Data Science Project Report

LCCDE: A Decision-Based Ensemble Framework for Intrusion Detection in the Internet of Vehicles

Johi Chawala | BSCS-V “B” | CMS: 023-21-0029
Ikram | BSCS-V “B” | CMS: 023-21-0253

Abstract—Modern vehicles, including autonomous vehicles and connected vehicles, have adopted an increasing variety of functionalities through connections and communications with other vehicles, smart devices, and infrastructures. However, the growing connectivity of the Internet of Vehicles (IoV) also increases the vulnerabilities to network attacks. To protect IoV systems against cyber threats, Intrusion Detection Systems (IDSs) that can identify malicious cyber-attacks have been developed using Machine Learning (ML) approaches. To accurately detect various types of attacks in IoV networks, we propose a novel ensemble IDS framework named Leader Class and Confidence Decision Ensemble (LCCDE). It is constructed by determining the best-performing ML model among three advanced ML algorithms (XGBoost, LightGBM, and CatBoost) for every class or type of attack. The class leader models with their prediction confidence values are then utilized to make accurate decisions regarding the detection of various types of cyber-attacks. Experiments on two public IoV security datasets (Car-Hacking and CICIDS2017 datasets) demonstrate the effectiveness of the proposed LCCDE for intrusion detection on both intra-vehicle and external networks.

Index Terms—Intrusion Detection System, Internet of Vehicles, CAN Bus, LightGBM, XGBoost, Ensemble Learning

➢ Introduction:

With the fast development of the Internet of Things (IoT) and the Internet of Vehicles (IoV) technologies, network-controlled automobiles, such as Autonomous Vehicles (AVs) and Connected Vehicles (CVs), have begun replacing conventional vehicles. IoV systems typically consist of intra-vehicle networks (IVNs) and external networks. In IVNs, the Controller Area Network (CAN) bus is the core infrastructure enabling communication between Electronic Control Units (ECUs) to implement various functionalities [2]. External vehicular networks, on the other hand, utilize Vehicle-To-Everything (V2X) technology to enable the connection between smart cars and other IoV entities, such as roadside units, infrastructures, and smart devices.

Due to the expanded network attack surfaces of IoV systems, the growing connectivity of vehicular networks has resulted in numerous security threats. In addition, there are no authentication or encryption mechanisms involved in the processing of CAN packets owing to their short length. In the absence of basic security mechanisms, cybercriminals can insert malicious messages into IVNs and execute a variety of attacks, such as Denial of Service (DoS), fuzzy, and spoofing attacks. On the other hand, the emergence of connectivity between connected cars and external networks has made these vehicles vulnerable to a number of conventional cyber-attacks.

Figure 1 depicts the IoV attack scenarios, including IVN and external network attacks. Intrusion Detection Systems (IDSs) have been developed as promising solutions for detecting intrusions and defending Internet of Vehicles (IoV) systems and smart automobiles from cyber-attacks.

The potential deployment of IDSs in IoV systems is also shown in Fig. 1. To protect IVNs, IDSs can be placed on top of the CAN-bus to identify malicious CAN messages. IDSs can also be incorporated into the gateways to detect malicious packets coming from external networks. Due to the advancement of Machine Learning (ML) methods, ML-driven IDSs in IoV applications have recently drawn the interest of researchers and automotive manufacturers. Through network traffic data analytics, ML approaches are commonly employed to construct classifier-based IDSs that can differentiate between benign network events and various cyber-attacks. To apply ML models to IDS systems, it is common to observe that the prediction performance of different ML models varies significantly for different types of cyber-attack detection. Thus, a novel ensemble approach named Leader Class and Confidence Decision Ensemble (LCCDE)1 is proposed in the paper to obtain optimal performance on all types of attack detection by integrating three advanced gradient-boosting ML algorithms, including Extreme Gradient Boosting (XGBoost), Light Gradient Boosting Machine (LightGBM) and Categorical Boosting (CatBoost).

LCCDE aims to achieve optimal model performance by identifying the best-performing base ML model with the highest prediction confidence for each class.

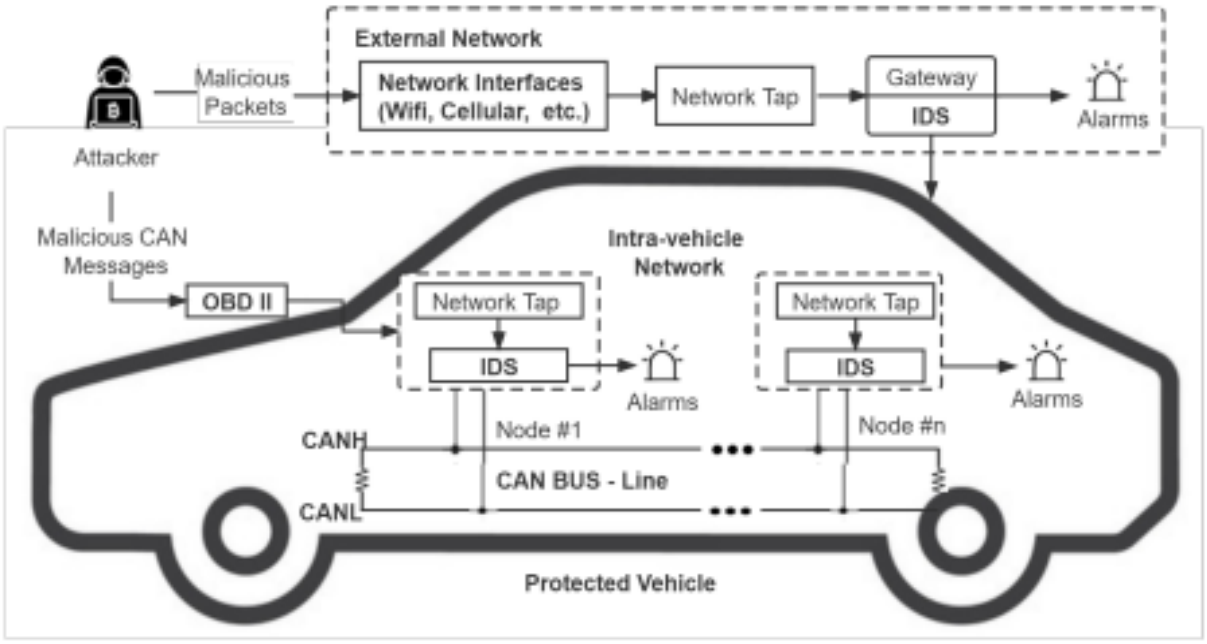


Fig. 1. The IDS-protected vehicle architecture.

➢ Problem Statement:

The growing connectivity in IoV systems has led to an increased vulnerability to network attacks, including Denial of Service (DoS), spoofing, and other cyber threats. Existing ML-based IDSs may not perform optimally for all types of attacks. The challenge is to develop an ensemble IDS framework that can effectively detect and differentiate between different types of attacks in IoV networks.

On the other hand, the emergence of connectivity between connected cars and external networks has made these vehicles vulnerable to a number of conventional cyber-attacks. **Figure 1** depicts the IoV attack scenarios, including IVN and external network attacks. Intrusion Detection Systems (IDSs) have been developed as promising solutions for detecting intrusions and defending Internet of Vehicles (IoV) systems and smart automobiles from cyber-attacks. The potential deployment of IDSs in IoV systems is also shown in Fig. 1. To protect IVNs, IDSs can be placed on top of the CAN-bus to identify malicious CAN messages. IDSs can also be incorporated into the gateways to detect malicious packets coming from external networks

➢ Proposed Methodology:

A. System Overview

The purpose of this work is to develop an ensemble IDS framework that can effectively detect various types of attacks on both IVN and external vehicular networks. Figure 2 demonstrates the overall framework of the proposed system, consisting of two phases: model training and model prediction. At the model training stage, three advanced ML algorithms, XGBoost, LightGBM, and CatBoost, are trained on the IoV traffic dataset to obtain the leader models for all classes/types of attacks. At the model prediction stage, the class leader models and their prediction confidences are used to accurately detect attacks. The algorithm details are provided in this section.

Figure 2: The overview of the LCCCDE IDS model.

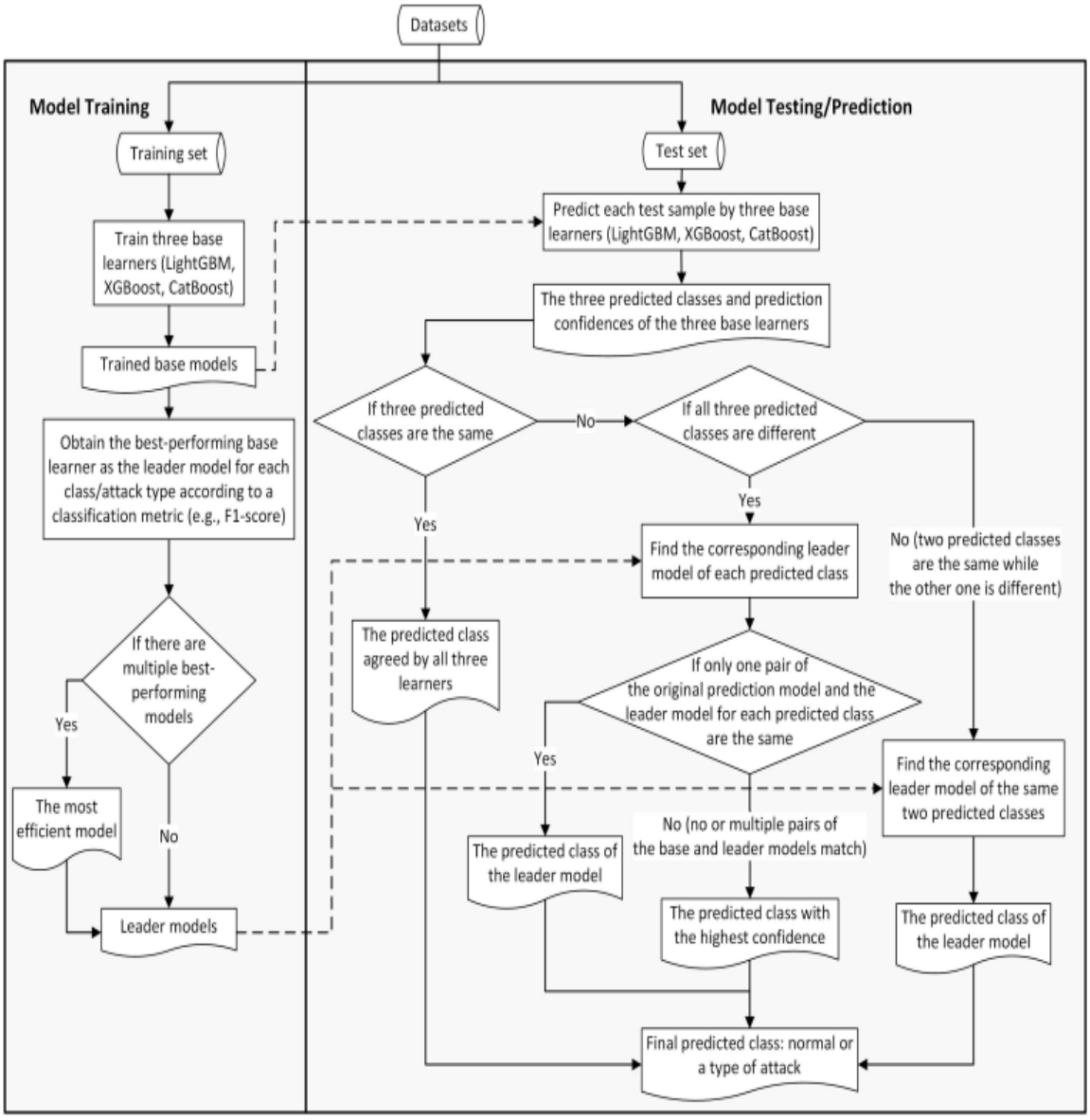


Fig. 2. The framework of the proposed MTH-IDS.

B. Base Machine Learning Models

A **decision tree (DT)** is a basic ML algorithm that uses a tree structure to make decisions based on the divide-and-conquer technique. In DTs, the decision nodes represent the decision tests, while the leaves indicate the result classes. **Gradient Boosting Decision Tree** (GBDT) is an iterative DT algorithm that constructs multiple DTs and aggregates their prediction outputs. To improve the performance of basic GBDTs, three advanced gradient-boosting algorithms**, XGBoost, LightGBM, and CatBoost,** have been developed and widely used in many applications. These three gradient-boosting algorithms are used in the proposed system to build the LCCDE ensemble framework.

XGBoost is a popular gradient-boosting DT algorithm designed for the speed and performance improvement of GBDTs. XGBoost uses a regularization term and a Second-Order Taylor Approximation for the summation of the squared errors to minimize the loss function and reduce over-fitting. XGBoost has a low computational complexity of O(Kd||x||log n), where K is the number of trees, d is the maximum tree depth, ||x|| is the number and non-missing samples, and n is the data size. Additionally, XGBoost supports parallel execution to save model learning time. LightGBM is a fast and robust ensemble ML model constructed by multiple DTs.

LightGBM's key advantage over other ML methods is its capacity to efficiently handle large-scale and high-dimensional data. Gradient-based OneSide Sampling (GOSS) and Exclusive Feature Bundling (EFB) are the two core strategies of LightGBM. GOSS is a down-sampling method that only preserves data samples with large gradients and randomly discards small gradient samples to accelerate model training and reduce memory consumption. EFB is a feature engineering method that regroups mutually exclusive features into bundles as single features to minimize feature size and improve model training efficiency. By employing GOSS and EFB, the data size can be reduced significantly without the loss of critical information. The time and space complexity of LightGBM has also been reduced to O(N'F'), where N' is the reduced number of samples after using GOSS, and F' is the bundled number of features after employing EFB.

CatBoost is another advanced gradient-boosting algorithm designed to process categorical features more effectively. CatBoost, in comparison to existing gradient-boosting models, includes three significant model enhancement components: symmetric trees, ordered boosting, and native feature support. In symmetric trees, leaves are split under the same condition as in prior trees, and the pair of feature splits with the lowest loss is applied to all nodes. Using symmetric trees can improve model prediction speed and reduce over-fitting. Ordered boosting is a permutation-driven technique that prevents overfitting on small datasets by training a model on a subset while calculating residuals on another subset. CatBoost's native feature support indicates that it can directly process all types of features, such as numerical, textual, and categorical features, without the need for extra pre-processing. CatBoost is an ensemble model with low computational complexity of O(SN), where S is the number of permutations of the subsets and N is the number of base DT models.

The primary reasons for selecting XGBoost, LightGBM, and CatBoost

1. These three ML models are all robust ensemble models that have had great success in a variety of data analytics applications.
2. These three ML models can automatically generate feature importance scores and select features during their training process, which saves time and resources by avoiding the need for extra feature engineering.
3. These three ML models are fast models with relatively low computational complexity. Additionally, they all support parallelization and Graphics Processing Unit (GPU) execution, which can further improve model learning speed.

4. These three ML models include randomness in their model construction process, enabling people to develop a robust ensemble model with high diversity and generalizability

C. LCCDE: Proposed Ensemble Algorithm

The performance of different ML models often varies on different types of attack detection tasks. For example, when applying multiple ML models on the same network traffic dataset, an ML model performs the best for detecting the first type of attack (e.g., DoS attacks), while another ML model may outperform other models for detecting the second type of attack (e.g., sniffing attacks). Therefore, this work aims to propose an ensemble framework that can achieve optimal model performance for the detection of every type of attack. Ensemble learning is a technique that combines multiple base L models to improve learning performance and generalizability. The proposed ensemble model is constructed using XGBoost, LightGBM, and CatBoost, three advanced gradient-boosting ML methods introduced in Section III-B. Figure 2 demonstrates the process of the proposed LCCDE framework in two phases: model training and model prediction. The detailed procedures of the training and prediction phases are also described in Algorithms 1 & 2, respectively

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

Algorithm 2: Leader Class and Confidence Decision Ensemble (LCCDE) - Model Prediction

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

The computational complexity of LCCDE is O(NCK), where N is the data size, C is the number of classes, K is the complexity of base models. Thus, its complexity mainly depends on the complexity of all base ML models. In the proposed framework, three fast gradient-boosting ML algorithms are used to achieve low overall complexity. These three algorithms can be replaced by other ML algorithms using the same generic LCCDE strategy according to specific tasks. LCCDE is designed to address the difficult samples that cannot be correctly predicted by individual ML models. By using LCCDE, the final ensemble model can achieve optimal performance for detecting every type of attack.

➢ Dataset Discussion:

The evaluation of the LCCDE framework is conducted on two public IoV security datasets: the Car-Hacking dataset and the CICIDS2017 dataset, representing intra-vehicle and external vehicular network data, respectively. These datasets contain various types of attacks, and the proposed model's performance is compared with other state-of-the-art methods.

To develop the proposed IDS for both IVNs and external vehicular networks, two datasets are used in this work. The first dataset is the Car-Hacking dataset that represents intra-vehicle data, as it is generated by transmitting CAN packets into the CAN-bus of a real vehicle. The CAN identifier (ID) and 8-bit data field of CAN packets (DATA[0]-DATA[7]) are the main features of the dataset.

The Car-Hacking dataset involves four main types of attacks: DoS, fuzzy, gear spoofing, and Revolutions Per Minute (RPM) spoofing attacks. The second dataset used is the CICIDS2017 dataset that represents external network data, as it is a state-of-the-art network security dataset that includes the most updated attack patterns. According to the dataset analysis, the attack patterns in the CICIDS2017 dataset can be summarized into five main types of attacks: DoS attacks, port-scan attacks, brute-force attacks, web-attacks, and botnets.

➢ Major Outcomes:

A. Experimental Setup

To develop the proposed IDS, the models were implemented using Scikit-learn, Xgboost, Lightgbm, and Catboost libraries in Python.

The proposed LCCDE framework is evaluated on two public benchmark IoV network security datasets, Car-Hacking and CICIDS2017 datasets, representing the IVN and external network data, respectively. The Car-Hacking dataset is created by transmitting CAN messages into a real vehicle's CAN bus. It has nine features (i.e., CAN ID and the eight bits of the CAN message data field) and four types of attacks (i.e., DoS, fuzzy, gear spoofing, and Revolutions Per Minute (RPM) spoofing attacks). The CICIDS2017 dataset is a state-of-the-art general cyber-security dataset including the most updated types of attacks (i.e., DoS, sniffing, brute-force, web-attacks, botnets, and infiltration attacks). To evaluate the proposed LCCDE model, five-fold cross-validation is used in the training process to select leader class models, and 80%/20% hold-out validation is then used in the testing process to evaluate the model on the unseen test set. As network traffic data is often highly imbalanced and contains only a small proportion of attack samples, four performance measures, including accuracy, precision, recall, and F1 scores, are

utilized to evaluate the model performance. The execution time, including the model training and test time, is used to evaluate the efficiency of the model.

B. Experimental Results and Discussion

The experimental results of evaluating the three base ML models (LightGBM, XGBoost, CatBoost) and the proposed LCCDE model on the Car-Hacking and CICIDS2017 datasets are shown in Tables I – III.

Table I illustrates the performance of the four models for detecting every type of attack in the two datasets based on their F1-scores. It is noticeable that the F1-scores of different base ML models vary for different types of attack detection. For example, on the CICIDS2017 dataset, LightGBM achieves the highest F1-score among the three base learners for detecting normal samples, DoS, sniffing, webattacks, botnets, and infiltration attacks, while XGBoost outperforms LightGBM for the brute-force attack detection. As shown in Table I, the proposed LCCDE ensemble model can achieve the highest F1 score for every class.

Thus, as shown in Tables II and III, the overall F1 scores of the proposed model are also the highest among the four utilized ML models on the two datasets. The proposed LCCDE model achieves a near-perfect F1-score on the Car-Hacking dataset (99.9997%) and improved its F1-score from 99.792% to 99.811% on the CICIDS2017 dataset. This demonstrates the benefits of identifying the best-performing base models for each class to construct the LCCDE ensemble model.

Tables II and III also compare the performance of the proposed technique with existing state-of-the-art methods on the two datasets. The proposed LCCDE model outperforms other methods by at least 0.09% and 0.11% F1-score improvements on the Car-Hacking and CICIDS2017 datasets, respectively. As an ensemble approach, the proposed LCCDE model has a longer execution time than the other three base gradient-boosting models, but it is still faster than many other ML algorithms, such as K-Nearest Neighbors (KNN) and Support Vector Machine (SVM). This is because the proposed ensemble model is built using low-complexity ML models with parallel execution and GPU support. To summarize, the proposed model can achieve the highest F1 scores among the compared methods with relatively low execution time on the two benchmark datasets.

TABLE I
MODEL PERFORMANCE COMPARISON FOR EACH CLASS IN THE TWO DATASETS

Method	Car-Hacking Dataset					CICIDS2017 Dataset						
	F1 (%) of Class 1: Normal	F1 (%) of Class 2: DoS	F1 (%) of Class 3: Fuzzy	F1 (%) of Class 4: Gear Spoofing	F1 (%) of Class 5: RPM Spoofing	F1 (%) of Class 1: Normal	F1 (%) of Class 2: DoS	F1 (%) of Class 3: Sniffing	F1 (%) of Class 4: Brute-Force	F1 (%) of Class 5: Web Attack	F1 (%) of Class 6: Botnets	F1 (%) of Class 7: Infiltration
LightGBM [11]	**99.9998**	**100.0**	**99.995**	**100.0**	**100.0**	99.863	**100.0**	**99.889**	99.222	**99.354**	**100.0**	**85.714**
XGBoost [10]	99.9996	**100.0**	99.990	**100.0**	**100.0**	99.863	**100.0**	**99.889**	**99.351**	99.137	**100.0**	**85.714**
CatBoost [12]	99.9996	**100.0**	99.990	**100.0**	**100.0**	99.794	99.754	99.557	99.094	**99.354**	**100.0**	**85.714**
Proposed LCCDE	**99.9998**	**100.0**	**99.995**	**100.0**	**100.0**	**99.876**	**100.0**	**99.889**	**99.351**	**99.354**	**100.0**	**85.714**

TABLE II
PERFORMANCE EVALUATION OF MODELS ON CAR-HACKING DATASET

Method	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	Execution Time (s)
KNN [23]	97.4	96.3	98.2	93.4	195.6
SVM [23]	96.5	95.7	98.3	93.3	1345.3
LSTM-AE [24]	99.0	99.0	99.9	99.0	-
DCNN [16]	99.93	99.84	99.84	99.91	-
LightGBM [11]	99.9997	99.9997	99.9997	99.9997	10.7
XGBoost [10]	99.9994	99.9994	99.9994	99.9994	45.3
CatBoost [12]	99.9994	99.9994	99.9994	99.9994	88.6
Proposed LCCDE	**99.9997**	**99.9997**	**99.9997**	**99.9997**	185.1

TABLE III
PERFORMANCE EVALUATION OF MODELS ON CICIDS2017

Method	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	Execution Time (s)
KNN [14]	96.3	96.2	93.7	96.3	1558.3
RF [14]	98.82	98.8	99.955	98.8	135.1
DBN [19]	98.95	95.82	95.81	95.81	-
Stacking [18]	99.80	99.75	99.89	99.70	278.6
LightGBM [11]	99.794	99.795	99.794	99.792	14.3
XGBoost [10]	99.794	99.795	99.794	99.792	44.7
CatBoost [12]	99.683	99.684	99.683	99.680	73.7
Proposed LCCDE	**99.813**	**99.814**	**99.913**	**99.811**	168.9

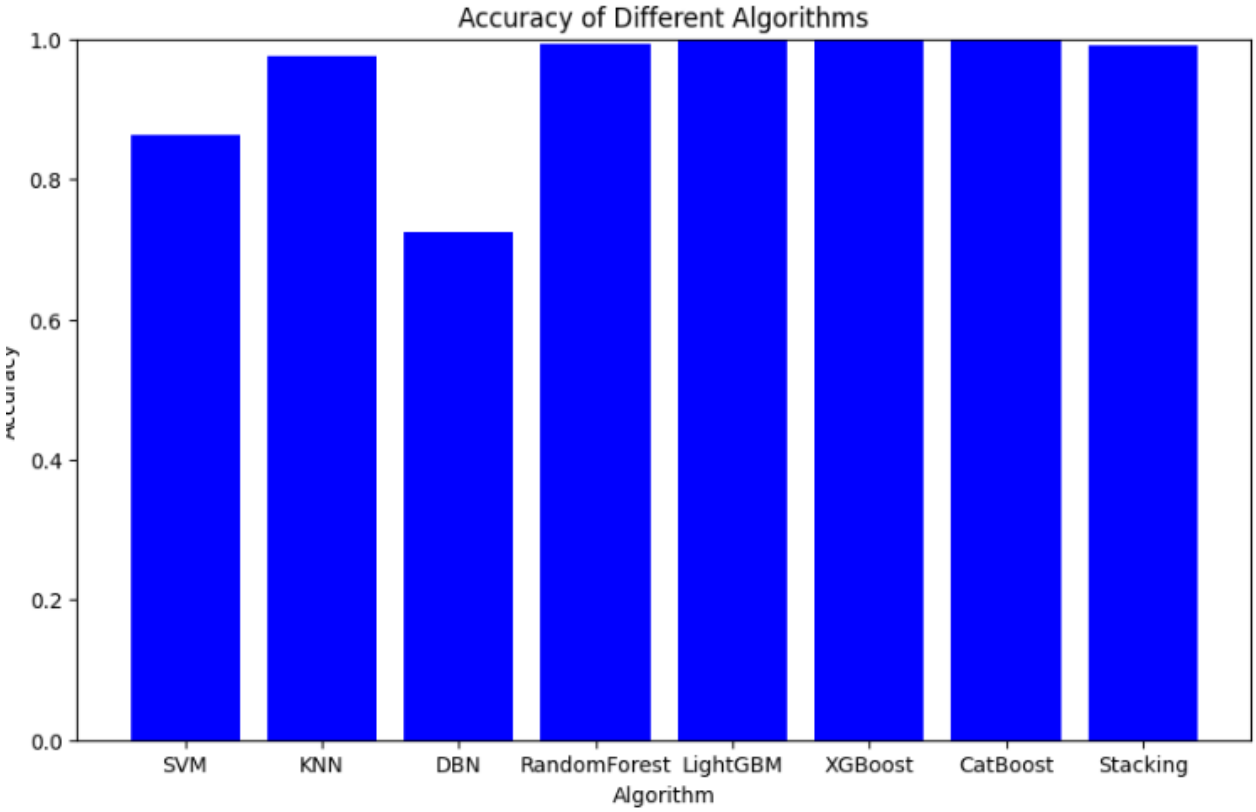
➢ Conclusion:

To enhance IoV security, Machine Learning (ML) algorithms have been used as promising solutions to detect various types of cyber-attacks. However, ML models often perform differently for different types of attack detection. To achieve optimal performance on all types of attack detection in IoV networks, a novel ensemble method, namely Leader Class and Confidence Decision Ensemble (LCCDE), is proposed in this paper.

It identifies the best-performing ML models for each type of attack detection as the leader class models to construct a robust ensemble model. Additionally, the prediction confidence information is utilized to help determine the final prediction classes.

Three advanced gradient-boosting ML algorithms, XGBoost, LightGBM, and CatBoost, are utilized to construct the proposed LCCDE ensemble model due to their high effectiveness and efficiency. Through the experiments, the proposed IDS framework achieves high F1- scores of 99.9997% and 99.811% on the Car-Hacking and CICIDS2017 datasets, representing intra-vehicle and external vehicular network data, respectively. Moreover, the proposed model's F1-scores are higher than other compared ML methods for detecting every type of attack. This illustrates the benefits of the proposed leader class-based strategy

➢ Project Implementation:

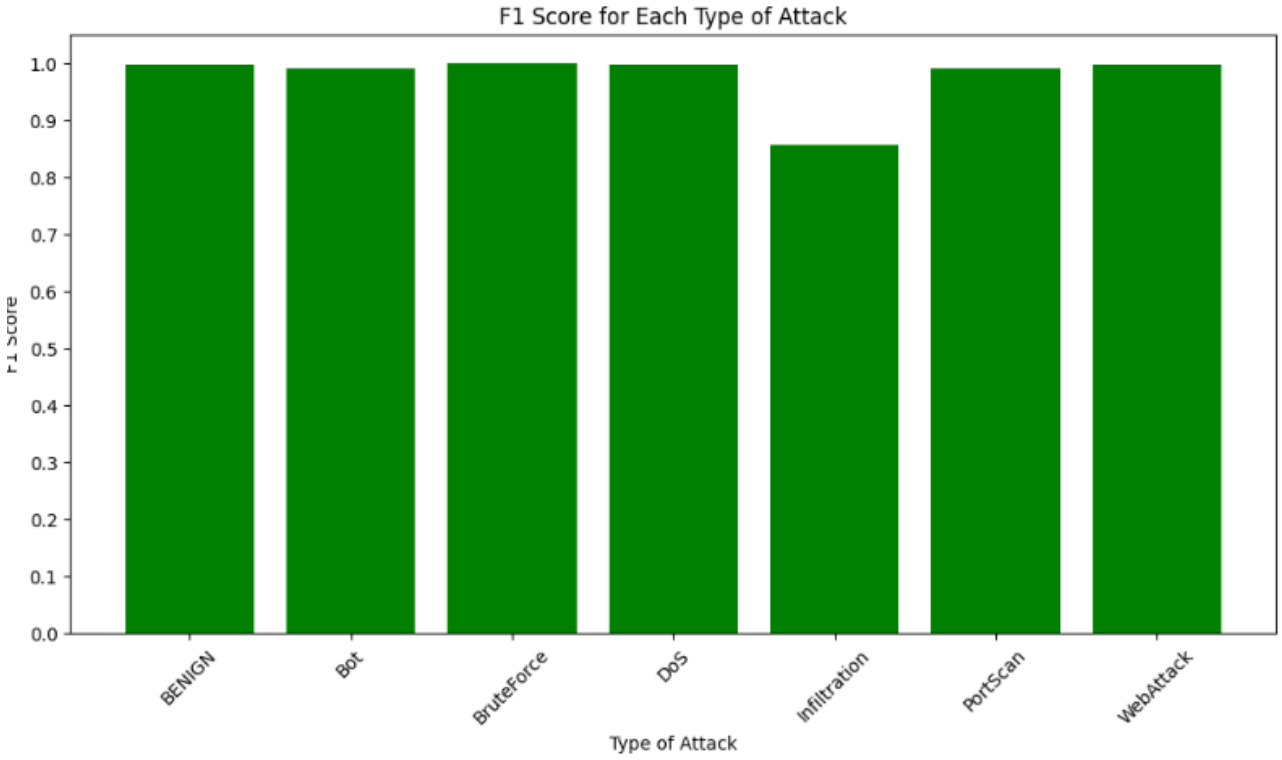



```
The best-performing algorithm is: LGBMClassifier
Accuracy: 0.9973880597014926
```

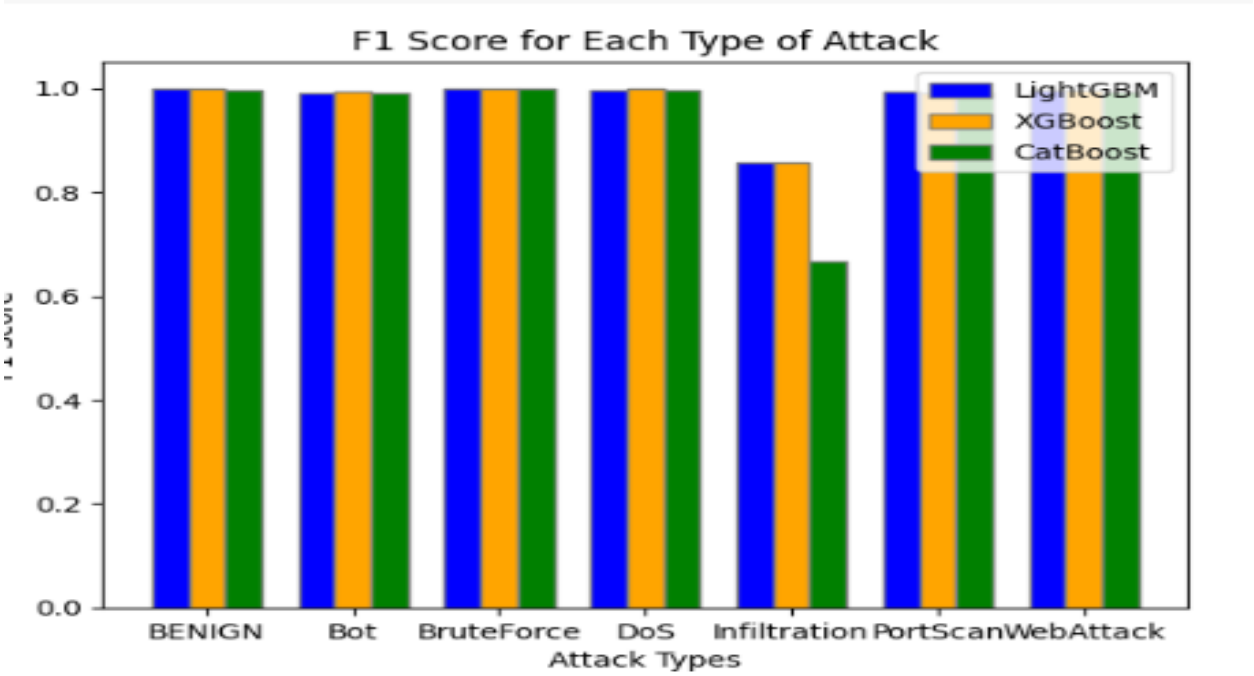
```
LGBMClassifier: Accuracy = 0.9973880597014926
XGBClassifier: Accuracy = 0.9968283582089552
CatBoostClassifier: Accuracy = 0.996455223880597
StackingClassifier: Accuracy = 0.9929104477611941
RandomForestClassifier: Accuracy = 0.9895522388059701
KNeighborsClassifier: Accuracy = 0.9759328358208955
MLPClassifier: Accuracy = 0.9406716417910448
SVC: Accuracy = 0.8623134328358208
```

```
# The performance of the proposed lCCDE model
print("Accuracy of LCCDE: "+ str(accuracy_score(yt, yp)))
print("Precision of LCCDE: "+ str(precision_score(yt,
                                                  yp, average='weighted')))
print("Recall of LCCDE: "+ str(recall_score(yt, yp, average='weighted')))
print("Average F1 of LCCDE: "+ str(f1_score(yt, yp, average='weighted')))
print("F1 of LCCDE for each type of attack: "+ str(f1_score(yt, yp, average=None)))
```

```
Accuracy of LCCDE: 0.9968283582089552
Precision of LCCDE: 0.9968351356243143
Recall of LCCDE: 0.9968283582089552
Average F1 of LCCDE: 0.9968013032165239
F1 of LCCDE for each type of attack: [0.99781421 0.99092088 1.         0.997543   0.85714286 0.99137931
 0.99778271]
```



Attack Type	F1 (LightGBM)	F1 (XGBoost)	F1 (CatBoost)
BENIGN	0.99794998	0.99808639	0.99726627
Bot	0.99094437	0.99222798	0.99094437
BruteForce	1.0	1.0	1.0
DoS	0.997543	0.99836601	0.99509804
Infiltration	0.85714286	0.85714286	0.66666667
PortScan	0.99354839	0.99137931	0.99137931
WebAttack	0.99778271	0.99778271	0.9944629



➢ Reference:

Dataset

CICIDS2017 dataset, a popular network traffic dataset for intrusion detection problems

- Publicly available at: https://www.unb.ca/cic/datasets/ids-2017.html

CAN-intrusion dataset, a benchmark network security dataset for intra-vehicle intrusion detection

- Publicly available at: https://ocslab.hksecurity.net/Datasets/CANsss-intrusion-dataset

[1] H. Bangui and B. Buhnova, "Recent Advances in Machine-Learning Driven Intrusion Detection in Transportation: Survey," Procedia Comput. Sci., vol. 184, pp. 877–886, 2021.

[2] O. Y. Al-Jarrah, C. Maple, M. Dianati, D. Oxtoby, and A. Mouzakitis, "Intrusion Detection Systems for Intra-Vehicle Networks: A Review," IEEE Access, vol. 7, pp. 21266–21289, 2019.

[3] L. Yang and A. Shami, "A Transfer Learning and Optimized CNN Based Intrusion Detection System for Internet of Vehicles," in 2022 IEEE Int. Conf. Commun. (ICC), 2022, pp. 1–6.

[4] K. Kim, J. S. Kim, S. Jeong, J.-H. Park, and H. K. Kim, "Cybersecurity for autonomous vehicles: Review of attacks and defense," Comput. Secur, vol. 103, p. 102150, 2021.

[5] L. Yang, A. Moubayed, and A. Shami, “MTH-IDS: A Multitiered Hybrid Intrusion Detection System for Internet of Vehicles,” IEEE Internet Things J., vol. 9, no. 1, pp. 616–632, 2022.

[6] J. Jiang, F. Liu, W. W. Y. Ng, Q. Tang, W. Wang, and Q.-V. Pham, “Dynamic Incremental Ensemble Fuzzy Classifier for Data Streams in Green Internet of Things,” IEEE Trans. Green Commun. Netw., pp. 1-14, 2022.

[7] M. Injadat, A. Moubayed, A. B. Nassif, and A. Shami, “Machine learning towards intelligent systems: applications, challenges, and opportunities,” Artif. Intell. Rev., 2021.

➢ Links:

https://github.com/JohiChawala/Intrusion-Detection-System-Using-Machine-Learning

https://colab.research.google.com/drive/1YQSstqtPqDPwxbjsZFI5RkznHGkGkCy2?usp=sharing