INTERNATIONAL JOURNAL OF ADVANCE RESEARCH, IDEAS AND INNOVATIONS IN TECHNOLOGY

ISSN: 2454-132X
Impact Factor: 6.078
(Volume 9, Issue 5 – V9I5-1187)
Available online at: https://www.ijariit.com

Hybrid deep approach for malware detection

Vadduri Uday Kiran
udaykiranjune2@gmail.com
Vasireddy Venkatadri Institute of Technology, Guntur, Andhra Pradesh

P. Shiva Prasad Reddy
shivaprasad250128@gmail.com
Vasireddy Venkatadri Institute of Technology, Guntur, Andhra Pradesh

V. Sri Harsha
harshavelaga56@gmail.com
Vasireddy Venkatadri Institute of Technology, Guntur, Andhra Pradesh

R. Vijay Kumar
vijay810srini@gmail.com
Vasireddy Venkatadri Institute of Technology, Guntur, Andhra Pradesh

Y. Venkata Narayana
naarayanaa808@gmail.com
Vasireddy Venkatadri Institute of Technology, Guntur, Andhra Pradesh

ABSTRACT

Malware is a malicious software designed to compromise computer systems, poses a significant threat to businesses, with potential repercussions ranging from financial losses to damaged reputations and eroded customer trust. To address this challenge, we propose a hybrid deep learning approach that combines the power of Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRUs), both of which are models in the Recurrent Neural Network (RNN) family. Our research focuses on assessing the potential improvements achieved by this hybrid approach, leveraging a benchmark dataset known as NSL-KDD+. This dataset offers a temporal dimension and encompasses a diverse array of malware samples and network traffic scenarios for comprehensive testing and evaluation. We employ a range of performance metrics, including Accuracy, Precision, F1 Score, Mean Absolute Error (MAE), and others, to comprehensively gauge the effectiveness of our proposed approach.

Keywords: *Intrusion, Malware Detection, LSTM-GRU, and RNN neural network, Deep Learning RNN Intrusion Detection.*

1. INTRODUCTION

In the ever-evolving landscape of technology, the threat of malware remains a persistent and formidable challenge for businesses and organizations. Malicious software, often designed with the intent to compromise computer systems, not only has the potential to inflict significant financial losses but also to tarnish the reputation and erode the trust that customers place in these entities. As cyber threats continue to grow in sophistication, it is imperative for security experts to stay ahead of the curve by developing innovative and effective strategies for detecting and mitigating malware.

To confront this daunting challenge, we propose a novel approach that harnesses the capabilities of Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRUs), both belonging to the Recurrent Neural Network (RNN) family. This hybrid deep learning approach presents a promising avenue to enhance our ability to detect and combat malicious software effectively.

© 2021, www.IJARIIT.com All Rights Reserved

LSTMs are renowned for their proficiency in capturing long-range dependencies and temporal patterns within binary executables. This makes them exceptionally well-suited for unveiling intricate patterns concealed within the complex and dynamic nature of malware code. On the other hand, GRUs specialize in the learning of compact representations and denoising of data, which renders them adept at discerning subtle yet critical indicators of malicious behavior. The combination of these two RNN models can provide a holistic solution for improving the accuracy and efficiency of malware detection.

Our research endeavors to assess the potential enhancements that can be achieved through the deployment of this hybrid deep learning approach. To do this, we employ a benchmark dataset known as NSL-KDD+, which offers a unique temporal dimension and comprises a diverse range of malware samples and network traffic scenarios for comprehensive testing and evaluation. This dataset serves as a robust foundation for our analysis, allowing us to replicate real-world conditions and challenges faced by businesses in the realm of cyber security.[1]
Depending on what it does and how it behaves, malware goes by many names, including ransom ware, adware, spyware, virus, worm, Trojan, root kit, backdoor, and command and control (C&C) bot. Malware detection and mitigation is a developing issue in the world of cyber security.[2]

Software created with malevolent purpose is known as malware, and it has grown to be a major global cyber hazard. Malware may be found in a variety of ways. In general, malware detection techniques that rely on signatures are commonly employed. Using a signature gathered from previously discovered malware, it finds the infection. However, this approach has the drawback of making it extremely difficult to identify hidden or altered malware.[3]

2. NEED OF MALWARE DETECTION

Malware detection is crucial for various reasons, as it helps protect computer systems and networks from a wide range of malicious software threats. Here are four key points highlighting the need for effective malware detection.

Security Protection: Malware, such as viruses, Trojans, ransom-ware, and spyware, can compromise the security of computer systems and data. Malware detection tools help identify and remove these threats, preventing unauthorized access and data breaches.

Data Integrity: Malware can corrupt or delete files, causing data loss and system instability. Detecting malware early can help safeguard the integrity of important data and prevent costly recovery efforts.

System Performance: Malware can consume system resources and slow down computer performance, leading to a frustrating user experience. Effective malware detection helps maintain optimal system performance and responsiveness.

Network Security: In the case of networked environments, malware can spread quickly, infecting multiple devices and compromising the network's integrity. Malware detection tools can identify and contain infections to prevent the further spread of malware within the network.

3. TYPES OF MALWARE ATTACKS

Malware attacks come in various forms and are designed to compromise the security and functionality of computer systems and networks. These are the many kinds of assaults.

Viruses: Viruses are malicious programs that attach themselves to legitimate files and replicate when the infected file is executed. They can spread through infected files and can cause a wide range of issues, including data loss and system instability.

Worms: Worms are self-replicating malware that do not need to attach to a host file. They spread independently through network vulnerabilities, email attachments, or removable media. Worms can propagate rapidly, causing widespread damage.

Trojans (Trojan Horses): Trojans disguise themselves as legitimate software to trick users into installing them. Once installed, they can create backdoors, steal data, or perform other malicious actions without the user's knowledge.

Ransomware: Ransomware encrypts the victim's files and demands a ransom in exchange for the decryption key. It has been a significant threat to individuals and organizations, often causing data loss and financial damage.

Spyware: Spyware is designed to spy on users' activities without their knowledge or consent. It can track keystrokes, capture screenshots, record web browsing habits, and send this information to malicious actors.

Adware: Adware displays unwanted and intrusive advertisements on a user's device. While not always malicious, adware can slow down system performance and compromise user privacy.

Rootkits: Rootkits are a form of malware that hides within a system or network and provides unauthorized access to malicious actors. They can be challenging to detect and remove because they often operate at a deep level of the operating system.

© 2021, www.IJARIIT.com All Rights Reserved

Botnets: A botnet is a network of compromised computers or devices controlled by a single entity (botmaster). These compromised devices can be used for various malicious purposes, such as launching distributed denial-of-service (DDoS) attacks or distributing spam.

4. METHODOLOGY

The methodology used in the proposed research, which aims to enhance malware detection through a hybrid deep learning approach combining Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRUs), typically follows the steps commonly found in machine learning and deep learning research. Below, I outline the general methodology that might be employed. The below information is showed in fig-1.

Data Collection

Gather a suitable dataset for the research. In this case, the NSL-KDD+ dataset is chosen, which contains a diverse array of malware samples and network traffic scenarios.

Data Preprocessing

Clean and preprocess the dataset. This may include data normalization, handling missing values, and encoding categorical variables. Data splitting into training, validation, and testing sets is also done.

Feature Engineering

Extract relevant features from the dataset that are essential for malware detection. In the context of deep learning, these features are often transformed into a suitable format for the neural network model.

Model Selection

In this case, LSTM and GRU networks are selected, both of which are part of the Recurrent Neural Network (RNN) family. This choice is based on their suitability for handling sequential data.

Model Design and Training

Design the architecture of the hybrid deep learning model, which combines LSTM and GRU layers. Specify hyper parameters, such as the number of layers, units, learning rate, and dropout rates. Train the model on the training data, monitoring its performance on the validation set. Gets appropriate loss functions and optimization techniques.

Evaluation

Assess the model's performance using various evaluation metrics, such as accuracy, precision, recall, F1 score, and AUC-ROC. The research might also employ confusion matrices to visualize the results.

Comparison

Compare the performance of the hybrid deep learning approach with existing methods or models. This step aims to demonstrate the potential improvements achieved by the proposed approach.

5. PROPOSED ARCHITECTURE OF MALWARE DETECTION SYSTEM:

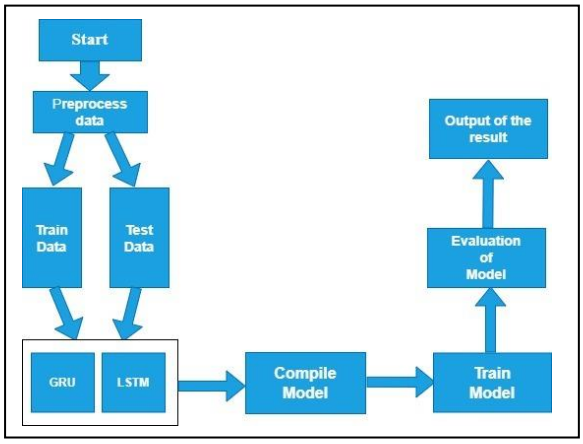

Fig-1: Architecture of Proposed System

6. ADVANTAGES OF THE SYSTEM

Enhanced Generalization

By combining the strengths of both GRU and LSTM models into a single, scalable hybrid unit, the proposed system gains the advantage of enhanced generalization. This means that the model becomes more adept at recognizing various types of malware attacks and can effectively adapt to previously unseen or unknown threats. Enhancing its practicality and adaptability.

Improved Handling of Complex Binary Data

© 2021, www.IJARIIT.com All Rights Reserved

Binary data, such as the code within malware executables, is inherently complex and non-linear in nature. Traditional machine learning models, particularly those designed for structured or tabular data, often struggle to effectively process and extract meaningful insights from such intricate data.

7. OBJECTIVES OF SYSTEM

Enhanced Malware Detection Accuracy
The primary objective of the proposed system is to significantly improve the accuracy of malware detection. By leveraging the strengths of LSTM and GRU models, the system aims to correctly identify and classify malware instances while minimizing false positives and false negatives. Enhanced accuracy is essential for safeguarding computer systems and networks from malicious threats.

Increased Generalization to Diverse Malware
The goal of the system is to successfully generalize to various malware attack methods and unknown threats. It seeks to create a model that is adaptable and durable in the face of changing cyber security threats by combining elements from both the LSTM and GRU models. This allows the model to adapt to a wide range of harmful behaviors.

Scalability and Efficiency
The system is designed to be scalable and efficient, capable of handling a large volume of data and network traffic. It aims to provide an effective and resource-efficient solution that can be applied to diverse network environments, from small businesses to large enterprises.

8. ALGORITHM OF PROPOSED SYSTEM

Input: Intrusion detection dataset
Output: Trained hybrid GRU-LSTM model and evaluation metrics
Step-1: Preprocess the dataset.
Step-2: Divide the data into sets for testing and training.
Step-3: Define the hybrid model architecture.
Step-4: Compile the model.
Step-5: Train the model.
Step-6: Evaluate the model on the test data.
Step-7: Output the trained model and evaluation results.

9. MODEL AND RESULT

Defined Model

```
# Define the model
# A neural network model is defined using Keras Sequential API.
#The model consists of GRU, LSTM, and Dense layers.

model = Sequential()
model.add(GRU(64, return_sequences=True, input_shape=(1, X_train.shape[2])))
model.add(Dropout(0.2))
model.add(LSTM(64, return_sequences=True))
model.add(Dropout(0.2))
model.add(GRU(64, return_sequences=True))
model.add(LSTM(64, return_sequences=True))
model.add(Dropout(0.2))
model.add(GRU(64, return_sequences=True))
model.add(Flatten())
model.add(Dense(units=50))
model.add(Dense(units=5, activation='softmax'))
```

Fig-2: Model Definition

Model Training

```
#Model Training:
#The model is trained on the training data (X_train and y_train) using model.fit() for 20 epochs with a batch size of 32.

history = model.fit(X_train, y_train, epochs=10, batch_size=32,validation_split=0.2)
Epoch 1/10
2971/2971 [==============================] - 52s 13ms/step - loss: 0.1222 - accuracy: 0.9604 - val_loss: 0.0565 - val_accuracy: 0.9825
Epoch 2/10
2971/2971 [==============================] - 37s 13ms/step - loss: 0.0617 - accuracy: 0.9802 - val_loss: 0.0509 - val_accuracy: 0.9843
Epoch 3/10
2971/2971 [==============================] - 54s 18ms/step - loss: 0.0526 - accuracy: 0.9826 - val_loss: 0.0429 - val_accuracy: 0.9865
Epoch 4/10
2971/2971 [==============================] - 37s 12ms/step - loss: 0.0469 - accuracy: 0.9847 - val_loss: 0.0431 - val_accuracy: 0.9862
Epoch 5/10
2971/2971 [==============================] - 38s 13ms/step - loss: 0.0430 - accuracy: 0.9854 - val_loss: 0.0384 - val_accuracy: 0.9877
Epoch 6/10
2971/2971 [==============================] - 36s 12ms/step - loss: 0.0411 - accuracy: 0.9862 - val_loss: 0.0384 - val_accuracy: 0.9875
Epoch 7/10
2971/2971 [==============================] - 35s 12ms/step - loss: 0.0381 - accuracy: 0.9872 - val_loss: 0.0359 - val_accuracy: 0.9895
Epoch 8/10
2971/2971 [==============================] - 35s 12ms/step - loss: 0.0376 - accuracy: 0.9877 - val_loss: 0.0386 - val_accuracy: 0.9868
Epoch 9/10
2971/2971 [==============================] - 36s 12ms/step - loss: 0.0357 - accuracy: 0.9879 - val_loss: 0.0361 - val_accuracy: 0.9878
Epoch 10/10
2971/2971 [==============================] - 35s 12ms/step - loss: 0.0342 - accuracy: 0.9885 - val_loss: 0.0333 - val_accuracy: 0.9888
```

© 2021, www.IJARIIT.com All Rights Reserved

Fig-3: Training the model

Model Evaluation

```
#Model Evaluation:
'''The trained model is evaluated on the test data (X_test and y_test), and
the test loss and accuracy are printed.'''

test_results = model.evaluate(X_test, y_test, verbose=1)
print(f'Test results - Loss: {test_results[0]} - Accuracy: {test_results[1]*100}%')
929/929 [==============================] - 3s 3ms/step - loss: 0.0334 - accuracy: 0.9884
Test results - Loss: 0.033381473273038864 - Accuracy: 98.83853793144226%
```

Fig-4: Evaluating the model

Result

```
from sklearn.metrics import f1_score, precision_score, confusion_matrix,recall_score
# Make predictions on the test data
y_pred = model.predict(X_test)
y_pred = np.argmax(y_pred, axis=1)
# Convert one-hot encoded labels back to class indices for y_test
y_true = np.argmax(y_test, axis=1)
# Calculate F1-score,precision and Recall_score
f1 = f1_score(y_true, y_pred, average='weighted')
precision = precision_score(y_true, y_pred, average='weighted')
recall = recall_score(y_true, y_pred, average='weighted')
print(f'F1 Score: {f1 * 100}')
print(f'Precision: {precision * 100}')
print(f'Recall: {recall * 100}')

929/929 [==============================] - 6s 4ms/step
F1 Score: 98.83298325502162
Precision: 98.82940597284615
Recall: 98.83854026393752
```

Fig-5: F1-Score, Precision, Recall

Confusion Matrix

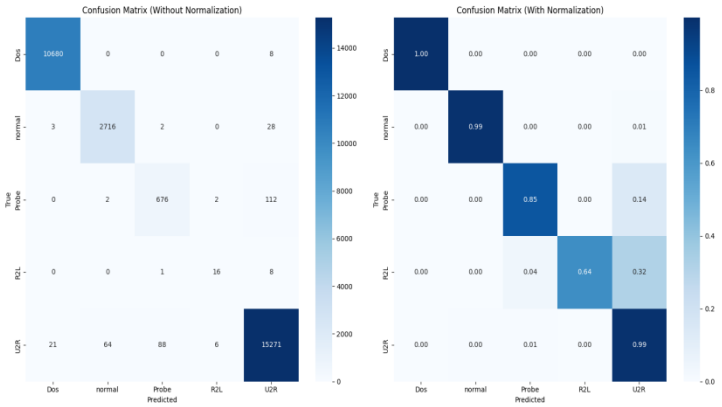

Fig-6: Confusion Matrix of a model

10. CONCLUSION

We have utilized a new method that combines the capabilities of Gated Recurrent Units (GRUs) and Long Short-Term Memory (LSTM) in a hybrid deep learning model in an effort to create an efficient malware detection system. Utilizing the NSL-KDD+ benchmark dataset, our study has concentrated on evaluating the effectiveness and possible benefits of this strategy. Finally, our work is a positive step toward the development of a reliable malware detection system. Even though we've made great strides, the model's accuracy and effectiveness in identifying different malware kinds will ultimately determine whether or not we were successful.

11. REFERENCES

[1] R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran, A. Al-Nemrat and S. Venkatraman, "Deep Learning Approach for Intelligent Intrusion Detection System," in *IEEE Access*, vol. 7, pp. 41525-41550, 2019, doi: 10.1109/ACCESS.2019.2895334

[2] Recurrent neural network for detecting malware":https://www.sciencedirect.com/science/article/pii/S0167404820303102

[3] M. Eian, “Fragility of the Robust Security Network: 80211,” Norwegian University of Science and Technology,2011.

[4] J. Xu, J. Wang, S. Xie, W. Chen and J. Kim, “Study on Intrusion Detection Policy for Wireless Sensor Networks”, International Journal of Security and Its Applications, vol.7, no. 1, (2013) January, pp. 1-6.

© 2021, www.IJARIIT.com All Rights Reserved

[5] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, “Wireless Sensor Networks: a Survey”, ComputerNetworks, vol. 38, no. 4, (2002), pp. 393-422.
[6] K. Martinez, J. Hart, and R. Ong, “Environmental Sensor Networks”, IEEE Computer, vol. 37, no. 8, (2004), pp. 50- 56.
[7] R. Abouhogail, “Security Assessment for Key Management in Mobile Ad Hoc Networks”, International Journal of Security and Its Applications, vol. 8, no. 1, (2014), pp. 169- 182,http://dx.doi.org/10.14257/ijsia.2014.8.1.16,.
[8] E. Ngai, J. Liu, and M. Lyu, “On the Intruder Detection for Sinkhole Attack in Wireless Sensor Networks”, IEEE International Conference on Communications, (2006).
[9] D. Martins and H. Guyennet, “Wireless Sensor Network Attacks and Security Mechanisms: A Short Survey”, 13th International Conference on Network-Based Information Systems, (2010).
[10] S.Ganapathy, P.Yogesh, and A.Kannan," An Intelligent Intrusion Detection System for Mobile Ad-Hoc Networks Using Classification Techniques," Advances in Power Electronics and Instrumentation Engineering, Communications in Computer and Information Science Vol.148, pp 117-122,2011.
[11] Yurcik W, "Contrlling intrusion detection systems by generating false positives:squealing proof-of-concept", in Proceeding of the IEEE local Computer Network Conference, 2002. pp.93--101.
[12] Yu Guan, Nabil Belacel and Ali A. Ghorbani,"Y-Means: A Clustering Method for Intrusion Detection," Canadian Conference on Electrical and Computer Engineering, vol.2, pp. 1083- 1086, 2003.
[13] Shafee, Ahmed & Baza, Mohamed & Talbert, Douglas & Fouda, Mostafa & Nabil, Mahmoud & Mahmoud, Mohamed. (2019). “Mimic Learning to Generate a Shareable Network Intrusion Detection Model”.
[14] M. Azizjon, A. Jumabek and W. Kim, “ 1D CNN based network intrusion detection with normalization on imbalanced data” , 2020 International Conference on Artificial Intelligence in Information and Communication (ICAIIC), Fukuoka, Japan, 2020, pp. 218-224, doi: 10.1109/ICAIIC48513.2020.9064976.
[15] J. Olamantanmi Mebawondu, Olufunso D. Alowolodu, Jacob O. Mebawondu, Adebayo O. Adetunmbi, “Network intrusion detection system using supervised learning paradigm”, Scientific African, Volume 9, 2020, e00497, ISSN2468-2276, https://doi.org/10.1016/j.sciaf.2020.e00497.

© 2021, www.IJARIIT.com All Rights Reserved