International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Performing Dynamic Malware Analysis in Software Defined Network using LSTM Technique

Vasantharaj Karunakaran[1], Angelina Geetha[2]

Submitted: 16/07/2023 **Revised**: 09/09/2023 **Accepted**: 25/09/2023

Abstract: Methods for analyzing harmful or benign packets are crucial for enhancing security systems. Those packets can only be detected to a limited extent by current security methods. A malware analysis architecture by incorporating Control Plane and Data plane with the Software-Defined Networking (SDN). This article uses a deep learning model to classify malware in order to perform accurately and effectively. The Long Short-Term Memory (LSTM) model is trained using the system suggested in this work, which extracts a number of properties. We present a Secure SDN Simulation in this research that is controlled by a POX Controller and we propose an improved Long Short-Term Memory (LSTM), to achieve improved accuracy using a Confusion Matrix. To achieve this, we train and test an LSTM model using TensorFlow and Keras package. Long short-term memory (LSTM), which analyses all potential data points that can handle big datasets. The results showed that the sigmoid function performed better than other activation functions and "relu"(Rectified Linear Unit) activation layer that gives a better result with a 97.7% accuracy rate. This work can aid in the detection of malware and enhance security measures.

Keywords: *Control Plane, Data Plane, POX, Controller, accuracy, LSTM, sigmoid, relu*

1. Introduction

Network administrators can use the data from internet traffic analysis to make informed decisions and allocate resources more effectively [1]. With a traditional network traffic system, the complexity of classification is in the coding, but with ML, the difficulty is in the net suggestions gathered as well as the algorithm employed.

The capacity to learn is the key component of machine learning algorithms. This skill develops as a result of repeated practise and improvement. Each network traffic class has unique statistical properties of the net hint, such as packet length, duration, and inter-arrival period transmission. With the help of the provided training data set, the computer can identify this distinctiveness in the network pattern. The structure of the gathered internet data is determined using a machine learning technique. It can be difficult to distinguish between different apps for emailing, file transfers, streaming videos, and web browsing.

An important network management solution is necessary to handle the quick growth in network traffic in order to utilise network resources efficiently. Intelligence needs to be integrated into networking hardware for ease of organisation, optimization, maintenance, and management. It is challenging to use machine learning for device control because of the networking system's flexibility. The SDN platform makes it possible to include intelligence into interfacing devices [2]. Organization, optimization, maintenance, and administration of network resources are hampered by the complex structure of network information. Adding intelligence to network devices is one possibility.

The prospect of integrating intelligence into networking hardware has been made possible by the development of infrastructure, including GPUs, data processing frameworks like Spark and Hadoop, as well as machine learning libraries like scikit-learn and TensorFlow [3].

In a typical networking system, applications and a command-line interface (CLI) are used to manage the network devices. A step ahead is SDN with semi-automated network management. The most recent iteration of network administration, known as intelligence-driven (Defined) networks, is entirely automated. In order to regulate the network for cost reduction and throughput maximization, machine learning learns and improves network and application patterns.

The following networking-related operations can benefit from SDN.

1. Route Optimization in networking: The controller modifies the flow tables to route traffic in SDN stage. By looking at flow table rules, the control can decide whether to advance, drop, or block. At the controller, machine

[1]*Department of Computer Science and Engineering, Hindustan Institute of Technology and Science, Chennai, Tamilnadu, India*
[2]*Department of Computer Science and Engineering, Hindustan Institute of Technology and Science, Chennai, Tamilnadu, India*
Corresponding author email: vasantharajk35@gmail.com

learning methods are used to create an optimum routing path.

2. Provide Security in Networking: Using machine learning methods, a survey has been described on traffic reporting, security mechanisms and device identification for IoT strategies. [4] describes the issues of establishing security in IoT networks.

3. Supervised Learning: With this learning process, a knowledge base is created from which new flow situations can be categorized into previously recognized classes. We will show a number of relevant works that are connected to our suggested effort in the next part two. A improved Long Short Term Memory (LSTM) for performance of the Malware analysis is represented along with the incorporation of Keras and TensorFlow package.

Section 3 elaborates the Proposed methodology, where Section 4 discusses the experimental outcomes, with performance measured in terms of Accuracy, Precision, Recall, and F-Score and finally Section 5 concludes with discussion of the approaches' success and their potential application in the future.

2. Related Works

The location of the controller(s) in the SDN control plane is a crucial design decision that affects a variety of network issues, including latency, resilience, energy efficiency, load balancing, and other issues. [5] provided a thorough analysis of the controller placement problem (CPP) in SDN in this paper. Also they described the CPP in SDN and emphasise its importance. We present the traditional CPP formulation and the system model that supports it. We also go over a variety of CPP modelling options and related metrics.

Due to the enormous growth of SDN, the malware writers on the web take full advantage of this by developing new varieties of malware and disseminating them via various channels to impact millions of users. Malware is a real threat that exposes computer security. Numerous studies have been done to increase the effectiveness of detection techniques [6]. The malware's goal is to damage every file on the system and carry out malicious acts as shown by [7]. Because of the signature-based security solution software, malwares are challenging to detect and defensive mechanisms frequently fail. Sandboxing technology proposed by [8] identifies non-trusted code of the malware and ascertain their behaviour by looking at the behavioural analysis of the malware using the cuckoo sandbox.

An unknown victim is considered as Ransomware. Ransomwares are also malwares which provides less chances of recovering the data [9] as well as it is very tedious if it starts spreading. As per [10] the controller uses the OpenFlow protocol to connect only with SDN switches and add the required entries to their flow tables when malignant activity is identified. The controller, in particular, stops machines that are thought to be infected by keeping track of and responding in real-time to the network traffic they generate.

Windows also encourage Malware analysis. The research community's top concern right now is the constantly emerging new categories of malware, which is a an Internet threat that is harmful. Numerous methods have been employed, but they are unable to detect unidentified malware. To do so, the proposed work combines machine learning with dynamic malware analysis methods for malware classification and identification for Windows. It entails using the Cuckoo Sandbox Tool to run the executable offers a constrained environment with a minimal amount of uncovered resources for execution and post-execution analysis the behavior patterns statistics. The features and their count frequencies have been chosen using the JSON report that was generated by [11].

There has been a lot of interest in the field of Android malware detection in both academia and business. Studies on malware families in particular have helped with malware behavior analysis and detection. However, identifying malware family traits and the features that can characterise a specific family have received less attention in previous research. In order to enable fingerprinting the malware families with these features, we are motivated to investigate the key features that can categorise and describe the behaviors of Android malware families. [12] proposed twenty key features in three categories are used to build the fingerprints of ten malware families. Results of extensive SVM experiments show that the accuracy of the malware family classification ranges from 92% to 99%.

Information Classification is a typical work in Machine Learning [13][14]. Expect that a few provided information focuses each have a place with one of two classes, and our motivation is to figure out which class another information point will have a place with. Aside from grouping, upgraded traffic characterization is a famous point among scientists. Because of Support Vector Machines(SVM), this model tends to the overfitting issue. The scientists explored networks security risk assessment and broke down laid out risk evaluation techniques. Rather than past gamble evaluation draws near, SVM has demonstrated to be a progressive type of learning machine technique in light of the primary gamble minimization idea. SVM gives various benefits with regards to tending to design acknowledgment issues with short example sizes, nonlinearity, and high dimensionality [15] SVM and twofold tree standards are made sense of inside and out and afterward used to organize security risk evaluation. They exhibited that the SVM strategy has higher Classification accuracy, better speculation execution, and less learning and testing time when contrasted with the ANN technique regarding

arrangement accuracy, speculation execution, and learning and testing time, particularly for little examples.

Long Short Term Memory (LSTM)[16] is a profound learning engineering in light of fake repetitive brain organizations (RNNs) Deep Learning(DL). LSTM has criticism associations, dissimilar to ordinary feedforward brain organizations. It can deal with individual data of interest, (for example, photographs), yet in addition complete information transfers (like discourse or video). LSTM can be utilized for assignments like unsegmented, connected penmanship acknowledgment, [17] discourse acknowledgment, [18] and oddity location in network traffic or IDSs, for instance (interruption discovery frameworks). A cell, an information door, a result entryway, and a neglect entryway make up a normal LSTM unit. The three entryways control the progression of data into and out of the cell, and the cell recollects values across erratic time stretch

3. Proposed Methodology

By running the executable files in isolated environments, virtual machines (VM), or emulators to observe the executable file behaviour during the run-time and then obtaining the desired dynamic data, numerous research teams have used a dynamic analysis approach to collect various types of data to distinguish between malicious and benign files. A dynamic analysis technique has been used to collect data of various kinds. Attack vectors can be dynamically depicted by keeping track of executable file behaviour and memory images during run-time as provided by [19][20]. The combination of SDN with Machine Learning learning algorithms for performing malware analysis is presented in the suggested system. A supervised learning technique is used to create the model for classification. Machine learning-based traffic classification is proposed to make network management activities easier. In software-defined networking technologies, the system structure for traffic categorization by deep learning models is depicted in fig 1., which The three components of the proposed paradigm are machine learning, real-time networks, and virtual networks. By linking the design industry to a virtual network for mail and data transfer applications, the hybrid network is constructed. The machine intelligence component of the proposed methodology is applied for traffic categorization.

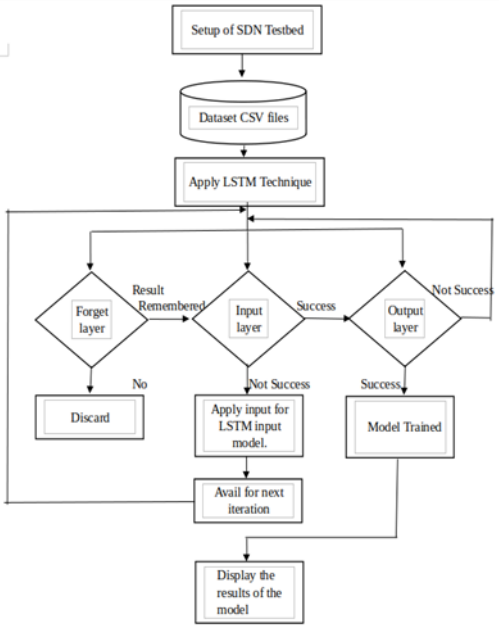

Fig 1. Architecture Diagram for the Proposed Work

The data packets are collected from Kaggle that flows through the network is managed through the Pox controller and detects kind of the traffic through the traffic detector. The traffic detector detects whether the packets are normal packets or anonymous data packets with elephant flow. If elephant flow occurs then the data is classified through the machine learning algorithms to classify it into.exe, dll, api or other files.

3.1 Software Defined Network (SDN)

The data traffic created by client side is classified using the Software Defined Network (SDN) architecture. In SDN, the transmitted data and the control plane are separated. The downstream interface between devices and controller is the Open Flow protocol. The north bound interface allows network applications to communicate with the controller. The topology's network dynamics are captured and delivered to the controllers for making decisions and network device management. For giving suitable control instructions and handling network dynamics, an adaptive control mechanism is provided, together with data analytics. The feedback mechanism offers network topology updates in the form of network status, such as connection failures or topological changes.

3.1.1 Topology

For wired network scenarios, there are three distinct topologies: single switch with 'n' hosts, straight topology about same controls as host, and tree like topology. The user's needs can be used to construct a bespoke topology.

3.1.2 Controller

The controllers are fundamental component of system. This controller allows for active network resource organization. Because the controller has a worldwide network view, it is feasible to monitor the devices. The

data transmission within the devices is handled by the POX controller. The Open Flow is being used to manipulate flow rubrics. The southbound interface is used to communicate between the devices connected in the architecture and controller. Communication with the north bound line is possible.

Data gathering, pre-processing, model creation, validation, prediction and labelling are all part of the network traffic categorization workflow. [21] proposed load balancing is used in the SDN system to maximize network resource use. The Quality of Service parameters are taken into account for categorization of traffic in the architecture by [22]. [23] describes how machine learning ideas are used in SDN for dynamic routing, QoE forecasting, resource planning, security, and traffic categorization.

3.2 Long-Short Term Memory (LSTM)

RNN-based architecture is used in deep learning for LSTM[24]. The Long short-term memory neural networks (LSTMs), for example, analyze one word at a time and provide the probabilities of probable values that the following word in a sentence may take in their states[25], which help with sentence comprehension. The kernel function and the scores of the neurons connected in the neural network are stored in a memory unit for feedback learning by the LSTM. The advantage of an LSTM over a traditional fully linked layer is that it can handle all data points, not just one. It gains power as a result. In our experiment, an LSTM model built with the Keras package and Tensorflow was used. Our hidden state neurons layer has the following dimensions: [(64,32), (32,16), (64,32, 32), (64,32, 32),]

The LSTM layer's architecture is shown below.

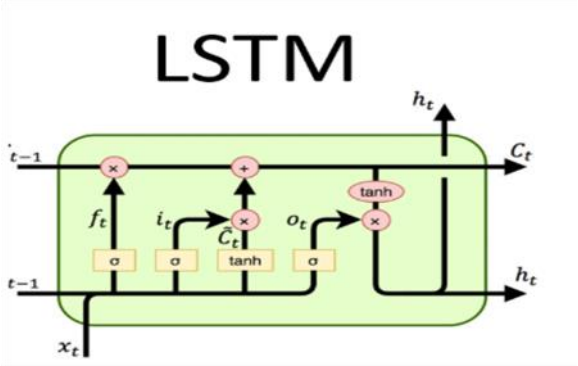

Fig 2. LSTM Architecture

3.3.1 Algorithm for Analyzing the Malwares

Step – 1 : Predict the learning rate, network weights and Hidden layer to define the LSTM networks.

Step – 2 : Collect the Dataset from (Di) from Kaggle and normalize into learning data rate values.

Step – 3 : Identify the learning rate and organize Di

Step – 4 : Train the network.

Step – 5 : end

Step – 6 : Run the Predictions

Step – 7 : Predict the success.

In LSTM, a block is a storage unit, and size is the amount

of space allotted to that block for storing memory pointers.

Long Short-Term Memory (LSTM) networks are a sort of recurrent neural network (RNN) architecture built for sequence data, and they can be implemented and trained using the popular deep learning frameworks Keras and TensorFlow. As the backend engine for Keras, TensorFlow enables you to construct neural networks, including LSTMs, using Keras as a high-level API while TensorFlow takes care of calculations and optimization.

Recurrent neural network (RNN) architectures such as Long Short-Term Memory (LSTM) are made to recognize and learn long-distance dependencies in sequential input. LSTMs can process and update data over a number of time steps thanks to certain equations that control their behavior. The main LSTM equations are shown below:

3.2.1. Forget layer

The "forget layer" is a crucial part of a Long Short-Term Memory (LSTM) neural network that is essential for managing information flow and memory retention within the network's memory cells. The "forget gate," which is connected to the forget layer, is used to decide whether or not to preserve certain data from the memory cell from the previous time step. In order to solve the vanishing gradient problem and make it possible for LSTMs to detect long-range dependencies in sequential data, this is crucial.

The below fig3 represents the working of the forget layer:

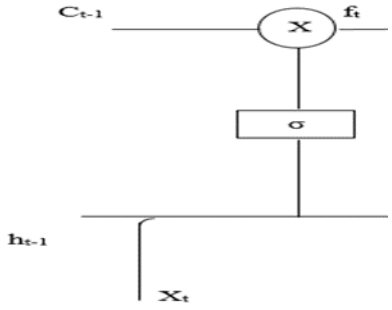

Fig 3. Interior function of Forget layer

The previous memory cell state (c_{t-1}) is multiplied element-wise by the forget gate. The memory cell's components whose associated forget gate values are close to 0 will be "forgotten" or muffled, but components whose forget gate values are close to 1 will be kept.

Algorithm for Forget layer

Step 0: Identify the previous hidden layer at timestamp (t-1), current input at time t.

Step 1 : Predict the Weight

Step 2 : Place the Sigmoid function σ

Step 4: Compute the f_t

Step 5: Update forget layer f_t along with C_{t-1} LSTM cell layer

The working of Forget layer is determined by the following equation:

$f_t = \sigma[(h_{t-1,}X_t).C_{t-1}]$ ---------- 1

f_t = Output of the forget layer

h_{t-1} = hidden state layer from the previous time stamp

X_t = input layer at time stamp t

c_{t-1} = cell state layer from the previous time stamp

The forget layer takes X_t as the initial input taken from our dataset along with the hidden state layer of the previous timestamp. Hence we start to train our model. Sigmoid layer acts as the activation layer for activating the inputs and then concatenate with the Cell state of the previous timestamp. The Parameter f_t acts as a result of the forget layer, which also acts as an input for the input layer. The main advantage of using LSTM when compared with the RNN model is it overcomes the Varnishing gradient issue, hence prevents collapsing the model.

3.2.2 Input layer

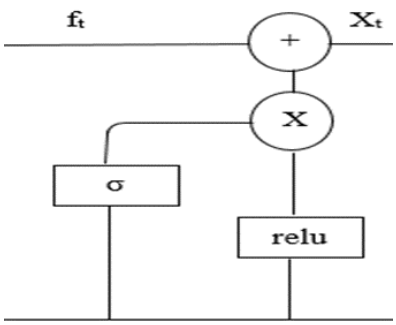

Fig 4. Interior function of Input layer

The input layer of a Long Short-Term Memory (LSTM) neural network is the first phase in which the network accepts input data and gets it ready for processing by the LSTM cells. The input layer carries out a number of tasks, including as converting input data into an appropriate format, communicating with the LSTM gates, and streamlining data transfer throughout the network.

Algorithm for Input layer

Step 0: Identify the previous hidden layer at timestamp (t-1), current input at time t.

Step 1 : Predict the Weight

Step 2 : Place the Sigmoid function σ and store the result.

Step 3 : Activate the input layer using Activation function "relu".

Step 4: Update the new input layer X_t and concatenate it along with Sigmoid function.

Step 5: Confirm the forget layer to forget or discard the f_t

The working of input layer is determined by the following equation:

$i_t = \sigma[W_i .(h_{t-1,}X_t) + b_i]$ ---------- 2

Here b_i is the bias vector for the input gate.

Sequences or time steps are the two types of input data that the input layer accepts. One or more features may be present in each time step. A three-dimensional array with the dimensions (batch_size, time_steps, and num_features) is typically how the input data is organized. The input data is divided into two portions at each time step: one portion is used to calculate the candidate cell state (c~t) through the input gate, and the other portion is utilized to affect the output gate.

3.2.3 Output Layer

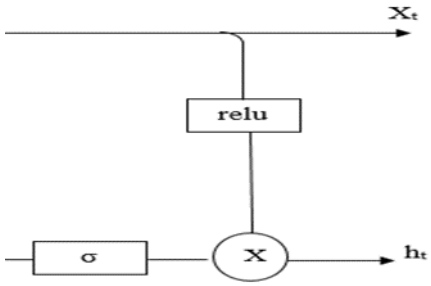

Fig 5. Interior function of Ouput Layer

The output layer of a Long Short-Term Memory (LSTM) neural network is in charge of producing the network's final output based on the data that the LSTM cells have learned and processed during the course of the sequence. Depending on the particular task the network is intended for, the output layer translates the hidden state of the final LSTM cell into the necessary output format.

Algorithm for Output layer

Step 0: Identify the previous hidden layer at timestamp (t-1), current input at time t.

Step 1 : Predict the Weight

Step 2 : Place the Sigmoid function σ and store the result.

Step 3 : Activate the input layer using Activation function "relu", for the new X_t value

Step 4: Update the new input layer X_t and concatenate it along with Sigmoid function.

Step 5: Update the output layer and store the result X_t and Ct

Step 6 : Repeat Step 5 for the next round, when X_t and Ct becomes X_{t-1} and C_{t-1}

Step 7 : Repeat the steps till the epoch satisfied.

Step 8 : End

The output layer is the result achieved, that can be calculated by the following equation:

$y_t = \text{Activation}(W_o \cdot ht + b_0)$ ---------- 3

Here y_t is the output layer and relu(Rectified linear unit) will be the activation function.

4. Results and Discussions

Long Short-Term Memory (LSTM) neural networks' results and conversations often entail assessing the performance of the trained LSTM model on a particular task or dataset, then interpreting and debating the results.

Model selection, training, and parameter adjustment are all done during model creation. LSTM method is chosen. The Keras and TensorFlow is used to train and test the models.

When it comes to creating and training deep learning models, such as Long Short-Term Memory (LSTM) models, Keras and TensorFlow are tightly linked. On top of TensorFlow (and other deep learning frameworks), Keras is a high-level neural network API that offers a straightforward and user-friendly interface for creating and refining deep learning models. Here is how LSTM models are produced using Keras and TensorFlow:

Import tensorflow as tf

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import LSTM, Dense

The above one is the command to import the Keras and Tensorflow packages in Python.

A Long Short-Term Memory (LSTM) neural network's typical training and testing times might vary greatly depending on a number of variables. These elements include the degree of model complexity, the amount of the dataset, the hardware configuration, and the particular task being carried out. As a result, it is difficult to provide a precise average training and testing duration because it relies on the particular circumstances. It is always necessary to minimize the Training time. To shorten training time, it's critical to monitor training progress, utilize the right hardware acceleration (such as GPUs), and take into account optimizations such batch processing and parallelism.

It can take several hours to several days to train an LSTM model on a moderately substantial dataset using a conventional configuration (e.g., a moderate number of LSTM layers and units). On large datasets, training particularly deep or complicated LSTM models could take several days or even weeks. The same tasks may take several times as long on a GPU as they do on a CPU during training. In our work, the model can able to work on both larger and smaller datasets.

Table 1. Confusion matrix for LSTM model

S.NO	**Supervised learning model**	**Confusion Matrix (N = 950)**				
		Actual/ Predicted	**.exe**	**.dll**	**.api**	**Others**
3	**LSTM**	**.exe**	290	0	6	1
		.dll	8	283	0	4
		.api	1	3	318	0
		Others	0	1	0	35

The confusion matrix is used to show the evaluation of the LSTM model in table 1. The four types of file extension taken for classification are .exe,.dll,.api and other type of files. The confusion matrix is represented for the number of data instances as 950. Figure 3, Figure 4 and Figure 5 depicts the performance of various layers on LSTM algorithms by means of Accuracy and Error Rate for data classification. The proposed method achieves up to 97% accuracy with very negligible error rate.

The learning rate is considered as the key parameter as it establishes the increment size at which the weights of the model are updated during training. Selecting the right learning rate is crucial because it influences the trained model's quality and convergence speed. Here are some important factors to take into account while choosing the learning rate for LSTM models. The below table – 2 shows the learning that is used to work in our model.

Table 2. Training of LSTM Record

Learning rate	Network weights	Hidden layers	Success rate
0.1	264	2 blocks, size 1	70%
0.2	264	2 blocks, size 1	83%
0.3	267	4 blocks, size 2	97%

The performance analysis for various LSTM different layers algorithms is depicted in Figure 3,Figure 4 and Figure 5 and table 1 by Accuracy, Precision, Recall and F-Score. The above said figures depicts the proposed

algorithms achieved better performance from 89% to 95% for all performance parameters.

The below graph represents that the above results obtained are true.

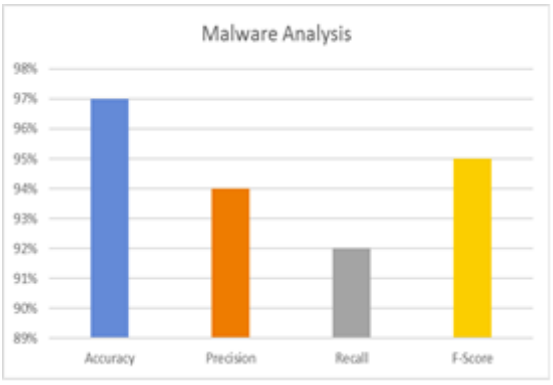

Fig 6. Malware analysis results

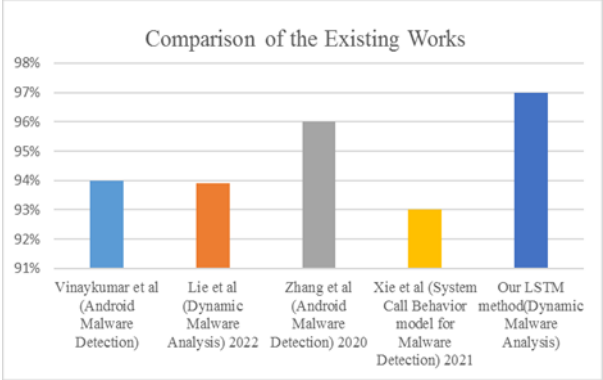

Fig 7. Comparison of the accuracy of the results

From the above figure – 7, X-axis represents the performance metrics of the different models and Y-axis represents the accuracy percentage.

The features of our work had been compared with various other works, as the performance of Malware analysis is a prolonged strategy. As our approach deals with LSTM layers algorithms and the results are displayed using Pandas. Some of the comparison works are shown in table – 3

Table 3. Comparison of the Existing works

S.No	Methodology used	OS used	Technology applied
1	Dynamic Malware analysis	Ubuntu 20.0	Python and Pandas(Analysis)
2	Cuckoo Sandbox (Jamalpur et al ., 2018)	Windows and Ubuntu	Cuckoo Sandbox
3	Cuckoo Sandbox (Irshad and Dutta)	Windows and Ubuntu	Cuckoo Sandbox
4	Ransomware Detection (Akbanov et al)	Ubuntu	Wannacry

5. Conclusion

In the SDN context, a supervised learning model for data traffic categorization is suggested. LSTM model is employed. Due to a thousand-fold growth in net data traffic, standard traffic categorization algorithms are limited. Recurrent neural network (RNN) architectures that excel at handling sequential data and time-series applications include Long Short-Term Memory (LSTM) networks. By including memory cells and gating mechanisms, which allow them to capture long-term relationships and prevent the vanishing gradient problem, LSTMs alleviate some of the drawbacks of conventional RNNs. The data is classified through the Forget layer, inut layer and outut layer algorithms to classify it into .exe, .dll,. api or other files. Sequential data analysis has been transformed by LSTM networks, which have also helped to solve a variety of practical issues. They have evolved into a core element in many deep learning systems and give the capacity to capture temporal dependencies. To realise the full potential of LSTMs, however, competent implementation and hyperparameter adjustment are necessary.The suggested framework for tool allocation in next-generation networks may be combined with deep more learning models in future.

Conflicts of Interest

The authors declare that there are no conflicts of interest.

Author contributions

The paper concepts and algorithm implementation, experimentation, validation, formal analysis, paper research, and original draft preparation of writing were developed by the first author. The project was supervised and managed by the second author.

References

[1] Wander Queiroz, Miriam A.M. Capretz, Mario Dantas, An approach for SDN traffic monitoring based on big data techniques, Journal of Network and Computer Applications, Volume 131, 2019, Pages 28-39, ISSN 1084-8045, https://doi.org/10.1016/j.jnca.2019.01.016.

[2] S. Srisawai and P. Uthayopas, "Rapid Building of Software-based SDN Testbed using SDN Owl," *2018 22nd International Computer Science and Engineering Conference (ICSEC)*, Chiang Mai, Thailand, 2018, pp. 1-4, doi: 10.1109/ICSEC.2018.8712636.

[3] Luca Parisi, Renfei Ma, Narrendar RaviChandran, Matteo Lanzillotta, hyper-sinh: An accurate and reliable function from shallow to deep learning in TensorFlow and Keras, Machine Learning with Applications, Volume 6, 2021, ISSN 2666-8270,

[4] P. M. S. Sánchez, J. M. J. Valero, A. H. Celdrán, G. Bovet, M. G. Pérez and G. M. Pérez, "A Survey on Device Behaviour Fingerprinting: Data Sources, Techniques, Application Scenarios, and Datasets," in IEEE Communications Surveys & Tutorials, vol. 23, no. 2, pp. 1048-1077, Second quarter 2021, https://doi: 10.1109/COMST.2021.3064259.

[5] T. Das, V. Sridharan and M. Gurusamy, "A Survey on Controller Placement in SDN," in IEEE Communications Surveys & Tutorials, vol. 22, no. 1, pp. 472-503, Firstquarter 2020, doi: 10.1109/COMST.2019.2935453.

[6] Muchammad Naseer et al 2021 J. Phys.: Conf. Ser. 1807 012011.

[7] Souri, A., Hosseini, R. A state-of-the-art survey of malware detection approaches using data mining techniques. *Hum. Cent. Comput. Inf. Sci.* **8**, 3 (2018). https://doi.org/10.1186/s13673-018-0125-x

[8] S. Jamalpur, Y. S. Navya, P. Raja, G. Tagore and G. R. K. Rao, "Dynamic Malware Analysis Using Cuckoo Sandbox," 2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT), 2018, pp. 1056-1060, doi: 10.1109/ICICCT.2018.8473346.

[9] Rouka, Elpida & Birkinshaw, Celyn & Vassilakis, Vassilios. (2020). SDN-based Malware Detection and Mitigation: The Case of ExPetr Ransomware. 10.1109/ICIoT48696.2020.9089514.

[10] Maxat Akbanov, Vassilios G. Vassilakis, Michael D. Logothetis, Ransomware detection and mitigation using software-defined networking: The case of WannaCry, Computers & Electrical Engineering, Volume 76, 2019, Pages 111-121, ISSN 0045-7906, https://doi.org/10.1016/j.compeleceng.2019.03.012.

[11] Irshad A., Dutta M.K. (2021) Identification of Windows-Based Malware by Dynamic Analysis Using Machine Learning Algorithm. In: Gao XZ., Tiwari S., Trivedi M., Mishra K. (eds) Advances in Computational Intelligence and Communication Technology. Advances in Intelligent Systems and Computing, vol 1086. Springer, Singapore. https://doi.org/10.1007/978-981-15-1275-9_18

[12] N., Wang, X., Wang, W. *et al*. Fingerprinting Android malware families. *Front. Comput. Sci.* **13**, 637–646 (2019). https://doi.org/10.1007/s11704-017-6493-y

[13] Xiong, Y. Zhang, D. Niyato, R. Deng, P. Wang and L. Wang, "Deep Reinforcement Learning for Mobile 5G and Beyond: Fundamentals, Applications, and Challenges," in IEEE Vehicular Technology Magazine, vol. 14, no. 2, pp. 44-52.

[14] Cao, J.; Wang, D.; Qu, Z.; Sun, H.; Li, B.; Chen, C.-L. An Improved Network Traffic Classification Model Based on a Support Vector Machine. *Symmetry* 2020, *12*, 301. https://doi.org/10.3390/sym12020301.

[15] Alzaharani, A.O.; Alazani, M.J.F. Designing a Network Intrusion Detection System Based on Machine Learning for Software Defined Networks. *Future Internet* 2021, *13*, 111. https://doi.org/10.3390/fi13050111

[16] Benjamin Lindeman, Benjamin Maschler, Nada Sahlab, Michael Weyrich, A survey on anomaly detection for technical systems using LSTM networks, Computers in Industry, Volume 131, 2021, ISSN 0166-3615, https://doi.org/10.1016/j.compind.2021.103498.

[17] Chu-Heng Lee, Shang-Juh Kao, Fu-Min Chang, LSTM-based ACB scheme for machine type communications in LTE-A networks, Computer Communications, Volume 152, 2020, Pages 296-304, ISSN 0140-3664, https://doi.org/10.1016/j.comcom.2020.01.047.

[18] Thapa, K.N.K., Duraipandian, N. Malicious Traffic Classification Using Long Short-Term Memory (LSTM) Model. *WirelessPers Commun* 119, 2707–2724(2021). https://doi.org/10.1007/s11277-021-08359-6

[19] McDole, A., Abdelsalam, M., Gupta, M., Mittal, S. (2020). Analyzing CNN Based Behavioural Malware Detection Techniques on Cloud IaaS. In: Zhang, Q., Wang, Y., Zhang, LJ. (eds) Cloud Computing – CLOUD 2020. CLOUD 2020. Lecture Notes in Computer Science(), vol 12403. Springer, Cham. https://doi.org/10.1007/978-3-030-59635-4_5

[20] W. -J. Eom, Y. -J. Song, C. -H. Park, J. -K. Kim, G. -H. Kim and Y. -Z. Cho, "Network Traffic Classification Using Ensemble Learning in Software-Defined Networks," 2021 International Conference on Artificial Intelligence in Information and Communication (ICAIIC), 2021, pp. 089-092,

[21] Hatma Suryotrisongko and Yasuo Musashi, “Hybrid Quantum Deep Learning and Variational Quantum

Classifier-BasedModel for Botnet DGA Attack Detection" in International Journal of Intelligent Engineering and Systems, Vol.15, No.3, 2022

[22] J. Xie *et al.,* "A Survey of Machine Learning Techniques Applied to Software Defined Networking(SDN): Research Issues and Challenges," IEEE Communications & Tutorials, Vol 21, n0 1, pp. 393-430, Firstquarter 2019.

[23] Mishra, A., Gupta, N. & Gupta, B.B. Defence mechanisms against DDoS attack based on entropy in SDN-cloud using POX controller. *Telecommun Syst* **77,**47–62(2021). https://doi.org/10.1007/s11235-020-00747-w

[24] Wang, Tao & Li, Jingcong. (2019). An improved support vector machine and its application in P2P lending personal credit scoring. IOP Conference Series: Materials Science and Engineering. 490. 062041. 10.1088/1757-899X/490/6/062041.

[25] Zhang, Zhaoqi & Qi, Panpan & Wang, Wei. (2020). Dynamic Malware Analysis with Feature Engineering and Feature Learning. Proceedings of the AAAI Conference on Artificial Intelligence. 34. 1210-1217. 10.1609/aaai.v34i01.5474.

[26] Eduardo de O. Andrade a , Jos´e Viterbo a , Cristina N. Vasconcelos a , Joris Gu´erin a , Flavia Cristina Bernardinia, "A Model Based on LSTM Neural Networks to Identify Five Different types of Malwares", Universidade Federal Fluminense, Gal. Milton Tavares de Souza Av., Niter´oi-RJ 24210-346, Brazi Procedia Computer Science pp-182-191(2022)

[27] Omondi, P., Rosenberg, D., Almeida, G., Soo-min, K., & Kato, Y. A Comparative Analysis of Deep Learning Models for Image Classification. Kuwait Journal of Machine Learning, 1(3). Retrieved from http://kuwaitjournals.com/index.php/kjml/article/view/128

[28] Indira, D., Alekhya, Y. S. ., Kishore, V. S. ., Ram, M. S. ., Namratha, S. ., & Kishore, B. N. . (2023). Color Image Encryption using Chaotic Algorithm and 2D Sin-Cos Henon Map for High Security . International Journal on Recent and Innovation Trends in Computing and Communication, 11(3), 263–272. https://doi.org/10.17762/ijritcc.v11i3.6346