SETSCI conference proceedings
SETSCI Conference Proceedings
17, 7-12, 2024

Cognitive Models and Artificial Intelligence Conference
May 25-26, 2024, İstanbul-Türkiye

AICCONF 2024

https://doi.org/10.36287/setsci.17.1.007

Hyperparameter Optimization for A Hybrid CNN-LSTM-based Intrusion Detection System

Pınar Ayyıldız[1*], Oğuzhan Karahan [2]

[1]Electronic and Communication Engineering, Kocaeli University, Kocaeli, Turkey
[2]Electronic and Communication Engineering, Kocaeli University, Kocaeli, Turkey

[*]prayildiz@gmail.com, oguzhan.karahan@kocaeli.edu.tr

Abstract – Network security is becoming more and more important as a result of the growing use of the internet and technological developments. Intrusion Detection Systems (IDS) are utilized to detect attacks, preferably by monitoring real-time events. Therefore, they hold a significant position in network security. By detecting network attacks based on anomalies, IDSs can ensure network security. Deep learning (DL) can be used to create efficient intrusion detection systems. In this study, three different intrusion detection systems are proposed using Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM), and their hybrid combination, CNN-LSTM. Deep learning models have complex structures and require adjustment of various hyperparameters, which can be time-consuming when done manually. To address this issue, particle swarm optimization (PSO) is employed in this study for hyperparameter optimization. PSO automatically selects hyperparameters and enhances performance. The CIC-IDS2017 dataset is used to validate the approaches. Models are evaluated with different performance metrics including accuracy, recall, precision, false alarm rate, F1-score, and error rate. The hybrid CNN-LSTM model performs better than other models due to its ability to capture both temporal and spatial features simultaneously. Consequently, the results clearly illustrate the significance and impacts of hyperparameter optimization in intrusion detection systems.

Keywords – Deep learning, Intrusion Detection Systems, Particle Swarm Optimization, Hyperparameter optimization, CIC-IDS 2017

I. Introduction

As networks evolve with the advancement of technology, they are exposed to various types of attacks. This situation underscores the need for network security. Attacks exploit security vulnerabilities in computer networks. Passive defense measures such as antivirus software or firewalls cannot completely prevent attacks. Intrusion Detection Systems (IDS), an active prevention method, are preferably real-time tools aimed at detecting unauthorized and abnormal activities. They automatically monitor the entire network and are responsible for detecting any breaches. IDS allows systems to protect themselves from threats that come with increased network connectivity and reliance on other networks. IDSs are designed in two ways: signature-based and anomaly-based. Utilizing signature-based detection, known attack patterns are detected. Anomaly-based detection uses the probability of detecting abnormal patterns. The accuracy and effectiveness of IDSs are increased by using machine learning and deep learning algorithms to identify anomalies that might point to an attack.

Machine learning (ML) and deep learning (DL) techniques are successful methods that integrate intelligence into IDSs for detecting attacks. Traditional ML techniques have limited generalization capabilities, which adversely affect their detection abilities [1]. Without the need for feature engineering or domain experts, deep learning approaches may automatically extract high-level features from network traffic [2].

Convolutional Neural Network (CNN) is a high-accuracy deep learning model used for recognition and classification tasks with short training times. With the help of filters or kernels, they can automatically extract features without human intervention and are more successful in obtaining the best features compared to other models [2]. Long Short Term Memory (LSTM) is a deep learning model used for modeling sequential information, an improved type of the recurrent neural network (RNN) structure, proposed to address the problem of RNNs' inability to capture long-term dependencies. They are successful in extracting temporal features [3]. The CNN-LSTM model combines the successful aspects of both models by extracting both temporal and spatial features, thereby increasing performance.

The success of deep learning techniques over other traditional techniques is acknowledged. However, they have many hyperparameter values, and these values significantly affect their performance. It is challenging and time-consuming to manually determine the optimal values. Therefore, various optimization methods are used to determine the optimum value. Optimization methods are used to develop models with acceptable performance and closeness to the best. They enhance the generalization ability of the models and achieve higher training and testing performance [4]. Meta-heuristic optimization algorithms enable more a dynamic and comprehensive exploration of the search space. Therefore, using metaheuristic optimization algorithms is more advantageous.

In this study, Particle Swarm Optimization (PSO) is selected as the optimization method. PSO is a well-known meta-heuristic optimization technique used for hyperparameter optimization in deep learning models. PSO mimics the behavior of herds like fish and bird flocks. In PSO, a group of particles adjusts its positions around the hyperparameter space based on their own and their neighbors' best performances [4].

Choas and Pawlicki [5] proposed an artificial neural network (ANN) for intrusion detection. The grid search method was used for hyperparameter optimization in this model. Utilizing the CIC-IDS 2017 and NSL-KDD datasets, the ANN model was evaluated.

Maseer and Mostafa [6] proposed intrusion detection system using ten different supervised and unsupervised machine learning algorithms. Among these supervised ML algorithms are support vector machine (SVM), artificial neural network (ANN), naïve Bayes (NB), convolutional neural network (CNN), decision tree (DT), random forest (RF), k-nearest neighbors (k-NN) and decision tree (DT) while unsupervised ML algorithms include self-organizing maps (SOM) algorithms, expectation-maximization (EM) and k-means. The evaluation of models was conducted using the CICIDS2017 dataset. The k-NN, DT, and NB methods achieved better results compared to others and showed better performance in intrusion detection.

Jose [7] analyzed the performance of deep learning methods for intrusion detection. They evaluated convolutional neural network (CNN), deep neural network (DNN) and long short-term memory (LSTM) methods using the CIC-IDS 2017 dataset. Principal component analysis (PCA) was used to extract highly correlated data. DNN, LSTM, and CNN achieved accuracy rates of 94.61%, 97.67%, and 98.61%, respectively.

Gao et al [8] proposed a model combining Adaptive Principal Component Analysis (A-PCA) with Incremental Extreme Learning Machine (I-ELM) for intrusion detection. After adaptively selecting network traffic features, detection is performed by I-ELM. The NSL-KDD and UNSW-NB15 datasets were used to assess the model. While this model exhibits fast detection speed, classification accuracy of 81.22% for NSL-KDD and 70.51% for UNSW-NB15 was achieved.

Zheng [9] proposed a convolutional neural network (CNN) model for intrusion detection. The KDD99 dataset was used to evaluate the model. Two convolutional layers were used in the study. Two different optimizers (SGD and Adam) were used during training. The Adam optimizer outperformed SGD. The model achieved an accuracy of 95.07% and an f1-score of 94.38% after 200 epochs.

Khan [10] proposed a Convolutional Recurrent Neural Network (CRNN) model for intrusion detection. The hybrid model combines local features from the CNN and temporal features from RNN. The CSE-CIC-DS2018 dataset was used to evaluate the model, and it produced an overall accuracy of about 97.75%.

Elmasry et al. [11] proposed an intrusion detection system based on Deep Neural Networks (DNNs), Deep Belief Networks (DBNs) and Long Short-Term Memory Recurrent Neural Networks (LSTM-RNNs) for intrusion detection. They addressed the issues of high false alarm rates and low detection rates in intrusion detection systems. To tackle these problems, they proposed a dual PSO based algorithm for selecting both feature subsets and hyperparameters. They used the CIC-IDS 2017 and NSL-KDD datasets for evaluating the models. In comparison to models without pre-training on the same dataset, the discovered that the proposed models raised the Detection Rate (DR) by 4% to 6% and reduced the False Alarm Rate (FAR) by 1% to 5%.

The problem of manual deep learning model adjustment is addressed in this study by proposing an intrusion detection system algorithm based hyperparameter optimization. CNN, LSTM, and CNN-LSTM hybrid models are used as classifiers. The goal is to optimization methods to improve the performance of models.

This study's contributions can be summarized as follows:

- An algorithm that automatically solves the problems of manually adjusting the hyperparameters of deep learning models is proposed. It enhances the performance of the models.
- The proposed three techniques were validated using the CICIDS2017 datasets for classification.
- Different evaluation metrics were used to assess models that were proposed through the use of optimization method. It was observed that the created hybrid CNN-LSTM model exhibited better performance.
- Other deep learning models created in the literature were examined for the evaluation of approaches.

II. Materials and Method

In this study, a deep learning-based intrusion detection system, adjusted using particle swarm optimization (PSO), is introduced for attack detection. While CNN excels in automatically extracting features from the dataset, LSTM produces better results when dealing with sequential data. The CNN-LSTM hybrid model combines the strengths of these two models to create an effective technique. The detection algorithm begins with preprocessing the dataset through data cleaning and processing steps. The selection of hyperparameter values is crucial for DL techniques. Therefore, PSO, a metaheuristic optimization method, is employed for hyperparameter selection. Hyperparameter optimization enhances the performance of the models. Finally, the evaluation step of the created models is conducted. Figure 1 displays the proposed algorithm.

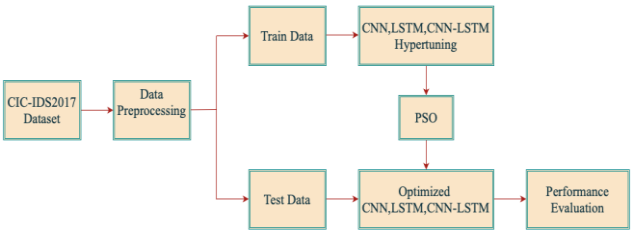

Fig 1- The proposed algorithm

A. Data Preprocessing and Dataset

Preparing the dataset is an essential step for the effectiveness of deep learning models. It is crucial for speeding up training and achieving higher accuracy. It involves various steps such as data cleaning, converting non-numeric data, and handling missing data. In this study, the CIC-IDS2017 dataset is checked for null, infinite, and NaN values to prevent any missing values that could adversely affect the study. Eight features containing no information are being removed from the

dataset. These features 'Bwd PSH Flags', 'Bwd Avg Bytes/Bulk', 'Fwd Avg Bulk Rate', 'Bwd URG Flags', 'Bwd Avg Bulk Rate', 'Fwd Avg Packets/Bulk', 'Fwd Avg Bytes/Bulk' and 'Bwd Avg Packets/Bulk'. The process of converting categorical features into numerical values is performed using label encoding. The dataset is split into three sets for training, validation, and testing respectively, with ratios of 70%, 20% and 10%. Standardization is required for numerical features to ensure consistency and regular scales. For this purpose, normalization is used. The primary goal of normalization is to maintain the data behavior by transforming all features between 0 and 1. In this study, the normalization step is performed using min-max scaling with equation (1).

$$x_i = \frac{x_i - Min}{Max - Min} \tag{1}$$

This study makes use of the CIC-IDS 2017 dataset. It comprises 8 types of attacks and consists of 78 features.

B. Deep Learning Models

Machine learning includes deep learning as a subfield. Artificial neural networks form its foundation. It is employed to resolve complex issues [12]. It holds a significant place in the field of data science due to its ability to learn data representations. With its layered structure, it enables automatic learning of feature representations, eliminating the need for time-consuming feature engineering and contributing to time efficiency. These days, deep learning is extensively used many different fields, such as natural language processing, sentiment analysis, image recognition, and cybersecurity.

Convolutional Neural Network (CNN): A popular deep learning model called CNN draws inspiration from the innate visual perception capabilities observed in living organisms [13]. Convolution, pooling, activation layers, fully connected layers for classification, and output layers make up the architecture of convolutional neural networks.

For feature learning, the convolution and pooling layers are employed, while the fully connected layers and output layer are classification and recognition. The convolutional layer is the most fundamental layer used in the architecture, focusing on the use of kernels or filters. The kernel is convolved over the input's spatial dimensions in order to produce a feature map. The main advantage of the feature map is that it can retain all distinctive and important features. By using pooling layer, the feature map's dimensions are decreased, keeping only significant features and lowering data errors. The activation layer is typically a non-linear layer that follows each layer. Sigmoid, hyperbolic tangent, softmax, and ReLU are commonly used activation functions in deep learning. The fully connected layer is where the classification and recognition processes take place. The matrix is flattened before reaching this layer, resembling the arrangement of neurons in a traditional neural network. Therefore, all nodes in a completely linked are directly connected to all nodes in the layers that come before and after them. CNNs can capture complex features of IDS quickly.

Long Short Term Memory (LSTM): LSTM is a type of RNN. It can learn long-term dependencies and is used to model sequences. It was developed to overcome the issues of gradient vanishing/exploding that arise in RNNs. LSTM is commonly used in time series predictions and is also highly suitable for addressing other problems requiring temporal memory [14].

Memory blocks are a groups of interconnected elements that make up the LSTM architecture. By using nonlinear gating units, these memory blocks seek to control the flow of information while preserving their state over time. In generally, the input gate (i_t), output gate (o_t), and forget gate (f_t) make up the LSTM architecture. The cell state at time t is represented by c_t. In the architecture, the input gate is in charge of updating the cell state. Which information from earlier cell states will be ignored is decided by the forget gate.

The equations (2)-(7) are provided below to calculate the gate values in the LSTM architecture.

$$f_t = \sigma(W_f \,.\, [h_{t-1}, x_t] + b_f) \tag{2}$$

$$i_t = \sigma(W_i \,.\, [h_{t-1}, x_t] + \, b_i) \tag{3}$$

$$\widetilde{C_t} = \tanh\,(W_C \,.\, [h_{t-1}, x_t] + b_C \tag{4}$$

$$C_t = f_t * C_{t-1} + i_t * \widetilde{C_t} \tag{5}$$

$$o_t = \, \sigma(W_o \,.\, [h_{t-1}, x_t] + b_o) \tag{6}$$

$$H_t = o_t * \tanh\,(C_t) \tag{7}$$

The input gate denoted by i_t,the forget gate by f_t and the output gate o_t in these equations. The bias values for the input gate, cell state, forget gate and output gate are b_i, b_c, b_f, and b_o in that order. Finally, the wight matrix is used to represent W. The weight matrices that link the input layer to the input gate, cell state, forget gate and output gate are W_i, W_c, W_f and W_o [15].

CNN-LSTM: Network data contains features with many analysis options. However, these features are often based on statistics. It contains time information thanks to its spatial properties and sequential structure. Spatial and temporal features play an important role, especially in the field of infiltration detection. CNN has disadvantages when it comes to extracting and evaluating temporal features from data, even though it is effective at capturing spatial features. LSTM, on the other hand, can extract the time feature by checking the memory and forgetting of past data. By combining CNN and LSTM networks, spatial and temporal features can be extracted simultaneously [16].

C. Optimization Tasks

Deep learning (DL) models have complex structures. Before learning begins, there are model-specific hyperparameters that need to be adjusted. These hyperparameters significantly influence the training, behavior, and performance of the model. Manual adjustment of model hyperparameters can be time-consuming due to the complexity of the models. Therefore, optimization methods have emerged where models are automatically optimized. Metaheuristic algorithms are techniques used to solve optimization problems in large search spaces. They have the ability to efficiently search the space for optimal or approximate solutions. Hence, they are suitable for use in optimization problems with large configuration spaces due to their efficiency. In this study, a metaheuristic algorithm called PSO is used. PSO allows each particle to communicate with other particles in each iteration to determine and update the current global optimum.

Particle Swarm Optimization (PSO): PSO algorithm is a swarm-based meta-heuristic optimization algorithm proposed by Eberhart and Kennedy (1995). Animal social behaviors, such as those of insects, flocks, birds and fish, are simulated by the PSO algorithm [17]. The primary goal of PSO is to mimic the behaviors performed by individuals to survive for the longest period. In PSO, each individual is considered a particle and interacts with other particles. Each particle moves based on its own past decisions and the decisions of the swarm. During this process, particles evaluate their performance and try to select the best decision for themselves by comparing it with their previous states.

The PSO algorithm is a multi-particle, swarm-based search method. The number of particles in the swarm is indicated by the integer S. In a given set, every particle has two vectors of length N. The problem's size is determined by N. The position vector, which shows the position at that precise instant, is the first vector. The velocity vector, which is the second vector, controls the particle's direction and speed in the subsequent iteration. A particle remembers the optimal position and velocity of the swarm in addition to its own. Based on its previous experience, each particle adjusts its position towards the best position. In each iteration, the information of particles is combined, adjusting the velocity of each dimension, and this velocity is used to calculate the new position of the particle. Particles continuously change their positions until they reach the optimal positions.

Equations (8) and (9) describe how the PSO algorithm updates the swarm's particle velocity and position:

$$V_{id}^{k+1} = WV_{id}^{k} + c_1 r_1^k \left(pbest_{id}^{k} - x_{id}^{k}\right) + c_2 r_2^k \left(gbest_{id}^{k} - x_{id}^{k}\right) \quad (8)$$

$$x_{id}^{k+1} = x_{id}^{k} + V_{id}^{k+1} \quad (9)$$

V_{id}^{k} and x_{id}^{k}represent the velocity and position, respectively, of the i-th particle in d dimensions in the k-th iteration. $pbest_{id}^{k}$ and $gbest_{id}^{k}$ represent the personal best position and the global best position (i.e., the best position of the group) of the i-th particle in d dimensions in the k-th iteration. W represents the inertia weight, which depends on the particle's previously obtained position and is typically kept within the range of 0.4 and 0.9 in studies. c_1 and c_2 represent the acceleration constants. There are two random numbers in the range [0,1]: r_1^k and r_2^k.

Proposed Optimization Framework: This section encompasses the steps of the proposed optimization framework. The hyperparameters of the proposed CNN, LSTM, and CNN-LSTM deep learning models will be obtained within this framework. In this study, hyperparameters to optimize for models are number of epochs, learning rate, number of dense layers, number of neurons in dense layers, dropout rate, batch size and optimizer type. The search space for these parameters is provided in Table 1. The proposed PSO algorithm selects the most suitable hyperparameter values by maximizing the accuracy over the training set. Bellow is an outline of the proposed algorithm's steps.

Step 1: Initialization is the first step. The population size, the stopping criterion maximum number of iterations, and the necessary parameters are determined. The population is initialized. The positions of individuals are the hyperparameter parameters that need to be optimized, as given in Table 1.

Step 2: Evaluation is the second step. The model is trained only on the training set, and the accuracy of the trained model is evaluated on the validation set.

Step 3: The particle's best position (pbest) is updated based on the fitness value.

Step 4: The personal best position with the best fitness value among the current personal best positions is updated as the global best position.

Step 5: For every particle, use equations (8) and (9) to update the particle positions and velocities.

Step 6: When the maximum iteration count, the stopping criterion, is reached, the optimized hyperparameters are obtained from the global best position gbest. If the stopping criterion is not met, the next iteration continues. The process resumes by returning to Step 2.

Table 1. The defined hyperparameters and their range

Hyperparameter	Range
Number of dense layers	(1,5)
Learning rate	$(1x10^{-5},1x10^{-2})$
Dropout rate	(0.1,0.9)
Number of neurons in dense layers	$[2^4,2^5,2^6,2^7,2^8]$
Batch size	$[2^4,2^5,2^6,2^7,2^8]$
Number of epochs	(1,100)
Optimizer type	(Adam,Adamax,Adelta,Nadam, SGD,Adagrad,RMSprop)

In the PSO algorithm, the population size is set to 10, and the maximum number of iterations is determined as 100. Additionally, other parameters of PSO, namely the scaling factors c_1=2 and c_2=2, and the inertia weight W=0.5, are configured.

III. RESULTS

The outcomes of the IDSs using CNN, LSTM, and CNN-LSTM with PSO-optimized hyperparameters are presented in this section. The CIC-IDS 2017 dataset is used in the experiments.

A. *Evaluation Measures*

The performance of the proposed models has been evaluated using the same evaluation metrics that have been applied to the majority of previous research on intrusion detection systems. These metrics consist of specificity, error rate, false negative rate (FNR), recall, false alarm rate (FAR), accuracy, precision and F1-score [18].

Equations (10)-(16) provide the mathematical formulas of the metrics

$$Accuracy = \frac{TP + TN}{TP + TN + FN + FP} \quad (10)$$

$$Precision = \frac{TP}{TP + FP} \quad (11)$$

$$Recall = \frac{FP}{TN + FP} \quad (12)$$

$$F1 - Score = \frac{2 * Precision}{Precision + Recall} \quad (13)$$

$$Specificity = \frac{TN}{TN + FP} \quad (14)$$

$$FAR = \frac{FP}{TN + FP} \quad (15)$$

$$FNR = \frac{FP}{TN + FP} \quad (16)$$

$$ErrorRate = \frac{FP + FN}{TP + TN + FP + FN} \quad (17)$$

Where,

- TP-True Positive
- TN-True Negative
- FP-False Positive
- FN-False Negative

B. *Performance Analysis*

The experiments in the study were conducted using the CIC-IDS2017 dataset, which contains network attacks. Three developed deep learning models were employed to classify the attacks. PSO was used to determine the best hyperparameter values for these models. The PSO algorithm selects the best hyperparameter values that maximize accuracy in the dataset. PSO facilitated the time-consuming and complex hyperparameter tuning process and had an impact on classification performance. The optimal hyperparameter values obtained for each model are presented in Table 2.

Table 2. The optimal hyperparameters

Hyperparameter	PSO-CNN	PSO-LSTM	PSO-CNN-LSTM
Number of dense layers	3	1	1
Learning rate	0.0099258	0.0090412	0.005387
Dropout rate	0.1	0.1	0.20150
Number of neurons in dense layers	256	256	256
Batch size	256	128	256
Number of epochs	54	75	95
Optimizer type	Adamax	Adam	Adamax

The models are trained using the training set. The accuracy of the training set is validated with the validation set. The performance of the models is evaluated on previously unused test data. CNN, LSTM, and CNN-LSTM models are compared using various metrics. Figure 2 shows the evaluation metrics of F1-score, precision, accuracy, recall, and specificity for the proposed models. Figure 3 shows the FAR, error rate and false FNR. The results indicate that the CNN-LSTM has superior performance as compared to the others. Specifically, the higher accuracy, precision, recall, F1-score, and specificity of the CNN-LSTM model suggest its more effective discrimination between positive and negative classes. Additionally, the lower false alarm rate, false negative rate, and error rate of the CNN-LSTM model demonstrate its reliability in reducing incorrect predictions. This highlights the balanced and dependable performance of the CNN-LSTM in both accurately identifying the positive class and reducing false alarm and false negative rates.

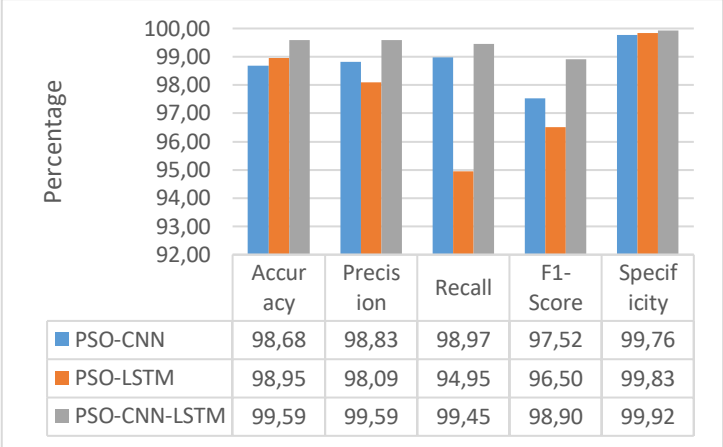

Fig 2- Evaluation results of the proposed models

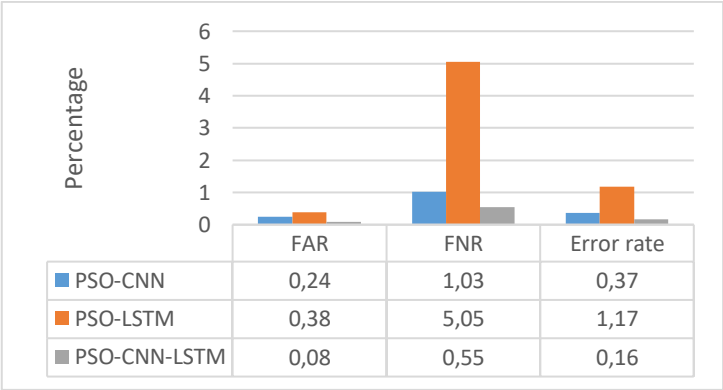

Fig 3- Comparison of the FAR, FNR and error rate in testing set

C. *Comparative Analysis*

Based on the outcomes shown in Table 3, in this part a comparison analysis is carried out in this part for assessing the validity of diverse machine learning models. Important measures like accuracy, precision, recall, and F1-score are employed in this analysis to assess the models' performance. The CIC-IDS2017 dataset is used for training, testing, and validation of all models. The findings show that the proposed models perform better than the other techniques.

Table 3. Comparison of proposed method with current methods

Models	Accuracy	Precision	Recall	F1-Score
DNN[19]	90.61	80.85	84.60	84.60
LSTM[19]	97.67	94.96	95.95	93.55
SMOTE+PCA[20]	81.47	81.69	95.76	88.17
ANN[21]	96.24	-	-	-
Proposed PSO-CNN	98.68	98.83	98.97	97.52
Proposed PSO-LSTM	98.95	98.09	94.95	96.50
Proposed PSO-CNN-LSTM	99.59	99.59	99.45	98.90

IV. DISCUSSION

This section discusses the superiority of the proposed methods over existing ones. Existing methods have some limitations. Examples of these limitations include increasing computational costs by spending a lot of time for training, local optimum problems, and population diversity issues. The proposed PSO-based CNN-LSTM model overcomes these limitations. Performing hyperparameter optimization with PSO ensures the creation of a more optimal model. The CNN-LSTM model, on the other hand, enhances the system's ability to detect attacks more effectively and accurately by combining the strengths of CNN and LSTM models.

V. CONCLUSION

In this study, intrusion detection systems based on particle swarm optimization-based CNN, LSTM and hybrid CNN-LSTM deep learning models were evaluated with the CIC-IDS 2017 dataset. For feature selection, no strategy was applied. The dataset was standardized using min-max normalization method. The PSO algorithm was used to address hyperparameter selection issues in deep learning models. Hyperparameters maximizing accuracy in the dataset were automatically determined. The models' capacity for classification, generalization, and portability were all greatly impacted by the application of PSO. The hybrid CNN-LSTM model illustrated superior classification performance compared to CNN and LSTM models due to its ability to learn both temporal and spatial features. To emphasize the performance of the proposed models, this study preserved the original dataset and its distribution. Balancing classes to increase the detection of minority classes in the future will enhance the success of intrusion detection systems.

REFERENCES

[1] P. Mishra, V. Varadharajan, E.S. Pilli "A detailed investigation and analysis of using machine learning techniques for intrusion detection", *IEEE Communications Surveys & Tutorials*, vol.21, pp. (99)1:1, June. 2018.

[2] P. Sun, P. Liu, Q. Li, C. Liu, X. Lu, R. Hao, J. Chen, "DL-IDS: extracting features using CNN-LSTM hybrid network for intrusion detection system", *Security and Communication Networks*, vol. 20, pp. 1-11, 2020.

[3] M. Jenckel, S.S. Bukhari, "Training LSTM-RNN with imperfect transcription: limitations and outcomes," in *The 4th international workshop on historical document imaging and processing*, 2017, paper 48-53.

[4] Erden E, Demir İE, Kökçam AH. "Enhancing Machine Learing Model Performance With Hyper Parameter Optimization", 2023

[5] M. Choraś, and M. Pawlicki, "Intrusion detection approach based on optimised artificial neural network", *Neurocomputing,* vol. 452, pp.705-715, Sep., 2021

[6] Z.K. Maseer, R. Yusof, N. Bahaman, S.A. Mostafa and C.F.M Foozy, "Benchmarking of machine learning for anomaly based intrusion detection systems in the CICIDS2017 dataset", *IEEE Access*, Feb, 2021

[7] J. Jose, D.V. Jose, "Deep learning algorithms for intrusion detection systems in internet of things using CIC-IDS 2017 dataset", *International Journal of Electrical and Computer Engineering (IJECE)*, 2022, paper 1134-1141.

[8] J. Gao,, S. Chai, B. Zhang, and Y. Xia "Research on network intrusion detection based on incremental extreme learning machine and adaptive principal component analysis", *Energies*, vol.12(7), pp. 1–17, March, 2019.

[9] W. Zheng, "Intrusion detection based on convolutional neural network", *International Conference on Computer Engineering and Application (ICCEA),* 2020

[10] M. A. Khan, "HCRNNIDS: hybrid convolutional recurrent neural network-based network intrusion detection system", *Processes ,*vol. 9, pp. 834, 2021.

[11] W. Elmasry, A. Akbulut and A.H. Zalim, "Evoling deep learning architectures for network intrusion detection using a double PSO metaheuristic", *Computer Networks*, vol.168, 2020.

[12] Y. Xın, L. Kong, Z. Lıu, Y. Chen, Y. Lı, H. Zhu, M. Gao, H. Hou and C. Wang, "Machine learning and deep learning methods for cybersecurity", *Ieee Access*, vol.6, July, 2018.

[13] J. Gu, Z. Wang, J. Kuen, L. Ma,, A. Shahroudy, B. Shuai, T. Liu, X. Wang, G. Wang, J. Cai, T. Chen, "Recent advances in convolutional neural networks", *Pattern Recognition*, vol.77, pp.354-377, May, 2018.

[14] G. V. Houdt, C. Mosquera and G. Nápoles, G, "A review on the long short-term memory model" *Artificial Intelligence Review,* vol.53(8), pp. 5929-5955, May, 2020

[15] T. Le, J. Kim, "An effective intrusion detection classifier using long short-term memory with gradient descent optimization" in *2017 International Conference on Platform Technology and Service (PlatCon)*, 2017, pp. 1-6

[16] A. Halbouni,, T. S. Gunawan., M. H. Habaebi, M. Halbouni, M. Kartiwi, R. Ahmad, "CNN-LSTM: hybrid deep neural network for network intrusion detection system" *IEEE Access,* 2017.

[17] Y. Shi, "Particle swarm optimization: developments, applications and resources" in *Proceedings of the 2001 congress on evolutionary computation,* 2001, pp. 81-86).

[18] Z. Vujović, "Classification model evaluation metrics", *International Journal of Advanced Computer Science and Applications*, vol. 12(6), pp. 599-606, 2021.

[19] J. Jose and D. V. Jose, "Deep learning algorithms for intrusion detection systems in internet of things using CIC-IDS 2017 dataset", *International Journal of Electrical and Computer Engineering (IJECE),* vol. 13(1), pp. 1134-1141, 2023.

[20] A. Yulianto, P. Sukarno, and N. A. Suwastika, ''Improving AdaBoost-based intrusion detection system (IDS) performance on CIC IDS 2017 dataset,'', *Journal of Physics Conference Series*, 2019.

[21] Z. Pelletier and M. Abualkibash," Evaluating the CIC IDS-2017 dataset using machine learning methods and creating multiple predictive models in the statistical computing language R", *International Research Journal of Advanced Engineering and Science*, 2020, paper. 187-191.