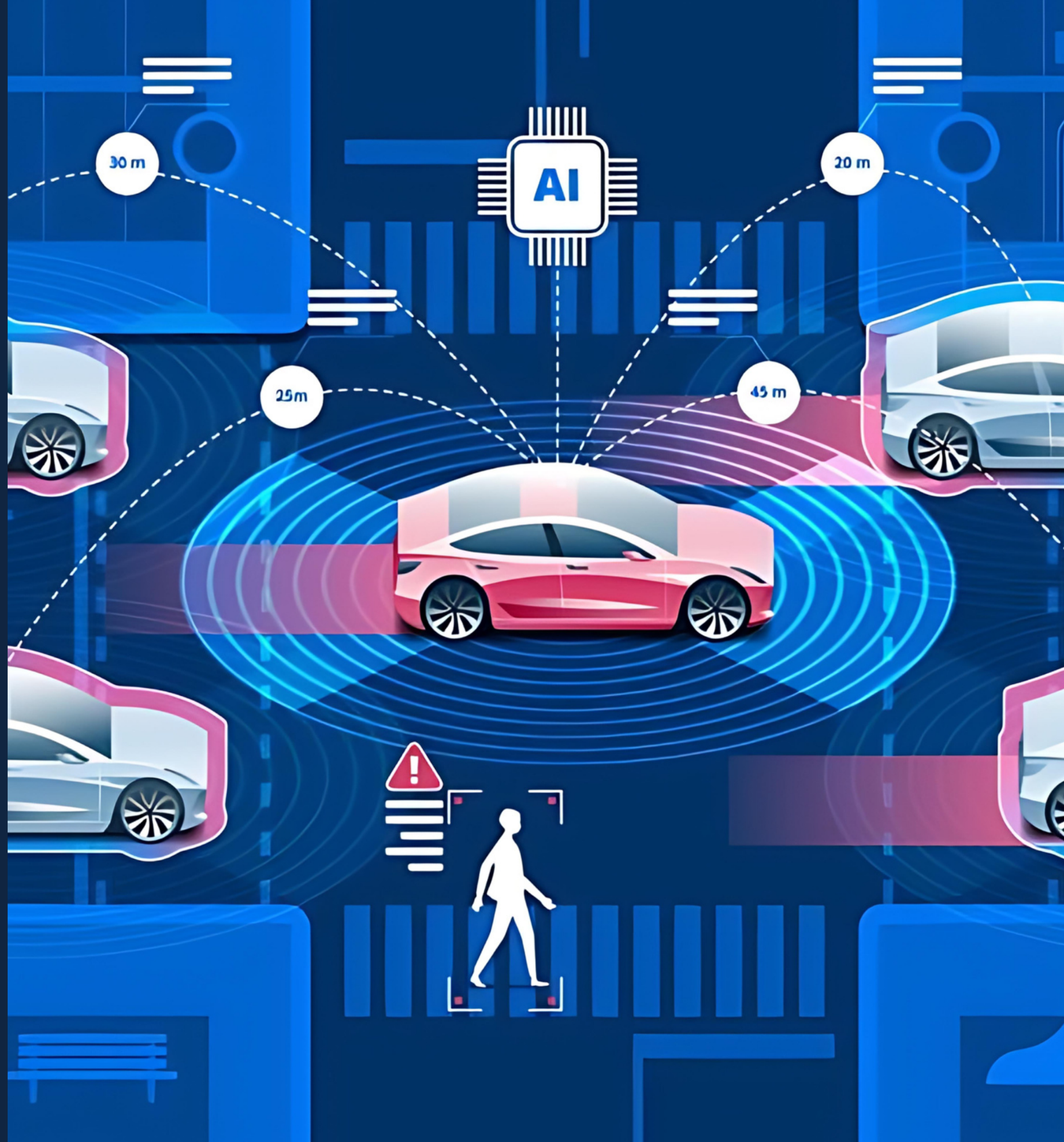
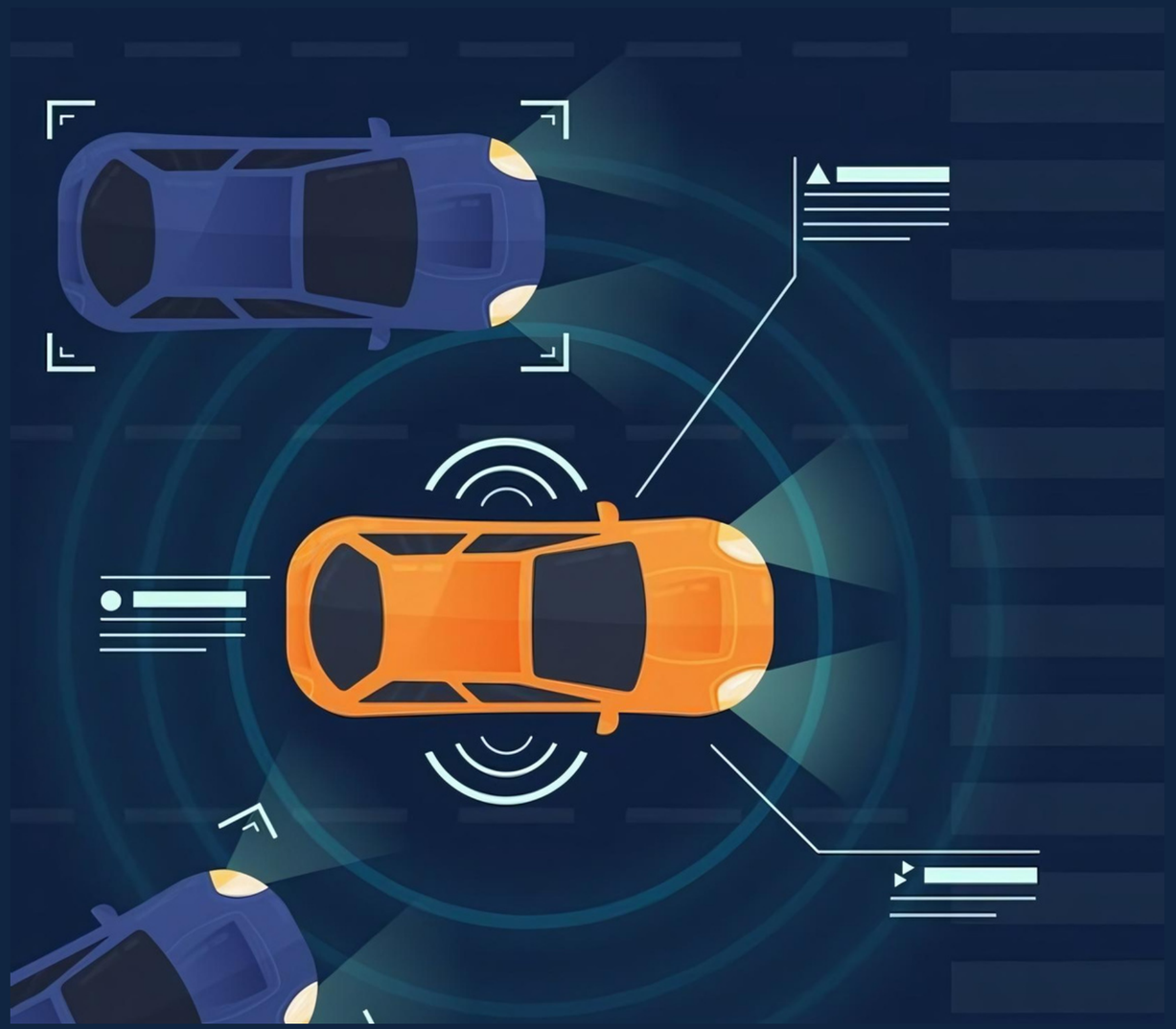
Catch before they breach !!!

INTRUSION DETECTION IN INTERNET OF VEHICLES



WHAT IS IOV ?



The Internet of Vehicles (IoV) seamlessly blends vehicles with internet connectivity and advanced communication technologies, ushering in a new era of network-controlled automobiles, including Autonomous Vehicles (AVs) and Connected Vehicles (CVs).

IoV enables the Intelligent Transportation System (ITS) to communicate with other vehicles, humans, infrastructure, the Internet, and the cloud.

IoV architecture comprises intra-vehicle networks (IVNs) and external vehicular networks, each playing a pivotal role in enabling advanced functionalities and ensuring seamless communication.

COMMUNICATION

VEHICLE TO VEHICLE

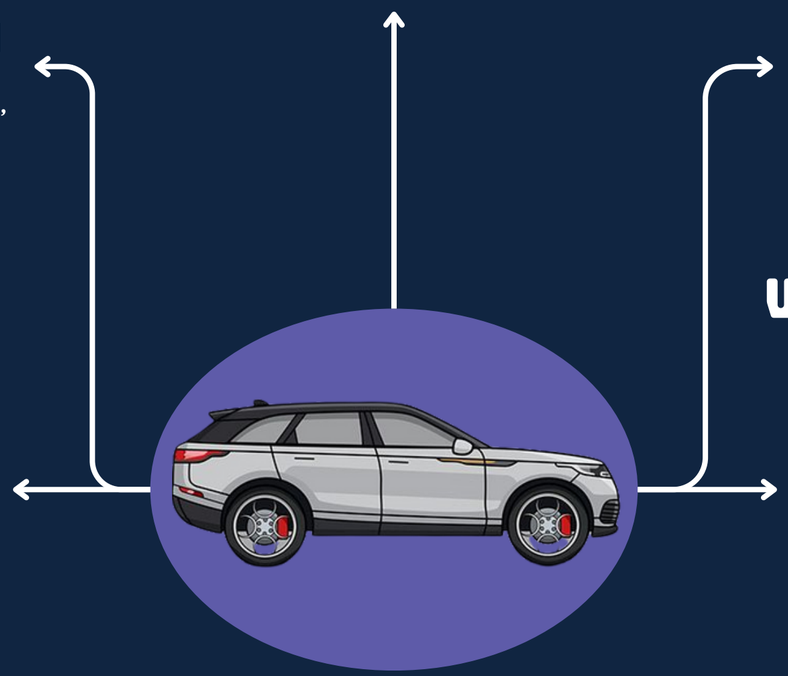
Direct communication between vehicles enables sharing of real-time data such as location, speed, and direction.

VEHICLE TO SENSOR

V2S facilitates data exchange with sensors for environmental monitoring, parking assistance, and obstacle detection.

VEHICLE TO INFRASTRUCTURE

Vehicles interact with infrastructure like traffic lights and sensors, receiving updates on traffic conditions and signal timing.



VEHICLE TO CLOUD

V2C platforms enable data storage, processing, and aggregation for analytics, predictive maintenance, and personalized services.

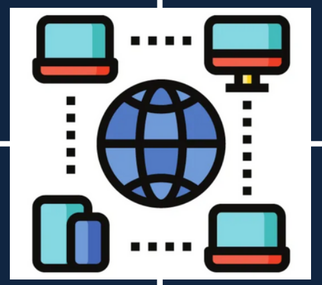
VEHICLE TO PERSONAL DEVICE

Vehicles interact with personal devices like smartphones, enabling remote control and personalized notifications and infotainment services.

WHAT IS CAN ?

CAN stands for Controller Area Network protocol.

CAN protocol is a standard designed to allow the microcontroller and other devices to communicate with each other without any host computer.

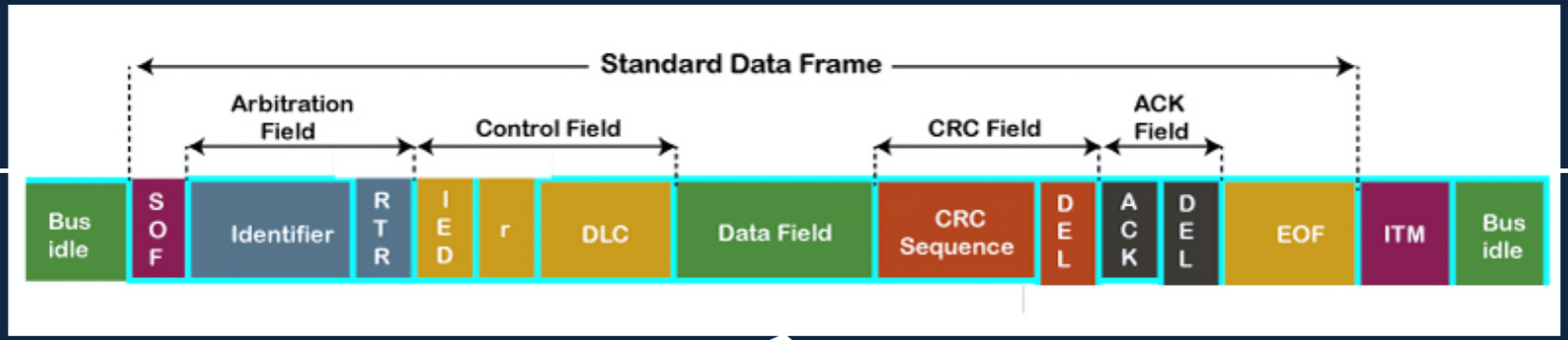


There is no need for node identification in the CAN network, so it becomes very easy to insert or delete it from the network.

Unique feature about this protcol is the broadcast typoe of bus. Here broadcast means that the information is transmitted to all the nodes.

The CAN is a message-based protocol, which means that message carries the message identifier, and based on the identifier, priority is decided.

WHAT IS CAN PACKET?



SOF
Start of frame

EOF
End of frame

Identifier
Sets the priority of the data frame

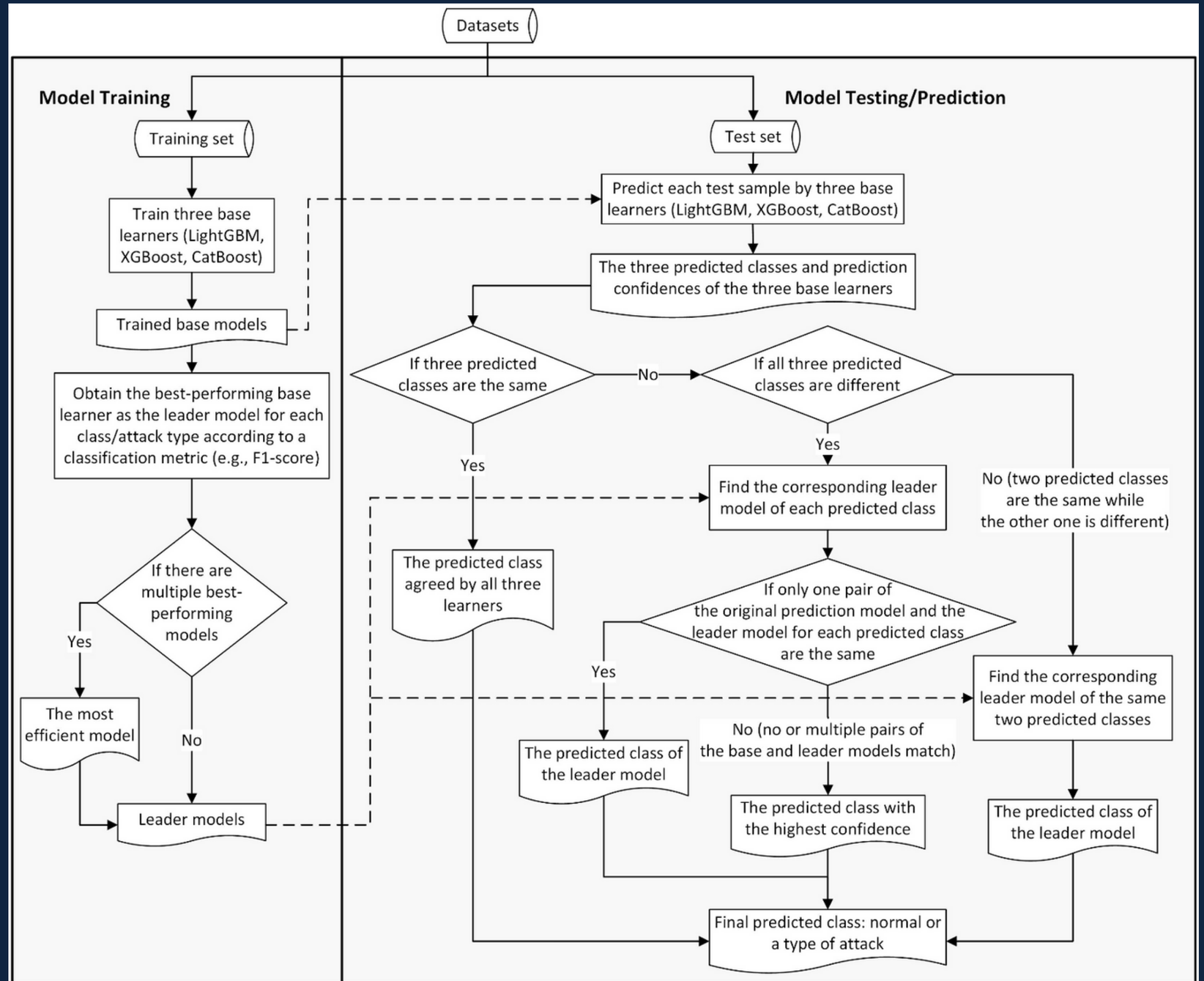
RTR
RTR stands for Remote Transmission Request, which defines the frame type, whether it is a data frame or a remote frame.

IDE
Stands for identifier extension..

DLC
Stands for Data Length Code, which defines the data length in a data field

ACK
In other protocols, a separate packet for an acknowledgment is sent after receiving all the packets, but in case of CAN protocol, no separate packet is sent for an acknowledgment.

OVERVIEW OF MODEL



BASIC PSEUDO CODE OF MODEL

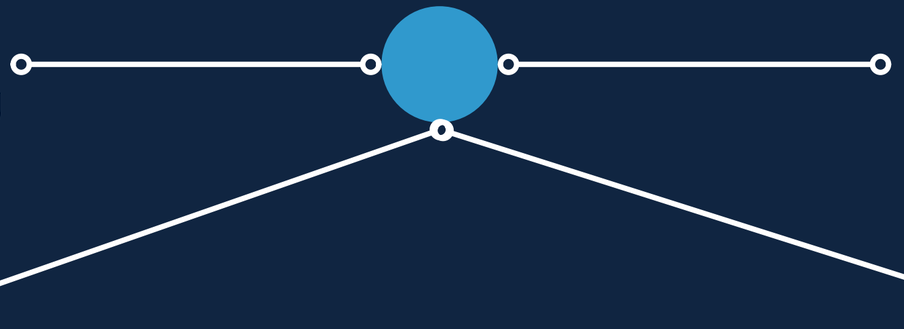
Input:
D_{train}: the training set,
$M = \{M_1, M_2, M_3\}$: the base ML model list, including M_1 = LightGBM, M_2 = XGBoost, M_3 = CatBoost,
$c = 1, 2, \ldots, n$: the class list for n different classes.
Output:
$M = \{M_1, M_2, M_3\}$: the trained base model list,
$LM = \{LM_1, LM_2, \ldots, LM_n\}$: 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

WHY LIGHTGBM, XGBOOST AND CATBOOST IS USED AS BASE LEARNER FOR THE MODEL?

These three ML models are all robust ensemble models that have had great success in a variety of data analytics applications

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



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.

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.

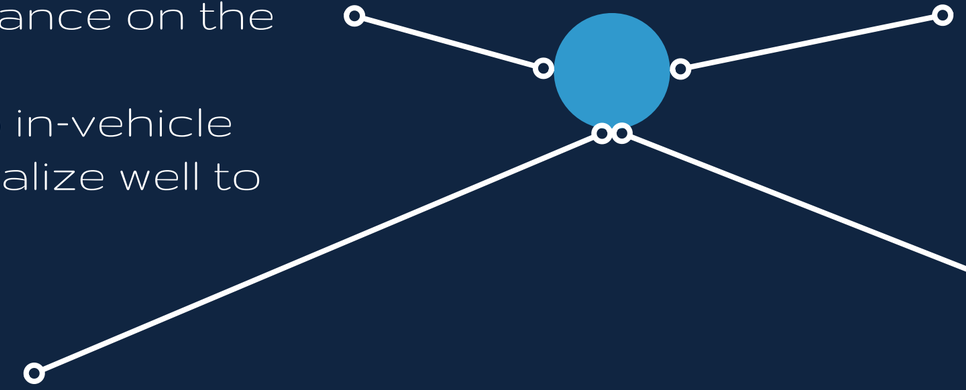
COMPARISON WITH OTHER RESEARCHES

Deep Convolutional Neural Network Model:

- Focus: Intrusion detection in in-vehicle networks.
- Strengths: High performance on the Car-Hacking dataset.
- Weaknesses: Specific to in-vehicle networks; may not generalize well to other IoT systems.

Ensemble Model with DNN, LSTM, and DBN:

- Focus: Network intrusion detection using deep learning models.
- Strengths: Utilizes deep learning models like DNN, LSTM, and DBN.
- Weaknesses: Complexity of deep learning models may require significant computational resources and training data.



Lightweight Deep Neural Network Model:

- Focus: IDS framework for IoT systems.
- Strengths: Utilizes PCA to reduce feature dimensionality and computational cost.
- Weaknesses: May sacrifice some accuracy for computational efficiency; may not perform as well on more complex datasets.

Stacking Ensemble Framework:

- Focus: Network intrusion detection in IoV systems using tree-based ML models.
- Strengths: High accuracy on CAN-Intrusion and CICIDS2017 datasets.
- Weaknesses: May not effectively leverage more advanced ML algorithms; limited to tree-based models.

PERFORMANCE OF OUR MODEL ON PUBLICALLY AVAILABLE DATASET

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**

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

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

THANK YOU