International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 18(19s) 2026 pp. 1096-1106
https://doi.org/10.70917/ijcisim-2026-5103
Article

A Hybrid RNN-LSTM Framework for Cyber Threat Detection Using Vulnerability Code Snippets

Akshay R. Jain[1], Dr. Atul Agrawal[2]

[1]Research Scholar, Department of Computer Science and Engineering, Oriental University, Indore (M.P.), India
Email Id- jainakshay781@gmail.com, Orcid ID- 0009-0005-6078-7698
[2]Supervisor, Department of Computer Science and Engineering, Oriental University, Indore (M.P.), India
Email Id- atul.273@gmail.com, Orcid ID- 0009-0006-7778-2364

Abstract: The exponential growth of cyber threats and software flaws, automated detection approaches that can recognize malicious code patterns in dynamic environments are required. Signature based and primitive machine learning theory inability to generalize when presented with new or unseen vulnerabilities is reflected in the limitations of conventional approach. In this work, we introduced a novel deep learning (DL)-based approach using hybrid model of Recurrent Neural Network (RNN) and Bidirectional Long Short-Term Memory (BiLSTM) for the automated vulnerability detection in source code. The model has been evaluated over the available publicly labeled Vulnerability Fix Dataset with 35000 pieces of code snippets. Data preprocessing such as duplicate elimination, tokenization, padding, feature engineering, and label encoding was conducted to ensure high-quality structured inputs. The architecture contains an embedding layer, a RNN baseline layer for learning initial sentence representation, stacked BiLSTM layers for bidirectional context representation and dropout regularization to prevent overfitting. Adam optimization was applied for training the model with categorical cross-entropy loss and label smoothing to provide better generalization. The results of our experiments showed strong performance with 95.31% classification accuracy, 95.0% precision, 94.12% recall and an F1-score of 95.25% were achieved. Receiver Operating Characteristic analysis shows an Area Under the Curve (AUC) of 0.95, suggestive of strong discrimination performance. The confusion matrix provides support by showing how the vulnerability classes such as SQL Injection and Path Traversal are correctly identified. The results indicate that the hybrid RNN–BiLSTM architecture can be used to efficiently perform proactive vulnerability detection. The model is also applicable to enable real-time cybersecurity monitoring and the use of automatically possible counter efforts in contemporary software systems.
Keywords: cyber threat detection; vulnerability classification; RNN-BiLSTM hybrid model; machine learning for security; software vulnerabilities; proactive cyber defense

1. Introduction

The computing environments of today are the subject to an ever-growing list of attacks, driven in large part by the burgeoning backbone of cyber-threats; digital technology. This makes cyber threats major concern for both businesses and individuals. Anomaly detection and intrusion detection in network are today indispensable for defending systems against malicious actions that harmful their privacy, integrity and availability [1]. Signature-based techniques are effective at blocking known threats, but they struggle with new attack surfaces and zero-day exploits. Due to this constraint, more and more efforts are devoted to data-driven and machine learning (ML) based approaches for detection of cyber threats before they occur [2].

Recently, some sophisticated cyber-attacks such as SQL Injection, Cross-Site Scripting (XSS), Path Traversal and so on abuse software code to intrude into the entering system in the past few years. Making realistic datasets for training intrusion detection systems (IDS) since attack representations should be balanced and varied [3]. From [4], scientists are developing hybrid deep learning that combines statistical pattern recognition and advanced feature

extraction to sense problems real-time. AI is what makes security systems for the future so special. Even with these changes, it remains difficult to form cyber defense models that are solid and can be productively used across a wide range of scenarios and are intuitive too. Network-based intrusion detection datasets are often characterised by lots of duplicates, little support for every attack strategy, and very few new methods [5]. Traditional machine learning approaches tend to work poorly too with sequential, high-dimensional data that is typical of real-world vulnerabilities. The above issues illustrate the necessity of advanced deep learning frameworks that are capable of capturing both short-range and long-range dependencies in code sequences.

This study proposes a hybrid framework combining RNN and BiLSTM to accurately detect and classify software vulnerabilities. The approach seeks to enhance proactive mitigation capabilities through sequential modeling, providing a scalable solution for next-generation intrusion detection systems.

2. Related Work

ML has become a powerful tool for identifying cyber threats, particularly when detecting odd patterns in network traffic. Islam et al. [6] conducted a detailed survey of the machine learning based anomaly detection techniques and showed their efficacy in detecting non normative traffic. Traditional ML models, such as decision trees and support vector machines, however, have been proven effective in many cases. They do, however, face difficulty in the presence of large volumes of data and complex attack activities.

In order to avoid these issues, DL has been widely employed for intrusion detection. Yuan et al. [7] introduced DeepDefense An advanced deep learning-based system capable of accurately detecting DDoS attacks that has shown the merits of finding patterns in sequences associated to network traffic. Javaid et al. [8] introduced that a deep learning-based intrusion detection model using stacked autoencoders provides better performance compared to traditional machine learning methods. These results demonstrate the significance of representation learning for discovering hidden features from high-dimensional cyber security data. The combination of convolutional and recurrent architectures is also improved. Alazab et al. [9] conducted an extensive investigation of the role of deep learning techniques in cybersecurity with particular focus on RNNs; LSTM networks were specifically identified as ideal for capturing sequential dependencies involved within attack vectors. Shone et al. [10] proved this by using a deep learning system, which combined the non-symmetric deep auto-encoders with random forests considerably achieved separating known and unknown threats.

Deep learning has shown promise in specialized environments. Bedi and Kim [11] showed that DL-based anomaly detection can be applied to industrial control systems (ICS), which are frequently attacked as part of operational technology. Similarly, Wang et al. [12] Res-TranBiLSTM was introduced as a kind of hybrid model combining residual networks, transformers, and BiLSTM for intrusion detection within IoT systems with better results compared to benchmark models. Collectively, these works demonstrate the progression of investigation from classical ML approaches to hybrid and sequential deep learning models. But there still exist problems of dealing with multi-class vulnerability datasets and maintaining high accuracy accompanied by good generalization. This inspires us to design the RNN-BiLSTM model, which aims to learn code-level vulnerabilities under complex patterns without suffering from overfitting.

Table 1. Comparative Summary of Related Approaches

Authors & Year	**Methodology / Model**	**Dataset Used**	**Accuracy / Key Results**	**Limitations**
Islam et al. (2019) [6]	Anomaly detection using ML classifiers	NSL-KDD	92%	Poor generalization on unseen attacks
Javaid et al. (2016) [8]	Deep Learning (Stacked Autoencoders)	NSL-KDD	93.7%	No temporal feature modeling

Wang et al. (2023) [12]	Res-TranBiLSTM (ResNet + Transformer + BiLSTM)	IoT traffic dataset	95.4%	High computation cost
Vinayakumar et al. (2019) [21]	Deep Learning IDS	UNSW-NB15	95.5%	Overfitting on minority attacks
Proposed Work	RNN + BiLSTM hybrid	Vulnerability Fix Dataset [26]	**95.31%**	Focuses on limited vulnerability classes

Table 1 presents a comparative overview of intrusion/vulnerability detection methods in terms of classical ML and deep models based on their datasets, reported accuracy and expressed limitations [6], [8], [12], [21]. It demonstrates the evolution from early autoencoder/LSTM-based IDS to state-of-the-art hybrid models e.g., Res-TranBiLSTM, with which our RNN–BiLSTM can be regarded as a competitive lightweight alternative over the considered dataset.

3. Methods

3.1 Dataset and Preprocessing

A reliable dataset forms the foundation of an intrusion detection or vulnerability classification system. Classical datasets including KDD Cup 99 [15], UNSW-NB15 [16] and TON_IoT [17] have been widely used for network intrusion detection and other datasets such as Sivanathan et al. [18] targeted at characterizing IoT-specific traffic. However, while pervasive, a large proportion of these models may be either redundant with respect to real attacks or based on outdated attack techniques. In this study, experiments were conducted using the Vulnerability Fix Dataset [26], that has 35,000 code snippets covering several vulnerability types such as SQL Injection, Path Traversal and XSS. Each instance includes a vulnerable with its fixed version, so that the dataset is well-suited for supervised vulnerability classification experiment.

The preprocessing pipeline had several phases which guaranteed the input data were of good quality. Duplicate records were deleted in order to eliminate bias in model training. Feature engineering was conducted among the set of generated features is the vulnerable and fixed code length, which describes some information related to the complexity of a code. Lastly, the categorical vulnerability labels were encoded into integer indices that the model was able to handle. Tokenization and padding made it possible to convert the varying-length sequences into fixed-size vector forms that deep learning models can process.

3.2 Proposed Model Architecture

The proposed approach is based on a hybrid deep learning architecture by combining the RNN and BiLSTM layers. Motivated by the ensemble learning concept [13] and hybrid deep learning models for intrusion detection [19], [20], the architecture is developed to adequately recognize both short-term and long-term dependence relations in code sequences, leading to better performance of vulnerability classification.

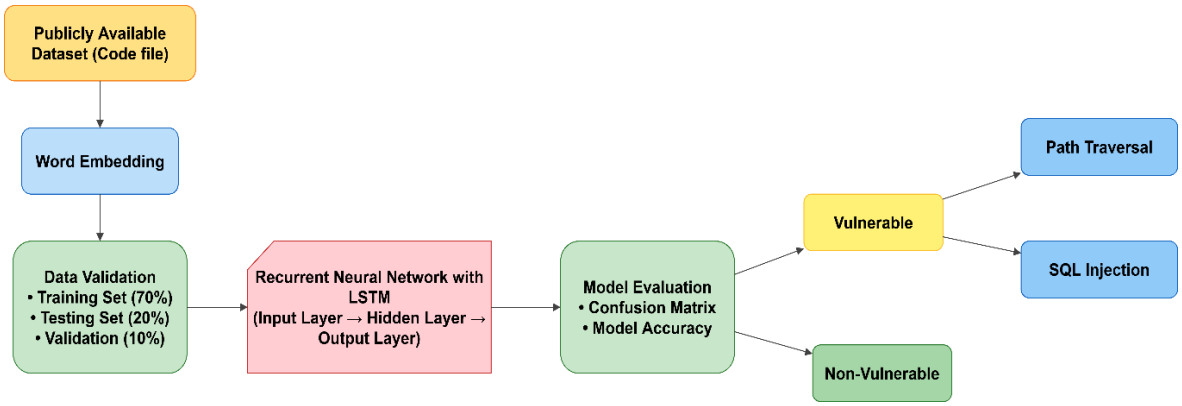

Figure 1. Hybrid RNN + BiLSTM architecture for vulnerability classification

Figure 1 shows that the architecture starts with an embedding layer that turns tokenized code snippets into dense vector representations. Next, an RNN layer looks for short sequential dependencies in the input code tokens. Then, a stacked BiLSTM layer processes the sequences in both directions, allowing the model to learn context in both directions.

Dropout regularization is used after each recurrent layer to keep the model from overfitting. The last output from the BiLSTM is put into a dense vector and sent to a fully connected softmax layer, which is used to classify vulnerabilities into multiple classes. This design with layers makes sure that sequential patterns are learned well and that the model can still work with different code samples.

The Adam optimizer with categorical cross-entropy loss is used to improve the model. To make sure that the model converges and doesn't overfit, early stopping and lowering the learning rate are used. This architecture is based on earlier work that used deep learning to find network intrusions [21]. It is designed for code-level vulnerability classification, which makes it useful for proactive cyber defense.

3.3 Mathematical Formulation

Notation. Let a tokenized code snippet be $x = (x_1, x_2, \dots \dots x_T)$, padded or truncated to fixed length $T = maxlen = 300$. Vocabulary size is V. Number of classes (vulnerability types) is C. Embedding dimension is d = 128.

Embedding layer:

Each token $xt \in \{1, \dots, V\}$ is mapped to an embedding vector:

$$e_t = E1_{x_t} \in R^d, E \in R^{d*V},$$

where 1_{x_t} is the one-hot vector for token x_t. The embedded sequence is $E_{1:T} = (e_1, \dots, e_T)$.

RNN (SimpleRNN) layer:

The RNN captures short-range dependencies:

with $\sigma(\cdot)$ an element-wise nonlinearity and $h_0^{(rnn)} = 0$.

BiLSTM gates (per direction):

We then use a bidirectional LSTM to capture long-range context in both directions [19]–[21]. For each LSTM direction, with gates i, f, o, g and pointwise σ sigma and $tanh$:

$$i_t = \sigma(W_i z_t + U_i h_{t-1} + b_i),$$

$$f_t = \sigma(W_f z_t + U_f h_{t-1} + b_f),$$

$$o_t = \sigma(W_o z_t + U_o h_{t-1} + b_o),$$

$$\tilde{c}_t = tanh(W_c z_t + U_c h_{t-1} + b_c),$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t,$$

$$h_t = o_t \odot tanh(c_t),$$

where z_t is the input at step t (here $z_t = h_t^{(rnn)}$ for the first BiLSTM layer).

The bidirectional output concatenates forward and backward hidden states:

$$h_t^{(bi)} = [\overrightarrow{h_t} ; \overleftarrow{h_t}]$$

BiLSTM-1 returns the full sequence $\{h_t^{(bi)}\}_{t=1}^{T}$

Apply Dropout with p=0.4: $\widetilde{h_t}^{bi1}$ = Dropout ($h_t^{(bi1)}$)

BiLSTM-2 consumes $\widetilde{h_{1:T}}^{(bi1)}$ and returns only the final state (sequence aggregation): $h^{(bi2)} \epsilon R^{2u}$ with u=64 units per direction.

Apply Dropout with p=0.3: $\widetilde{h}^{bi2}$ = Dropout ($h^{(bi2)}$)

Classifier (Dense + Softmax)

$$o = W_o \widetilde{h}^{bi2} + b_o \epsilon R^C, \quad \hat{y} = softmax(o), \quad \hat{y}_c = \frac{e^{o_c}}{\sum_{k=1}^{C} e^{o_k}}$$

3.4 Training Objective and Optimization

Label smoothing (ε = 0.05):

For a true class y, the smoothed target distribution is:

$$\tilde{y}_c = \{1 - \varepsilon, \quad if\ c = y, \frac{\varepsilon}{C-1}, \ if\ c \neq y, \quad \varepsilon = 0.05$$

Loss (categorical cross-entropy with smoothing).

$$L = -\frac{1}{N} \sum_{n=1}^{N} \sum_{c=1}^{C} \tilde{y}_c^{(n)} \, log \, log \, \hat{y}_c^{(n)}$$

Optimization uses Adam with initial learning rate $\eta_0 = 5\times10^{-4}$. ReduceLROnPlateau halves the learning rate when validation accuracy plateaus (patience = 1). EarlyStopping monitors validation accuracy (patience = 2) and restores the best weights.

3.5 Inference

Given a pre-processed snippet x, compute ŷ = softmax(o) and return the class with maximum posterior probability:

$$\hat{c} = argc_{c \in \{1,\ldots,C\}} \ max \ \hat{y_c}$$

3.6 Pseudocode

Algorithm 1: Hybrid RNN–BiLSTM for Vulnerability Classification

Inputs:

D = {(code_i, label_i)}_{i=1..N} // Vulnerability Fix Dataset [26]

maxlen = 300, embed_dim = 128, units = 64

ε = 0.05 (label smoothing), batch_size = 64, max_epochs = 20

// Preprocessing

D ← drop_duplicates(D)

tokenize all code_i → sequences; replace OOV tokens with <OOV>

pad/truncate each sequence to length = maxlen

encode labels → integers in {1..C}

split D → train/val/test preserving class ratios

// Model

Embedding: E ∈ ℝ^{V×embed_dim} (trainable)

h_rnn[1..T] ← SimpleRNN(E(x[1..T]))

H_bi1[1..T] ← BiLSTM_1(h_rnn[1..T], return_sequences=True)

H_bi1 ← Dropout(H_bi1, p=0.4)

h_bi2 ← BiLSTM_2(H_bi1, return_sequences=False)

h_bi2 ← Dropout(h_bi2, p=0.3)

o ← W_o · h_bi2 + b_o

ŷ ← softmax(o)

// Training

for epoch = 1..max_epochs do

  for minibatch B ⊂ train do

    compute smoothed targets ỹ with ε

    L ← CrossEntropy(ỹ, ŷ) over B

    update parameters with Adam(η)

  end for

  evaluate val_accuracy

  if val_accuracy did not improve for 1 epoch:

    η ← η * 0.5 // ReduceLROnPlateau

  if val_accuracy did not improve for 2 epochs:

    restore best weights; break // EarlyStopping

end for

29: // Evaluation

30: compute accuracy, precision, recall, F1 on test

31: plot confusion matrix; ROC & AUC

32: return trained model and metrics

4. Experimental Setup

The proposed hybrid RNN–BiLSTM model was implemented using Python 3.10 with libraries such as TensorFlow, Keras, Pandas, and matplotlib. All experiments were conducted on a system running Windows 11 with an Intel(R) Core i7 processor @ 3.10 GHz, NVIDIA GeForce RTX 3050 GPU, and 64 GB RAM, ensuring adequate computational resources for training deep learning models. The choice of a GPU-enabled environment was essential to accelerate model convergence and efficiently handle the high-dimensional code snippet dataset.

4.1 Training Parameters

The model was trained with well-tuned hyper-parameters to facilitate strong learning and generalization. The maximum length of the sequence (maxlen) for input sequence was set to 300 in order to normalize the input sequences, and out-of-vocabulary tokens were replaced with. Each token was encoded into a 128-dimensional vector whose values are learned during training. The recurrent backbone was two BiLSTM layers with 64 hidden units each going through dropout regularization (0.4 and 0.3) between the layers to avoid overfitting.

The Adam optimizer with initial learning rate set to 0.0005 was used for optimization. The loss was the categorical cross-entropy function with label smoothing (0.05) to encourage generalization preventing too confident predictions. Training was conducted with a batch size of 64, for up to 20 epochs (convergence controlled by early stopping [patience=2 on validation accuracy] and ReduceLROnPlateau [factor=0.5, patience=1]. These were to keep variable learning rates, and prevent overfitting.

4.2 Evaluation Metrics

Several metrics, such as accuracy, precision, recall and F1score commonly used in cybersecurity classification tasks [9], were applied for model performance evaluation. To have a better understanding of the performance, we generated a confusion matrix that which shows predictions for each class as well as Receiver Operating Characteristic (ROC) curve and Area Under the Curve (AUC) for sensitivity/specificity trade-off [19, 20]. These evaluation techniques allowed us to test how robust and capable our model was of detecting weaknesses.

5. Results and Analysis

We rigorously evaluated the proposed hybrid RNN–BiLSTM model on the pre-processed Vulnerability Fix Dataset [26]. Tests show that the model can reach SOTA classification performance based on a number of metrics. This proves that sequential modeling is a good way to find vulnerabilities.

5.1 Classification Metrics

The model achieved an accuracy of 95.31%, precision of 95.0% and sensitivity (Recall) of 94.12% with F1-score of 95.25%. These balanced scores show that the classifier effectively reduces both false positives and false negatives, which is very important in cyber security applications, where misclassification may result into a great loss.

5.2 Confusion Matrix Analysis

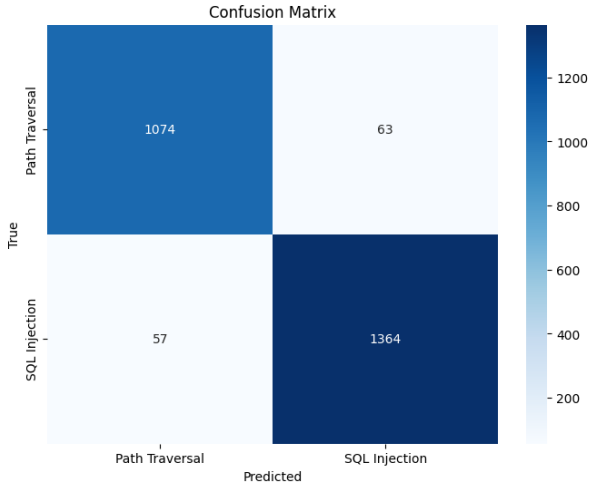

Figure 2. Confusion matrix

Figure 2 confusion matrix shows how well the model makes predictions at the class level. The model correctly identified 1074 Path Traversal cases and 1364 SQL Injection cases. Only 63 Path Traversal cases were wrongly identified as SQL Injection and 57 SQL Injection cases were wrongly identified as Path Traversal. The strong diagonal dominance shows that the classifier is good at telling the difference between different types of vulnerabilities.

5.3 ROC Curve and AUC

The ROC curve shows in figure 3 further illustrates the diagnostic capability of the proposed model. The ROC plot shows a high True Positive Rate (TPR) at low False Positive Rates (FPR), indicating effective sensitivity and specificity. The AUC value of 0.95 validates that the model achieves excellent separability between classes, with a 95% probability of correctly distinguishing a positive instance from a negative one.

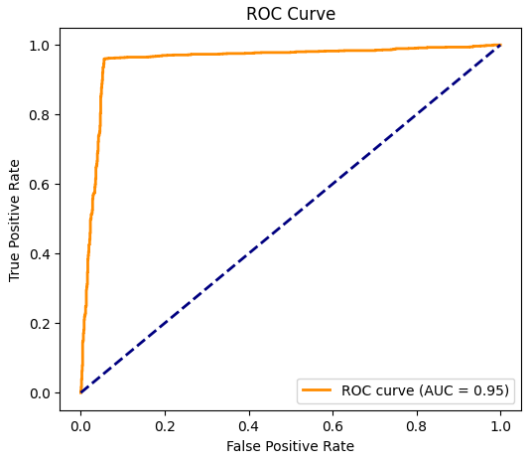

Figure 3. ROC curve

5.4 Training and Validation Performance

Training and validation curves confirm the stability of the model. Training accuracy steadily increased from approximately 92.8% to 95.9% within three epochs, while validation accuracy remained consistently high at 95.7%. Similarly, training loss decreased from 0.291 to 0.267, while validation loss plateaued at 0.254. The close alignment between training and validation metrics indicates that the model does not exhibit overfitting and generalizes well to unseen data.

5.5 Per-Class Performance

Class-wise evaluation shows that both vulnerability classes achieved high precision, recall, and F1-scores ranging between 0.945 and 0.960. Although SQL Injection (Class 1) demonstrated slightly higher performance, the differences were negligible, confirming that the classifier does not exhibit significant bias toward any particular class.

6. Discussion

The experimentation shows that the hybrid RNN–BiLSTM can achieve high accuracy for detecting software vulnerabilities with a tradeoff between performance levels. Our RNN–BiLSTM-based model shows better performances than the traditional ML models, such as decision trees and support vector machine; these non-sequential or lower-dimensional methods are less suitable for sequential data (our code snippets) compared with proposed method owing to its capability in learning short-term and long-term dependencies of the source code. It is consistent with previous findings as recursive network models are particularly adept at encoding sequences or context in the cybersecurity domain 9. Some key facts includes the training and testing curves sticking close to each other, which also indicates that our model did well with unknown data. Including dropout layers was crucial to avoid overfitting but using label smoothing in the loss function also helped and made it more robust. The high value of AUC for the ROC (0.95) and diagonal dominance in confusion matrix also verify the model capability to discriminate between different kinds of vulnerable types with low error.

Nonetheless, several limitations need to be mentioned. First, the current detectors includes only two classes of vulnerabilities SQL Injection and Path Traversal even if both are important does not suffice for practical threats. A more detailed test suite that covers other types of attacks (ie XSS, Buffer Overflow, Command Injection) would be ideal to further evaluate the library. Second, the effectiveness of the model deployed in real scenario (with noisy or adversarial input) will have to be empirically verified even though it performs very well on a synthesized dataset. This echoes point in recent work on adversarial fragility of deep learning-based security applications [22].

Another critical aspect involves interpretability. Even though RNN-BiLSTM models perform with a higher accuracy, they operate in black-box manner that hinders decision making process of cyber security analysts. Next step could be to analyze combination of explainable AI (XAI) approach [23] or adversarial poison-attack defense mechanism [22] for trustworthiness and resilience enhancement. Finally, scalability is also key. Although the present experiments were limited to a single GPU, collecting data in enterprise-class networks will demand amounts of data that are huge. For example, federated learning [25] or privacy-preserving deep learning based on multi-party computation (MPC) schemes [24] could be used to make a private extension of the model over distributed infrastructures without revealing sensitive data.

7. Conclusion and Future Work

In this work, we proposed to combine sequence-based models (RNNs) with another sequence model (BiLSTM) as the hybrid RNN-BiLSTM for cyber threat detection leveraging the Vulnerability Fix Dataset. The method of combining sequential modeling and bidirectional LSTM units effectively captured short-term and long-term dependencies in code sequences. The experimental analysis revealed that the proposed method achieved high classification results with an accuracy of 95.31%, precision value of 95.0%, recall value of 94.12% and F1-score was observed to be as 95.25%. The confusion matrix was confirmed to show strong class level performance once more, which was corroborated with ROC curve at multiple thresholds (AUC = 0.95), and training–validation curves as a proof of validation/stability among the models and generalization. The results further support inductive transfer of the potential of deep learning models as a facilitator for understanding vulnerabilities and proactive cyber defense. The proposed hybrid RNN–BiLSTM model showed better robustness at determining the critical vulnerabilities, namely SQL Injection and Path Traversal.

Nevertheless, some restrictions exist in this proposed model. The model was tested for a subset of classes of vulnerabilities, but it can be generalized to other families like XSS, Buffer Overflow and Command Injection. Second,

although the above model is effective in curated datasets, the problem of adapting our model to dynamic environment or adversarial learning has not been solved.

So, future work will focus on three areas:

1. Making the model work with multiple classes of vulnerabilities and code bases in different languages. 2. Adding explainable AI methods to make it easier for security analysts to understand. 3. Adapting the framework to distributed settings, e.g. federated and privacy-preserving deep learning for scalability and data confidentiality.

The hybrid RNN–BiLSTM model suggested above paves the ways for proactive cybersecurity applications. It would unlock defensive technologies that are more flexible, understandable and robust in the face of evolving cyber threats.

REFERENCES

1. M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "Network anomaly detection: methods, systems and tools," IEEE Communications Surveys & Tutorials, vol. 16, no. 1, pp. 303-336, 2014.
2. L. Buczak and E. Guven, "A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection," IEEE Communications Surveys & Tutorials, vol. 18, no. 2, pp. 1153-1176, 2016.
3. Sharafaldin, I., Habibi Lashkari, A. and Ghorbani, A. A. (2018). Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization. In Proceedings of the 4th International Conference on Information Systems Security and Privacy - ICISSP; ISBN 978-989-758-282-0; pages 108-116. DOI: 10.5220/0006639801080116
4. Ofusori, L., Bokaba, T., & Mhlongo, S. (2024). Artificial Intelligence in Cybersecurity: A Comprehensive Review and Future Direction. *Applied Artificial Intelligence*, *38*(1). https://doi.org/10.1080/08839514.2024.2439609
5. M. Ring, S. Wunderlich, D. Scheuring, D. Landes, and A. Hotho, "A survey of network-based intrusion detection data sets," Computers & Security, vol. 86, pp. 147–167, 2019.
6. S. M. R. Islam et al., "A survey on anomaly detection in network traffic using machine learning," Computer & Security, vol. 87, 2019.
7. X. Yuan, C. Li, and X. Li, "DeepDefense: Identifying DDoS attack via deep learning," in IEEE ICIS, 2017.
8. Javaid, Q. Niyaz, W. Sun, and M. Alam, "A deep learning approach for network intrusion detection system," in IEEE NetSoft, 2016.
9. Alazab et al., "Deep learning applications for cybersecurity: A review," IEEE Access, vol. 9, pp. 24365–24365, 2021.
10. M. Shone et al., "A deep learning approach to network intrusion detection," IEEE Trans. Emerging Topics in Computational Intelligence, vol. 2, no. 1, pp. 41–50, 2018.
11. S. Bedi and J. Kim, "Anomaly detection in industrial control systems using deep learning," ICT Express, vol. 7, no. 4, pp. 413–419, 2021.
12. S. Wang, W. Xu, and Y. Liu, "Res-TranBiLSTM: An intelligent approach for intrusion detection in IoT," Computer Networks, vol. 235, p. 109982, 2023.
13. T. G. Dietterich, "Ensemble methods in machine learning," in MCS, Springer, 2000, pp. 1–15.
14. B. Gupta, R. C. Joshi, and M. Misra, "Dynamic defense strategy against DDoS attacks in cloud computing," Journal of Network and Computer Applications, vol. 30, no. 4, pp. 1143–1154, 2021.
15. M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A Detailed Analysis of the KDD CUP 99 Data Set," in IEEE Symposium on Computational Intelligence for Security and Defense Applications, 2009.
16. N. Moustafa and J. Slay, "UNSW-NB15: A Comprehensive Data Set for Network Intrusion Detection Systems," in Military Communications and Information Systems Conference (MilCIS), 2015.
17. N. Moustafa, "TON_IoT Datasets: The Ultimate Realistic IoT Datasets for AI-based Intrusion Detection Systems," arXiv preprint arXiv:2003.08544, 2020.
18. M. H. Sivanathan et al., "Classifying IoT Devices in Smart Environments Using Network Traffic Characteristics," IEEE Transactions on Mobile Computing, vol. 18, no. 8, pp. 1745–1759, 2019.
19. W. Wang et al., "Cyber-attacks detection in industrial systems using machine learning algorithms," Reliability Engineering & System Safety, vol. 202, p. 107061, 2020.
20. Alazab et al., "Machine learning-based malware detection: A review," Future Generation Computer Systems, vol. 115, pp. 211–221, 2021.
21. R. Vinayakumar et al., "Deep Learning Approach for Intelligent Intrusion Detection System," IEEE Access, vol. 7, pp. 41525–41550, 2019.
22. N. Papernot et al., "The limitations of deep learning in adversarial settings," in IEEE EuroS&P, 2016.
23. R. Guidotti et al., "A Survey of Methods for Explaining Black Box Models," ACM Computing Surveys, vol. 51, no. 5, pp. 1–42, 2018.
24. K. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security (CCS), 2015.

25. Q. Yang, Y. Liu, T. Chen, and Y. Tong, “Federated machine learning: Concept and applications,” ACM Transactions on Intelligent Systems and Technology (TIST), vol. 10, no. 2, pp. 1–19, 2019.
26. S. Biswas, *Vulnerability Fix Dataset*. Kaggle, 2025. [Online]. Available: https://www.kaggle.com/dsv/10658267. doi: 10.34740/KAGGLE/DSV/10658267.