جامعة نايف العربية للعلوم الأمنية
Naif Arab University for Security Sciences

Naif Arab University for Security Sciences
Journal of Information Security & Cybercrimes Research
مجلة بحوث أمن المعلومات والجرائم السيبرانية
https://journals.nauss.edu.sa/index.php/JISCR

JISCR

Automatic Intrusion Detection System Using Deep Recurrent Neural Network Paradigm

Ahmed Elsherif *
Department of Information Security, College of Computer and Information Security, Naif Arab University for Security Sciences, Riyadh, Saudi Arabia.

Received 07 Feb. 2018; Accepted 17 Apr. 2018; Available Online 30 May 2018.

Abstract

Network security field had gained research community attention in the last decade due to its growing importance. This paper addresses directly one vital problem in that field is "Intrusion Detection System" (IDS). As much as many researchers tackle this problem, many challenges arise while converting this research to reliable automatic system. The biggest challenge is to make the system works with low false alarm with new unseen threats. In this paper, we address this challenge by building a descriptive model using different models of deep Recurrent Neural Network (RNNs). (RNN) models has the ability to generalize the knowledge that can be used to identify seen and unseen threats. This generalization comes from RNN capabilities to define in its terms the normal behavior and the deviation accepted to be normal. Four different models of RNN were tested on a benchmark dataset, NSL-KDD, which is a standard test dataset for network intrusion. The proposed system showed superiority over other previously developed systems according to the standard measurements: accuracy, recall, precision and f-measure.

I. Introduction

Computer and networking technologies have grown in the last decade exponentially. All e-commerce, e-banking, etc… have evolved because they are dependent on online transactions. Also, new concepts and technologies emerged such as the internet of things (IoT) [1], [2], smart homes [3], online educational [4], economics and marketing [5], and other services. Considering this growing number of internet services and heavy traffic of information, internet and information security became a must not optional.

"Intrusion Detection Systems" (IDS) became extremely vital due to the fact that networks might be threatened by both internal and external intruders' attacks [6]. It is defined as a detection system located to observe and monitor computer networks requests [7]. These have been in use since the 1980's [8]. The threats can have harmful damages such as: Denial of service (DoS) which causes prevention of legitimate users from using network resources by streaming irrelevant heavy traffic [9]. Malware also could also cause harm, where attackers use malicious software to foul up systems [10]. IDS is evolving as a response to current and future attacks from internal and external intruders. Most IDS have been developed and implemented to be strict system but they have suffered from the problem of "false positive" or "false

Keywords: *Intrusion Detection System; Machine Learning; Deep Learning, Long Short Term Memory (LSTM); Recurrent Neural Network (RNN), Bi-Directional Recurrent Neural Network (BRNN).*

Production and hosting by NAUSS

* Corresponding Author: Ahmed Elsherif
Email: a.elsherif@nauss.edu.sa
doi: 10.26735/16587790.2018.003

negative" alarms [11]. These high rates of false detection causes IDS to lose credibility in practical large scale systems. Also, another problem that attacks behaviours and methodologies change very rapidly which makes preparing a clear list of attacks behaviour and signature is unrealistic. These issues increases dramatically the difficultly for network administrators to handle intrusion alarms and reports. Scientific researchers in that field aim to develop IDS that accomplishes an accurate threat identification with minimum false alarm rate.

There are two major challenges that appear while developing and implementing effective and dependable IDS for unseen future attacks detection:

- Appropriate extraction and selection of rich discriminative feature set from inside the network traffic raw, unprocessed and noisy dataset for attack detection is very challenging. As mentioned before, attacks behave differently and are continuously evolving and changing over time. The chosen features set selected for identifying and detecting a class of attacks, is highly unlikely that it can identify and detect other classes of attacks.
- The lack for labelled practical traffic dataset from real-world in production networks, this inhibits the ability for developing a generalized and unbiased IDS.

Fully automated IDS that exploit "Machine Learning" (ML) techniques have been developed as an orientation to tackle these issue [12]. ML-based IDS actually learn from normal and abnormal traffics by being trained on a dataset to predict an attack by using classification methods [13].

Many (ML) techniques have been used to implement IDS [14], such as "Artificial Neural Networks" (ANN), "Support Vector Machines" (SVM), "Naive-Bayesian" Classifier (NBC), etc. All these methods follow a hand-crafted feature extraction and selection methods [15, 16]. Recently, "deep learning" solutions have been explored, implemented and gained noticeable results in various (ML) problems such as Big Data, video analytics, Security, image, and speech processing [17]. "Deep Learning" (DL) architectures and learning algorithms tend to identify and capture the features' general representation through a large-scale set of unlabelled noisy data. These features are considered as knowledge and are applied on unseen data in the supervised learning.

The main theory that (DL) is based on that the assumption that data can be formulated mathematically as multi-layered composition of factors or features in a hierarchical composition graph [18]. This seems a simple and helpful assumption that allows exponentially mining for relationships among some of the regions and samples. This is very efficient to some of the tremendously dense-dimensional challenges of many problems with high degree of non-linearity [19].

In this work, we are building IDS system based on Deep Learning architecture: "Recurrent Neural Network" (RNN) with different variations: bi-directional RNN, Long Short Term Memory (LSTM) and bi-directional (LSTM). The proposed solution detects anomaly inside a sequence of user's requests. There are 2 main advantages of the proposed techniques:

- First, they have the ability to define the normal behaviour and so detect any defined or unseen attack signature.
- Second, the bi-directional techniques can neutralize sequence dependencies via considering forward and backward order of request sequences.

Bi-directional (LSTM) has shown superiority over other proposed techniques and other solutions these are mentioned in related work (Section 3). This paper illustrates in section 2 the problem definition, how "IDS" can be formulated in previous studies and what are the controlling parameters these affect (IDS) effectiveness. Section 3 exposes the related work and state-of-art in that field. Section 4 elaborates in the proposed methods, formulation of ML and DL anticipated factors, and benchmark data set that is used to evaluate the proposed system accuracy. Section 5 draws the experimental results and the conclusion section summarizes the work and suggests some improvements.

II. Problem Definition

First of all, IDS differs from Intrusion Provision System (IPS) and they have different problem definition. Both, IDS and IPS do increase the networks security level, perform traffic real-time continues monitoring, inspecting and packets scanning for suspicious data [20]. The key difference between IDS and IPS is the followed

procedure they follow when a suspicious activity is identified in the start stage.

- (IDS) supports and provides the network administrator with a level of security and prevention against any abnormal attack or observed anomalous sequence of actions. "IDS" accomplishes its main target via providing early messages of warnings and alarms addressed to systems administrators [21].
- (IPS) can be viewed as a component that manages IT networks access, this component provides the required protection for the network systems from attacks. It is dedicated to perform deep analysis of attack data and proceed in the action. Furthermore, "IPS" blocks it as it is growing and generates a set of rules in the enterprise other security components such as firewall [22].

In this work, the focus is being on IDS, Heady et al. suggested a definition of an intrusion as: "a set of activities which performs hits to challenge the resource integrity or appropriate authorized usage[23]. Overall, the practice of "IDS" contains the monitoring and logging of relevant actions those are done in a network systems and deep analyzing them for the purpose of detecting the potential existence of intrusions [24]. IDS can be described in a more comprehensive definition as: a set of practices those are exploited to identify and track any abnormal actions these may result a failure in security policy, this is done by usage of anomaly and misuse tracking and by diagnosing intrusions and attacks [23]. This definition embraces both (software and/or hardware) solutions these run on a machine for the purpose of monitoring, identifying and tracking users' behaviors. Also, they are performing checks and inspections tasks for network traffic [24]. The practices followed in "IDS" differs from other network security blocks such as access control, firewalls or encryption. Different security components are integrated to protect a single network and computer system, Fig. 1 illustrates different security components and IDS modules.

From origin, the system admin performs the task of "Intrusion Detection" (ID) by his own experience and manual tools. They were required to monitor every single action on a console for the purpose of identifying and detecting any abnormal behaviors [25]. This primary version of "ID" has shown ineffectiveness because of the

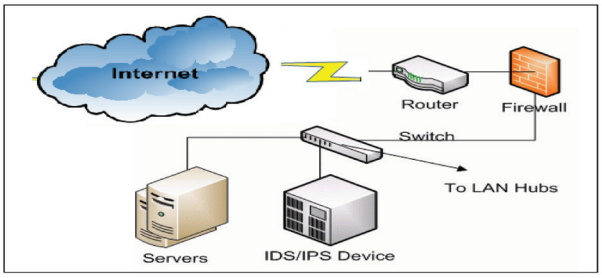

Fig. 1. Network Security Modules and IDS.

errors and false alarms it generates. A step toward automation by employing "the logging file" readers, when they were developed permitting quick searching for any irregularities or un-authorized access [26]. During 80's (ID) was a kind of way for further processing (post) and analysis of suspicious activities, any alternation in the existing network structure accessing rules could be allocated and identified only after the time of the event has passed. This offline identification by logging analysis, was a step although is inefficient and biased [9]. Moving forward to 90's, research in that field caused the appearance of the 1st version of IDS that operates in practical environment and responds immediately making the option of "attack-preemption" jumps to life [27]. In 2000's, IT security enforced the market to be a "marketing trend", advanced features to IDS were added and customized versions have been built to meet large scale organization's needs.

III. State of the Art

This section illustrates the research work in that point and different techniques to solve it. First, for achieve a level of comparison credibility, only the research and results those mainly addressed "NSL-KDD" [28] (available http://www.unb.ca/cic/datasets/index.html) as training and testing database for performance control experiment is considered. Consequently, any dataset mentioned to in this review part should be understood as NSL-KDD. The basic taxonomy IDS divides IDS solutions into "Anomaly oriented" and the other type is "Signature oriented" [9]. Anomaly-Oriented identification can be defined as a behavioral-based IDS, it observes and tracks any variations in the usual activity (normal) within a system through constructing a profile of the system which is being monitored [15], [29]. The key advantage is that it provides

the ability to detect and identify attacks which are unseen before to the system [30]. Anomaly detection is classified into: self-learning and programmed, based on the way a normal profile of a system is structured. The self-learning is a class of systems that operate mainly by baseline examples for normal behavior. While programmed model requires an expert to teach the system the basic rules those can be used to differentiate between normal and attack behaviors. Meanwhile, Signature based detection draws a rules set these are employed to identify the patterns. In case of a mismatch incident is raised, it declares a warning [31], [32]. The key advantage in that approach is that it has the capability to identify the attacks and raising a relatively low false positive alarm ratio.

One of the fundamental work found in related-work, R. S. Naoum et al [33], they developed Neural Network with improved "resilient back-propagation". The dataset was divided as (training 70%), (validation 15%) and (testing 15%). When they experimented the usage of unlabeled data for testing, the performance degraded dramatically. Chae [34] exploited "J48 decision tree" and employed eliminated feature set of 22 features (The full set consists of 41 features). Thaseen et al [35] developed a close work that measured the accuracy of different popular "supervised tree-based classifiers", they concluded that "Random Tree Model" (RTM) gained the optimal accuracy considering also the false alarm rate. Another approach of "2-level cascading classification" has been explored in different research trials: Panda et al [36] employed "Discriminative-Multinomial Naïve Bayes" as the base level classifier and "Nominal to Binary" supervised filtering at the second consequent level. Eid et al [40] implemented another 2-level Architecture using "Principle Component Analysis" (PCA) for feature reduction followed by (SVM), the system considered the full features set and gained reasonable accuracy for training set. Another noticeable work that used unsupervised clustering algorithms by Syraif et al [38].

A new trend for building (IDS) is applying "Deep Learning" with various models, Jihyun et al. [39] exploited "Long Short Term Memory" (LSTM) model to RNN and trained the IDS using KDD Cup '99 dataset. They draw the results against the accuracy with other IDS classifiers, (LSTM-RNN) gained 96.93% accuracy with a rate of detection 98.88%. Also, Fiore et al. [11] proposed "Restricted Boltzmann Machine" (RBM) in anomaly detection by training a network with practical data traces extracted from a work station traffic. This research was intended to measure the accuracy of RBM to classify normal data and data infected by bot.

Also, Alom et al. [40] employed "Deep Belief Network" (DBN) abilities to detect intrusion via sequence of experiments. They trained DBN with NSL-KDD data to locate unknown attack on it. Furthermore, they achieved an accuracy of 97.5% which was compared with existing (DBN-SVM) and (SVM) classifiers which it out performed.

IV. Proposed Model

In this work, RNN and variations: Bi-Directional RNN (BRNN), Long Short Term Memory (LSTM) and Bi-Directional LSTM (BLSTM), are used to detect any anomaly in sequence of requests. The proposed models can learn the definition of being normal and abnormal from labeled datasets and can apply these knowledge for unseen requests and unseen harmful requests.

Why Deep Learning and RNN

- Deep learning is a wide range family of ML techniques considering many factors such as: complexity in the learning and structure data representations. The key distinguishing feature between other machine learning techniques and deep structure is the scalability when data increases.
- (DL) techniques can extract higher degrees of non-linear relationships among data just they demand a tremendous amount of data to detect and identify the patterns.
- (RNN) has applied successfully in many regression and classification problems.
- (RNN) has the capability to capture the knowledge from both the current time step and the previous ones. The hidden layer nodes accepts inputs from the current input layer besides the outputs of the same nodes in previous time window. And if the network trained in stateful mode then the network can learn the normal behavior from different combinations and sequences of actions.
- Recurrent Neural Network exposes a very effective technique for handling sequential-nature

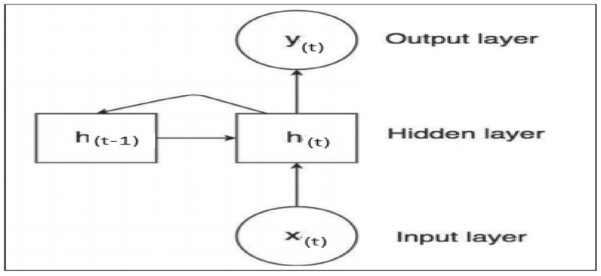

Fig. 2. RNN basic architecture.

input: The original difference between normal feed-forward networks and RNN is the existence of feedback loops in hidden units. These loops could produce the recurrent connection in the unfolded network. According to the recurrent nature and structure, RNN can express and model the contextual structure and information of a temporal sequence.

A. Recurrent Neural Network

According to sequence data, it's most probable statistically dependent in both forward and backward manner. Given request signatures (treated as time series) and as training data, the main objective is to extract and learn the rules to infer and predict the output data given the test input data. Inputs and outputs can be either continuous, categorical variables or both. In the case of continuous outputs, the problem is defined as a "regression", meanwhile is defined as "classification" when they are categorical (class labels). In this research, the term prediction is used as a general term that includes regression and classification and the problem is categorized as a classification problem. Fig. 2 shows a basic architecture for RNN, the input vectors are fed one at a time into the input layer of RNN. This network architecture could get benefit from all the available input information up to the current time frame, the current output of hidden units affects the next time step calculations. The structure and the training algorithm mainly control how much of this information is captured by a particular RNN.

It's very common that future input data seen later is usually useful for accurate classification or regression. With an RNN, this can be accomplished by delaying the output by a certain number of time steps to consider the future information.

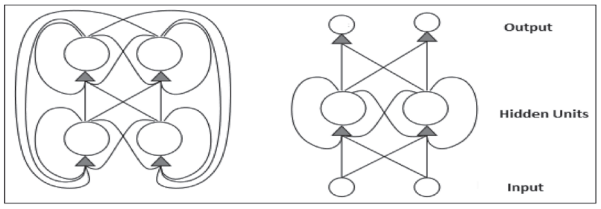

Fig. 3. RNN Models.

Let we consider the input sequence x= (x_0, x_1, x_2,..., x_{T-1}), the recurrent layer hidden states is given by:

h= (h_0, h_1, h_2,..., h_{T-1}). The output values of the output layer y= (y_0, y_1, y_2,..., y_{T-1}). The hidden value and output value is governed by [40]:

$$h_T = F(W_{xh}\, x_T + W_{hh} h_{T-1} + b_h) \quad (1)$$

$$y_T = O(W_{ho}\, h_T + b_0) \quad (2)$$

Where W_{xh}, W_{hh} *and* W_{ho} are the input-hidden, hidden-hidden and hidden-output weights respectively. (F) and (O) are the squashing functions for hidden and output layers. As mentioned before, RNN allows the parameters sharing through each layer, this sharing implementation eliminates the overall number of parameters needed to be tuned. The topology for RNN has many variations, Fig. 3 shows a schematic diagram for two types of RNN, the left figure illustrates the fully connected RNN in which each node percepts inputs from all other nodes. The right figure shows the partial connected RNN which the recurrence is delimited to the hidden layers.

RNN Training process is close to training a feed-forward Neural Network, since the parameters are shared, the gradient takes into consideration two basic measurements: the current time step and the previous time window steps and this process is called "Back-propagation Through Time".

B. Long Short Term Memory (LSTM)

RNNs have the capability to capture the knowledge of sequences could be mapped, these sequences have the property that I/P and O/P alignment is pre-known [41]. Although, RNN shown accuracy and success to learn the knowledge in certain problems these have "short-time lag" between both and desired responses [42], "short-term memory" inhibits its ability to model and handle practical sequence data processing accurately. To tackle

this problem, Schmidhuber et al. [42] have developed a modified recurrent architecture,"Long Short-Term Memory" (LSTM). These networks architecture employs a node called "Constant Error Carousel" (CEC), which permits for the propagation of some "constant error signal" via time. Also, LSTM controls the "Constant Error Carousal" access through using "Multiplicative Gates"

The main idea of (LSTM) model revolves around what we call "Memory Cell" (M) which basically translates and caches the inputs knowledge over that time. (M) is processed and by gates which are basically "sigmoid based functions". The gates make the decision whether the (LSTM) still caches the value from the gate simply forgets it, Fig. 4 [42]:

- Input gate (I) determines LSTM reaction toward the current input (Xt).
- Forget gate (F) that permits the LSTM to discard the previous cached value in memory (Mt-1).
- Output gate (O) that controls the memory to be transferred to the hidden state (h_t).

Where θ and δ are the non-linear sigmoid and non-linear hyperbolic relatively, $\mathcal{O}$ denotes the product and are the trained weights parameters. Based on supervised training mechanism, once the error signal reaches (M) output it is multiplied by the output gate. Then it enters M's linear (CEC) where it can propagate backward without exposing to be altered. This causes that (LSTM) has the ability to bridge the lags in time lags (even for long interval) between input and desired signals. Forget gate learns to reset the memory blocks which have expired and useless contents. It's important to notice that the reset operation does in a gradual form that mainly corresponds to slowly fading cell states.

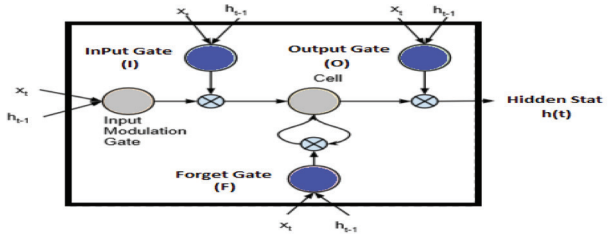

Fig. 4. LSTM unit.

$$I(t) = \theta(W_{xi}X_t + W_{hi}h_{t-1}) \quad (3)$$

$$F(t) = \theta\left(W_{xf}X_t + W_{hf}h_{t-1}\right) \quad (4)$$

$$O(t) = \theta(W_{xo}X_t + W_{ho}h_{t-1}) \quad (5)$$

$$M(t) = F(t)\odot M(t-1) + I(t)\odot\delta(W_{xc}X_t + W_{hc}h_{t-1}) \quad (6)$$

$$H(t) = O(t)\odot\delta\left(M(t)\right) \quad (7)$$

C. RNN and LSTM Variations

Bi-directional RNN (BRNN) mainly depends on the core of "the output value at time (t) might not only rely on the preceding sequence elements, but also the expected future ones. For our problem, relying on the order of activities sequence may be misguiding. Bidirectional RNNs could be simply considered as 2 stacked RNNs stacked and the output is mainly a dependent variable on the hidden state for these 2 stacked RNNs, Fig. 5.

The proposed solution captures the knowledge from large labeled dataset, NSL-KDD, using one of RNN models described above. Although the dataset is labeled, RNN stage behaves in an unsupervised manner. These recurrent networks captures the normal and abnormal activities in a descriptive manner. The proposed system adds a supervised layer (Fully Connected Multi-layer Perceptron) to classify the inputs in 2 ways, Fig. 6:

- Binary classifier (normal/abnormal) behaviors
- 5-class classifier (normal and 4 abnormal categories).

V. Experimental Results

As mentioned in introduction, NSL-KDD dataset was used in the proposed work which is a reduced and enhanced version of "KDD Cup 99" dataset. Although it

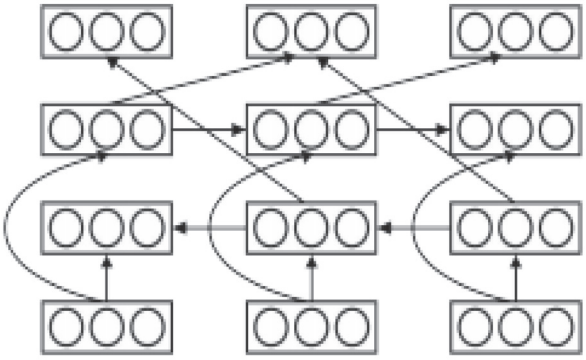

Fig. 5. A schematic diagram for BRNN.

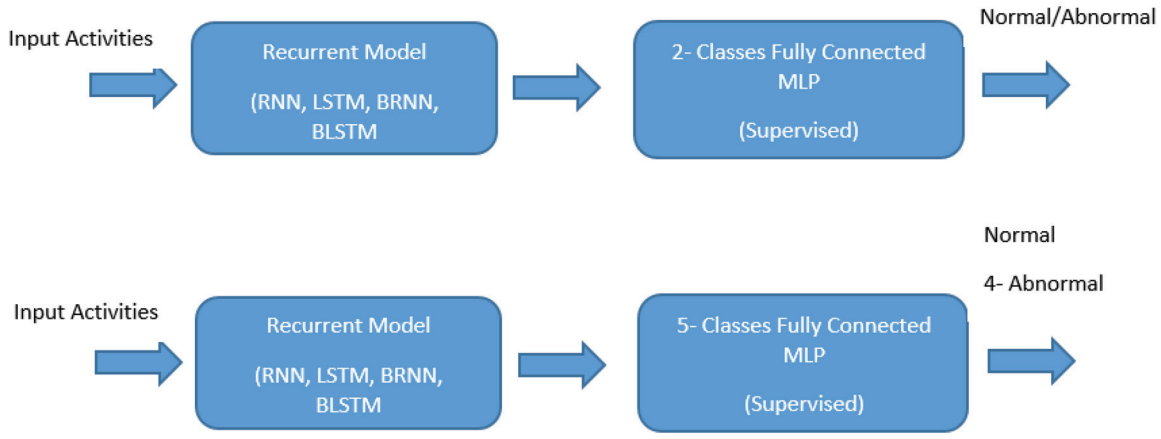

Fig. 6. System Implementation.

was suggested and designed to resolve major issues in the original version, the improved version still suffers from some of the issued, addressed by McHugh [43]. The paucity of available publicly allowed datasets for practical networks incidents, has caused the imperfection of real-network representation. NSL-KDD could still be exploited as a reliable dataset to measure various IDS approaches' accuracy.

The training have been collected for 7 weeks and the testing by the following two weeks tcpdump raw format. Testing file includes several attacks these intentionally were not exposed during the training phase. The purpose of that is to be sure that the intrusion detection testing is realistic and practically used. The basic theory behind that is the fact that, it is highly likely that the new attacks can be generated from the already seen attacks with traceable changes. Therefore, the training and testing collections consists of 5 million and 2 million TCP/IP requesting records, respectively.

Every training/ testing instant in "NSL-KDD" dataset is made of total number of features 41 and the final column is training desired target (normal/ attack type). These features contain:

- 4 binary.
- 3 nominal features.
- 34 continuous.

The training set includes (23 traffic classes) these are configured as:

- 22 different classes of attack.
- 1 normal.

Meanwhile, the testing set includes (38 traffic classes) and is configured as follow:

- 21 attacks classes seen in the training data.
- 16 unseen attacks.
- 1 normal class.

As explained before, the proposed solution emits the feature engineering step, no need for feature extraction, selection or grouping. The performance of the proposed system is measured against the hyper-parameters of RNN. Different values are compared: training accuracy, Recall (R), Precision (P) and F-Measure (F) with learning rate and time-steps to analyze the model behavior variance against changes in hyper-parameters.

- Recall (R): is the ratio (%) of true positive records over the sum of true positive and false negative.

$$R=\frac{Tp}{Tp+Fn} * 100\% \quad (8)$$

- Precision (P): is the ratio (%) of true positive records over the sum of true positive and false positive.

$$P=\frac{Tp}{Tp+Fp} * 100\%. \quad (9)$$

- F-measure (F): is the harmonic mean of both recall and precision.

$$F=\frac{2xPxR}{P+R} \quad (10)$$

Different topologies have been evaluated, RNN topologies are exploited through all running steps, with a single connected hidden layer (recurrent) and zero direct

TABLE I
BINARY CLASSIFIER RESULTS

Time Steps	Precision				Recall				F-Measure			
	BLSTM	LSTM	BRNN	RNN	BLSTM	LSTM	BRNN	RNN	BLSTM	LSTM	BRNN	RNN
10	83.4	85.6	82.4	92.5	94.5	68.7	92.1	62.7	88.60	76.22	86.98	74.74
15	84.2	88.8	83.1	92.5	93.6	71.3	95.2	63.9	88.65	79.09	88.74	75.59
20	82	91.2	85.3	88.2	97.4	70.9	93.9	62.7	89.04	79.78	89.39	73.30
25	87.6	88.4	83.3	92.8	95.2	74.5	94.7	62.4	91.24	80.86	88.63	74.62
30	86.1	88.5	86	91.6	96.5	75	91	60.3	91.00	81.19	88.43	72.73
35	88.9	92.8	90	93.5	98.1	72.3	91.3	64.9	93.27	81.28	90.65	76.62
40	84.6	92.1	90.1	93.1	97.2	77.7	92.5	64.6	90.46	84.29	91.28	76.27
45	85.7	91.7	89.2	91.9	96.1	74.8	94.6	66.3	90.60	82.39	91.82	77.03
50	86.2	91.3	88.6	91.3	93.8	73.5	94.2	66.8	89.84	81.44	91.31	77.15

input/output connections. The LSTM (BLSTM) hidden layers contained 100 and 1000 blocks of one cell in each, and the RNN (BRNN) hidden layers contained 100 and 500 units. Every LSTM block internal activity is being squashed by "logistic sigmoids" as an activation function. The non-LSTM hidden layer units is squashed by logistic sigmoid between [0, 1]. All developed networks were trained with "Back-propagation Through Time". The output layers obtain softmax activations, and the cross entropy objective function was used for training. At each frame, the output activations were viewed as the posterior probabilities of input record. The results are based on 2 classifiers: the first is a binary classifier (normal/abnormal) behaviors, Table I, and the second is 5-class classifier (normal and 4 abnormal categories), Table II. **Note that these results are obtained for testing unseen dataset.**

Another study has been conducted over the fully connected consequent feed-forward network, this study addresses the recall measurement against the number of hidden layers and number of neurons in each layer, Table III and Table IV. These results obtained for 35 time step (Binary Classifier) and 25 time step (5-Classes Classifier)

Table III and Table IV concluded that a single hidden layer for fully connected network is better than using 2 hidden layers. Also, for binary classifiers 15 neurons in hidden layers gives the best recall while using 30 in 5-classes classifier gives the best. With respect of these measurements, the proposed system using Bi-directional LSTM outperforms other RNN models beside previous work mentioned in related work (section 3).

Another study has been conducted is the impact of hidden units over the training time. The training is implemented over GPU card (NVidia GeForce GTX 860M) and the implementation was in python and Keras. Fig. 7 draws the training time for different topologies through variant hidden units.

The impact of hidden units is almost linear for the training time over GPU, and BRNN gained the maximum training time. A training procedure we can enhance the accuracy is preprocess the sequences with feed-forward layers which can help by projecting the data into a space with easier temporal dynamics. This can improve performance for the whole system.

VI. CONCLUSION

In this work, the practical problems of existing IDS have been addressed and different Deep Recurrent Neural Networks models are proposed to solve these problems. The models have been implemented and tested on benchmark dataset, NSL-KDD, to compare the results against other systems. 4 different models have been tested and Bi-Directional LSTM showed superiority over other proposed RNN models and previously developed systems. The reason behind superiority of RNN in general that, it's ability to define normal behavior from large dataset and can be used to detect a new unseen threat. This work can

TABLE II

CLASSES BASED CLASSIFIER RESULTS

Time Steps	Precision				Recall				F-Measure			
	BLSTM	LSTM	BRNN	RNN	BLSTM	LSTM	BRNN	RNN	BLSTM	LSTM	BRNN	RNN
10	72.5	73.4	71.6	79.4	83.7	61.1	80	52.4	77.70	66.69	75.57	63.13
15	70.3	75.5	72.7	80.1	81.2	59.4	81.4	51.8	75.36	66.49	76.80	62.91
20	74.6	77.8	73.5	80	84.7	58.5	82.7	51.6	79.33	66.78	77.83	62.74
25	77.4	75.3	71.7	79.5	87.0	61.3	81.5	52.3	79.82	67.58	76.29	63.09
30	72.9	75.9	74.3	80.2	86.8	61.2	82.9	52.9	79.25	67.76	78.36	63.75
35	71.3	73.8	77.8	80.9	85.5	60.8	80.8	56.1	77.76	66.67	79.27	66.26
40	73.4	78.5	78.8	81.4	84.9	61.5	81.4	56	78.73	68.97	80.08	66.35
45	73.8	77.6	77.2	80.5	83.2	60.7	82.8	54.9	78.22	68.12	79.90	65.28
50	75	77.2	76.1	78.4	83.2	60.1	83	54.1	78.89	67.59	79.40	64.02

TABLE III

HIDDEN LAYERS IN FULLY CONNECTED FEED-FORWARD NETWORK (BINARY CLASSIFIER)

Number of neurons	1- Hidden layer				2- Hidden Layers			
	BLSTM	LSTM	BRNN	RNN	BLSTM	LSTM	BRNN	RNN
5	83.6	75.3	82.7	83.4	78.1	75.7	74.9	71.1
10	93.5	88.4	83.5	80.6	75.4	73.3	73.1	65.2
15	98.1	83.6	88.2	83.0	77.6	74.8	75.0	66.0
20	93.0	89.4	84.4	84.9	79.2	71.8	69.3	69.2
25	91.4	90.2	83.7	79.5	80.0	73.2	72.1	68.4
30	88.7	86.0	85.7	77.4	77.6	74.9	66.8	62.5
35	88.2	86.0	82.3	80.8	74.9	70.6	71.9	63.4
40	89.1	86.3	84.7	80.7	75.6	68.0	70.0	65.6

TABLE IV

HIDDEN LAYERS IN FULLY CONNECTED FEED-FORWARD NETWORK (5-CLASSES CLASSIFIER)

Number of neurons	1- Hidden layer				2- Hidden Layers			
	BLSTM	LSTM	BRNN	RNN	BLSTM	LSTM	BRNN	RNN
5	77.2	71.6	74.1	74.2	72.6	70.2	69.4	68.8
10	75.3	72.1	77.1	76.9	72.3	70.9	71.4	62.4
15	82.1	77.4	74.2	79.4	71.9	71.4	70.8	69.5
20	84.5	78.5	78.6	75.3	73.8	71.7	71.5	61.4
25	86.5	77.1	81.3	79.1	75.2	72.0	69.0	61.0
30	87.0	73.8	83.0	71.9	71.8	70.1	70.4	60.3
35	86.3	71.0	77.5	72.2	69.3	66.5	64.4	61.8
40	81.4	70.4	77.5	72.0	68.1	64.3	61.9	57.6

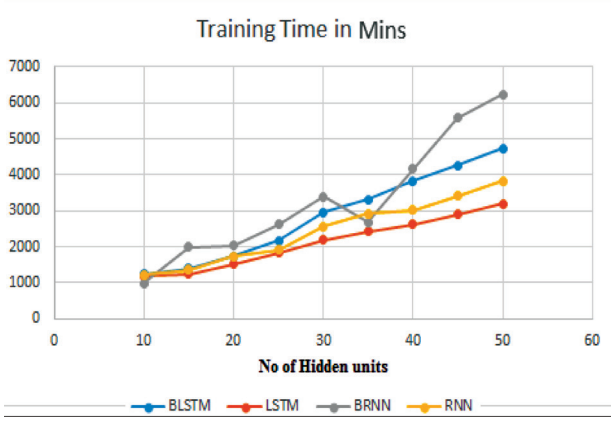

Fig. 7. Training Time in minutes.

be extended in 2 directions: first, implement other deep models and build a hybrid models and voting systems among different models to detect and identify threats with low false alarm. Secondly, provide the existing systems with real-world data for multiple networks such that the model can enhance its accuracy by adapting the definition of normal activities through different un-calibrated datasets.

References

[1] H. Abie and I. Balasingham, "Risk-Based Adaptive Security for Smart IoT in eHealth," *BodyNets'12: Proc. 7th Int. Conf. Body Area Netw.*, Oslo, NO, Feb. 2012, pp. 269-275.

[2] X. Luo, J. Liu, D. Zhang and X, Chang, "A large-scale web QoS prediction scheme for the Industrial Internet of Things based on a kernel machine learning algorithm," *Comput. Netw.*, vol. 101, pp. 81-89, June 2016, doi: 10.1016/j.comnet.2016.01.004.

[3] K. Xu, X. Wang, W. Wei, H. Song, and B. Mao, "Toward software defined smart home," *IEEE Commun. Mag.*, vol. 54, no. 5, pp.116-122, May 2016, doi: 10.1109/MCOM.2016.7470945.

[4] D. L. Clinefelter and C. B. Aslanian, "Online College Students 2015: Comperhensive Data on Demands and Performance," The learning House, Inc., Louisville. KY, USA, 2015.

[5] S. Mullainathan and J. Spiess, "Machine Learning: An Applied Econometric Approach," *J. Econ. Persp.*, vol. 31, no. 2, pp.87-106, Spring 2017.

[6] T. Le, Ji. Kim, Ja. Kim, H. Kim, "An Approach to Build an Efficient Intrusion Detection Classifier," *J. PLT. Tech.*, vol. 3, no.4, pp.43-52, Dec. 2015.

[7] B. Mukherjee, L. T. Heberlein and K.N. Levitt, "Network intrusion detection," *IEEE. Netw.*, vol. 8, no. 3, pp. 26-41, May/June 1994, doi: 10.1109/65.283931.

[8] G. Bruneau, "The History and Evolution of Intrusion Detection," SANS Inst., 2001. [Online]. Available: https://www.sans.org/reading-room/whitepapers/detection/history-evolution-intrusion-detection-344

[9] R. Mitchell and I. R. Chen, "A survey of intrusion detection techniques for cyber-physical systems," *ACM Comput. Surv.*, vol. 46, no. 4, pp. 1-29, Mar. 2014, doi: 10.1145/2542049.

[10] M. U. Farooq, M. Waseem, A. Khairi and S. Mazhar, "A Critical Analysis on the Security Concerns of Internet of Things (IoT)," *Int. J. Comput. Appl.*, vol. 111, no. 7, Feb. 2015, doi: 10.5120/19547-1280.

[11] U. Fiore, F. Palmieri, A. Castiglione and A. D. Santis, "Network anomaly detection with the restricted Boltzmann machine," *Neurocomputing*, vol. 122, pp. 13-23, Dec. 2013.

[12] S. Dua and X. Du, *Data Mining and Machine Learning in Cybersecurity*, 1st ed. New York, NY, USA: CRC press, 2016.

[13] M. Esmalifalak, L. Liu, N. Nguyen, R. Zheng and Z. Han, "Detecting Stealthy False Data Injection Using Machine Learning in Smart Grid," *IEEE Syst. J.*, vol. 11, no. 3, pp. 1644-1652, Sept. 2017. doi: 10.1109/JSYST.2014.2341597.

[14] H. Debar, M. Dacier and A. Wespi, "A revised taxonomy for intrusion-detection systems," *Ann. Télécommun.*, vol. 55, no. 7-8, pp. 361-378, Jul. 2000, doi: 10.1007/BF02994844.

[15] J. Shun and H. A. Malki, "Network Intrusion Detection System Using Neural Networks," presented at *2008 Fourth Int. Conf. Nat. Comput.*, Jinan, pp. 242-246, doi: 10.1109/ICNC.2008.900.

[16] A. Hassan, M. Nasser, S. Ahmed and K. Molla, "Feature Selection for Intrusion Detection Using Random Forest" *J. Inf. Secur.*, vol. 7, no. 3, pp. 129-140, Apr. 2016, doi: 10.4236/jis.2016.73009.

[17] L. Deng, "A tutorial survey of architectures, algorithms, and applications for deep learning," *APSIPA Trans. Sig. Inf. Process*, vol. 3, p. e2, Jan. 2014, doi: 10.1017/ATSIP.2013.99.

[18] Y. Bengio, A. Courville, and P. Vincent. "Unsupervised feature learning and deep learning: A review and new perspectives," arXiv:1206.5538v1, [Online]. Available: https://128.84.21.199/abs/1206.5538v1

[19] I. Goodfellow, Y. Bengio and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT press, 2016.

[20] D. G. Conorich, "Monitoring Intrusion Detection Systems: From Data to Knowledge" *Inf. Syst. Secur.*, vol. 13, no. 2, pp. 19-30, Dec. 2006, doi: 10.1201/1086/44312.13.2.20040501/81649.4

[21] M. Garuba, C. Liu and D. Fraites, "Intrusion Techniques: Comparative Study of Network Intrusion Detection Systems," in *Fifth Int. Conf. Inf. Technol.: New Gen.(itng 2008)*, Las Vegas, NV, 2008, pp. 592-598, doi: 10.1109/ITNG.2008.231

[22] T. Bansode, B.B.Meshram, "Intrusion Prevention System: for

End Users," presented at *Int. Conf. Ahmednagar*, March 2009.

[23] R. Heady, G. Luger, A. Maccabe and M. Servilla, "The architecture of a network level intrusion detection system," *Univ. New Mexico,* Albuquerque, NM, USA, Rep. LA-SUB-93-219, 1990.

[24] R. Bace and P. Mell, "NIST Special Publication on Intrusion Detection Systems," *Booz-Allen And Hamilton Inc*, Mclean, VA, Rep. ADA393326, 2001.

[25] B. Shanmugam and N. Idris, "Hybrid Intrusion Detection Systems (HIDS) using Fuzzy Logic," in *Intrusion Detection Systems*, P. Skrobanek, Ed, Rijeka, Croatia: InTech, 2011, ch. 8, pp. 135-155.

[26] S. Jonnalagadda and R. Reddy, "A Literature Survey and Comprehensive Study of Intrusion Detection," *Int. J. Comput. Appl.*, vol. 81, no. 16, pp. 40-47, Nov. 2013.

[27] J. McHugh, A. Christie and Julia Allen, "Defending Yourself: The Role of Intrusion Detection Systems" *IEEE Softw.*, vol. 17, no. 5, pp. 42-51, Sept.-Oct. 2000, doi: 10.1109/52.877859.

[28] M. Tavallaee, E. Bagheri, W. Lu and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *2009 IEEE Symp. Comput. Intell. Secur. Def. Appl.*, Ottawa, ON, 2009, pp. 1-6, doi: 10.1109/CISDA.2009.5356528.

[29] N. K. Mittal, "A survey on Wireless Sensor Network for Community Intrusion Detection Systems," *2016 3rd Int. Conf. Recent Adv. Inf. Technol. (RAIT)*, Dhanbad, pp. 107-111, doi: 10.1109/RAIT.2016.7507884.

[30] A. G. Tokhtabayev and V. A. Skormin, "Non-Stationary Markov Models and Anomaly Propagation Analysis in IDS," in *Third Int. Symp. Inf. Assurance Secur.*, Manchester, 2007, pp. 203-208, doi: 10.1109/IAS.2007.72.

[31] R. Mandal and S. Yadav, "An Improved Intrusion System Design using Hybrid Classification Technique," *Int. J. Comput. Appl.*, vol. 117, no. 10, pp. 20-23, May 2015, doi: 10.5120/20589-3028

[32] S. H. Vasudeo, P. Patil and R. V. Kumar, "IMMIX-intrusion detection and prevention system," in *2015 Int. Conf. on Smart Technol. Manage. Comput., Commun., Controls, Energy and Mater. (ICSTM)*, Chennai, pp. 96-101, doi: 10.1109/ICSTM.2015.7225396.

[33] R. Naoum, N. Abid and Z. Al-Sultani, "An Enhanced Resilient Backpropagation Artificial Neural Network for Intrusion Detection System," *Int. J. Comput. Sci. Netw. Secur. (IJCSNS)*, vol. 12, no. 3, pp. 11-16, Mar. 2012.

[34] H.-s. Chae, B.-o. Jo, S.-H. Choi and T.-k. Park, "Feature Selection for Intrusion Detection using NSL-KDD," *Recent Adv. Comput. Sci.*, pp. 184-187, Nov. 2013.

[35] S. Thaseen and C. A. Kumar, "An analysis of supervised tree based classifiers for intrusion detection system," in *2013 Int. Conf. on Pattern Recognit., Inform. and Mobile Eng.*, Salem, pp. 294-299, doi: 10.1109/ICPRIME.2013.6496489.

[36] M. Panda, A. Abraham and M. R. Patra, "Discriminative multinomial Naïve Bayes for network intrusion detection," in *2010 Sixth Int. Conf. Inf. Assurance Secur.*, Atlanta, GA, pp. 5-10, doi: 10.1109/ISIAS.2010.5604193.

[37] F. E. Heba, A. Darwish, A. E. Hassanien and A. Abraham, "Principle components analysis and Support Vector Machine based Intrusion Detection System," in *2010 10th Int. Conf. Intell. Syst. Des. Appl.*, Cairo, pp. 363-367, doi: 10.1109/ISDA.2010.5687239

[38] I. Syarif, A. Prugel-Bennett and G. Wills, "Unsupervised Clustering Approach for Network Anomaly Detection," in *Int. Conf. Netw. Digi. Technol.*, Berlin, Heidelberg, DE, 2012, pp. 135-145.

[39] J. Kim, J. Kim, H. L. Thi Thu and H. Kim, "Long Short Term Memory Recurrent Neural Network Classifier for Intrusion Detection," in *2016 Int. Conf. Platform Technol. Service (PlatCon)*, Jeju, pp. 1-5, doi: 10.1109/PlatCon.2016.7456805.

[40] M. Z. Alom, V. Bontupalli and T. M. Taha, Intrusion detection using deep belief networks," in *2015 Nat. Aerosp. Electron. Conf. (NAECON)*, Dayton, OH, pp. 339-344, doi: 10.1109/NAECON.2015.7443094.

[41] I. Sustskever, O. Vinyals and Q. Le, "Sequence to Sequence Learning with Neural Networks," *Adv. Neural Info. Process. Syst.*, pp. 3104-3112, 2014.

[42] F. Gers, N. Schraudolph and J. Schmidhuber, "Learning Precise Timing with LSTM Recurrent Networks," *J. Mach. Learn. Res.*, vol. 3, pp. 115-143, 2002.

[43] J. McHugh, "Testing Intrusion detection systems: a critique of the 1998 and 1999 DARPA intrusion detection system evaluations as performed by Lincoln Laboratory," *ACM Trans. Info. Syst. Secur.*, vol. 3, no. 4, pp. 262-294, Nov. 2000.