International Journal of Digital Crime and Forensics
Volume 11 • Issue 3 • July-September 2019

A Comparative Analysis of Deep Learning Approaches for Network Intrusion Detection Systems (N-IDSs):
Deep Learning for N-IDSs

Vinayakumar R, Center for Computational Engineering and Networking (CEN), Amrita School of Engineering, Coimbatore, Amrita Vishwa Vidyapeetham, India

Soman KP, Center for Computational Engineering and Networking (CEN), Amrita School of Engineering, Coimbatore, Amrita Vishwa Vidyapeetham, India

Prabaharan Poornachandran, Center for Cyber Security Systems and Networks, Amrita School of Engineering, Amritapuri, Amrita Vishwa Vidyapeetham, India

ABSTRACT

Recently, due to the advance and impressive results of deep learning techniques in the fields of image recognition, natural language processing and speech recognition for various long-standing artificial intelligence (AI) tasks, there has been a great interest in applying towards security tasks too. This article focuses on applying these deep taxonomy techniques to network intrusion detection system (N-IDS) with the aim to enhance the performance in classifying the network connections as either good or bad. To substantiate this to NIDS, this article models network traffic as a time series data, specifically transmission control protocol / internet protocol (TCP/IP) packets in a predefined time-window with a supervised deep learning methods such as recurrent neural network (RNN), identity matrix of initialized values typically termed as identity recurrent neural network (IRNN), long short-term memory (LSTM), clock-work RNN (CWRNN) and gated recurrent unit (GRU), utilizing connection records of KDDCup-99 challenge data set. The main interest is given to evaluate the performance of RNN over newly introduced method such as LSTM and IRNN to alleviate the vanishing and exploding gradient problem in memorizing the long-term dependencies. The efficient network architecture for all deep models is chosen based on comparing the performance of various network topologies and network parameters. The experiments of such chosen efficient configurations of deep models were run up to 1,000 epochs by varying learning-rates between 0.01-05. The observed results of IRNN are relatively close to the performance of LSTM on KDDCup-99 NIDS data set. In addition to KDDCup-99, the effectiveness of deep model architectures are evaluated on refined version of KDDCup-99: NSL-KDD and most recent one, UNSW-NB15 NIDS datasets.

KEYWORDS

Clock-Work Recurrent Neural Network, Deep Learning, Gated Recurrent Unit, Identity-Recurrent Neural Network, KDDCup-99, Long Short-Term Memory, NSL-KDD and UNSW-NB15, Recurrent Neural Network

DOI: 10.4018/IJDCF.2019070104

INTRODUCTION

Information and communication technology (ICT) systems have played a major role in most of the organizations, business, and so on. Human activities highly depend on this system. Alongside, the cyber-crimes to ICT systems are versatile in cyberspace and it has been exists since the birth of the computers. As ICT systems continues to evolve, cyber-crimes change accordingly. The taxonomy of cybercrimes, issues and methods is discussed in detail by (Lallement, 2013). The various attacks and its techniques used for cyber-crime are briefly reported by (Vaidya, 2015). To attack these cyber-crime activities and forensic investigation require a glaring need of comprehensive research study of an appropriate solutions system. One commonly studied critical area by industries and organization for the past several years is intrusion detection (ID). It is an important approach in network security. Many concepts and approaches of machine learning are transferred to ID with the aim to enhance the performance in distinguishing between the abnormal behaviors on the system from the normal network behavior. (Anderson, 1980) is an initial contributor towards the work in ID through a paper "Computer Security threat monitoring and surveillance" published in 1931. Fundamentally, the IDSs are categorized into two types based on the network type and its behaviors such as (1) network basis IDS (N-IDS): depend as far as the data prior to packets in network traffic to identify the malicious activities (2) host basis IDS: rely on the contents as far as the log files such as software logs, system logs, sensors, file systems, disk resources of particular host or a system. An organization uses the intercross as far as network and host-based system to effectively attack the malicious activities in real time environment. This has become an indispensable part of ICT systems and networks. However, the performances of detecting the unforeseen attacks are not acceptable with the existing traditional approaches in N-IDS.

Anomaly detection, state full protocol analysis and misuse detection are main significant methods used for network traffic data classification. Misuse detection is also termed as signature detection that depends on the predefined signatures and filters to efficiently determine the familiar intrusions. For anonymous intrusions, the performance is unacceptable, which may be due to the fact that signature detection relies on human task to constantly update the corpus of signatures with the aim to maintain the signature of new attacks. Anomaly detection aims at detecting the unknown intrusions based on heuristic approaches. Anomaly detection is not a reliable method for unknown intrusions mainly due to results in high false positive rate. Most of the commercial tools that exists in the market have used the hybrids of misuse detections and anomaly detections. A most commonly used powerful approach is state full protocol analysis. State full protocol analysis uses features that proprietarily designed by the software vendor to determine the divergence of specific conventions and applications.

The commercial tools prevailing in market are based on threshold computing approaches or statistical measures that utilize parameters for trafficking such as flow size, inter-arrival time, packet length and so on as features to learn the trafficking patterns for the network in a particular time window. The commercial system may limit the performance in detecting the complex attacks mainly due to the measures computed statistically based on packet header and packet length.

In recent days, to detect and classify the malicious activities from benign characteristics, self-learning systems are used. Self-learning systems are machine learning methods that are either supervised or unsupervised. These methods use a large corpus of malicious and benign connection records to learn the network traffic behaviors with the aim to distinguish between the benign and attack connections. Self-learning system has capability to detect the unknown intrusions that helps in taking the necessary countermeasures against all types of contingencies in a time constrained fashion. Machine learning (ML) methods are current prominent methods used largely for IDS. These ML based solutions to real-time IDS is not an effective approach mainly due to the model's outputs in a high false positive rate and ineffective in identifying the novel intrusions (Lee, Fan, Miller, Stolfo, & Zadok, 2002). The main reason is that the machine learning models learns the attack patterns of simple features of TCP/IP packets locally. However, the recent development of machine learning models resulted in a robust and advanced learning technique, named as 'deep learning'. Deep learning models have achieved significant results in various fields includes natural language processing (NLP), image processing (IP) and speech recognition (SR) (LeCun, Bengio, & Hinton, 2015) comes under the purview of artificial intelligence (AI) tasks. Deep learning approaches have two essential characteristics (1) Ability to learn the complex hierarchical feature representation of TCP/IP packets globally (2) Ability to memorize the past information in large sequences of TCP/IP packets. The performance of deep learning methods is transferred to ID (Staudemeyer, & Omlin, 2014; (Staudemeyer, 2015; Kim, & Kim, 2017). Moreover, recently (Hodo, Bellekens, Hamilton, Tachtatzis, & Atkinson, 2017) outlined the taxonomies and the precursory works of trivial deep learning algorithms to ID. Following, this paper compares the effectiveness of IRNN and other approaches introduced to solving the long-range temporal dependencies for N-IDS. Both LSTM and IRNN network is complex and remained as a black-box. This makes reverse engineering the system with exact same specifications by a malicious adversary quite impossible unless he/she is in possession of the exact same training sample used to build the system.

The rest of the paper contains the literature study of machine learning and neural network-based solutions for IDS, necessary details about what are deep learning algorithms and how they are trained, details of evaluation of experiments of various network structure and parameters of topologies, the detailed evaluation results, future work and conclusion are positioned at last.

DEEP LEARNING

Convolutional neural network (CNN) and recurrent neural network (RNN) are most commonly used deep learning methods. CNN are heavily used in the field of computer vision that extracts the complex features through layer by layer by applying the filters on rectangular area. The complex features represent the hierarchical feature representations in which the features exist in higher level are composed from a set of lower level features. The hierarchical feature representation allows CNN to learn the data in various levels of abstraction. A single or a set of convolution and pooling operations and a non-linear activation functions are primary building blocks of CNN. In recent days, the advantage of using the ReLU as an activation function in deep learning architectures is widely discussed due to ReLU as an activation function is easy to train in comparison to sigmoid or tanh function (Nair, & Hinton, 2010). RNN is mainly used for sequential data modeling in which the hidden sequential relationships in variable length input sequences is learnt by them. RNN mechanism has significantly performed well in the field of NLP and SR (LeCun, Bengio, & Hinton, 2015). In initial time the applicability of Re*LU* activation function in RNN was not successful due to the fact that RNN results in large outputs. As the research evolved, authors showed that RNN outputs vanishing and exploding gradient problem in learning long range temporal dependencies of large scale sequence data modeling. To overcome this issue, research on RNN progressed on the 3 significant directions. One was towards on improving optimization methods in algorithms; Hessian-free optimization methods belong to this category (Martens, 2010). Second one was towards introducing complex components in recurrent hidden layer of network structure; (Hochreiter, & Schmidhuber, 1997) introduced long short-term memory (LSTM), a variant of LSTM network reduced parameters set; gated recurrent unit (GRU) (Cho, Van Merriënboer, Gulcehre, Bahdanau, Bougares, Schwenk, & Bengio, 2014), and clock-work RNN (CWRNN) (Koutnik, Greff, Gomez, & Schmidhuber, 2014). Third one was towards the appropriate weight initializations; recently, (Le, Jaitly, & Hinton, 2015) authors have showed RNN with Re*LU* involving an appropriate initialization of identity matrix to a recurrent weight matrix is able to perform closer in the performance in compared to LSTM. This was substantiated with evaluating the 4 experiments on two toy problems, language modeling and SR. They named the newly formed architecture of RNN as identity-recurrent neural network (IRNN). The basic idea behind IRNN is that, while in the case of deficiency in inputs, the RNN stays in same state indefinitely in which the RNN is composed of Re*LU* and initialized with identity matrix.

RELATED WORK

Researchers have introduced various machine learning based solutions to N-IDS. Applying machine learning for the enhancement of improvement in detecting the malicious activities is largely used in recent days. This section delves into detailed studies of machine learning basis solutions to IDS till date and disclosed with the issues of KDDCup-99 data set.

Most of the research studies considered N-IDS as a classification task that finds a separating plane for the behaviors of normal and attacks in network traffic data. A substantial number of methods introduced following the first paper about N-IDS (Denning, 1987). In that, machine learning and data mining techniques appeared as prominent methods that are largely used towards IDS. However, their performance is less due to N-IDS is an evolving problem with new types of attacks. Due to this, IDS has been under active research for the past 25 years.

The openly available data set for creating an effective machine learning model for N-IDS is very less. A few data sets exist, but each suffers from its own issues. A most commonly occurred issue is that the data set have failed to represent the real-time network traffic characteristics. KDDCup-99 is the most commonly used de facto standard N-IDS data set for benchmarking the performance of machine learning models. KDDCup-99 data set was used as part of N-IDS challenge conducted by third International Knowledge Discovery and Data Mining Tools Competition. The task was to classify the connection records as either benign or attack. Totally, 24 teams were participated and submitted their results. The detailed evaluations of these results were published by (Lippmann, Haines, Fried, Korba, & Das, 2000) and KDDCup-98 evaluation results were published by (Lippmann, Fried, Graf, Haines, Kendall, McClung, ... & Zissman, 2000). The first 3 place occupied teams were used the decision tree and its variants. The 9th place occupied team used 1-nearest neighbor classifier. The first 17 place occupied teams have a very tiny difference in detection rate and a large difference is found between 17th and 18th entries. This infers that the first 17 winning entries methods for N-IDS were robust. After challenge, KDDCup-99 data set is most regularly utilized data set for assessment as far as various machine learning classifiers. Most of the research studies used 10% data set for N-IDS in the case of feature reduction. Other few studies have used their own custom-built data sets (Sung, & Mukkamala, 2003; Kayacik, Zincir-Heywood, & Heywood, 2005). These custom-built in data set are constructed using the random selection of connection records from 10% train and test connection records.

After the KDDCup-99 challenge many research studies have used KDDCup-99 for N-IDS. These studies are not directly comparable to triumphant entries as far as KDDCup-99 trial due to the reason that the research studies have used variations of KDDCup-99 data. With misuse detection context, (Sabhnani, & Serpen, 2003) analyzed the effectiveness of various machine learning algorithms for the same KDDCup-99 trial data set. In addition, the authors proposed the multi-expert classifier and achieved the enhancement in false alarm and detection rate in correlation to the KDDCup-99 triumphant entries. They claimed with various experiments using any machine learning classifiers achieving acceptable detection rate for low frequency attacks ('U2R' nearly 30% and 'R2L' nearly 10%) may not be possible in the context of the misuse detection. The following published results are not directly comparable. (Sinclair, Pierce, & Matzner, 1999) used domain knowledge with machine learning approaches particularly genetic algorithms and decision trees to create the rule for expert systems that classifies the network connections as benign and malicious in expert systems. (Yeung, & Chow, 2002) discussed the non-parametric density estimation approach using the gaussian kernels for parzen-window density estimation. (Mukkamala, Sung, & Abraham, 2004) authors have discussed the effectiveness of linear genetic algorithms over artificial neural networks (ANN) and support vector machines (SVM).

A few research studies have used hybrid classifiers such as neural networks with other classifiers. (1) Neural network with SVM (Mukkamala, Sung, & Abraham, 2003) (2) ANN with a Fuzzy networks (Peddabachigari, Abraham, Grosan, & Thomas, 2007) (3) decision trees with neural networks

(Golovko, Kachurka, & Vaitsekhovich, 2007) (4) neural network with RNNs (Shah, Dave, & Chavon, 2004). To understand the sequential relationships between network events, (Ourston, Matzner, Stump, & Hopkins, B. 2003) authors modeled the temporal patterns of normal and malicious behaviors of network connection records with hidden markov model (HMM). The HMM model is significantly performed well in comparison to MLP. (Debar, & Dorizzi, 1992) observed the application of RNN for IDS. The relative analysis between neural network and RNN is analyzed by (Chowdhury, 2008). The effectiveness of RNN and a variant of RNN, LSTM is comprehensively studied and published (Staudemeyer, & Omlin, 2014; October & Staudemeyer, 2015).

BACKGROUND

The section discusses the concepts of various deep learning algorithms and how they are trained. In addition, gives notion of how deep learning algorithms are adopted towards N-IDS.

Recurrent Neural Network (RNN)

The introduction of long established neural network in 1980's for sequence data modeling (Elman, J. L. 1990) is supplemented by recurrent neural network (RNN). It is similar to feed-forward networks (FFNs) except the past states information influences the current states. The cyclic connections from a unit to it facilitate to store and update the values at a particular time step based on its present and past information. In addition, the traditional FFN adopts different parameter for each input feature, RNN use the same parameters across time steps. The shared parameters in RNN help to make it to generalize the characteristics in sequences such as the positions and its length. The shared parameters generalize the RNN in various positions of data in time series and different time-series lengths. RNN have established as a promising method for modeling sequence data of arbitrary length over traditional FFN. RNN network architecture is a basic and remained as baseline architecture for the newly introduced architectures.

In general, RNN consider $x = (x_1, x_{2,} ...., x_{T-1,} x_T)$ as input and maps them to hidden and output vector sequences as $hl = (hl_1, hl_2, ..., hl_{T-1}, hl_T)$ and $op = (op_1, op_2, ..., op_{T-1}, op_T)$ from $t = 1$ to T through the following equations in the forward direction:

$$HL(x, hl) = A(w_{xhl}x + w_{hlhl}hl + b)$$

$$HL : \mathbb{R}^m \times \mathbb{R}^n \to \mathbb{R}^n \ w_{hx} \in \mathbb{R}^{n \times m} \ w_{hh} \in \mathbb{R}^{n \times n} \ b \in \mathbb{R}^k$$

where A, b, w denotes activation function, bias vector and weight matrices respectively.

To learn the temporal dependencies, RNN feds the initial step $hl0$ layer value to the next time step hidden layer $hl_1 = HL(x_1, hl_0)$ and this is defined recursively as $hl_T = HL(x_T, hl_{T-1})$. Next, we can feed hl_T to stacked recurrent hidden layer or output layer through $soft\max$ or $sigmoid$ as non-linear activation function. At each time-step t, the output node op value is estimated using the hidden node value hl at time-step t as:

$$op_t = sf(w_{ophl}hl_t + b_{op})$$

RNN are converted to FFN using Unfolding or Unrolling. This helps to understand the inherent dynamics of each time-step t. Unfolded RNN contains hl hidden layers for the input sequences of length l.

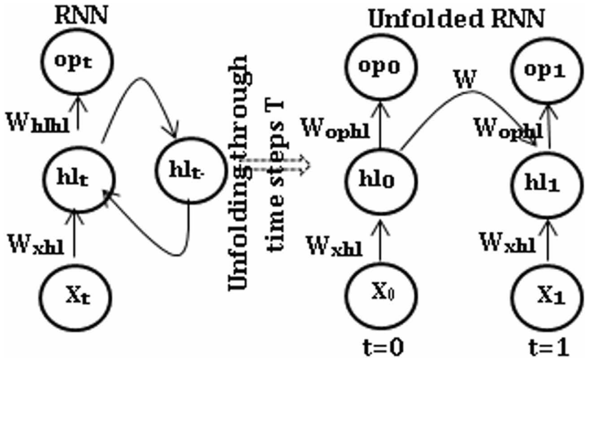
As in Figure 1, the unfolded RNN looks similar to deep neural network except that the weights w_{xh}, w_{hh}, w_{ho} are shared across time steps.

RNN is a parameterized function and to find the right parameters, loss function is used. Loss function gives the units of difference between the predicted and target values. This is defined for subnet in unfolded RNN at each time-step t as:

$$L = d(tv, pv) = \sum_{i=1}^{T} d(tv, pv)$$

The full sequence is considered as one training vector for example $(x_1, \cdots, x_{41}, cl)$. So, total loss is estimated by summing the loss at each time-step. Next, to minimize the loss, gradient of the loss with respect to the weight parameters (w) has to be found and the appropriate parameters for weight parameters is selected through stochastic gradient descent (SGD). Like the previously mentioned process of total loss, one training vector require sum of gradient vector at each time step t. A gradient is calculated using backpropagation with a chain rule to iteratively compute the gradient from the unfolded computational graph. However, unfolded RNNs share its weight parameters across time-steps. So, this process is called backpropagation through time (BPTT) (Sutskever, 2013). While in the process of backpropagating error across many time-steps, the weight matrix has to be multiplied with the gradient signal. This causes the vanishing issue when a gradient becomes too small and exploding gradient issue when a gradient becomes too large (Bengio, Simard, & Frasconi, 1994). As results RNN found to be inefficient in learning the long-range context in sequence data modeling. To overcome from these issues, authors recommended regularization to automatically set the appropriate value at each time step (Pascanu, Mikolov, & Bengio, 2013). They also recommended gradient clipping and a soft constraint for exploding and vanishing issue. Further, Real time recurrent learning (RTRL) approach is introduced (Williams, & Zipser, 1989) that estimates the error derivative at each time step to do a single update in forward direction. However, this RTRL was not familiar due to the reason RTRL produces more computational cost in comparison to BPTT. Truncated back propagation through time (TBPTT) is an amendment of back propagation through time (BPTT) that offers flexibility in removal of exploding gradient issue in continuously running networks (Williams, & Peng, 1990). TBTT required setting the number of time steps in which the error can be propagated. As further the research on RNN in handling vanishing and exploding gradient issue, (Hochreiter, & Schmidhuber, 1997) introduced long short-term memory (LSTM) that followed entirely a new kind of architecture to enhance the storing capacity of values for long time-steps. LSTM contains a memory block and adaptive multiplicative gating units such as input, forget and output gate to control a memory

Figure 1. Unfolded over time steps $t = 0, t = 1$ and RNN



cell (Gers, Schmidhuber, & Cummins, 1999; Gers, Schraudolph, & Schmidhuber, 2002). Generally, the forward pass of LSTM is formulated as follows:

$$x_t, hl_{t-1}, mc_{t-1} \rightarrow hl_t, mc_t$$

$$in\_g_t = \sigma(w_{xin\_g} x_t + w_{hlin\_g} hl_{t-1} + w_{mcin\_g} ml_{t-1} + b_{in\_g})$$

$$fr\_g_t = \sigma(w_{xfr\_g} x_t + w_{hlfr\_g} hl_{t-1} + w_{mcfr\_g} ml_{t-1} + b_{fr\_g})$$

$$mc_t = fr\_g_t \odot mc_{t-1} + in\_g_t \odot \tanh(w_{xmc} x_t + w_{hlmc} hl_{t-1} + b_{mc})$$

$$op_t = \sigma(w_{xop} x_t + w_{hlop} hl_{t-1} + w_{mcop} mc_t + b_{op})$$

$$hl_t = op_t \odot \tanh(mc_t)$$

where in_g, fr_g, op are input, forget and output gating functions respectively, mc denotes memory cell, hl output of hidden layer.

As Figure 2 shows that a memory cell contains complex set of operations with a single memory cell, 3 adaptive multiplicative units and a self-connection with a fixed weight 1.0. To alleviate the number of units, many variants of LSTM is introduced. CWRNN and GRU is most prominent one.

Gated recurrent unit (GRU) is a variant of LSTM network (Cho, Van Merriënboer, Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. 2014). Generally, the forward pass of GRU is formulated as follows:

$$x_t, hl_{t-1} \rightarrow hl_t$$

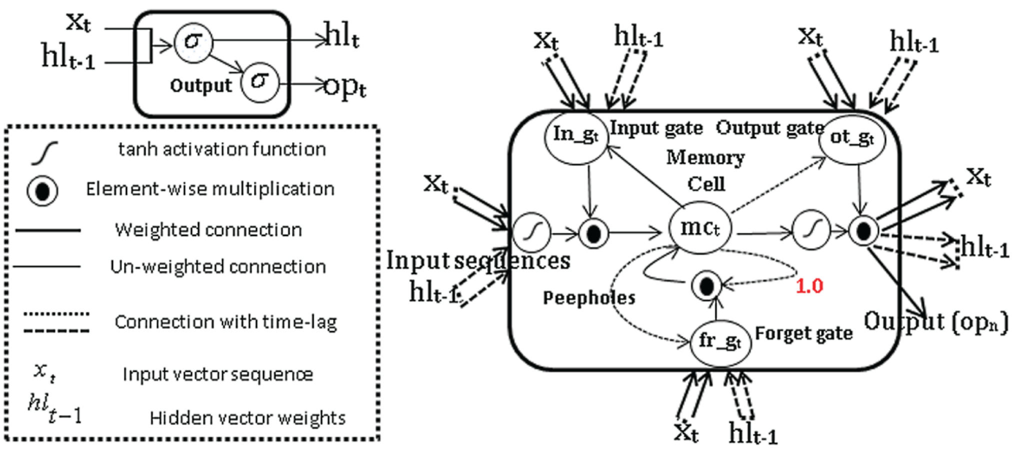
$$i\_f_t = \sigma(w_{xi\_f} x_t + w_{hli\_f} hl_{t-1} + b_{i\_f}) \text{ (Update gate)}$$

$$f_t = \sigma(w_{xf} x_t + w_{hlf} hl_{t-1} + b_f) \text{ (Forget or reset gate)}$$

$$ml_t = \tanh(w_{xml} x_t + w_{hlml}(fr \odot hl_{t-1}) + b_{ml}) \text{ (Current memory)}$$

$$hl_t = f \odot hl_{t-1} + (1 - f) \odot ml \text{ (Updated memory)}$$

Figure 2. Schema of RNN unit (left) and LSTM memory block (right) as adopted from (Elman, 1990; Hochreiter, & Schmidhuber, 1997; Gers, Schmidhuber, & Cummins, 1999; Gers, Schraudolph, & Schmidhuber, 2002)



The GRU comprises of gates (update and forget) which is dissimilar to LSTM memory cell gate list (input, output and forget) that are collaboratively balance the inflow as far as data within the unit. Figure 3 shows the architecture of gated recurrent unit.

Clockwork Recurrent Neural Network (CWRNN)

Clockwork RNN (CWRNN) is a variant to standard RNN architecture (Koutnik, Greff, Gomez, & Schmidhuber, 2014) in which the hidden layer subdivided into parallel M modules. Each such M module runs at various clock rates T_m and weight matrices in modules are get updated based on the condition $t \bmod T_m = 0$ across time steps t otherwise the previous states are retained. In addition, hidden layer with many time steps of CWRNN network facilitate to learn both the short term and long term dependencies of the temporal patterns in sequence data. The modules are connected from hidden to context layer in an increasing ordered fashion ($T_n \geq T_m$) (Figure 4). The formulae and the network representation is given below:

$$hl_t = \sigma(w_{xhl}x_t + w_{hlhl}hl_{t-1} + b_{hl})$$

$$op_t = \sigma(w_{ophl}.hl_{t-1} + b_{op})$$

Figure 3. Architecture of Gated recurrent unit as adopted from Cho, Van Merriënboer, Gulcehre, Bahdanau, Bougares, Schwenk, & Bengio (2014)

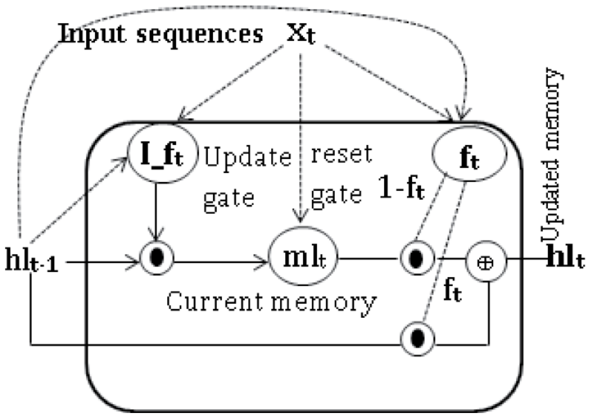
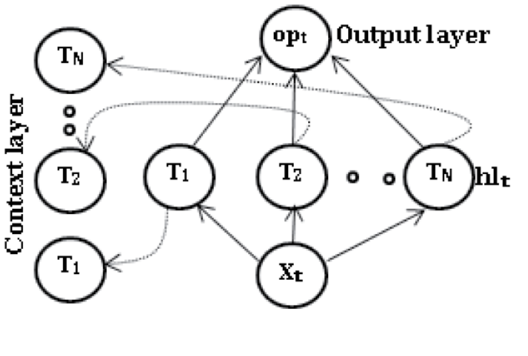


Figure 4. Schematic view of CWRNN, similar to SRNNs with X_t as input layer h_t as hidden layer and o_t as output layer



w_{xhl}, w_{hlhl} are get updated based on the $t \bmod T_m = 0$ at each time step t.

Identity-Recurrent Neural Network (IRNN)

(Le, Jaitly, & Hinton, 2015) proposed a new RNN, named as identity-recurrent neural network (IRNN) with minor changes to RNN that has significantly performed well in capturing long-range temporal dependencies. The minor changes are related to initialization tricks; to initialize the appropriate RNNs weight matrix using an identity matrix or its scaled version and use Re*LU* as non-linear activation function. Moreover, this method performance is closer to LSTM in 4 important tasks; two toy problems, language modeling and speech recognition. In one of the toy problem, IRNN outperformed the LSTM networks.

IRNN use $\mathrm{Re}LU$ activation function and identity matrix is initialized to hidden layer weight matrix w_{hlhl} and the terms of bias are set to zero. By following the described section of RNN, the objective function $op(\theta)$ with a single training set (x, y) is defined as:

$$op = \sum_t S_t(op, y(\theta))$$

where θ denotes the weights and biases parameters in the case of multi-class classification:

$$S_t = -\sum_j op_{tj} \log(y_{tj})$$

According to (Werbos, 1990), backpropogation through time training algorithm (BPTT) follows the gradient computation as given below:

$$\frac{\partial C}{\partial \theta} = \sum_{t \le T} \frac{\partial S_t}{\partial \theta}$$

$$= \sum_{t \le T} \frac{\partial S_t}{\partial hl_T} \frac{\partial hl_T}{\partial hl_t} \frac{\partial^+ hl_t}{\partial \theta}$$

$$= \frac{\partial hl_T}{\partial hl_t} = \frac{\partial hl_T}{\partial hl_{T-1}} \frac{\partial hl_{T-1}}{\partial hl_{T-2}} \cdots \frac{\partial hl_{t+1}}{\partial hl_t}$$

Each Jacobian:

$$\frac{\partial hl_{t+1}}{\partial hl_t}$$

is obtained by multiplying the hidden layer recurrent weight matrix and diagonal matrix with a non-linearity activation function $\mathrm{Re}LU$. When $x_t = 0$ (infers no input), each Jacobian:

$$\frac{\partial hl_{t+1}}{\partial hl_t}$$

produces $L2$ norm as 1. It infers that the gradient of errors exponentially slow at decaying or growing over time-steps.

To understand IRNN network design, let's define the recurrent relation with two hidden units. In IRNN, these units are composed of $\mathrm{Re}LU$, inputs and biases are zero and hidden layer weigh matrix w_{hlhl} is positive definite:

$$\tilde{hl}_t = D\,\tilde{hl}_{t-1}$$

If $\tilde{hl}_{t-1} > 0$

$\tilde{hl}_t = 0$ Otherwise

where $D = Q^{-1}W_{hlhl}Q$ is a diagonal matrix that composed of eigenvectors of W_{hlhl} and $\tilde{hl}_t = D^{-1}hl_t$.

EXPERIMENTS

GPU enabled Tensorflow is utilized to run all experiments (Abadi, Barham, Chen, Chen, Davis, Dean, ... Kudlur, 2016, November). Back propagation through time (BPTT) approach is used to train all the deep learning algorithms.

Description of Data Sets for Network Intrusion Detection System

Due to privacy issues, the openly available data set for N-IDS is very less. Though a few data sets exists; KDDCup-99, modified KDDCup-99 i.e. NSL-KDD and recently introduced data set namely, UNSW-NB15, each data set suffers from a major issue as not real representative of real network traffic. In the following, we outline the details regarding the methods employed to collect TCP/IP packets.

As the first time with the sponsorship from Defense Advanced Research Projects Agency (DARPA ITO) and Air Force Research Laboratory, the DARPA ID Assessment Group (currently the Cyber Systems and Technology Group) has collected network traffic data of 1000's UNIX machines in raw tcpdump format for 9 weeks. The data was distributed in KDDCup-98 challenge. The attacks were ingested to UNIX machines from outside perimeter of its area using SunOS, Windows NT, UNIX, Linux, and Solaris environments. Later, to enhance the detection rate, the same raw tcpdump data were preprocessed and converted to a set of connection records with features and a corresponding class label using the Mining Audit information for automated models for ID (MADMAID) data mining feature construction framework. The detailed evaluation of KDDCup-98 and KDDCup-99 were published in (Lippmann, Haines, Fried, Korba, & Das, 2000; Lippmann, Fried, Graf, Haines, Kendall, McClung, ... Zissman, 2000). The KDDCup-99 data set has made available, its data set in two forms (1) full data set (2) 10% of full data set. The comprehensive description of 10% of KDDCup-99 is displayed in Table 1.

The published results of KDDCup-99 challenge and other published results after the challenge achieved the higher detection rate for the attack categories DoS and Probe and performance of R2L and U2R attacks categories is very less, particularly not acceptable. The background reason for behind this was evaluated by (Al-Subaie, & Zulkernine, 2007) and made a statement that achieving the higher detection rate is not possible. One way to improve may be to add a few more connection records of R2L and U2R to the existing KDDCup-99 data set. Though with a mixed data set, authors didn't show the performance of attack categories of R2L and U2R. Another study (Sabhnani, & Serpen, 2003) reported, the connection records of 'snmpgetattack' were similar in both the Normal and R2L

Table 1. Description of KDDCup-99 and NSL-KDD challenge data set

Attack Category	Data Instances – 10% Data			
	KDDCup-99		NSL-KDD	
	Train	Test	Train	Test
Normal	97,278	60,593	67,343	9,710
DoS	3,91,458	2,29,853	45,927	7,458
Probe	4,107	4,166	11,656	2,422
R2L	1,126	16,189	995	2,887
U2R	52	228	52	67
Total	**4,94,021**	**3,11,029**	**1,25,973**	**22,544**

classes. Subsequently the machine learning classifiers performance is not effective in classifying the connection records of ‘snmpgetattack’ exist in Normal and R2L classes.

In initial time the performance of the traditional IDS was not shown for the KDDCup-99 data set. To overcome, (Bouzida, & Cuppens, 2006) used Snort IDS with the KDDCup-98 traces as input. Snort system performance was good for the connection records belong to low frequency attacks; R2L and U2R in comparison to the high frequency attack; DoS and Probe. This is due to the fact that the Snort used the fixed signatures as an approach. Still, with the harsh criticisms, the KDDCup-99 data set was used in many research studies to evaluate the effectiveness of various machine learning classifiers in the last years (Brugger, & Chow, 2007).

NSL-KDD is a filtered version of KDDCup-99 (Tavallaee, Bagheri, Lu, & Ghorbani, 2009). The applied filters are (1), duplicate connection records were removed and as a result it protects the classifier from being biased towards the frequent connection records. (2) Test Connection records existing in index number 136,489 and 136,497 were removed entirely. (3) The connection records for NSL-KDD were chosen randomly with maintaining the degree of difficulty inversely proportional to KDDCup-99. (4) The existing number of connection vectors in train and test data set is reasonable complement to each class difficulty levels. (5) The records of NSL-KDD are balanced in both the train and test data. NSL-KDD performance is acceptable for misuse or anomaly detection. Even, NSL-KDD lack behind in representing the characteristics of real world network traffic. The other issues are (1) Instead of time to live value as 126 or 253 the NSL-KDD attack packets have 127 or 254 (McHugh, J. 2000). The probability distribution of attack vectors between train and test data are not unique (Mahoney, M. V., & Chan, P. K. 2003, September). As a result, the machine learning classifiers are biased or skewed towards the more frequent connection records. The NSL-KDD is not a complete representative of present-time connection vectors of normal and attacks.

To overcome the reported issues of KDDCup-99 and NSL-KDD, the Australian Centre for Cyber Security group introduced UNSW-NB15 (Moustafa, & Slay, 2016). Mainly the data set includes the recent data patterns of network traffic. The data set consist of Normal and malicious connection features as related to intrusions, applications, protocols, or lower level network entities including the various profiles as e-commerce, military, academia, social media, and banks. The data packets of normal and attack were generated using the IXIA PerfectStorm tool. Common Vulnerabilities and Exposures (CVE) houses the present-day attacks and IXIA tool continuously looks at CVE for knowing the latest malicious activities. The test bed configuration contains 2 servers for normal activities and one server for malicious activities. Using the tcpdump tool the traces were collected nearly around 100GBs on Jan. 22 for the duration of 16 hours and Feb 17, 2015 for the duration of 15 hours. After, the collected Pcaps are transformed to 1,000 MB file by loading in tcpdump tool. Argus and Bro IDS are used to extract features from each 1,000 MB pacap’s file and in addition to extract the higher-level features, the contents of packets are analyzed using the 12 algorithms. These are developed by them using C# programming. The data set is openly available in two forms (1) full data set (2) a small subset of full data set. A small sub set of data set has 175,341 connection records for train and 82,332 connection records for test. The comprehensive description of UNSW-NB15 is displayed in Table 2.

Hyperparameter Selection in IRNN and LSTM Networks

The deep networks such as, RNN, IRNN, CWRNN, LSTM and GRU are parameterized functions, finding the best parameter is an important task. This is primarily because the attack detection rate implicitly depends on the optimal network parameters. At beginning, the experiments are done on basic IRNN network. This composed of 3 layers such as input layer, hidden layer and an output layer. An input layer contains 41 neurons, and hidden layer contains memory blocks in the range [4-64] containing one memory cell each. The connection between the units in input layer and the memory blocks in the hidden layer are fully connected. The output layer contains the five neurons in categorizing the attacks to corresponding categories or two neurons in categorizing the connection record as either normal or an attack.

The models are trained on 10% of KDDCup-99 training data set and the performance of them is evaluated on the 10% corrected test data set. To monitor the train accuracy, 30% of 10% KDDCup-99 training data set is used. The feature values in the connection records are normalized. Various configurations of experiments are done for network parameters such as learning-rate within the limit 0.01-0.5, number of units / memory blocks in the range [4-64] and network structure (hidden recurrent layers in the range [1-4]). The performance of each network is identified by using the

Table 2. Description of partition from UNSW-NB15

Attack Category	Description	Data Instances – A Partition From UNSW-NB15	
		Train	Test
Normal	Connection records without malicious activities	56,000	37,000
Fuzzers	Intruder puts network resources down by feeding data randomly to them continuously	18,184	6,062
Analysis	Port scan, html file penetrations and spam attacks belongs to the analysis attacks category	2,000	677
Backdoors	Intruder gets access to a specific computer by evading the baseline security	1,746	583
DoS	Dos is an attack conducted to put network resources down and as a result even a legal user not able to access network resources	12,264	4,089
Exploits	The security hole of a software is understood by intruder and make an attempt to exploit vulnerability	33,393	11,132
Generic	Generic is an attack on a block cipher	40,000	18,871
Reconnaissance	An attacker work closely to the targeted system to capture the information related to vulnerability	10,491	3,496
Shell code	A small piece of code used in exploitation of software vulnerability	1,133	378
Worms	With replicating themselves, worms are distributed through computer network	130	44
Total		175,341	82,332

confusion matrix. All experiments were run for 300 epochs with batch size 32, ADAM optimizer and categorical cross entropy as loss function. Two challenges of tests are run for the parameters related to the number of units in IRNN with one hidden recurrent layer in the range [4-64]. Experiments with 64 units performed well in comparison to the other units. Moreover, the experiments with less number of units have completely learned the high frequency attacks. But they are completely ineffective in detecting the low frequency attacks. Based on the detection rate of low frequency attacks, 64 units are set for the remaining parts of the experiments. The performance of experiments associated with lower learning-rate is good in identifying the connection record as normal or an attack and categorizing to their corresponding categories. Most importantly, the experiments with learning-rate 0.05 have performed well in comparison to the other learning-rates. Experiments with lower learning-rate i.e. less than 0.05 has required more than 5,000 epochs to attain considerable performance for low frequency attacks. By reviewing the training time and the detection rate, the learning-rate is set to 0.05 for the rest of the test. Moreover, the test with learning-rate 0.05 has required at least 1000 epochs to assimilate the ambush patterns as far as low prevalence attacks. The best performed IRNN network detection rate of each attack is shown in Figure 5.

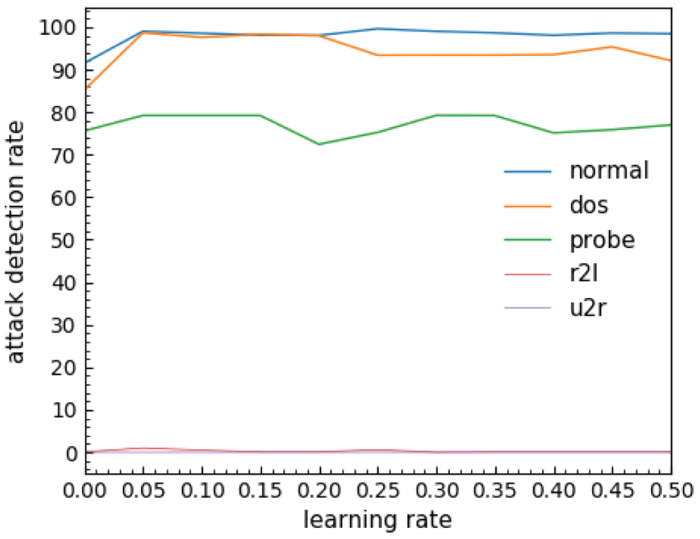
Network Topologies

The following network architectures are used to choose the best network structure for training an IDS model with KDDCup-99:

1. RNN/IRNN/LSTM/CWRNN/GRU 1 layer
2. RNN/IRNN/LSTM/CWRNN/GRU 2 layer
3. RNN/IRNN/LSTM/CWRNN/GRU 3 layer
4. RNN/IRNN/LSTM/CWRNN/GRU 4 layer

Each network configuration is subjected to two trails of experiments, for 400 epochs. Each of the network configurations contains 64 memory blocks. Most of the network topologies gave higher detection rate for the high frequency attacks such as DoS and Probe, but performed underwhelmingly for low frequency attacks, where the detection rates were low. This evidence that the 400 epochs is not adequate to acquire the attack patterns as far as low prevalence attacks. Next, the experiments

Figure 5. Performance as far as IRNN network for learning-rate between 0.01-0.5



associated with each of the network topologies are run till 1,000 epochs. In all configurations of experiments, IRNN and LSTM executed well in relative to the other networks. The best performed IRNN network structure performance of each attacks is shown in Figure 5. The IRNN network has achieved highest detection rate for the normal and DoS for epochs in the range [50-150]. After that, it followed fluctuations in detection rate till 1,000 epochs. This is primarily due to over fitting. It specifically signifies that the network has initiated to keep records or began to remember connection records from the train data, which results in lower generalization performance over any given task. The Probe attack has seen higher detection rate at epoch in range [150-300] and after it seen sudden decrease in detection rate and at last it achieved highest detection rate too. The detection rate of low frequency attack has followed improvement in detection rate for 1,000 epochs. This infers that each attack has required different number of iterations to learn its behaviors.

In next configuration of experiments, three sets of trials are performed for all network architectures to detect a specific interdependence record as normal or attack. IRNN has completely learned the patterns to differentiate between the interdependence records as either normal or attack epochs in range [500-600], see Figure 6. More importantly, complex IRNN network has took numerous number of iterations to attain acceptable detection rate. And finally, they are able to attain highest detection rate.

Experiments With Minimal Feature Sets

Three trials of experiments were performed on 4 layer RNN/IRNN/LSTM/GRU/ network for each of the minimal feature sets that are proposed by (Staudemeyer, & Omlin, 2014). The train data set is randomly divided into two sets such as 70% for training and 30% for validation. The validation accuracy of network topologies of three types as far as minimal attribute sets is shown in Figure 7. The validation accuracy of all network topologies with 11 attribute set is good compared to the other attribute sets. This is primarily due to over fitting in the deep networks. The performance of all network topologies with 11 and eight minimal attribute sets is good in relation to the 4 minimal attribute set. Moreover, the network topology with 11 attribute set is performed well in comparison to the 8 attribute set. The detailed statistics is displayed in Table 3. As Table 3 shows that the IRNN network outperformed LSTM in all three types of minimal attributes sets.

Figure 6. Performance as far as IRNN network with 4 layer over epochs between 0-1000

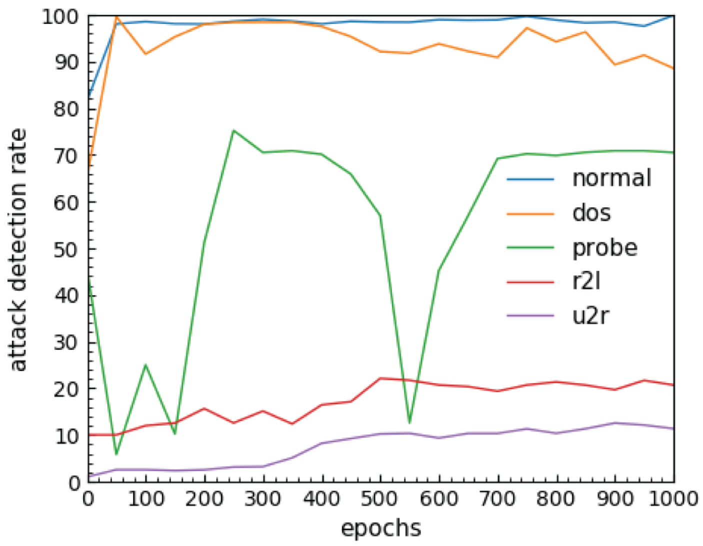


Figure 7. Performance of IRNN networks for minimal feature sets KDDCup-99 and NSL-KDD

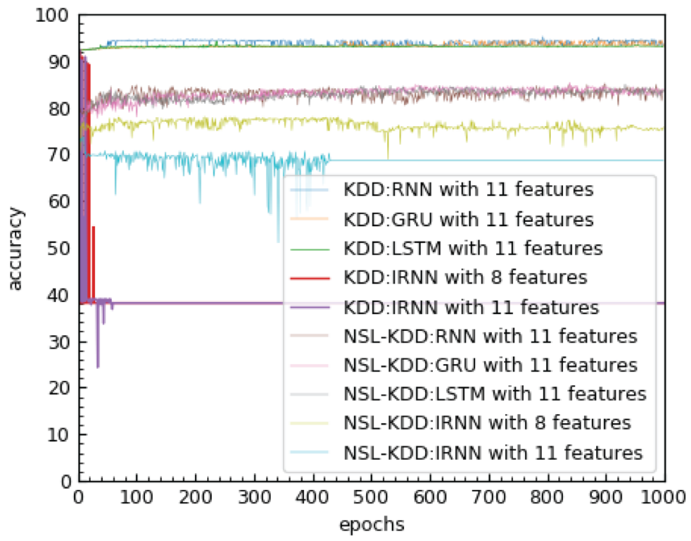


Table 3. Summary of test results for minimal attribute sets of KDDCup-99 and NSL-KDD

KDDCup-99											
Algorithm	NORMAL		DoS		Probe		U2R		R2L		ACC
	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR	
LSTM 11	0.999	0.058	0.941	0.002	0.948	0.019	0.414	0.0	0.258	0.001	0.933
RNN 11	0.999	0.076	0.941	0.003	0.927	0.01	0.129	0.0002	0.107	0.0001	0.928
GRU 11	0.999	0.058	0.941	0.003	0.939	0.02	0.5	0.0	0.215	0.0	0.932
IRNN 11	0.994	0.073	0.959	0.014	0.754	0.002	0.0	0.0	0.0	0.0	0.936
LSTM 8	0.998	0.082	0.939	0.038	0.782	0.002	0.0	0.0	0.014	0.001	0.922
RNN 8	0.994	0.093	0.938	0.015	0.733	0.002	0.0	0.0	0.0	0.0	0.92
GRU 8	0.998	0.092	0.938	0.01	0.762	0.002	0.0	0.0	0.001	0.0	0.921
IRNN 8	1.0	0.081	0.938	0.059	0.642	0.001	0.0	0.0	0.0	0.0	0.924
LSTM 4	0.982	0.093	0.938	0.015	0.769	0.003	0.0	0.0	0.0	0.0	0.918
RNN 4	0.983	0.097	0.934	0.018	0.744	0.002	0.0	0.0	0.041	0.0	0.916
GRU 4	0.984	0.093	0.938	0.015	0.772	0.003	0.0	0.0	0.001	0.0	0.919
IRNN 4	0.996	0.081	0.939	0.051	0.721	0.001	0.0	0.0	0.006	0.0	0.921
NSL-KDD											
LSTM 11	0.997	0.124	0.747	0.018	0.898	0.093	0.45	0.001	0.429	0.002	0.832
RNN 11	0.996	0.111	0.809	0.022	0.879	0.088	0.493	0.001	0.344	0.002	0.840
GRU 11	0.996	0.093	0.781	0.024	0.848	0.076	0.448	0.001	0.53	0.015	0.849
IRNN 11	0.992	0.299	0.738	0.034	0.625	0.058	0.0	0.0	0.047	0.005	0.750
IRNN 8	0.999	0.199	0.805	0.08	0.586	0.063	0.0	0.0	0.089	0.001	0.777
IRNN 4	0.999	0.112	0.757	0.190	0.755	0.047	0.0	0.0	0.001	0.0	0.768

EVALUATION RESULTS

Two experimental scenarios were executed for each network on KDDCup-99, NSL-KDD and UNSW-NB 15. The detailed result of best executed network is described in Table 4. IRNN performed better compared to LSTM and other network topologies. Additionally, IRNN has attained highest AUC of 0.999, as shown in Figure 8. For NSL-KDD, LSTM attained highest AUC of 0.999, as shown in Figure 9. The performance of all network topologies is good for KDDCup-99 and NSL-KDD in comparison to the UNSW-NB 15. This is primarily due to the fact that the hyper parameter selection is not followed for the UNSW-NB 15. Each topology is made to undergo three experimental trials for identifying and categorizing an attack to their corresponding classification. All challenges of tests are run for 500 epochs. The best executed model is evaluated on the test data set of 10% KDDCup-99. The detailed analyses of test results are reported in Table 5. Additionally, the same network topologies are evaluated on the NSL-KDD and UNSW-NB 15 data set; results are shown in Table 6 and Table 7 respectively.

Generally, the input connection records are propagated through more than one hidden recurrent layers in IRNN to capture the time dependencies. The activation score in each layer helps to distinguish an interdependence record as normal or attack, and further classifies the attack to the particular class. The activation values of penultimate layer are passed to t-SNE (Maaten, & Hinton, 2008). This gives us a reduced 2D representation of the original high dimensional feature vector. The two-dimensional vectors are shown in Figure 10. The connection records belong to normal and DoS are well separated. The IRNN network is also able to learn the characteristics of Probe attacks. The attacks related to low frequency attacks such as U2R and R2L have not completely appeared in a separate cluster.

To identify the significant features which contribute towards identifying an attack, the backpropagation methodology is employed (Simonyan, Vedaldi, & Zisserman, 2013). The Taylor expansion is applied on the feature vectors penultimate layer to estimate the first order partial derivative of them. This gives the silent features which are most significant towards identifying an attack. As shown in Figure 11, the connection record is belonging to the Probe. But the features also have similar characteristics of the DoS attack. There are few features of R2L contains same characteristics of features of Probe. The confusion matrix of IRNN and LSTM for KDDCup-99 data set is shown in Figure 12 and Figure 13 respectively. Likewise, confusion matrix of IRNN and LSTM for NSL-KDD is shown in Figure 14 and Figure 15 respectively. The connection records belong to U2R are completely misclassified in both KDDCup-99 and NSL-KDD.

CONCLUSION AND FUTURE WORKS

This paper examines the effectiveness as far as the IRNN and other RNN variants for ID. The detection rates attained for KDDCup-99 intrusion data set from IRNN mechanisms are closely comparable to other RNN variants. The rationale behind the network structure and its parameters are studied in detail with the various experiments of IRNN and RNN variants architectures. Experiments are evaluated with full data set and minimal feature sets to understand the importance of each features. IRNN and RNN variants are shown effective performance for 'DoS' and 'Probe' attacks due to the fact that they formed a unique time series of network events. However, the performance in classifying low frequency attacks is good in relation to the KDDCup-99 challenge triumphant entries. This might be improved by promoting training or stacking a few more layer to the existing architectures or adding new features to the existing data. In most of the cases, the low frequency attack categories produce a single connection record. These low frequency attacks extraction appears to be hard, when information of them concealed in other connection records. Overall the RNN and its variants have enhanced the performance in detection rates in relation to the KDDCup-99 challenge triumphant entries and other previously published results. In addition, the comprehensive performances of IRNN mechanisms are evaluated for NSL-KDD and UNSWNB-15 N-IDS data sets.

Table 4. Detailed results of KDDCup-99, NSL-KDD and UNSW-NB15

Algorithm	Accuracy	Precision	Recall	F-Score
MLP 1	0.924	0.996	0.908	0.950
MLP 2	0.934	0.995	0.922	0.958
MLP 3	0.938	0.988	0.934	0.960
MLP 4	0.939	1.0	0.924	0.960
IRNN l	0.933	1.0	0.917	0.957
IRNN 2	0.949	1.0	0.937	0.968
IRNN 3	0.981	1.0	0.977	0.98
IRNN 4	0.999	1.0	0.999	0.99
LSTM 1	0.924	0.996	0.909	0.950
LSTM 2	0.929	0.999	0.913	0.954
LSTM 3	0.938	0.999	0.923	0.960
LSTM 4	0.983	1.00	0.979	0.989
RNN 4	0.942	1.0	0.928	0.962
GRU 4	0.997	1.0	0.997	0.998
NSL-KDD				
MLP 1	0.799	0.717	0.879	0.790
MLP 2	0.811	0.701	0.879	0.817
MLP 3	0.861	0.766	0.977	0.859
MLP 4	0.861	0.766	0.977	0.859
IRNN l	0.946	0.891	0.995	0.940
IRNN 2	0.951	0.901	0.996	0.946
IRNN 3	0.968	0.931	1.0	0.964
IRNN 4	0.989	0.976	1.0	0.988
LSTM 1	0.926	0.859	0.992	0.921
LSTM 2	0.896	0.814	0.984	0.891
LSTM 3	0.914	0.838	0.992	0.909
LSTM 4	0.973	0.944	0.996	0.969
RNN 4	0.978	0.958	0.992	0.975
GRU 4	0.989	0.974	1.0	0.987
UNSW-NB 15				
MLP 1	0.661	0.619	0.999	0.765
MLP 2	0.687	0.676	0.829	0.745
MLP 3	0.690	0.649	0.951	0.771
MLP 4	0.694	0.647	0.977	0.778
IRNN 1	0.873	0.863	0.966	0.912
IRNN 2	0.894	0.878	0.980	0.926

continued on following page

Table 4. Continued

Algorithm	Accuracy	Precision	Recall	F-Score
IRNN 3	0.895	0.882	0.976	0.926
IRNN 4	0.899	0.889	0.973	0.929
LSTM 1	0.747	0.945	0.667	0.782
LSTM 2	0.753	0.782	0.766	0.774
LSTM 3	0.798	0.927	0.764	0.838
LSTM 4	0.845	0.897	0.872	0.884
RNN 4	0.883	0.876	0.965	0.918
GRU 4	0.897	0.886	0.973	0.928

Figure 8. ROC curve of classical and deep networks for KDDCup-99

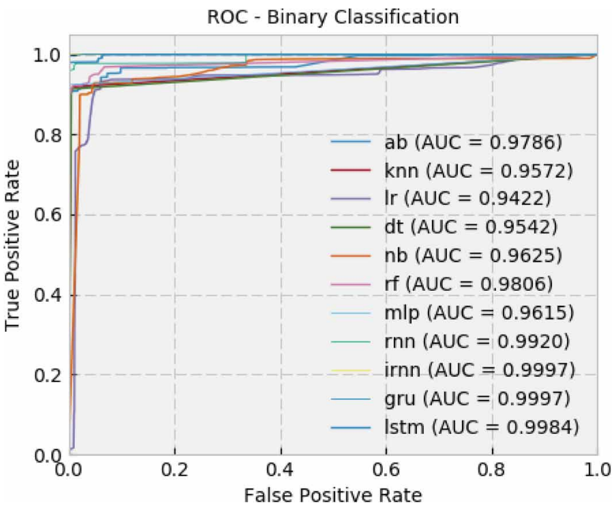


Figure 9. ROC curve of classical and deep networks for NSL-KDD

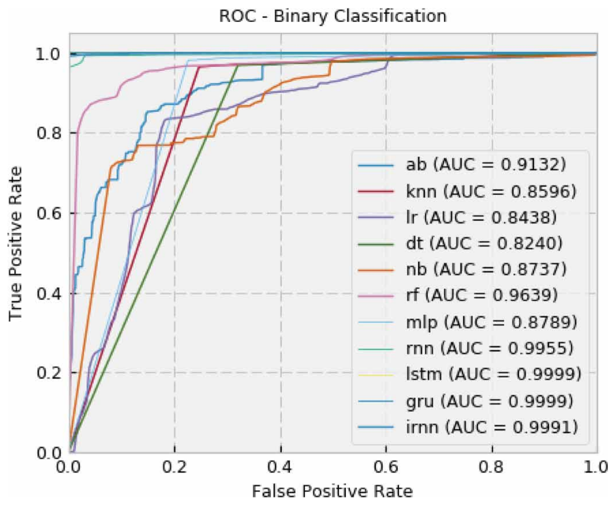


Table 5. Summary as far as test results KDDCup-99 in multi-categorize setting

Algorithm	NORMAL		DoS		Probe		U2R		R2L		ACC
	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR	
LSTM 1	0.996	0.082	0.939	0.005	0.819	0.01	0.0	0.0	0.008	0.0	0.923
LSTM 2	0.997	0.08	0.94	0.005	0.82	0.009	0.0	0.0	0.067	0.0	0.925
LSTM 3	0.999	0.056	0.939	0.003	0.932	0.019	0.326	0.0001	0.33	0.001	0.934
LSTM 4	0.998	0.073	0.941	0.003	0.842	0.001	0.0	0.0	0.466	0.002	0.937
IRNN 1	0.986	0.078	0.940	0.050	0.616	0.001	0.0	0.0	0.132	0.001	0.922
IRNN 2	0.986	0.076	0.939	0.022	0.709	0.002	0.0	0.0	0.332	0.003	0.928
IRNN 3	0.979	0.071	0.959	0.023	0.708	0.003	0.0	0.0	0.6	0.001	0.933
IRNN 4	0.999	0.072	0.94	0.001	0.862	0.001	0.171	0.0	0.499	0.003	0.938
RNN 4	0.998	0.073	0.941	0.005	0.867	0.009	0.029	0.0004	0.186	0.001	0.93
CWRNN 4	0.998	0.077	0.940	0.004	0.831	0.001	0.0	0.0	0.395	0.002	0.935
GRU 4	0.998	0.076	0.94	0.003	0.84	0.001	0.0	0.0	0.401	0.002	0.935
MLP 7	0.997	0.078	0.941	0.005	0.819	0.009	0.0	0.0	0.111	0.0004	0.927

Table 6. Summary as far as test results for NSL-KDD in multi categorize setting

Algorithm	NORMAL		DoS		Probe		U2R		R2L		ACC
	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR	
LSTM 1	0.985	0.124	0.782	0.007	0.968	0.121	0.403	0.0009	0.164	0.002	0.814
LSTM 2	0.986	0.126	0.782	0.007	0.978	0.095	0.493	0.026	0.134	0.001	0.812
LSTM 3	0.991	0.048	0.800	0.016	0.919	0.099	0.388	0.006	0.394	0.025	0.845
LSTM 4	0.994	0.053	0.841	0.012	0.934	0.071	0.299	0.0003	0.680	0.001	0.896
IRNN 1	0.995	0.084	0.828	0.073	0.839	0.113	0.0	0.0	0.002	0.004	0.799
IRNN 2	0.974	0.232	0.774	0.015	0.854	0.091	0.0	0.0	0.023	.001	0.776
IRNN 3	0.976	0.149	0.777	0.059	0.816	0.065	0.0	0.0	0.343	0.007	0.812
IRNN 4	0.976	0.149	0.777	0.059	0.816	0.065	0.0	0.0	0.343	0.007	0.783
RNN 4	0.985	0.113	0.780	0.014	0.939	0.104	0.433	0.001	0.33	0.002	0.83
CWRNN 4	0.980	0.075	0.773	0.009	0.967	0.114	0.448	0.002	0.325	0.022	0.828
GRU 4	0.99	0.101	0.839	0.005	0.988	0.09	0.388	0.002	0.305	0.001	0.855
MLP 4	0.982	0.091	0.777	0.014	0.922	0.111	0.0	0.0	0.261	0.026	0.816

The attacks to computers and its networks are dynamically evolving in contemporary days. DARPA IDS evaluation is remained as a baseline work in most of the IDS for the past several years. However, in recent days this is considered as outdated due to the fact that the attacks are common and inherent issues in connection records. Though NSL-KDD introduced, it appeared as not representative of real network traffic. UNSW-NB15 introduced recently with containing the characteristics of modern day attacks, we shortfall behind in evaluating the detailed examine of them. This will be considered as one of future direction.

Table 7. Summary as far as test results for UNSW-NB15 in multi categorize setting

Algorithm	Normal		Worms		Shell code		Reconnaissance		Generic		Fuzzers		Exploits		DoS		Backdoors		Analysis		Acc
	T	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F	
LSTM 1	0.793	0.238	0.0	0.0	0.0	0.0	0.370	0.008	0.962	0.002	0.116	0.043	0.469	0.177	0.015	0.002	0.0	0.0	0.0	0.0	0.653
LSTM 2	0.538	0.04	0.0	0.0	0.1	0.0	0.379	0.001	0.962	0.001	0.377	0.145	0.82	0.253	0.079	0.001	0.058	0.0	0.025	0.0	0.622
LSTM 3	0.838	0.178	0.0	0.0	0.0	0.0	0.356	0.004	0.978	0.001	0.438	0.077	0.586	0.162	0.05	0.002	0.0	0.0	0.0	0.0	0.673
LSTM 4	0.735	0.025	0.0	0.0	0.0	0.0	0.02	0.001	0.978	0.001	0.759	0.171	0.689	0.187	0.073	0.001	0.039	0.001	0.0	0.0	0.675
IRNN 1	0.952	0.359	0.0	0.0	0.0	0.0	0.367	0.003	0.961	0.005	0.074	0.022	0.241	0.076	0.048	0.006	0.0	0.0	0.0	0.0	0.70
IRNN 2	0.991	0.376	0.0	0.0	0.0	0.0	0.372	0.003	0.962	0.003	0.0007	0.0001	0.24	0.0763	0.036	0.005	0.0	0.0	0.0	0.0	0.717
IRNN 3	0.99	0.38	0.0	0.0	0.0	0.0	0.37	0.004	0.961	0.001	0.007	0.003	0.25	0.078	0.007	0.001	0.0	0.0	0.0	0.0	0.716
IRNN 4	0.99	0.37	0.0	0.0	0.0	0.0	0.363	0.001	0.962	0.006	0.0003	0.0001	0.273	0.080	0.0	0.0	0.0	0.0	0.0	0.0	0.718
RNN 4	0.518	0.07	0.0	0.0	0.0	0.0	0.371	0.002	0.961	0.0005	0.158	0.043	0.77	0.389	0.0	0.0	0.0	0.0	0.0	0.0	0.585
CWRNN 4	0.696	0.042	0.0	0.0	0.0	0.0	0.344	0.001	0.978	0.023	0.376	0.081	0.773	0.280	0.008	0.0003	0.0	0.0	0.0	0.0	0.653
GRU 4	0.739	0.21	0.0	0.0	0.0	0.0	0.375	0.003	0.961	0.0009	0.1089	0.028	0.509	0.236	0.05	0.0008	0.0	0.0	0.0	0.0	0.648
DNN 7	0.873	0.34	0.0	0.0	0.0	0.0	0.0	0.0	0.97	0.240	0.0	0.0	0.036	0.01	0.0	0.0	0.0	0.0	0.0	0.0	0.619

Figure 10. t-SNE visualization of penultimate layer activation values

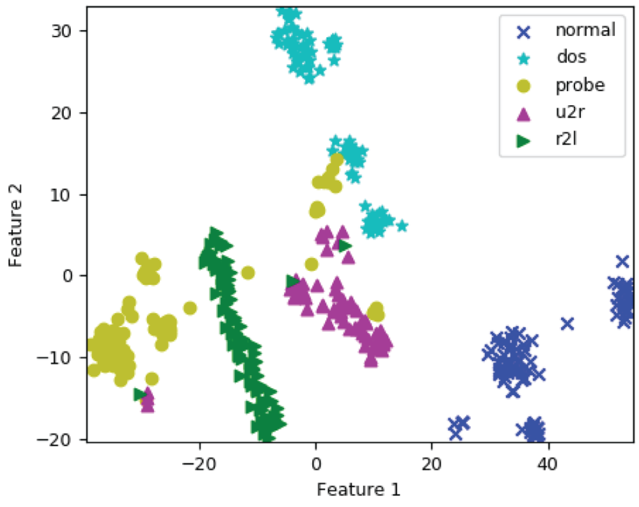


Figure 11. Saliency map for Probe attack

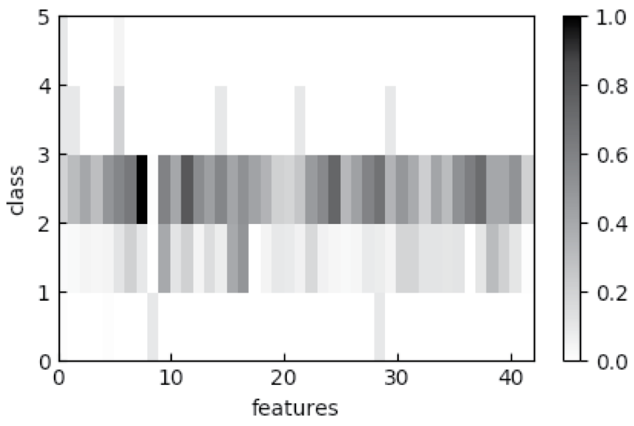


Figure 12. KDDCup '99' confusion matrix for the best performed IRNN network

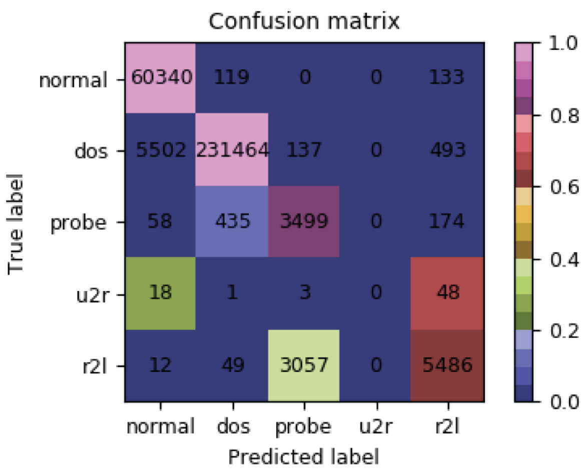


Figure 13. KDDCup '99' confusion matrix for the best performed LSTM network

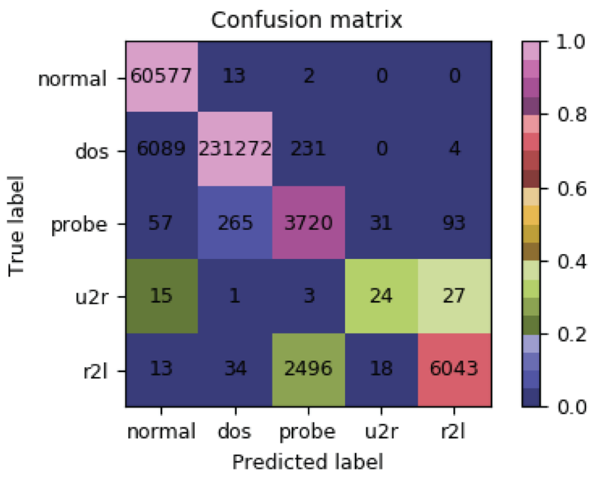


Figure 14. NSL-KDD confusion matrix for the best performed IRNN network

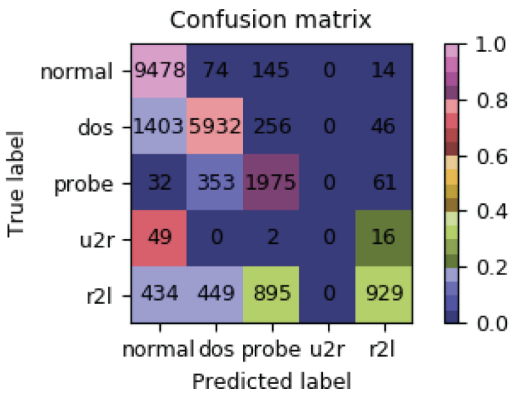
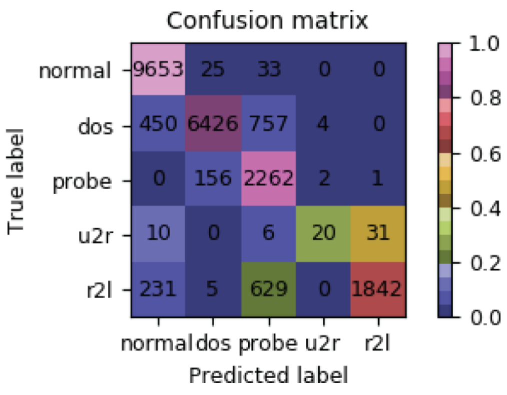


Figure 15. NSL-KDD confusion matrix for the best performed LSTM network



And instead of openly accessible data sets such as KDDCup-99, NSL-KDD and UNSW-NB15, it is better to evaluate the performance on the real time generated intrusion information. While, transforming the real time tcpdump data to connection records, there is a possibility in enriching them by appending the secondary behaviors from various logs, firewalls, alarms of each system, syslog servers, routers, and switches. Each connection records will be manually labeled by domain experts and this process may be extremely time consuming. Even once the labeled data set will be available, the behaviors of traffic outdated and the machine learning model perform in detecting the malicious activities inconsistently. We make concrete statement such that the deep learning will be a promising direction in detecting novel intrusions with the present-day network traffic connection records. To substantiate this, in second direction of our future works we will create such labeled connection records and effectiveness of deep learning family mechanisms will be evaluated on the same connection records.

ACKNOWLEDGMENT

This research was supported in part by Paramount Computer Systems and Lakhshya Cyber Security Labs Pvt Ltd. We are also grateful to NVIDIA India, for the GPU hardware support and to Computational Engineering and Networking (CEN) department for encouraging the research.

REFERENCES

Al-Subaie, M., & Zulkernine, M. (2007, June). The power of temporal pattern processing in anomaly intrusion detection. In *IEEE International Conference on Communications ICC'07* (pp. 1391-1398). IEEE. doi:10.1109/ICC.2007.234

Anderson, J. P. (1980). Computer security threat monitoring and surveillance (Technical report). James P. Anderson Company, Fort Washington, PA.

Bouzida, Y., & Cuppens, F. (2006, September). Neural networks vs. decision trees for intrusion detection. In *IEEE/IST Workshop on Monitoring, Attack Detection and Mitigation (MonAM)* (Vol. 28, p. 29).

Brugger, S. T., & Chow, J. (2007). An assessment of the DARPA IDS Evaluation Dataset using Snort. *UCDAVIS department of Computer Science*, 22.

Chavan, S., Shah, K., Dave, N., Mukherjee, S., Abraham, A., & Sanyal, S. (2004, April). Adaptive neuro-fuzzy intrusion detection systems. In *Proceedings International Conference on Information Technology: Coding and Computing ITCC 2004* (Vol. 1, pp. 70-74). IEEE. doi:10.1109/ITCC.2004.1286428

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv:1406.1078

Chowdhury, N. (2008, December). A comparative analysis of feed-forward neural network & recurrent neural network to detect intrusion. In *International Conference on Electrical and Computer Engineering ICECE 2008* (pp. 488-492). IEEE. doi:10.1109/ICECE.2008.4769258

Debar, H., & Dorizzi, B. (1992, June). An application of a recurrent network to an intrusion detection system. In *International Joint Conference on Neural Networks* (Vol. 2, pp. 478-483). IEEE. doi:10.1109/IJCNN.1992.226942

Denning, D. E. (1987). An intrusion-detection model. *IEEE Transactions on Software Engineering*, *SE-13*(2), 222–232. doi:10.1109/TSE.1987.232894

Elman, J. L. (1990). Finding structure in time. *Cognitive Science*, *14*(2), 179–211. doi:10.1207/s15516709cog1402_1

Gers, F. A., Schraudolph, N. N., & Schmidhuber, J. (2002). Learning precise timing with LSTM recurrent networks. *Journal of Machine Learning Research*, *3*(Aug), 115–143.

Golovko, V., Kachurka, P., & Vaitsekhovich, L. (2007, September). Neural network ensembles for intrusion detection. In *4th IEEE Workshop on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications IDAACS 2007.* (pp. 578-583). IEEE. doi:10.1109/IDAACS.2007.4488487

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780. doi:10.1162/neco.1997.9.8.1735 PMID:9377276

Hodo, E., Bellekens, X., Hamilton, A., Tachtatzis, C., & Atkinson, R. (2017). Shallow and Deep Networks Intrusion Detection System: A Taxonomy and Survey. arXiv:1701.02145

Kayacik, H. G., Zincir-Heywood, A. N., & Heywood, M. I. (2005, October). Selecting features for intrusion detection: A feature relevance analysis on KDD 99 intrusion detection datasets. In *Proceedings of the third annual conference on privacy, security and trust.*

Kim, J., & Kim, H. (2017, February). An Effective Intrusion Detection Classifier Using Long Short-Term Memory with Gradient Descent Optimization. In *2017 International Conference on Platform Technology and Service (PlatCon)* (pp. 1-6). IEEE.

Koutnik, J., Greff, K., Gomez, F., & Schmidhuber, J. (2014, January). A clockwork rnn. In *International Conference on Machine Learning* (pp. 1863-1871).

Lallement, P. (2013). The cybercrime process: an overview of scientific challenges and methods. *Editorial Preface, 4*(12).

Le, Q. V., Jaitly, N., & Hinton, G. E. (2015). A simple way to initialize recurrent networks of rectified linear units. arXiv:1504.00941

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436–444. doi:10.1038/nature14539 PMID:26017442

Lippmann, R., Haines, J., Fried, D., Korba, J., & Das, K. (2000). Analysis and results of the 1999 DARPA off-line intrusion detection evaluation. In Recent Advances in Intrusion Detection (pp. 162-182). Springer Berlin/Heidelberg.

Lippmann, R. P., Fried, D. J., Graf, I., Haines, J. W., Kendall, K. R., McClung, D., & Zissman, M. A. et al. (2000). Evaluating intrusion detection systems: The 1998 DARPA off-line intrusion detection evaluation. In *DARPA Information Survivability Conference and Exposition. DISCEX'00. Proceedings* (Vol. *2*, pp. 12–26).

Maaten, L. V. D., & Hinton, G. (2008). Visualizing data using t-SNE. *Journal of Machine Learning Research*, *9*(Nov), 2579–2605.

Mahoney, M., & Chan, P. (2003). An analysis of the 1999 DARPA/Lincoln Laboratory evaluation data for network anomaly detection. In Recent advances in intrusion detection (pp. 220-237). Springer.

Mahoney, M. V., & Chan, P. K. (2003, September). An analysis of the 1999 DARPA/Lincoln Laboratory evaluation data for network anomaly detection. In *International Workshop on Recent Advances in Intrusion Detection* (pp. 220-237). Springer. doi:10.1007/978-3-540-45248-5_13

Martens, J. (2010, June). Deep learning via Hessian-free optimization. In ICML (Vol. 27, pp. 735-742).

Moustafa, N., & Slay, J. (2015, November). UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In *2015 Military Communications and Information Systems Conference (MilCIS)* (pp. 1-6). IEEE. doi:10.1109/MilCIS.2015.7348942

Moustafa, N., & Slay, J. (2016). The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set. *Information Security Journal: A Global Perspective, 25*(1-3), 18-31.

Mukkamala, S., Sung, A. H., & Abraham, A. (2003). Intrusion detection using ensemble of soft computing paradigms. In *Intelligent systems design and applications* (pp. 239–248). Springer. doi:10.1007/978-3-540-44999-7_23

Mukkamala, S., Sung, A. H., & Abraham, A. (2004, May). Modeling intrusion detection systems using linear genetic programming approach. In *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems* (pp. 633-642). Springer. doi:10.1007/978-3-540-24677-0_65

Nair, V., & Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)* (pp. 807-814).

Ourston, D., Matzner, S., Stump, W., & Hopkins, B. (2003, January). Applications of hidden markov models to detecting multi-stage network attacks. In *Proceedings of the 36th Annual Hawaii International Conference on System Sciences*. IEEE. doi:10.1109/HICSS.2003.1174909

Pascanu, R., Mikolov, T., & Bengio, Y. (2013, February). On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning* (pp. 1310-1318)

Peddabachigari, S., Abraham, A., Grosan, C., & Thomas, J. (2007). Modeling intrusion detection system using hybrid intelligent systems. *Journal of Network and Computer Applications*, *30*(1), 114–132. doi:10.1016/j.jnca.2005.06.003

Sabhnani, M., & Serpen, G. (2003, June). *Application of Machine Learning Algorithms to KDD Intrusion Detection Dataset within Misuse Detection Context. In* MLMTA (pp. 209–215).

Simonyan, K., Vedaldi, A., & Zisserman, A. (2013). Deep inside convolutional networks: Visualising image classification models and saliency maps. arXiv:1312.6034

Sinclair, C., Pierce, L., & Matzner, S. (1999). An application of machine learning to network intrusion detection. In *Proceedings. 15th Annual Computer Security Applications Conference (ACSAC'99)* (pp. 371-377). IEEE. doi:10.1109/CSAC.1999.816048

Staudemeyer, R. C., & Omlin, C. W. (2014). Extracting salient features for network intrusion detection using machine learning methods. *South African computer journal, 52*(1), 82-96.

Sung, A. H., & Mukkamala, S. (2003, January). Identifying important features for intrusion detection using support vector machines and neural networks. In *Proceedings. 2003 Symposium on Applications and the Internet* (pp. 209-216). IEEE. doi:10.1109/SAINT.2003.1183050

Sutskever, I. (2013). *Training recurrent neural networks*. Toronto, Ont., Canada: University of Toronto.

Tavallaee, M., Bagheri, E., Lu, W., & Ghorbani, A. A. (2009, July). A detailed analysis of the KDD CUP 99 data set. In *IEEE Symposium on Computational Intelligence for Security and Defense Applications CISDA 2009* (pp. 1-6). IEEE. doi:10.1109/CISDA.2009.5356528

Vaidya, T. (2015). 2001-2013: Survey and Analysis of Major Cyberattacks. arXiv:1507.06673

Williams, R. J., & Peng, J. (1990). An efficient gradient-based algorithm for on-line training of recurrent network trajectories. *Neural Computation*, *2*(4), 490–501. doi:10.1162/neco.1990.2.4.490

Williams, R. J., & Zipser, D. (1989). A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, *1*(2), 270–280. doi:10.1162/neco.1989.1.2.270

Yeung, D. Y., & Chow, C. (2002). Parzen-window network intrusion detectors. In *Proceedings. 16th International Conference on Pattern Recognition* (Vol. 4, pp. 385-388). IEEE.

Vinayakumar R is a Ph.D. student in the Computational Engineering \& Networking, Amrita School of Engineering, Coimbatore, Amrita Vishwa Vidyapeetham, India since July 2015. He has received BCA from JSS college of Arts, Commerce and Sciences, Ooty road, Mysore in 2011 and MCA from Amrita Vishwa Vidyapeetham, Mysore in 2014. He has several papers in Machine Learning applied to Cyber Security. His Ph.D. work centers on Application of Machine learning (sometimes Deep learning) for Cyber Security and discusses the importance of Natural language processing, Image processing and Big data analytics for Cyber Security. He has participated in several international shared tasks and organized a shared task on detecting malicious domain names (DMD 2018) as part of SSCC'18 and ICACCI'18. More details available at https://vinayakumarr.github.io/.

Soman K.P. has 25 years of research and teaching experience at Amrita School of Engineering, Coimbatore. He has around 150 publications in national and international journals and conference proceedings. He has organized a series of workshops and summer schools in advanced signal processing using wavelets, kernel methods for pattern classification, deep learning, and big-data analytics for industry and academia. He authored books on "Insight into Wavelets", "Insight into Data mining", "Support Vector Machines and Other Kernel Methods" and "Signal and Image processing-the sparse way," published by Prentice Hall, New Delhi, and Elsevier.

Prabaharan P. is professor at Amrita Vishwa Vidyapeetham, Amrita University. He has more than two decades of Experience in Computer Science and Security areas. His areas of interests are malware, critical infrastructure security, complex binary analysis, AI and machine learning.