JOURNAL OF CRITICAL REVIEWS

ISSN- 2394-5125 VOL 10, ISSUE 03, 2023

Toward Detection and Attribution of Cyber Attacks in IoT Enabled Cyber Physical Systems

N. Sravani[1], Sathya Pranathi Valluri[2], Chetan Sai Kosanam[2], Aavala Samuel Raj[2]

[1,2]Department of Information Technology

[1,2] CMR Engineering College, Kandlakoya, Medchal, Hyderabad.

ABSTRACT

Internet of Things (IoT) enabled cyber physical systems such as Industrial equipment's and operational IT to send and receive data over internet. This equipment's will have sensors to sense equipment condition and report to centralized server using internet connection. Sometime some malicious users may attack or hack such sensors and then alter their data and this false data will be report to centralized server and false action will be taken. Due to false data many countries equipment and production system got failed and many algorithms was developed to detect attack, but all these algorithms suffer from data imbalance (one class my contains huge records (for example NORMAL records and other class like attack may contains few records which lead to imbalance problem and detection algorithms may failed to predict accurately). To deal with data imbalance, existing algorithms were using OVER and UNDER sampling which will generate new records for FEWER class only. To overcome from this issue, we are introducing novel technique without using any under or oversampling algorithms. The proposed technique consists of 2 parts.

Auto encoder: It will get trained on imbalanced dataset and then extract features from it and these extracted features will get trained with DECISION TREE algorithm to predict label for known or unknown attacks. Decision tree get trained on reduced number of features obtained from PCA (principal component analysis) algorithm.

Deep Neural Network (DNN): In this level, DNN algorithm get trained on known and unknown attacks. If any records contain attack signature, then DNN will identify attack label or class and attribute them.

Keywords: Cyber-attacks, IoT, Cyber physical systems.

1. INTRODUCTION

Sensors are most commonly used in numerous applications ranging from body-parameters' measurement to automated driving. Moreover, sensors play a key role in performing detection- and vision-related tasks in all the modern applications of science, engineering and technology where the computer vision is dominating. An interesting emerging domain that employs the smart sensors is the Internet of Things (IoT) dealing with wireless networks and sensors distributed to sense data in real time and producing specific outcomes of interest through suitable processing. In IoT-based devices, sensors and artificial intelligence (AI) are the most important elements which make these devices sensible and intelligent. In fact, due to the role of AI, the sensors act as smart sensors and find an efficient usage for a variety of applications, such as general environmental monitoring [1]; monitoring a certain number of environmental factors; weather forecasting; satellite imaging and its use; remote sensing based applications; hazard events' monitoring such as landslide detection; self-driving cars; healthcare and so on. In reference to this latter sector, recently the usage of smart devices has been hugely increased in hospitals and diagnostic centers for evaluating and monitoring various health conditions of affected patients, remotely as well as physically [2].

Practically, there is no field of science or research which performs smartly without using the modern sensors. The wide usage and need of sensors; and IoT employed in remote sensing, environment and

ISSN- 2394-5125 VOL 10, ISSUE 03, 2023

human health monitoring make the applications as intelligent. In the last decade, the agriculture applications have also included [3] the utilization of many types of sensors for monitoring and controlling various types of environmental parameters such as temperature, humidity, soil quality, pollution, air quality, water contamination, radiation, etc. This paper also aims to highlight the use of the sensors and IoT for remote sensing and agriculture applications in terms of extensive discussion and review.

In recent years, SHM of civil structures has been a critical topic for research. SHM helps to detect the damage of a structure, and it also provides early caution of a structure that is not in a safe condition for usage. Civil infrastructure like [4] bridges get damaged with time, and the reason for the damage is heavy vehicles, loading environmental changes, and dynamic forces such as seismic. These types of changes mainly occur at existing structures constructed long ago, and various methods will detect that damage. The strategy of SHM involves observing the structure for a certain period to notice the condition of the structure and the periodic measurements of data will be collected, and the features of data will be extracted from these computation results, and the process of analysis can be done with the help of a featured data to find out the present-day health of the structure. The information collected from the process can be updated periodically to monitor the structure and based on the data collected through monitoring a structure, and the structure can be strengthened and repaired, and rehabilitation and maintenance can be completed [5].

2. LITERATURE SURVEY

Ullo et. al [6] focused on an extensive study of the advances in smart sensors and IoT, employed in remote sensing and agriculture applications such as the assessment of weather conditions and soil quality; the crop monitoring; the use of robots for harvesting and weeding; the employment of drones. The emphasis has been given to specific types of sensors and sensor technologies by presenting an extensive study, review, comparison and recommendation for advancements in IoT that would help researchers, agriculturists, remote sensing scientists and policy makers in their research and implementations.

Sivasuriyan et. al [7] provides a detailed understanding of bridge monitoring, and it focuses on sensors utilized and all kinds of damage detection (strain, displacement, acceleration, and temperature) according to bridge nature (scour, suspender failure, disconnection of bolt and cables, etc.) and environmental degradation under static and dynamic loading. This paper presents information about various methods, approaches, case studies, advanced technologies, real-time experiments, stimulated models, data acquisition, and predictive analysis. Future scope and research also discussed the implementation of SHM in bridges. The main aim of this research is to assist researchers in better understanding the monitoring mechanism in bridges.

Dazhe Zhao et. al [8] proposed an easy-fabricated and compact untethered triboelectric patch with Polytetrafluoroethylene (PTFE) as triboelectric layer and human body as conductor. We find that the conductive characteristic of human body has negligible influence on the outputs, and the untethered triboelectric patch has good output ability and robustness. The proposed untethered triboelectric patches can work as sensor patches and energy harvester patches. Three typical applications are demonstrated, which are machine learning assisted objects distinguishing with accuracy up to 93.09–94.91 %, wireless communication for sending typical words to a cellphone, and human motions energy harvesting for directly powering electronics or charging an energy storage device.

Bacco et. al [9] described, both analytically and empirically, a real testbed implementing IEEE 802.15.4-based communications between an UAV and fixed ground sensors. In our scenario, we found that aerial mobility limits the actual IEEE 802.15.4 transmission range among the UAV and the

ground nodes to approximately 1/3 of the nominal one. We also provide considerations to design the deployment of sensors in precision agriculture scenarios.

Verma et. al [10] discussed the existing state-of-the-art practices of improved intelligent features, controlling parameters and Internet of things (IoT) infrastructure required for smart building. The main focus is on sensing, controlling the IoT infrastructure which enables the cloud clients to use a virtual sensing infrastructure using communication protocols. The following are some of the intelligent features that usually make building smart such as privacy and security, network architecture, health services, sensors for sensing, safety, and overall management in smart buildings. As we know, the Internet of Things (IoT) describes the ability to connect and control the appliances through the network in smart buildings. The development of sensing technology, control techniques, and IoT infrastructure give rise to a smart building more efficient. Therefore, the new and problematic innovation of smart buildings in the context of IoT is to a great extent and scattered. The conducted review organized in a scientific manner for future research direction which presents the existing challenges, and drawbacks.

Hu et. al [11] presented a real-time, fine-grained, and power-efficient air quality monitor system based on aerial and ground sensing. The architecture of this system consists of the sensing layer to collect data, the transmission layer to enable bidirectional communications, the processing layer to analyze and process the data, and the presentation layer to provide a graphic interface for users. Three major techniques are investigated in our implementation for data processing, deployment strategy, and power control. For data processing, spatial fitting and short-term prediction are performed to eliminate the influences of incomplete measurement and the latency of data uploading. The deployment strategies of ground sensing and aerial sensing are investigated to improve the quality of the collected data. Power control is further considered to balance between power consumption and data accuracy. Our implementation has been deployed in Peking University and Xidian University since February 2018, and has collected almost 100,000 effective values thus far.

Famila et. al [12] proposed an Improved Artificial Bee colony optimization based ClusTering(IABCOCT) algorithm by utilizing the merits of Grenade Explosion Method (GEM) and Cauchy Operator. This incorporation of GEM and Cauchy operator prevents the Artificial Bee Colony (ABC) algorithm from stuck into local optima and improves the convergence rate. The benefits of GEM and Cauchy operator are embedded into the Onlooker Bee and scout bee phase for phenomenal improvement in the degree of exploitation and exploration during the process of CH selection. The simulation results reported that the IABCOCT algorithm outperforms the state of art methods like Hierarchical Clustering-based CH Election (HCCHE), Enhanced Particle Swarm Optimization Technique (EPSOCT) and Competitive Clustering Technique (CCT) in-terms of different measures such as throughput, packet loss, delay, energy consumption and network lifetime.

3. PROPOSED SYSTEM

3.1 Data Preprocessing in Machine learning

Data pre-processing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model.

When creating a machine learning project, it is not always a case that we come across the clean and formatted data. And while doing any operation with data, it is mandatory to clean it and put in a formatted way. So, for this, we use data pre-processing task.

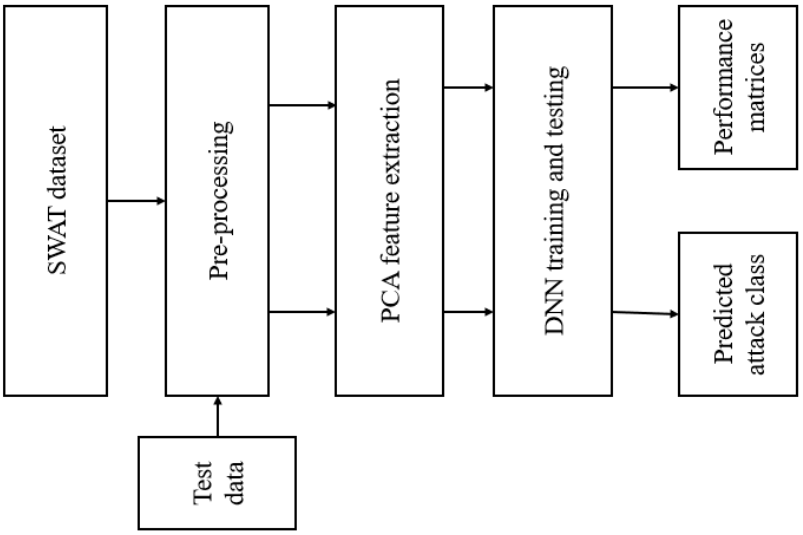


Fig. 1: Block diagram of proposed system.

Why do we need Data Pre-processing?

A real-world data generally contains noises, missing values, and maybe in an unusable format which cannot be directly used for machine learning models. Data pre-processing is required tasks for cleaning the data and making it suitable for a machine learning model which also increases the accuracy and efficiency of a machine learning model.

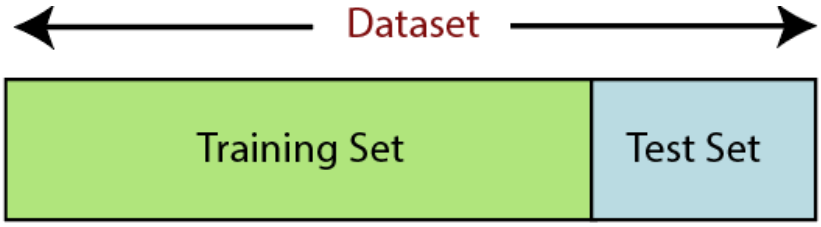
- Getting the dataset
- Importing libraries
- Importing datasets
- Finding Missing Data
- Encoding Categorical Data
- Splitting dataset into training and test set
- Feature scaling

Splitting the Dataset into the Training set and Test set

In machine learning data pre-processing, we divide our dataset into a training set and test set. This is one of the crucial steps of data pre-processing as by doing this, we can enhance the performance of our machine learning model.

Suppose if we have given training to our machine learning model by a dataset and we test it by a completely different dataset. Then, it will create difficulties for our model to understand the correlations between the models.

If we train our model very well and its training accuracy is also very high, but we provide a new dataset to it, then it will decrease the performance. So we always try to make a machine learning model which performs well with the training set and also with the test dataset. Here, we can define these datasets as:



Training Set: A subset of dataset to train the machine learning model, and we already know the output.

Test set: A subset of dataset to test the machine learning model, and by using the test set, model predicts the output.

3.2 PCA feature reduction

The Principal Component Analysis is a popular unsupervised learning technique for reducing the dimensionality of data. It increases interpretability yet, at the same time, it minimizes information loss. It helps to find the most significant features in a dataset and makes the data easy for plotting in 2D and 3D. PCA helps in finding a sequence of linear combinations of variables.

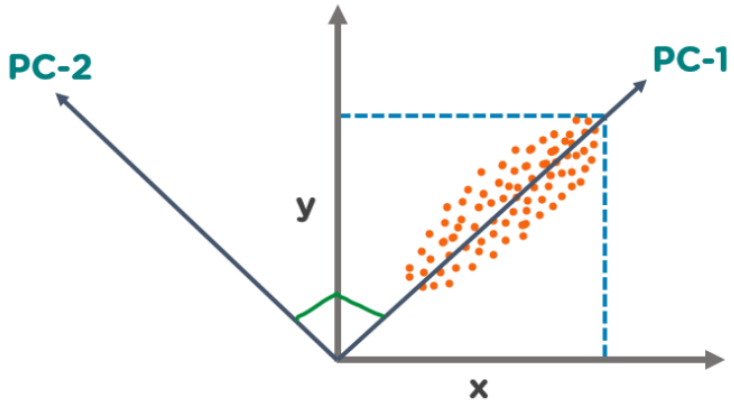


Fig. 2: PCA analysis.

In the above figure, we have several points plotted on a 2-D plane. There are two principal components. PC1 is the primary principal component that explains the maximum variance in the data. PC2 is another principal component that is orthogonal to PC1.

Fig. 3: Applications of PCA in Machine Learning.

- PCA is used to visualize multidimensional data.
- It is used to reduce the number of dimensions in healthcare data.
- PCA can help resize an image.
- It can be used in finance to analyze stock data and forecast returns.
- PCA helps to find patterns in the high-dimensional datasets.

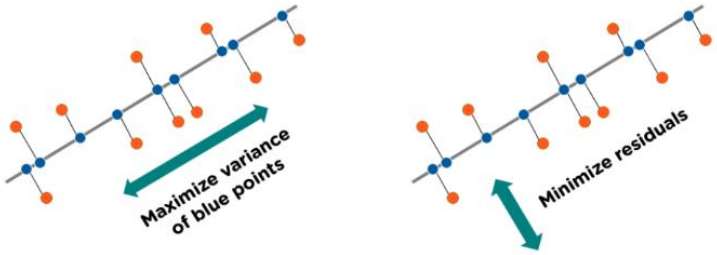


Fig. 4: PCA working.

Step 1: Normalize the data: Standardize the data before performing PCA. This will ensure that each feature has a mean = 0 and variance = 1.

$$Z = \frac{x - \mu}{\sigma}$$

Step 2: Build the covariance matrix: Construct a square matrix to express the correlation between two or more features in a multidimensional dataset.

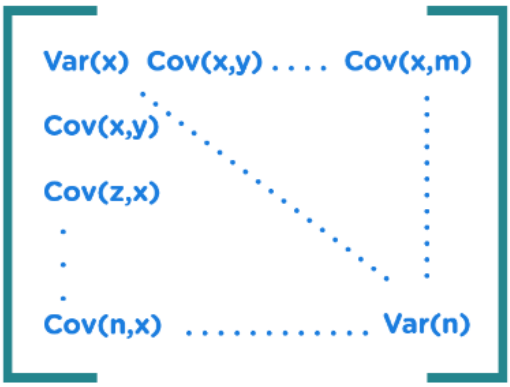


Fig. 5: Covariance matrix formulation.

Step 3: Find the Eigenvectors and Eigenvalues: Calculate the eigenvectors/unit vectors and eigenvalues. Eigenvalues are scalars by which we multiply the eigenvector of the covariance matrix.

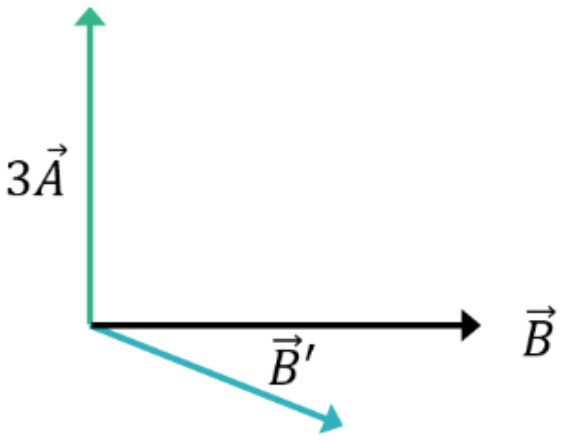


Fig. 6: PCA dimension reduction.

Step 4: Sort the eigenvectors in highest to lowest order and select the number of principal components.

3.3 DNN

3.3.1 Perceptron

Although today the Perceptron is widely recognized as an algorithm, it was initially intended as an image recognition machine. It gets its name from performing the human-like function of perception, seeing, and recognizing images.

In particular, interest has been centered on the idea of a machine which would be capable of conceptualizing inputs impinging directly from the physical environment of light, sound, temperature, etc. — the "phenomenal world" with which we are all familiar — rather than requiring the intervention of a human agent to digest and code the necessary information. Rosenblatt's perceptron machine relied on a basic unit of computation, the neuron. Just like in previous models, each neuron has a cell that receives a series of pairs of inputs and weights. The major difference in Rosenblatt's model is that inputs are combined in a weighted sum and, if the weighted sum exceeds a predefined threshold, the neuron fires and produces an output.

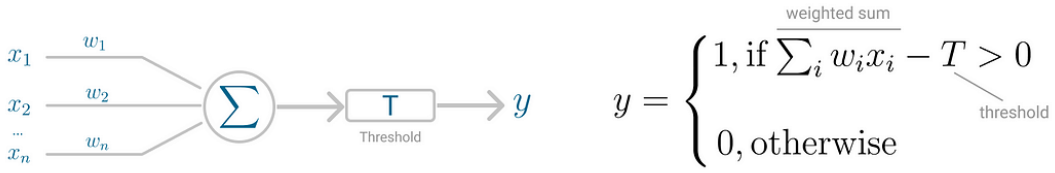


Fig. 7: Perceptron neuron model (left) and threshold logic (right).

Threshold *T* represents the activation function. If the weighted sum of the inputs is greater than zero the neuron outputs the value 1, otherwise the output value is zero.

Perceptron for Binary Classification

With this discrete output, controlled by the activation function, the perceptron can be used as a binary classification model, defining a linear decision boundary.

It finds the separating hyperplane that minimizes the distance between misclassified points and the decision boundary. The perceptron loss function is defined as below:

$$\underbrace{D(w,c)}_{\text{distance}} = -\sum_{i \in \mathrm{M}} \overbrace{y_i}^{\text{output}} (x_i w_i + c)$$

misclassified observations

To minimize this distance, perceptron uses stochastic gradient descent (SGD) as the optimization function. If the data is linearly separable, it is guaranteed that SGD will converge in a finite number of steps. The last piece that Perceptron needs is the activation function, the function that determines if the neuron will fire or not. Initial Perceptron models used sigmoid function, and just by looking at its shape, it makes a lot of sense! The sigmoid function maps any real input to a value that is either 0 or 1 and encodes a non-linear function. The neuron can receive negative numbers as input, and it will still be able to produce an output that is either 0 or 1.

But, if you look at Deep Learning papers and algorithms from the last decade, you'll see the most of them use the Rectified Linear Unit (ReLU) as the neuron's activation function. The reason why ReLU became more adopted is that it allows better optimization using SGD, more efficient computation and is scale-invariant, meaning, its characteristics are not affected by the scale of the input.

The neuron receives inputs and picks an initial set of weights random. These are combined in weighted sum and then ReLU, the activation function, determines the value of the output.

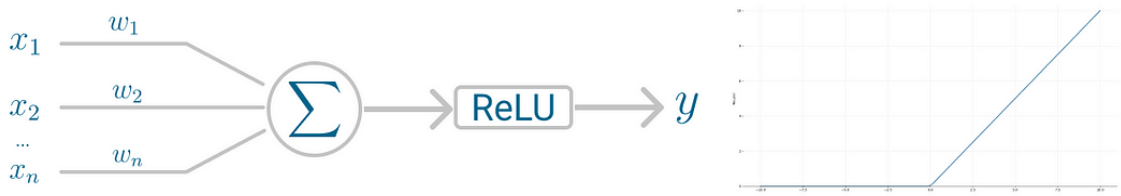


Fig. 8: Perceptron neuron model (left) and activation function (right).

Perceptron uses SGD to find, or you might say learn, the set of weight that minimizes the distance between the misclassified points and the decision boundary. Once SGD converges, the dataset is separated into two regions by a linear hyperplane. Although it was said the Perceptron could represent any circuit and logic, the biggest criticism was that it couldn't represent the XOR gate, exclusive OR, where the gate only returns 1 if the inputs are different. This was proved almost a decade later and highlights the fact that Perceptron, with only one neuron, can't be applied to non-linear data.

3.3.2 MLP

The MLP was developed to tackle this limitation. It is a neural network where the mapping between inputs and output is non-linear. A MLP has input and output layers, and one or more hidden layers with many neurons stacked together. And while in the Perceptron the neuron must have an activation function that imposes a threshold, like ReLU or sigmoid, neurons in a MLP can use any arbitrary activation function.

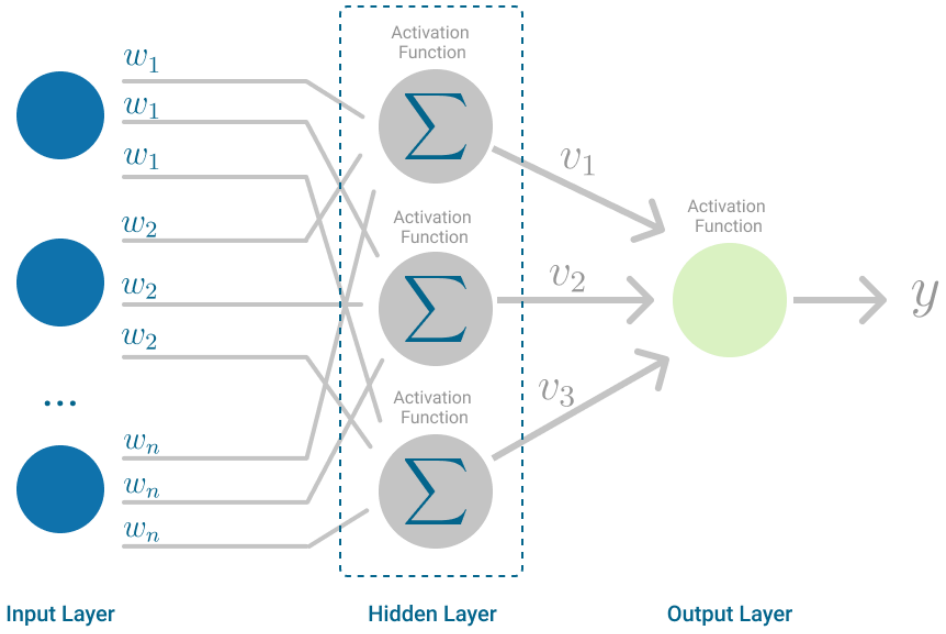


Fig. 9: Architecture of MLP.

MLP falls under the category of feedforward algorithms, because inputs are combined with the initial weights in a weighted sum and subjected to the activation function, just like in the Perceptron. But the difference is that each linear combination is propagated to the next layer. Each layer is feeding the next one with the result of their computation, their internal representation of the data. This goes all the way through the hidden layers to the output layer.

ISSN- 2394-5125 VOL 10, ISSUE 03, 2023

If the algorithm only computed the weighted sums in each neuron, propagated results to the output layer, and stopped there, it wouldn't be able to learn the weights that minimize the cost function. If the algorithm only computed one iteration, there would be no actual learning. This is where Backpropagation comes into play.

Backpropagation

Backpropagation is the learning mechanism that allows the MLP to iteratively adjust the weights in the network, with the goal of minimizing the cost function. There is one hard requirement for backpropagation to work properly.

The function that combines inputs and weights in a neuron, for instance the weighted sum, and the threshold function, for instance ReLU, must be differentiable. These functions must have a bounded derivative because Gradient Descent is typically the optimization function used in MLP.

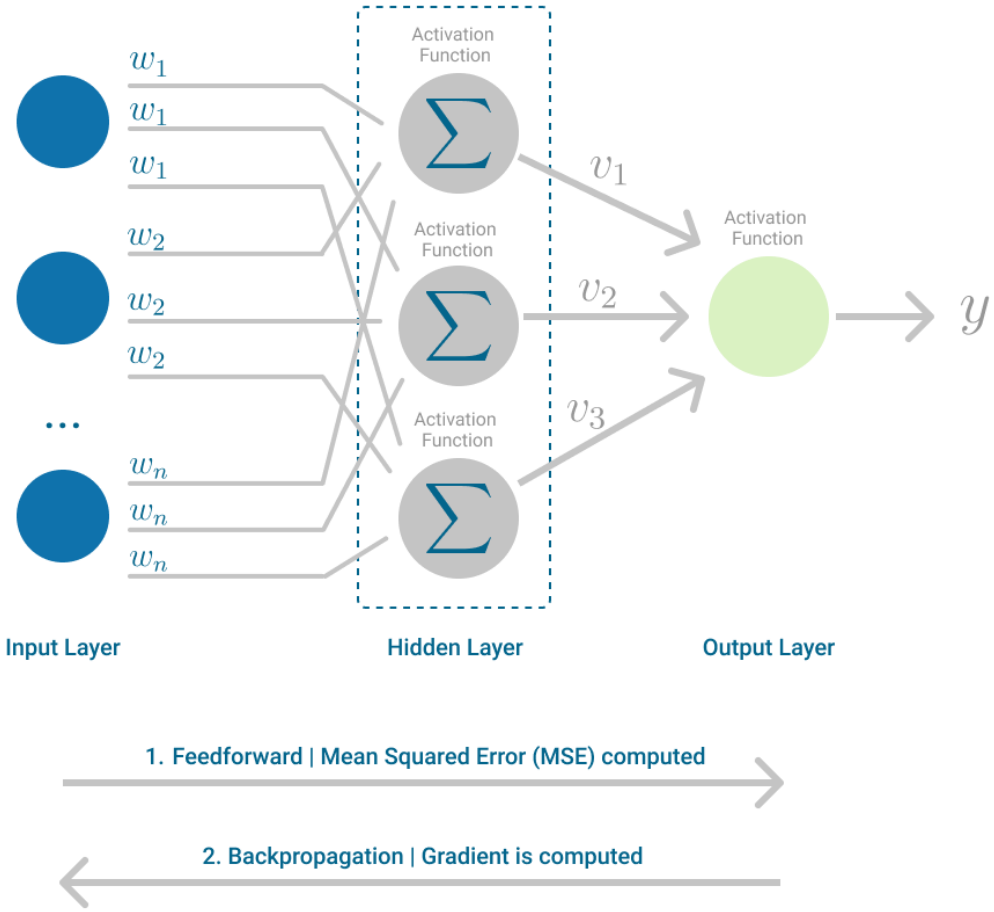


Fig. 10: MLP, highlighting the Feedforward and Backpropagation steps.

In each iteration, after the weighted sums are forwarded through all layers, the gradient of the Mean Squared Error is computed across all input and output pairs. Then, to propagate it back, the weights of the first hidden layer are updated with the value of the gradient. That's how the weights are propagated back to the starting point of the neural network. One iteration of Gradient Descent is defined as follows:

$$\underbrace{\Delta_w(t)}_{\text{Gradient Current Iteration}} = \underbrace{-\varepsilon}_{\text{Bias}} \frac{\overbrace{dE}^{\text{Error}}}{\underbrace{dw_{(t)}}_{\text{Weight vector}}} + \underbrace{\alpha}_{\text{Learning Rate}} \underbrace{\Delta_{w(t-1)}}_{\text{Gradient Previous Iteration}}$$

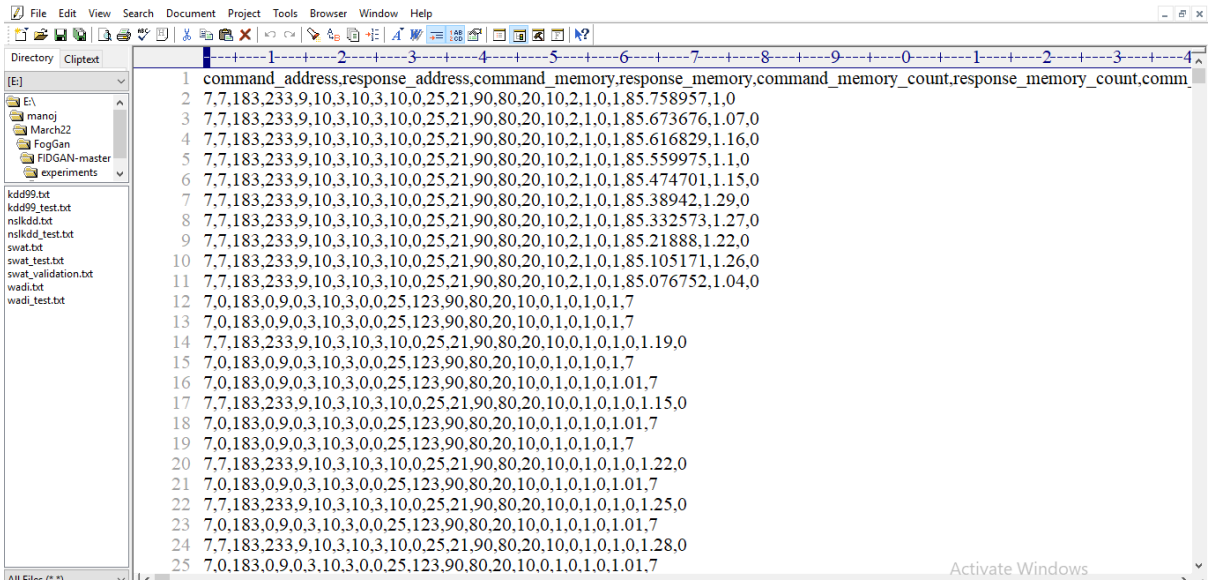
This process keeps going until gradient for each input-output pair has converged, meaning the newly computed gradient hasn't changed more than a specified convergence threshold, compared to the previous iteration.

4. RESULTS AND DISCUSSION

To implement this project, we have used SWAT (secure water production treatment) and this dataset contains IOT request and response signature and associate each dataset with unique attack label and dataset contains below cyber-attack labels.

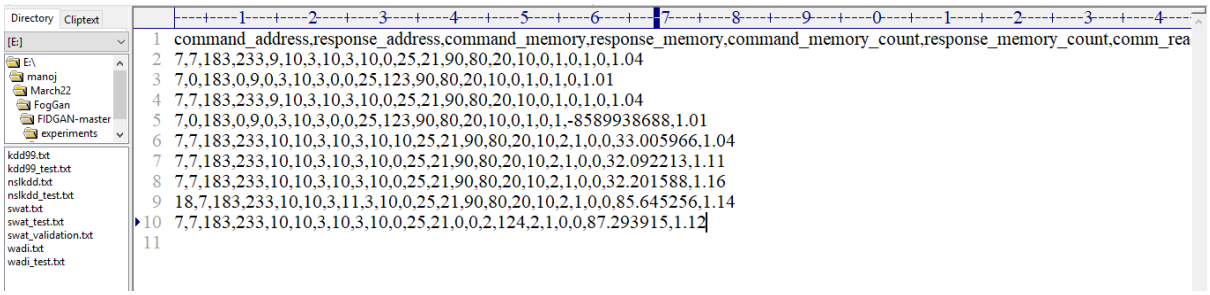
'Normal', 'Naive Malicious Response Injection (NMRI)', 'Complex Malicious', 'Response Injection (CMRI)', 'Malicious State Command Injection (MSCI)', 'Malicious Parameter Command Injection (MPCI)', 'Malicious Function Code Injection (MFCI)', 'Denial of Service (DoS)'

Above are the attacks found in dataset and dataset contains above labels as integer value of its index for example Normal label index will be 0 and continues up to 8 class labels. Below screen showing dataset details.



In above dataset screen first row contains dataset column names and remaining rows contains dataset values and in last column we have attack type from label 0 to 7. We used above dataset to train propose Auto Encoder, decision tree and DNN algorithms.

In below screen we are using NEW test data which contains only signature and there is no class label and propose algorithm will detect and attribute class labels.

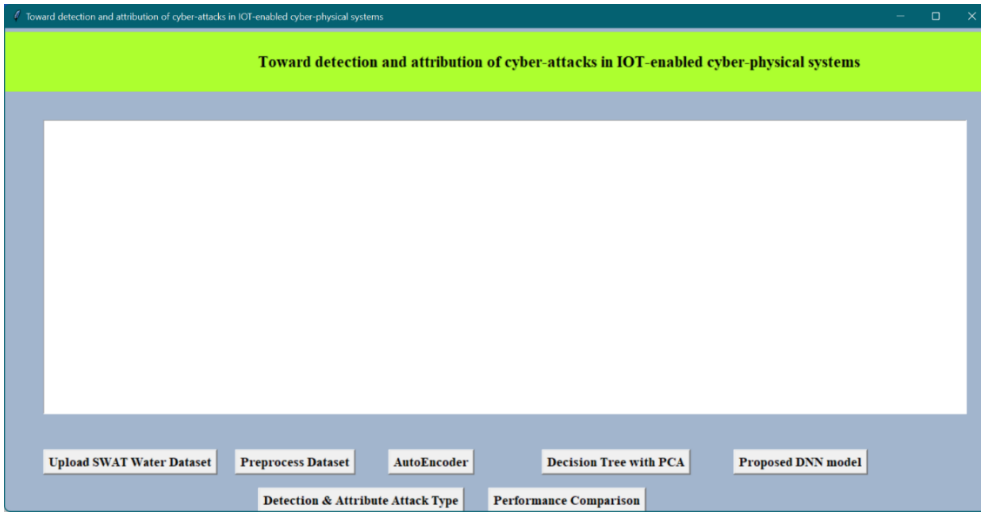


In above test data we have IOT request signature without class labels.

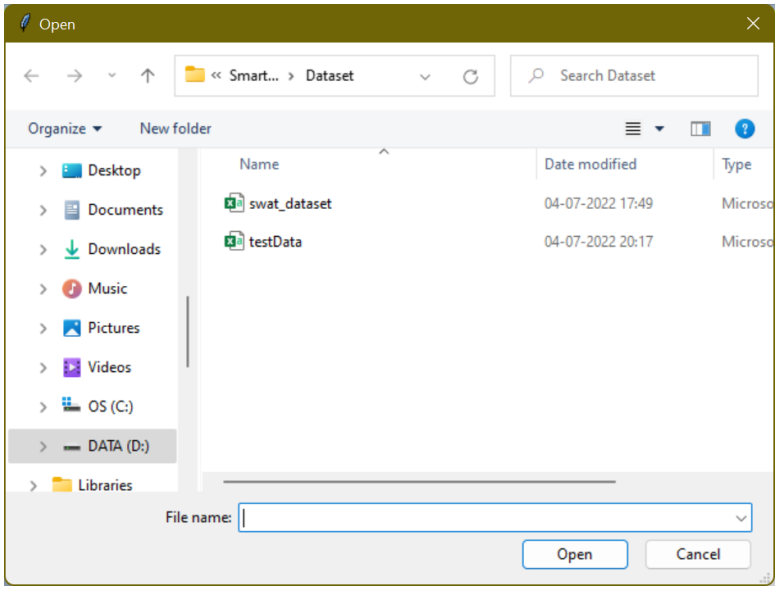
To implement this project, we have designed following modules.

- Upload SWAT Water Dataset: This module used to upload and read the dataset to the UI application and then find different attacks found in dataset.
- Preprocess Dataset: This module used to replace all missing values with 0 and applies MIN-MAX scaling algorithm to normalize the features values and then it split the dataset into train and test where this application used 80% dataset for training and 20% for testing.
- AutoEncoder: This module used to train AutoEncoder and then extract features from that model.
- Decision Tree with PCA: This module used to reduce the size of features (using PCA) extracted features from AutoEncoder and then retrain with them with Decision tree for predicting the label for each record based on dataset signatures.
- Proposed DNN model: predicted decision tree label will further train with DNN (deep neural network) algorithm to enhance the detection accuracy.
- Detection & Attribute Attack Type: This module used to detect the type of attack from uploaded unknown or un-label TEST DATA.
- Performance Comparison: This module display the comparison table of all algorithms which contains metrics like accuracy, precision, recall and FSCORE.

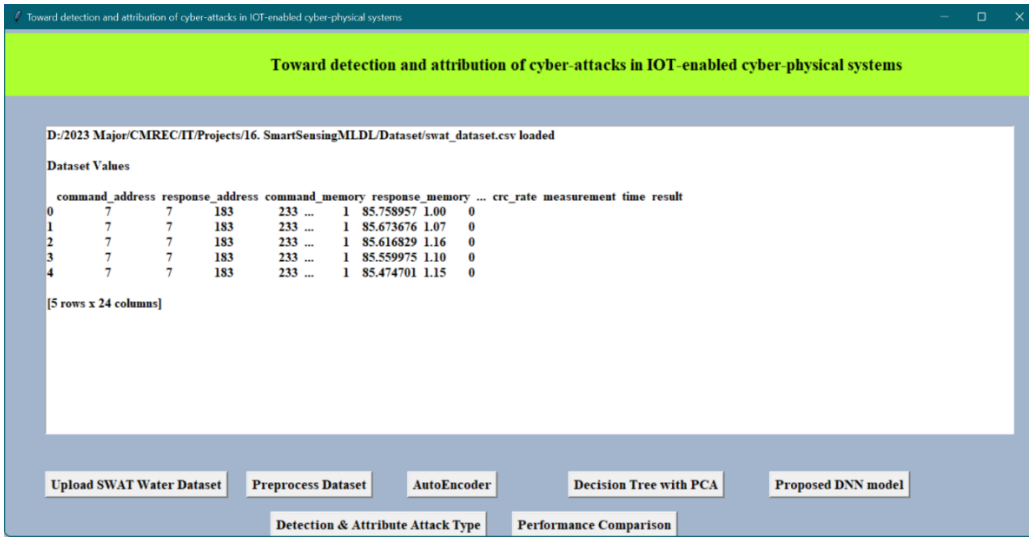
UI OUTPUTS

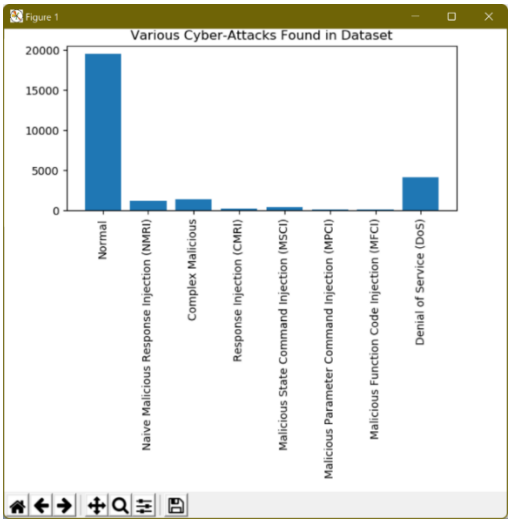


In above screen click on 'Upload SWAT Water Dataset' button to upload dataset to application and get below output



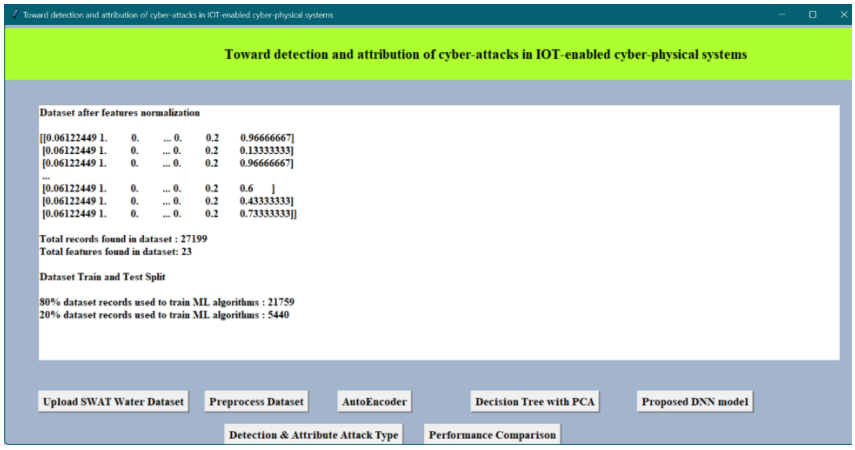
In above screen selecting and uploading SWAT dataset file and then click on 'Open' button to load dataset and get below output



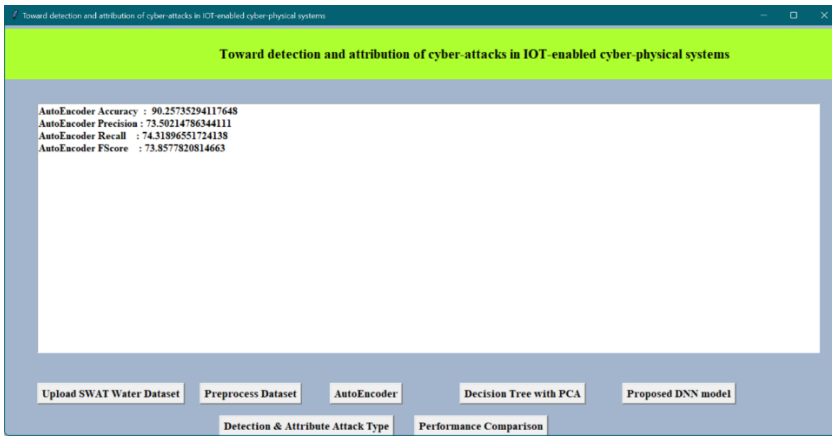


In above screen dataset loaded and in graph x-axis contains ATTACK NAME and y-axis contains count of those attacks found in dataset and we can see 'NORMAL' class contains so many records

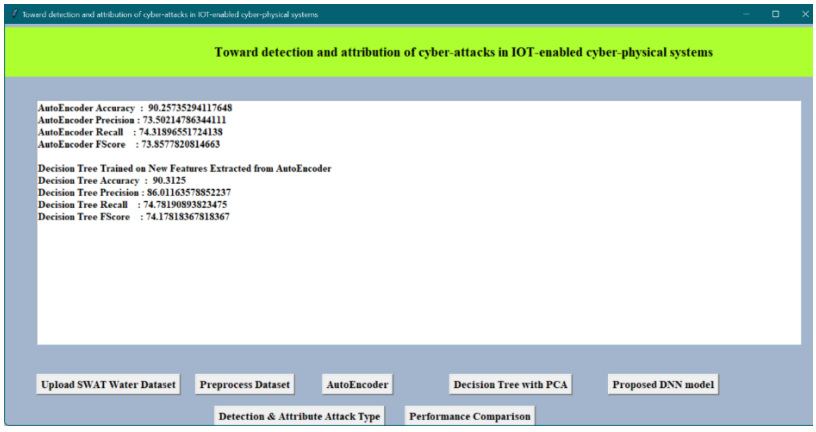
and other attacks contains very few records so it will raise data imbalance problem which can be solved using AutoEncoder, Decision Tree and DNN. Now close above graph and then click on 'Preprocess Dataset' button to remove missing values and then normalized values with MIN-MAX algorithm



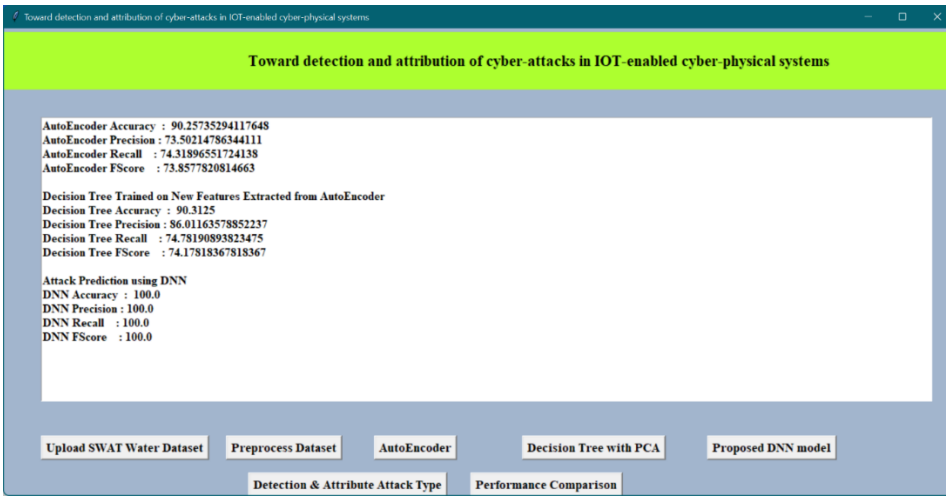
In above screen all values are normalized (converting data between 0 and 1 called as normalization) and then we can see total records in dataset and then dataset train and test split records count also displaying. Now dataset is ready and now click on 'AutoEncoder' button to train dataset with AutoEncoder and get below accuracy.



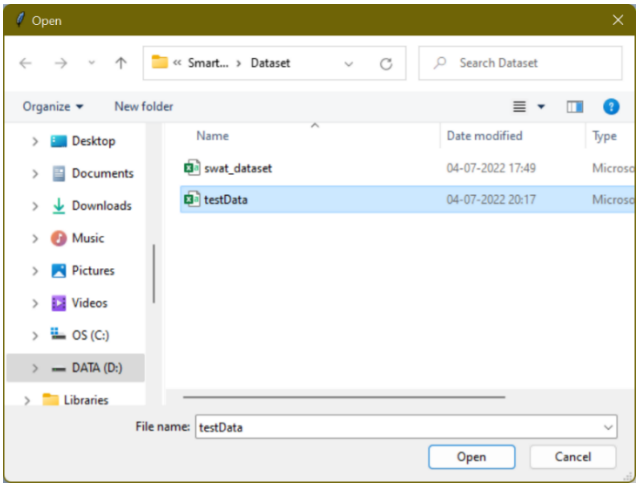
In above screen with AutoEncoder we got 90.18% accuracy, and this accuracy can be enhance by implementing Decision Tree with PCA algorithm and now click on 'Decision Tree with PCA' button to get below output



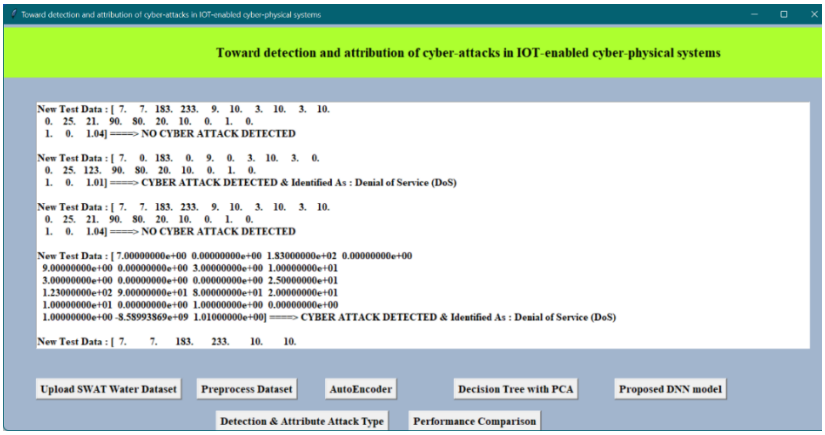
In above screen we can see with decision tree accuracy and precision value is enhanced and now click on 'Proposed DNN model' button to further enhance accuracy and get below output



In above screen with DNN we got 100% accuracy, and this accuracy may vary from 95 to 100% as we are splitting dataset into random train and test. Now click on 'Detection & Attribute Attack Type' button to upload test DATA and detect attack attributes



In above screen selecting and uploading 'TEST DATA' file and then click on 'Open' button to get below output.



In above screen in square bracket, we can see TEST data values and after arrow =➔ symbol we can see detected ATTACK TYPE and scroll down above text area to view all detection.

In above screen we can see detected various attacks and now click on 'Performance Comparison' to get below comparison table of all algorithms

Algorithm Name	Accuracy	Precision	Recall	FSCORE
AutoEncoder	90.0	73.14281070224018	73.58689458689459	73.29616654463219
Decision Tree with PCA	90.4779411764706	85.75313833952161	74.62748067246231	73.96952834086689
DNN	100.0	100.0	100.0	100.0

In above table we can see algorithm names and its metrics values such as accuracy and precision and other.

5. CONCLUSION

Internet of Things enabled cyber physical systems such as Industrial equipment's and operational IT to send and receive data over internet. This equipment's will have sensors to sense equipment condition and report to centralized server using internet connection. Sometime some malicious users may attack or hack such sensors and then alter their data and this false data will be report to centralized server and false action will be taken. Due to false data many countries equipment and production system got failed and many algorithms was developed to detect attack, but all these algorithms suffer from data imbalance (one class my contains huge records (for example NORMAL records and other class like attack may contains few records which lead to imbalance problem and detection algorithms may failed to predict accurately). To deal with data imbalance existing algorithms were using OVER and UNDER sampling which will generate new records for FEWER class, but this technique improve accuracy but not up to the mark. Therefore, to overcome from this issue, this project introduced an efficient deep learning model without using any under or oversampling algorithms with the usage of auto encoder, decision tree with PCA, and DNN for identifying the attack and classify the type of attack. In addition, the performance evaluation of three models also compared and proven that proposed DNN obtained enhanced accuracy 99.98%.

REFERENCES

[1] Kayad, A.; Paraforos, D.; Marinello, F.; Fountas, S. Latest advances in sensor applications in agriculture. Agriculture 2020, 10, 362.

[2] Elahi, H.; Munir, K.; Eugeni, M.; Atek, S.; Gaudenzi, P. Energy harvesting towards self-powered IoT devices. Energies 2020, 13, 5528.

[3] Ullo, S.L.; Sinha, G.R. Advances in smart environment monitoring systems using IoT and sensors. Sensors 2020, 20, 3113.

[4] Carminati, M.; Sinha, G.R.; Mohdiwale, S.; Ullo, S.L. Miniaturized pervasive sensors for indoor health monitoring in smart cities. Smart Cities 2021, 4, 146–155.

[5] Ullo, S.L.; Addabbo, P.; Di Martire, D.; Sica, S.; Fiscante, N.; Cicala, L.; Angelino, C.V. Application of DInSAR technique to high coherence Sentinel-1 images for dam monitoring and result validation through in situ measurements. IEEE J. Sel. Top. Appl. Earth Obs. Remote. Sens. 2019, 12, 875–890.

[6] Ullo, S.L. and Sinha, G.R., 2021. Advances in IoT and smart sensors for remote sensing and agriculture applications. Remote Sensing, 13(13), p.2585.

[7] Sivasuriyan, A., Vijayan, D.S., LeemaRose, A., Revathy, J., Gayathri Monicka, S., Adithya, U.R. and Jebasingh Daniel, J., 2021. Development of smart sensing technology approaches in structural health monitoring of bridge structures. Advances in Materials Science and Engineering, 2021.

[8] Dazhe Zhao, Kaijun Zhang, Yan Meng, Zhaoyang Li, Yucong Pi, Yujun Shi, Jiacheng You, Renkun Wang, Ziyi Dai, Bingpu Zhou, Junwen Zhong, Untethered triboelectric patch for

wearable smart sensing and energy harvesting, Nano Energy, Volume 100, 2022, 107500, ISSN 2211-2855, https://doi.org/10.1016/j.nanoen.2022.107500.

[9] M. Bacco, A. Berton, A. Gotta and L. Caviglione, "IEEE 802.15.4 Air-Ground UAV Communications in Smart Farming Scenarios," in IEEE Communications Letters, vol. 22, no. 9, pp. 1910-1913, Sept. 2018, doi: 10.1109/LCOMM.2018.2855211.

[10] A. Verma, S. Prakash, V. Srivastava, A. Kumar and S. C. Mukhopadhyay, "Sensing, Controlling, and IoT Infrastructure in Smart Building: A Review," in IEEE Sensors Journal, vol. 19, no. 20, pp. 9036-9046, 15 Oct.15, 2019, doi: 10.1109/JSEN.2019.2922409.

[11] Z. Hu, Z. Bai, Y. Yang, Z. Zheng, K. Bian and L. Song, "UAV Aided Aerial-Ground IoT for Air Quality Sensing in Smart City: Architecture, Technologies, and Implementation," in IEEE Network, vol. 33, no. 2, pp. 14-22, March/April 2019, doi: 10.1109/MNET.2019.1800214.

[12] Famila, S., Jawahar, A., Sariga, A. et al. Improved artificial bee colony optimization-based clustering algorithm for SMART sensor environments. Peer-to-Peer Netw. Appl. 13, 1071–1079 (2020). https://doi.org/10.1007/s12083-019-00805-4