Communications on Applied Nonlinear Analysis
ISSN: 1074-133X
Vol 31 No. 2s (2024)

Statistical Implementation for SD-RNN Model for Multi-Class Classification for Network Intrusion Detection System

Nekita Chavhan[1], Dr. Prasad Lokulwar[2]

[1]PhD Scholar, G H Raisoni University, Amravati, India

Assistant Professor Department of Data Science IoT and Cyber Security,

G H Raisoni College of Engineering, Nagpur.

Email id: nekita.chavan@raisoni.net

[2]Professor, Department of Computer Science and Engineering , G H Raisoni College of Engineering, Nagpur

Email id: prasad.lokulwar@raisoni.net

Article History:

Received: *16-03-2024*

Revised: *08-05-2024*

Accepted: *28-05-2024*

Abstract:

With the increasing popularity of technologies that depend on computer networks, providing security is of paramount importance. Consequently, intrusion detection systems (IDS) play a vital role in monitoring these networks. An essential element in securing the network of an organization is the Intrusion Detection System (IDS), which serves as the primary barrier against cyber threats and is tasked with thwarting illegal entry into the network. Flow-based network traffic analysis is commonly used in IDS solutions to identify security concerns. In recent years, several innovative strategies have been proposed and implemented to address the issue of network security, with a specific focus on Intrusion Detection Systems (IDSs). The creation of intrusion detection systems using AI methods is a modern method for finding breaches in a network. Since there are many possible approaches, it is important to have a standardized method that facilitates good judgement when classifying intrusions. This paper presents a novel approach utilizing deep learning and machine learning modelling to develop a model for multi-class classification. The principal objective is to create IDS that can effectively detect anomalies using flow-based analysis. The latest CICIDS2017 dataset is used in experimentation for testing and training. The experiments performed with a deep learning model for IDS produced promising results, achieving a 99.77% accuracy rate for multi-class classification while employing the specific dataset.

Keywords: Cybersecurity, Deep Learning, Machine Learning, Multi-Class Classification, Network Intrusion Detection System.

1. INTRODUCTION

Cyberattacks come in many forms nowadays, and the Internet is always under siege. The flaws in the system are what produce the intrusions. As more of our lives are conducted online, it's crucial that we take precautions to protect our personal data. There is a growing need to create a dependable and adaptable Intrusion Detection System (IDS) [1] since the nature and methods of infiltration evolve over time with the sophistication of attackers. IDS serves the function of monitoring network traffic and identifying any potentially malicious or unauthorized actions. Various approaches have been proposed for the construction of the IDS. There is a lot that has to be done to make IDS more effective. Various data mining and machine learning approaches have been developed throughout the years to improve IDS. All of these systems, however, have shortcomings that make it easy for an attacker to compromise the system. While some Intrusion Detection Systems only monitor network traffic and issue alerts if something out of the ordinary occurs, others take corrective measures once threats are identified [2].

Today's computer programmers are indispensable since they facilitate so many aspects of daily living. Modern computer applications have resulted in the transmission of large amounts of data, including private information, across the Internet. Given their capacity to constantly monitor networks in search of signs of malicious activity, IDSs hold great promise as a means of keeping sensitive user information safe [3]. However, because of the large amount of network traffic and the wide range of threats, it is necessary to have a system with intelligence that can independently acquire and recognize attacks. Malicious actions may be categorized as a classification and dimensionality reduction challenge from the perspective of machine learning (ML) and deep learning (DL). It is standard practice to utilize ML and DL to incorporate intelligence into an IDS, making it simple to identify assaults of any form and protecting systems from any danger. However, picking the correct dataset is crucial for developing an effective ML model for intrusion detection. This section discusses the use of various ML and DL-based methods that are currently accessible to publicly available IDS datasets.

The primary objective of the study [5] is to implement ML-supervised algorithms-based IDS for IoT and identify attacks on networks connected to the IoT [5]. For preprocessing impact analysis, the SVM classifier [6] was utilized as a classifier due to its high accuracy and ease of use. The trained model uses the pattern recognition capabilities of a neural network for machine learning to successfully identify virtually all conceivable attacks in a software-defined networking (SDN) environment, as presented in [7]. The machine learning approach to detecting attacks on network traffic used many categories [8]. Model of computer attack detection created [9] using machine learning techniques, and its implementation. The novel hybrid method [10] uses a stacking system to merge decision tree and random forest algorithms. With the IDS design proposed in [11], a methodological approach to decision-making support for algorithm selection is provided. Advanced Intrusion Detection System (IDS) developed by utilizing a deep learning approach [12], which suggests a combination of Long Short-Term Memory Network (LSTM) and Convolutional Neural Network (CNN) in a hybrid network. The recurrent neural networks and random forests comprises of two machine learning classifiers to classify the IDS [13]. A new multi-step process-based universal intrusion detection framework was suggested [14]. DL model-based IDS developed [15] that enhances intrusion detection system which utilizes a hybrid network consisting of LSTM and CNN to extract both temporal and spatial features from network traffic data. Network assaults have been classified [16] using the optimal feature set and six popular classifiers. An intrusion detection system (IDS) was introduced [17] that utilizes flow-based anomaly detection for multi-class categorization and employs DL based model, namely a neural network with many stacked fully connected layers. Discrete and non-discrete versions of classification and feature selection algorithms for network intrusion and anomaly detection were investigated [18]. The effectiveness of deep learning systems for detecting intrusion was studied [19]. To identify modern intrusion assaults and protect medical records, a Deep Learning-based hybrid architecture called "ImmuneNet" was presented [20]. The CIC IDS-2017 dataset was subjected to preprocessing, analysis, and the development of a predictive model in the language R [21]. This model can determine whether or not a given network connection is malicious. To find useful and efficient ML-AIDS of computers and networks, [22] makes use of supervised and unsupervised ML methods.

Intrusion Detection System with Improved Anomaly Detection Extensive information dimensionality reduction model (EIDM) was suggested by deep learning [23]. There is some evidence [24] that tree-based ML approaches are useful in the flow-based intrusion detection issue. NIDS Using selected features with ML model evaluation [25]. The Proximal Policy Optimisation (PPO) used [26], an effective hyperparameter control approach for an NIDS. SDN-IoT networks are equipped [27] with a Multi-Attack Intrusion Detection System (MAIDS).

Data sets are replicas of known network-based attacks used in intrusion detection systems. To enhance the usefulness of real-time applications, it is crucial to use datasets that accurately represent contemporary assault scenarios. Consequently, the proposed experiments are utilizing the standard benchmark dataset available CICIDS2017. The aggregate quantity of traffic records in CICIDS2017 amounts to 2.8 million. Hence, the aim of this paper was to assess the effectiveness of different intelligence learning techniques for the goal of categorizing intrusions. The primary contributions of this proposed research paper are as follows:

- Developed a multiclass intrusion classification system specifically designed to categorize various forms of network traffic using the CICIDS 2017 dataset.
- Examined the influence of optimizing classifier parameters on both feature selection and classification.
- Evaluated the proposed model based on classification performance parameters accuracy and training time.
- Conducted a performance comparison between the proposed Intrusion Detection System (IDS) utilizing DL and ML models.

The following sections of this work are structured in the following manner. Section 2 provides a detailed explanation of the materials and procedures employed in this paper. Section 3 presents comprehensive information on the experimental configuration and analyzes the obtained experimental outcomes. The conclusion of this research is presented in the final section.

2. MATERIALS AND METHODS

This section explores several design components of the proposed multiclass Intrusion Detection System (IDS) paradigm. Figure 1 illustrates the classification-based framework that is being presented. The system goes through several stages in the pipeline, namely; Data Acquisition, Data Pre-processing, Feature Selection/Optimization, Modelling and Classification using DL/ML Models and Model Tuning and Evaluation.

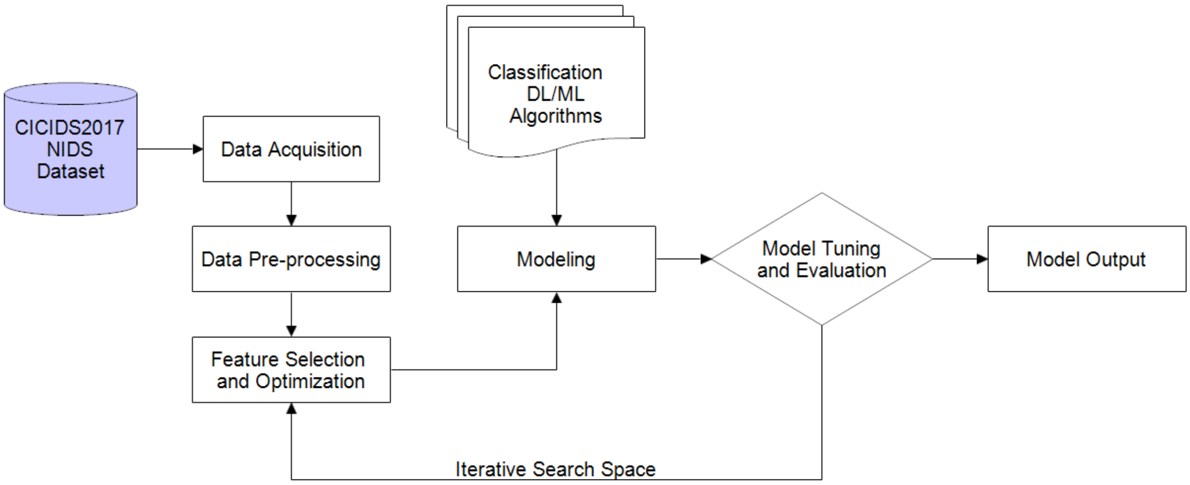


Figure 1: Proposed Multiclass Intrusion Classification System Using DL/ML Algorithms

A. Data Acquisition – CICIDS2017 Dataset

In this work, the Canadian Institute for Cybersecurity (CIC) and the University of New Brunswick (UNB) collaborated to create a public intrusion detection dataset known as CICIDS2017 [9]. The CICIDS2017 dataset encompasses state-of-the-art network attack scenarios and fulfills all the criteria of real assaults. The data was collected over the course of five days and is split into eight files, one each for regular and aberrant network activity and for various sorts of assaults. As can be seen in Table I, a new kind of attacks was used every day.

The CICFlowMeter program was used to extract 83 characteristics from the network traffic, one for each row in the dataset. The CICFlowMeter includes both forward and backward data for the 83 statistical features. This is because it creates Bidirectional Flows (Biflow), where the direction of travel is determined by the first packet. A portion of the initial 83 attributes utilized, removing attributes such as source and destination IPs, Biflow ID, and date. This resulted in a dataset consisting of 79 attributes, with the 79th attribute serving as the label identifying the type of traffic shown in the current Biflow.

Table 1: Network Traffic Data of CICIDS2017 Dataset

Day Activity	File name	Output Classes
Monday	Monday-WorkingHours.pcap ISCX.csv	Benign (Normal activities)
Tuesday	Tuesday-WorkingHours.pcap ISCX.csv	Benign and BruteForce, SSHPatator, FTP-Patator
Wednesday	Wednesday-workingHours.pcap ISCX.csv	Benign and DoSHulk, DoS/DDoS, DoSSlowhttptest, DoSslowloris, Heartbleed, DoSGolden-Eye

Thrusday	Thursday-WorkingHours-Afternoon-Infilteration.pcap ISCX.csv	Benign and MetaExploit, Infiltration
	Thursday-WorkingHours-Morning-WebAttacks.pcap ISCX.csv	Benign and WebAttack-XSS, WebAttack-Sql, WebAttack-BruteForce
Friday	Friday-WorkingHours-Morning.pcap ISCX.csv	Benign and Botnet
	Friday-WorkingHours-Afternoon-DDos.pcap ISCX.csv	Benign and DDoS
	Friday-WorkingHours-Afternoon-PortScan.pcap ISCX.csv	Benign and PortScan

B. Data Pre-processing

Our studies involve preprocessing the data, applying a data cleaning transformation, scaling the data, and encoding the labels. All the CSV files' contents were first loaded into data frames, and then the frames were combined to form a single dataset. The initial field is a header that contains the column names and their corresponding features. When the CICIDS 2017 CSVs were combined, it was found that many of the header fields were repeated. After excluding these, the remaining sample size was 2,660,377. Duplicate entries and unrealistic characteristics were removed from the data sample. This cuts the total number of features down to 69 useable ones during ML training. All records in the CICIDS 2017 validation set were checked for duplication and deleted if necessary. Classes were formatted to return integers instead of strings as part of the encoding process. To round out the preprocessing, we employ the min-max data transformation, which adjusts the scale and direction of each feature so that it falls inside a narrow band. The distribution of each feature was plotted and visually inspected before we began putting the cleaned data into our ML algorithm. This was possible because our set of traits was small enough that we could examine each one separately. During this procedure, we discovered that several characteristics were extremely asymmetrical. To alleviate the issues that arise from using a distance-based classifier with a typical scaling strategy, the feature scaling and transformation method is implemented.

Changing the distribution of classes to make it more even is a direct approach to resolving class differences. Class allocations can be made more equitable in two ways. Both under- and over-sampling, in which members of one group are ignored in favor of those from another, are forms of bias. An unbalanced dataset can be sampled in a variety of ways, including both under- and over-sampling. The SMOTEENN approach employed to cleanse the data, which integrates the Synthetic Minority Over-Sampling approach (SMOTE) for increasing the representation of the minority class, and the under-sampling technique for reducing the representation of not just the majority class but also all other classes. Someone needs to use extreme caution throughout the assessment phase when employing over-sampling techniques. If over-sampling is conducted before dividing the dataset into training, development, and test sets, the assessment's reliability is compromised. This is because the test set will be considered deliberately mixed with new data, resulting in a distribution that differs from the original. The correct procedure is to first divide the dataset in half and then use the over-sampling approach only to the training set. In this approach, the ML algorithm may get additional data for training, but the development and test sets are kept unmodified and can be relied upon for model evaluation and refinement.

Once we were done cleaning the data, we divided the dataset in half, allocating 20% of the total to testing and 80% to training. The learning set was utilized for that purpose, the validation set was employed for rapid evaluation of the prototypes during train, and the test set was utilized for the final assessment of the model.

C. Feature Selection and Optimization

Feature Selection refers to the procedure of picking out the data points that will be used to train a model. During model learning, feature selection methods seek to identify the most informative data points from which to construct accurate representations of the phenomena under investigation. Useful for both a categorical (class) target variable and numerical input data, Analysis of Variance (ANOVA) is a feature selection approach, which is used to compare the means of many groups that have known differences. In general, it seeks to understand how different aspects of the data are interconnected. By comparing the standard deviations of representative samples, it may also be used to test the null hypothesis that two or more populations have the same mean value. The offered data set is biased by systematic rather than random variables. Researchers employ the ANOVA test often to assess the influence of potential confounders on a dependent variable. The ANOVA method is a statistical technique that involves using an F-statistic, which is specifically referred to as an ANOVA f-test in this context. The F-Statistic computed using

$$F = \frac{MST}{MSE}$$

$$MST = \frac{\sum_{i=1}^{k}\left({T_i^2}/{n_i}\right) - {G^2}/{n}}{k-1}$$

$$MSE = \frac{\sum_{i=1}^{k}\sum_{j=1}^{n_i} Y_{ij}^2 - \sum_{i=1}^{k}({T_i^2}/{n_i})}{n-k}$$

Where, MSE - Mean Sum of Squares due to errors, MST - Mean Sum of Squares due to treatment, T - sample mean response for group i, G - overall sample mean response, Y - sample variance for group i,j.

D. Classification DL/ML Algorithms

Sequential Deep Recurrent Neural Network (SDRNN)

Recurrent Neural Networks (RNN) are a type of Deep Learning model used to simulate sequential input. While traditional CNN are best suited for dealing with geographical data, Sequential Deep RNN can simulate time-dependent and sequential data issues successfully. Before attention models were developed, recurrent neural networks (RNNs) were commonly used for handling sequential input. A deep feedforward model may require parameters that are customized to a particular sequence. There's a chance it can't generalize to sequences of varying lengths, either. By assigning the same weights to each sequence element, Recurrent Neural Networks may generalize to sequences of variable lengths while using fewer parameters. The architecture of RNNs allows them to generalize to various types of structured data outside sequential data, such as geographical or pictorial. All deep neural networks, including RNNs, include identical input and output structures. Within the RNN, each time step is denoted by a distinct unit that possesses a activation function which is predetermined. The "hidden state" of a unit refers to its true internal status. The latent state refers to the past data that the network has access to at a certain moment in time. The hidden state is modified at each time interval to represent the progression of the network's comprehension of the previous events. The subsequent recursive equation is employed to modify the hidden state with the ultimate anticipated outcome.

$$h_{t=\tanh W_{hh}h_{t-1}+W_{xh}X_t}$$

$$Y_t = W_{hy}h_t$$

Where, W_{hy} - weight at output layer, Y_t – output, W_{xh} - weight at input neuron, W_{hh} - weight at recurrent neuron.

Decision Tree

A decision tree is a classifier in machine learning that uses a tree structure. Each node in the tree symbolizes a feature in a dataset, while the branches symbolize decision rules and the leaf nodes indicate the outcomes. The objective is to develop a model that can predict the value of a target variable by leveraging decision rules derived from the properties of the data. The system is comprised of two nodes: the Leaf Node and the Decision Node. When a choice has to be made, it is made at a choice node, which has numerous branches, and the results are displayed at a Leaf node, which has no branches. The main concept is to partition the data space into areas with low density and areas with high density using a decision tree. Both binary and multiway splits are possible in a binary tree. When data is not sufficiently similar, the algorithm will continue to divide the tree. The final output of the training process is a decision tree that may be utilized to generate accurate predictions in a given category. This approach is very dependent on a fundamental notion known as "Entropy". Its value characterizes the degree of randomness of a certain node and it is computed by using the same formula as the standard deviation.

$$Entropy = \sum_{i=1}^{c} -p_i * log_2(p_i)$$

Random Forest

Random forests are a form of meta-estimator employed in machine learning. It is created by training several decision tree classifiers on various subsets of the dataset. The predictions of these classifiers are then averaged to enhance prediction accuracy and prevent overfitting. To improve the accuracy of a dataset, one may utilize a classifier such as Random Forest. This classifier employs an ensemble approach by aggregating the predictions of many decision trees, each trained on different subsets of the dataset. It combines many classifiers to address difficult issues and improve the accuracy of the model. The procedures necessary for putting the Random Forest algorithm into practice are.

Step 1: Subset of the features data utilized for each decision tree. A random sample of n records is taken from a collected data consisting of k records, each with m attributes.

Step 2: Each sample has its own set of decision trees built just for it.

Step 3: Generate an output on each decision tree.

Step 4: For both regression and classification, the final result is determined by majority vote or average.

E. Model Tuning and Evaluation

The proposed experiment utilizes the cross-k-fold validation approach. K-fold cross-validation is a technique used to assess the efficacy of prediction models. The original dataset is divided into k folds or subsets. For each of the k iterations in the training and evaluation process, a distinct fold is used as the validation set. The estimated generalization performance of a model is determined by calculating the average of the performance metrics obtained from all folds. This approach enables enhanced model evaluation, selection, and hyperparameter tweaking. Grid Search was employed to systematically explore all potential combinations of hyperparameters, as each machine learning classifier possesses its own distinct set of tuning parameters. A series of tests were conducted to evaluate the most effective strategy for reducing the feature space in each modified learning classifier. The results are put to use by comparing the performance of linked systems with those of the various output learning models. Because it treats each category the same, the macro-average is the recommended measure. In contrast to a micro-average, which combines all classes' contributions into one score, this method calculates the f1 metric for each class separately before averaging them.

F. Algorithm Steps

Input:

Network CSV Files from Monday to Friday Working Hours

Output:

Attack Types – {Attacks (Bot, Brute-Force, DoS, Infiltration, Port-Scan, Web-Attack) or Benign}

Procedure:

1. Load all *.csv file of attacks
2. Combine all attacks data into single *.csv file
3. Apply data pre-processing
4. Apply feature transformation, scaling, label encoding
5. Feature dataset splitting into train and test set
6. Initialize the parameters for DL/ML algorithms
7. Train and validate the model with fine tuning of hyperparameters and k-fold cross validation
8. Prepare trained model for each algorithm
9. Load test feature datasets
10. Evaluation of results to predict the attack types

3. EXPERIMENTAL RESULT AND DISCUSSION

A. Experimental Setup

The experimental evaluation conducted on a system with MAC OS Ventura-13.4.1, 8GB RAM and MAC operating system. The CICIDS 2017 is the benchmark dataset utilized for all classification experiments to perform multiclass intrusion classification. This research was conducted using the Pycharm Integrated Development Environment (IDE) software, which was built using the Anaconda Distribution and programmed in Python. The open-source libraries Scikit-learn, Tensorflow with Keras employed for the construction and assessment of the classifier.

B. Evaluation Metrics

Since the CICIDS2017 dataset is distributed in several different CSV files, we began our preparation by combining all of the files into a single CSV, cleaning up the column names by deleting any characters that didn't belong, and converting all textual information to numerical form. Records that had the phrase "Infinity" in the middle of a series of numbers were deemed to be irrelevant, therefore they were omitted. The ANOVA f-test technique used on the entire dataset to identify features. To train the classifiers, Scikit-learn libraries used with a unique combination of hyperparameters. The CICIDS2017 dataset was transformed into a multiclass classification problem since it contains a large number of attack labels. This research considered performance metrics such as Accuracy (Acc), Precision (Pr), Recall (Rc), and F1-score(F1). The metrics of Pr, Rc, and F1 were weighted and averaged in this study.

$$Precision\ (Pr) = Tp / (Tp + Fp)$$

$$Recall\ (Rc) = Tp / (Tp + Fn)$$

$$F1\text{-}Score\ (F1) = (2 * Pr * Rc) / (Pr + Rc)$$

Where, Tp – Positive instances that are true; Fp - Positive instances that are false; Tn - Negative instances that are true; Fn - Negative instances that are false.

C. Result Evaluation

Using the CICIDS2017 dataset, the experiments are accomplished to get more about the usefulness of the low-dimensional characteristics extracted and to evaluate the efficiency of the suggested framework in the categorization of many types of intrusions. In figure 3, the ANOVA test's score value plot can be visualized for all feature sets, indicating that features with a high f-test value are preferred for model learning and validation.

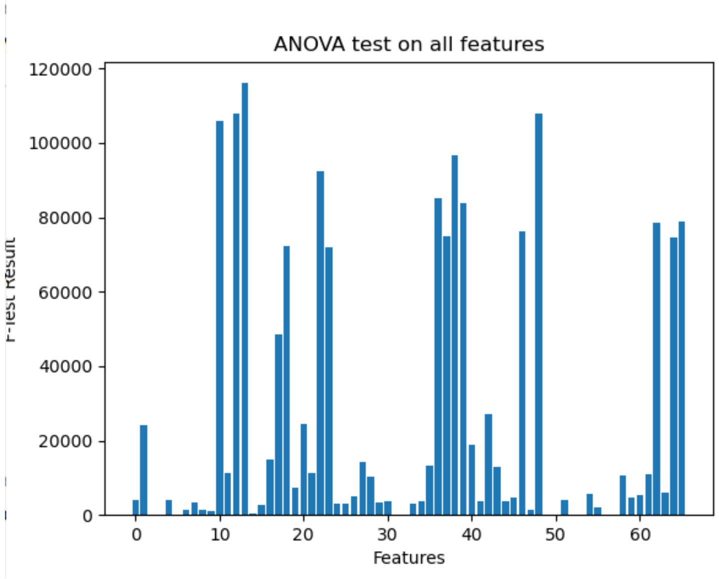
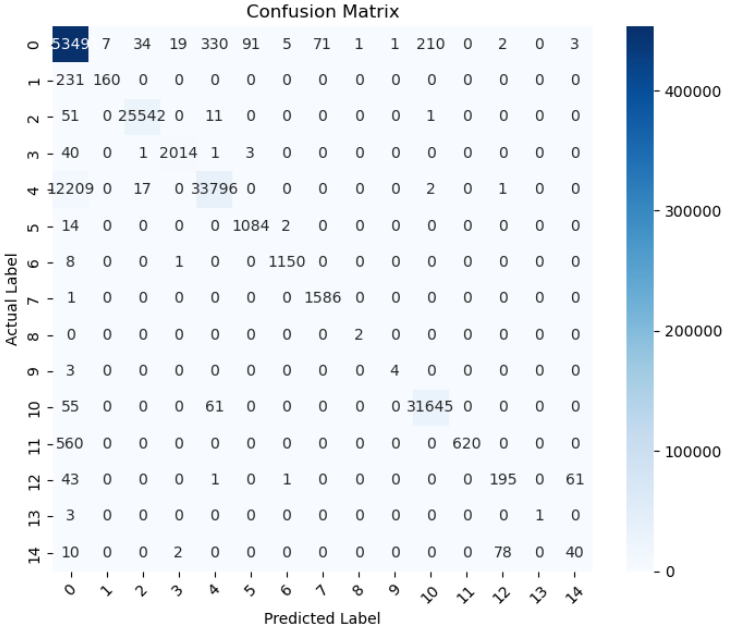


Figure 2: F-test score plot using ANOVA

The performance of the model was evaluated based on its ability to differentiate between 15 distinct categories, using metrics such as recall, precision, and F1 score. The model's effectiveness was determined using five-fold cross-validation. Each of the 5 subsets is divided into 70% train data and 30% test data. The metrics used to evaluate the model were informed by the confusion matrices obtained from each of the 5 cross-validation splits. The three learning models are employed to simulate the suggested framework and evaluate its performance on the confusion matrix. In multiclass intrusion classification, where 0-14 represents the aforementioned attack types, the confusion matrix performance of the three classifiers algorithm using DT, RF, and SD-RNN classifier is shown in figures 3, 4, and 5.


Confusion Matrix

Actual Label \ Predicted Label	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0	5349	7	34	19	330	91	5	71	1	1	210	0	2	0	3
1	231	160	0	0	0	0	0	0	0	0	0	0	0	0	0
2	51	0	25542	0	11	0	0	0	0	0	1	0	0	0	0
3	40	0	1	2014	1	3	0	0	0	0	0	0	0	0	0
4	12209	0	17	0	33796	0	0	0	0	0	2	0	1	0	0
5	14	0	0	0	0	1084	2	0	0	0	0	0	0	0	0
6	8	0	0	1	0	0	1150	0	0	0	0	0	0	0	0
7	1	0	0	0	0	0	0	1586	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0
9	3	0	0	0	0	0	0	0	0	4	0	0	0	0	0
10	55	0	0	0	61	0	0	0	0	0	31645	0	0	0	0
11	560	0	0	0	0	0	0	0	0	0	0	620	0	0	0
12	43	0	0	0	1	0	1	0	0	0	0	0	195	0	61
13	3	0	0	0	0	0	0	0	0	0	0	0	0	1	0
14	10	0	0	2	0	0	0	0	0	0	0	0	78	0	40

Figure 3: Confusion matrix using DT Classifier

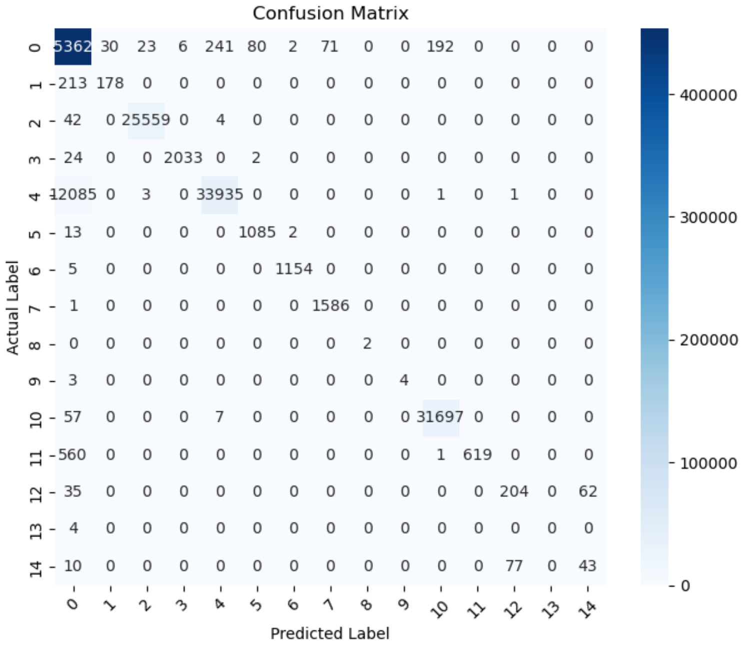


Figure 4: Confusion matrix using RF Classifier

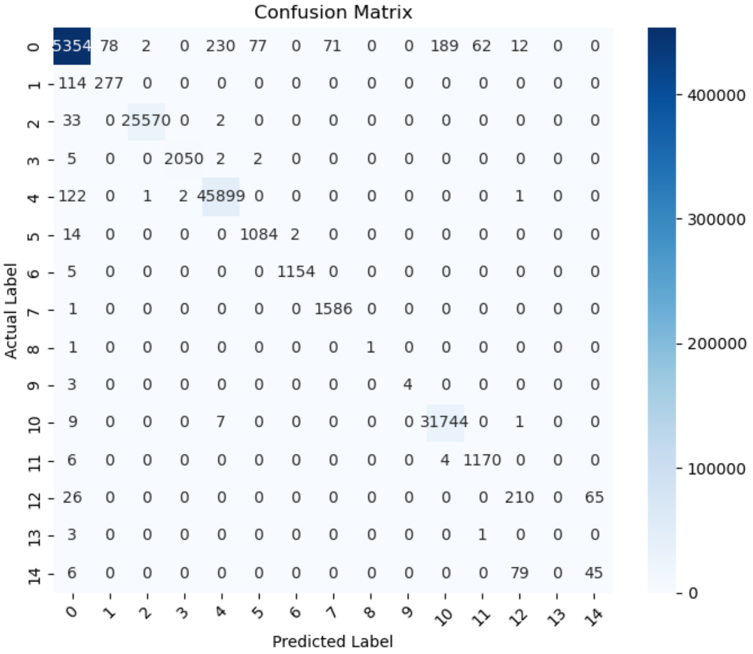


Figure 5: Confusion matrix using SD-RNN Classifier

Figures 6, 7, and 8 present a comprehensive overview of the proposed framework's performance in terms of recall, precision, and f1-score for all 15 attack categories. The DT, RF, and SD-RNN classifiers were used for the evaluation. Based on the figure, it can be observed that the incorrect categorization is also identified, with a higher proportion of traffic categorized as an attack compared to traffic classified as benign. Some assault samples had a smaller number of records compared to other attack samples, yet all of them were successfully recognized. The total attack categorization achieved with SD-RNN outperforms that of other machine learning models.

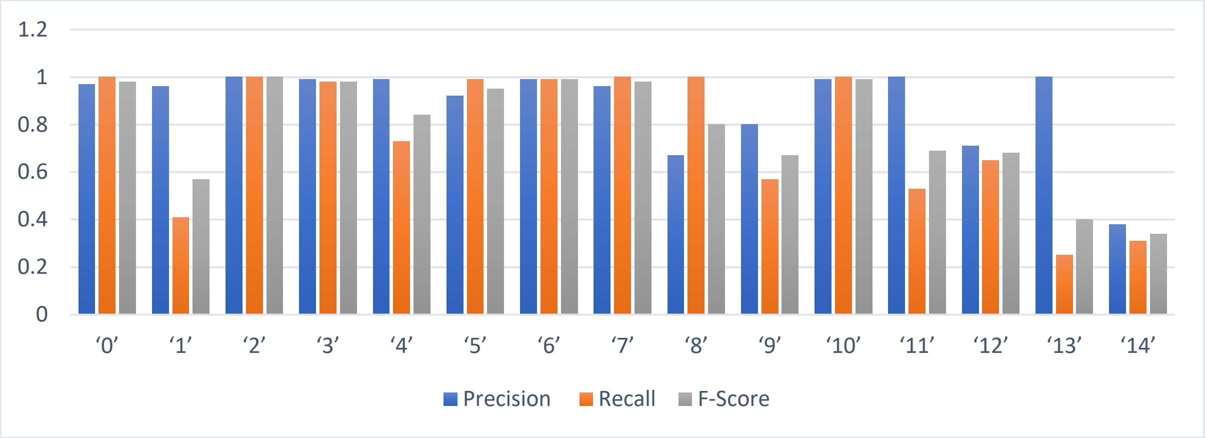


Figure 6: Performance evaluation using DT Classifier

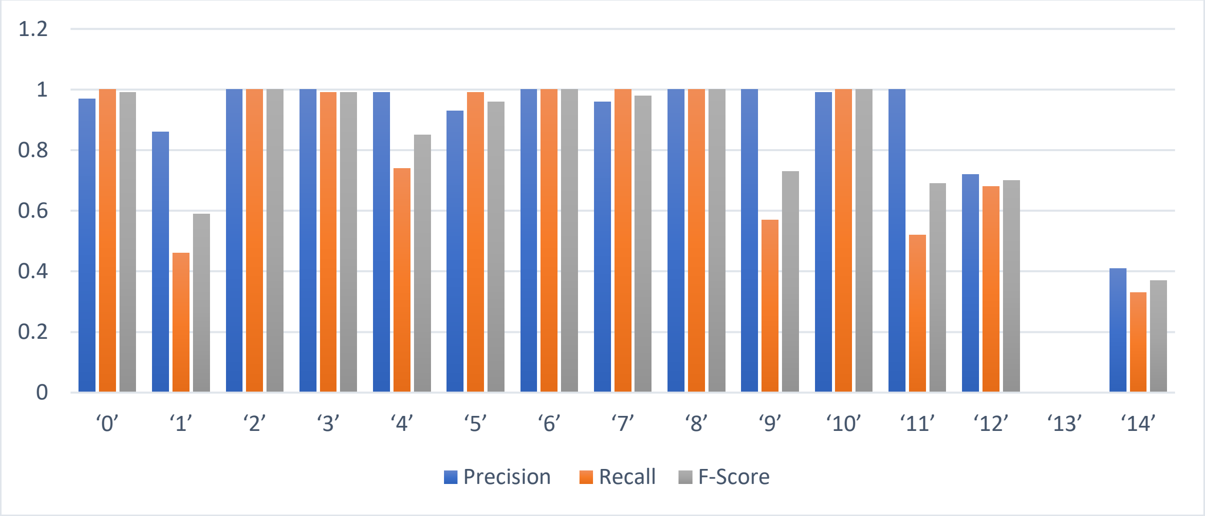


Figure 6: Performance evaluation using RF Classifier

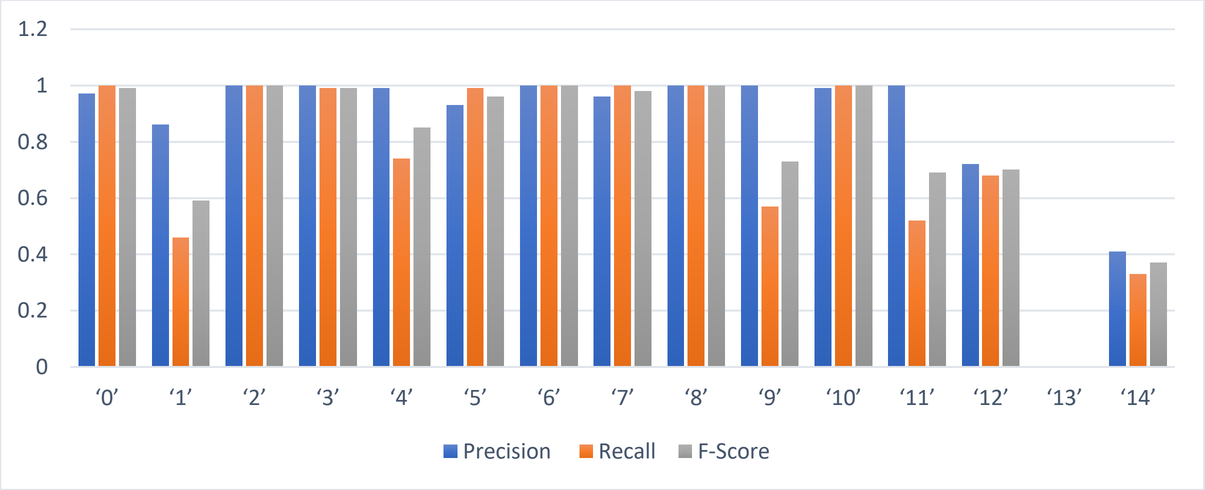


Figure 6: Performance evaluation using SD-RNN Classifier

Table 2 presents a comparison of all-classifier models in terms of accuracy, precision, recall, and f-score parameters for multiclass intrusion classification. The SD-RNN deep learning system presented achieves superior performance compared to the DT and RF classifiers in terms of accuracy, precision, recall, and f1-score. It has an average accuracy rate of 99.78%, precision rate of 99.77%, recall rate of 99.78%, and f-score rate of 99.77%.

Table 2: Average Result Performance Evaluation

Parameters/Classifiers	DT	RF	SD-RNN
Accuracy (%)	97.48	97.54	99.78
Precision (%)	97.51	97.61	99.77
Recall (%)	97.48	97.58	99.78
F-score (%)	97.39	97.38	99.77

D. Comparative Analysis

To demonstrate the effectiveness of proposed model, a comparison conducted with the classification methods employed in existing studies [12][15][19][21-22] using the CICIDS2017 Dataset. This comparison was particularly focused on the multi-class intrusion classification scenario. Table 3 displays a comparative analysis of the proposed deep learning framework, SD-RNN, with other relevant research and several classic DL/ML approaches stated in their publication. Although comparing the suggested work with similar research is not a simple task, it demonstrates efficient performance despite using a relatively modest model for categorization.

Table 3: Comparative Analysis

Ref Model	Accuracy	Precision	Recall	F-score
[12]	0.979	-	-	-
[15]	0.97	0.97	0.97	0.97
[19]	0.9888	0.9622	0.9566	0.9644
[21]	0.9699	0.9677	0.9456	0.9675
[22]	0.9866	0.8536	0.8424	0.8554
Proposed Model	0.9978	0.9978	0.9977	0.9977

4. CONCLUSION

With the proliferation of sophisticated networks and software, cyber security has emerged as a pressing concern in the IT industry. Since people constantly reveal private information online, security of this kind is paramount. That's why, an intelligent intrusion detection system will be useful. Developers face a difficult choice when faced with a wide variety of potential solutions to a given issue area. This paper involved the creation and implementation of a sophisticated deep learning and machine learning model. The model's purpose was to detect and classify malicious behavior in network data, accurately assigning it to one of 15 specific attack categories. The several tests are conducted on the CICIDS2017 dataset, which is generally acknowledged as a research standard, to evaluate the suggested technique. To evaluate the proposed model's capacity to detect attack data, both normal traffic data and a subset of attack data representing 14 common forms of attacks were chosen. Data analysis and pre-processing allowed us to drastically significantly decrease the number of features inputted into the model without compromising accuracy. The good recall and accuracy achieved by employing over-sampling and down-sampling approaches to find minority classes with relatively few samples in the original dataset. The suggested model is both very straightforward and compact, and it yields highly encouraging results in a multiclass classification task. Based on these findings, we may conclude that the suggested IDS is 99.73% accurate when using the SD-RNN model. When compared to previous studies, our SD-RNN model clearly achieves the best accuracy. In the future, the deep learning algorithms can be implemented on the data from live networks traffic data. In addition, more research may examine feature reduction strategies to further minimize the input characteristics.

REFERENCES

[1] Al Lail, Mustafa, Alejandro Garcia, and Saul Olivo. 2023. "Machine Learning for Network Intrusion Detection—A Comparative Study" Future Internet 15, no. 7: 243. https://doi.org/10.3390/fi15070243.

[2] Zeeshan Ahmad, Adnan Shahid Khan, Cheah Wai Shiang, Johari Abdullah, Farhan Ahmad, "Network intrusion detection system: A systematic study of machine learning and deep learning approaches", Trans Emerging Tel Tech. 2021; 32:e4150. https://doi.org/10.1002/ett.4150.

[3] Leevy, J.L., Khoshgoftaar, T.M. "A survey and analysis of intrusion detection models based on CSE-CIC-IDS2018 Big Data". J Big Data 7, 104 (2020). https://doi.org/10.1186/s40537-020-00382-x.

[4] Emad E. Abdallah, Wafa' Eleisah, Ahmed Fawzi Otoom, Intrusion Detection Systems using Supervised Machine Learning Techniques: A survey, Procedia Computer Science, Volume 201, 2022, Pages 205-212, ISSN 1877-0509, https://doi.org/10.1016/j.procs.2022.03.029.

[5] Yakub Kayode Saheed, Aremu Idris Abiodun, Sanjay Misra, Monica Kristiansen Holone, Ricardo Colomo-Palacios, A machine learning-based intrusion detection for detecting internet of things network attacks, Alexandria Engineering Journal, Volume 61, Issue 12, 2022, Pages 9395-9409, ISSN 1110-0168, https://doi.org/10.1016/j.aej.2022.02.063.

[6] H. Güney, "Preprocessing Impact Analysis for Machine Learning-Based Network Intrusion Detection", Sakarya University Journal of Computer and Information Sciences, vol. 6, no. 1, pp. 67-79, Apr. 2023, doi:10.35377/saucis...1223054

[7] A. Abubakar and B. Pranggono, "Machine learning based intrusion detection system for software defined networks," 2017 Seventh International Conference on Emerging Security Technologies (EST), Canterbury, UK, 2017, pp. 138-143, doi: 10.1109/EST.2017.8090413.

[8] E. Yusuf Güven, S. Gülgün, C. Manav, B. Bakır and G. Zeynep Gürkaş Aydın, "Multiple classification of cyber-attacks using machine learning," Electrica., 22(2), 313-320, 2022.

[9] Goryunov, Maxim & Matskevich, Andrey & Rybolovlev, Dmitry. (2020). Synthesis of a Machine Learning Model for Detecting Computer Attacks Based on the CICIDS2017 Dataset. Proceedings of the Institute for System Programming of the RAS. 32. 81-94. 10.15514/ISPRAS-2020-32(5)-6.

[10] Baha Rababah, Srija Srivastava, "Hybrid Model For Intrusion Detection Systems", Cryptography and Security, https://doi.org/10.48550/arXiv.2003.08585

[11] SILVA NETO, Manuel G. da; G. GOMES, Danielo., "Network Intrusion Detection Systems Design: A Machine Learning Approach", In: SIMPÓSIO BRASILEIRO DE REDES DE COMPUTADORES E SISTEMAS DISTRIBUÍDOS (SBRC), 37., 2019. ISSN 2177-9384. DOI: https://doi.org/10.5753/sbrc.2019.7413.

[12] Nasheeda P.P and Anusree B.2022, "An Intrusion Detection System Based on CNN-LSTM Hybrid Network On Cicids2017 Dataset". Int J Recent Sci Res. 13(07), pp. 1881-1888. DOI: http://dx.doi.org/10.24327/ijrsr.2022.1307.0394

[13] Hatitye Chindove and Dane Brown. 2021. "Adaptive Machine Learning Based Network Intrusion Detection". In Proceedings of the International Conference on Artificial Intelligence and its Applications (icARTi '21). Association for Computing Machinery, New York, NY, USA, Article 15, 1–6. https://doi.org/10.1145/3487923.3487938.

[14] Chongzhen Zhang, Yanli Chen, Yang Meng, Fangming Ruan, Runze Chen, Yidan Li, Yaru Yang, "A Novel Framework Design of Network Intrusion Detection Based on Machine Learning Techniques", Security and Communication Networks, vol. 2021, Article ID 6610675, 15 pages, 2021. https://doi.org/10.1155/2021/6610675

[15] Pengfei Sun, Pengju Liu, Qi Li, Chenxi Liu, Xiangling Lu, Ruochen Hao, Jinpeng Chen, "DL-IDS: Extracting Features Using CNN-LSTM Hybrid Network for Intrusion Detection System", Security and Communication Networks, vol. 2020, Article ID 8890306, 11 pages, 2020. https://doi.org/10.1155/2020/8890306

[16] G. Çetin, "An effective classifier model for imbalanced network attack data," Computers, Materials & Continua, vol. 73, no.3, pp. 4519–4539, 2022.

[17] P. Toupas, D. Chamou, K. M. Giannoutakis, A. Drosou and D. Tzovaras, "An Intrusion Detection System for Multi-class Classification Based on Deep Neural Networks," 2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA), Boca Raton, FL, USA, 2019, pp. 1253-1258, doi: 10.1109/ICMLA.2019.00206.

[18] Panwar, Lokesh & Panwar, Shailesh. (2019). "Implementation of Machine Learning Algorithms on CICIDS-2017 Dataset for Intrusion Detection using WEKA". 8. 2195. 10.35940/ijrte.C4587.098319.

[19] Jose, Jinsi & Jose, Deepa. (2023). Deep learning algorithms for intrusion detection systems in internet of things using CIC-IDS 2017 dataset. International Journal of Electrical and Computer Engineering (IJECE). 13. 1134. 10.11591/ijece.v13i1.pp1134-1141.

[20] Akshay Kumaar M, Samiayya D, Vincent PMDR, Srinivasan K, Chang CY, Ganesh H. A Hybrid Framework for Intrusion Detection in Healthcare Systems Using Deep Learning. Front Public Health. 2022 Jan 12;9:824898. doi: 10.3389/fpubh.2021.824898. PMID: 35096763; PMCID: PMC8790147.

[21] Zachariah Pelletier and Munther 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, Volume 5, Issue 2, pp. 187-191, 2020.

[22] 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," in IEEE Access, vol. 9, pp. 22351-22370, 2021, doi: 10.1109/ACCESS.2021.3056614.

[23] Elnakib, O., Shaaban, E., Mahmoud, M. et al. "EIDM: deep learning model for IoT intrusion detection systems". J Supercomput 79, 13241–13261 (2023). https://doi.org/10.1007/s11227-023-05197-0.

[24] Rodríguez, María, Álvaro Alesanco, Lorena Mehavilla, and José García. 2022. "Evaluation of Machine Learning Techniques for Traffic Flow-Based Intrusion Detection" Sensors 22, no. 23: 9326. https://doi.org/10.3390/s22239326.

[25] Singh Panwar, Shailesh and Raiwani, Y. P. and Panwar, Lokesh Singh, Evaluation of Network Intrusion Detection with Features Selection and Machine Learning Algorithms on CICIDS-2017 Dataset (March 15, 2019). International Conference on Advances in Engineering Science Management & Technology (ICAESMT) - 2019, Uttaranchal University, Dehradun, India, http://dx.doi.org/10.2139/ssrn.3394103.

[26] Han, Hyojoon, Hyukho Kim, and Yangwoo Kim. 2022. "An Efficient Hyperparameter Control Method for a Network Intrusion Detection System Based on Proximal Policy Optimization" Symmetry 14, no. 1: 161. https://doi.org/10.3390/sym14010161

[27] T. Ferrão, F. Manene and A. A. Ajibesin, "Multi-attack intrusion detection system for software-defined internet of things network," Computers, Materials & Continua, vol. 75, no.3, pp. 4985–5007, 2023.

[28] Abbas, M.I., "Picard and Picard-Krasnoselskii iteration methods for generalized proportional Hadamard fractional integral equations". (2022) Advances in the Theory of Nonlinear Analysis and its Applications, 6 (4), pp. 538-546.

[29] Vijay, Chand, A.K.B., "Zipper Fractal Functions with Variable Scalings" (2022) Advances in the Theory of Nonlinear Analysis and its Applications, 6 (4), pp. 481-501.

[30] Goyal, Dinesh , Kumar, Anil , Gandhi, Yatin & Khetani, Vinit (2024) Securing wireless sensor networks with novel hybrid lightweight cryptographic protocols, Journal of Discrete Mathematical Sciences and Cryptography, 27:2-B, 703–714, DOI: 10.47974/JDMSC-1921