# A Novel AI-Based Stacking Of optimized Heterogeneous Neural Networks with hyper-parameters tuning For Detecting real-time Multi-Class Zero-Day Attacks in IOT

**V. Kanimozhi[1]*, Dr. T. Prem Jacob[2]**

[1,2] Department of CSE, Sathyabama Institute of Science and Technology, Chennai 600119, India

Email: [1]kanimv@yahoo.co.in, [2]premjac@yahoo.com

**Abstract**

The detection of IoT and internet traffic real-time zero-day multi-class attacks (novel or unseen attacks) are clearly explained in this research proposal. The proposed Stacking of four heterogeneous neural networks with a special type of Neural Network Autoencoder with ensemble Machine Learning Random Forest classifier delivers the best accuracy ratings and F1 Score of 0.999896 and 0.9998898 with the most minor loss function and the quickest execution times. Five heterogeneous neural networks with bagging machine learning make up this novel proposed stacking ensemble model. The highest scores were determined by comparing and evaluating the ensemble Random Forest Classifier with other ML classifiers, extreme Gradient Boosting, and Support Vector Machine are included. (XGB Boost), and the Naive Bayes. The applications of the main AI-deep learning models, ML classifiers, stacked deep learning models, and Stacked Ensemble Neural Network models with ML are then shown in this research analysis experimented over more than 20 lakhs of dataset instances on the realistic cyber and IoT datasets, which helps illuminate how different AI models are implemented for detecting zero-day attacks in network intrusion detection systems. Utilizing cutting-edge AI by implementing the proposed Stacking, Ensemble Stacking of DL, and ML Neural Networks with Feature Extractor significantly improves anomaly detection in identifying zero-day attacks. Therefore, it would effectively lessen attacks on IoT and cyber-security firms.

Keywords: Artificial Neural Networks, Stacked ensemble Heterogeneous Models, IoT, Zero-day multi-class attacks detection, Accuracy, F1 Score

## 1. Introduction

IoT gadgets are prone to cyber assaults extra than network attacks because an innumerable number of IoT devices have emerged in these recent days. IoT gadgets are prone to safety threats. A system or network's harmful behavior or policy violations are searched for using the software. A breach or illegal behavior is often noted by an administrator. Intrusion detection systems are prone to raise false warnings when monitoring networks for potentially dangerous behavior. As a result, businesses need to modify their IDS products after the first installation. IDSs must be appropriately configured to differentiate between malicious activities and legitimate network traffic.

Even though ML methods existed, the present article aims to boost the benefits of IoT devices by applying emerging and effective AI-DL technologies and involving the realistic dataset IoT-23 big data of network traffic. Deploying the major Supervised DL algorithmic strategies of Convolutional Neural Network, Multi-Layer Perceptron, produces a comparative and detailed analysis, of an IoT-23 dataset to deliver a remarkable accuracy score and minimal loss of IoT anomaly detection rate.

## 2. Iot-23 Dataset

IoT-23 as a component of the "Avast Software" subsidizing, can be found on the web link, "https://www.stratosphereips.org/datasets-iot23", is a network of IoT dataset records of 8.8GB that carries major 3 benign subsets are 'Sub_cat', 'cat' and 'label'. Also aims to provide a large dataset of categorized benign and malware records originating from actual captures to increase attack identification to create proposing a system to know DL algorithms [1,2].

## 3. Exploratory Data Analysis Of An Iot-23 Dataset

IoT-23 dataset - There are 86 features and the dataset shape is [625783 rows x 86 columns] which carries 3 feature targets 'benign' columns. They are 'Label' (binomial classification), 'Sub_Cat' (multiclass classification), and 'Cat' (multilabel classification). Among these three target variables, one is 'Label', another is 'Category' and the third target feature is 'Subcategory'. In this article, the 'Label' feature has been deployed for binary classification in various DL Algorithms [3].

### 3.1. IoT Label 'Category' Description

Second, categorize the 'Category' feature target column.

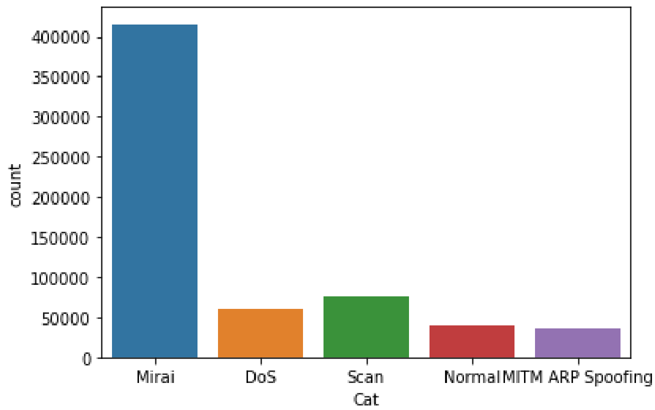


**Fig.1.** Category Feature of IoT-23 dataset

1. Total DoS attacks - 75265
2. Total Mirai attacks - 415677
3. Total MITM_ARP attacks - 35377
4. Total Normal records - 40073
5. Total NScan attacks - 59391

- Mirai attack: "Mirai" is a new malware focused on IoT devices such as printers, camcorders, switches, and smart Televisions are spreading. The malware examines the local area devices or Internet of Things and tests out to bargain those designs, specifically the ones covered with default qualifications or hardcoded username passwords

- DoS attack: A DOS attack is the most common and easiest for implementing attacks on IoT devices [4]. Further, IoT devices cannot access the rest of the network for communication when a DoS attack occurs [5].
- Scan attack: The open port of IoT services is scanned by this IoT Scan, and malware can easily gain access to these IoT products.
- Normal: Records without any IoT attacks or malware.
- MITM_ARP attack: MITM is a Man in Middle assault wherein the attacker (hacker) sends solid ARP Messages [6]. This permits the attacker to be faux as a valid consumer because it connects the legitimate IP Address to the MAC Address of the attacking machine. After connecting, the attacker would then obtain messages intended for the correct IP Address[7]. Furthermore, ARP Spoofing permits the attacker can intercept, modify, and relinquish the incoming messages [8].

## 4. Addressing An Unbalanced Set For A Classification Issue With Smote

"Imbalanced data" refers to pieces of data where there are disproportionately more observations for one class label than there are for the other class name [9]. This dataset has a large number of anomalous attacks (585710) compared to normal records (40073). This disparity is shown graphically in Figure 9.1. Our target class appears unbalanced, according to observations [10]. Let's strive to increase the sample size of data for making the minority class more representative of the majority class [11].

Unbalanced data classification issues are described. The key issue with unbalanced dataset prediction, to put it simply, is how well we truly forecast both the majority and minority classes. When there are considerably more records in one class than in the other, our classifier may sometimes become biased in favor of the prediction[12]. The current challenge is that the model predicts 100 percent accuracy ie. overfitting. The technique for systematically oversampling underrepresented groups is the SMOTE (Synthetic Minority Oversampling Technique). Adding duplicate minority class records to the model is frequently ineffective in introducing new data [13,14]. The existing data is used to create new instances in SMOTE. To put it simply, SMOTE analyses minority class cases and picks a random closest neighbor from among them using k nearest neighbors to produce a synthetic instance at random in feature space.

Code snippet of the resampling technique below, which resamples the dataset –

*from imblearn.over_sampling import SMOTE*

*X_smote,y_smote = smote.fit_resample(X,y)*

Thus, the resampling method is used in this research and can be used while handling an imbalanced dataset.

## 5. Proposed Method Of Stacking Ensemble Of Deep Learning For Multi-Class Attack Detection

Autoencoder - Developing a Neural Network for Feature Extraction

Let's develop an autoencoder model for feature extraction for the realistic Intrusion data.

The two distinct parts of the encoders and decoders together make up an autoencoder.

$$\boldsymbol{\Phi : D \rightarrow £}$$
$$\boldsymbol{\psi : £ \rightarrow D}$$
$$\boldsymbol{\Phi, \psi = \arg\min_{\Phi, \psi} \| D - (\psi , \Phi) D \|_2} \qquad \textbf{(5.1)}$$

Through Φ the function of an encoder, the original data D is translated to the latent space £. The decoder function then grabs the latent representation ψ to produce the outputs.
As a result, the loss function can be expressed in terms of these network functions. The output from PCA directly correlates to the latent vector representation if the autoencoder is linearly enabled. But typically, autoencoders' activation functions sigmoid and ReLU are nonlinear. Using all of the input columns, the autoencoder model will output the same results. It will eventually master precise replication of the input pattern. The encoder and decoder are their two constituent pieces. The encoder gains knowledge on how to decode the input and compress it into a bottleneck-specified internal representation. The decoder attempts to replicate the input by using the same number of inputs as the encoder's outputs.

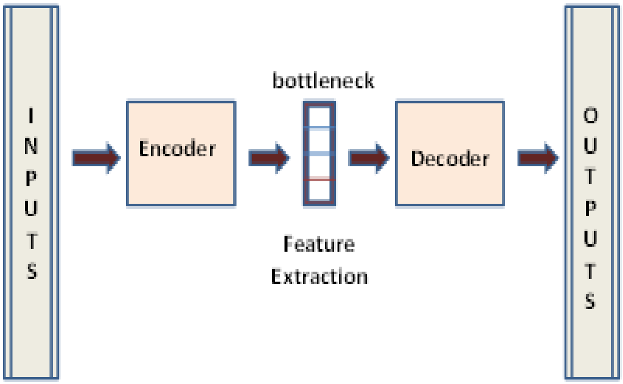


**Fig. 2.** Feature Extraction Encoder Model

Only the encoder portion of the autoencoder is kept after the autoencoder has been trained and is utilized to compress input vector example examples into vectors produced by the bottleneck. For this incursion dataset with shape (80), the encoder can be defined as a pair of hidden layers, the first with twice as many inputs, and the next with the shape (80), followed by the bottleneck with only the extracted encoded dataset features. To ensure the model learns well, we will use batch normalization and an activation 'relu', to compile and fit the model to the dataset. To define an encoder model,
*encoder_model = Model (inputs = inputs, outputs = bottleneck)*
Now create the encoder model and save it as encoder_model.h5, the extracted features as X_train_encode and X_test_encode can be extracted from this model. These extracted features can be implemented into the ensembling stacking models.

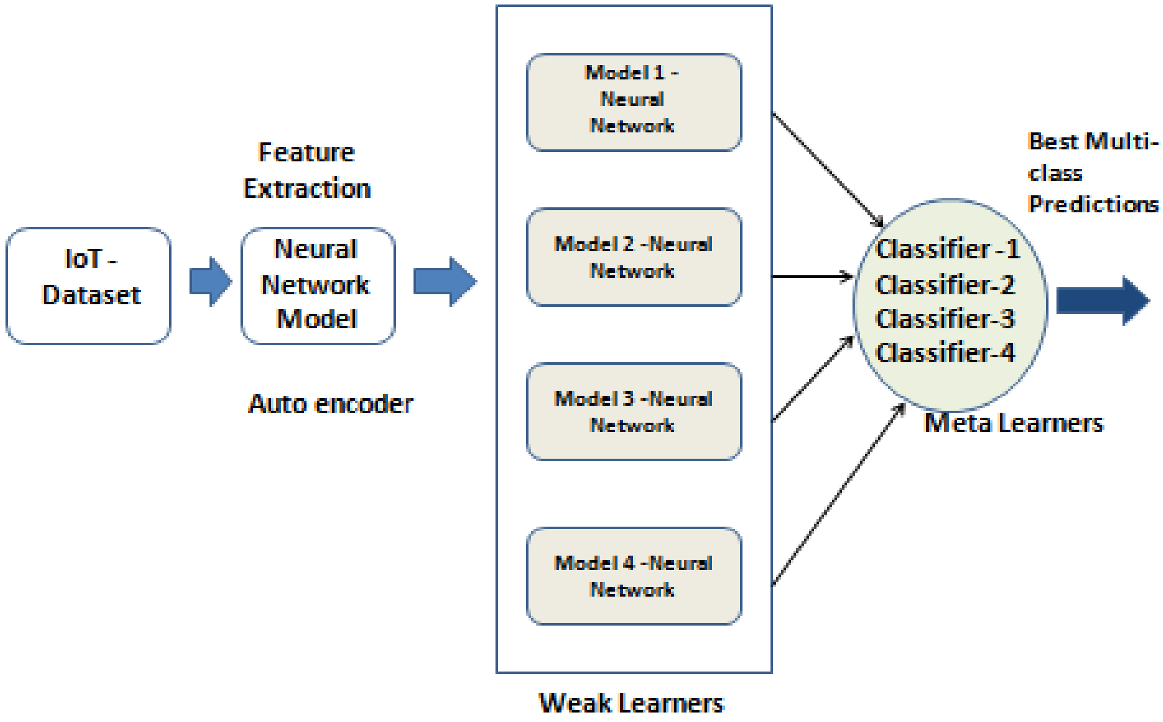


**Fig.3.** Proposed Stacked Ensemble Model with 5 Neural Networks and meta learners for Multiclass Anomaly Detection

### 5.1 Algorithm steps to create a Stacked Ensemble with heterogeneous Neural Networks and meta-learner classifiers.

Step 1: Input Intrusion data into the loaded encoder.h5 and get the top extracted features as X_train_encode and X_test_encode [15].

Step 2: Building a simple Neural Networks design - The equation for the neural network is formed by combining the independent variables linearly with the corresponding weights and bias factors for each neuron [16]. The Neural Network model architecture has been compiled by utilizing the optimizer to be Adam, the loss function to be categorical cross-entropy, and the metric to be Accuracy and F1-score [17].

Step 3: Create a performance recording for the simple model [18]. Likewise, build other three distinct neural network architectures by tuning the hyper-parameters like increasing or decreasing the hidden layers or the hidden nodes, changing the epochs value, and learning rate of the optimizer, using the different settings to train each of them. save the model like model1.h5, model2.h5, etc...

Step 4: Load the models from the saved h5 files. By giving instances from the test set to the four neural networks, it generates a model input stacked dataset(more than 20 lakhs of records) by stacking the train and test datasets [19]. Now the stacked dataset count is 20,78,385 records. The model architecture runs with count 1662708 as training and 415677 as testing records with 10 epochs.

Step 5: Gather the predictions to train the meta-learner classifiers like SVM, Naive Bayes, XGB boost, and Random Forest [20]. Each ensemble will produce a five-class anomaly and normal attack in this scenario, and the ensemble models with greater accuracy scores are evaluated and determined [21].

**Table.1**. Model Training on Proposed Stacked Ensemble by optimized heterogeneous neural networks, Autoencoder with Ensemble Classifier

| |
|---|
| Train on 1662708 records, validate on 415677 records |
| Epoch 1/10 |
| 1662708/1662708 [==============================] - 83s 50us/sample - loss: 0.4860 - acc: 0.8172 - f1_score: 0.9405 - precision_m: 0.9977 - recall_m: 0.8957 - val_loss: 0.3734 - val_acc: 0.8626 - val_f1_score: 0.9743 - val_precision_m: 1.0000 - val_recall_m: 0.9505 |
| Epoch 2/10 |
| 1662708/1662708 [==============================] - 78s 47us/sample - loss: 0.3675 - acc: 0.8633 - f1_score: 0.9737 - precision_m: 1.0000 - recall_m: 0.9494 - val_loss: 0.3346 - val_acc: 0.8710 - val_f1_score: 0.9806 - val_precision_m: 1.0000 - val_recall_m: 0.9625 |
| Epoch 3/10 |
| 1662708/1662708 [==============================] - 78s 47us/sample - loss: 0.3360 - acc: 0.8740 - f1_score: 0.9783 - precision_m: 1.0000 - recall_m: 0.9581 - val_loss: 0.3497 - val_acc: 0.8727 - val_f1_score: 0.9871 - val_precision_m: 1.0000 - val_recall_m: 0.9750 |
| Epoch 4/10 |
| 1662708/1662708 [==============================] - 78s 47us/sample - loss: 0.3202 - acc: 0.8800 - f1_score: 0.9804 - precision_m: 1.0000 - recall_m: 0.9621 - val_loss: 0.2866 - val_acc: 0.9057 - val_f1_score: 0.9734 - val_precision_m: 1.0000 - val_recall_m: 0.9490 |
| Epoch 5/10 |
| 1662708/1662708 [==============================] - 77s 47us/sample - loss: 0.3101 - acc: 0.8845 - f1_score: 0.9817 - precision_m: 1.0000 - recall_m: 0.9646 - val_loss: 0.2784 - val_acc: 0.8927 - val_f1_score: 0.9854 - val_precision_m: 1.0000 - val_recall_m: 0.9717 |
| Epoch 6/10 |
| 1662708/1662708 [==============================] - 77s 47us/sample - loss: 0.3008 - acc: 0.8885 - f1_score: 0.9829 - precision_m: 1.0000 - recall_m: 0.9668 - val_loss: 0.2713 - val_acc: 0.9015 - val_f1_score: 0.9886 - val_precision_m: 1.0000 - val_recall_m: 0.9778 |
| Epoch 7/10 |
| 1662708/1662708 [==============================] - 78s 47us/sample - loss: 0.2928 - acc: 0.8921 - f1_score: 0.9848 - precision_m: 1.0000 - recall_m: 0.9704 - val_loss: 0.2654 - val_acc: 0.9049 - val_f1_score: 0.9875 - val_precision_m: 1.0000 - val_recall_m: 0.9757 |

| |
|---|
| Epoch 8/10 |
| 1662708/1662708 [==============================] - 79s 48us/sample - loss: 0.2857 - acc: 0.8954 - f1_score: 0.9857 - precision_m: 1.0000 - recall_m: 0.9722 - val_loss: 0.2696 - val_acc: 0.9008 - val_f1_score: 0.9840 - val_precision_m: 1.0000 - val_recall_m: 0.9689 |
| Epoch 9/10 |
| 1662708/1662708 [==============================] - 78s 47us/sample - loss: 0.2784 - acc: 0.8990 - f1_score: 0.9866 - precision_m: 1.0000 - recall_m: 0.9738 - val_loss: 0.2528 - val_acc: 0.9062 - val_f1_score: 0.9890 - val_precision_m: 1.0000 - val_recall_m: 0.9786 |
| Epoch 10/10 |
| 1662708/1662708 [==============================] - 79s 47us/sample - loss: 0.2731 - acc: 0.9014 - f1_score: 0.9870 - precision_m: 1.0000 - recall_m: 0.9746 - val_loss: 0.2512 - val_acc: 0.9130 - val_f1_score: 0.9883 - val_precision_m: 1.0000 - val_recall_m: 0.9773 |
| >loaded C:\Users\model1.h5 |
| >loaded C:\Users\model2.h5 |
| >loaded C:\Users\model3.h5 |
| >loaded C:\Users\model4.h5 |
| Loaded 4 models |
| Stacked_Accuracy_Score: 0.9998965542957633<br>Stacked F1_Score: 0.9998984412123435 |

The experimentations have been carried out with this stacked ensembled model into major meta-learner classifiers usually the previous research experimentations were carried out only with Logistic regression to check for the best ensemble multi-class prediction results [22] such as,

- ❖ Naive Bayes Classifier
- ❖ Support Vector Machine Classifier
- ❖ XGB Classifier
- ❖ Random Forest Classifier

**Table.2.** Metric Scores of Proposed Ensemble Stacking Neural Networks with Various Machine Learning Classifiers

| **Metrics** | **Basic MLP Model** | **Proposed Stacking Model with Naive Bayes** | **Proposed Stacking Model with SVM** | **Proposed Stacking Model with XGB** | **Proposed Stacking Model with Random Forest** |
|---|---|---|---|---|---|
| Accuracy | 0.910685 | 0.9498720 | 0.958003 | 0.988236 | 0.999896 |
| F1 value | 0.910774 | 0.9500038 | 0.958080 | 0.988235 | 0.999898 |

## 6. Conclusion and Future Enhancement

Henceforth the observations from the evaluation metrics proved that the Proposed Stacked Model increased the 8 percent accuracy zero-day multi-class attacks detection score more than the individual Artificial Neural Network Model and other ensemble Models. This enhanced Stacking Ensemble model can address the shortcomings of any stacking or individual model and also provides high accuracy than any other stacking or basic neural network model. To identify zero-day attacks in Anomaly Detection, the Proposed Stacking Ensemble with bagging Random Forest machine learning classifier performs better than individual neural networks or stacking network classifiers. The benefit of a stacking ensemble is that it may undertake a multi-class classification of novel attacks using a range of efficient models and get forecasts that outperform any individual or stacked models. It can be further enhanced by combining a fusion of algorithms like GANs, and optimizer as Nadam, etc.

References

[1] Kolias, C., Kambourakis, G., Stavrou, A., & Voas, J, "DDoS in the IoT: Mirai and other botnets". *Computer*, *50*(7), 80-84, 2017.

[2] Radanliev, P. "Future developments in cyber risk assessment for the internet of things". *Comput. Ind.*, *102*, 14–22, 2018.

[3] Bertino, E.; Islam, N.," Botnets and Internet of Things Security.", *50*, 76–79, 2017.

[4] Al-Garadi, M. A., Mohamed, A., Al-Ali, A. K., Du, X., Ali, I., & Guizani, M." A survey of a machine and deep learning methods for the Internet of things (IoT) security. *IEEE Communications Surveys & Tutorials*, *22*(3), 1646-1685", 2020.

[5] A Mirzaei, A Najafi Souha, "Towards optimal configuration in MEC Neural networks: deep learning-based optimal resource allocation" - Wireless Personal Communications – Springer,(2020).

[6] Tang, T. A., Mhamdi, L., McLernon, D., Zaidi, S. A. R., & Ghogho, M. Deep learning approach for network intrusion detection in software-defined networking. In *2016 international conference on wireless networks and mobile communications (WINCOM)* (pp. 258-263). IEEE. October 2016.

[7] A Gautam, S Singh. "Deep Learning Based Object Detection Combined with Internet of Things for Remote Surveillance"- Wireless Personal Communications – Springer.2021.

[8] Zhang, Q., Yang, L. T., Chen, Z., & Li, P. "A survey on deep learning for big data. *Information Fusion*, *42*, 146-157", 2018.

[9] Berman, D. S., Buczak, A. L., Chavis, J. S., & Corbett, C. L. " A survey of deep learning methods for cybersecurity. *Information"*, *10*(4), 122, 2019.

[10] Liu, W., Wang, Z., Liu, X., Zeng, N., Liu, Y., & Alsaadi, F. E "A survey of deep neural network architectures and their applications. *Neurocomputing*, *234*, 11-26.", 2017.

[11] S Bhardwaj, M Dave, "Crypto-Preserving Investigation Framework for Deep Learning Based Malware Attack Detection for Network Forensics- Wireless Personal Communications, – Springer", 2022.

[12] Milenkoski, A., Vieira, M., Kounev, S., Avritzer, A., & Payne, B. D.. "Evaluating computer intrusion detection systems: A survey of common practices. *ACM Computing Surveys (CSUR)*, *48*(1), 1-41, 2015.

[13] Hussain, F., Abbas, S. G., Fayyaz, U. U., Shah, G. A., Toqeer, A., & Ali, A." Towards a Universal Features Set for IoT Botnet Attacks Detection". *arXiv preprint arXiv:2012.00463*,2022.

[14] Ullah, I., & Mahmoud, Q. H. " A Scheme for Generating a Dataset for Anomalous Activity Detection in IoT Networks. In *Canadian Conference on Artificial Intelligence* (pp. 508-520). Springer, Cham", April 2020.

[15] Sakurada, M., & Yairi, T, "Anomaly detection using autoencoders with nonlinear dimensionality reduction. In *Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis* (pp. 4-11)", .2014, December.

[16] Foley, J., Moradpoor, N., & Ochen, H, "Employing a Machine Learning Approach to Detect Combined Internet of Things Attacks against Two Objective Functions Using a Novel Dataset". *Security and Communication Networks*, *2020*.

[17] Ferrag, M. A., Maglaras, L., Moschoyiannis, S., & Janicke, H, "Deep learning for cybersecurity intrusion detection: Approaches, datasets, and comparative study". *Journal of Information Security and Applications*, *50*, 102419, 2020.

[18] Saito, Y., Benjebbour, A., Kishiyama, Y., & Nakamura, T." System level performance evaluation of downlink non-orthogonal multiple access (NOMA), In Proc. *IEEE Annu. Symp. Pers. Indoor Mobile Radio Commun. (PIMRC)*, London, U.K., pp. 611–615",2013.

[19] Zenati, H., Foo, C. S., Lecouat, B., Manek, G., & Chandrasekhar, V. R. "Efficient gan-based anomaly detection. *arXiv preprint arXiv:1802.06222*",2018.

[20] Di Mattia, F., Galeone, P., De Simoni, M., & Ghelfi, E. "A survey on gans for anomaly detection. *arXiv preprint arXiv:1906.11632*",2019.

[21] M Premkumar, TVP Sundararajan, "Defense countermeasures for DoS attacks in WSNs using deep radial basis networks- Wireless Personal Communications, Springer",2021.

Kanimozhi, V., & Jacob, T. P. (2021). Artificial Intelligence outflanks all other machine learning classifiers in Network Intrusion Detection Systems on the realistic cyber dataset CSE-CIC-IDS2018 using cloud computing. *ICT Express*, *7*(3), 366-370.