Performance Analysis of Deep Learning Algorithms for Malware Detection

Sanjana Mahapatra [1], Avantika Hatmode [1], Vaishnavi Ratan [1], Radhika Edlabadkar [1], Latesh Bhagat [2]

[1]*Student, Computer Science & Engg. Department, Government College of Engineering, Nagpur, India.*

[2]*Assistant Professor, Computer Science & Engg. Department, Government College of Engineering, Nagpur, India.*

Corresponding Author: sanjanamahapatra46206@gmail.com

Abstract: - Malware attack is one of the most critical issues that electronic device users are facing in recent years. New variants of these malwares continue to grow and the traditional approaches like signature based static analysis, dynamic analysis are not suitable for detecting them. So, keeping this in mind, researchers are taking help of Machine Learning and Deep Learning approaches. These approaches have shown a great level of accuracy for determining existing as well as new malwares. In this study, our aim was to analyse the performance of different yet existing algorithms and do the comparison to find the best algorithm amongst them. We choose CNN and LSTM algorithm, and in the end, we combined them to check whether the accuracy is increased or not. After the groundwork, we got 86.50% accuracy for CNN on large dataset. LSTM showed a slightly high accuracy of 89.7% and when these two were combined (CNN+LSTM), the accuracy which we got was 92.01%.

Key Words: — ***Malware Detection, Deep Learning, CNN, LSTM.***

I. INTRODUCTION

Malware is any software intentionally designed to cause damage to a computer, server, client, or computer network [1]. Cyber Attackers develop the code to cause severe damage to the data, steal data, spying on someone or even gain unauthorized access. The coronavirus pandemic drove our life online and due to this, the number of malware attacks is increasing day by day. Cybersecurity experts are doing their best to make cyberspace safe. But malicious hacker groups develop revolutionary equivocate malware techniques such as code obfuscation, code encryption, etc., that outmatch many traditional malware diminution systems.

The traditional techniques for malware analysis are static analysis and dynamic analysis. In static analysis, the malware files are disassembled without actual execution.

Manuscript revised April 28, 2022; accepted April 29, 2022.
Date of publication May 01, 2022.

This paper available online at www.ijprse.com

ISSN (Online): 2582-7898; SJIF: 5.59

The features such as opcodes, strings, system calls, etc are extracted. Prerequisites of static analysis for disassembling PE files are knowledge about Hex codes, assembly codes and a deep understanding of malware operations. Furthermore, it also requires memory let alone time. In dynamic analysis, the malware codes are executed at run time in a virtual environment so that they can't harm our system. But this requires good expertise in the domain and a controlled environment for execution. The features such as network activity, system calls, file operations, registry modifications, etc., are extracted by this analysis [2]. The disadvantage of the static detection method is that it fails in identifying the new malware as the signatures of those new variants are not stored in the database beforehand. Dynamic analysis is better than the static method but it is time-consuming and sometimes it gives a false prediction.

Whatever we have seen so far, we have understood that traditional approaches are lacking behind somewhere. That is why one needs to come up with the solution which overcomes these loopholes. Machine Learning and Deep Learning is one of the solutions for removing these loopholes. Algorithms like SVM, Random Forest, Decision tree, CNN, etc. has already proven to be one of the best solutions in terms of accuracy.

Our research is to find best out of these best algorithms. So, for that matter we have chosen some of the Deep Learning algorithms for our research. And those include CNN and

LSTM, CNN+LSTM. We will compare the accuracy of these algorithms and will try to draw a conclusion about which algorithm has performed well. The reason for choosing Deep Learning algorithms is twofold. It learns the features given and also automatically extracts features from the data at runtime along with the good accuracy.

In our study, we first implemented CNN. The data which we were using was PE files. The very first step in the implementation was to convert the PE binary into grayscale images. The idea was taken from Natraj et al. A given malware binary is read as a vector of 8-bit unsigned integers (uint) and then organized into a 2D array. This can be visualized as a grayscale image in the range [0,255] [3]. And the images that are similar can be categorized and then classified as malware or benign. The intuition was most of the hackers change a small part of existing malicious code and create a new malware. Due to which traditional methods were failing. Here, images of the same family will be very similar making it easy for CNN to classify it as a malware.

Sudan Jha et al has done work on RNN based classification [4]. But, in order to increase the accuracy LSTM will be preferred by us. Long Short-Term Memory (LSTM) is a type of recurrent neural network that can learn the order dependence between items in a sequence [5]. The order of the operating system API calls decides the behaviour of the malware. And that is why LSTM can be considered as one of the solutions for such time-sequential data.

II. Related Work

Till now there exists different solutions for the malware detection problem. Signature based detection (static analysis), behavioural pattern-based detection (dynamic analysis) [14, 15,16], these traditional techniques can't stand alone as there are many loopholes in it.

P. V. Shijo et al., has implemented Machine Learning algorithms using static and dynamic analysis methods. In static analysis, the PSI (printable string information) is used as feature and feature list is created according to frequency of PSI. The malware and benign samples are compared with this list. In dynamic analysis, the samples are run on virtual environment. N-grams are created for API calls and feature vector is created which is used for checking malware and benign files.[6]

Ajit Kumar et al., on the other hand, followed Machine Learning algorithm for Malware detection. In their research, they used integrated feature set for determining the PE malware. An integrated feature set was nothing but a combination of PE header raw values and derived values. The algorithms like Decision Tree, KNN, Logistic regression, Random Forest, naïve Bayes, etc were used for the classification purpose. 98.4% accuracy was noted for the integrated set whereas for novel test dataset it was noted 89.23%. And when the same dataset was tested against raw features, it gave approximately 75% accuracy. [7]

Nataraj et al., proposed a very efficient method of classifying malwares using image processing, Malware binaries are converted into grayscale image and that image is fed to CNN model as an input. The model has achieved an accuracy of 98%. This is very effective method for visualization of malware as well for their classification. [3]

M. Santacroce et al., worked on deep CNN model for classifying machine code as malware or benign. The accuracy obtained by them on a small dataset was 95.1% and the model with minor modification achieved 88% accuracy in classifying 9 types of malwares on larger dataset. [8]

In the research paper of Renjie Lu, referring the idea of natural language processing, he proposed a novel approach to perform static analysis of malware. The model learnt opcode sequence pattern automatically. This opcode sequence was obtained by using IDA pro disassembler. And the techniques like word embedding, etc were used for learning feature vector representation of opcode. They proposed a two-stage LSTM model in which the first step was 2 LSTM layers and the next was one mean-pooling layer. His proposed model achieved average AUC of 0.99 and average AUC of 0.987 in the best case. [9]

Jinbo Zhang, in his work, proposed a novel approach by combining the architecture of CNN and RNN to classify malwares. He joined two different units to capture the important correlation and learn from long-term sequential dependencies. The training accuracy for this model was increased and was approximately around 1 and testing accuracy was very close to that of training accuracy. [10]

III. Proposed Methodology

3.1 Convolutional Neural Network

CNN is a type of neural network which is used to analyse images and classify them. So instead of analysing the complete code of malware, one can convert malware binary into images and use CNN to classify them. This method does not require much pre-processing on data. Also, it is capable of capturing spatial and temporal dependencies. The method gives a good accuracy and a better fitting of dataset as reduction of parameters (without losing features) can be done. This

reduction is done with the help of convolutional layers (kernel) and pooling layers.

The data, which we had, was in the form of malware binary. We got this data from the Microsoft Malware Classification challenge [11], which we felt as a trustworthy source. It contained .asm and. bytes files. The bytes data was converted into grayscale images using Pillow library and other tools (One may directly use the converted malware files from the available datasets like Malimg dataset). Those images were fed to CNN model for classification.

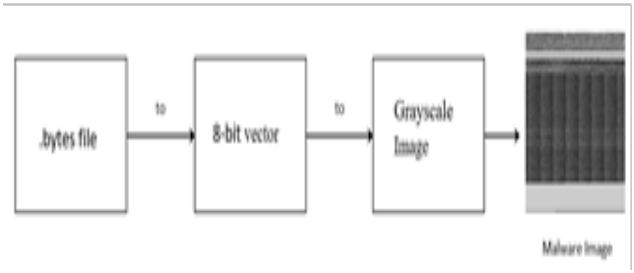

Fig.1. Conversion of bytes file to Grayscale image

3.2 Architecture of CNN

The images which we generated were of dimensions 2048 X 586 (height was variable). Even this is in the range of 10^5 neurons for each input layer, which is of course not feasible for the computation. So, these images were reduced to the size of 32 X 32 X 1, with the help of CNN layers and it was flattened to 1 X 1 X 1024. These layers extracted the important features without actually losing characteristics of an image. The first layer used was convolutional layer. This layer extracts features from the image. Layers of convolutional and layers of max pooling was applied (Convolutional - Max pooling – Convolutional - Max pooling). The primary aim of max-pooling layer is to decrease the size of the image, which we got as an output from the previous convolutional layer. Then, the reduced image was flattened using flatten layer. And then Dense layer with Relu activation function, a Dropout layer and, again a Dense layer with Relu function were used. In the end, SoftMax activation function was applied. Activation function decides which information is important and which is not. Accordingly, it propagates the useful information forward. This CNN model came across the problem of overfitting, so as a solution for that we used transfer learning technique [12]. ResNet50 model which has 48 convolutional layers, 1 Max Pool and 1 average pool layer were applied.

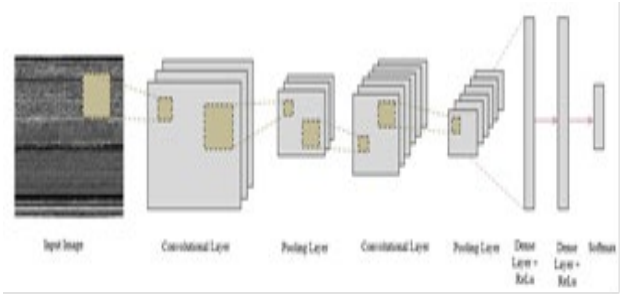

Fig.2. Basic architecture of CNN

3.3 Long-Short Term Memory (LSTM)

LSTM is a feedback connection network. It is used for classifying, processing, and makingpredictionsn based on certaitimesme. It overcomes the vanishing gradient problem of RNN. The vanishing gradient strengths as it back propagates through time A gradient value becomes extremely smaland l, it doesn't conttoobute to much learning. So RNN layer that gets a small gradient update doesn't learn. LSTM has an internal mechanism called gates that can regulate the flow of information. These gates can learn which data is important to keep or throw away, by doing that it learns to use relevant information to maa predictionsion. In RNN, first words getransferredrs to machine-readable vectors then processed the sequence one by one. LSTM has the same control flow as RNN, it processes data sequentially passing on information as it propagates forward. The differeis are operations within cells.

The core idea of LSTM is cell states and their various gates. The cell acts as a transport highway that transfers relative information all way down to the sequence chain. The gate keeps relevant information, otherwise forgets it. Gates contain sigmoid function. LSTM contains forget gate, input gate and output gate. Forget gate decides what information should be thrown or kept away from previous hidden state and information from current state. Input is passed through sigmoid function. And values come out between 0 and 1. Closer to 0 means forget and closer to 1 means to keep. Input state updates the cell state. We passed previous head and current input to sigmoid function that decides which values to be updated by transforming values between 0 and 1. Then we multiply tan output with sigmoid output. Then the forget cell has possibility of dropping values. After this, the output from input gate is taken and polarize addition is done which updates all state to new values. This gives us new cell state. Last, we have output gate. This applicate decides what next hidden state should be remembering. The hidden state also used for prediction. The previous hidden state is passed to the sigmoid function which is a current input function. After that, we pass modified cell

state to tan function. We multiply tan output with sigmoid to decide what information hidden state should carry. The new cell state and hidden state is carried out as input for adjacent LSTM cell. This is the basic working of LSTM.

The assembly code which we had was then broken down into tokens. The opcodes are considered as important feature in many studies. Given a character sequence and a defined document unit, tokenization is the task of chopping it up into pieces, called tokens, perhaps at the same time throwing away certain characters, such as punctuation [13]. The commas and semicolons are discarded. After tokenization, vectorization is done. The input is fed to LSTM model and the output is predicted.

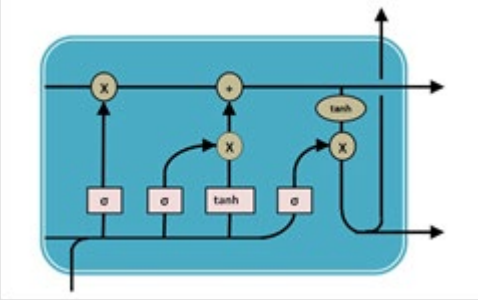

Fig.3. Internal Mechanism of LSTM

3.4 CNN+LSTM

CNN and LSTM individually has shown a satisfactory performance, but CNN and LSTM together do wonders (in terms of accuracy) with the help of Ensemble blending technique.

3.4.1 *Blending Ensemble Technique*:

Blending is an ensemble technique in which the best predictions from different ensemble member models are combined to determine the final output. The blending ensemble finally uses a machine learning model to learn how to combine and make the predictions.

It has first level of base models. This level is also referred as Level-0 models. It may have two or more base models. The second level is Level-1 model or Meta-Model. The model in this level is trained on the basis of the predictions that the base models have made. For the Meta-Model, generally, linear models such as linear regression or logistic regression are used. Blending ensemble is an informal name of stacking-type architecture. The difference between blending and stacking technique is the meta-model is trained using predictions on 'holdout' validation set in blending whereas training is done on out-of-fold predictions.

Process: *The process has two steps. In the first step, CNN and LSTM will have the input in their respective format, where CNN will have Grayscale image and LSTM will have the opcode sequence. In the second step, the predictions made by CNN and LSTM will be used by Meta-Model, Logistic Regression to get trained and final output will be produced.*

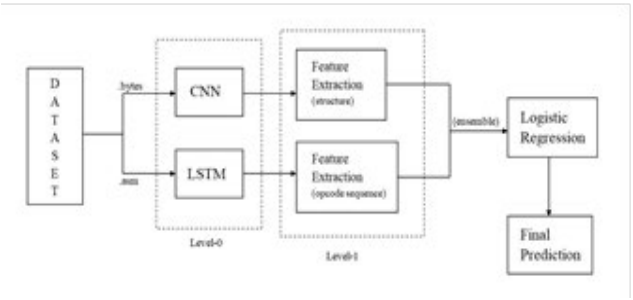

Fig.4. CNN+LSTM architecture

3.4.2 Accuracy

We proposed three algorithms for malware detection: Convolutional Neural Network (CNN), Long-Short Term Memory (LSTM) and CNN+LSTM. The CNN model consisting of 2 convolutional layers, 2 max pooling layers along with ResNet50 gave us 86.5% accuracy. As CNN suffers from vanishing gradient problem, we used LSTM. LSTM is able to handle the vanishing gradient problem and gives better results than CNN on the same data. With LSTM, we got accuracy of about 89.73%. Combined the advantage of both CNN and LSTM in our third algorithm which is CNN+LSTM we achieved 92.01% accuracy which is greatest among all.

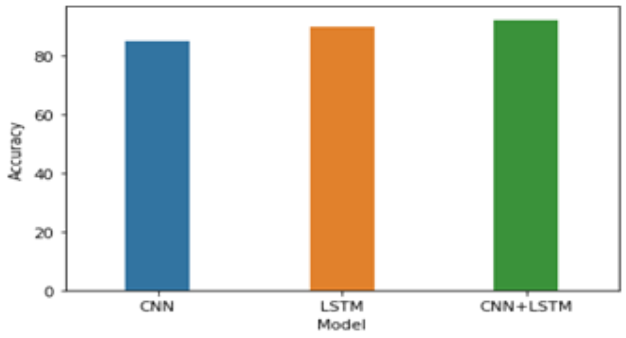

Fig.5. Bar graph showing accuracy of algorithms

IV. CONCLUSION

In this study, we revisited the existing algorithms to check their accuracy and chose the best algorithm. There are plenty of researches based on Deep Learning approach. We wanted to search for the algorithm which shows the highest accuracy. For that matter we choose CNN and LSTM for our

study. And as per our expectations, CNN+LSTM has shown a remarkable accuracy. CNN and LSTM individually also did a great job. But, CNN+LSTM has shown that we should come up with a novel approach by stacking 2 or more algorithms together or combining them in such a way that they result in increased accuracy.

REFERENCES

[1]. Hemalatha, J.; Roseline, S.A.; Geetha, S.; Kadry, S.; Damaševi˘cius, R. An Efficient DenseNet-Based Deep Learning Model for Malware Detection. Entropy 2021, 23, 344.

[2]. Nataraj, L., Karthikeyan, S., Jacob, G., and Manjunath, B. (2011b). Malware Images: Visualization and Automatic Classification. In Proceedings of International Symposium on Visualization for Cyber Security.

[3]. Prashar, Deepak & Jha, Sudan. (2020). Recurrent Neural Network for Detecting Malware. Computers & Security. 99.

[4]. P. Shijo, A. Salim, Integrated static and dynamic analysis for malware detection. Procedia Computer Science 46, 804–811 (2015) Integrated Static and Dynamic Analysis for Malware Detection – ScienceDirect.

[5]. Ajit Kumar, K.S. Kuppusamy, G. Aghila, A learning model to detect maliciousness of portable executable using integrated feature set, Journal of King Saud University - Computer and Information Sciences, Volume 31, Issue 2, 2019, Pages 252-265.

[6]. M. Santacroce, Daniel Koranek, David Kapp, Anca Ralescu, and R. Jha, "Detecting Malicious Assembly with Deep Learning", 2008.

[7]. Lu, Renjie. (2019). Malware Detection with LSTM using Opcode Language.

[8]. J. Zhang, "DeepMal: A CNN-LSTM Model for Malware Detection Based on Dynamic Semantic Behaviours," 2020 International Conference on Computer Information and Big Data Applications (CIBDA), 2020, pp. 313-316.

[9]. Moser, Andreas & Kruegel, Christopher & Kirda, Engin. (2007). Limits of Static Analysis for Malware Detection.

[10]. Sihwail, Rami & Omar, Khairuddin & Zainol Ariffin, Khairul Akram. (2018). A Survey on Malware Analysis Techniques: Static, Dynamic, Hybrid and Memory Analysis. 8. 1662.

[11]. Damodaran, Anusha & Di Troia, Fabio & Visaggio, Corrado Aaron & Austin, Thomas & Stamp, Mark. (2017). A comparison of static, dynamic, and hybrid analysis for malware detection. Journal of Computer Virology and Hacking Techniques.

[12]. R. Vyas, X. Luo, N. McFarland and C. Justice, "Investigation of malicious portable executable file detection on the network using supervised learning techniques," 2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM), 2017, pp. 941-946.

[13]. Ki, Youngjoon & Kim, Eunjin & Kim, Huy Kang. (2015). A Novel Approach to Detect Malware Based on API Call Sequence Analysis. International Journal of Distributed Sensor Networks. 2015. 1-9.

[14]. LU Xiaofeng, JIANG Fangshuo, ZHOU Xiao, et al. API based sequence and statistical features in a combined malware detection architecture[J]. Journal of Tsinghua University (Science and Technology), 2018, 58(5): 500-508.

[15]. B. Athiwaratkun and J. W. Stokes, "Malware classification with LSTM and GRU language models and a character-level CNN," 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2017, pp. 2482-2486.