Journal of
Data Acquisition and Processing

ISSN: 1004-9037
https://sjcjycl.cn/
DOI: 10.5281/zenodo.7766130

DESIGN OF A NOVEL ENSEMBLE INTRUSION DETECTION FRAMEWORK USING THE CICIDS 2017 DATASET

A.Prashanthi
Research Scholar, Department of CSE, Osmania University,Hyd.
aindala.prashanthi@gmail.com

Dr. R. Ravinder Reddy
Associate Professor, Dept. of CSE, Chaitanya Bharathi Institute of Technology, TS, India
rravinderreddy_cse@cbit.ac.in

Abstract— Complexity and diversity of today's cyber assaults make it challenging to create a multi-attack categorization intrusion detection system. Intrusion Detection Systems need efficient classification to counteract hackers' advanced strategies. A single classifier can't properly detect several sorts of attacks, which is another problem. We suggested an unique ensemble architecture named Leader Class and Confidence Decision Ensemble Technique (LC&CDET) that accurately detects threats. The recommended strategy for spotting attacks involves ranking the detection capabilities of different base classifiers. The voting technique utilises a majority of the classifiers, regardless of whether the algorithm can detect the assault. It chooses the best ML model from three advanced methods (XGBoost, LightGBM, and CatBoost) for each attack category. Class leader models with confidence values are used to assess cyberattack detection. The proposed LC&CDE successfully detects intrusions using publicly accessible CICIDS2017 dataset with an accuracy and F1-score of 99.813 and 99.811%.

Keywords— Intrusion Detection System, CAN Bus, LightGBM, XGBoost, Ensemble Learning CICIDS2017;

I. INTRODUCTION

Cybercrime and other risks to computer and internet security are on the rise as the digital world develops. As a result, classic Intrusion Detection Systems (IDS) are rapidly becoming defunct. Previous IDS-based security solutions depended on signatures concepts that had already been specified [1], making them unable to spot newly generated abnormalities and attack variations [2]. The main issue was that the signature database wasn't being updated and expanded quickly enough to keep up with the rapidly changing nature of threats [2]. With the increasing complexity of modern assaults, researchers are constantly developing new methods for anomaly-based threat detection and protection [3]. In order to equip systems with robust intrusion detection tactics for the future, researchers are utilizing cutting-edge methods based on machine learning (ML) and deep learning (DL). However, not all modern-day threats can be detected using a single machine learning approach.

Numerous studies have been suggested in this area, with authors claiming that their IDS systems can accurately classify attacks even when presented with just historical data. The success of an Intrusion Detection model in the real world, however, hinges on how well it can spot sophisticated attacks as they happen in real time. Most of the research in this field is conducted using outdated data sets [4]. In Table 1 are listed the most frequently employed datasets [5] in IDS studies. Attack detection using models built on historical data is ineffective with today's traffic.

Machine learning is used to spot invaders. Random Forest, Support Vector Machine, Decision Trees, and k-Nearest Neighbor (kNN). Complex IDS demands can't be covered by a single machine learning algorithm [7], [4], [8], [9]. IDS attacks have gotten more diversified and sophisticated as traffic has increased [6]. Every ML algorithm has pros and cons. Some algorithms may be effective against some attacks, but not others. When ML models are utilised in IDS systems, cyber-attack prediction performance might vary widely. In this research, we present Leader-Class and Confidence-Decision Ensemble Technique (LC&CDET), which incorporates three gradient-boosting ML algorithms: XGBoost [10], LightGBM [11], and CatBoost [12]. LC&CDET maximizes model performance by identifying which base ML model has the most confidence for each class.

The primary contributions of this study are as follows:

1) It recommends a new ensemble framework called LC&CDET for efficient intrusion detection by combining class leader and confidence decision procedures with gradient-boosting ML algorithms.

2) The framework is evaluated with CICIDS2017 datasets, which represent IVN and external network data, and public security datasets.

3) It evaluates the given model in relation to other, more advanced techniques.

The following is how the paper is organized.

a. Section II presents an overview of recent research on intrusion detection using machine learning and ensemble models.

b. Part III describes the proposed LC&CDET framework.

c. Part IV presents and discusses the findings of the experiments.

d. The conclusion is the last portion of the paper.

II. RELATED WORK

Deep learning was used by W. Wang [1] to efficiently and automatically extract essential feature representations for use in detection. They provide a strong stacked contractive auto encoder (SCAE) approach for unsupervised feature extraction. With a high degree of resilience, the SCAE approach enables the automatic learning of superior low-dimensional properties from unprocessed network information. The studies show that the proposed SCAE+SVM methodology outperforms three current state-of-the-art approaches on the KDD 99 and NSL-KDD intrusion detection assessment datasets.

L. Qi [2] recommends utilising MDS AD. It's a combination of PCA, isolation forest, and locality-sensitive hashing (LSH). MDS AD has the following characteristics. 1) The new LSH can handle data from several angles. Second, MDS AD has a great degree of accuracy in detecting anomalies in experimental groups. Finally, the principal component analysis may be used to minimise the dimension of correlations between numerous attributes. Finally, MDS

AD is a streaming approach that can continually analyse data and update models while consuming little memory and processing power. Several studies are planned and carried out utilising the UNSW-NB15 dataset to validate the effectiveness of MDS AD. MDS AD outperforms the best current alternatives in tests, as proved.

R. Conde Camillo da Silva [3] used the CSIC 2010 dataset for both training and evaluation in his work. J48, Naive Bayes, OneR, Random Forest, and IBM Watson LGBM algorithms were compared. The measurements used were T-rate, accuracy, recall, and f-measure. The Watson tool's algorithm (LGBM) fared the best across the board when compared against other algorithms in the literature.

A two-stage pipeline was proposed by P. Barnard [4] for effective network intrusion detection. As a starting point, we employ an XGBoost model for supervised intrusion detection while developing model justifications with the SHapley Additive exPlanation (SHAP) framework. Using these explanations in the second stage teaches an auto-encoder to distinguish between known and unknown attacks. While the overall performance of our approach is comparable to many cutting-edge attempts in the cybersecurity literature, trials on the NSL-KDD dataset show that it can consistently identify new threats discovered during testing.

R. Zhao [5] used the CFS-DE technique to choose the best feature subset to minimize the size of the features. After that, a weighted Stacking method is presented, which raises the levels of base classifiers with excellent training results and lowers those with poor training results to enhance classification performance. As a result, the model improves classification efficiency while also generating higher accuracy. The NSL-KDD and CSE-CIC-IDS2018 data sets were used for all experiments throughout this study.

L. Wang [6] developed an efficient method for filtering only a few examples of exceptionally long round-trip durations from Internet packets. Experiments corroborate our findings that our proposed SSI detection method has a detection efficiency of more than 85.7% when used over the internet.

Z. Wu [7] proposes a dynamic ensemble evolutionary algorithms for intrusion detection approach (DEIL-RVM) and implements a dynamically adaptable ensemble intrusion detection model. The faulty ensemble model base component is eliminated and substituted with an alternate one that utilizes the fresh overall misclassification probability weighting factor (OMPW) produced from an incremental set or data chunk in this procedure. They are able to create a solid balance between accuracy and robustness by using the RVM's high feature space as the foundation.

S. Subbiah [8] proposes the Boruta feature extraction with grid search random forest (BFS-GSRF) approach in this study as a unique framework for IDS that can assist address these issues. BFS-efficiency GSRFs are compared to other ML approaches such as linear discrimination analysis (LDA) and regression tree and classification methods (CART), among others. The recommended work was carried out and assessed using data from of the Knowledge on Discoveries dataset there at Network Security Laboratory (NSL-KDD). The experimental results show that the suggested model BFS-GSRF detects attacks more accurately than LDA, CART, as well as other existing algorithms, with such an accuracy of 99%.

M. A. Siddiqi [9] proposed an image-processing-based NIDS architecture based on best practises. The framework includes an image improvement and modification mechanism with an improved feature selection flow. In order to maximise efficiency, the proposed framework first reduces the number of features. Data that did not previously have any visual display is then transformed into visuals. To show the effectiveness of the suggested architecture, a comparison with numerous recent researches on vision processing systems for network intrusion detection is done.

The primary focus of this study is X. Zhou's [10], who created a Graph neural net (GNN)-based intrusion detection in IoT systems, as well as an unique hierarchical adversarial attack (HAA) generation approach is suggested for implementing the level-aware black-box adversarial assault approach. Using the publicly accessible data set UNSW-SOSR2019, the proposed HAA generating approach is compared against three reference techniques. When compared to new state GNN algorithms for NIDS in IoT environments, the results demonstrate that it can lower their classification accuracy by more than 30%.

By comparing features from of the UNSWNB15 and Bot-IoT data-sets based upon flow and Transmission Control Protocol, M. Zeeshan et alproposal .'s of a Protocol Driven Deep Intrusion Detection (PB-DID) framework was made in this work (TCP). To appropriately classify regular, DoS, and DDoS traffic, we address problems such imbalanced and over-fitting. The deep learning (DL) technique allowed us to increase the classification accuracy to 96.3%.

In order to identify assaults without any prior information, G. Pu [12] introduced an unsupervised anomaly detection method that utilizes Sub-Space Clustering (SSC) and A Class Support Vector Machine (OCSVM). Another well NSL-KDD dataset is used to assess the suggested method. The experimental findings show that our strategy outperforms several of the currently used methods.

J. Yang [13] proposed an image-processing-based NIDS architecture based on best practises. The model integrates an image improvement and modification mechanism with an improved feature selection flow. A deep-learning classifier is used to help in the detection of anomalies in the pictures. The suggested technique is tested using three distinct intrusion detection benchmark datasets. To show the effectiveness of the suggested architecture, a comparison with numerous recent research on vision processing systems for network intrusion detection is done.

Y. Xie [14], proposed hybrid method that considers the anomaly degree of the entire provenance graph as well as the anomaly degree of a single provenance path. Additionally, it encodes objects in the rule database that are duplicates, and it filters out noise that does not carry any incursion information. Effectiveness and efficiency have been demonstrated experimentally across a wide range of practical applications.

III. PROPOSED FRAMEWORK

A. An Overview of the System

This study seeks to create an ensemble IDS architecture capable of identifying network threats on a consistent basis. Figure 1 depicts the two main components of the proposed system: model training and model prediction.

The CICIDS-2017 dataset is used to train three cutting-edge ML algorithms, XGBoost, LightGBM, and CatBoost, during model training. To identify assaults, model prediction employs class leader models and prediction confidences. This section outlines the algorithm.

B. Base Machine Learning Models

XGBoost, LightGBM, and CatBoost are all popular gradient boosting frameworks that can be used for various machine learning tasks, including anomaly detection. These frameworks are known for their speed, scalability, and accuracy, making them suitable for handling large datasets and complex feature sets.

In addition to these frameworks, there is a technique called Leader Class and Confidence Decision Ensemble (LC-CDE) that can be used for anomaly detection. This technique involves building multiple models, where each model is trained on a subset of the data, and then combining the results to make a final decision.

To implement this technique for anomaly detection, we first split our data into training and validation sets. Then, we train multiple models using XGBoost, LightGBM, and CatBoost, where each model is trained on a different subset of the training data. Once the models are trained, we use them to make predictions on the validation set and compute the anomaly score for each data point.

The anomaly score can be computed using a variety of methods, such as the Mahalanobis distance or the isolation forest algorithm. Once we have the anomaly scores for each data point, you can combine them using the LC-CDET technique to make a final decision on whether a data point is anomalous or not.

The LC-CDET technique involves assigning a leader class to each model based on its performance on the validation set. The leader class is the model that performs the best on the validation set. Then, for each data point, we can compute the confidence of each model's prediction and assign a confidence score based on its distance from the leader class prediction. The confidence scores can be combined using a weighted average to make the final decision.

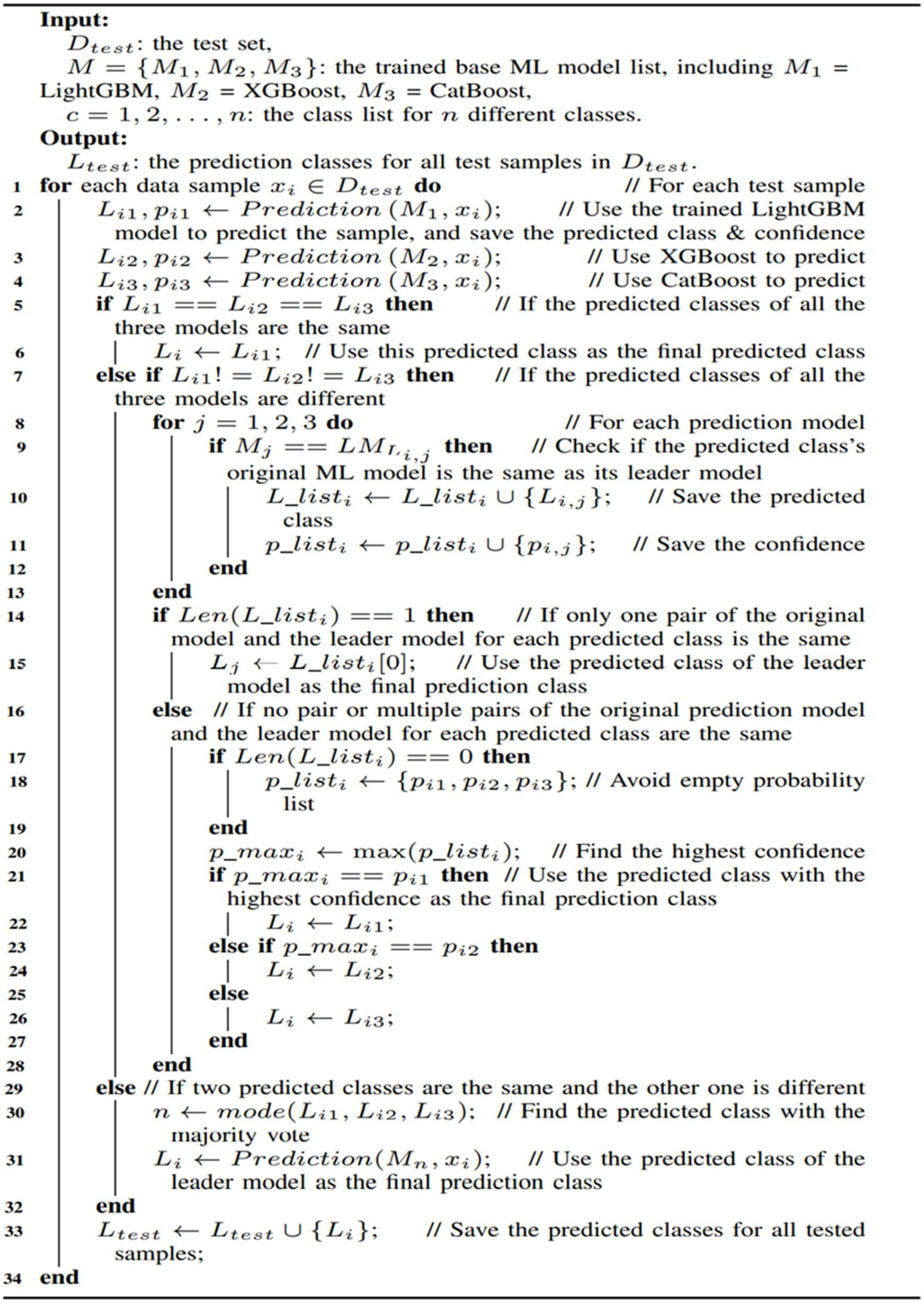
Overall, using XGBoost, LightGBM, and CatBoost with the LC-CDE technique can be an effective approach for anomaly detection, as it combines the strengths of multiple models and can handle complex datasets and feature sets. However, it's important to carefully tune the hyperparameters of the models and the LC-CDE technique to achieve the best performance.

C. LC&CDE: Proposed Ensemble Algorithm

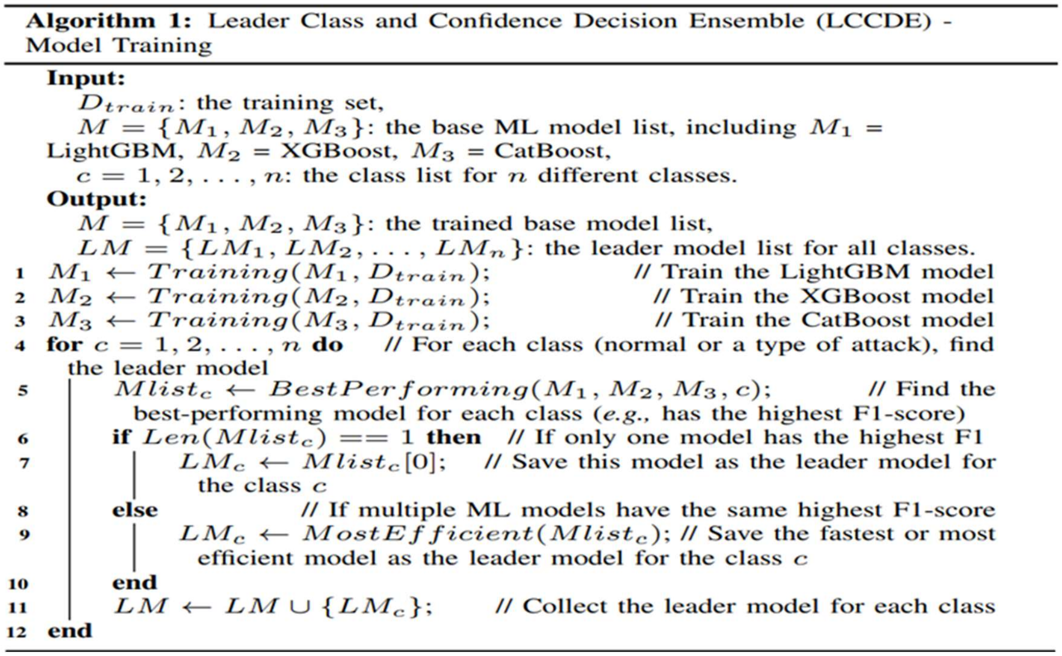
Anomaly detection using ensemble techniques can be done using XGBoost, LightGBM, and CatBoost. The Leader Class and Confidence Decision Ensemble technique can be used to improve the performance of the algorithm. Here's an outline of the algorithm:

1. Data preprocessing: The first step is to preprocess the data. This includes data cleaning, data normalization, and feature selection.
2. Ensemble training: Ensemble training involves training multiple models using different algorithms such as XGBoost, LightGBM, and CatBoost. Each model is trained on a different subset of the data using different hyperparameters.
3. Model selection: Once the models are trained, they are evaluated on a validation set. The best-performing models are selected for the final ensemble.
4. Leader class selection: In the leader class selection step, the best-performing model is selected as the leader. This model is used as the primary classifier in the final ensemble.
5. Confidence threshold selection: The confidence threshold is the threshold at which the model classifies a sample as an anomaly. The confidence threshold is selected based on the validation set's performance.
6. Confidence decision ensemble: In the confidence decision ensemble step, the other models in the ensemble are used to provide a confidence score for each sample. If a sample is classified as an anomaly by the leader model and has a low confidence score, it is rejected.
7. Anomaly detection: Finally, the ensemble of models is used to classify samples as anomalous or non-anomalous. Samples that are classified as anomalous by the leader model and have a high confidence score are considered to be true anomalies.

This algorithm can be further optimized by tuning hyperparameters using cross-validation and selecting the optimal threshold using the ROC curve.



```
Input:
    $D_{test}$: the test set,
    $M = \{M_1, M_2, M_3\}$: the trained base ML model list, including $M_1$ =
LightGBM, $M_2$ = XGBoost, $M_3$ = CatBoost,
    $c = 1, 2, \ldots, n$: the class list for $n$ different classes.
Output:
    $L_{test}$: the prediction classes for all test samples in $D_{test}$.
for each data sample $x_i \in D_{test}$ do                    // For each test sample
    $L_{i1}, p_{i1} \leftarrow Prediction\ (M_1, x_i)$;    // Use the trained LightGBM
      model to predict the sample, and save the predicted class & confidence
    $L_{i2}, p_{i2} \leftarrow Prediction\ (M_2, x_i)$;    // Use XGBoost to predict
    $L_{i3}, p_{i3} \leftarrow Prediction\ (M_3, x_i)$;    // Use CatBoost to predict
    if $L_{i1} == L_{i2} == L_{i3}$ then    // If the predicted classes of all the
      three models are the same
        $L_i \leftarrow L_{i1}$;  // Use this predicted class as the final predicted class
    else if $L_{i1}! = L_{i2}! = L_{i3}$ then    // If the predicted classes of all the
      three models are different
        for $j = 1, 2, 3$ do                    // For each prediction model
            if $M_j == LM_{L_{i,j}}$ then    // Check if the predicted class's
              original ML model is the same as its leader model
                $L\_list_i \leftarrow L\_list_i \cup \{L_{i,j}\}$;    // Save the predicted
                  class
                $p\_list_i \leftarrow p\_list_i \cup \{p_{i,j}\}$;    // Save the confidence
            end
        end
        if $Len(L\_list_i) == 1$ then    // If only one pair of the original
          model and the leader model for each predicted class is the same
            $L_j \leftarrow L\_list_i[0]$;    // Use the predicted class of the leader
              model as the final prediction class
        else  // If no pair or multiple pairs of the original prediction model
          and the leader model for each predicted class are the same
            if $Len(L\_list_i) == 0$ then
                $p\_list_i \leftarrow \{p_{i1}, p_{i2}, p_{i3}\}$; // Avoid empty probability
                  list
            end
            $p\_max_i \leftarrow \max(p\_list_i)$;    // Find the highest confidence
            if $p\_max_i == p_{i1}$ then  // Use the predicted class with the
              highest confidence as the final prediction class
                $L_i \leftarrow L_{i1}$;
            else if $p\_max_i == p_{i2}$ then
                $L_i \leftarrow L_{i2}$;
            else
                $L_i \leftarrow L_{i3}$;
            end
        end
    else // If two predicted classes are the same and the other one is different
        $n \leftarrow mode(L_{i1}, L_{i2}, L_{i3})$;  // Find the predicted class with the
          majority vote
        $L_i \leftarrow Prediction(M_n, x_i)$;    // Use the predicted class of the
          leader model as the final prediction class
    end
    $L_{test} \leftarrow L_{test} \cup \{L_i\}$;    // Save the predicted classes for all tested
      samples;
end
```



Algorithm 1: Leader Class and Confidence Decision Ensemble (LCCDE) - Model Training

```
Input:
    $D_{train}$: the training set,
    $M = \{M_1, M_2, M_3\}$: the base ML model list, including $M_1$ =
  LightGBM, $M_2$ = XGBoost, $M_3$ = CatBoost,
    $c = 1, 2, \ldots, n$: the class list for $n$ different classes.
Output:
    $M = \{M_1, M_2, M_3\}$: the trained base model list,
    $LM = \{LM_1, LM_2, \ldots, LM_n\}$: the leader model list for all classes.
$M_1 \leftarrow Training(M_1, D_{train})$;                // Train the LightGBM model
$M_2 \leftarrow Training(M_2, D_{train})$;                // Train the XGBoost model
$M_3 \leftarrow Training(M_3, D_{train})$;                // Train the CatBoost model
for $c = 1, 2, \ldots, n$ do    // For each class (normal or a type of attack), find
  the leader model
    $Mlist_c \leftarrow BestPerforming(M_1, M_2, M_3, c)$;         // Find the
      best-performing model for each class (e.g., has the highest F1-score)
    if $Len(Mlist_c) == 1$ then   // If only one model has the highest F1
        $LM_c \leftarrow Mlist_c[0]$;   // Save this model as the leader model for
          the class $c$
    else               // If multiple ML models have the same highest F1-score
        $LM_c \leftarrow MostEfficient(Mlist_c)$; // Save the fastest or most
          efficient model as the leader model for the class $c$
    end
    $LM \leftarrow LM \cup \{LM_c\}$;      // Collect the leader model for each class
end
```

IV. PERFORMANCE EVALUATION

A. Experimental Setup

Python Scikit-learn, Xgboost, Lightgbm, and Catboost packages were used to develop the recommended IDS. The experiments carried out on Google colab. Anomaly detection using ensemble techniques can be done using XGBoost, LightGBM, and CatBoost. The Leader Class and Confidence Decision Ensemble technique can be used to improve the performance of the algorithm. Here's an outline of the algorithm:

```
. # Load the data
df = pd.read_csv('data.csv')

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(df.drop('target', axis=1), df['target'],
test_size=0.2, random_state=42)

# Scale the features
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# Train the XGBoost model
xgb_model = xgb.XGBClassifier()
xgb_model.fit(X_train, y_train)

# Train the LightGBM model
lgb_model = lgb.LGBMClassifier()
```

```
lgb_model.fit(X_train, y_train)

# Train the CatBoost model
cb_model = cb.CatBoostClassifier()
cb_model.fit(X_train, y_train)

# Use the ensemble technique to combine the results of the three models
def ensemble_predict(X):
    xgb_pred = xgb_model.predict_proba(X)
    lgb_pred = lgb_model.predict_proba(X)
    cb_pred = cb_model.predict_proba(X)
    preds = np.vstack([xgb_pred[:,1], lgb_pred[:,1], cb_pred[:,1]])
    confs = np.vstack([xgb_pred[:,1] > 0.5, lgb_pred[:,1] > 0.5, cb_pred[:,1] > 0.5])
    leader = np.argmax(np.sum(confs, axis=0))
    return preds[leader,:]

# Use the ensemble model to predict anomalies in the testing dataset
y_pred = ensemble_predict(X_test)
y_pred_binary = y_pred > 0.5
tn, fp, fn, tp = confusion_matrix(y_test, y_pred_binary).ravel()
precision = tp / (tp + fp)
recall = tp / (tp + fn)
f1_score = 2 * (precision * recall) / (precision + recall)
print('Precision: {:.4f}'.format(precision))
print('Recall: {:.4f}'.format(recall))
print('F1 Score: {:.4f}'.format(f1_score))
```

B. Results and Discussion from the Experiments

Tables I and II compare the LC&CDE model against LightGBM, XGBoost, and CatBoost on the CICIDS2017 datasets. Table I shows the F1-scores for detecting each type of attack in the two datasets. Different basic ML models' F1-scores for assault detection vary. Using the CICIDS2017 dataset, LightGBM achieves the highest F1-score among the three base learners for normal samples, DoS, sniffing, webattacks, botnets, and infiltration attacks, while XGBoost beats LightGBM for brute-force attack detection.

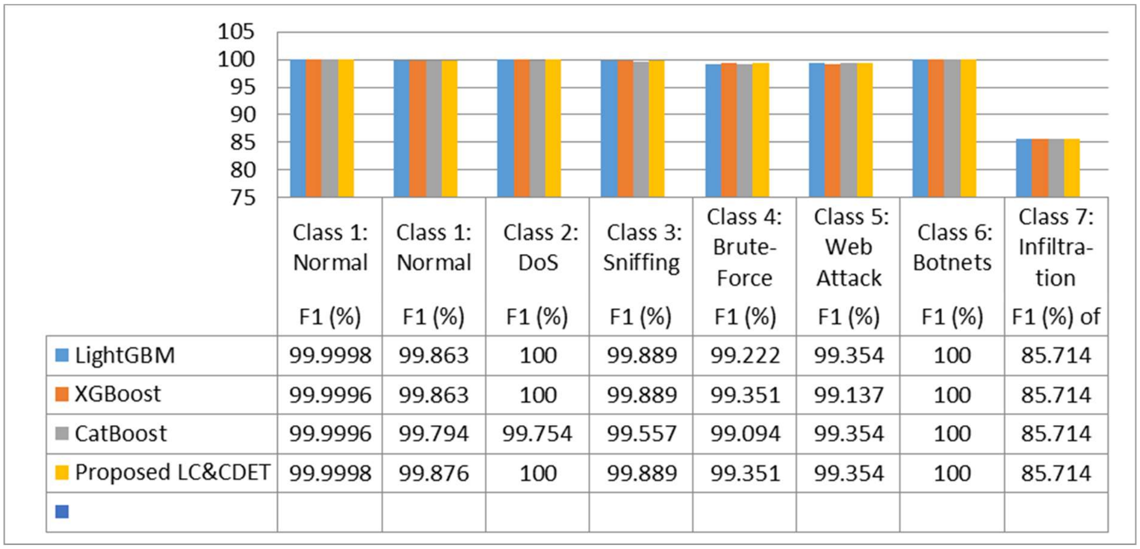
Table I shows that the proposed LC&CDE ensemble model outperforms state-of-the-art approaches in all classes. The recommended model has the highest F1-scores among the four ML models employed on the two datasets (Tables II and III). The LC&CDE model boosted the CICIDS2017 F1-score from 99.792% to 99.811%. Using the top-performing base models for each class as the LC&CDE ensemble model's starting point provides apparent advantages.

Tables II and III compare suggested method's performance on two datasets to state-of-the-art techniques [14]. The recommended LC&CDE model enhances state-of-the-art techniques by 0.09% and 0.11% on CICIDS2017 datasets. The LC&CDE model takes a bit longer to run than the three gradient-boosting models, but it's significantly faster than other base learners line

KNN and SVM provided in the literature. The recommended ensemble model is built with low-complexity ML models that can run in parallel and benefit from GPU support. The recommended model has the lowest execution time and highest F1-scores across both benchmark datasets.

TABLE I

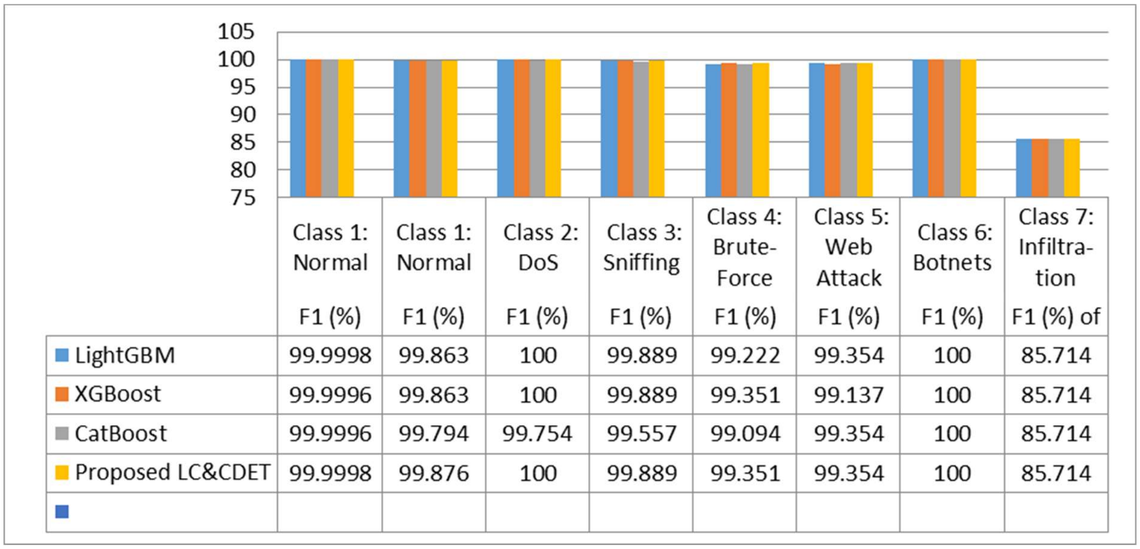
MODEL PERFORMANCE COMPARISON FOR EACH CLASS IN DATASET



	Class 1: Normal F1 (%)	Class 1: Normal F1 (%)	Class 2: DoS F1 (%)	Class 3: Sniffing F1 (%)	Class 4: Brute-Force F1 (%)	Class 5: Web Attack F1 (%)	Class 6: Botnets F1 (%)	Class 7: Infiltra-tion F1 (%) of
🟦 LightGBM	99.9998	99.863	100	99.889	99.222	99.354	100	85.714
🟧 XGBoost	99.9996	99.863	100	99.889	99.351	99.137	100	85.714
⬜ CatBoost	99.9996	99.794	99.754	99.557	99.094	99.354	100	85.714
🟨 Proposed LC&CDET	99.9998	99.876	100	99.889	99.351	99.354	100	85.714
🟦								

TABLE II

PERFORMANCE EVALUATION OF MODELS ON CICIDS2017



	Class 1: Normal F1 (%)	Class 1: Normal F1 (%)	Class 2: DoS F1 (%)	Class 3: Sniffing F1 (%)	Class 4: Brute-Force F1 (%)	Class 5: Web Attack F1 (%)	Class 6: Botnets F1 (%)	Class 7: Infiltra-tion F1 (%) of
🟦 LightGBM	99.9998	99.863	100	99.889	99.222	99.354	100	85.714
🟧 XGBoost	99.9996	99.863	100	99.889	99.351	99.137	100	85.714
⬜ CatBoost	99.9996	99.794	99.754	99.557	99.094	99.354	100	85.714
🟨 Proposed LC&CDET	99.9998	99.876	100	99.889	99.351	99.354	100	85.714
🟦								

V. CONCLUSION

Methods of machine learning (ML) are used to detect attacks and improve security. The effectiveness of ML models relies on the type of assault analysed. In this study, we offer LC&CDET for top network attack performance on all attack types. It assesses which ML models better detect certain risks and employs them as "leading class models" in order to create a credible ensemble model. To decrease the likelihood of outcomes and develop prediction classes, data on the predictor's degree of confidence is employed. Three gradient boosting machine learning algorithms—XGBoost, LightGBM, and CatBoost—are used in the proposed LC&CDET ensemble model. The suggested IDS framework can get 99.811% F1- scores on the CICIDS2017 datasets, according to experiments. The suggested model outperforms previous ML methods in terms of F1-scores for recognising all attacks. This demonstrates the viability of the leader-class-based Technique.

REFERENCES

[1] W. Wang, X. Du, D. Shan, R. Qin and N. Wang, "Cloud Intrusion Detection Method Based on Stacked Contractive Auto-Encoder and Support Vector Machine," in IEEE Transactions on Cloud Computing, vol. 10, no. 3, pp. 1634-1646, 1 July-Sept. 2022, doi: 10.1109/TCC.2020.3001017.

[2] L. Qi, Y. Yang, X. Zhou, W. Rafique and J. Ma, "Fast Anomaly Identification Based on Multiaspect Data Streams for Intelligent Intrusion Detection Toward Secure Industry 4.0," in IEEE Transactions on Industrial Informatics, vol. 18, no. 9, pp. 6503-6511, Sept. 2022, doi: 10.1109/TII.2021.3139363.

[3] R. Conde Camillo da Silva, M. P. Oliveira Camargo, M. Sanches Quessada, A. Claiton Lopes, J. Diassala Monteiro Ernesto and K. A. Pontara da Costa, "An Intrusion Detection System for Web- Based Attacks Using IBM Watson," in IEEE Latin America Transactions, vol. 20, no. 2, pp. 191- 197, Feb. 2022, doi: 10.1109/TLA.2022.9661457.

[4] P. Barnard, N. Marchetti and L. A. DaSilva, "Robust Network Intrusion Detection Through Explainable Artificial Intelligence (XAI)," in IEEE Networking Letters, vol. 4, no. 3, pp. 167- 171, Sept. 2022, doi: 10.1109/LNET.2022.3186589.

[5] R. Zhao, Y. Mu, L. Zou and X. Wen, "A Hybrid Intrusion Detection System Based on Feature Selection and Weighted Stacking Classifier," in IEEE Access, vol. 10, pp. 71414-71426, 2022, doi: 10.1109/ACCESS.2022.3186975.

[6] L. Wang, J. Yang, M. Workman and P. Wan, "Effective algorithms to detect stepping-stone intrusion by removing outliers of packet RTTs," in Tsinghua Science and Technology, vol. 27, no. 2, pp. 432-442, April 2022, doi: 10.26599/TST.2021.9010041.

[7] Z. Wu, P. Gao, L. Cui and J. Chen, "An Incremental Learning Method Based on Dynamic Ensemble RVM for Intrusion Detection," in IEEE Transactions on Network and Service Management, vol. 19, no. 1, pp. 671-685, March 2022, doi: 10.1109/TNSM.2021.3102388.

[8] S. Subbiah, K. S. M. Anbananthen, S. Thangaraj, S. Kannan and D. Chelliah, "Intrusion detection technique in wireless sensor network using grid search random forest with Boruta feature selection algorithm," in Journal of Communications and Networks, vol. 24, no. 2, pp. 264-273, April 2022, doi: 10.23919/JCN.2022.000002.

[9] M. A. Siddiqi and W. Pak, "Tier-Based Optimization for Synthesized Network Intrusion Detection System," in IEEE Access, vol. 10, pp. 108530-108544, 2022, doi: 10.1109/ACCESS.2022.3213937.

[10] X. Zhou, W. Liang, W. Li, K. Yan, S. Shimizu and K. I. -K. Wang, "Hierarchical Adversarial Attacks Against Graph-Neural-Network-Based IoT Network Intrusion Detection System," in IEEE Internet of Things Journal, vol. 9, no. 12, pp. 9310-9319, 15 June15, 2022, doi: 10.1109/JIOT.2021.3130434.

[11] M. Zeeshan et al., "Protocol-Based Deep Intrusion Detection for DoS and DDoS Attacks Using UNSW-NB15 and Bot-IoT Data-Sets," in IEEE Access, vol. 10, pp. 2269-2283, 2022, doi: 10.1109/ACCESS.2021.3137201.

[12] G. Pu, L. Wang, J. Shen and F. Dong, "A hybrid unsupervised clustering-based anomaly detection method," in Tsinghua Science and Technology, vol. 26, no. 2, pp. 146-153, April 2021, doi: 10.26599/TST.2019.9010051.

[13] J. Yang, X. Chen, S. Chen, X. Jiang and X. Tan, "Conditional Variational Auto-Encoder and Extreme Value Theory Aided Two-Stage Learning Approach for Intelligent Fine-Grained Known/Unknown Intrusion Detection," in IEEE Transactions on Information Forensics and Security, vol. 16, pp. 3538-3553, 2021, doi: 10.1109/TIFS.2021.3083422.

[14] Y. Xie, D. Feng, Y. Hu, Y. Li, S. Sample and D. Long, "Pagoda: A Hybrid Approach to Enable Efficient Real-Time Provenance Based Intrusion Detection in Big Data Environments," in IEEE Transactions on Dependable and Secure Computing, vol. 17, no. 6, pp. 1283-1296, 1 Nov.-Dec. 2020, doi: 10.1109/TDSC.2018.2867595.