Review research
УДК 004.056
https://doi.org/10.31854/1813-324X-2025-11-3-72-86
EDN:HSXTLS

A Comprehensive Review of Deep Learning in Intrusion Detection Systems

Mokhalad M.A. Al-Tameemi[1], Almokhalad44@gmail.com
Abbas A.H. Alzaghir[2], a.a.h.alzagi@mtuci.ru
Malik A.M. Alsweity[3], al-sveiti.mam@sut.ru

[1]Saint Petersburg Electrotechnical University "LETI",
St. Petersburg, 197022, Russian Federation
[2]Moscow Technical University of Communication and Informatics,
Moscow, 123423, Russian Federation
[3]The Bonch-Bruevich Saint Petersburg State University of Telecommunications,
St. Petersburg, 193232, Russian Federation

Annotation

Deep learning methods play a crucial role in enhancing the effectiveness of intrusion detection systems. This study presents a comparative analysis of seven deep learning models, including autoencoders, restricted Boltzmann machines, deep belief networks, convolutional and recurrent neural networks, generative adversarial networks, and deep neural networks. The primary focus is on accuracy, precision, and recall metrics, evaluated using the NSL-KDD dataset. The analysis demonstrated the high effectiveness of recurrent neural networks, which achieved an accuracy of 99.79 %, precision of 99.67 %, and recall of 99.86 %.

The objective of the study: *of this paper is to enhance the effectiveness of intrusion detection systems through a comparative analysis of the performance of various deep learning models and an assessment of their applicability in the context of dynamic network security threats.*

The proposed solution involves *a comparative analysis of seven deep learning models to identify the most effective ones for network security tasks. This analysis aids in selecting the optimal models for specific security requirements.*

The evaluation methodology *involves the use of the benchmark dataset NSL-KDD, which contains various types of attacks and normal connections. The key evaluation metrics are accuracy, precision, and recall.*

The system implementation *is based on deep learning frameworks such as TensorFlow. The results of the system's performance and their interpretation are presented in the paper.*

Experiments *with the NSL-KDD dataset demonstrated accuracy, precision, and recall for all the deep learning models considered.*

The scientific novelty *is the ability to obtain formal performance evaluations of various deep learning models for intrusion detection systems, taking into account their architectural features, the processing of temporal and spatial data, as well as the characteristics of network traffic and attack types.*

The theoretical significance *is the expansion of methods for evaluating the effectiveness of intrusion detection systems through the analysis and comparison of the performance of deep learning models in the context of processing complex and high-dimensional network data.*

The practical significance *is the application of the comparative analysis results for selecting the most effective solutions in intrusion detection systems and optimizing them for real-world operating conditions.*

Keywords: *deep learning, intrusion detection systems, autoencoders, restricted boltzmann machines, deep belief networks, convolutional neural networks, recurrent neural networks, generative adversarial networks, network security*

For citation: Al-Tameemi M.M.A., Alzaghir A.A.H., Alsweity M.A.M. A Comprehensive Review of Deep Learning in Intrusion Detection Systems. *Proceedings of Telecommunication Universities.* 2025;11(3):72–86. DOI:10.31854/1813-324X-2025-11-3-72-86. EDN:HSXTLS

Обзорная статья
https://doi.org/10.31854/1813-324X-2025-11-3-72-86
EDN:HSXTLS

Комплексный обзор глубокого обучения в системах обнаружения вторжений

Мохалад М.А. Аль-Тамими[1], Almokhalad44@gmail.com
Аббас А.Х. Алзагир[2], a.a.h.alzagi@mtuci.ru
Малик А.М. Аль-Свейти[3], al-sveiti.mam@sut.ru

[1]Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В.И. Ульянова (Ленина), Санкт-Петербург, 197022, Российская Федерация
[2]Московский технический университет связи и информатики, Москва, 123423, Российская Федерация
[3]Санкт-Петербургский государственный университет телекоммуникаций им. проф. М.А. Бонч-Бруевича, Санкт-Петербург, 193232, Российская Федерация

Аннотация

Методы глубокого обучения играют ключевую роль в повышении эффективности систем обнаружения вторжений. В работе проведен сравнительный анализ семи моделей глубокого обучения, включая автоэнкодеры, ограниченные машины Больцмана, сети глубокого убеждения, сверточные и рекуррентные нейронные сети, генеративно-состязательные сети и глубокие нейронные сети. Основное внимание уделено метрикам точности, прецизионности и полноты на основе датасета NSL-KDD. Анализ показал высокую эффективность рекуррентных нейронных сетей, достигших точности 99,79 %, прецизионности 99,67 % и полноты 99,86 %. ***Цель статьи*** *– повышение эффективности систем обнаружения вторжений через сравнительный анализ производительности различных моделей глубокого обучения и оценку их применимости в условиях динамичных угроз сетевой безопасности.*

Предлагаемое решение состоит *в сравнительном анализе семи моделей глубокого обучения, чтобы выявить наиболее эффективные для задач защиты сети. Данный анализ помогает выбрать оптимальные модели для конкретных условий безопасности.* ***Методика оценки*** *включает использование эталонного набора данных NSL-KDD, который содержит различные типы атак и нормальных соединений. Ключевые метрики оценки – точность, прецизионность и полнота.* ***Реализация*** *системы выполнена на основе фреймворков глубокого обучения, таких как TensorFlow.* ***Эксперименты*** *с набором данных NSL-KDD показали точность, прецизионность и полноту для всех рассмотренных моделей глубокого обучения.*

Научная новизна *заключается в возможности получения формальных оценок производительности различных моделей глубокого обучения для систем обнаружения вторжений, с учетом их архитектурных особенностей, обработки временны́х и пространственных данных, а также характеристик сетевого трафика и типов атак.*

Теоретическая значимость *заключается в расширении методов оценки эффективности систем обнаружения вторжений путем анализа и сравнения производительности моделей глубокого обучения в условиях обработки сложных и высокоразмерных сетевых данных.*

Практическая значимость *заключается в применении результатов сравнительного анализа для выбора наиболее эффективных решений в системах обнаружения вторжений и их оптимизации для реальных условий эксплуатации.*

Ключевые слова: *глубокое обучение, системы обнаружения вторжений, автоэнкодеры, ограниченные машины Больцмана, сети глубоких убеждений, сверточные нейронные сети, рекуррентные нейронные сети, генеративно-состязательные сети, сетевая безопасность*

Ссылка для цитирования: Аль-Тамими М.М.А., Алзагир А.А.Х., Аль-Свейти М.А.М. Комплексный обзор глубокого обучения в системах обнаружения вторжений // Труды учебных заведений связи. 2025. Т. 11. № 3. С. 72–86. (in Russ.) DOI:10.31854/1813-324X-2025-11-3-72-86. EDN:HSXTLS

1. INTRODUCTION

Deep learning is widely used in computer or network security and we can define it as a type of machine learning methods based on artificial neural networks and the idea of learning representations. The term "deep" comes from the multilayered structure of the network. In general, deep learning models are of several types, including DNNs [1], CNNs [2], DBNs [3], and RNNs [4], which have been categorized under the different exemplars of the supervised learning model. Concurrently, restricted Boltzmann machines (RBMs) [5], autoencoders (AEs) [5,6], and generative adversarial networks (GANs) [7] are some of the models that can be used for unsupervised learning. In deep learning, features can be automatically extracted from raw data, like images and text and analyses the data [8, 9], this capability is particularly useful in intrusion detection systems, where deep learning models can process complex and high-dimensional data to identify potential threats [9, 10], so feature engineering by hand is not necessary. This skill allows the use of deep models on many kinds of data and gives it a considerable advantage over shallow models, mainly in dealing with vast datasets [11]. Deep learning techniques may be applied in the anomaly detection field for dimensionality reduction and classification tasks. Handcrafted feature engineering becomes insufficient for the increasingly larger, high-dimensional datasets; instead, deep learning models have been able to automatically capture complicated knowledge in these forms of data. At the same time, they may adapt to network behavior and dynamically varying attributes in attack scenarios [12]. Figure 1 shows the basic architecture for a deep learning-based IDS.

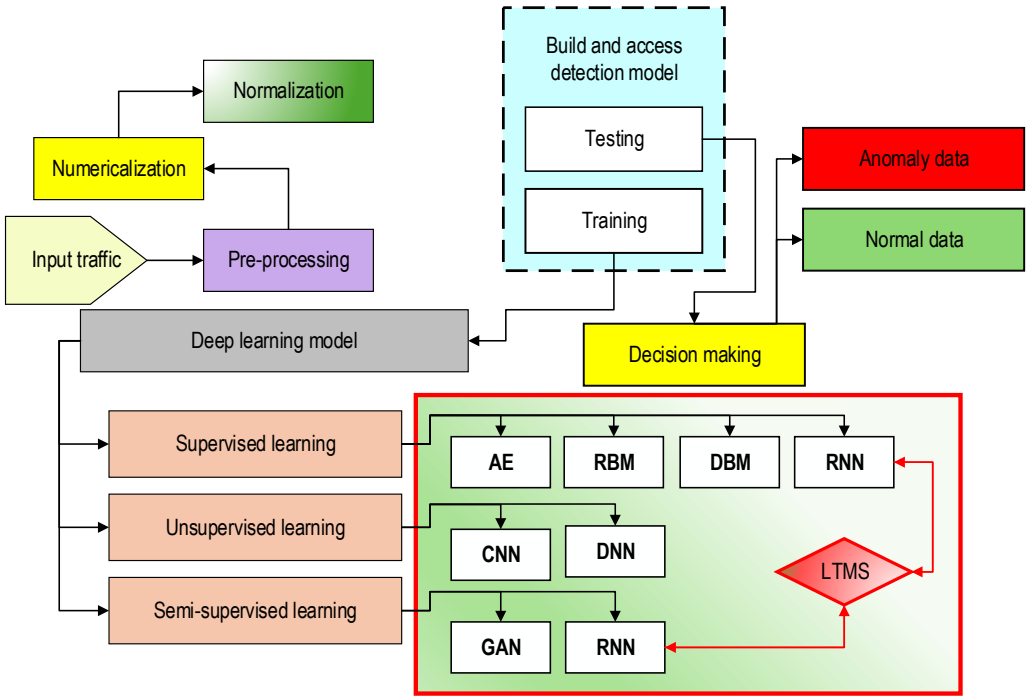

Fig. 1. Architecture of Deep Learning Based Intrusion Detection System

The use of deep learning techniques requires large and intensive computational resources during the training phase, involving many hidden layers with many elements, leading to high computational complexity. However, deep learning algorithms naturally involve large-scale matrix multiplication thanks to the development of modern processing technologies. In recent years, fast advancements in processing technologies enabled the improved availability of graphics processing units (GPUs) and artificial intelligence (AI) accelerators. Integration of these technologies into mobile devices and Internet of Things (IoT) devices has made it possible to deploy deep learning models on resource-constrained environments.

2. DEEP LEARNING MODELS

Deep learning models can be classified in to supervised learning, unsupervised learning and simi- supervised learning [13].

2.1. Supervised Learning

2.1.1. Autoencoder

An autoencoder is a specialized neural network architecture comprising two principal components the encoder and the decoder [14]. As illustrated in Figure 2, the encoder is responsible for extracting important features from the input data, while the decoder reconstructs the original data using these extracted features.

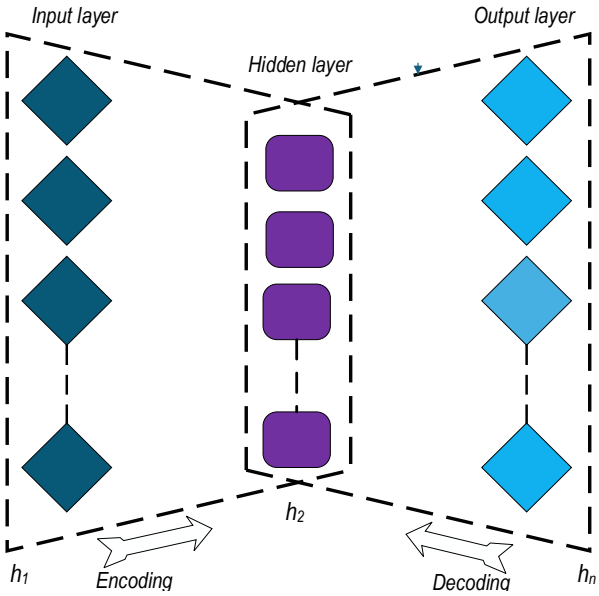

Fig. 2. Structural Model of the Auto-Encoder

This method is fundamentally similar to the traditional autoencoder framework [5, 15, 16], though it operates in a supervised manner. Throughout the training process, the gap between the encoder's input and the decoder's output narrows progressively, indicating that the features learned by the encoder accurately capture the significant information embedded in the original data, as evidenced by the decoder's ability to reconstruct the input data from these features. The encoder functions as a neural network with one or more hidden layers [14]. It transforms the input data, often noisy, into a compressed representation, or latent space, which contains fewer dimensions than the input data. This compression is achieved through the following equation:

$$h = f_{AE}(m) + W_{enc}M + b_{enc}\,, \qquad (1)$$

where m represents the input data; W_{enc} is the encoder's weight matrix; b_{enc} is the bias vector; h is the encoded representation. The function f_{AE} applies an activation function, such as ReLU or sigmoid, to the linear transformation of the input. The decoder is responsible for reconstructing the original data from the compressed encoding. It receives the encoded data h from the encoder and produces an approximation of the original input data.

The decoding process is mathematically defined as:

$$\hat{m} = g_{AE}(h) + W_{dec}h + b_{dec}, \qquad (2)$$

where $\hat{m}$ is the reconstructed input data; W_{dec} is the decoder's weight matrix; b_{dec} is the corresponding bias vector. The function g_{AE} applies an activation function to the decoder's output. The training objective is to minimize the loss function, which consists of several terms: the reconstruction error, weight regularization, and a sparsity constraint.

The reconstruction error is the mean squared difference between the original input data m_i and its reconstructed counterpart $\hat{m}_i$:

$$\mathcal{L}_{recon} = \frac{1}{N_{train}} \sum_{i=1}^{N_{train}} \sum_{j=1}^{N_{para}} (m_{ij} - \hat{m}_{ij})^2, \qquad (3)$$

where N_{train} is the number of training samples; N_{para} is the number of parameters (or features) per sample.

Additionally, weight regularization Ω_W is employed to prevent overfitting by penalizing excessively large weights:

$$\Omega_W = \frac{1}{2} \sum_{i=1}^{N_{train}} \sum_{j=1}^{N_{para}} w_{ij}^2. \qquad (4)$$

The sparsity constraint Ω_s encourages the model to activate only a small subset of the hidden units, thus promoting efficient learning. It is defined as:

$$\Omega_s = \sum_{k=1}^{N_{node}} \rho \log\left(\frac{\rho}{\hat{\rho}_k}\right) + (1-\rho)\log\left(\frac{1-\rho}{1-\hat{\rho}_k}\right), \qquad (5)$$

where ρ is the desired average activation for the sparsity constraint; $\hat{\rho}_k$ is the actual activation of the $k-th$ hidden unit.

The average activation $\hat{\rho}_k$ is computed as the mean activation across the training samples:

$$\hat{\rho}_k = \frac{1}{N_{train}} \sum_{i=1}^{N_{train}} k_h(m_i), \qquad (6)$$

where $k_h(m_i)$ is the activation function of the $k-th$ hidden unit for the $i-th$ training sample, the total loss function $\mathcal{L}$ to be minimized is the sum of the reconstruction error, weight regularization, and sparsity constraint:

$$\mathcal{L} = \mathcal{L}_{recon} + \lambda\Omega_W + \beta\Omega_s, \qquad (7)$$

where λ and β are regularization parameters that control the importance of the weight regularization and sparsity terms, respectively. By comparing the reconstruction error for new input data, the trained autoencoder widely used for anomaly detection (IDS).By comparing the reconstruction error for fresh input data m to a predetermined threshold, the trained autoencoder may be utilized for anomaly detection (IDS). The input is deemed abnormal if the reconstruction error is greater than this cutoff.

This is how the reconstruction error is calculated:

$$error = \|m - \hat{m}\|. \qquad (8)$$

This ability to reconstruct normal data patterns and detect deviations makes the autoencoder a good tool for identifying anomalous activities in IDS applications.

According to [14] Autoencoder can process two main categories of attack kinds are assault and regular. There are around 38 subclasses for the attacks. The class is transformed into a binary class for normal and attack because of numerous distinct attack kinds and feature sets. Evaluation metrics are utilized to evaluate this data. Next, we will adjust our dense autoencoder for this model. Figure 3 showing the evaluation metrics, ROC curve, and confusion matrix look like this with using NSL-KDD dataset.

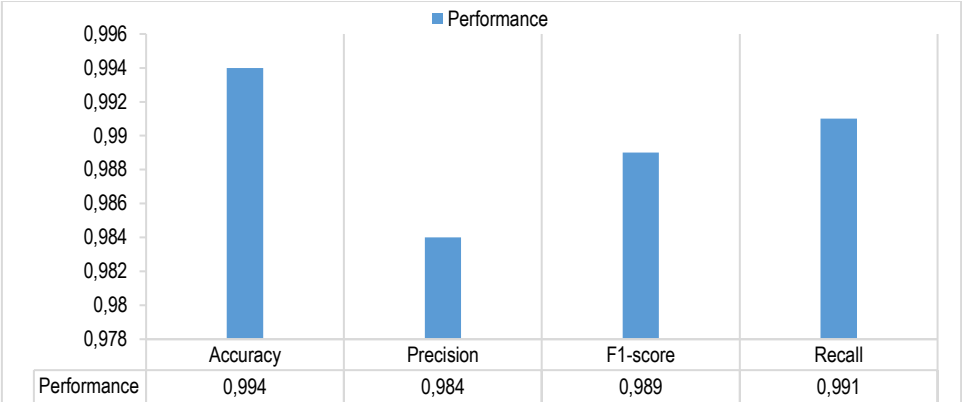

Fig. 3. Autoencoder Evaluation Metrics Using NSL-KDD Dataset [14]

2.1.2. Restricted Boltzmann Machine (RBM)

A Restricted Boltzmann Machine (RBM) is a stochastic neural network widely used in intrusion detection systems (IDS) for its ability to extract features from complex network data and detect anomalies [5]. It consists of a visible layer *V* that represents the input data and a hidden layer *H* that encodes latent features [17] Figure 4 showing Structural model of the RBM.

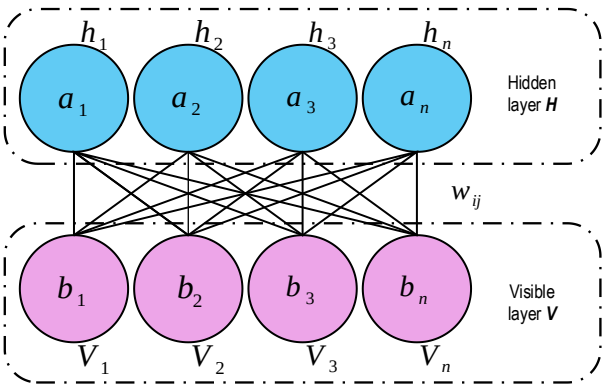

Fig. 4. Structural Model of the Restricted Boltzmann Machine (RBM)

The layers are fully connected to each other, but there are no connections within a single layer [17]. RBMs do not distinguish between forward and backward directions, making the weights symmetric. This property is critical for training RBMs effectively using contrastive divergence. The energy function of the RBM is defined as:

$$E(\varphi, h) = -\sum_{i-1}^{N} a_i \varphi_i - \sum_{j-1}^{M} b_j h_j - \sum_{i,j} \varphi_i h_j w_{ij}. \quad (9)$$

During training, the probability of activating a hidden unit given the visible units is computed as:

$$P(h_i) = \sigma\left(b_i + \sum_{i=1}^{n} \varphi_i w_{ij}\right), \quad (10)$$

where $\sigma(x)$ is the sigmoid activation function:

$$\sigma(x) = \frac{1}{1 + \exp(-x)}. \quad (11)$$

To reconstruct the visible layer, the conditional probability for each visible unit given the hidden units is calculated as:

$$P(\varphi_i) = \sigma\left(a_i + \sum_{j=1}^{m} h_j w_{ij}\right). \quad (12)$$

The RBM is trained by minimizing the difference between the data and its reconstruction [18].

Weight updates are calculated using the contrastive divergence algorithm:

$$\delta w_{ij} = \epsilon\left(\langle \varphi_i h_j \rangle_{data} - \langle \varphi_i h_j \rangle_{model}\right), \quad (13)$$

where ϵ is the learning rate; $\langle \varphi_i h_j \rangle_{data}$ represents the expectation over the training data; $\langle \varphi_i h_j \rangle_{model}$ represents the expectation over the model distribution.

Bias updates for the visible and hidden layers are defined as:

$$\delta a_i = \epsilon(\langle \varphi_i \rangle_{data} - \langle \varphi_i \rangle_{recon}), \quad (14)$$

$$\delta b_j = \epsilon\left(\langle h_j \rangle_{data} - \langle h_j \rangle_{recon}\right). \quad (15)$$

A sparsity constraint is often introduced to enforce meaningful feature extraction. The sparsity penalty is:

$$\Omega_s = \sum_{j=1}^{M} \left[\rho \log\left(\frac{\rho}{\hat{\rho}_j}\right) + (1-\rho) \log\left(\frac{1-\rho}{1-\hat{\rho}_j}\right)\right], \quad (16)$$

where the average activation of a hidden unit is:

$$\hat{\rho}_j = \frac{1}{N}\sum_{i=1}^{N} P(1|\varphi_i). \quad (17)$$

The reconstruction error, typically measured as the mean squared error (MSE), is minimized during training to enhance model performance:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M}(\varphi_{ij} - \varphi_{ij}, recon)^2, \quad (18)$$

these equations and principles allow the RBM to effectively model normal network behavior while identifying deviations that signify potential intrusions. According to [19] Experiment was performed by only adjusting the size of training data on the model set with the batch data of 10 and the learning rate of 0.01 showing the top performance in the above trial. Figure 5 showing Accuracy, Precision, Recall and F-measure with batch size = 10 and learning rate = 0.1 using NSL-KDD dataset.

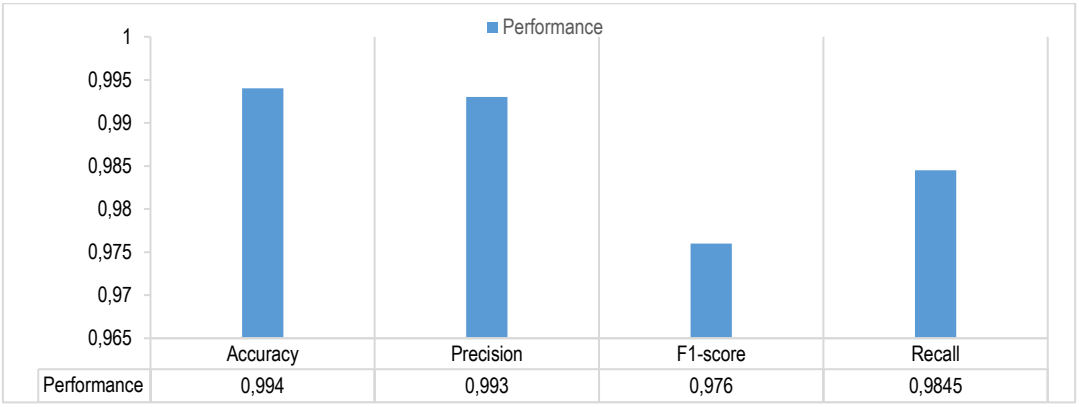

Fig. 5. RBM Evaluation Metrics Using NSL-KDD Dataset [19]

2.1.3. Deep Belief Networks (DBNs)

Deep Belief Networks (DBNs) are highly effective tools for Intrusion Detection Systems (IDS). A DBN is constructed by stacking multiple Restricted Boltzmann Machines (RBMs), where the output of one layer serves as the input to the next. As shown in Figure 6, the DBN architecture consists of an input layer, multiple hidden layers, and a softmax layer at the top for classification.

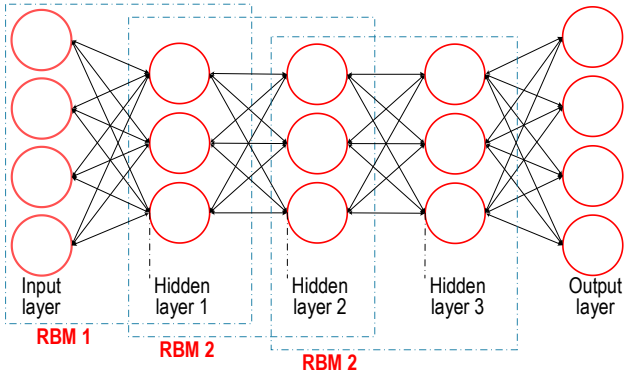

Fig. 6. Structural Model of the DBNs

The DBN is trained in a layer-wise manner using the following principles: First, in unsupervised pretraining, each RBM in the DBN is trained individually [3, 20]. The energy function of an RBM is defined as:

$$E(v,h) = -\sum_{i=1}^{N} a_i v_i - \sum_{j=1}^{M} b_j h_j - \sum_{i=1}^{N}\sum_{j=1}^{M} v_i h_j w_{ij}, \quad (19)$$

where v and h represent the visible and hidden layers, respectively a_i and b_j are their biases, and w_{ij} are the weights connecting the layers. During training, the probability of hidden units and visible units being activated is calculated as:

$$P(h_i) = \sigma\left(b_j + \sum_i v_i w_{ij}\right), \quad (20)$$

$$P(v_i) = \sigma\left(a_j + \sum_j h_j w_{ij}\right), \quad (21)$$

where $\sigma(x)$ is the sigmoid activation function:

$$\sigma(x) = \frac{1}{1+e^{-x}}. \quad (22)$$

After unsupervised pretraining, the entire DBN is fine-tuned using supervised learning, which involves minimizing a loss function, such as cross-entropy for classification tasks, and optimizing the weights and biases across all layers [3, 20]. For final classification in IDS, a softmax layer is added at the top of the DBN.

The probability of class k is given by:

$$P(y,k|x) = \frac{\exp(z_k)}{\sum_{j=1}^{k}\exp(z_j)}, \quad (23)$$

where z_k represents the activation of the $k-th$ neuron in the softmax layer. This hierarchical approach allows DBNs to extract meaningful features from raw

data and enhance detection performance, making them well-suited for IDS applications. By combining unsupervised feature learning and supervised classification, DBNs improve the ability to detect and classify intrusions effectively.

According to [21] DBNs integrate feature extraction and classification modules into a system that can automatically extract features and classify them. This is an effective way to improve the detection performance. Figure 7 showing the classification performance of DBN was evaluated on the NSL-KDD.

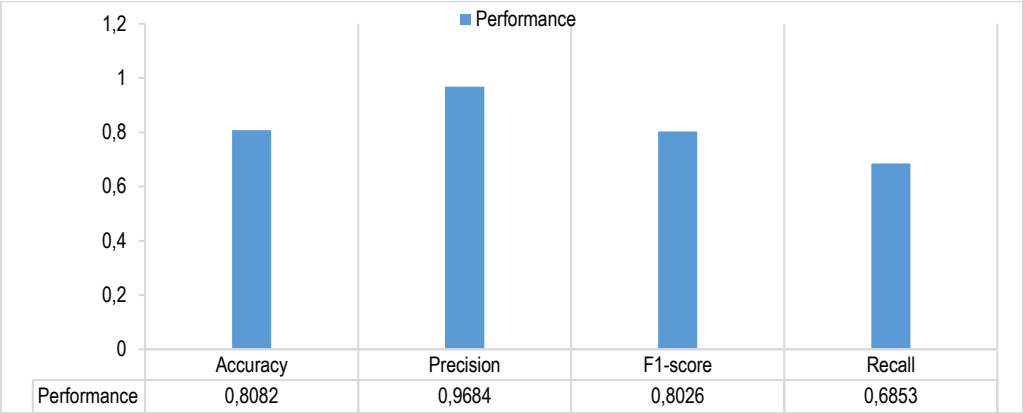

Fig. 7. DBM Evaluation Metrics Using NSL-KDD Dataset [21]

2.1.4. Recurrent Neural Networks (RNN)

RNN Conventional systems frequently depended on streamlined encoding techniques, including one-hot encoding [22]. Nevertheless, one-hot encoding's incapacity to accurately capture semantic similarities between features is a major drawback. Researchers are using LSTM networks to tackle this problem [23]. Long-term selective retention of pertinent information is made easier by LSTM networks, which incorporate memory cells and control mechanisms to manage information flow. Additionally, because RNN-LSTM can capture long-term dependencies, the model can identify subtle deviations that may be signs of intrusions the network is recurrent sequential analysis is made possible, which is crucial for comprehending the temporal dynamics of network behavior [23]. The architecture allows IDS-RNN model to adaptively learn from historical data, improving detection accuracy over time, IDS-RNN model is fully capable of detecting intrusions in dynamic network environments. In RNN architecture hidden layers have a simple structure (e.g. single tanh layer), while the LSTM architecture is more complex, It is constituted of 4 hidden layers (Figure 8) [24] showing LTMS unit and RNN unit.

To add or remove information from the cell state, the gates are used to protect it, using sigmoid function (one means allows the modification, while a value of zero means denies the modification).

We can identify three different gates.

1) Input/Update gate layer (Figure 8): which controls how information enters the memory cell, is the key component of the LSTM. The computation of the input gate involves the current input x and the previous hidden state h_{t-1}:

$$\tilde{c}_t = \tanh(W_c x_t + U_c h_{t-1} + b_c), \qquad (24)$$

where W_c and U_c are weight matrices; b_c represents the bias parameter. The function, tanh denotes the hyperbolic tangent activation function.

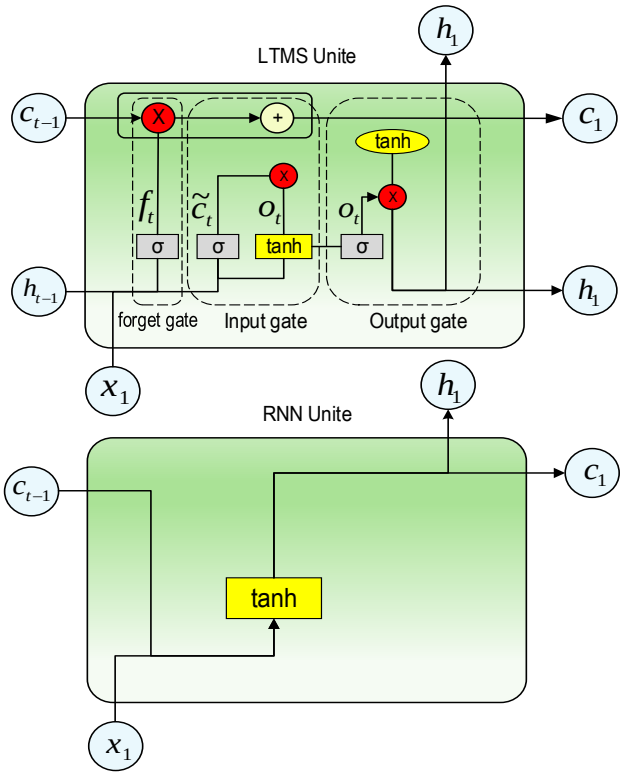

Fig. 8. LTMS Unit and RNN Unit

2) Forget gate layer (Figure 8): The forget gate is a crucial part that works in tandem with the input gate to filter out old data from the memory cell. This gate is controlled by the sigmoid activation function, and its

operation at time t is represented by the following equation:

$$f_t = \sigma\left(W_f x_t + U_f h_{t-1} + b_f\right). \quad (25)$$

3) Output gate Layer (Figure 8): The output gate serves as the last arbiter in the intrusion detection system (IDS) its decides what will be our output by executing a sigmoid function that decides which part of the cell LSTM is going to output, the result is passed through a tanh layer (value between – 1 and 1) to output only the information we decide to pass to the next neuron. The output gate helps the RNN-LSTM-based IDS to deliver actionable information to security analysts or supporting systems by selectively broadcasting pertinent signals, allowing prompt reactions to abnormalities and possible security breaches [25].

Mathematically, the output gate o_t at the time t is represented by the equation:

$$O_t = \sigma\left(W_o x_t + U_o h_{t-1} + \tilde{C}_t\right), \quad (26)$$

according to [24] the performance model of RNN, a series of experiments were conducted using varying hidden layer sizes. The highest accuracy was achieved with a hidden layer size of 100, and based on this, the hyperparameters were set for further model training. The model was implemented on an intrusion detection system (IDS) using the NSL-KDD dataset Figure 9 showing the classification performance of RNN was evaluated on the NSL-KDD dataset.

2.2. Unsupervised Learning

2.2.1. Deep Neural Network (DNN)

A Deep Neural Network (DNN) is a kind of neural network that has many layers which are set up in a feedforward topology [26]. It is made up of the input layer, several hidden layers, and the output layer. DNN operates in a unidirectional manner unlike recurrent neural networks, where data will move from the input nodes, through the hidden nodes to the output nodes, in the edge-weighted DAG without any loops or cycles. Each neuron in a layer is connected directly with the neurons in the subsequent layers. In the training process, the network's weights are adjusted through backpropagation method [27] Figure 10 illustrates the structure of a DNN.

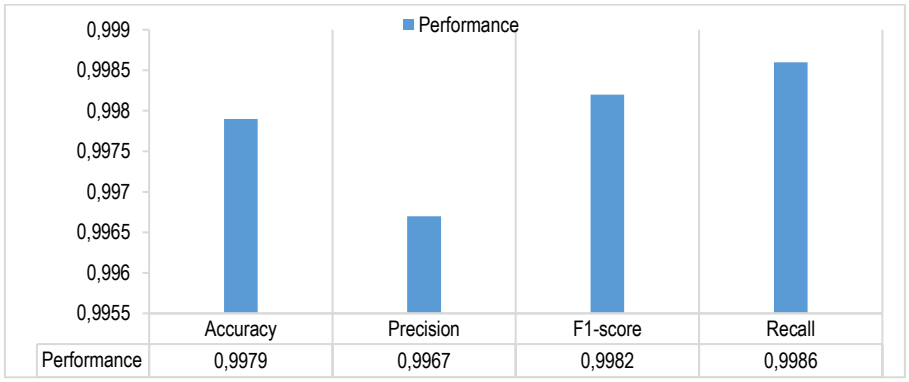

Fig. 9. RNN Evaluation Metrics Using NSL-KDD Dataset [24]

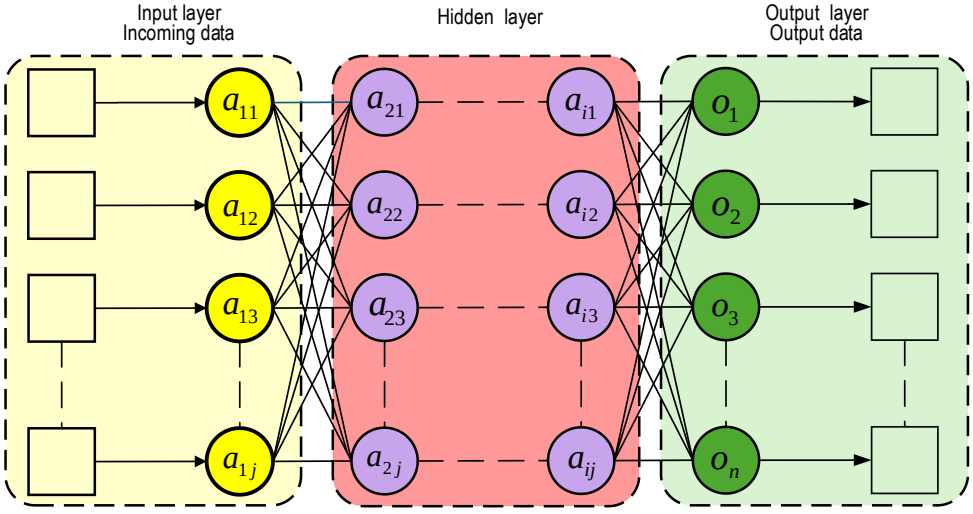

Fig. 10. The Structure of DNN

DNN famous for two main reasons [24, 27]:

1) they deliver a close to 100% accurate approximation of complex multidimensional and nonlinear functions built directly from the input data;

2) they are the foundation of the strong approximation theory which is adapted to a much wider list of natural and artificial occurring phenomena.

The training process of a DNN involves minimizing a cost function that measures the discrepancy between the network's predicted output and the actual output.

The cost function, denoted as $C(\vec{w}, \vec{x}, y)$ is typically defined as the mean squared error between the predicted output $h_w(\vec{x})$ and the true output y:

$$C(\vec{w}, \vec{x}, y) = \frac{1}{2}\|h_w(\vec{x}) - y\|^2, \quad (27)$$

where w represents the weights of the network $\vec{x}$ is the input vector to regularize the model and prevent overfitting, a regularization term is added to the cost function.

The regularized cost function $C(w\vec{x})$ is expressed as follows:

$$C(\vec{w}) = \frac{1}{N}C(\vec{w}; \vec{x}^n, y^n) + \frac{\lambda}{2}\sum_{K}^{k}\sum_{i}^{L_m}\sum_{j}^{L_{m+1}}\left(w_{ij}^k\right)^2, \quad (28)$$

where N represents the number of data points in the training set; λ is the regularization parameter; w_{ij}^k denotes the weights between the neurons in layers m and $m+1$ at the $k-th$ layer. During the training process, the weights of the network are updated iteratively using gradient descent.

The weight update rule is given by:

$$w_{ij}^k = w_{ij}^{k-1} + \xi\frac{\partial}{\partial w_{ij}^{k-1}}C(\vec{w}), \quad (29)$$

where ξ is the learning rate, and the update is based on the gradient of the cost function with respect to the weights, which guides the network towards minimizing the cost.

According to [28], the proposed model uses DNN to classify network traffic as either normal or attack. Due to the complexity of attack types, the model simplifies detection by grouping attacks into a binary classification. Evaluation metrics, including the ROC curve and confusion matrix shown in Figure 11, are used to assess performance on the NSL-KDD dataset. The DNN employs ReLU activation in hidden layers and Softmax activation in the output layer, ensuring efficient and accurate intrusion detection.

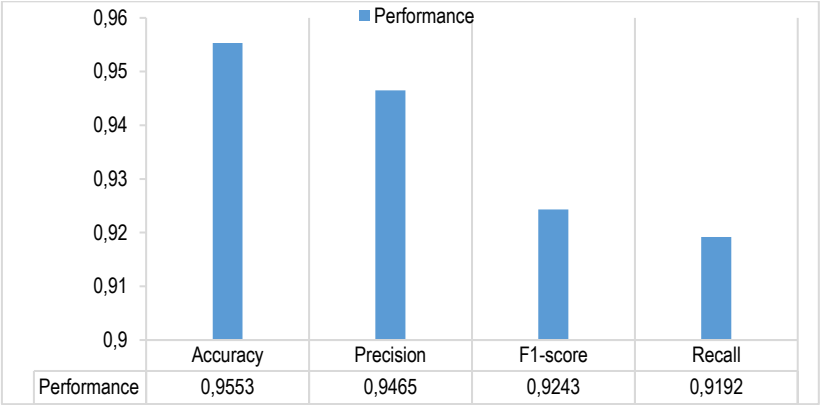

Fig. 11. DNN Evaluation Metrics Using NSL-KDD Dataset [28]

2.2.2 Convolutional Neural Networks (CNNs)

Convolutional Neural Networks (CNNs) are powerful systems that mimic the characteristics of the human visual system (HVS) and have led to significant breakthroughs in the field of computer vision [29, 30]. CNNs have become some of the most advanced algorithms in image recognition, object detection, and classification tasks, drawing inspiration from natural vision mechanisms. A typical CNN architecture, as shown in Figure 12, consists of multiple layers arranged in a sequence. These layers include convolutional layers, pooling layers, and fully connected layers.

The convolutional layers are the first to process the input data, where they apply a set of filters to extract fundamental features such as patterns and spatial relationships. These filters create feature maps that highlight important areas of the image, allowing the network to focus on key patterns [31].

The convolution operation itself is expressed as:

$$F(X, W) = Y, \quad (30)$$

where X is the input data; W represents the convolutional filter weights; Y is the output feature map. In a convolutional layer, the activation of a feature map at position (i, j) is calculated using the following formula:

$$a_{ij} = \sigma\left((W * X)_{ij} + b\right), \quad (31)$$

where $a_{ij}$is the activated value at position (i, j); σ is the activation function (such as ReLU), and b is the bias term.

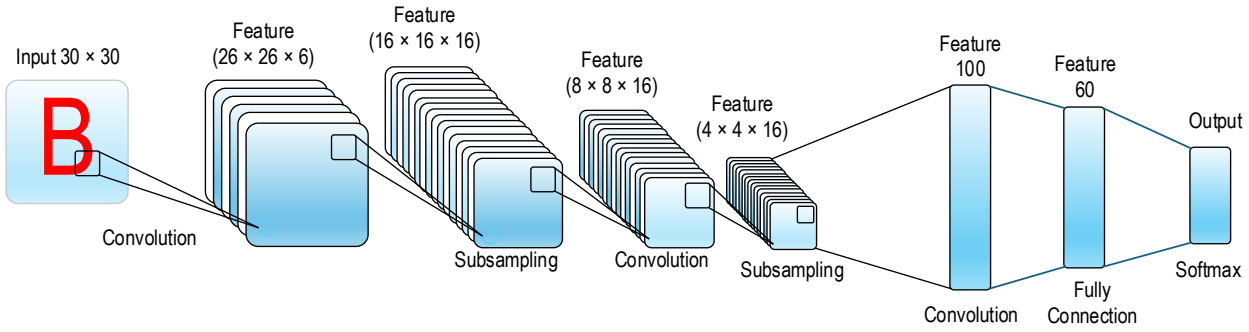

Fig. 12. CNN Architecture

The output of the convolution operation for a feature map C_q^l in layer l is computed by applying the filter $k_{p,q}^l$ to the input feature map S_p^{l-1} from the previous layer, along with a bias term b_q^l:

$$C_q^l = \left(\sum_{p=1}^{n} S_p^{l-1} * k_{p,q}^l + b_q^l\right). \quad (32)$$

To handle spatial shifts in the input data, this convolution operation can be extended as:

$$C_q^l = \left(\sum_{p=1}^{n}\sum_{u=-x}^{x}\sum_{v=-x}^{x} S_p^{l-1}(i-u, j-v) \times k_{p,q}^l(u,v) + b_q^l\right). \quad (33)$$

After applying the convolution operation, pooling layers are used to reduce the spatial dimensions of the feature maps. For example, the output $S_q^l(i,j)$ of a pooling layer can be computed as:

$$S_q^l(i,j) = \frac{1}{4}\sum_{u=0}^{z}\sum_{v=0}^{z} C_q^l(2i-u, 2j-v). \quad (34)$$

Once the convolutional and pooling layers have processed the input, the network typically uses fully connected layers for the final prediction. The output $\hat{y}(i)$ for each class is obtained using the softmax function, which normalizes the raw output scores:

$$\hat{y}(i) = \frac{e^{\widehat{output}}}{\sum_1^{labels} e^{\widehat{output}}}. \quad (35)$$

The objective during training is to minimize the loss function, which measures the difference between the predicted output $\hat{y}(i)$ and the true output $y(i)$.

The loss function is often represented as:

$$L = \frac{1}{2}\sum_{i=1}^{training\ patters}\left(\hat{y}(i) - y(i)\right)^2. \quad (36)$$

The weights of the network are updated using the gradient of the loss with respect to each weight:

$$\Delta W(i,j) = \frac{\partial L}{\partial W(i,j)} = \frac{\partial L}{\partial \hat{y}}.\frac{\partial \hat{y}}{\partial w(i,j)}. \quad (37)$$

Expanding the derivative:

$$\Delta W(i,j) = \left(\hat{y}(i) - y(i)\right).\frac{\partial L}{\partial W(i,j)} \times \times \left(\sigma\left(\sum_{j=1}^{nodes} W(i,j) * f(j) + b(i)\right)\right). \quad (38)$$

Additionally, the derivative of the output $\hat{y}(i)$ with respect to the loss is:

$$\Delta\hat{y}(i) = \left(\hat{y}(i) - y(i)\right).\hat{y}(i)\left(1 - \hat{y}(i)\right). \quad (39)$$

These gradients are used to update the weights through an optimization technique like stochastic gradient descent (SGD).

According to [32] the experimental results on the NSL-KDD dataset showing in Figure 13.

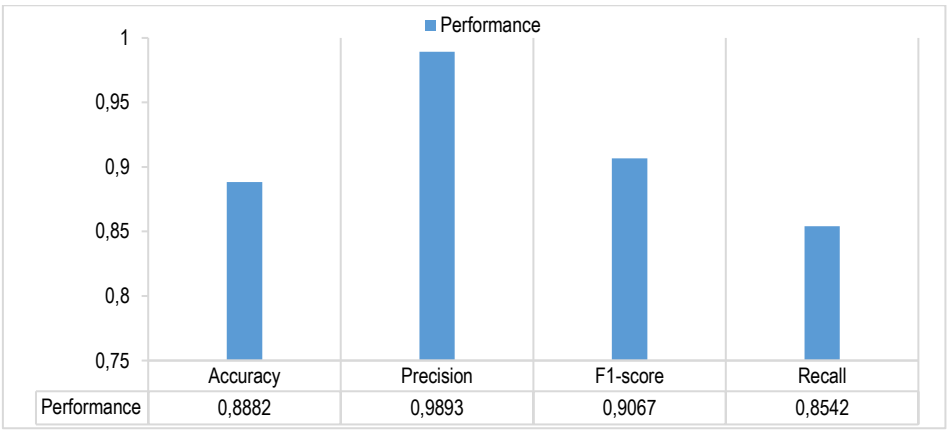

Fig. 13. CNN Evaluation Metrics Using NSL-KDD Dataset [32]

2.3. Sim-Supervised Learning

2.3.1. Generative Adversarial Networks (GANs)

Generative Adversarial Networks (GANs) have gained significant attention in various application domains, with anomaly detection being one of the most prominent [33]. The ability of GANs to learn complex representations and generate data that appears realistic makes them highly effective for anomaly detection tasks. These networks are trained using a data distribution approach in an unsupervised manner, allowing them to detect irregularities without requiring labeled datasets [33], Figure 14 showing GANs architecture.

Mathematically, the operation of GANs can be described using the divergence function:

$$D_f\left(P_{data}\|P_g\right) = \int_x P_g(x) f\left(\frac{P_{data}(x)}{P_g(x)}\right) dx, \qquad (40)$$

where $f(1) = 0$ indicates that P_g and P_{data} are equivalent. In the origin of GAN, the Jensen-Shannon JS divergence was adopted as f:

$$f(t) = t\log(t) - (t+1)\log(t+1). \qquad (41)$$

This function measures the discrepancy between the real data distribution P_{data} and the generator's distribution P_g quantifying how closely the generated data approximates the real data [34].

Another crucial metric in GANs is the Jensen-Shannon Divergence JS defined as:

$$JS\left(P_{data}\|P_g\right) = \frac{1}{2}KL(P_{data}\|P_m) + \frac{1}{2}(P_s\|P_m), \qquad (42)$$

where:

$$P_m = \frac{1}{2}\left(P_{data}\|P_g\right). \qquad (43)$$

The JS divergence evaluates the similarity between P_{data} and P_g with lower values indicating a better match. This property makes GANs particularly well-suited for distinguishing anomalies, as deviations from the expected distribution can be easily detected.

The core of GAN functionality lies in the minimax optimization framework (44), where $D(x)$ is the discriminator function (45), which assigns a higher probability to real samples. The generator seeks to minimize the term $\log\left(1 - D\left(G(z)\right)\right)$ improving its ability to produce realistic data. Further refinement involves the optimization objectives for the discriminator (46) and generator (47).

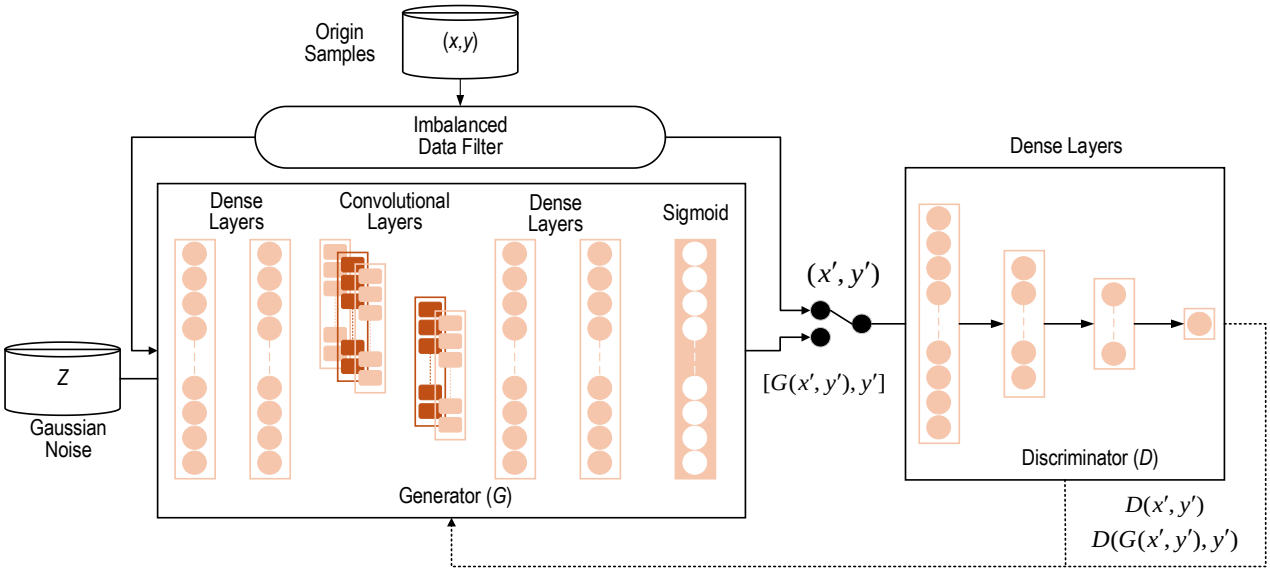

Fig. 14. GANs Architecture

$$\min_G \max_D V(D,G) = \min_G \max_D \begin{pmatrix} \mathbb{E}_{x\sim P_{data(x)}}[\log D(x)] \\ +\mathbb{E}_{z\sim P_{data(z)}} \\ \left[\log\left(1 - D\left(G(z)\right)\right)\right] \end{pmatrix}. \qquad (44)$$

$$D(x) = \frac{P_{data}(x)}{P_{data}(x) + P_g(x)}. \qquad (45)$$

$$\max_{\theta_D} \hat{V}(D) = \max_{\theta_D} \frac{1}{m}\sum_{i=1}^{m} \begin{pmatrix} \log D(\acute{x}_i, \acute{y}_i) + \\ \log\left(1 - D(G(z_i, \acute{y}_i)\acute{y}_i)\right) \end{pmatrix}. \qquad (46)$$

$$\min_{\theta_G} \hat{V}(D) = \max_{\theta_G} \frac{1}{m} \sum_{i=1}^{m} \left(\log\left(1 - D(G(z_i, \acute{y}_i)\acute{y}_i)\right)\right). \quad (47)$$

The probabilistic classification in GANs can be modeled as:

$$P(y|a) = P(y = C_\tau|v) = \frac{e^{v_\tau}}{\sum_{v_{\tau\in v}} e^{v_\tau}}, \quad (48)$$

where the probability of a sample belonging to a particular class is determined by the class scores. These mathematical foundations enable GANs to perform effectively in anomaly detection, making them invaluable for applications like fraud detection, network intrusion detection, and industrial fault diagnosis. According to [33] showing the evaluation of the metrics GAN using NSL-KDD (Figure 15).

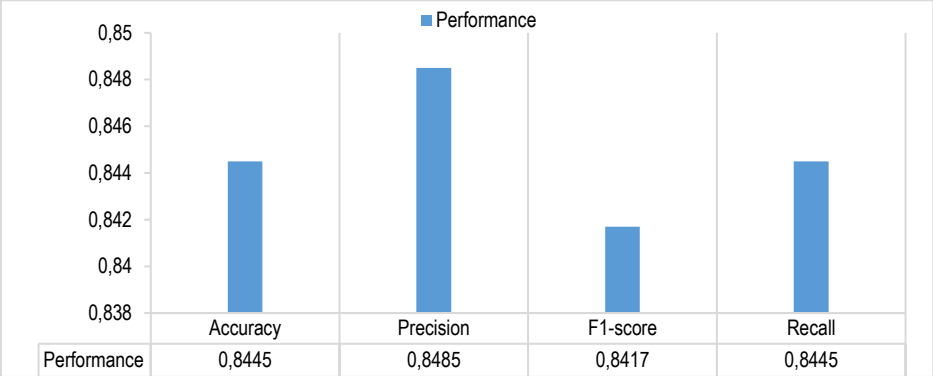

Fig. 15. GAN Evaluation Metrics Using NSL-KDD Dataset [33]

3. RESULTS AND DISCUSSIONS

In order to evaluate the deep learning models discussed, this study utilized the NSL-KDD dataset, which comprises 41 features and 125,973 instances categorized into normal and four attack types: DoS, Probe, R2L, and U2R. Various deep learning models – Autoencoders, Restricted Boltzmann Machines (RBMs), Deep Belief Networks (DBNs), Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), Deep Neural Networks (DNNs), and Generative Adversarial Networks (GANs – were implemented and tested to assess their effectiveness in intrusion detection. The results showed varying performance across the models based on evaluation metrics, as seeing in Figures (3, 5, 7, 9, 11, 13, 15).

Among the tested models, RNNs achieved the highest accuracy (99.79 %), precision (99.67 %), and recall (99.86 %), demonstrating their effectiveness in detecting temporal patterns in network traffic. Autoencoders and RBMs also performed well, with accuracy metrics of 99.4 %. However, DBNs exhibited lower accuracy (80.82 %) and recall (68.53 %), suggesting limitations in handling complex datasets. CNNs and GANs showed promising results but lagged behind RNNs in overall performance metrics. DNNs provided a balanced performance, achieving high accuracy (95.53 %) and precision (94.65 %).Figures illustrating these results emphasize the superiority of RNNs and the suitability of Autoencoders and RBMs for intrusion detection tasks. The findings also align with prior studies, underscoring the importance of selecting appropriate models based on the specific requirements of IDS applications and the characteristics of available datasets.

4 CONCLUSIONS

The findings of this study emphasize the transformative impact of deep learning on intrusion detection systems. By leveraging advanced neural network architectures, IDS can achieve superior performance in detecting and responding to cyber threats. Models like autoencoders and RBMs enable efficient anomaly detection through automatic feature learning, while DBNs, RNNs, and CNNs provide enhanced capabilities for processing temporal and spatial data patterns. The integration of these models has demonstrated significant improvements in accuracy, precision, and recall metrics, as evidenced by evaluations on the NSL-KDD dataset. While computational complexity remains a challenge, recent advancements in hardware technologies, such as GPUs and AI accelerators, have made it feasible to implement deep learning-based IDS in resource-constrained environments. Future research should focus on improving model scalability, reducing computational demands, and adapting to the ever-changing landscape of network security threats. These advancements will be crucial for developing IDS that are both effective and practical across diverse applications.

Список источников

1. Navya V.K., Adithi J., Rudrawal D., Tailor H., James N. Intrusion Detection System Using Deep Neural Networks (DNN) // Proceedings of the International Conference on Advancements in Electrical, Electronics, Communication, Computing and Automation (ICAECA, Coimbatore, India, 08–09 October 2021). IEEE, 2022. DOI:10.1109/ICAECA52838.2021.9675513

2. Vinayakumar R., Soman K.P. Poornachandran P. Applying convolutional neural network for network intrusion detection // Proceedings of the International Conference on Advances in Computing, Communications and Informatics (ICACCI, Udupi, India, 13–16 September 2017). IEEE, 2017. PP. 1222–1228. DOI:10.1109/ICACCI.2017.8126009

3. Wu Y., Lee W.W., Xu Z., Ni M. Large-scale and robust intrusion detection model combining improved deep belief network with feature-weighted SVM // IEEE Access. 2020. Vol. 8. PP. 98600–98611. DOI:10.1109/ACCESS.2020.2994947. EDN:APJZGY

4. Alpaydin E. Introduction to Machine Learning. MIT Press, 2020.

5. Aldwairi T., Perera D., Novotny M.A. An evaluation of the performance of Restricted Boltzmann Machines as a model for anomaly network intrusion detection // Computer Networks. 2018. Vol. 144. PP. 111–119. DOI:10.1016/j.comnet.2018.07.025

6. Choi H., Kim M., Lee G., Kim W. Unsupervised learning approach for network intrusion detection system using autoencoders // The Journal of Supercomputing. 2019. Vol. 75. Iss. 9. PP. 5597–5621. DOI:10.1007/s11227-019-02805-w. EDN:RJBUQU

7. Shahriar M.H., Haque N.I., Rahman M.A., Alonso M. G-ids: Generative adversarial networks assisted intrusion detection system // Proceedings of the 44th Annual Computers, Software, and Applications Conference (COMPSAC, Virtual, Madrid, 13–17 July 2020). IEEE, 2020. PP. 376–385. DOI:10.1109/COMPSAC48688.2020.0-218. EDN:DJVEFI

8. Al-Qatf M., Lasheng Y., Al-Habib M., Al-Sabahi K. Deep Learning Approach Combining Sparse Autoencoder With SVM for Network Intrusion Detection // IEEE Access. 2018. Vol. 6. PP. 52843–52856. DOI:10.1109/ACCESS.2018.2869577

9. Shone N., Ngoc T.N., Phai V.D., Shi Q. A Deep Learning Approach to Network Intrusion Detection // IEEE Transactions on Emerging Topics in Computational Intelligence. 2018. Vol. 2. Iss. 1. PP. 41–50. DOI:10.1109/TETCI.2017.2772792

10. Li Z., Qin Z., Huang K., Yang X., Ye S. Intrusion Detection Using Convolutional Neural Networks for Representation Learning // Proceedings of the 24th International Conference on Neural Information Processing (ICONIP, Guangzhou, China, 14–18 November 2017). Lecture Notes in Computer Science. Cham: Springer, 2017. Vol. 10638. PP. 858–866. DOI:10.1007/978-3-319-70139-4_87

11. Wang S., Wang J., Lu H., Zhao W. A novel combined model for wind speed prediction–Combination of linear model, shallow neural networks, and deep learning approaches // Energy. 2021. Vol. 234. P.121275. DOI:10.1016/j.energy.2021.121275. EDN:FAJRCK

12. Javaid A., Niyaz Q., Sun W., Alam M. A Deep Learning Approach for Network Intrusion Detection System // Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS, New York, USA, 3–5 December 2015). 2016. PP. 21–26. DOI:10.4108/eai.3-12-2015.2262516

13. Manimaran A., Chandramohan D., Shrinivas S.G., Arulkumar N. A comprehensive novel model for network speech anomaly detection system using deep learning approach // International Journal of Speech Technology. 2020. Vol. 23. Iss. 2. PP. 305–313. DOI:10.1007/s10772-020-09693-z. EDN:URAWUT

14. Alrayes F.S., Zakariah M., Amin S.U., Khan Z.I., Helal M. Intrusion Detection in IoT Systems Using Denoising Autoencoder // IEEE Access. 2024. Vol. 12. PP. 122401–122425. DOI:10.1109/ACCESS.2024.3451726. EDN:VCPDLT

15. Vincent P., Larochelle H., Bengio Y., Manzagol P.A. Extracting and composing robust features with denoising autoencoders // Proceedings of the 25th International Conference on Machine Learning (Helsinki, Finland, 5–9 July 2008). Association for Computing Machinery, 2008. PP. 1096–1103. DOI:10.1145/1390156.1390294

16. Vincent P., Larochelle H., Lajoie I., Bengio Y., Manzagol P.A., Bottou L. Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion // Journal of Machine Learning Research. 2010. Vol. 11. PP. 3371–3408. EDN:OCLKDJ

17. Zhang N., Ding S., Zhang J., Xue Y. An overview on restricted Boltzmann machines // Neurocomputing. 2018. Vol. 275. PP. 1186–1199. DOI:10.1016/j.neucom.2017.09.065

18. Mayuranathan M., Murugan M., Dhanakoti V. Retracted article: best features based intrusion detection system by RBM model for detecting DDoS in cloud environment // Journal of Ambient Intelligence and Humanized Computing. 2021. Vol. 12. Iss. 3. PP. 3609–3619. DOI:10.1007/s12652-019-01611-9. EDN:LAAOLK

19. Seo S., Park S., Kim J. Improvement of Network Intrusion Detection Accuracy by Using Restricted Boltzmann Machine // Proceedings of the 8th International Conference on Computational Intelligence and Communication Networks (CICN, Tehri, India, 23–25 December 2016). IEEE, 2016. PP. 413–417. DOI:10.1109/CICN.2016.87

20. Balakrishnan N., Rajendran A., Pelusi D., Ponnusamy V. Deep Belief Network Enhanced Intrusion Detection System to Prevent Security Breach in the Internet of Things // Internet of Things. 2021. Vol. 14. P. 100112. DOI:10.1016/j.iot.2019.100112. EDN:CZRBGW

21. Yang Y., Zheng K., Wu C., Niu X., Yang Y. Building an effective intrusion detection system using the modified density peak clustering algorithm and deep belief networks // Applied Sciences. 2019. Vol. 9. Iss. 2. P. 238. DOI:10.3390/app9020238. EDN:ABGPER

22. Parhizkari S. Anomaly Detection in Intrusion Detection Systems. 2023. DOI:10.5772/intechopen.112733

23. Mehibs S.M., Hashim S.H. Proposed Network Intrusion Detection System in Cloud Environment Based on Back Propagation Neural Network // Journal of University of Babylon for Pure and Applied Sciences. 2018. Vol. 26. Iss. 1. PP. 29–40. DOI:10.29196/jub. v26i1.351

24. Al-Tameemi M.M.A., Alzaghir A.A.H. Improving Network Security Through Deep Learning RNN Approach // Computational Nanotechnology. 2024. Vol. 11. Iss. 4. PP. 114–121. DOI:10.33693/2313-223X-2024-11-4-114-121. EDN:GPCZUD

25. Smagulova K., James A.P. A survey on LSTM memristive neural network architectures and applications // The European Physical Journal Special Topics. 2019. Vol. 228. Iss. 10. PP. 2313–2324. DOI:10.1140/epjst/e2019-900046-x. EDN:HRKIKB

26. Han K., Yu D., Tashev I. Speech emotion recognition using deep neural network and extreme learning machine // Interspeech 2014. DOI:10.21437/Interspeech.2014-57

27. Roy S.S., Mallik A., Gulati R., Obaidat M.S., Krishna P.V. A deep learning based artificial neural network approach for intrusion detection // Proceedings of the Third International Conference on Mathematics and Computing (ICMC 2017, Haldia, India, 17–21 January 2017). Communications in Computer and Information Science. Singapore: Springer, 2017. Vol. 655. PP. 44–53. DOI:10.1007/978-981-10-4642-1_5

28. Gowdhaman V., Dhanapal R. An intrusion detection system for wireless sensor networks using deep neural network // Soft Computing. 2022. Vol. 26. Iss. 23. PP. 13059–13067. DOI:10.1007/s00500-021-06473-y. EDN:KHFOPY

29. Yang Y., Zheng K., Wu C., Niu X., Yang Y. Building an Effective Intrusion Detection System Using the Modified Density Peak Clustering Algorithm and Deep Belief Networks // Applied Sciences. 2019. Vol. 9. Iss. 2. P. 238. DOI:10.3390/app9020238. EDN:ABGPER

30. Mohammadpour L., Ling T.Ch., Liew Ch.S., Aryanfar A. A Survey of CNN-Based Network Intrusion Detection // Applied Sciences. 2022. Vol. 12. Iss. 16. P. 8162. DOI:10.3390/app12168162. EDN:EFJJTR

31. Razavian A.S., Azizpour H., Sullivan J., Carlsson S. CNN features off-the-Shelf: An Astounding Baseline for Recognition // arXiv. 2014. DOI:10.48550/arXiv.1403.6382

32. Jo W., Kim S., Lee C., Shon T. Packet preprocessing in CNN-based network intrusion detection system // Electronics. 2020. Vol. 9. Iss. 7. P. 1151. DOI:10.3390/electronics9071151. EDN:XVUGFM

33. Sabuhi M., Zhou M., Bezemer C.P., Musilek P. Applications of Generative Adversarial Networks in Anomaly Detection: A Systematic Literature Review // IEEE Access. 2021. Vol. 9. PP. 161003–161029. DOI:10.1109/ACCESS.2021.3131949. EDN:TJDTSD

34. Dunmore A., Jang-Jaccard J., Sabrina F., Kwak J. A Comprehensive Survey of Generative Adversarial Networks (GANs) in Cybersecurity Intrusion Detection // IEEE Access. 2023. Vol. 11. PP. 76071–76094. DOI:10.1109/ACCESS.2023.3296707. EDN:NOKCVG

References

1. Navya V.K., Adithi J., Rudrawal D., Tailor H., James N. Intrusion Detection System Using Deep Neural Networks (DNN). *Proceedings of the International Conference on Advancements in Electrical, Electronics, Communication, Computing and Automation, ICAECA, 08–09 October 2021, Coimbatore, India*. IEEE; 2022. DOI:10.1109/ICAECA52838.2021.9675513

2. Vinayakumar R., Soman K.P. Poornachandran P. Applying convolutional neural network for network intrusion detection. *Proceedings of the International Conference on Advances in Computing, Communications and Informatics, ICACCI, 13–16 September 2017, Udupi, India*. IEEE; 2017. p.1222–1228. DOI:10.1109/ICACCI.2017.8126009

3. Wu Y., Lee W.W., Xu Z., Ni M. Large-scale and robust intrusion detection model combining improved deep belief network with feature-weighted SVM. *IEEE Access*. 2020;8:98600–98611. DOI:10.1109/ACCESS.2020.2994947. EDN:APJZGY

4. Alpaydin E. *Introduction to Machine Learning*. MIT Press; 2020.

5. Aldwairi T., Perera D., Novotny M.A. An evaluation of the performance of Restricted Boltzmann Machines as a model for anomaly network intrusion detection. *Computer Networks*. 2018;144:111–119. DOI:10.1016/j.comnet.2018.07.025

6. Choi H., Kim M., Lee G., Kim W. Unsupervised learning approach for network intrusion detection system using autoencoders. *The Journal of Supercomputing*. 2019;75(9):5597–5621. DOI:10.1007/s11227-019-02805-w. EDN:RJBUQU

7. Shahriar M.H., Haque N.I., Rahman M.A., Alonso M. G-ids: Generative adversarial networks assisted intrusion detection system. *Proceedings of the 44th Annual Computers, Software, and Applications Conference, COMPSAC, 13–17 July 2020, Virtual, Madrid*. IEEE; 2020. p.376–385. DOI:10.1109/COMPSAC48688.2020.0-218. EDN:DJVEFI

8. Al-Qatf M., Lasheng Y., Al-Habib M., Al-Sabahi K. Deep Learning Approach Combining Sparse Autoencoder With SVM for Network Intrusion Detection. *IEEE Access*. 2018;6:52843–52856. DOI:10.1109/ACCESS.2018.2869577

9. Shone N., Ngoc T.N., Phai V.D., Shi Q. A Deep Learning Approach to Network Intrusion Detection. *IEEE Transactions on Emerging Topics in Computational Intelligence*. 2018;2(1):41–50. DOI:10.1109/TETCI.2017.2772792

10. Li Z., Qin Z., Huang K., Yang X., Ye S. Intrusion Detection Using Convolutional Neural Networks for Representation Learning. *Proceedings of the 24th International Conference on Neural Information Processing, ICONIP, 14–18 November 2017, Guangzhou, China. Lecture Notes in Computer Science, vol.10638*. Cham: Springer; 2017. p.858–866. DOI:10.1007/978-3-319-70139-4_87

11. Wang S., Wang J., Lu H., Zhao W. A novel combined model for wind speed prediction–Combination of linear model, shallow neural networks, and deep learning approaches. *Energy*. 2021;234:121275. DOI:10.1016/j.energy.2021.121275. EDN:FAJRCK

12. Javaid A., Niyaz Q., Sun W., Alam M. A Deep Learning Approach for Network Intrusion Detection System. *Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies, formerly BIONETICS, 3–5 December 2015, New York, USA*. 2016. p.21–26. DOI:10.4108/eai.3-12-2015.2262516

13. Manimaran A., Chandramohan D., Shrinivas S.G., Arulkumar N. A comprehensive novel model for network speech anomaly detection system using deep learning approach. *International Journal of Speech Technology*. 2020;23(2):305–313. DOI:10.1007/s10772-020-09693-z. EDN:URAWUT

14. Alrayes F.S., Zakariah M., Amin S.U., Khan Z.I., Helal M. Intrusion Detection in IoT Systems Using Denoising Autoencoder. *IEEE Access*. 2024;12:122401–122425. DOI:10.1109/ACCESS.2024.3451726. EDN:VCPDLT

15. Vincent P., Larochelle H., Bengio Y., Manzagol P.A. Extracting and composing robust features with denoising autoencoders. *Proceedings of the 25th International Conference on Machine Learning, 5–9 July 2008, Helsinki, Finland*. Association for Computing Machinery; 2008. p.1096–1103. DOI:10.1145/1390156.1390294

16. Vincent P., Larochelle H., Lajoie I., Bengio Y., Manzagol P.A., Bottou L. Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion. *Journal of Machine Learning Research*. 2010;11:3371–3408. EDN:OCLKDJ

17. Zhang N., Ding S., Zhang J., Xue Y. An overview on restricted Boltzmann machines. *Neurocomputing*. 2018;275:1186–1199. DOI:10.1016/j.neucom.2017.09.065

18. Mayuranathan M., Murugan M., Dhanakoti V. Retracted article: best features based intrusion detection system by RBM model for detecting DDoS in cloud environment. *Journal of Ambient Intelligence and Humanized Computing*. 2021;12(3):3609–3619. DOI:10.1007/s12652-019-01611-9. EDN:LAAOLK

19. Seo S., Park S., Kim J. Improvement of Network Intrusion Detection Accuracy by Using Restricted Boltzmann Machine. *Proceedings of the 8th International Conference on Computational Intelligence and Communication Networks, CICN, 23–25 December 2016, Tehri, India*. IEEE; 2016. p.413–417. DOI:10.1109/CICN.2016.87

20. Balakrishnan N., Rajendran A., Pelusi D., Ponnusamy V. Deep Belief Network Enhanced Intrusion Detection System to Prevent Security Breach in the Internet of Things. *Internet of Things*. 2021;14:100112. DOI:10.1016/j.iot.2019.100112. EDN:CZRBGW

21. Yang Y., Zheng K., Wu C., Niu X., Yang Y. Building an Effective Intrusion Detection System Using the Modified Density Peak Clustering Algorithm and Deep Belief Networks. *Applied Sciences*. 2019. Vol. 9. Iss. 2. P. 238. DOI:10.3390/app9020238. EDN:ABGPER

22. Parhizkari S. *Anomaly Detection in Intrusion Detection Systems*. 2023. DOI:10.5772/intechopen.112733

23. Mehibs S.M., Hashim S.H. Proposed Network Intrusion Detection System in Cloud Environment Based on Back Propagation Neural Network. *Journal of University of Babylon for Pure and Applied Sciences*. 2018;26(1):29–40. DOI:10.29196/jub.v26i1.351

24. Al-Tameemi M.M.A., Alzaghir A.A.H. Improving Network Security Through Deep Learning RNN Approach. *Computational Nanotechnology*. 2024;11(4):114–121. DOI:10.33693/2313-223X-2024-11-4-114-121. EDN:GPCZUD

25. Smagulova K., James A.P. A survey on LSTM memristive neural network architectures and applications. *The European Physical Journal Special Topics*. 2019;228(10):2313–2324. DOI:10.1140/epjst/e2019-900046-x. EDN:HRKIKB

26. Han K., Yu D., Tashev I. Speech emotion recognition using deep neural network and extreme learning machine. *Interspeech 2014*. DOI:10.21437/Interspeech.2014-57

27. Roy S.S., Mallik A., Gulati R., Obaidat M.S., Krishna P.V. A deep learning based artificial neural network approach for intrusion detection. *Proceedings of the Third International Conference on Mathematics and Computing, ICMC 2017, 17–21 January 2017, Haldia, India. Communications in Computer and Information Science, vol.655*. Singapore: Springer; 2017. p.44–53. DOI:10.1007/978-981-10-4642-1_5

28. Gowdhaman V., Dhanapal R. An intrusion detection system for wireless sensor networks using deep neural network. *Soft Computing*. 2022;26(23):13059–13067. DOI:10.1007/s00500-021-06473-y. EDN:KHFOPY

29. Yang Y., Zheng K., Wu C., Niu X., Yang Y. Building an Effective Intrusion Detection System Using the Modified Density Peak Clustering Algorithm and Deep Belief Networks. *Applied Sciences*. 2019:9(2):238. DOI:10.3390/app9020238. EDN:ABGPER

30. Mohammadpour L., Ling T.Ch., Liew Ch.S., Aryanfar A. A Survey of CNN-Based Network Intrusion Detection. *Applied Sciences*. 2022;12(16):8162. DOI:10.3390/app12168162. EDN:EFJJTR

31. Razavian A.S., Azizpour H., Sullivan J., Carlsson S. CNN features off-the-Shelf: An Astounding Baseline for Recognition. *arXiv*. 2014. DOI:10.48550/arXiv.1403.6382

32. Jo W., Kim S., Lee C., Shon T. Packet preprocessing in CNN-based network intrusion detection system. *Electronics*. 2020; 9(7):1151. DOI:10.3390/electronics9071151. EDN:XVUGFM

33. Sabuhi M., Zhou M., Bezemer C.P., Musilek P. Applications of Generative Adversarial Networks in Anomaly Detection: A Systematic Literature Review. *IEEE Access*. 2021;9:161003–161029. DOI:10.1109/ACCESS.2021.3131949. EDN:TJDTSD

34. Dunmore A., Jang-Jaccard J., Sabrina F., Kwak J. A Comprehensive Survey of Generative Adversarial Networks (GANs) in Cybersecurity Intrusion Detection. *IEEE Access*. 2023;11:76071–76094. DOI:10.1109/ACCESS.2023.3296707. EDN:NOKCVG

Статья поступила в редакцию 24.01.2025; одобрена после рецензирования 10.05.2025; принята к публикации 02.06.2025.
The article was submitted 24.01.2025; approved after reviewing 10.05.2025; accepted for publication 02.06.2025.

Информация об авторах:

АЛЬ-ТАМИМИ Мохалад Мохсин Абдульхасан — аспирант кафедры информационной безопасности Санкт-Петербургского государственного электротехнического университета «ЛЭТИ» им. В.И. Ульянова (Ленина)
https://orcid.org/0009-0005-5316-1689

АЛЗАГИР Аббас Али Хасан — кандидат технических наук, доцент кафедры «Сети и системы фиксированной связи» Московского технического университета связи и информатики
https://orcid.org/0000-0003-2937-9934

АЛЬ-СВЕЙТИ Малик А.М. — кандидат технических наук, доцент кафедры сетей связи и передачи данных Санкт-Петербургского государственного университета телекоммуникаций им. проф. М. А. Бонч-Бруевича
https://orcid.org/0000-0002-6267-4727

Авторы сообщают об отсутствии конфликтов интересов.

The authors declare no conflicts of interests.