This article has been accepted for publication in IEEE Transactions on Consumer Electronics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCE.2023.3319439

Leveraging Gametic Heredity in Oversampling Techniques to Handle Class Imbalance for Efficient Cyberthreat Detection in IIoT

Priyanka Verma, Sr. *Member, IEEE*, John G. Breslin, Sr., *Member, IEEE*, Donna O'Shea, Sr., *Member, IEEE*, Nakul Mehta, Nitesh Bharot, *Member, IEEE*, Ankit Vidyarthi

***Abstract*—In recent years Cyber-Physical Systems (CPS) and Industrial Internet of Things (IIoT) have gained significant attraction; however, it remains a vulnerable target for cyberattacks. Machine learning techniques have garnered interest in security applications due to their rapid processing capabilities and real-time predictions. However, imbalanced data distribution is a prevalent issue in IIoT environments, adversely affecting ML-based attack detection systems. In this work, we present a novel gametic heredity-based oversampling technique for addressing imbalanced data challenges in cybersecurity applications, specifically targeting IIoT systems. The proposed model enhances diversity in the minority classes by generating unique synthetic minority samples, creating diverse synthetic data while restricting instances to the minority class region. The proposed model outperforms complex and conventional methods in terms of precision, recall & F-Score while mitigating over-generalization by evenly distributing newly generated samples within minority class boundaries and regions. To validate the proposed model and verify its efficacy in identifying cyber threats, we used the UNSW-NB15 dataset. Simulation results demonstrate that the proposed model efficiently detects attacks with high performance compared to state-of-the-art techniques. Our research contributes to developing robust & efficient machine learning models for enhancing the security of IIoT systems while handling class imbalance issues.**

***Index Terms*—IIoT, CPS, Cyber Threats, Imbalance Data, Oversampling, Machine Learning**

I. Introduction

THE rapid development and integration of the Industrial Internet of Things (IIoT) have revolutionized the manufacturing and automation industries, driving efficiency, productivity, and cost savings. However, the widespread adoption of these interconnected systems has also increased their vulnerability to cyber threats. Cybercriminals are continually devising sophisticated attack strategies, targeting IIoT to cause disruption, data breaches, and damage to critical assets. Hence, there is a growing need to develop robust & efficient methods for detecting and mitigating cyber threats.

Machine Learning (ML) and Deep Learning (DL) approach shows promise to detect cyber threats in IIoT and ICS environments. However, these techniques face challenges while encountering imbalanced data [1], a common issue in the context of cybersecurity. In realistic scenarios, the amount of normal (benign) data instances is typically higher than the attacks (malicious) instances, leading to a skewed distribution [2]. This imbalance could significantly affect the ML and DL model performance, due to biasing towards majority instances (i.e., normal data points) and resulting in poor detection of cyber threats [3]. Because these algorithms usually consider datasets having equal classes [4], [5].

Handling imbalanced data is a critical aspect of developing effective ML models for cybersecurity applications, especially in the context of IIoT and ICS. Several techniques and methodologies are developed to tackle the data imbalance problem, including data preprocessing techniques, algorithm-based techniques, and cost-sensitive learning methods [6], [7]. Data preprocessing methods, such as oversampling and undersampling, modify the dataset before training the model. Oversampling, in particular, produces synthetic data for the minority class instances, potentially improving the model's performance in detecting cyber threats. The majority of data sampling techniques used till date are synthetic in nature [8], [9], [10], [11]. The Synthetic Minority Oversampling Technique (SMOTE) [8] is well-known among them, which intelligently introduces fresh synthetic or new data instances to the class having low samples. Whereas algorithm-based techniques modify the learning algorithm itself, and cost-sensitive learning methods assign additional misclassification costs to the minority and majority classes [7].

Oversampling techniques offer several advantages over other approaches for handling imbalanced data in cybersecurity applications. They enable the model to understand the characteristics of the minority class instances in a better way and are universally applicable to any dataset.

However, while oversampling techniques like SMOTE and Adaptive Synthetic (ADASYN) improves classifier performance in imbalanced data scenarios, but there are some issues associated with their use. According to Barua et al. [11], these synthetic methods can unanimously enlarge the minority class region, leading to the misclassification of majority class samples. This occurs as synthetic instances are sometimes introduced into majority-class regions. Additionally, these approaches may generate non-diverse and large amounts of similar data samples, as they undertake only nearest-neighbor samples.

In summary, while oversampling techniques have shown promise in addressing imbalanced data issues, they still have some limitations. Erroneous enlargement of the minority class region, generation of non-diverse data samples, data sample

The project is funded by the Science Foundation of Ireland (SFI) under the Grant 16/RC/3918, EU's MSCA with agreement Number 847577 and SFI 12/RC/2289-P2 (Insight).

duplication and increased false alarm rates are some of the challenges associated with these methods. It's crucial to cautiously consider the trade-offs and pick the most appropriate method for handling imbalanced data in cybersecurity applications.

In this paper, we proposed a novel genetics-based oversampling method that achieves high precision and recall with low false positives. In the proposed approach for the minority attack samples, clusters are created using the Gaussian mixture model. Then within the cluster, the gametic hereditical oversampling technique which is inspired by the field of genetics biology for producing synthetic data samples is applied. The generated samples are unique and not the copy of old samples but lie within the boundary of the class. The significant contributions of this paper are described as:

1) A novel gametic heredity based oversampling technique inspired by the field of genetics biology is proposed to enhance the cyberthreat detection in IIoT. The proposed model is capable to deal with imbalanced data situations as well, thereby providing more robust and secure architecture.
2) Our proposed model enhances the diversity within the minority class by generating unique synthetic minority samples inspired by the diversity observed within populations of humans or other living organisms despite being of the same species. This approach creates as much diverse synthetic data as possible while maintaining the instances within the region of the minority class.
3) By generating data instances from two dissimilar instances, the proposed model ensures that overfitting is avoided as it does not create duplicated samples and stops the generation of new instances after balancing the classes.
4) The proposed model demonstrates superior precision, recall, and F-Score compared to other complex and conventional methods while also addressing the over-generalization problem for detecting cyber threats in IIoT.

The rest of the paper is organized as: Section II presented the related work, whereas Section III describes the proposed approach. Section IV presents the result evaluation and section V concludes the work and gives future direction.

II. Related Work

A. Intrusion detection systems

A significant amount of scrutinization and research to achieve a network that is secure, using ML methods is done in both academia and industry because of their high potential benefits. The majority of traditional ML methods employed in intrusion detection are grounded in supervised learning models [12]–[14]. Liang et al. [15] presents a data clustering optimization model. Formally, it sorts data according to the weighted distance and safety factor based on each node's data features properties and the priority threshold. Chang et al. [16] explored the applicability of using the Forest-RI (Random Input), a technique of Random Forest (RF) which uses feature selection, and combined it with an SVM to classify the selected efficient attributes.

Bhattacharya et al. [17] devised a model that firstly employs a hybrid PCA-firefly algorithm to reduce data dimensions before using the XGBoost algorithm for the classification of the reduced data. After the introduction of DL theory, its excellent feature learning capabilities have attracted significant attention from researchers. As a result, several scholars have started incorporating DL methods for intrusion detection [18]–[20]. Shone et al. [21] put forth a non-symmetric deep autoencoder (NDAE) for feature learning and created a new classification model by integrating NDAE with the RF classifier in a unsupervised way.

However, existing Intrusion Detection Systems (IDS) face challenges when handling class imbalance in IIoT environments. Imbalanced data could tend to build models that prioritize majority classes, resulting in decreased detection rates of cyber threats. This issue also causes increased false alarms, inefficient resource allocation, and reduced overall security. Additionally, traditional performance metrics may not accurately reflect the model's capability to detect threats, & the model may struggle to adapt to different or emerging cyber threats due to insufficient minority class representation.

B. Data imbalance processing methods

As stated in the beginning, this paper handles the issue of the class imbalance problem. In the context of cyber threat detection for IIoT, researchers have focused on tackling the class imbalance problem to increase the performance of ML models in detecting cyber threats. The solutions for data imbalance problems in cyber threat detection for IIoT are majorly classified as data based techniques, algorithm based technique, and cost-sensitive learning based techniques [22], [23]. Several studies have investigated the application of oversampling, undersampling, and hybrid resampling techniques to balance the class distribution in IIoT intrusion detection datasets. By modifying the representation of minority and majority classes, these techniques could enhance the performance of ML models in detecting cyber threats in IIoT environments.

In addressing the class imbalance scenario, the most straightforward technique is random oversampling. This method entails selecting minority samples at random and duplicating them until a desired size is reached. However, this can result in overfitting due to the strong similarities between original and duplicated instances. To tackle this challenge, Chawla et al. introduced the Synthetic Minority Oversampling Technique [8], which generates synthetic samples between randomly chosen minority instances and their NN-nearest neighbors, where NN is determined by the user. Despite its advantages, SMOTE may lead to over-generalization, as it generates new instances not taking majority instances into account, thereby increasing the overlap between classes [24]–[26]. This issue can become more pronounced in datasets with high imbalance ratios, as sparse minority instances may end up within the majority class after oversampling, further impairing overall performance [27].

Several methods are developed for overcoming over-generalization. One approach, Safe-level SMOTE [28], cal-

This article has been accepted for publication in IEEE Transactions on Consumer Electronics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCE.2023.3319439

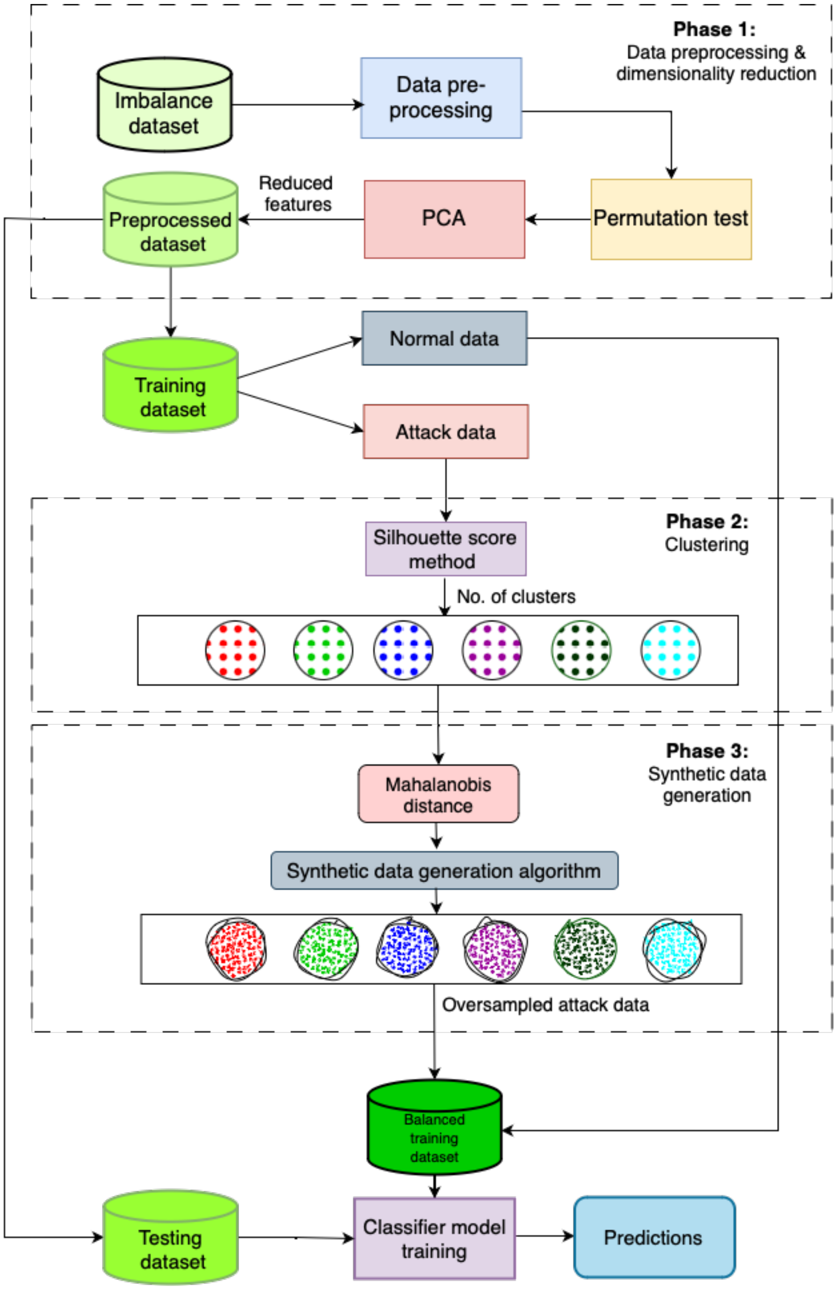


Fig. 1. Block diagram of proposed approach

culates a "safe-level" value for each minority calss and synthesizes samples nearer to the largest safe level. This safe-level value is determined by the amount of minority samples among its nearest neighbors. However, this method may result in overfitting since synthetic samples are created further away from the decision boundary.

Another approach, Borderline-SMOTE (B.SMOTE) [10], finds the boundary in-between the two classes and focuses on oversampling of the samples which exists close to the boundary line. In contrast, ADASYN [9] focuses on the minority class by generating synthetic samples based on their density distribution. ADASYN adaptively generates more synthetic examples for the minority class in regions with fewer instances, effectively balancing the dataset. However, both the techniques may not detect all minority samples in proximity within the boundary of decision.

Alternative sampling methods involve the use of clusters to segment the dataset before applying any sampling strategy. Barua et al. [11] presents a technique called Majority Weighted Minority Oversampling Technique (MWMOTE), that partitions data with the help of clustering before employing the euclidean distance similarity to get closely related class instances. Synthetic instances are then produced depending on weights allocated for minority class instances. Various researchers showed a preference for oversampling instead of under-sampling [29]–[31], as under-sampling eliminates class samples and may lead to the loss of essential information necessary for constructing an efficient predictive model. Thus, undersampling is ruled out from this study.

While our methodology is also developed on synthetic data generation, it distinguishes itself by creating diverse data samples. In comparison to other methods that generate new data instances from highly similar parent samples according to their similarity distance measure values, proposed model creates different data samples originating with help of 2 distinct parent instances.

III. Theory of Inheritance and Diversity

Genes, the fundamental building block of heredity, are thought to be located in chromosomes and are transferred equally to offspring from each parent after the fertilization of the gametes (egg and sperm). Sutton [32] proposed this theory in 1902 and is known as the Chromosomal Theory of Inheritance (CTI). According to the hypothesis, new kids inherit 50% of each parent's chromosomes, making them both identical to and different from both parents at the same time. The sex (or gender) of the species is crucial for maintaining diversity within species because it aids in choosing two opposing members that can procreate. Two unique groups of chromosomes population S = (s_1; s_2; s_3; .... ;s_r) and T = (t_1; t_2; t_3; .... :t_r) are generated with same size r. Every chromosome Z = (z_{j1}; z_{j2}; z_{j3}; .... :z_{jp}) represents a d-dimensional vector comprised of genes, where z_{jk} is the k^{th} gene value (k = 1; 2; ... ; d) of the j^{th} chromosome (j = 1; 2; .... ; p) in each of the population. Two chromosomes, one from each set, are united to create a new offspring, with every parent randomly providing half of the genes the offspring would inherit. Researchers have used CTI to solve problems in different domains such as agriculture and animal research as well [33], [34]. An unbalanced defect dataset is subjected to the basic principles of selective animal and plant mating for generating more data instances of defective instances. Methods including outcrossing, inbreeding, and line-breeding [35] were employed in selective breeding to separate the pool of animals and plants. Similarly to this, by taking into account a similarity metric to help us separate our data samples, we can logically oversample the minority groups. In order to create new samples, we plan to develop an oversampling method based on the CTI and the partitioning approach that can combine various samples to create synthetic data samples which are as unique from their producers as possible & add to the diversity of the minority class distribution. This method will be able to identify minority defective samples and examine their sample defects.

IV. Proposed Approach

We exploited the challenge of using the chromosomal theory of inheritance for generating synthetic data using the gaussian clustering-based gametic oversampling technique. Taking Walter Sutton's theory [32] into consideration we used the features of the attack dataset as the chromosomes of the parents taking part in the reproduction mechanism for generating diverse instances of data. We aim to produce new minority instances of attack labels that inherit the features from two different data instances. These features produced are also unique and not a copy of the existing sample.

Algorithm 1 Gaussian clustering-based gametic hereditical oversampling technique

```
Input value: An Imbalanced Dataset $D_I$
Output value: A Balanced Dataset $D_B$
 1: Use label encoder to add a label to the dataset;
 2: $D_{I.new\_features}$ = PCA.fit($D_{I.features}$), as mentioned in Algo 2;
 3: ($D_{Train}$ and $D_{Test}$) = train-test-split($D_I$);
 4: Now in $D_{Train}$ :
        if y[i] = attack_label then
        Attack_D.append($D_{Train}$.iloc[i,:]),
        where i = {0....len($D_{Train}$)}
 5: Define k = best of Silhouette Score(Attack_D) to get the number of clusters, as mentioned in Algo 3
 6: Create cluster $C_j$ = GMM(Attack_D,k) as described in Phase 2
        $C_j$= $j^{th}$ cluster, j ∈ {0,1,2...,k-1} & k ≈ l (for best results)
        l is the total number of attack labels
 7: Initialize total_count=0 & required_count = no. of attack label needed to balance dataset
 8: In a cluster $C_j$, for each data point $C_{jq}$, obtain Mahalanobis distance $(M_q^2)$
        $(M_q^2) = (C_{jq} - \mu)'\Sigma^{-1}(C_{jq} - \mu)$
        Where $C_{jq}$ = Object vector, j = cluster no.∈ {0,1,2...,k-1},
        q = row or data point of $C_j$, $\mu$ = mean vector, $\Sigma^{-1}$ = covariance matrix
 9: Sort $C_{jq}$ according to $M_q^2$ value in decreasing order
10: Say, mid = length $(C_j)$/2,
        Divide $(C_j)$ into upper and lower divisions as
        $X_u$ = $(C_j)$.elements from [0 to mid]
        $X_l$ = $(C_j)$.elements from [mid to length $(C_j)$]
11: Sort $X_l$ and $X_u$ on basis of attack labels.
12: Synthetic Data $(SD)_j$ = SDG ($X_l$ and $X_u$)
        total_count = total_count + len($(SD)_j$)
13: If total_count < required_count
        repeat steps 7 to 11 for the remaining clusters
14: Else;
        End
```

Based upon the same attack label of two data samples, the child class is produced that will have the same attack label as its parents. The basic intuition behind this could be explained with an example as, let's say both the male and female have brown iris then the chances of their child having brown iris are high as compared to having blue, black, or green iris. Such a concept is known as inheritance, as explained in Walter Sutton's chromosomal theory of Inheritance [32]. The proposed approach seeks to produce synthetic data that has both unique and common features with the help of three fundamental phases. The overall proposed approach is shown in Figure 1 and a list of all notations used in the the proposed model is shown in Table I.

TABLE I
MEANING AND NOTATIONS

Meaning	Notation
Imbalanced dataset	D_I
Balanced dataset	D_B
Training data	D_{Train}
Testing data	D_{Test}
Cluster	C_j
Total no. of attack labels in dataset	l
Mahalanobis distance for sample q	M_q
Upper-division in cluster j	X_u
Lower-division in cluster j	X_l
Optimal no. of features for PCA	f_1
Weight of feature j giving feature i	W_{ij}
Total no. of features of dataset	dt
Reduced feature dataset	D_t
Optimal no. of clusters	k
Data sample	x

Before generating the synthetic instances for the minority dataset, data pre-processing is applied to the complete dataset. As a part of this phase, the feature selection technique is also applied to the preprocessed dataset to reduce the overall number of attributes of the dataset. After feature selection comes to the outlier detection step using Isolation Forest. Then, the preprocessed data is divided into train and test sub-datasets. In the second phase, the minority samples of the training dataset are divided into various clusters based on the clustering algorithm. The idea behind clustering is to increase the accuracy of the offspring falling into the vicinity of its parent.

As a part of the third Phase, Mahalanobis Distance (MD) says M_q for every data sample of cluster C_i is calculated. The data samples are then arranged in descending order based on their MDs. Then, the data samples of the cluster C_i are halved into upper and lower segments from its midpoint. The data elements of the upper and lower sections are arranged in a specific order (either ascending or descending) based on their attack label. In the last step, the parents with the same attack label are chosen from upper and lower segments respectively to breed and produce offspring which fall into the same class label as their parents. The breeding part includes taking the average of the feature values of its parents to get the features of the child. Algorithm 1 describes the entire process of the proposed oversampling technique.

A. Phase 1: Data pre-processing and dimensionality reduction

In data pre-processing, we applied, label encoding to convert categorical data into numeric and Min-Max Scaling to bring the data in one range. Then further we applied dimensionality reduction to the pre-processed data. Since, for effective clustering lower number of features works well so, we applied principal component analysis (PCA) on the scaled data to reduce its number of features. But due to the human factor of assigning the number of components and reducing the human error a method called permutation test is used on the mined dataset to identify the true number of features that could effectively represent the whole features of the dataset. After gathering the number of attributes, we applied PCA to the dataset, described in Algorithm 2.

Let W_{ij} be the weight of feature j giving PCA feature i
$P_{ij} = W_{i1}x_1 + W_{i2}x_2 + ...W_{in}x_n$
$i = \{0, 1, 2, ...f_1\}$

Where f_1 = Optimum no. of features and P is the principal components
$j = \{0, 1, 2, ...dt\}$

Algorithm 2 Permutation test and PCA for feature selection

```
Input: Preprocessed data $D_I$ with n no. of features
Output: Reduced feature dataset
Define N as no. of permutation and correlation dataset;
        (i) X_aux = data.copy()
        (ii) For each Column in data.column
        X_aux[col] = data[col].samples(len($D_I$)).value
        (iii) return X_aux
Run PCA for $D_I$ and save variance by each $P_i$
Plot a graphical view for analysis of explained variance
  v/s permuted versions
The knee point in the plot would be the desired number
  of features '$f_1$'
$D_I = PCA(D_I, f_1)$
Return $D_I$
```

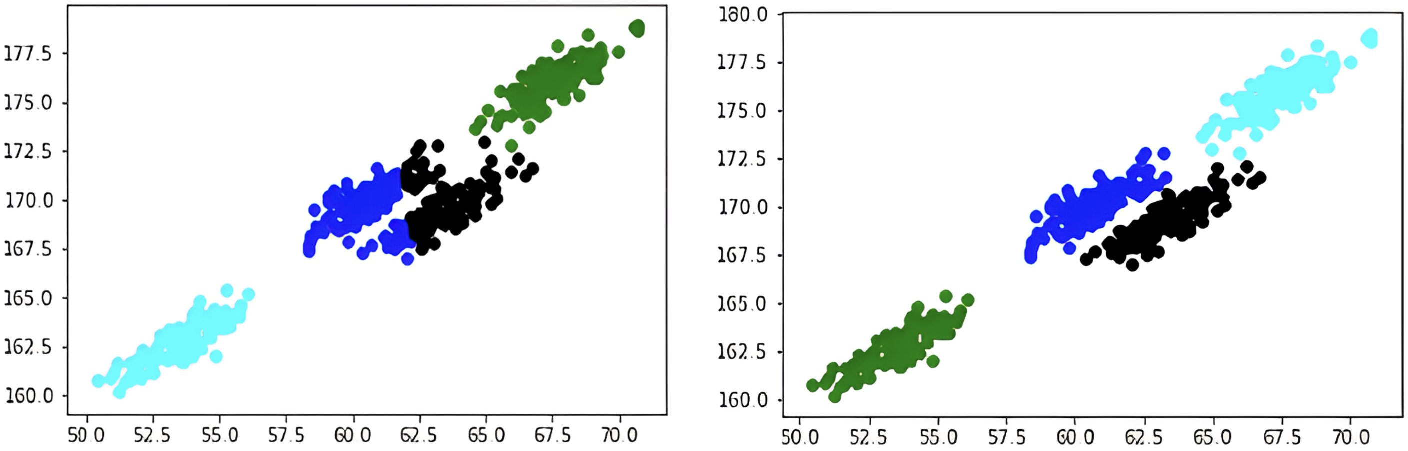
The next step is to remove the outliers, so we opted for an isolation forest to remove the outliers from dataset. Then on the reduced and pre-processed dataset, train-test-split is applied to divide data into train and test datasets. The training data is again scrutinized for separating the majority & minority classes. Here, minority class generally falls under the criterion of attack. The data samples of these classes are separated from the majority class and are sent to the next step for synthetic data generation.

B. Phase 2: Clustering

Clustering is an unsupervised learning methodology used to split the data into clusters (say groups) to classify them uniquely. In a broad sense, the cluster is a collection of data instances which share more similarities with each other than with those in different clusters. Before performing oversampling in our data, we are interested in creating clusters of attack data representing the minority class to oversample.

We preferred the Gaussian clustering method over k-means clustering as k-means relies on distance measures to assign data points to clusters, resulting in circular-shaped clusters. This occurs as the cluster centroids are iteratively updated using the mean value. Circular shape of clusters makes this technique less effective when applied to datasets where the attack data may not be distributed in a circular pattern. If the data points form a non-circular pattern, K-means would struggle to identify the correct clusters, and consequently, the accuracy of the synthetic data would be compromised due to this inefficiency. Figure 2 shows the creation of a cluster using k-means and GMM.

In this work, the silhouette measure (or silhouette coefficient) is employed to determine the quality of clustering and to



(a) KMeans

(b) GMM

Fig. 2. Cluster creation using (a) KMeans (b) GMM

Algorithm 3 Silhouette Scores

Input: Reduced feature Dataset D_t

Output: ‘k’ Number of clusters to be created

```
Initialize silhouette.score = []
For i in range (l+2), where l is no. of attack labels
    gmm = gaussian_mixture(n_component = i,n_init = 50 (say), init_param = Kmeans++)
    gmm.fit($D_t$)
    Silhouette_score.append(Silhouette_score($D_t$), gmm.predict($D_t$))
Plot Silhouette_score v/s i graph
Return max[Silhouette_score] $\longleftarrow$ $i^{th}$ value
```

evaluate the optimal number of clusters ‘k’ for a given dataset as mentioned in Algorithm 3. It provides a way to measure how effectively every data point is assigned to its respective cluster by taking into account both the cohesion within the cluster and the separation between different clusters. Silhouette coefficient is formulated as:

$$S_i = \frac{b_i - a_i}{max(a_i, b_i)} \tag{1}$$

Where,
b_i = min. average distance from the i^{th} point to points in different cluster,
a_i = average distance from i^{th} point to other points within same cluster.

The silhouette coefficient ranges from -1 to 1. Its values close to 1 indicates a high quality clustering assignment, values near 0 imply that the data point could belong to either its current cluster or a neighboring cluster, and negative values suggest that the data point could have been assigned to the incorrect cluster.

After getting the number of clusters ‘k’, Gaussian Mixture Model (GMM) is used to assign elements to the clusters. With GMM each cluster is represented by a gaussian distribution. GMM estimates the parameters of these gaussian distributions and assigns data instances to clusters depending on their probability of belonging to each distribution.

When implementing GMM to cluster, the goal is to partition the data into groups or clusters, where every cluster is depicted by a gaussian distribution. The steps for creating clusters using GMM are as follows:

1) **Initialization:** Select the ‘k’ number of clusters (given by Silhouette Score) for the dataset. Initialize the parameters of the gaussian distributions (means μ_i, covariance matrices Σ_i) and the mixing coefficients w_i using K-means++ initialization.

2) **Expectation step (E-step):** Given the current estimates of parameters, calculate the probabilities of all data points hailing to every gaussian distribution using Bayes' theorem:

$$P(z_i = k|x_n) = \frac{(w_k * N(x_n; \mu_k, \Sigma_k))}{\Sigma(w_j * N(x_n; \mu_j, \Sigma_j))} \tag{2}$$

where:
z_i is cluster assignment of data instance x_n, $P(z_i = k|x_n)$ represents the probability that x_n belongs to cluster k. $N(x_n; \mu_k, \Sigma_k)$ is the Gaussian distribution with mean μ_k and covariance matrix Σ_k for cluster k. w_k is the mixing coefficient for cluster k. The summation is over all clusters $(j = 1, 2, ..., K)$.

3) **Maximization step (M-step):** Update the estimates of the Gaussian parameters (μ, Σ) and the mixing coefficients (w) based on the probabilities computed in the E-step:

$$\mu_k = \frac{(\Sigma P(z_i = k|x_n) * x_n)}{(\Sigma P(z_i = k|x_n))} \tag{3}$$

$$\Sigma_k = \frac{(\Sigma P(z_i = k|x_n) * (x_n - \mu_k)(x_n - \mu_k)')}{(\Sigma P(z_i = k|x_n))} \tag{4}$$

$$w_k = \frac{(\Sigma P(z_i = k|x_n))}{N} \tag{5}$$

where, the summations are over all data points $(n = 1, 2, ..., N)$. N is the total no. of data points in the dataset.

4) **Convergence step:** Repeat steps 2 (E-step) and 3 (M-step) until the GMM parameters (μ, Σ, w) converge or a stopping criterion is matched (e.g., a max. no. of iterations or a min. change in the log-likelihood).

5) **Cluster assignment:** After convergence, add every data point to a cluster with the highest probability:

$$cluster(x_n) = argmax_k P(z_i = k|x_n) \tag{6}$$

C. Phase 3: Synthetic Data Generation

First, we calculate the MD and sort data points of cluster C_i in descending order based on MD. Then we divide the sorted data samples of cluster C_i from the previous step into two groups X_l and X_u by identifying a mid-value, which is half of the total number of points in the cluster. After several iterations of simulation trials, using the upper half and lower half yields the best and most universal outcome of synthetic data. This strategy still functions even when we have just a less number of attack samples. We think that using two parents is practical and is in favor of inheritance theory. One partition say X_u is made up of all data samples with MD greater than or equal to the center data point, and the second partition say X_l is made up of all other data samples. Samples are progressively tagged and then paired within the two partitions. To actively identify 2 separate examples from both segments which have been methodologically and symmetrically matched as "parents" are taken from X_l and X_u with the same attack category. Before the pairing process, the data sample in each upper and lower half are again sorted on the basis of their attack label. The pairing is then carried out sequentially using SDG, described in Algorithm 4. These parents can be recognized by the "attack labels" they have. This is done to make sure that there are no samples that overlap and that the resulting samples created by the partitioning process are located inside the cluster decision boundary. The child, hence produced, is generated by taking the average value from the feature values of parents.

The decision to use MD instead of Euclidean distance (ED) is motivated by the limitations of ED when dealing with correlated attribute data. MD, measures how far a point is from its distribution (such as a cluster), is better suited for capturing proximity information. By considering the diversity between two data examples using MD, we can overcome drawbacks associated with the Euclidean metric. ED fails to differentiate strongly correlated or duplicate samples, impeding the classifier training process. MD, with its unitless nature, provides a relative measurement of sample distance from a reference point, facilitating outlier detection and similarity identification between known and unknown datasets. MD accounts for correlation and scale, making it a robust measure. To address scale and correlation concerns in ED, covariance within the data instances is also taken into consideration during distance calculation.

Algorithm 4 Synthetic Data Generation (SDG)

```
Input: $X_l$ and $X_u$ and the required count
Output: Synthetic data for minority samples
Initialize i = 0, j = 0, check = 0
X_new = []
While check < required_count and i < len($X_l$) and j < len($X_u$)
    if j $\geq$ len ($X_l$)
    j = 0
    i+=1
    if $label_{(X_l)i}$ = $label_{(X_u)j}$
    feature_of_child = average($X_{l\_feature}$, $X_{u\_feature}$)
    check+=1
    X_new.append(feature_of_child)
    else
    j+=1
Return X_new
```

The mathematical term for the same is:

$$M_q^2 = (x - m)^T.C^{-1}.(x - m) \tag{7}$$

Where M_q^2 denotes the squared MD, the mean of independent variables is represented by m, x represents the data samples, and C^{-1} denotes the inverse covariance matrix of independent variables.

Considering two samples of the defective class $x = (x_1, x_2, ...x_n)^T$ and $y = (y_1, y_2, ...y_n)^T$, the MD is evalauted as ;

$$M_q(x, y) = \sqrt{(x - y)^T S^{-1}(x - y)} \tag{8}$$

Where S^{-1} is the covariance matrices. By aggregating and calculating the average of two paired samples from both partitions, new data is generated which is then added to the data X_new, known as the final phase of the proposed model.

V. Experiment and Anaalysis

A. Dataset used

For experimental validation, we employ the widely recognized UNSW-NB15 [36] dataset which is majorly utilized in the field of attack detection. It has been extensively used in research & provides a diverse range of network traffic scenarios to evaluate the efficiency of attack detection systems. UNSW-NB15 dataset is also highly imbalanced in nature thus it is also used to evaluate the solutions for the imbalance data problem in network intrusion detection systems. The dataset was created to simulate real-world network traffic scenarios. The dataset comprises network traffic data collected in a controlled environment, including both benign and nine types of attacks with 49 features with the class label. It serves as a valuable resource to evaluate and develop intrusion detection systems in diverse network environments.

B. Evalaution metrics

When evaluating an intrusion detection approach, several metrics are commonly used to assess its performance. These metrics provide insights into the efficacy of the approach in detecting and classifying intrusions. Some of the key metrics used for evaluating intrusion detection approaches include accuracy, recall, precision, F-Score, Area Under Curve (AUC), and False Positive Rate (FPR). Accuracy measures the overall correctness of the intrusion detection approach for classifying attack and normal samples. Precision presents the ratio of correctly classified attack requests over total samples classified as an attack. Recall (or Sensitivity or True Positive Rate) represents the correctly classified attack over actual attack samples. F-Score gives the harmonic mean of precision and recall, which provides a balanced measure of both metrics. AUC quantifies the model's ability to discriminate between positive and negative instances across various threshold settings, providing an overall performance measure. FPR represents the incorrect prediction of attack samples.

$$Accuracy = \frac{w + x}{w + x + y + z} \tag{9}$$

$$Precision = \frac{w}{w + y} \tag{10}$$

$$Recall(TPR) = \frac{w}{w + z} \tag{11}$$

$$F - Score = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{12}$$

$$FPR = \frac{y}{y + x} \tag{13}$$

Where w is true positive, x is true negative, y is false positive and z is false negative respectively.

C. Experimentation procedure

We used Keras* and TensorFlow† frameworks to develop the model and conduct experimental testing. As described in Section 4, the original data is processed from preprocessing step, and then outlier detection is performed using an isolation forest. On UNSW-NB15 label encoding is applied for categorical data, making the data values numeric. Next, we used PCA to reduce the dimensionality of the features. We determined the number of PCA components to be used for feature selection by testing the final classification effect of RF. The dimension of data in the UNSW-NB15 data set initially was 47 (+2 of output), and we reduced it to 14 features to be used.

In the proposed model-based oversampling, to identify the number of clusters, the silhouette score method is used which determined that 4 number of clusters suits best for attack dataset. Then GMM was initialized with the parameters as $n\_init = 50$, $init\_params = kmeans{+}{+}$, and $reg\_covar = 1$ to create the clusters. Once the clusters are created then we calculated MD between the data points within each cluster. Next every data value was arranged in descending order depending on their calculated MD and the cluster was divided into upper and lower half. Then the parents are chosen from the upper and lower half with the same attack labels to generate the child by averaging the parent's feature values. After the oversampling process, an RF classifier is applied for model training on the combined dataset of old traces with newly generated samples.

TABLE II
CLASSIFIERS AND THEIR CONFIGURATIONS

Classifier	Configuration
RF	n_estimators = 100, random_state = 42
KNN	Number of neighbors = 5, weights = 'uniform', metric = 'minkowski'
LR	penalty ='l2', random_state = 42, multi_class = 'auto'
SVM	kernel = 'linear', random_state = 42, C = 1
DT	criterion = 'entropy', random_state = 42

D. Analysis of experimental results

1) Results of binary classification: This section presents the comparison of the proposed model for the classification results of benign and attack instances on the UNSW-NB15 dataset. Table III shows the comparison of proposed with other oversampling methods such as SMOTE, ADASYN, Random, and Borderline SMOTE (B. SMOTE) on different classifiers with table II describing the configuration of each classifier.

The evaluation results for different sampling techniques and classifiers in Table III show varying performance in the context of attack detection.

- SMOTE: SMOTE generally performs well across the evaluated classifiers, with high precision, recall, and F-Scores. It effectively addresses the class imbalance issue by generating synthetic samples, resulting in improved detection of minority instances. The FPR is relatively low, indicating a good balance between false positives and true positives.
- ADASYN: ADASYN also demonstrates favorable performance in terms of given performance metrics. It assigns weights to minority instances based on their neighborhood, resulting in more effective oversampling. However, the performance of ADASYN is a bit lesser than SMOTE in terms of precision and recall, particularly for some classifiers.
- Random: Random oversampling demonstrates consistent performance across various classifiers. However, it carries the risk of overfitting as it generates similar instances. Nevertheless, this technique exhibits high precision, recall, and F-Scores, highlighting its effectiveness in enhancing the detection of positive instances.
- Borderline SMOTE: Borderline SMOTE performs well in terms of given performance metrics. It focuses on generating synthetic instances along the borderline between the two classes, improving the detection of minority class instances. However, its performance is relatively lower compared to SMOTE and ADASYN, particularly in terms of precision and recall.

*https://keras.io/api/

†TensorFlow is an open-source library used in Python for deep learning applications

TABLE III
EXPERIMENTAL RESULTS OF BINARY CLASSIFICATION ON DIFFERENT PERFORMANCE METRICS FOR UNSW-NB15 DATASET BETWEEN PROPOSED AND OTHER OVERSAMPLING TECHNIQUES

Sampling Technique	Classifier	Precision	Recall	F-Score	FPR	AUC	ACC
SMOTE	RF	**0.917**	**0.917**	**0.917**	**0.089**	**0.911**	**0.917**
	KNN	0.898	0.894	0.895	0.106	0.894	0.894
	LR	0.784	0.764	0.768	0.230	0.770	0.764
	SVM	0.879	0.880	0.879	0.138	0.862	0.880
	DT	0.891	0.889	0.890	0.114	0.886	0.889
ADASYN	RF	0.910	**0.903**	**0.904**	**0.090**	**0.910**	**0.903**
	KNN	0.892	0.882	0.884	0.110	0.890	0.882
	LR	0.789	0.741	0.746	0.232	0.768	0.741
	SVM	**0.929**	0.802	0.861	0.153	0.847	0.834
	DT	0.890	0.885	0.886	0.113	0.887	0.885
Random	RF	**0.918**	**0.918**	**0.918**	**0.091**	**0.909**	**0.918**
	KNN	0.897	0.894	0.895	0.106	0.894	0.894
	LR	0.782	0.763	0.767	0.231	0.769	0.763
	SVM	0.880	0.880	0.880	0.135	0.865	0.880
	DT	0.894	0.892	0.893	0.112	0.888	0.895
Borderline SMOTE	RF	**0.912**	**0.905**	**0.906**	**0.089**	**0.911**	**0.905**
	KNN	0.895	0.886	0.888	0.107	0.893	0.886
	LR	0.789	0.739	0.744	0.233	0.767	0.739
	SVM	0.852	0.833	0.836	0.155	0.845	0.833
	DT	0.887	0.882	0.883	0.116	0.884	0.882
Proposed	RF	**0.977**	**0.991**	**0.981**	**0.068**	**0.932**	**0.973**
	KNN	0.966	0.968	0.967	0.071	0.923	0.967
	LR	0.834	0.780	0.800	0.292	0.708	0.780
	SVM	0.942	0.944	0.941	0.156	0.844	0.944
	DT	0.961	0.961	0.961	0.074	0.926	0.961

- Proposed: The proposed model outperforms other oversampling methods in terms of precision, recall, F-Score, and AUC for most classifiers. It demonstrates superior performance in handling the class imbalance issue, resulting in highly accurate and effective intrusion detection.

The proposed model shows promising results, indicating its potential for enhancing the detection of cyber threats in the context of IIoT.

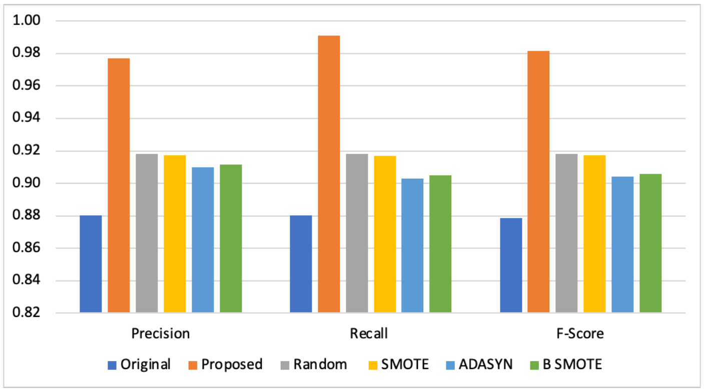


Fig. 3. Comparion of results for binary classification

Figure 3 shows that on the UNSW-NB15 data set, precision, recall, and F-Score enhanced to 6%, 8%, and 7% respectively in comparison to the existing technique achieving the highest results. The high metrics of the proposed approach are owed to the fact that the generated minority data is the subset of the original data. It is built upon the foundation that the produced data should fall under the same cluster with its domain value in the range of its parents. Here, the data values are generated by considering the average of the values of its parent which fall inside the parent dominion. A bold view in the Table III indicates the outcome of the presented approach indicating that the proposed model along with RF achieves 97.3% of accuracy which is greater than the highest accuracy of 91.8% shown by the random oversampling technique. Along with accuracy proposed model achieved a 6% increased precision over all existing techniques by showing a precision of 97.7%. In addition to precision, a boost of 8% is achieved in recall value with 99.1% being the actual number. Whereas F-Score show a significant value of 98.1% with an increased value of 7% in comparison to the highest value of other oversampling technique. These findings demonstrate the effectiveness of the proposed model in detecting attacks, thereby enhancing the security of the IIoT system against potential cyber threats.

Techniques like B. SMOTE, SMOTE, ADASYN, and random oversampling are seen to perform lower to provide accurate synthetic data in comparison to proposed model. This accounts for various facts such as in the case of SMOTE and random oversampling, they may generate samples that might not actually represent the underlying distribution and introduce noise in the datase [11]. Randomly duplicating samples leads to dilution of the unique existing patterns of the dataset [11], whereas in the case of B.SMOTE, it may generate comparatively better samples than SMOTE and random technique but the process of selection of correct borderline samples for oversampling adds challenges to it and incorrect selection of such leads of suboptimal performance. Also, the challenges in estimating the density distribution ratio between majority and minority classes account for the suboptimal performance of ADASYN in comparison to the proposed model. Since the data samples are generated based on the specificity of data points within each cluster, it easily handles the concept of density distribution, no borderline samples need to be selected thus reducing the noise in the dataset as compared to SMOTE and allowing the system to preserve its nature making it less prone to dilution.

2) Multiclass classification results: Handling multiclass classification is a bit more arduous than binary. As in the case of multiclass the imbalance ratio gradually tends to increase in comparison to binary data values.

Table IV presents the results for multi-class classification of proposed oversampling in comparison to classification without oversampling (original) on selected features using PCA and RF as classifiers. Unlike binary classification, multi-class classification involves more complex data, often with potential class overlaps among different pairs of classes. This complexity poses additional challenges in accurately classifying instances into multiple categories. Results show that the proposed oversampling technique achieved good results in comparison to the original approach where no oversampling is performed.

TABLE IV
COMPARISON OF RESULTS FOR MULTI-CLASSIFICATION OF ATTACK CATEGORY IN UNSW-NB15 WITH AND WITHOUT PROPOSED MODEL

	Original			Proposed		
Class	Recall	Precision	F-Score	Recall	Precision	F-Score
A	0.002	0.012	0.003	0.120	0.125	0.122
B	0.002	0.067	0.004	0.139	0.158	0.133
C	0.030	0.055	0.039	0.453	0.431	0.442
D	0.241	0.172	0.201	0.642	0.637	0.640
E	0.071	0.098	0.083	0.722	0.742	0.732
F	0.223	0.228	0.225	0.992	0.998	0.993
G	0.386	0.359	0.372	0.873	0.961	0.877
H	0.040	0.055	0.046	0.850	0.868	0.859
I	0.003	0.006	0.004	0.885	0.853	0.869
J	0.000	0.000	0.000	0.000	0.000	0.000
Average	0.100	0.105	0.098	0.568	0.577	0.567

Further, the proposed model is also compared with other states of art oversampling methods against precision, recall, and F-Score for the multi-classification of different attacks present in the UNSW-NB15 dataset. Figure 4 shows that the proposed model consistently outperforms the other sampling techniques across 8 out of 10 classes. It achieves higher precision values for all classes except for classes A and J. Similarly, recall of the proposed model outperforms the other sampling techniques across 7 out of 10 classes and 8 out of 10 classes for F-Score as shown in Figure 5 and 6 respectively.

This indicates that the proposed model improves the precision, recall, and F-Score of the minority classes and is effective in accurately detecting instances from these classes. The proposed model is effective for detecting attack samples of classes even with low training samples, thus securing the IIoT from these rare cyber threats as well.

Similarly, among the other sampling techniques, Random, SMOTE, ADASYN, and B SMOTE show varying levels of improvement compared to the no oversampling (original). Here, to maintain the same level of comparison, the actual imbalanced dataset is passed through PCA to generate the dataset with the reduced number of features and then used RF as for classification purposes. However, their precision, recall, and F-Score values are generally lower than those achieved by proposed, indicating that the proposed model provides better precision for most classes.

It's worth noting that the performance of the different techniques varies across the classes. Some techniques may perform better for certain classes while underperforming for others. This suggests the importance of selecting an appropriate sampling technique based on the specific class characteristics and the desired performance.

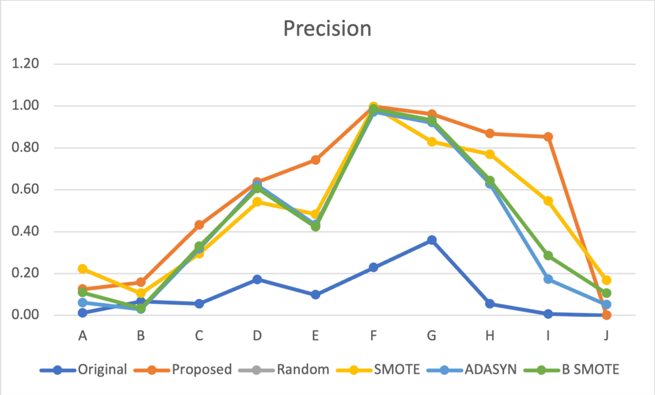


Fig. 4. Precision comparison for multi-classification on UNSW-NB15 dataset

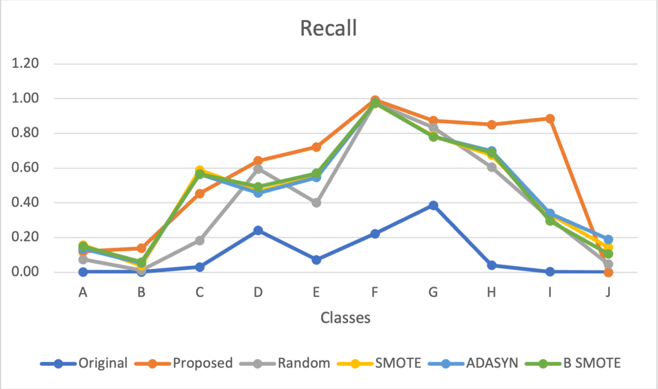


Fig. 5. Recall comparison for multi-classification on UNSW-NB15 dataset

Figure 7 presents a comparison of mean results for multi-class classification on the UNSW-NB15 dataset. The proposed method shows substantial improvement in all metrics. It achieves the best results for average recall, precision, and F-Score among all the techniques, indicating its effectiveness in detecting instances from multiple classes accurately.

Among the other sampling techniques, SMOTE, ADASYN, and B.SMOTE demonstrate relatively better performance compared to the original dataset but still, fall short when compared to the proposed method. The random oversampling technique shows the lowest performance among all the techniques. The better performance of the proposed model accounts here for the fact that the proposed technique uniquely focuses on

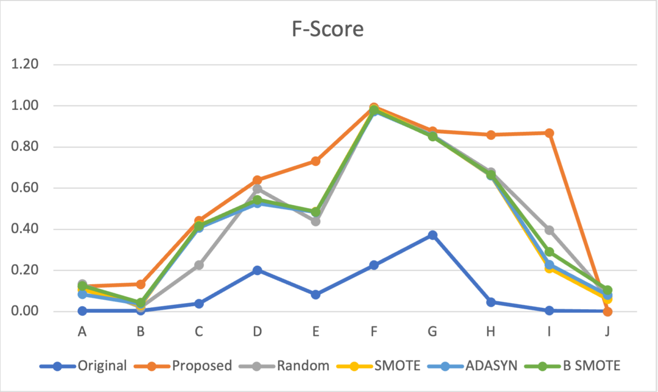


Fig. 6. F-Score comparison for multi-classification on UNSW-NB15 dataset

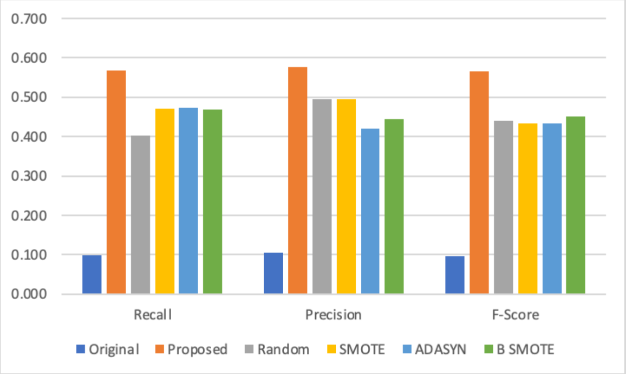


Fig. 7. Comparison of Mean results for multi-classification on UNSW-NB15 dataset

the specific classes and tries to produce an exact amount of samples of them to make them balanced. These samples are generated in such a way that they fall under the same space or distribution thus making those samples closely similar to their class.

Overall, the results highlight the superiority of the proposed method in terms of average recall, precision, and F-Score, suggesting its potential to enhance multi-class intrusion detection on the UNSW-NB15 dataset.

VI. Conclusion

Ensuring network security relies heavily on effective intrusion detection. However, the presence of imbalanced data poses a significant challenge to the performance of intrusion detection systems. Imbalanced learning, where one class dominates the dataset, can lead to biased models and lower accuracy in detecting intrusions. Therefore, addressing the issue of imbalanced learning is crucial for enhancing the overall effectiveness of intrusion detection systems and enhancing network security. This work proposed a gametic hereditical oversampling technique that successfully tackles this challenge by generating diverse synthetic minority instances inspired by genetic biology principles. The evaluation using the UNSW-NB15 dataset validates the effectiveness of the proposed model in accurately detecting cyber threats, by achieving a high precision value of 0.977, and recall value of 0.991, and an F-Score value of 0.981. Importantly, the proposed model prevents over-generalization by ensuring the spread of synthetic samples remains within the boundaries of the minority class. The superiority of the proposed model over conventional methods highlights its potential for developing robust and efficient machine-learning models that enhance the security of IIoT systems. It enhances the precision, recall, and F-Score values by 6%, 8%, and 7% respectively as compared with the conventional methods, thus establishing its supremacy in securing the IIoT from the cyber threats.

However, the limitation of the proposed approach is that it would require a certain amount of samples of each class to produce synthetic data belonging to that class. If all the samples fall in either upper or in lower division of the class then it would be difficult to produce new samples for that specific class. In the future, we would work on improving the proposed approach to focus more on individual attack labels and try to address the above-mentioned limitation of the proposed approach.

Acknowledgments

The research presented is emanated by the European Union’s Horizon 2020 program for research and innovation, which provided funding under Grant Number 847577 (SMART 4.0 Marie Skłodowska-Curie Actions COFUND), and also by a grant from Science Foundation Ireland (SFI) under Grant Number SFI/12/RC/3918 and also SFI 12/RC/2289-P2 (Insight). In order to promote open access, the author has chosen to apply a CC BY public copyright license to any version of the Author Accepted Manuscript that results from this submission.

References

[1] Z. Ahmad, A. Shahid Khan, C. Wai Shiang, J. Abdullah, and F. Ahmad, “Network intrusion detection system: A systematic study of machine learning and deep learning approaches,” *Transactions on Emerging Telecommunications Technologies*, vol. 32, no. 1, p. e4150, 2021.

[2] S. Bagui and K. Li, “Resampling imbalanced data for network intrusion detection datasets,” *Journal of Big Data*, vol. 8, no. 1, pp. 1–41, 2021.

[3] H. Ding, L. Chen, L. Dong, Z. Fu, and X. Cui, “Imbalanced data classification: A knn and generative adversarial networks-based hybrid approach for intrusion detection,” *Future Generation Computer Systems*, vol. 131, pp. 240–254, 2022.

[4] G. M. Weiss and F. Provost, “The effect of class distribution on classifier learning: an empirical study,” tech. rep., Rutgers University, 2001.

[5] K. Yoon and S. Kwek, “A data reduction approach for resolving the imbalanced data issue in functional genomics,” *Neural Computing and Applications*, vol. 16, pp. 295–306, 2007.

[6] A. Thakkar and R. Lohiya, “Attack classification of imbalanced intrusion data for iot network using ensemble learning-based deep neural network,” *IEEE Internet of Things Journal*, 2023.

[7] H. Ding and X. Cui, “A clustering and generative adversarial networks-based hybrid approach for imbalanced data classification,” *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–16, 2023.

[8] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “Smote: synthetic minority over-sampling technique,” *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[9] H. He, Y. Bai, E. A. Garcia, and S. Li, “Adasyn: Adaptive synthetic sampling approach for imbalanced learning,” in *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence)*, pp. 1322–1328, IEEE, 2008.

[10] H. Han, W.-Y. Wang, and B.-H. Mao, “Borderline-smote: a new oversampling method in imbalanced data sets learning,” in *Advances in Intelligent Computing: International Conference on Intelligent Computing, ICIC 2005, Hefei, China, August 23-26, 2005, Proceedings, Part I 1*, pp. 878–887, Springer, 2005.

[11] S. Barua, M. M. Islam, X. Yao, and K. Murase, “Mwmote–majority weighted minority oversampling technique for imbalanced data set learning,” *IEEE Transactions on knowledge and data engineering*, vol. 26, no. 2, pp. 405–425, 2012.

[12] D. Papamartzivanos, F. G. Mármol, and G. Kambourakis, “Dendron: Genetic trees driven rule induction for network intrusion detection systems,” *Future Generation Computer Systems*, vol. 79, pp. 558–574, 2018.

[13] S. Roshan, Y. Miche, A. Akusok, and A. Lendasse, “Adaptive and online network intrusion detection system using clustering and extreme learning machines,” *Journal of the Franklin Institute*, vol. 355, no. 4, pp. 1752–1779, 2018.

[14] T. Hamed, R. Dara, and S. C. Kremer, “Network intrusion detection system based on recursive feature addition and bigram technique,” *Computers & Security*, vol. 73, pp. 137–155, 2018.

[15] W. Liang, K.-C. Li, J. Long, X. Kui, and A. Y. Zomaya, “An industrial network intrusion detection algorithm based on multifeature data clustering optimization model,” *IEEE Transactions on Industrial Informatics*, vol. 16, no. 3, pp. 2063–2071, 2019.

[16] Y. Chang, W. Li, and Z. Yang, “Network intrusion detection based on random forest and support vector machine,” in *2017 IEEE international conference on computational science and engineering (CSE) and IEEE international conference on embedded and ubiquitous computing (EUC)*, vol. 1, pp. 635–638, IEEE, 2017.

[17] S. Bhattacharya, P. K. R. Maddikunta, R. Kaluri, S. Singh, T. R. Gadekallu, M. Alazab, and U. Tariq, “A novel pca-firefly based xgboost classification model for intrusion detection in networks using gpu,” *Electronics*, vol. 9, no. 2, p. 219, 2020.

[18] H. Wang, Z. Cao, and B. Hong, “A network intrusion detection system based on convolutional neural network,” *Journal of Intelligent & Fuzzy Systems*, vol. 38, no. 6, pp. 7623–7637, 2020.

[19] H. Choi, M. Kim, G. Lee, and W. Kim, “Unsupervised learning approach for network intrusion detection system using autoencoders,” *The Journal of Supercomputing*, vol. 75, pp. 5597–5621, 2019.

[20] P. Devan and N. Khare, “An efficient xgboost–dnn-based classification model for network intrusion detection system,” *Neural Computing and Applications*, vol. 32, pp. 12499–12514, 2020.

[21] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, “A deep learning approach to network intrusion detection,” *IEEE transactions on emerging topics in computational intelligence*, vol. 2, no. 1, pp. 41–50, 2018.

[22] J. F. Díez-Pastor, J. J. Rodríguez, C. I. García-Osorio, and L. I. Kuncheva, “Diversity techniques improve the performance of the best imbalance learning ensembles,” *Information Sciences*, vol. 325, pp. 98–117, 2015.

[23] M. Galar, A. Fernandez, E. Barrenechea, H. Bustince, and F. Herrera, “A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches,” *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 4, pp. 463–484, 2011.

[24] H. He and E. Garcia, “Learning from imbalanced data ieee transactions on knowledge and data engineering,” *vol*, vol. 21, pp. 1263–1284, 2009.

[25] S.-J. Yen and Y.-S. Lee, “Cluster-based under-sampling approaches for imbalanced data distributions,” *Expert Systems with Applications*, vol. 36, no. 3, pp. 5718–5727, 2009.

[26] V. López, A. Fernández, S. García, V. Palade, and F. Herrera, “An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics,” *Information sciences*, vol. 250, pp. 113–141, 2013.

[27] S. Kotsiantis, D. Kanellopoulos, and P. Pintelas, “Handling imbalanced datasets: A review, gests international transactions on computer science and engineering 30 (2006) 25–36,” *Synthetic Oversampling of Instances Using Clustering*.

[28] C. Bunkhumpornpat, K. Sinapiromsaran, and C. Lursinsap, “Dbsmote: density-based synthetic minority over-sampling technique,” *Applied Intelligence*, vol. 36, pp. 664–684, 2012.

[29] A. A. Shanab, T. M. Khoshgoftaar, R. Wald, and A. Napolitano, “Impact of noise and data sampling on stability of feature ranking techniques for biological datasets,” in *2012 IEEE 13th International Conference on Information Reuse & Integration (IRI)*, pp. 415–422, IEEE, 2012.

[30] N. Japkowicz and S. Stephen, “The class imbalance problem: A systematic study,” *Intelligent data analysis*, vol. 6, no. 5, pp. 429–449, 2002.

[31] V. García, J. S. Sánchez, and R. A. Mollineda, “On the effectiveness of preprocessing methods when dealing with different levels of class imbalance,” *Knowledge-Based Systems*, vol. 25, no. 1, pp. 13–21, 2012.

[32] W. S. Sutton, “The chromosomes in heredity,” *The Biological Bulletin*, vol. 4, no. 5, pp. 231–250, 1903.

[33] T. Raza and K. Pandav, “Chromosome manipulations for crop improvement,” *Biotechnologies and Genetics in Plant Mutation Breeding: Volume 2: Revolutionizing Plant Biology*, 2023.

[34] M. Johnsson, “Genomics in animal breeding from the perspectives of matrices and molecules,” *Hereditas*, vol. 160, no. 1, pp. 1–11, 2023.

[35] H. S. Hamad, E. E. Gewaily, A. Elmoghazy, and M. Elsayed, “Outcrossing rate as influenced by optimizing the method of parental lines synchronization for hybrid rice seed production,”

[36] N. Moustafa and J. Slay, “Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set),” in *2015 military communications and information systems conference (MilCIS)*, pp. 1–6, IEEE, 2015.