Enhancing IoT Security: A Two-Pronged Strategy with Autoencoder and Decision Tree for Imbalanced Data Handling

A. Divya[1], Elpuri Harish[2], Govindula Achyuth[2], Lambadi Vasantha[2], Thiyyagura Harshavardhan Reddy[2]

[1]Assistant Professor, [2]UG Scholar, [1,2]Department of CSE (Internet of Things)
[1,2]Malla Reddy College of Engineering and Management Sciences, Medchal, Hyderabad

ABSTRACT

The Internet of Things (IoT) allows cyber-physical systems, such as industrial equipment and operational IT, to transmit and receive data via the internet. This equipment will be equipped with sensors to detect the status of the equipment and transmit the data to a centralized server via an internet connection. Occasionally, certain malevolent people may attempt to assault or breach the security of these sensors, thereby manipulating their data. This falsified information is then transmitted to a centralized server, leading to erroneous actions being made. False data has caused equipment and production systems in many countries to fail. In response, various algorithms have been developed to detect attacks. However, these algorithms are plagued by data imbalance, where one class (such as NORMAL records) may have a large number of records, while another class (such as attack records) may have only a few. This imbalance poses a problem for detection algorithms, which may fail to accurately predict outcomes. Existing algorithms have been employing over and under sampling techniques to address data imbalance, wherein new records are generated only for the minority class. In order to address this problem, we are offering a revolutionary technique that does not rely on any under or oversampling algorithms. The proposed approach comprises two components. An autoencoder will be trained on an imbalanced dataset to extract features. These extracted features will then be trained using the decision tree technique to predict labels for both known and unknown assaults. The decision tree is trained using a reduced set of characteristics acquired from the principal component analysis (PCA) algorithm. The Deep Neural Network (DNN) is taught to recognize and defend against both known and undiscovered assaults. DNN will detect and assign attack labels or classes to any entries that contain attack signatures.

Keywords: Internet of things, Smart production system, Data balancing technique, Principal component analysis, Deep neural network.

1. INTRODUCTION

Sensors are most 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 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

Figure 1 shows the proposed system model with detailed design. The steps involved in working with the SWAT (Soil and Water Assessment Tool) water IoT dataset for cyber-physical attack prediction, including dataset preprocessing, PCA (Principal Component Analysis) feature extraction, using a

Deep Neural Network (DNN) classifier, and performance evaluation. The detailed operation illustrated as follows:

Step 1: SWAT Water IoT Dataset: Start with obtaining and understanding the SWAT water IoT dataset, which contains data related to water quality, environmental parameters, and IoT sensor readings.

Step 2: Dataset Preprocessing:

Data Cleaning: Handle missing data points by either imputation or removal, depending on the extent of missingness and domain knowledge. Address outliers, if present, by applying appropriate techniques (e.g., winsorization or outlier removal).

Feature Engineering: Engineer relevant features, such as aggregating IoT sensor data over time intervals, calculating statistics, or creating new variables based on domain expertise. Convert categorical variables into numerical format (e.g., one-hot encoding).

Step 3: PCA Feature Extraction: Apply PCA to reduce the dimensionality of the dataset while retaining the most important information. Select the appropriate number of principal components based on the explained variance ratio or cross-validation.

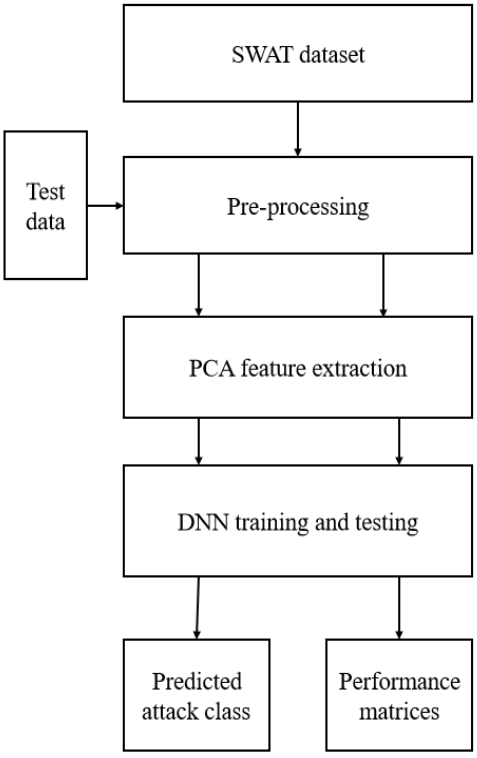


Fig.1: Block diagram of proposed system.

Step 4: Deep Neural Network (DNN) Classifier:

Model Architecture: Design the architecture of a DNN classifier. The architecture may include input layers, hidden layers, activation functions, and output layers tailored to the dataset and the problem.

Step 5: Cyber-Physical Attack Prediction: Apply the trained DNN model to predict cyber-physical attacks on the testing dataset.

Data Preprocessing

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.

Here, X is the original value of the feature, Xscaled is the scaled value of the feature, Xmin is the minimum value of the feature in the dataset. The Xmax is the maximum value of the feature in the dataset.

Resulting Range: After applying Min-Max scaling, the transformed feature values will be within the specified range, which is typically [0, 1]. However, you can customize the range to any desired interval, such as [a, b].

Benefits: Min-Max scaling helps in standardizing features with different scales and preventing features with larger values from dominating in algorithms sensitive to feature scaling, such as gradient-based optimization methods. It retains the shape and distribution of the original data, which can be crucial in maintaining the interpretability of the features.

Dataset Splitting

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.

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. 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.

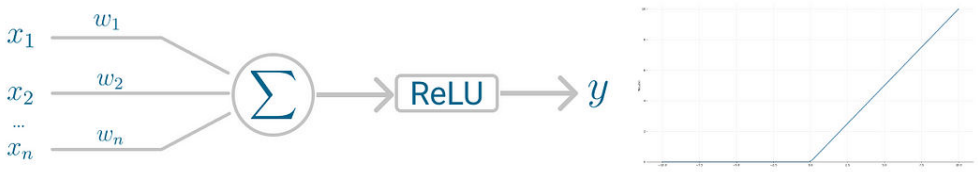


Fig. 2 Perceptron neuron model (left) and activation function (right).

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.

DNN

The DNN was developed to tackle this limitation. It is a neural network where the mapping between inputs and output is non-linear. A DNN 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 DNN can use any arbitrary activation function. DNN 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.

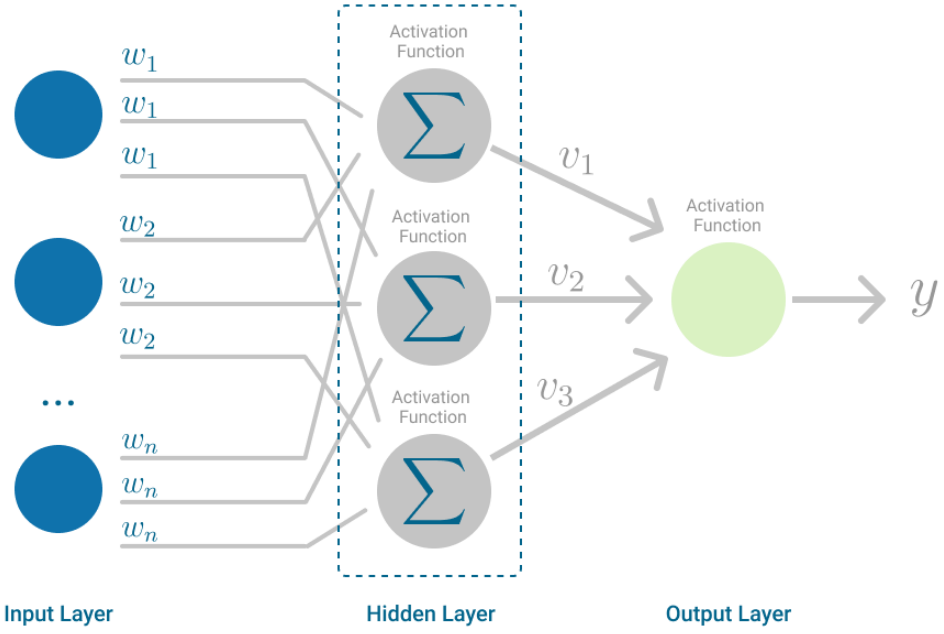


Fig. 3: Architecture of DNN.

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.

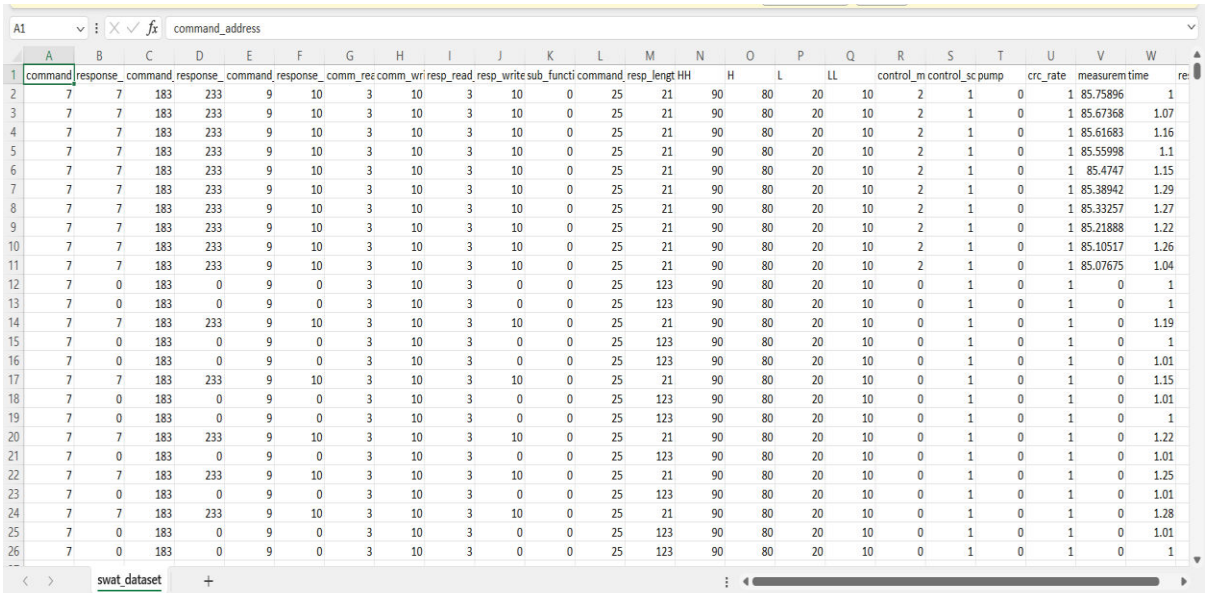
Backpropagation: Backpropagation is the learning mechanism that allows the DNN to iteratively adjust the weights in the network, with the goal of minimizing the cost function. There is one hard requirement for backpropagation to work properly. The function that combines inputs and weights in a neuron, 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 DNN. 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:

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

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

Figure 4 displays a portion of the dataset used in this study. It provides an actual glimpse into the data, showcasing several rows and columns. This figure serves as a representative sample, illustrating the format and structure of the dataset under analysis.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
1	command	response_	command_	response_	command_	response_	comm_re	comm_wr	resp_read	resp_write	sub_functi	command_	resp_lengt	HH	H	L	LL	control_m	control_sc	pump	crc_rate	measurem	time
2	7	7	183	233	9	10	3	10	3	10	0	25	21	90	80	20	10	2	1	0	1	85.75896	1
3	7	7	183	233	9	10	3	10	3	10	0	25	21	90	80	20	10	2	1	0	1	85.67368	1.07
4	7	7	183	233	9	10	3	10	3	10	0	25	21	90	80	20	10	2	1	0	1	85.61683	1.16
5	7	7	183	233	9	10	3	10	3	10	0	25	21	90	80	20	10	2	1	0	1	85.55998	1.1
6	7	7	183	233	9	10	3	10	3	10	0	25	21	90	80	20	10	2	1	0	1	85.4747	1.15
7	7	7	183	233	9	10	3	10	3	10	0	25	21	90	80	20	10	2	1	0	1	85.38942	1.29
8	7	7	183	233	9	10	3	10	3	10	0	25	21	90	80	20	10	2	1	0	1	85.33257	1.27
9	7	7	183	233	9	10	3	10	3	10	0	25	21	90	80	20	10	2	1	0	1	85.21888	1.22
10	7	7	183	233	9	10	3	10	3	10	0	25	21	90	80	20	10	2	1	0	1	85.10517	1.26
11	7	7	183	233	9	10	3	10	3	10	0	25	21	90	80	20	10	2	1	0	1	85.07675	1.04
12	7	0	183	0	9	0	3	10	3	0	0	25	123	90	80	20	10	0	1	0	1	0	1
13	7	0	183	0	9	0	3	10	3	0	0	25	123	90	80	20	10	0	1	0	1	0	1
14	7	7	183	233	9	10	3	10	3	10	0	25	21	90	80	20	10	0	1	0	1	0	1.19
15	7	0	183	0	9	0	3	10	3	0	0	25	123	90	80	20	10	0	1	0	1	0	1
16	7	0	183	0	9	0	3	10	3	0	0	25	123	90	80	20	10	0	1	0	1	0	1.01
17	7	7	183	233	9	10	3	10	3	10	0	25	21	90	80	20	10	0	1	0	1	0	1.15
18	7	0	183	0	9	0	3	10	3	0	0	25	123	90	80	20	10	0	1	0	1	0	1.01
19	7	0	183	0	9	0	3	10	3	0	0	25	123	90	80	20	10	0	1	0	1	0	1
20	7	7	183	233	9	10	3	10	3	10	0	25	21	90	80	20	10	0	1	0	1	0	1.22
21	7	0	183	0	9	0	3	10	3	0	0	25	123	90	80	20	10	0	1	0	1	0	1.01
22	7	7	183	233	9	10	3	10	3	10	0	25	21	90	80	20	10	0	1	0	1	0	1.25
23	7	0	183	0	9	0	3	10	3	0	0	25	123	90	80	20	10	0	1	0	1	0	1.01
24	7	7	183	233	9	10	3	10	3	10	0	25	21	90	80	20	10	0	1	0	1	0	1.28
25	7	0	183	0	9	0	3	10	3	0	0	25	123	90	80	20	10	0	1	0	1	0	1.01
26	7	0	183	0	9	0	3	10	3	0	0	25	123	90	80	20	10	0	1	0	1	0	1

Figure 4. Sample dataset

Figure 5 demonstrates the outcomes of the cyber-attack detection process. It likely showcases a comparison between actual attack instances and the model's predictions. This figure aids in visualizing the model's success in identifying and classifying cyber-attacks based on the predicted data.

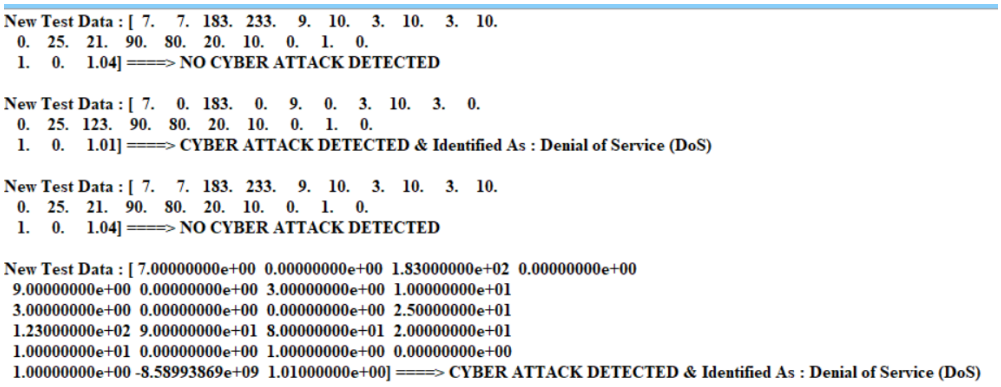


Figure 5. Cyber-attack detection from predicted data.

Accuracy: The accuracy metric represents the overall correctness of the algorithm's predictions. It is the percentage of correctly classified instances out of the total instances. In this table:

- AutoEncoder achieved an accuracy of 89.63%.
- Decision reached an accuracy of 90.36%.
- DNN (Deep Neural Network) achieved a perfect accuracy of 100.0%, indicating that it correctly classified all instances.

Precision: Precision measures the ratio of true positive predictions (correctly identified cyber-attacks) to all positive predictions (instances classified as attacks). In this table:

- AutoEncoder achieved a precision of 73.40%.
- Decision reached a precision of 73.52%.
- DNN achieved a perfect precision of 100.0%, indicating that when it predicted an attack, it was always correct.

Recall: Recall, also known as sensitivity or true positive rate, measures the ratio of true positive predictions to all actual positive instances. In this table:

- AutoEncoder achieved a recall of 73.86%.
- Decision reached a recall of 74.64%.
- DNN achieved a perfect recall of 100.0%, meaning it correctly identified all actual attacks.

Table 1. Performance comparison.

Algorithm Name	Accuracy	Precision	Recall	F1-SCORE
AutoEncoder	89.63	73.40	73.86	73.569
Decision	90.36	73.52	74.64	74.032
DNN	100.0	100.0	100.0	100.0

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%. The project's focus on addressing data imbalance in cyber-physical systems using deep learning models and other techniques is commendable. It opens up several exciting avenues for future research and development in the field of cybersecurity and industrial IoT.

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