Secure Systems Design and Technology Development. Veliko Tarnovo, Bulgaria
Proceedings of the International Scientific Conference, 2026. Volume III, 45-52

Real-Time Network Traffic Capture and Analysis Using a Deep Neural Network

Angela R. Borisova
Computer Systems and Technologies Department
"Vasil Levski" National Military University
Shumen, Bulgaria
arborisova@nvu.bg

Krasimir Slavyanov
Computer Systems and Technologies Department
"Vasil Levski" National Military University
Shumen, Bulgaria
k.o.slavyanov@gmail.com

Abstract—**Nowadays, computer networks generate large and complex data sets, which leads to problems related to the detection of new, modified threats. Classic Intrusion Detection Systems – IDS, based on static rules and signatures, have difficulty identifying masked - as legitimate or previously unseen malicious activities. The challenge they face is reverse shell attacks, in which the attacker establishes a seemingly reliable connection with the victim. This study presents an approach aimed at solving a problem related to the cybersecurity of computer systems and networks. This is done by building, training, and testing the effectiveness of deep neural architectures for recognizing illegitimate reverse shell packets passing through a computer network in a controlled and real-world environment.**

The simulation of the attacks is carried out in a protected environment – Oracle Virtual Box. The interception of network packets in real time is carried out using a software tool written in the Python programming language. Packet sniffing itself is performed both in the virtual environment and in the real environment. The sniffer script extracts significant sets of characteristics from network packets, such as packet size, protocols used, ports, TCP flags, IP addresses - of the source (attacker) and the recipient (victim), and others. The current report compares several deep neural networks, namely One-dimensional convolution neural network/1D CNN, Long Short-Term Memory/LSTM and Autoencoder. As 1D CNN, it extracts local dependencies between network packet features for traffic classification. LSTM models capture temporal dependencies and sequences from network data, and autoencoders reconstruct normal network behavior patterns. The results of the experiments reveal the ability of deep neural networks to correctly model the behavior of network traffic and ultimately increase the recognition of compromised data. The proposed approach offers the opportunity to further automate the processes of detection and monitoring of computer systems, which in turn would lead to the protection of network infrastructure in modern dynamic environments.

Keywords — LSTM, 1D CNN, Autoencoder, reverse shell attacks

I. INTRODUCTION

The continuous development of network technologies and the rapidly increasing complexity of cyber threats make network security an extremely important component in communication and information systems. Classic machine learning algorithms and traditional intrusion detection systems (IDS - which are inherently signature-based) are unable to cope with the constantly changing modern cyberattacks, especially those in real-time conditions - known as unknown or zero-day.

Deep learning algorithms have the potential to overcome cyber challenges by extracting complex dependencies from large volumes of network data. Deep neural architectures such as Long Short-Term Memory (LSTM), convolutional neural networks (CNN), and autoencoders, are a good option due to their properties. Autoencoders capture illegitimate network packets thanks to their unsupervised learning nature, which helps them learn the normal behavior of network data flows. Long Short-Term Memory is capable of modeling temporal dependencies in data sequences, and convolutional neural networks capture spatial dependencies between network traffic characteristics.

To date, most studies have been based on pre-collected (offline) datasets and do not adequately address latency in the detection of cyberattacks. Since network traffic in real-world conditions is continuous and massive, it requires processing with minimal delay. Therefore, network anomaly detection models must be automated and capable of operating in a streaming environment in order to respond instantly [9] to a cyber incident [10].

This research paper presents an approach for capturing and analyzing network traffic both in controlled conditions and in real time, by leveraging the capabilities of three types of deep neural architectures. It compares the effectiveness of LSTM, 1D CNN, and autoencoder networks in detecting various types of network behavior—normal traffic, reverse shell attacks, and unknown anomalies.

The article is structured as follows: Section 2 – describes existing research in the field. Section 3 – describes the methodology and architecture of the model. Section 4 – describes the results of the study and the subsequent analysis. Section 5 – describes the conclusions

Online ISSN 3044-7771
https://doi.org/10.68302/std2026.vol3.112

from the work performed and provides guidelines for future work.

II. Overview of existing solutions

Article [1] proposes a deep learning-based **hybrid model** that combines **LSTM** and **CNN**. The main idea of the proposed hybrid architecture, called Attention-CNN-LSTM, is that an attention layer is added, aiming to improve accuracy by reducing noise pollution. For training and testing the neural architecture, the following datasets are used: NSL-KDD and Bot-IoT (they are realistic and publicly available datasets collected in a controlled laboratory environment).

Attention-CNN-LSTM achieves high accuracy of 94.8% (experiment conducted on NSL-KDD dataset) and 97.5% (experiment conducted on Bot-IoT dataset) and low false positive rate. However, the study is based on previously collected (offline) network data, which is outdated and does not contain encrypted traffic. Furthermore, the proposed model does not address the challenges associated with processing network packets in a streaming environment.

In the paper [2], a **hybrid model** for anomaly detection in photovoltaic systems is proposed, which combines a convolutional neural network and an autoencoder - **CNN-Autoencoder.** Although the paper is not directly related to network traffic analysis, it considers deep-structured neural architectures for recognizing normal and abnormal behavior, which is conceptually similar to the approaches in intrusion detection systems. The dataset used is GPVS-Faults. The GPVS-Faults data represent real measurements collected in advance by a photovoltaic system emulator via sensors.

The results show that the proposed CNN–Autoencoder hybrid architecture significantly improves the diagnostic accuracy compared to standard CNN models, reaching 98.84% accuracy. In addition, a significant reduction in the false alarm rate is observed. However, despite the high performance, the method is validated mainly on offline data collected from an emulator system, which limits its direct applicability in real dynamic environments.

The proposed neural architecture for intrusion detection in [3] combines two levels of deep learning, namely **Hybrid Autoencoder (HAE)** and **Hybrid Residual LSTM-CNN network (HRL)**. The datasets used in the study are previously collected and publicly available, namely: NSL-KDD, UNSW-NB15 and CICIDS-2018.

The analysis of the reviewed research shows that the model training is stabilized by using residual connections. However, the improvement in accuracy compared to classical models based on LSTM and CNN is relatively limited and is usually within 1–3%. The proposed combined model is characterized by increased architectural complexity, which leads to the need for significant computational resources and a higher implementation cost. In addition, there is a partial overlap of functionalities between the individual components – the autoencoder compresses and represents the input space, while the CNN performs feature extraction, with both modules implementing different aspects of data representation. Additionally, the increased complexity of the model creates prerequisites for the occurrence of overfitting. Furthermore, the experimental evaluation was performed in an offline environment, which limits the possibility of evaluating the behavior of the model under real network traffic.

In the paper [4], a comparative analysis is made between the efficiency and accuracy of **CNN + Bidirectional LSTM (BiLSTM)** and **machine learning models**. The datasets involved in the study are: NSL-KDD, UNSW-NB15.

The experimental results show that deep neural models achieve higher accuracy (98.96% – 99.78%) compared to classical machine learning approaches (95% – 98%) in intrusion detection tasks. However, it should be noted that the evaluation was performed on static data in an offline environment, which limits the direct applicability of the results to real-world conditions.

Paper [5] addresses the security issue in software-defined networking, SDN, emphasizing that the centralization and programmability of the network create new vulnerabilities. The authors propose the application of advanced deep-structured models - a **hybrid CNN-LSTM** model and a **Transformer encoder** model - for attack detection, focusing on SDN security. The dataset used is InSDN. This dataset was previously collected in a controlled laboratory environment and contains both normal traffic and simulated attacks.

According to the research, the accuracy of CNN-LSTM, relative to the number of input features, is in the range of 98.20% - 99.01%, and the accuracy of Transformer encoder, again relative to the number of input features, is in the range of 98.50% - 99.02%. After merging rare classes and applying binary classification, the accuracy reaches 99.19%. However, the results were obtained on controlled data and do not guarantee the same efficiency in a real environment.

Paper [6] presents a hybrid architecture - **Autoencoder-LSTM/BiLSTM** for intrusion detection. Initially, the autoencoder is used to detect anomalies by comparing the input data and its reconstruction error. Then, the LSTM network classifies the suspicious flows by analyzing their time sequence. The dataset used is NSL-KDD.

The Autoencoder-LSTM model shows improvement over classical ML methods, but the achieved accuracy – 89% and the high false alarm rate – 11% make it less reliable than modern deep-structured anomaly detection architectures. The results are valid mainly for offline controlled conditions and do not fully reflect the behavior in real network environments.

Paper [7] presents an overview of **CNN-based approaches** for network intrusion detection, analyzing different architectures, feature extraction methods, and datasets used. The main conclusion is that CNN models are effective in automatically extracting features from network traffic, but their use alone is limited in terms of

modeling temporal dependencies. Therefore, hybrid architectures combining CNN with other deep models show better overall performance. The study itself was conducted in a controlled environment and was not tested in a real-world setting.

Article [8] proposes an **LSTM** model optimized with **metaheuristic algorithms** for detecting network attacks. The aim of the study is to automatically tune the hyperparameters of LSTM – number of neurons, layers, etc. The neural model was trained and tested using 3 standard datasets: NSL-KDD, CICIDS2017 and BoT-IoT.

After the experiments, it was found that the best result is achieved by SSA-LSTM (LSTM + Salp Swarm Optimization). SSA-LSTM reaches an accuracy of approximately 99%, and it also has the lowest false positive rate. LSTM is a good option for detecting intrusions, but on its own it is not optimal. The research in the article was conducted in an offline environment.

Contrary to the research conducted and the good results of hybrid models, three separate, non-combined deep neural networks will be examined and an attempt will be made to prove that with appropriate model design, they would give almost excellent results in detecting anomalies in computer networks, both in a controlled and real environments.

III. Materials and methods

To fulfill the task set in this report, namely the collection and analysis of network data in real time, three neural network multilevel models – LSTM, One-Dimensional Convolutional Neural Network (1D CNN) and Autoencoder – have been developed, trained and tested for extracting features from network traffic. The three models model network communication through statistical, behavioral, and semantic representations. In order to describe the temporal and statistical characteristics of network traffic, parameters are used that describe the dynamics of the quantitative properties of the traffic, namely: length of a specific packet, number of packets per second, interval between consecutive packets, indicators of burst behavior - many packets for a very short time interval, throughput in bytes per second. To describe the behavioral characteristics of the network, the dynamics and abnormal behaviors in the network data are monitored, such as: small packet indicator, retransmission detection, keep-alive and zero-window states, repeating ACK events, consistency of the communication direction - describing the degree of consistency in the alternation of incoming and outgoing packets within a flow. The semantics of the network is represented by examining the meaning of the traffic behavior, using shell/command pattern recognition.

For the purposes of the experiment, the following are used: Python 3.13 programming language, PyCharm 2025.2.1 integrated script development environment, Oracle VirtualBox virtual environment, Kali Linux machine, Windows Server 2025 machine and Wireshark 4.6.4 software. A comprehensive system for identifying normal and abnormal network behavior is created using Python programming language and PyCharm development environment. Reverse shell type network traffic is simulated in a controlled, virtual Oracle VirtualBox environment, using a Kali Linux machine - playing the role of an attacker and a Windows Server 2025 machine - playing the role of a victim. Network data is collected using the Wireshark platform. To make the three neural networks used equal, they use a single pipeline architecture based on flow-aware segmentation and time windows of network traffic. Each sequence of packets represents a local time segment of network traffic, on which feature extraction is performed with a unified set of 60 features. Network flows are built based on a five-element identifier, including the sender's IP address, sender's port, recipient's IP address, recipient's port, TCP/UDP protocol. The data is organized into sliding windows of length 5 and step 1, which allows capturing local time dependencies in the traffic. To prevent mixing of unrelated sessions and detect intermittent or intermittent attacks, a time threshold of 30 seconds is introduced in all three models. Sequence labeling is based on majority voting, assuming that a sequence belongs to an attack class if more than 30% of the packets it contains are malicious. This ensures consistency between supervised 1D CNN and LSTM models and the unsupervised autoencoder.

There are three classes involved in the study, namely: two main classes - a normal traffic class and a reverse shell attack class, and one additional class called Unknown - if the network data is unknown to the models.

Again, to ensure that the networks used are equal, training is performed by setting the same hyperparameters: Adam optimizer, learning rate 0.0005, batch size 64, L2 regularization 1e-4, and up to 150 epochs. To prevent overtraining, techniques such as: L2 regularization, learning rate reduction at standstill, early stopping, and dropout (0.3-0.5) are used. To balance the distribution of classes over training, validation, and testing in LSTM and 1D CNN, class weights and Focal Loss with parameters $\gamma = 2.0$, $\alpha = 0.25$ are used.

The data set used to train the three networks is the same. It is collected both in a controlled virtual environment and in a real one. Its size is approximately 44,000 packets. The raw data is converted to csv format and subsequently labeled with appropriate labels (for normal traffic – BenignTraffic, for reverse shell attack – RevShellAttack). Each record in comma separated value format contains sender information, recipient information, time between two consecutive packets within a flow, protocol, length of a specific packet, and a text field with additional metadata – TCP flags, TCP window size, HTTP, DNS, TLS handshake, shell commands, and more. To be brought into a single numerical standard, the data is normalized using StandardScaler. Categorical data (text data) is processed using label encoding (to convert labels to numerical format) and one-hot encoding (to convert protocol data to numerical format).

To prevent information leakage, data splitting is done at the data stream level, not at the individual packet level. Data for LSTM, 1D CNN networks is split as follows: 80% for training – normal traffic + recursive attacks, 10% for validation – normal traffic + recursive attacks, and 10% for testing – normal traffic + recursive attacks. In the case of the autoencoder, due to its nature of unsupervised learning,

the data is divided as follows: 80% for training - only data from normal traffic, 10% for validation of data from normal traffic and 50% for validation of data from reverse shell attacks, and 10% for testing data from normal traffic and 50% for testing data from reverse shell attacks.

The architectures of the three main deep neural networks look like this:

1. 1D CNN model architecture:

- input layer with the form: (sequence_length, num_features)
- first convolutional layer with dilation 1 and 32 neurons;
- Batch normalization;
- introducing nonlinearity through the Swish activation function;
- dropout layer – 0.3 for regularization;
- second convolutional layer with dilation 2 and 64 neurons;
- Batch normalization;
- introducing nonlinearity through the Swish activation function;
- max pooling with a window size of 2;
- dropout layer – 0.4 for regularization;
- global average pooling;
- fully connected layer with L2 regularization;
- Batch normalization;
- introducing nonlinearity through the Swish activation function;
- Softmax output layer for binary classification.

2. LSTM model architecture:

- input layer with the form: (sequence_length, num_features)
- first Bidirectional LSTM layer with 64 neurons and L2 regularization – processing sequences in both directions, capturing local and medium-term dependencies;
- Batch normalization;
- dropout layer - 0.3 for regularization;
- second Bidirectional LSTM layer with 32 neurons and L2 regularization;
- Batch normalization;
- dropout layer - 0.4 for regularization;
- fully connected layer with L2 regularization;
- Batch normalization;
- introducing nonlinearity through the Swish activation function;
- dropout layer – 0.5 for regularization;
- output Softmax layer for binary classification.

3. Autoencoder model architecture:

- input layer with the form: (sequence_length, num_features)

Encoder:

- first convolutional layer in the encoding part with 64 neurons and L2 regularization;
- Batch normalization;
- introducing nonlinearity through the Swish activation function;
- dropout layer - 0.3 for regularization;
- second convolutional layer in the encoding part with 32 neurons and L2 regularization;
- Batch normalization;
- introducing nonlinearity through the Swish activation function;
- dropout layer - 0.4 for regularization;
- flatten layer and L2 regularization.

Decoder:

- first fully connected layer and L2 regularization;
- dropout layer - 0.4 for regularization;
- second fully connected layer and L2 regularization;
- reshape layer;
- dropout layer - 0.3 for regularization;
- first convolutional layer in the decoding part with 64 neurons and L2 regularization;
- Batch normalization;
- introduction of nonlinearity through the Swish activation function;
- second, reconstructing convolutional layer in the decoding part with 64 neurons and L2 regularization.

What distinguishes the three neural networks built in this study from most models used in practice for detecting malicious packets is the use of L2 regularization, Swish activation function, and convolutional layers with dilation.

In the framework of this study, experiments were conducted to collect, monitor and analyze the behavior of network traffic using sniffer scripts based on LSTM, 1D CNN and Autoencoder models. The study was carried out simultaneously, both in controlled conditions, through statically configured network parameters, and in real conditions, through dynamic IP address allocation. The metrics for evaluating the performance of the convolutional neural network, LSTM and autoencoder networks are: accuracy, precision, recall, F1-score, ROC-AUC and Confusion Matrix.

IV. Results and Discussion

For the purposes of this report, experiments were conducted on network data recognition in a controlled and real-world environment. The Long Short-Term Memory, Convolutional Neural Network, and Autoencoder sniffers showed really good results – Table 1 and Fig. 1. – Fig. 9. It can be seen that each model processes between 7650 and 8565 network packets, generating approximately 6000 classification sequences. As far as possible, a balanced data set was collected to ensure unbiased evaluation of the models. Temporal LSTM models achieve 99.95% accuracy, with only 3 classified as Unknown, real recursive packet sequences. Spatial 1D CNN methods achieve 99.92% accuracy, with 5 uncertain detections, 3 of which are real recursive attacks and 2 of which are real normal traffic. The autoencoder, despite its unsupervised nature during training, achieves 100% accuracy, and proves to be an excellent solution for scenarios involving zero-day attacks. All three deep neural architectures achieve amazing ROC-AUC results – 1.000, indicating perfect discrimination between normal and malicious traffic. The ROC curves show that the models achieve 100% true positive rate, 0% false positive rate for LSTM and autoencoder. For 1D CNN, due to the negligible false positive rate of 0.07%, the ROC curve returns a value of 1.000 again.

TABLE 1 SUMMARY OF PERFORMANCE INDICATORS OF DEEP NEURAL MODELS – LSTM, CNN, AUTOENCODER

Model	Packets	Sequences	Benign Traffic	RevShell Attack	Unknown	Accuracy	Precision	Recall	F1-score	AUC
LSTM	8565	6049	2919	3130	3	99.95%	100%	99.90%	99.95%	1.000
1D CNN	7650	6039	2910	3129	5	99.92%	99.94%	99.90%	99.92%	1.000
Autoencoder	8180	6033	2903	3130	0	100%	100%	100%	100%	1.000

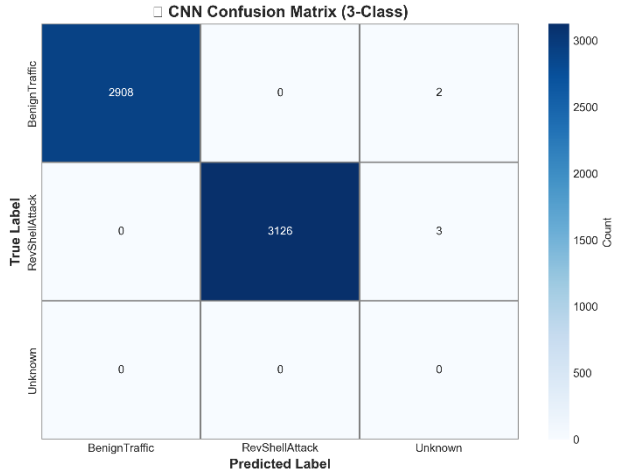

Fig. 1. Confusion matrix results in 1D CONVOLUTIONAL NEURAL NETWORK.

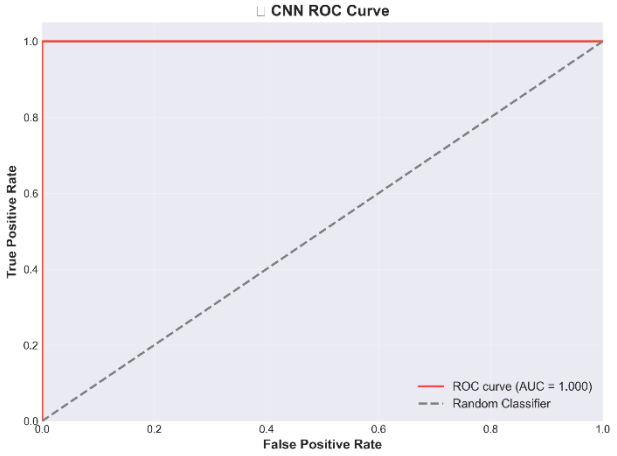

Fig. 3. 1D CONVOLUTIONAL NEURAL NETWORK performance evaluation via Receiver Operating Characteristics

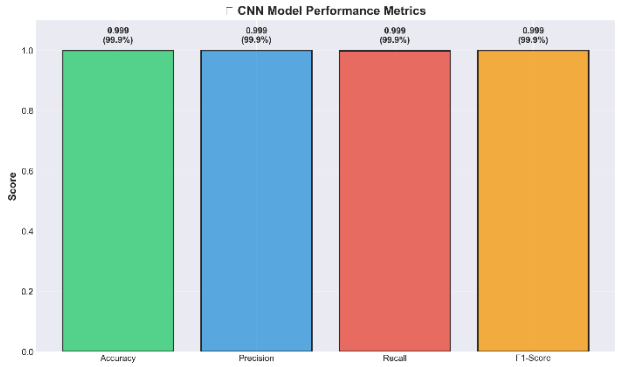

Fig. 2. Accuracy, precision, recall and F1-score of a 1D CONVOLUTIONAL NEURAL NETWORK.

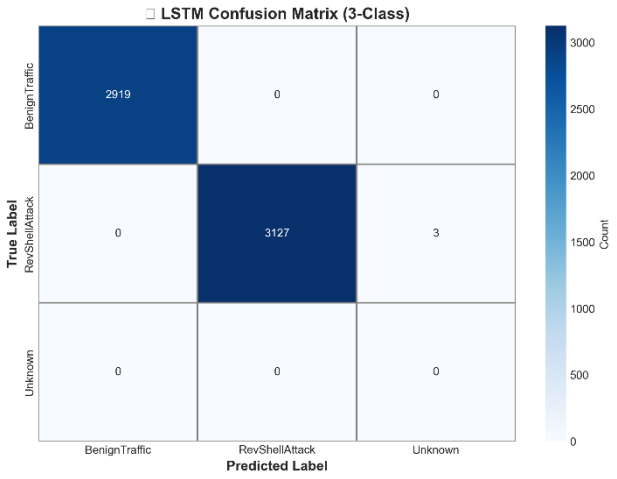

Fig. 4. Confusion matrix results in LSTM architecture.

Online ISSN 3044-7771
https://doi.org/10.68302/std2026.vol3.112

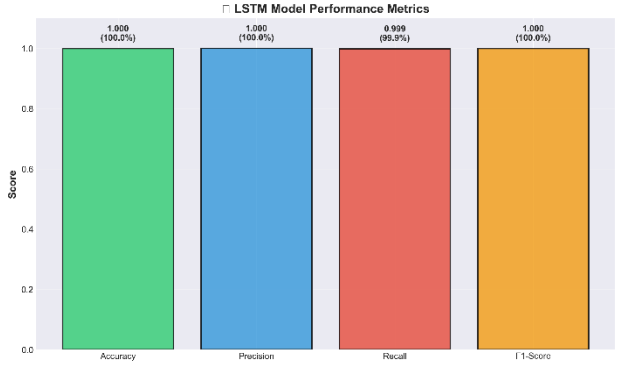

Fig. 5. Accuracy, precision, recall and F1-score of a LSTM

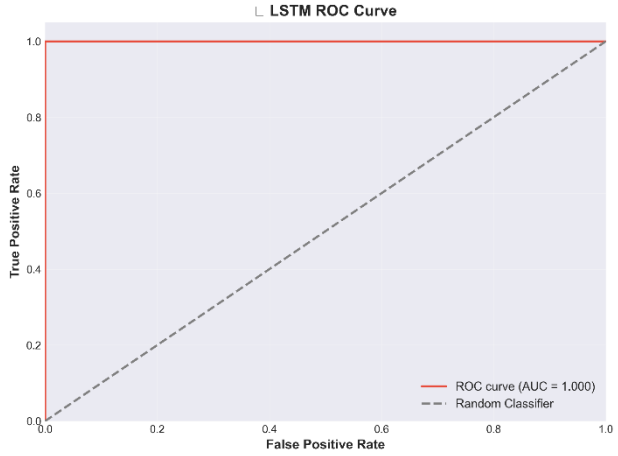

Fig. 6. LSTM performance evaluation via Receiver Operating Characteristics.

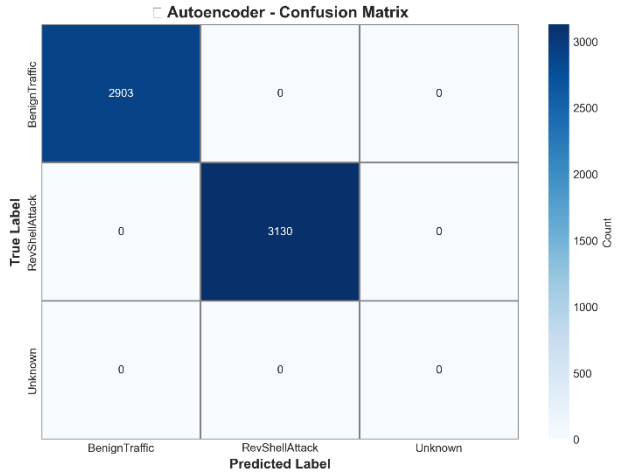

Fig. 7. Confusion matrix results in Autoencoder architecture

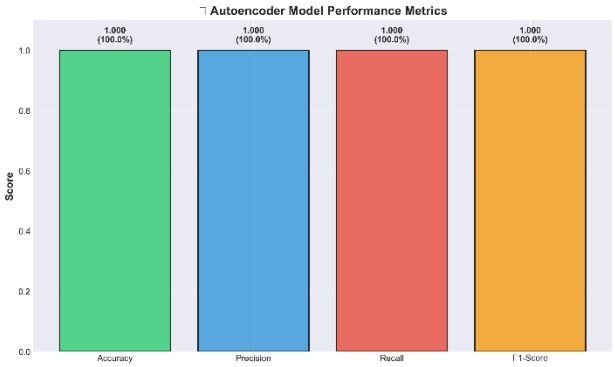

Fig. 8. Accuracy, precision, recall and F1-score of an Autoencoder

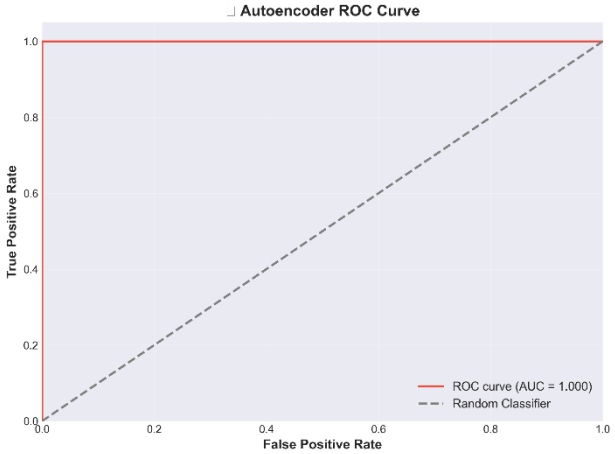

Fig. 9. Autoencoder performance evaluation via Receiver Operating

Table 2 compares the three neural architectures considered in this study with the 8 models considered in section 2 - Overview of existing solutions.

TABLE 2 COMPARISON OF OF DEEP NEURAL MODELS

Study	Year	Model	Dataset	Accuracy	Attack type
Present study	2026	LSTM	real live traffic	99.95%	Reverse Shell Attack + Unknown

Present study	2026	1D CNN	real live traffic	99.92%	Reverse Shell Attack + Unknown
Present study	2026	Autoencoder	real live traffic	100%	Reverse Shell Attack + Unknown
[1]	2025	Attention-CNN-LSTM	Bot-IoT	97.5%	Multi-class
[1]	2025	Attention-CNN-LSTM	NSL-KDD	94.8%	Multi-class
[2]	2025	CNN-Autoencoder	GPVS-Faults	98.84%	photovoltaic data
[3]	2025	Hybrid Autoencoder/HAE и Hybrid Residual LSTM-CNN	NSL-KDD, UNSW-NB1, CICIDS-2018	~98-99%	Multi-class
[4]	2024	CNN + Bidirectional LSTM	NSL-KDD	99.78%,	Multi-class
[4]	2024	CNN + Bidirectional LSTM	UNSW-NB15	98.96%	Multi-class
[5]	2024	CNN-LSTM	InSDN	98.20% - 99.01%	SDN-attacks
[5]	2024	Transformer	InSDN	98.50% - 99.02%	SDN-attacks
[6]	2022	Autoencoder-LSTM/BiLSTM	NSL-KDD	89%	Multi-class
[7]	2022	CNN based models	KDD Cup99	~98%-99.9%	Multi-class
[7]	2022	CNN based models	NSL-KDD	~95%-99%	Multi-class
[7]	2022	CNN based models	UNSW-NB15	~ 80% - 90% (CNN), 90% - 99% (hybrid CNN)	Multi-class
[7]	2022	CNN based models	CIC-IDS2017	~98%-99.9%	Multi-class
[7]	2022	CNN based models	CSE-CIC-DS2018	~ 98%-99.9%	Multi-class
[8]	2025	NSL-KDD, CICIDS2017, BoT-IoT	LSTM based models	best score – 99%	Multi-class

The results obtained show that the proposed models (1D CNN, LSTM and autoencoder) achieve very high attack detection accuracy, reaching 100%. Compared to existing studies [1]–[8], where the accuracy usually varies between 95% and 99%, the proposed approaches demonstrate comparable or higher efficiency. It is important to note that the models proposed in the paper are tested on real network traffic, which makes them more applicable in practical conditions. In addition, an ability to detect unknown attacks is observed, which is a significant advantage over traditional methods trained on predefined classes.

V. CONCLUTIONS

The experimental results show very high detection accuracy, reaching 99.95% for the LSTM model, 99.92% for the 1D CNN, and 100% for the autoencoder. Compared to existing studies [1] – [8], where the accuracy typically varies between 95% and 99% on standard datasets such as NSL-KDD, CIC-IDS2017, and UNSW-NB15, the proposed models demonstrate better performance.

A significant advantage of the present study is the use of real network traffic.

Despite the results achieved, it should be noted that the very high accuracy, especially for the autoencoder, needs to be confirmed by simulating at least one more type of cyberattack. Furthermore, direct comparison with other studies is limited due to differences in the datasets and methodologies used.

Future work will be aimed at testing the models on more types of network attacks, improving their generalization ability, as well as their integration into systems for real-time network traffic analysis.

In conclusion, the obtained results confirm that deep learning represents an effective approach for detecting network attacks and has significant potential for improving modern cybersecurity systems.

ACKNOWLEDGEMENTS

This paper is created in favor of Bulgarian National Scientific program “Security and Defense”, Ministry Council decision No 731/21.10.2021, Agreement No Д01-74/19.05.2022, Ministry Council decision No 386/07.05.2026.

REFERENCES

[1] A. M. Alashjaee, "Deep learning for network security: an Attention-CNN-LSTM model for accurate intrusion detection," Scientific Reports, vol. 15, Art. no. 21856, 2025, https://doi.org/10.1038/s41598-025-07706-y.

[2] S. Titouni et al., "Hybrid CNN-Autoencoder model for accurate and efficient fault diagnosis in grid-connected photovoltaic systems," International Journal of Electrical Power & Energy Systems, vol. 173, Art. no. 111454, Dec. 2025, https://doi.org/10.1016/j.ijepes.2025.111454.

[3] Y. Xue, C. Kang, and H. Yu, "HAE-HRL: A network intrusion detection system utilizing a novel autoencoder and a hybrid enhanced LSTM-CNN-based residual network," Computers & Security, Art. no. 104328, 2025, https://doi.org/10.1016/j.cose.2025.104328.

[4] M. Udurume, V. Shakhov, and I. Koo, "Comparative Analysis of Deep Convolutional Neural Network—Bidirectional Long Short-Term Memory and Machine Learning Methods in Intrusion Detection Systems," Applied Sciences, vol. 14, no. 16, Art. no. 6967, 2024, https://doi.org/10.3390/app14166967.

[5] M. S. Ataa, E. E. Sanad, and R. A. El-khoribi, "Intrusion detection in software defined network using deep learning approaches," Scientific Reports, vol. 14, Art. no. 29159, 2024, https://doi.org/10.1038/s41598-024-79001-1.

[6] E. Mushtaq, A. Zameer, M. Umer, and A. A. Abbasi, "A two-stage intrusion detection system with auto-encoder and LSTMs," Applied Soft Computing, vol. 128, Art. no. 108768, 2022, https://doi.org/10.1016/j.asoc.2022.108768.

[7] L. Mohammadpour, T. C. Ling, C. S. Liew, and A. Aryanfar, "A Survey of CNN-Based Network Intrusion Detection," Applied Sciences, vol. 12, Art. no. 8162, 2022, https://doi.org/10.3390/app12168162.

[8] N. Dash, S. Chakravarty, A. K. Rath et al., "An optimized LSTM-based deep learning model for anomaly network intrusion detection," Scientific Reports, vol. 15, Art. no. 1554, 2025, https://doi.org/10.1038/s41598-025-85248-z .

[9] L. G. Nikolov, "Email social engineering in action," in Proceedings of International Scientific Conference “Defense Technologies” (DefTech 2024), Faculty of Artillery, Air Defense and Communication and Information Systems, 2024, ISSN 2815-4282, pp. 395–402.

[10] L. G. Nikolov, "Social engineering cyberattacks," in Proceedings of International Scientific Conference “Defense Technologies” (DefTech 2024), Faculty of Artillery, Air Defense and Communication and Information Systems, 2024, ISSN 2815-4282, pp. 403–408.

Online ISSN 3044-7771
https://doi.org/10.68302/std2026.vol3.112