The use of machine learning algorithms for detecting advanced persistent threats.

EKE, H.N., PETROVSKI, A. and AHRIZ, H.

2019

The Use of Machine Learning Algorithms for Detecting Advanced Persistent Threats

Hope Nkiruka Eke*
h.eke@rgu.ac.uk
School of Computing Science and Digital Media, Robert Gordon University
Aberdeen, Scotland, UK

Andrei Petrovski
a.petrovski@rgu.ac.uk
School of Computing Science and Digital Media, Robert Gordon University
Aberdeen, Scotland, UK

Hatem Ahriz
h.ahriz@rgu.ac.uk
School of Computing Science and Digital Media, Robert Gordon University
Aberdeen, Scotland, UK

ABSTRACT

Advanced Persistent Threats (APTs) have been a major challenge in securing both Information Technology (IT) and Operational Technology (OT) systems. Due to their capability to navigates around defenses and to evade detection for a prolonged period of time, targeted APT attacks present an increasing concern for both cyber security and business continuity personnel. This paper explores the application of Artificial Immune System (AIS) and Recurrent Neural Networks (RNNs) variants for APT detection. It has been shown that the variants of the suggested algorithms provide not only detection capability, but can also classify malicious data traffic with respect to the type of APT attacks.

CCS CONCEPTS

• **Computing methodologies** → **Artificial intelligence**; **Machine learning**; *Neural networks.*

KEYWORDS

Advanced Persistent Threats(APTs), Artificial Immune System (AIS), Human Immune System (HIS), Long Short-Term Memory (LSTM), Recurrent Neural Network (RNN)

ACM Reference Format:
Hope Nkiruka Eke, Andrei Petrovski, and Hatem Ahriz. 2019. The Use of Machine Learning Algorithms for Detecting Advanced Persistent Threats. In *The 12th International Conference on Security of Information and Networks (SIN 2019), September 12–15, 2019, Sochi, Russian Federation.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3357613.3357618

1 INTRODUCTION

Threats to information and network security remain one of the biggest challenges facing organisations and industries at different levels of operation. There have been a number of successful breaches of critical infrastructure. Stuxnet is one example of a sophisticated APT attack purposefully launched to target critical nuclear infrastructure in Iran as highlighted in [9]. This type of attack has drawn special attention to the possibilities of APT attacks on the Industrial Control System (ICS) such as Supervisory Control and Data Acquisition (SCADA) network. It has also led to a number of research interests in developing methods to detect intrusions within network and isolated devices.

The application of Artificial Intelligence (AI) inspired by AIS [5, 19], Deep Learning (DL) [25, 31] and Machine Learning (ML) algorithm[13] in APT intrusion detection has attracted more research attention. Security practitioners believe that these approaches may be a solution to APT and other cybersecurity issues.

In spite of recent popularity of the above mentioned techniques and extensive academic research, there is a need to exploit this knowledge of the system in detection of operational behaviour since sensors are been controlled in real-time similar to how the human body immune system detects and removes or destroys threats from numerous pathogens such as viruses, bacteria and parasites [26, 33]. Various techniques and models have been developed and applied in securing Information system infrastructures such as ICS devices and different network devices and platforms against APT and cyber threats.

Sim et al [30] have applied the combination of immune metaphor with genetic programming in their work "Network for Lifelong Learning (NELLI) system". NELLI is a first step towards creating L2O systems that continue to adapt over time [15]. NELLI has been applied to bin-packing and job-shop scheduling domains. In any given domain, NELLI independently generates a group of optimisation algorithms that has the capability of solving a diverse range of problem instances. NELLI has demonstrated that, in order to improve its performance when exposed to more instances that exhibit different characteristics from those previously seen, it generates an ensemble of optimisation new algorithms that are capable of solving a diverse range of problem instances in any given domain. Also, since it retains memory, it quickly returns new algorithm that exhibits good performance when re-exposed to any instances it has seen in the past [30]. However, the NELLI model has not been tested for APT detection in the cybersecurity domain.

Hence, adaptation of artificial immune system combined with integration of optimised continues machine-learning approaches to predict attack instances offers a great potential to pre-generate algorithms in anticipation of future demand, thereby increasing the efficiency of the system. As suggested in [15], for a shift towards the direction of optimisation community rather than focusing effort on developing more complex algorithms that is trained on large static datasets, a move towards developing systems that independently and continually generate specialised algorithms on demand may bear considerable fruit.

https://doi.org/10.1145/3357613.3357618

This paper present the results of investigation of the application of deep learning optimised ensemble stacked RNNs and its variants as inspired by Life-Long Learning Optimiser (L2O) approach to enhance performance [15] in Intrusion Detection Systems (IDSs) and compared this with previously published results of AIS model - Negative Selection (NS) with Antigen Feedback (AF) [22].

The contribution of this paper can be summarised as follows:

- We propose a novel approach using deep neural networks for APT multi-step detection which takes stacked LSTM-RNNs networks to automatically learn features from the raw data to capture the malicious patterns.
- We carried out series of experiments to; evaluate the ability of this model to (i) accurately detect and classify an attack as abnormal and (ii) detect different type of attacks family accurately.
- The achieved results suggest that the proposed approach is a good candidate for developing attack detection systems.

The remainder of this paper is organised as follows. A brief background of the AIS and RNNs variants application in security domain is discussed in section 2. Experiments, evaluation metrics and analysis results are discussed in Section 3. Section 4 and 5 presents the conclusion of this paper and future work respectively.

2 BACKGROUND

This section contains a brief background of the AIS and RNN application in security domain through the examination of major components and basic definition.

2.1 Artificial Immune System (AIS)

An AIS is a system that is capable of self adaption, self-learning, self regulatory, distributed with self and non-self detection properties, capable of identifying and eliminating any intruding foreign body of antigenicity while maintaining the stability of the environment within the body [19, 28]. This is relatively similar to Intrusion Detection System (IDS) functionality in protecting the network systems.

Human body has an in built mechanism for protecting itself against harm from harmful bacteria and viruses, known as pathogens. This is achieved through the help of Human Immune System (HIS) without a previous knowledge of the pathogens structure [2]. Since HIS has the ability to detect and defend against previously and unseen harmful invaders, this approach can as well be adopted to protect computer system and critical infrastructure.

Application of AIS has gained popularity in many areas such as but not limited to processing of text [3], anomaly detection [12] and network security [17]. However, the most prominent AIS algorithms are centered around these four algorithms- (1) Artificial Immune network (AIN), (2) Clonal selection (CLONALG), (3) Negative Selection (NSA), and (4) Danger Theory and Dendritic Cell Algorithms (DCA), while other approaches are based on the combination of these four approaches [5, 10].

Four Major AIS Algorithms:

- Artificial Immune Network (AIN): was proposed by [18] which suggested that the immune system has the ability to attained immunology memory by the existence of a mutually reinforcing network of B cells as there is an interaction between its components to increase its tolerance and memory. Immune system cells work as a group in a networked system to eliminate any foreign body, this forms the basis for cooperative agents based IDS [5, 10].
- The Clonal Selection algorithm (CLONALG): was put forward by Frank in 1959 [7] This approach suggest that a clonal expansion of the original lymphocyte occurs when the original lymphocyte is activated by binding to the antigen; however, any low-affinity detectors (clone of the activated lymphocyte) are eliminated and replaced with the cloned detectors during the development of the lymphocyte.
- Negative Selection Algorithm (NSA): is based on negative representation of information using either string or real-valued vector representation [10]. There is a continuous improvement on the existing methods and new models being proposed, Bejoy et al. in [5] suggested that NSA is a major candidate for designing IDS algorithms.
- Danger Theory (DT) and Dendritic Cell Algorithms (DCA): was proposed by Matzinger in [23] which states that the immune system responds to danger signals from injured cells rather than self-nonself discrimination as danger signals should not be sent by healthy cells. Greensmith et al. in [14] proposed the Dendritic Cell Algorithm (DCA) which suggest that dendritic cells when stimulated, it differentiate and undergo maturation and migrate to secondary lymphoid tissues where they can only work if there is a danger signal by a cell when attacked by an antigen to stimulate an induce immune response

2.2 AIS Model Application in Intrusion Detection Systems (IDSs)

To implement AIS, this involves four different stages these includes (a) encoding, (b) similarity measure, (3) selection and (4) mutation as highlighted by [1], "Once an encoding has been fixed and a suitable similarity measure is chosen, the algorithm will then perform selection and mutation, both based on the similarity measure, until stopping criteria are met".

Gadi et al. in [12], applied AIS model in credit card fraud detection and compare their result to other classical classifiers such as Neural Nets (NN) and Bayesian Nets (BN), Naive Bayes (NB) and Decision Trees (DT).The result of this experiment indicates that AIS performed better when parameters optimised by Genetic Algorithm (GA) visualization procedure. Figure 1 is a representation of GA for parameters optimisation.

The following GA multi-resolution optimization steps algorithm are used in [12] are:

- They identified those parameters that have not changed, and freezes the values for the respective parameter.
- Parameters were screened and the 20 best parameter sets for each split and identify reasonable range.
- For all non-robust parameters, they choose an integer step s so the searching space does not explode
- Next, they evaluated the costs for all possible combinations according to the defined search space and find the parameter

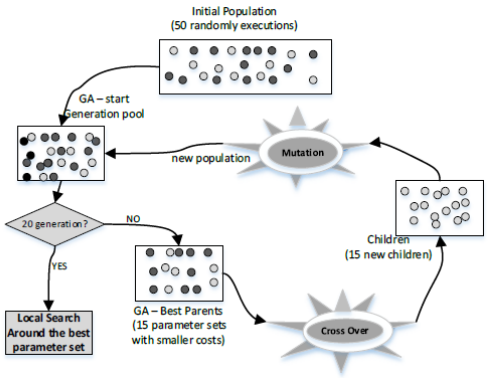

Figure 1: Genetic Algorithm for parameters optimization [12]

set P that brings the minimum average cost among all the different used splits

- Finally, zoomed the screen to the neighborhood of P, refine steps s, and repeat the process from then on, until no refinement is possible.

Investigation of system-level fault diagnosis using AIS model was carried by [35], in their work they have introduced AIS-based fault identification approach for multiprocessor and multi computer systems. This model emulated the ability of the immune system in recognising pathogenic agent (antigen) there by distinguishing body own cells and molecules (self) from foreign antigens (non-self) which was similar to the fault diagnosis problem, that aims at identifying processors (cells) in a system (body) to be a faulty (non-self) or fault-free (self). Their experimental results indicated that the immune diagnosis model can successfully identify the faulty processors and diagnose a faulty situation in short period of time.

An efficient proactive AIS based anomaly detection and prevention system (EPAADPS) was introduced by Saurabh et al. in [28]. EPAADPS was developed by combination of AIS ideas with agents to proactive defense system against unseen anomalies. Theses were achieved using three modules: Repertoire Training Module (RTM), Vulnerability Assessment Module (VAM) and Response module (RM). The authors utilised the NSA self-tuning of detectors and detector power in view to make a detector evolve and promote a better and correct self and non-self coverage. RTM generates and selects efficient detectors that forms detector set (DS) based on a self-tuning. VAM creates detector agents (DA) and assigns it Detector Set (DS) to evaluate Test Set instances. In a situation where VAM discovered any abnormality within the system, RM take action against the detected attacks.

A distributed multi-agent IDS using AIS approach that applied all four AIS algorithms were proposed by [29]. In this approach, elimination of low profile agents was done using NAS, the CLONALG was used to proliferate agents with best fitness value while agents communicate with each other was possible through AIN. This approach used mobile and static agents with detector agents as the main actors in MAIS-IDS. This approach also used NSL-KDD [11] dataset to evaluate the system based on three factors- accuracy, false alarm (FA) and detection rate (DR). According to the authors, this approach can be applied on both network and host based settings.

2.3 Recurrent Neural Network

Recurrent neural network (RNN) is an effective class of artificial neural network (ANN) that is used when dealing with very complex supervised and unsupervised tasks [20].

Recently, deep learning techniques have been applied in cyber security [24]. Since it can detect the cyber attacks by learning the complex underlying structure, hidden sequential relationships and hierarchical feature representations from a huge set of security data. The authors of [20] Proposed and evaluated RNN model against classical support vector machine classifier (SVM) for cybersecurity in Android malware classification, incident detection, and fraud detection.

RNN emerged as a powerful approach for deep learning architecture generally applicable for time-series data modelling. Despite the RNN and its variant networks remarkable performance in long standing AI sequence data modelling tasks such as time-series analysis, speech recognition and machine translation [21], applying the same in cyber security task is in early stage of development [34].

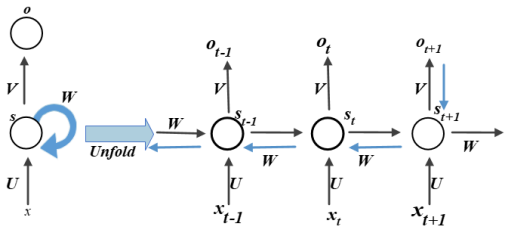

Figure 2: Schema of Unfolded Basic Recurrent Neural Network [4]

RNN were developed in the 1980s and it is similar to a feed-forward network (FFN) with an additional internal feedback loop (short-term memory to store and retrieve past information over time scales and thereby execute the temporal task) which is a circular connections between higher- and lower-layer neurons and optional self-feedback connections. These feedback connections enable RNNs to propagate data from earlier events to current processing steps which is in contrasts to FFN. The formulated mathematical representations of the RNN computational flow are represented in Equations 1 and 2.

$$s_t = f_w(s_{t-1}, x_t) \tag{1}$$

Equation 1 represent the basic recursive formula of RNN where fw is the recursive function. The network has an input layer x, hidden layer s (also known as hidden state) and output layer o. The current observed input to the network at time step t is denoted as x_t, the hidden state at time step t is denoted as s_t and s_{t-1} represent the previous hidden state.

$$s_t = f(Ux_t + Ws_{t-1}) \tag{2}$$

s_t is calculated based on the previous hidden state s_{t-1} and the input at the current step x_t as shown in Equation 2. s_t is the short-term "memory" of the network which captures information about

what happened in all the previous time steps, this can further be fed to other stacked recurrent layer or final layer where the layer has nonlinear activation function such as softmax function sf as represented in Equation 3.

$$o_t = sf(Vs_t) \tag{3}$$

o_t is the output at time step t and the vector of probabilities of initial hidden state is set to V.

f denote the nonlinearity mapping function from the input features to the output labels such as tanh or ReLU, where the weight matrices of the previous state weight at time step t and input state weight at time step t are represented as U_{x_t} and Ws_{t-1} respectively. s_{-1} is required to calculate the first hidden state and is usually initialised to zeroes.

Understanding the dynamics of RNN entirely is difficult due to its cyclic connection. To overcome this an RNN structure input sequence of length is transformed to a FFN structure by unfolding over time-steps as represented in Figure 2. FFN consist of hidden layers. This new structure can be analysed and also is adaptable to the backward propagation (BP) of errors, at this point the predefined error function are computed by comparing the output values with correct values and then distributed back throughout the network layers. This process is often used to train deep neural networks (DNN). An unfolded RNN at any given time t is defined as a function h_T Equation 4.

$$h_T = S_T(x_2x_1) \tag{4}$$

Where S_T represents the unfolded graph of time-steps t.

The sum of all input-output pairs in a sequence over all the time-steps is referred to as the loss L function represented in Equation 5.

$$L = d(tr, pr) = \sum_{i=1}^{T} d(tr, pr) \tag{5}$$

The schema of unfolding of RNN in time of the computation is shown in Fig. 4. In RNN cyclic connections, each layer represents per time information similar to DNN but the unfolded RNN shares weight parameters W across time-steps as represented in Figure 2. This indicates the fact that network performs the same task with various inputs over time-steps. In addition to learning the temporal patterns with cyclic connections, unfolding allows the RNN model to learn the association of static features between the input and output sequences. In order to apply the feedback concept, the BP is used to compute the gradients for weight parameters across time-step t.

To find the recurrent weights, the computation of gradient at $t = 2$ will involve back propagating 1 step and add them to find and update the recurrent weight . This technique is known as back propagation through time (BPTT) employed by RNN to reduce network cumulative error. However, modeling large scale data sequence with RNN and BPTT is not efficient due to vanishing and exploding gradient problem as stated in [6] which usually occurs when we BP the error back in many time-steps in the deep unrolled RNNs network models.

Since RNN shares the parameters across all time step reducing the amount of parameter to be trained, this is utilised to calculate the gradient at each time t as in Equation 6 and 7.

$$s_t = tanh(W_{xs}x_t + W_{ss}s_{t-1} + b_s) \tag{6}$$

$$o_t = softmax(w_{so}s_t + b_0) \tag{7}$$

where s_t is the hidden layer, $tanh$ is the hidden layer nonlinear activation function, the softmax function sf is used at the last layer as, b_s and b_o are the bias terms for the hidden state and prediction at time step t. Where the prediction at time step t is denoted as z_t while the weight shared between the hidden s and output o across all the time sequence are W_{os} and W_{ss} respectively.

2.4 Long Short-Term Memory (LSTM)

LSTM is a second order RNNs that is augmented by recurrent gates known as Forget Gates (FG) [16]. LSTM has the capability to remember information for long periods of time. It contains a memory block which is a complex processing unit that is composed of one or more memory cell and a pair of multiplicative gates known as input and output gate with in-built recurrent connection value 1 as constant error carousel (CEC). This value will be active across the time-step and triggered when a memory block has not received any value from outside signals [27]. In order to combat the issue of vanishing gradient that prevents RNN from learning long term dependencies through gating mechanism, the computation of recurrent hidden state S_t can be seen as mean elementwise multiplication as shown in Equation 8

$$\begin{aligned} i &= \sigma(x_tU^i + S_{t-1}W^i) \\ f &= \sigma(x_tU^f + S_{t-1}W^f) \\ o &= \sigma(x_tU^o + (S_{t-1}or)W^o) \\ g &= \tanh(x_tU^g + s_{t-1}orW^g) \\ c_t &= c_{t-1} + goi \\ s_t &= \tanh(c_t)oo \end{aligned} \tag{8}$$

LSTM comprises of one cell state c and three gates; the input i, forget f and output gates o used to illustrate interaction within LSTM architecture as represented within Equation 8. These gates are composed out of a sigmoid function σ that generate output vectors between 0 and 1 through elementwise multiplication operation o used in Equation 8 with another vector to decide how much of the newly computed state for the current input you want to let through by input gate i, the forget gate f defines how much of the previous state you want to let through. The output gate defines how much of the internal state to expose to next layer in time t step.

The candidate value g is calculated based on the current input and the previous hidden state, the input gate i will decide which part of this information to store in cell state memory c_t as the new hidden state g. The internal memory of the unit is denoted as c_t, this is the combination of the previous memory c_{t-1} multiplied by the forget gate f , and the newly computed hidden state g , multiplied by the input gate.

Finally, the hidden state output state s_t can be computed by multiplying the internal memory c_t with output gate o.

2.5 Gated Recurrent Unit (GRU)

GRUs are gating mechanism in recurrent neural networks. Its performance on polyphonic music modeling and speech signal modeling are similar to that of LSTM with fewer parameters due to there lack of output gate. A GRU has two gates, a reset gate r, and an update gate z shown in Equation 9

$$\begin{aligned} z &= \sigma(x_t U^z + S_{t-1} W^z) \\ r &= \sigma(x_t U^r + S_{t-1} W^r) \\ h &= \tanh(x_t U^h + (S_{t-1} o r) W^h) \\ s_t &= (1 - z) o h + z o s_{t-1}) \end{aligned} \tag{9}$$

The computation of GRU gating mechanism to learn long-term dependencies in neural network is similar to that of LSTM with few variation as listed below.

- GRU has two gate; the reset gate that determines how to combine new input with the previous memory, and the update gate decides how much of the previous memory to keep
- GRU does not have output gate o.
- GRUs does not have internal memory c_1 that differs from the exposed hidden state.

3 EXPERIMENT

The purpose of this study is to examine the performance of two different approaches, the AIS-NSA and LSTM-RNN in APTs detection. In this study, we have carried out two different tasks that involves LSTM-RNN model application using corrected 10% KDDCup99 dataset containing 494021 records. The first task focused on deriving hyper-parameter values for best performance model. In the second phase, we applied the achieved hyper-parameter values in measuring the model performance. We also compared the result of LSTM-RNN model to previously published AIS-NSA[22, 27] and [32] application.

All the standard data mining processes such as data cleaning and pre-processing, normalisation, visualisation and classification were implemented in Python. The batch size of 64 and epochs are run up to 50 and 300 with a learning rate set in the range of 0.01-0.5 on a GPU-enabled TensorFlow network architecture. Also, various traditional ML classification algorithms were used to perform the classification experiments on 10% KDDCup99 dataset in order to analyse the network protocol relationship with the attack used by intruders in generating anomalous network traffic. The ML classification result was compared to LSTM-RNN result in order to further evaluate the performance of LSTM-RNN model. The KDDCup99 dataset used consists of 22 attacks classes and 1 normal class. These attacks were grouped into 4 main attack classes - Denial of Service (DOS), Remote to User (R2L), User to Root (U2R) and Probe. Figure 3 and 4 shows the number of records in each of the classes. All features were used as input vector with 75% as training set and 25% as testing set for the binary and multi classification respectively . The training dataset were normalised from 0 to 1. This was trained using sigmoid activation function through time with ADAM optimiser, sigmoid function was used on all the three gates and categorical/binary cross entropy as loss function for multi and binary classification respectively.

3.1 Experimental Data

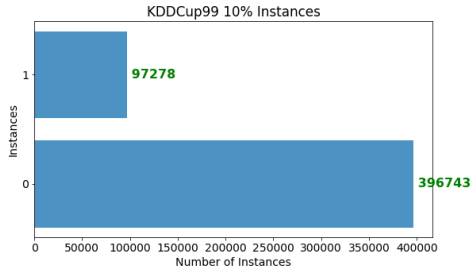

Figure 3: Visualisation of data classes in KDDCup99 dataset, with classes defined as (0) for normal and (1) for attacks (DOS, Probe, R2L and U2R as shown in Figure 4)

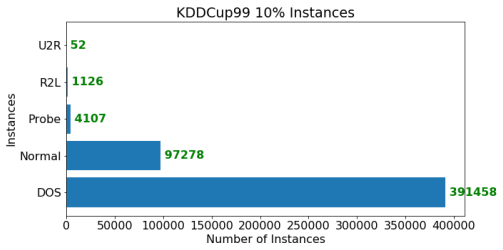

Figure 4: Four Main Attack Group and Normal Classes

3.2 Evaluation Metrics

The true positive rate (TPR) and false positive rate (FPR) were used to evaluate the effectiveness of LSTM-RNN model.

- True Positive (TP) - abnormal instances correctly predicted as abnormal.
- True Negative (TN) - normal instances correctly predicted as normal
- False Positive (FP) - normal instances incorrectly predicted as abnormal
- False Negative (FN) - abnormal instances incorrectly predicted as normal

3.3 Results and Discussions

To validate the approach of using the LSTM-RNNs model for detecting attacks statistical matrices such as accuracy (Acc), precision (Prec), true positive rate (TPR), false positive rate (FPR), recall (Rec) and f-score are calculated (i) to evaluate the ability of the LSTM-RNNs model to accurately detect and classify an attack as abnormal and also (ii) to check the ability of this model to detect different type of attacks accurately.

Table 1 contains an expanded name of all the algorithms as used on this paper. Table 2 contains the comparative summary result of the ML algorithms, the LSTM-RNNs network and result of previously published AIS (NS-AFB to be more precise), while Table 4 shows the summary of the performance of each of the algorithms

in detecting all the four attack groups including 1 normal class. This is also shown in Figure 12, where AG represent algorithms, DT represent detection and classification rate - other abbreviations have been mentioned previously.

Table 1: List of Expanded Algorithms Names as Used

Algorithms (AG)	Expanded Algorithms Name
SVM	Support Vector Machine
KNN	k-nearest Neighbors
DTC	DecisionTree Classifier
RF	Random Forest Classifier
LR	Logistic Regression
ADB	AdaBoost Classifier
NB	Naive Bayes
LSTM	Long Short-Term Memory
RNN	Recurrent Neural Network
GRU	Gated Recurrent Unit
AIS	Artificial Immune System
(NS-AFB)	Negative Selection with Antigen Feedback

Table 2: Average Binary Summary Results On 10% of KDD-Cup99 Dataset

AG	DT	Acc	TPR	FPR	Prec	Rec	F-score
SVM	0.994	0.981	0.998	0	1	0.999	1
KNN	0.999	0.999	0.998	0	1	0.999	1
DTC	0.999	0.999	0.998	0	1	0.999	1
RF	0.999	0.999	0.998	0	1	0.999	1
LR	0.997	0.998	0.993	0.001	0.999	0.998	0.998
ADB	0.998	0.999	0.997	0.001	0.999	0.999	0.999
NB	0.945	0.945	0.784	0	1	0.932	0.965
LSTM	0.999	0.999	0.999	0	1	1	1
RNN	0.999	0.999	0.998	0.001	0.999	1	0.999
GRU	0.999	0.999	0.998	0	1	1	1
AIS[22] (NS-AFB)							
Attacks	0.952	-	0.998	0.479	-	-	-
Normal	0.992	-	0.998	0.790	-	-	-

- `Confusion Matrix`: The model was trained on the full development training set and scores are computed on the full evaluation set. The confusion matrix of the LSTM-RNNs shows the predicted and the actual true binary classifications of normal/attack and detection of all the four attacks group for each of the RNNs as represented on Figure 5-7 and 8-10 respectively. Visual observation of the Figure 4 shows a clear picture of the number of instances of the R2L, U2R and Probes with lower connection records while normal and DOS appear to have more connection records. Those group with more records are learnt properly without confusing their identity while those with fewer connection records during training did not show good true positive rate and precision as it was had to identify them. This indicates data imbalance problem. The dataset contains many examples for "neptune" that belongs to DOS attack class, "satan" attacks that belongs to Probe and "normal" but fewer examples of the others.

The LSTM-RNNs model was used as classifier and detector. As a binary classifier to separate normal from attacks instances, the LSTM-RNNs were able to achieve a significant result of 99.99% average accuracy. A closer observation of the individual performance of each of the RNNs indicates that LSTM model outperformed the RNN with insignificant result, indicating an outstanding overall performance of this model as indicated with good DOS attack detection and acceptable detection of Probe.

Table 3: Performance Matrix Table for Binary Classification

Criteria	LSTM	RNN	GRU
Total No of Samples		494021	
True Positive (TP)	32187	32177	32178
False Positive (FP)	34	44	43
True Negative (TN)	130765	130734	130766
False Negative (FN)	41	72	40
True Positive Rate (TPR)	99.90%	99.80%	99.80%
False Positive Rate (FPR)	0	0	0
		Score	
Accuracy	99.90%	99.90%	99.90%
F-Score	1	99.90%	1
Recall	1	1	1
Precision	1	99.90%	1

Table 4: Average Multi-Class Summary Results On 10% KDDCup99 Dataset

AG	DT	Acc	TPR	FPR	Prec	Rec	F-score
SVM	0.991	0.999	1	0	0.999	0.999	0.999
KNN	0.999	0.999	1	0	0.999	0.999	0.999
DTC	0.999	0.999	1	0	0.999	0.999	0.999
RF	0.999	0.999	1	0	0.999	0.999	0.999
LR	0.997	0.998	1	0.001	0.998	0.998	0.998
ADB	0.923	0.924	0.998	0.205			
NB	0.931	0.931	1	0.033	0.984	0.931	0.954
LSTM	0.999	0.999	1	0	0.999	0.999	0.999
RNN	0.999	0.999	1	0	0.999	0.999	0.999
GRU	0.999	0.999	1	0	0.999	0.999	0.999

4 CONCLUSION

In this paper, we applied IDS based on LSTM-RNNs and evaluated the effectiveness of RNN model and its variants, we also compared our result to previously published work on AIS-NSA [22, 27] and [32] application. We went further to implement attacks classification with seven different classifiers as contained in Table 1, 2 and 4. The result from this classification were also compared to LSTM-RNN results. We noticed that most of the algorithms applied in this

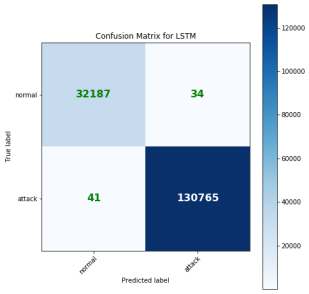

Figure 5: Binary Confusion Matrix for LSTM

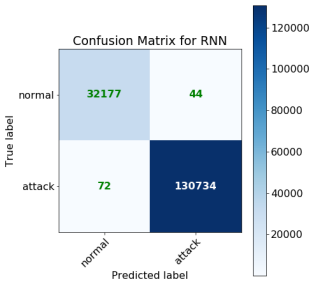

Figure 6: Binary Confusion Matrix for RNN

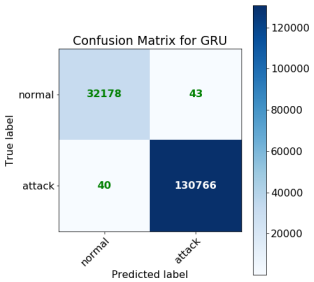

Figure 7: Binary Confusion Matrix for GRU

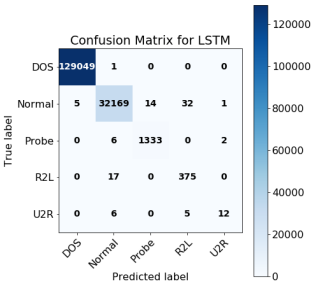

Figure 8: Multi-Class Confusion Matrix for LSTM

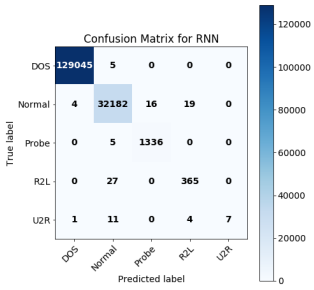

Figure 9: Multi-Class Confusion Matrix for RNN

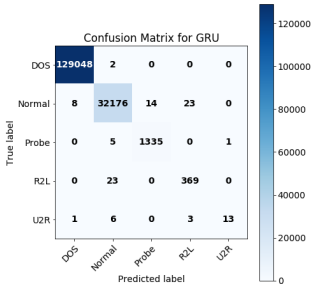

Figure 10: Multi-Class Confusion Matrix for GRU

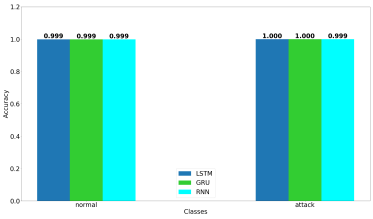

Figure 11: Normal and Attack Binary Detection Accuracy

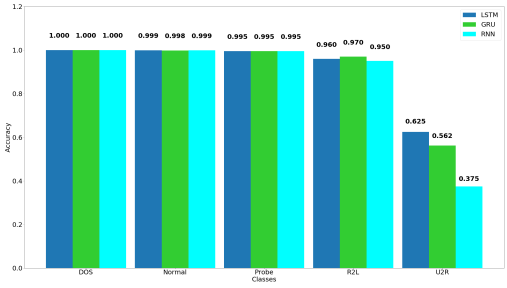

Figure 12: Performance Accuracy of Each Model on all Five Classes

study, achieved a competitive accuracy rate with insignificant FAR while few such as ADB and NS-AFB achieved a noticeable FAR, although NS-AFB achieved a good percentage detection accuracy of 95.20% on attacks, 99.20% on normal with TPR of 99.809% on both attack and normal classes. During the training, we also noticed that LSTM-RNN appear to be suitable for classifying high-frequency attacks and also the low frequency attacks with lower confidence prediction of 62.50%, 56.20% and 37.50% for LSTM, GRU and RNN respectively on multi attack detection, while achieving a very significant average accuracy of 99.99% for LSTM, GRU and RNN on differentiating attacks from normal instances. The percentage accuracy of LSTM-RNNs model achieved on this study as represented on the Figure 11 and 12 shows that the LSTM model performed slightly better than GRU and RNN model especially in differentiating attack from normal instances. Overall, the result suggest that the LSTM-RNNs model is a good candidate for developing attack detection systems.

5 FUTURE WORK

This work on the application of stacked LSTM-RNN model on IDS using a KDDCup99 dataset is an ongoing study. Further work will explore modelling combination of an optimised LSTM-RNN model and CNN on a time-series dataset - UNSW-NB15 datasets (University of New South Wales 2015 Datasets) over a multi-stage APT detection architecture. As APT is a multi-step attacks, detecting a single stage of an APT technique itself does not imply detecting an APT attack as mentioned by [13]. Patterns embedded in large generated datasets through industrial processes may be dynamic, hence the need for a system that can accurately detect APT in a systematic way at different time step and has the ability to learn, store and update existing patterns with the collection of new data, and also be scalable to process data in large volumes [8]. Hence, the combination of this two model to determine the efficiency of this approach since RNN has the capability to learn temporal dynamic behaviour over a time sequence data. The authors are currently engaged in work in this domain.

REFERENCES

[1] Uwe Aickelin, Dipankar Dasgupta, and Feng Gu. 2013. Artificial immune systems (intros 2). *arXiv preprint arXiv:1308.5138* (2013).

[2] Uwe Aickelin, Julie Greensmith, and Jamie Twycross. 2004. Immune system approaches to intrusion detection–a review. In *International Conference on Artificial Immune Systems*. Springer, 316–329.

[3] Luca Albergante. 2008. Wireless discussion forums: Automatic management via artificial immune systems. In *Performance Evaluation of Computer and Telecommunication Systems, 2008. SPECTS 2008. International Symposium on*. IEEE, 74–81.

[4] Abdelrahman Ayad, Mohsen Khalaf, and Ehab El-Saadany. 2018. Detection of false data injection attacks in automatic generation control systems considering system nonlinearities. In *2018 IEEE Electrical Power and Energy Conference (EPEC)*. IEEE, 1–6.

[5] BJ Bejoy and S Janakiraman. 2017. Artificial immune system based intrusion detection systemsâĂŤa comprehensive review. *Int J Comput Eng Technol* 8, 1 (2017), 85–95.

[6] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. 1994. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks* 5, 2 (1994), 157–166.

[7] Sir Frank Macfarlane Burnet et al. 1959. The clonal selection theory of acquired immunity. (1959).

[8] Meng-Hui Chen, Pei-Chann Chang, and Jheng-Long Wu. 2016. A population-based incremental learning approach with artificial immune system for network intrusion detection. *Engineering Applications of Artificial Intelligence* 51 (2016), 171–181.

[9] Thomas Chen and Saeed Abu-Nimeh. 2011. Lessons from stuxnet. *Computer* 44, 4 (2011), 91–93.

[10] Dipankar Dasgupta, Senhua Yu, and Fernando Nino. 2011. Recent advances in artificial immune systems: models and applications. *Applied Soft Computing* 11, 2 (2011), 1574–1587.

[11] L Dhanabal and SP Shantharajah. 2015. A study on NSL-KDD dataset for intrusion detection system based on classification algorithms. *International Journal of Advanced Research in Computer and Communication Engineering* 4, 6 (2015), 446–452.

[12] Manoel Fernando Alonso Gadi, Xidi Wang, and Alair Pereira do Lago. 2008. Credit card fraud detection with artificial immune system. In *International Conference on Artificial Immune Systems*. Springer, 119–131.

[13] Ibrahim Ghafir, Mohammad Hammoudeh, Vaclav Prenosil, Liangxiu Han, Robert Hegarty, Khaled Rabie, and Francisco J Aparicio-Navarro. 2018. Detection of advanced persistent threat using machine-learning correlation analysis. *Future Generation Computer Systems* 89 (2018), 349–359.

[14] Julie Greensmith, Uwe Aickelin, and Steve Cayzer. 2005. Introducing dendritic cells as a novel immune-inspired algorithm for anomaly detection. In *International Conference on Artificial Immune Systems*. Springer, 153–167.

[15] Emma Hart. 2017. Towards Lifelong Learning in Optimisation Algorithms. (2017).

[16] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[17] Hailing Huang, Weiqiang Guo, and Yu Zhang. 2008. Detection of copy-move forgery in digital images using SIFT algorithm. In *Computational Intelligence and Industrial Application, 2008. PACIIA'08. Pacific-Asia Workshop on*, Vol. 2. IEEE, 272–276.

[18] Niels K Jerne. 1974. Towards a network theory of the immune system. *Ann. Immunol.* 125 (1974), 373–389.

[19] Bin Jia, Zhaowen Lin, and Yan Ma. 2014. Advanced Persistent Threat Detection Method Research Based on Relevant Algorithms to Artificial Immune System. In *International Conference on Trustworthy Computing and Services*. Springer, 221–228.

[20] Soman KP et al. 2019. RNNSecureNet: Recurrent neural networks for Cyber security use-cases. *arXiv preprint arXiv:1901.04281* (2019).

[21] Y LeCun, Y Bengio, and G Hinton. 2015. Deep learning. nature 521 (7553): 436. *Google Scholar* (2015).

[22] Wanli Ma, Dat Tran, and Dharmendra Sharma. 2008. Negative selection with antigen feedback in intrusion detection. In *International Conference on Artificial Immune Systems*. Springer, 200–209.

[23] Polly Matzinger. 2002. The danger model: a renewed sense of self. *Science* 296, 5566 (2002), 301–305.

[24] Christopher D McDermott, John P Isaacs, and Andrei V Petrovski. 2019. Evaluating Awareness and Perception of Botnet Activity within Consumer Internet-of-Things (IoT) Networks. In *Informatics*, Vol. 6. Multidisciplinary Digital Publishing Institute, 8.

[25] Christopher D McDermott, Farzan Majdani, and Andrei V Petrovski. 2018. Botnet detection in the internet of things using deep learning approaches. In *2018 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 1–8.

[26] Lalit K Mestha, Olugbenga M Anubi, and Masoud Abbaszadeh. 2017. Cyber-attack detection and accommodation algorithm for energy delivery systems. In *2017 IEEE Conference on Control Technology and Applications (CCTA)*. IEEE, 1326–1331.

[27] Marek Ostaszewski, Franciszek Seredynski, and Pascal Bouvry. 2006. Immune anomaly detection enhanced with evolutionary paradigms. In *Proceedings of the 8th annual conference on Genetic and evolutionary computation*. ACM, 119–126.

[28] Praneet Saurabh and Bhupendra Verma. 2016. An efficient proactive artificial immune system based anomaly detection and prevention system. *Expert Systems with Applications* 60 (2016), 311–320.

[29] Neda Afzali Seresht and Reza Azmi. 2014. MAIS-IDS: A distributed intrusion detection system using multi-agent AIS approach. *Engineering Applications of Artificial Intelligence* 35 (2014), 286–298.

[30] Kevin Sim, Emma Hart, and Ben Paechter. 2015. A lifelong learning hyper-heuristic method for bin packing. *Evolutionary computation* 23, 1 (2015), 37–67.

[31] Ralf C Staudemeyer. 2015. Applying long short-term memory recurrent neural networks to intrusion detection. *South African Computer Journal* 56, 1 (2015), 136–154.

[32] Thomas Stibor, Jonathan Timmis, and Claudia Eckert. 2005. A comparative study of real-valued negative selection to statistical anomaly detection techniques. In *International Conference on Artificial Immune Systems*. Springer, 262–275.

[33] S Venkatesan, Ramachandran Baskaran, C Chellappan, Anurika Vaish, and P Dhavachelvan. 2013. Artificial immune system based mobile agent platform protection. *Computer Standards & Interfaces* 35, 4 (2013), 365–373.

[34] R Vinayakumar, KP Soman, and Prabaharan Poornachandran. 2017. Evaluation of Recurrent Neural Network and its Variants for Intrusion Detection System (IDS). *International Journal of Information System Modeling and Design (IJISMD)* 8, 3 (2017), 43–63.

[35] Hui Yang, Mourad Elhadef, Amiya Nayak, and Xiaofan Yang. 2008. Network fault diagnosis: an artificial immune system approach. In *Parallel and Distributed Systems, 2008. ICPADS'08. 14th IEEE International Conference on*. IEEE, 463–469.