pushed down the tree. It is assigned the label of the training sample in the terminal node it ends up in. This procedure is iterated over all trees in the ensemble, and the average vote of all trees is reported as random forest prediction.

4. Rule Status Monitoring Algorithm:

Variables: The following variables are used: i, the rule set name index; j, the rule index; k, the rule action index; m, the numberofrulesets; n, the number of rule actions; off_counter, counts of the number of rules that are off; on_counter, counts the numberof rules that are off; ON, array containing the line numbers of rules that are on for a given action and ruleset; OFF, arraycontaining the line numbers of rules that are off for a given action and ruleset.

Algorithm:

Step 1. Obtain and store the name of each rule set into a file named rulesets
Step 2. Determine the total number of rule sets, m, in the rulesets file
Step 3. Assume that each rule is stored on a separate and single line index by j
Step 4. Assume that there are n possible rule actions stored in array action
Step 5. Read the rulesets file
Step 6. For(k=1:n)
Step 7. Select a rule action of interest
Step 8. For(i=1:m)
Step 9. Obtain the name for rule set, S_i, by reading line i of the rulesets file
Step 10. Read the file consisting of the rules for S_i,
Step 11. Initialize arrays ON and OFF to 0; to keep track of rules that are on and off, respectively
Step 12. Let j=1, on_counter = 0, off_counter = 0
Step 13. While (not end of file)
Step 14. Read rule[j]
Step 15. If (rule[j] uses action a[k])
Step 16. If(rule[j] is on)

| e-ISSN: 2319-8753, p-ISSN: 2320-6710| www.ijirset.com | Impact Factor: 7.512|

||Volume 9, Issue 7, July 2020||

Step 17. on_counter = on_counter + 1
Step 18. ON[on_counter] = j
Step 19. Else
Step 20. off_counter = off_counter + 1
Step 21. OFF[off_counter] = j
Step 22. j = j + 1
Step 23. End While
Step 24. If(on_counter != 0)
Step 25. Write ON array to file ['on_rules' + ruleset[i] + date + time]
Step 26. If(off_counter != 0)
Step 27. Write OFF array to file 'off_rules' + ruleset[i] + date + time
Step 28. End For
Step 29. End For

IV. RESULT AND DISCUSSIONS

WSN-Trace is the wireless dataset for researchers. The WSN Trace dataset contains total 19 attributes which are given below:

Table I WSN Trace Dataset Attributes

Total Attributes	
Id	SCH_R
Time	Rank
Is_CH	DATA_S
who CH	DATA_R
Dist_To_CH	Data_Sent_To_BS
ADV_S	dist_CH_To_BS
ADV_R	send_code
JOIN_S	Consumed Energy
JOIN_R	Class
SCH_S	

For experimental set up, use Windows 7 operating system, Intel i5 processor, 4 GB RAM, 200GB Hard disk, Eclipse Luna JDK 8 tool and Tomcat server. To calculate the results, WSN Trace dataset is used. WSN Trace dataset is a real-time wireless dataset which gets information from router which contains node details and packet information. In training and testing dataset, there are 5 types of attack which are subtypes of normal, probing, dos, u2r and r2l attacks.

The performance evaluation of network intrusion detection(NIDPS) is held using different parameters. The Parameters are Detection Accuracy, False Positive Rate, and Detection Time.

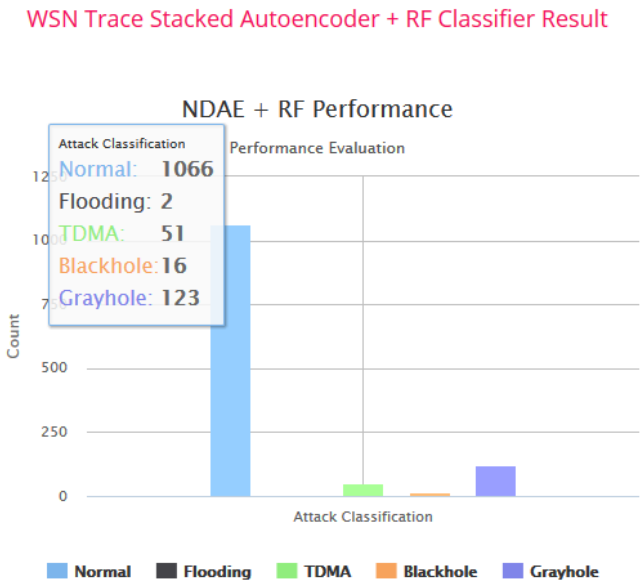

Fig. 2 Performance analysis graph to count the attacks

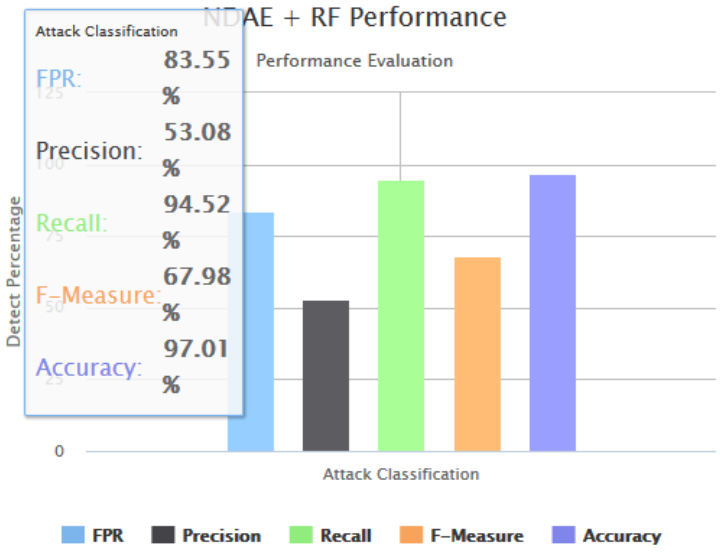

Fig. 3 Performance graph of S-NDAE + RF classifier

V. CONCLUSION

In this paper, we have discussed the problems faced by existing NIDS techniques. In response to this we have proposed our novel NDAE method for unsupervised feature learning. We have then built upon this by proposing a novel classification model constructed from stacked NDAEs and the RF classification algorithm. Also, we implemented the Intrusion Detection system. The result shows that our approach offers high levels of accuracy, precision and recall together with reduced training time. The proposed NIDS system is improved only 5% accuracy. So, there is need to further improvement of accuracy. And also further work on real-time network traffic and to handle zero-day attacks.

| e-ISSN: 2319-8753, p-ISSN: 2320-6710| www.ijirset.com | Impact Factor: 7.512|

||Volume 9, Issue 7, July 2020||

REFERENCES

1. B. Dong and X. Wang, "Comparison deep learning method to traditional methods using for network intrusion detection," in Proc. 8th IEEE Int.Conf. Commun. Softw. Netw, Beijing, China, Jun. 2016, pp. 581–585.
2. R. Zhao, R. Yan, Z. Chen, K. Mao, P. Wang, and R. X. Gao, "Deep learning and its applications to machine health monitoring: A survey," Submitted to IEEE Trans. Neural Netw. Learn. Syst., 2016. [Online]. Available: http://arxiv.org/abs/1612.07640
3. H. Lee, Y. Kim, and C. O. Kim, "A deep learning model for robust wafer fault monitoring with sensor measurement noise," IEEE Trans. Semicond. Manuf., vol. 30, no. 1, pp. 23–31, Feb. 2017.
4. L. You, Y. Li, Y. Wang, J. Zhang, and Y. Yang, "A deep learning based RNNs model for automatic security audit of short messages," in Proc. 16th Int. Symp. Commun. Inf. Technol., Qingdao, China, Sep. 2016, pp. 225–229.
5. R. Polishetty, M. Roopaei, and P. Rad, "A next-generation secure cloud based deep learning license plate recognition for smart cities," in Proc. 15th IEEE Int. Conf.Mach. Learn. Appl., Anaheim, CA, USA, Dec. 2016, pp. 286–293.
6. K. Alrawashdeh and C. Purdy, "Toward an online anomaly intrusion detection system based on deep learning," in Proc. 15th IEEE Int. Conf. Mach. Learn. Appl., Anaheim, CA, USA, Dec. 2016, pp. 195–200.
7. Javaid, Q. Niyaz, W. Sun, and M. Alam, "A deep learning approach for network intrusion detection system," in Proc. 9th EAI Int.Conf. Bio-Inspired Inf. Commun. Technol., 2016, pp. 21–26. [Online]. Available: http://dx.doi.org/10.4108/eai.3-12-2015.2262516
8. S. Potluri and C. Diedrich, "Accelerated deep neural networks for enhanced intrusion detection system," in *Proc. IEEE 21st Int. Conf. Emerg. Technol. Factory Autom.*, Berlin, Germany, Sep. 2016, pp. 1–8.
9. C. Garcia Cordero, S. Hauke, M. Muhlhauser, and M. Fischer, "Analyzing flow-based anomaly intrusion detection using replicator neural networks," in *Proc. 14th Annu. Conf. Privacy, Security. Trust*, Auckland, New Zeland, Dec. 2016, pp. 317–324.
10. T. A. Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi, and M. Ghogho, "Deep learning approach for network intrusion detection in software defined networking," in *Proc. Int. Conf. Wireless Netw. Mobile Commun.*,Oct. 2016, pp. 258–263.