Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2024.0429000

Fed-CALiBER: Federated Lightweight BERT Intrusion Detection on CAN Bus Protocol in Autonomous Vehicle

HAMMAN A. BIMMO, BUDI RAHARDJO, (Member, IEEE)
School of Electrical Engineering and Informatics, Bandung Institute of Technology, Bandung 40132, Indonesia
Corresponding author: Hamman A. Bimmo (e-mail: 23223055@mahasiswa.itb.ac.id).

This work was supported in part by Kementerian Komunikasi dan Digital Republik Indonesia.

ABSTRACT Recent advancements in autonomous vehicle (AV) technology have highlighted critical cybersecurity vulnerabilities within In-Vehicle Networks (IVNs), particularly the Controller Area Network (CAN) bus. While numerous Intrusion Detection Systems (IDS) exist, significant gaps remain in addressing resource efficiency and the challenge of Non Independent and Identically Distributed (Non-IID) data in distributed vehicular environments. This study proposes Fed-CALiBER, a novel framework that synergistically combines a compact, pre-trained Lightweight BERT model with a Federated Learning (FL) architecture. By training collaboratively on distinct datasets assigned to Raspberry Pi edge clients, our approach preserves data privacy by keeping raw data localized and is explicitly designed to enhance generalization across Non-IID data distributions. With a reduced communication overhead by transmitting a small parameter footprint (approx. 13 MB) during federated updates, Fed-CALiBER minimizes network overhead during parameter aggregation. Our two-cycle experimental results demonstrate that the federated global model significantly outperforms standalone models in cross-dataset generalization—improving F1-scores on unseen datasets from as low as 71.39% to over 96.59%—and successfully adapts to shifting data distributions. The framework is validated as a practical edge solution, achieving real-time inference (3–4 ms per sample) with low computational overhead on Raspberry Pi clients, representing a lightweight edge client.

INDEX TERMS Autonomous Vehicle, BERT, CAN Bus Protocol, Federated Learning, In-Vehicle Networks, Intrusion Detection System, Lightweight, Transformers

I. INTRODUCTION

As autonomous vehicles (AVs) currently becoming increasingly reliant on a complex in-vehicle networks (IVNs) to facilitate communication among various Electronic Control Unit (ECU), the necessity for a robust Intrusion Detection Systems (IDS) is of high priority. These ECUs, primarily utilizing protocols such as Controller Area Network (CAN), are essential for the operation of critical functions such as engine control, braking, navigation systems, etc. The interconnected nature of these systems also bring a concern as it broadens the attack surface, exposing vehicles to a various cyber threats. Attackers can exploit the vulnerabilities in in-vehicle communication protocols to manipulate vehicle functions, potentially compromising passenger safety and vehicle integrity, just like an experiment in 2015, a remote hacking involving Jeep Cherokee [1], resulting in the attacker gaining controls of critical vehicle functions, higlighting the consequences that may arise out of such exploits.

In order to address this problem, many researchers have concentrated on creating security techniques, such as the usage of intrusion detection systems (IDS) to examine CAN traffic in order to find anomaly that can point to possible attacks or intrusions such as Denial of Service (DoS), Fuzzy, and Spoofing attack. These systems are typically divided into two categories: Conventional IDS and the more advanced, ML/DL-based IDS. Conventional IDS usually used a rule-based approaches in detecting intrusion, utilizing pre-established rules to identify anomalies in CAN Bus communication. Because of their simplicity, they are easy to design and frequently used in the industry. These devices provide quick and affordable vehicle incursion detection. However, a significant drawback of rule-based intrusion detection systems is their dependence on extensive rulesets that cover every potential CAN traffic feature [2]. As a result, rule-based intrusion detection systems (IDSs) are excellent at identifying known attacks, but they have trouble spotting new or unexpected attack patterns.

Current trends indicates an increasing number of both researchers and practitioners on implementing an advanced IDS, by utilizing Machine Learning (ML) and Deep Learning (DL) techniques to detect anomalies and malicious activities on Autonomous Vehicle. Researchers conducting experiments and researching on attacks such as replay attacks, injection attacks, and denial of service attempts, the systems are able to identify them, enhancing the resilience against cyber intrusions. Continuous development and deployment of IDS that incorporate advanced approaches like deep learning for anomaly detection are crucial to securing IVNs and ensuring the safety and reliability of future autonomous driving technologies. The underlying problem on leveraging ML approaches to develop an IDS is, it is heavily dependent to the quality of the manual feature engineering, while deep learning doesn't, since it extract the features automatically [3].

A lot of research have been done to create and develop an IDS using Deep Learning approaches. A research by Lo et al.[4], aimed to develop an IDS that leverages both spatial and temporal data representation to improve detection accuracy, by employing a combination of Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) networks, allowing for the automatic extraction of features from CAN traffic. This approach has its own limitations, such as evaluated by using only one type of Datasets and yet to be tested by other data or different environment. The study conducted by Althunayyan et al.[5], proposed a hybrid approach that combines supervised and unsupervised learning methods, utilizing an artificial neural network (ANN) for detecting known attacks and an LSTM-autoencoder for identifying previously unseen attacks while also applying Federated Learning (FL) framework that leverages diverse driving behaviors while ensuring privacy protection during data training.

Nevertheless, there is also an observational gap regarding how well current models would perform under different driving conditions and environments, which necessitates more thorough simulation implementation to represent real-world circumstances even more. To bridge this gap, this paper presents Fed-CALiBER, a framework utilizing Federated Learning for privacy-preserving, collaborative intrusion detection with a lightweight BERT model suited for edge devices. This approach is evaluated via a realistic simulation environment where distinct CAN datasets are assigned to Raspberry Pi clients, representing diverse real-world vehicular contexts. This study offers the following contributions:

- Proposed Fed-CALiBER, a framework incorporating Federated Learning (FL) to preserve the privacy of each client's local data, utilizing the attention mechanism of a lightweight BERT model optimized for high performance on resource-constrained devices; and
- Designed a simulation to more accurately represent real-world conditions (Non-IID) by assigning distinct datasets to each client, represented by a Raspberry Pi.

The remainder of this paper is structured as follows: Chapter 2 presents the background of the CAN Bus protocol, common attack types, and the motivation for adopting a lightweight approach. Chapter 3 provides a review of related work in this domain. Chapter 4 outlines the proposed method, detailing its key components?including the lightweight BERT model, federated learning framework, and experimental setup. Chapter 5 presents and discusses the experimental results. Chapter 6 concludes the study and outlines potential directions for future work.

II. BACKGROUND AND MOTIVATION

This section will discuss the overview of CAN Bus protocol and highlights common attack types, including Denial of Service (DoS), Fuzzy, and Spoofing attacks. It also outlines the rationale behind implementation of lightweight model and federated learning approach in this study.

A. CONTROLLER AREA NETWORK (CAN) BUS

The Controller Area Network (CAN Bus) was developed as a multi-master bus protocol (without a central controller) designed for broadcasting messages [6]. It was engineered to serve as a reliable communication medium among Electronic Control Units (ECUs) and has become the most widely used communication protocol within in-vehicle networks (IVNs).

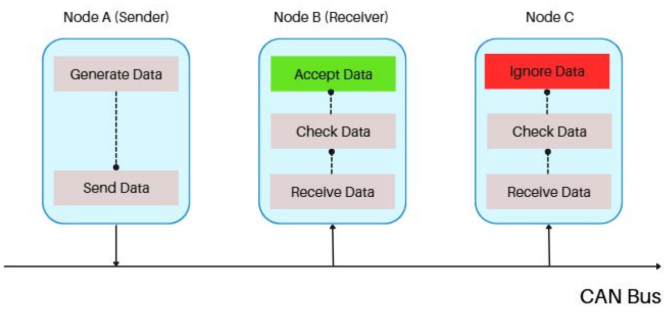
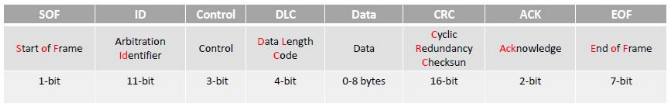


FIGURE 1. CAN Bus Topology

The Controller Area Network (CAN) bus is a multi-master, message-oriented protocol fundamental to in-vehicle communication. Unlike traditional networks, CAN does not use source or destination addresses; instead, messages are broadcast and identified by a unique CAN ID, with this study adopting the standard 11-bit format [7].Each Electronic Control Unit (ECU) is configured to process only messages with specific IDs relevant to its function. To manage network traffic, CAN employs a priority-based arbitration system where messages with numerically lower CAN IDs are given transmission priority, ensuring that critical data is not delayed. This ID-based prioritization and broadcasting mechanism, while efficient, forms the basis for several potential attack vectors [8].

B. COMMON ATTACK TYPES AGAINST CAN BUS

The operational mechanism of the CAN Bus, which relies on broadcast-based message distribution and a priority scheme,



SOF	ID	Control	DLC	Data	CRC	ACK	EOF
Start of Frame	Arbitration Identifier	Control	Data Length Code	Data	Cyclic Redundancy Checksun	Acknowledge	End of Frame
1-bit	11-bit	3-bit	4-bit	0-8 bytes	16-bit	2-bit	7-bit

FIGURE 2. **CAN Bus Frame**

combined with the absence of fundamental security measures such as encryption and authentication, renders the system highly vulnerable to cyberattacks. Several studies have demonstrated that once an attacker gains access to the in-vehicle network (IVN) either through direct connection via the OBD-II port or remotely via wireless interfaces, they are capable of injecting tampered messages into the network, leading to various types of attacks such as Denial of Service (DoS), Fuzzy, and Spoofing attacks [9].

1) Denial of Service (DoS)

A DoS attack aims to disrupt network availability by continuously sending (spamming) messages. Given the message prioritization scheme implemented by the CAN Bus, an attacker can inject high-priority messages to dominate the network and block legitimate traffic. As illustrated in fig. 3, Node B (which is compromised/malicious) persistently sends messages with ID 0x00 (the highest priority), causing messages from Node A and Node C to be deferred until the high-priority message transmission is complete.

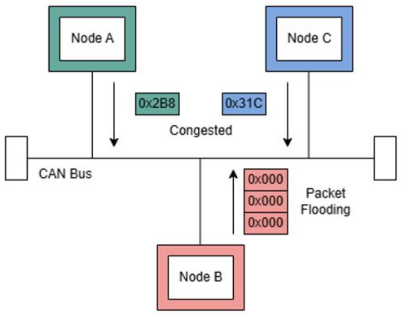


FIGURE 3. **Illustration of DoS Attack on CAN Bus**

2) Fuzzy Attack

Fuzzy attacks are intended to disrupt vehicle functions and delay normal message delivery. In this type of attack, the adversary injects multiple messages with random CAN IDs and arbitrary data payloads to trigger malfunctions in specific vehicle components. This is illustrated in fig. 4, where the attacker transmits messages with randomly generated CAN IDs into the network.

3) Spoofing Attack

Spoofing attacks aim to impersonate a specific message (identified by a particular CAN ID) to gain control over a certain vehicle function. The attacker first identifies the target CAN ID or ECU and then injects manipulated messages using

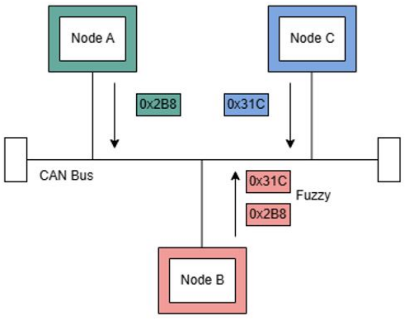


FIGURE 4. **Illustration of Fuzzy Attack on CAN Bus**

that ID. As depicted in fig. 5, Node B (malicious) impersonates Node A and repeatedly sends forged messages under that CAN ID, exploiting the lack of authentication mechanisms in the CAN Bus system.

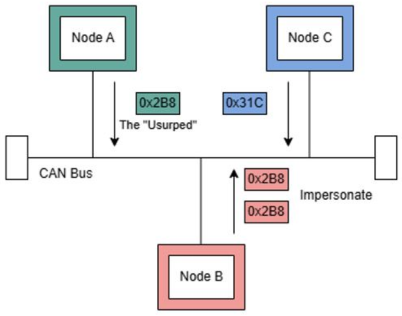


FIGURE 5. **Illustration of Spoofing Attack on CAN Bus**

C. MOTIVATION FOR LIGHTWEIGHT AND FEDERATED IDS DESIGN

Deploying a modern IDS in a real-world vehicular fleet presents several significant challenges that traditional approaches fail to address. In a realistic multi-vehicle scenario, the data is inherently Non-Independent and Identically Distributed (Non-IID), as each vehicle generates CAN traffic with unique statistical properties and attack manifestations, hindering the generalization of a single, centrally trained model. Furthermore, conventional centralized training paradigms require the collection of sensitive CAN data, which not only raises significant privacy concerns but also creates a single point of failure. Compounding these issues, many sophisticated deep learning models are too computationally intensive ("heavy") for practical on-board deployment within resource-constrained ECUs or on cost-effective edge devices like the Raspberry Pi, limiting the feasibility of real-time, on-device inference.

Addressing these multifaceted challenges necessitates a paradigm shift towards a solution that is simultaneously resource-efficient, privacy-preserving, and robust to data heterogeneity. This study is therefore motivated by the need for an IDS that can operate effectively under such constraints. We propose leveraging a lightweight BERT variant, inspired by architectures like TinyBERT, which is specifically adapted to learn efficient representations of CAN message features without the prohibitive computational cost of larger models, making it ideal for low-latency, on-device inference. To tackle the issues of data privacy and the Non-IID nature of distributed datasets, we incorporate Federated Learning (FL). By training this compact model collaboratively across multiple clients (each on its local, distinct dataset) without sharing raw data, FL inherently preserves privacy and enables the global model to learn from a diverse range of conditions, thereby improving its ability to generalize and enhancing overall detection robustness for an entire fleet.

III. RELATED WORKS

This section will discuss the recent researches developments and problems regarding utilization of Transformers and FL on developing IDS for CAN Bus.

A. TRANSFORMERS FOR IDS ON AV

A study by Cobilean et al.[10], proposes a decoder-only Transformers, named CAN-Former IDS, for CAN Bus anomaly detection, leveraging self-attention over tokenized CAN messages (9 tokens consists of 1 ID + 8 payload bytes) to model long-range dependencies and predict the next token. It outperforms LSTM, ResNet, and graph-based IDS in accuracy and inference speed but suffers from higher computational cost and increased false positives due to payload noise. CANBERT, which was proposed by Nwafor and Olufowobi[11], uses BERT to classify CAN messages, pretraining on normal traffic to learn semantics and fine-tuning on attack-labeled data, achieving perfect detection scores across multiple attack types. Albeit its superiority, the authors acknowledge that pretraining is computationally expensive and real-time processing is still an issue, which could limit deployment in resource-constrained environments. The authors also mentions that the experiment lacks of diverse datasets. Furthermore, on their result table, the authors compared a normal messages to each attack one by one, and did not apply a multiclass classification in one go. A proposed bidirectional GPT-based Transformers IDS is introduced by Jo et al.[12], enhancing detection of subtle CAN Bus attacks (e.g., spoofing) by processing input in both forward and backward directions. It surpasses CAN-BERT, and CAN-Former in precision and recall but poses challenges for real-time ECU deployment due to long input sequences and computational overhead.

In CAN-BERT do it paper by Alkhatib et al.[13], the authors applies self-supervised BERT with masked language modeling to capture bidirectional context in CAN ID sequences, successfully improving injection attack detection over RNNs, PCA, and Isolation Forest models. Evaluation was done using the Car Hacking Challenge 2020 dataset, resulting in real-time inference, taking between 0.8 to 3 ms with high F1-scores between 0.81 and 0.99 across three car types and multiple attack types (Flooding, Fuzzy, Malfunction). The downside of this research is it requires high computational cost. A study by Fu et al.[14], proposed IoV-BERT-IDS, combining pre-trained BERT with custom semantic extractors to transform CAN frames into structured byte sentences, to make it fit as an input for Transformer. It achieves superior accuracy over AE, VAE, and ByteSGAN across several IDS datasets. Despite its strong performance, the model faces challenges including high computational cost, difficulties in detecting certain attack types like Malfunction attacks. To address these, the authors recommend TinyBERT variants as future research directions.

A research by Guan et al.[15] proposed a Transformers-based Intrusion Detection System (called PTIDS) for CAN Bus, they utilized Principal Component Analysis (PCA) for feature selection and a multi-head self-attention mechanism by using only the encoder block of a Transformers architecture, to effectively capture temporal dependencies in vehicle network traffic. PTIDS achieves 99.80% accuracy on the Car Hacking Dataset, with the trade-offs of requiring high computational costs and large amounts of data for training due to its complexity. Zhang et al.[16] on their study, proposed a federated two-stage Transformers IDS using an encoder-only architecture with multi-head attention, tailored to handle non-IID, imbalanced CAN Bus datasets in a privacy-preserving manner. The model outperforms LSTM, CNN, and other baseline models in both detection accuracy and speed. However, the authors admits that there are several drawbacks behind its superiority, such as requiring more computational resources than a lightweight LSTM model, and a limited onboard deployment due to its complexity and its required cost, suggesting a model optimization for edge devices deployment. Taneja and Kumar[17] extended the CNN-LSTM paradigm by integrating an attention mechanism to capture spatial and temporal features across multiple time horizons without requiring feature preprocessing. However, their experiments were limited to PCgrade hardware, lacking evaluation on embedded platforms like Raspberry Pi or ECU environments. Although their model did not use Transformers, it remains relevant for highlighting deployment limitations on lightweight platforms, which this study addresses by adopting a lightweight BERT-based approach and implementing it on Raspberry Pi.

B. FL FOR IDS ON AV

In recent years, the application of FL in enhancing security for IDS in AV has garnered significant attention due to its potential to address privacy concerns associated with conventional centralized machine learning approaches. Driss et al.[18], proposed an FL framework specifically for detecting cyberattacks within Vehicular Sensor Networks (VSNs), achieving detection accuracy of 99.52% by leveraging GRU and an RF ensemble method on the "Car Hacking: Attack and Defense Challenge 2020" dataset. Expanding on the advantages of FL in their study, Bhavsar et al.[19], introduced a Federated Learning-based IDS that allows edge devices to learn from local data while preserving user privacy. Their experiments conducted using the NSL-KDD and Car-Hacking datasets on Raspberry Pi and NVIDIA Jetson Xavier platforms to representate edge/client and central server, showcased high detection accuracies of up to 99% with reduced loss values.

Similarly, Althunayyan et al.[5], in their study presented a multi-stage IDS leveraging Artificial Neural Networks (ANN), LSTM networks, and FL, specifically targeting in-vehicle networks's CAN Bus. The experiments resulting in a remarkable F1-score exceeding 0.95 for unseen attacks, alongside a detection rate of 99.99%, maintaining a lightweight architecture suitable for resource-constrained environments with the model taking only 2.98MB in the storage. Zhang et al.[16] proposed a two-stage intrusion detection system using an encoder-only Transformer trained under a federated learning framework to address privacy concerns and data imbalance in crowdsourced CAN Bus datasets. By sharing only model parameters across clients, their approach effectively mitigates the challenges of non-IID data and enhances detection accuracy without requiring centralized access to sensitive in-vehicle data

Despite these advancements, notable gaps remain in the current research landscape. While various studies, such as those by Althunayyan et al.[5] and Zhang et al.[16], demonstrate the effectiveness of deep learning models like ANN-LSTM and Transformers Encoder block in achieving high detection accuracy rates, they also reveal significant limitations, particularly the lack of real-time application environments. To the best of my knowledge, there has been no prior research exploring the use of Federated Learning in Autonomous Vehicle In-Vehicle Networks utilizing lightweight BERT models. This study aims to address this gap by simulating the conditions of distributed clients (AVs) in various environments, which leads to different data collections. By implementing lightweight BERT model, we seek to achieve high detection accuracy that reflects real-world scenarios involving multiple clients and diverse datasets. Fig. 1 illustrates the literature map for this research, based on existing studies. Subsequently, Tables 1 and 2 show the comparison between the proposed work and existing studies.

IV. PROPOSED METHOD

This section details our proposed Fed-CALiBER framework for CAN bus intrusion detection, designed to be both resource-efficient and privacy-preserving. We first describe the architecture and pre-training of our lightweight BERT model tailored for CAN data, followed by an explanation of the federated learning scheme used for collaborative training, and finally, the experimental setup for evaluation.

A. LIGHTWEIGHT BERT

The core of our client-side intrusion detection is the Lightweight BERT model specialized for CAN Bus, a lightweight Transformer encoder specifically designed for efficient processing of CAN bus messages. Fig. 7 illustrates the whole pipeline of the lightweight BERT and Fig.8 shows the proposed lightweight BERT architecture. Each CAN message, initially comprising a timestamp, CAN ID, Data Length

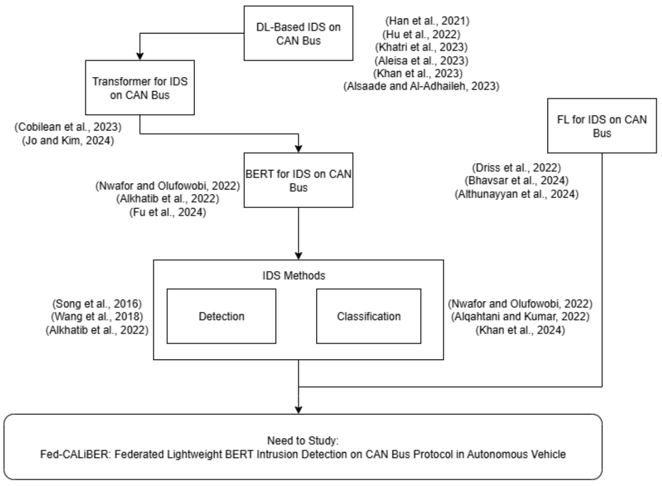


FIGURE 6. Literature Map

Code (DLC), and up to eight data bytes, undergoes preprocessing. To handle missing values, any null entries in the DATA column that were less than eight bytes were padded with zeros. Similarly, for every value that is fewer than four bytes in CAN ID column is also padded with zero. To ensure uniformity, the values in the Timestamp column were adjusted to display six digits after the decimal point, and set the payload's value into uppercase (for hexadecimal letter). The Label column is also mapped to transforms the categorical labels (R/T) into numerical values, with 0 as a normal message, 1 identifies as DoS attack, 2 as Fuzzy attack and 3 as Spoofing attack. The Label column on the dataset that's going to be used for pretraining is omitted. The subsequemt step are formatting the components into a space-delimited sentence.

This sentence is then tokenized using a trained Byte-Level BPE tokenizer with a vocabulary size of 1000. Special tokens [CLS], [UNK], and [SEP] are added, and the resulting token sequence is padded or truncated to a maximum sequence length of 32. The Byte-Level BPE tokenizer was trained on a corpus of approximately 6.3 million normal CAN messages merged from the normal traffic portions of all three datasets used in this research. The vocabulary size of 1000 was chosen as a balance to capture common CAN message components as single tokens while still allowing for sub-word tokenization of more variable elements. The sequence length of 32 was selected based on an empirical analysis of our tokenized data, which showed that the vast majority of CAN message "sentences" tokenize to a length between 19 and 24 tokens. This choice avoids information loss from truncation while minimizing the computational overhead from excessive padding.

The Fed-CALiBER's lightweight BERT architecture consists of an embedding layer, which computes the sum of token, position, and token type embeddings, followed by 4 Transformer encoder layers. Each encoder layer is composed of two main sub-layers: a multi-head self-attention

TABLE 1. Comparison of Transformer-based IDS for In-Vehicle Networks

Feature/Aspect	**Cobilean et al. [10]** (CAN-Former)	**Nwafor & Olufowobi [11]** (CANBERT)	**Alkhatib et al. [13]** (CAN-BERT)	**Fu et al. [14]** (IoV-BERT-IDS)	**Guan et al. [15]** (PTIDS)	**Zhang et al. [16]** (Federated 2-stage)	**Fed-CALiBER (Proposed)**
Base Model Architecture	Decoder-only Transformer	BERT (Encoder)	Self-supervised BERT (Encoder)	Pre-trained BERT + Custom Extractors	Encoder-only Transformer	Encoder-only Transformer	**Lightweight BERT (Encoder)**
Pre-training	Next-token prediction (not MLM)	On Normal CAN (MLM, RoBERTa-style)	Self-supervised MLM on CAN ID seq.	Uses pre-trained (NLP) BERT	N/A (No Pre-training)	N/A (Trained on task)	**On Normal CAN (MLM, RoBERTa-style)**
Input Representation	CAN ID + DATA sequence (separate)	Full CAN sentence (linguistic style)	CAN ID sequences	Structured Byte Sentences	PCA-selected CAN features (14-dim)	Raw CAN features (unspecified)	**Formatted CAN Sentences (Full Msg)**
Key Novelty/Focus	Next token prediction for anomaly	BERT for CAN classification	MLM for CAN ID context	Hybrid BERT with Byte-sentence input	PCA + Transformer Encoder	Two-stage FL, Non-IID handling	**Federated Learning + Lightweight BERT**
Federated Learning	No	No	No	No	No	Yes (Two-stage)	**Yes (Iterative FedAvg)**
Lightweight Focus	Moderate (faster than some baselines)	No (acknowledges pretrain cost)	No (high compute cost)	Suggests TinyBERT for future	No (high compute cost)	Mentions optimization needed	**Yes (Explicit Design & RPi Eval)**
Dataset Diversity	Single dataset implied	Lacking (acknowledged)	Car Hacking 2020 (3 car types)	Yes (4 distinct IDS datasets)	Car Hacking Dataset	Yes (3 distinct datasets)	**Yes (3 Distinct Datasets, Cross-Eval)**
Real-time/Edge Deployment	Potential (some concerns)	Limited by processing cost	Yes (0.8-3ms inference)	High compute cost	High compute cost	Limited by complexity	**Yes (Demonstrated on RPi, 3-4ms)**
Privacy	Not addressed	Not addressed	Not addressed	Not addressed	Not addressed	Yes (via FL)	**Yes (via FL Data Localization)**

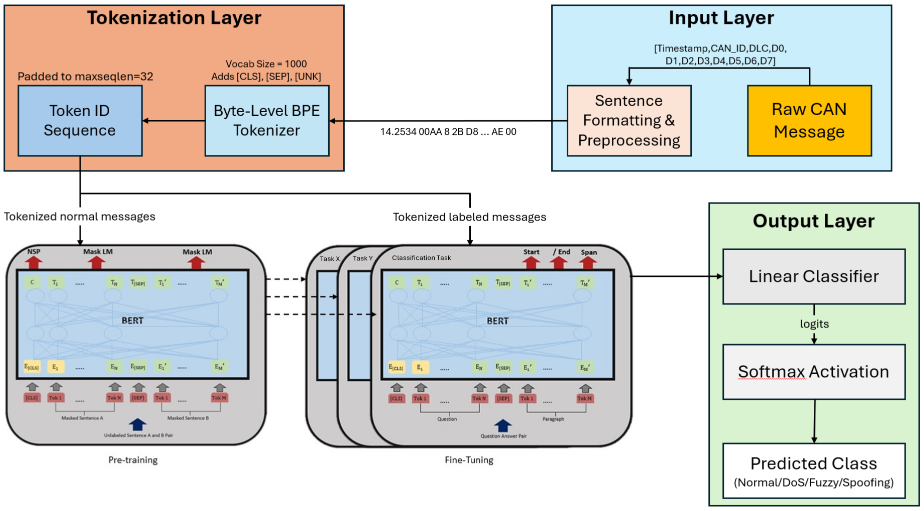


FIGURE 7. Proposed Lightweight BERT Pipeline Illustration

mechanism and a position-wise fully connected feed-forward network, with residual connections and layer normalization applied around each sub-layer. The multi-head self-attention mechanism, in our configuration utilizing 4 attention heads, allows the model to jointly attend to information from different representation subspaces. For each head, scaled dot-product attention is computed as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \qquad (1)$$

where Q (Query), K (Key), and V (Value) matrices are linear projections of the input hidden states, and d_k is the dimension of the keys. The outputs of these heads are concatenated and linearly projected. The subsequent position-wise feed-forward network (FFN) consists of two linear transformations with a GELU activation function in between:

$$\text{FFN}(x) = \text{GELU}(xW_1 + b_1)W_2 + b_2 \qquad (2)$$

The hidden size throughout the model is 256, and the FFN's intermediate size is 1024. For classification, the representation derived from the [CLS] token's final hidden state from the last Transformer layer is passed through a final linear

TABLE 2. Comparison of Federated Learning Approaches for In-Vehicle IDS

Feature/Aspect	**Althunayyan et al. [5]** (ANN+LSTM-AE FL)	**Bhavsar et al. [19]** (Generic FL IDS)	**Driss et al. [18]** (GRU+RF FL for VSN)	**Zhang et al. [16]** (Federated 2-stage)	**Fed-CALiBER (Proposed)**
Base Model Architecture	ANN (known), LSTM-AE (unknown)	Logistic Regression (LR) and (CNN)	GRU + RF Ensemble	Transformer Encoder	**Lightweight BERT (Encoder)**
FL Algorithm	Not specified (implies FedAvg)	Federated Averaging (FedAvg)	Not specified (implies FedAvg)	Not specified (implies FedAvg)	**Federated Averaging (FedAvg)**
Focus on Lightweight Client Model	Yes (2.98MB model, RAM usage < 1GB)	Usage not explicitly stated but mentions edge devices (RPi, Jetson)	Not explicitly for on-device VSN nodes	Acknowledges their Transformer is not lightweight for edge (RAM usage around 5GB)	**Yes (TinyBERT inspired), 3-4 ms / sample on inference, RAM usage 400 - 600 MB, 13 MB model, around 3.4 million parameters**
Handling Non-IID Data	Leverages diverse driving (implies Non-IID)	Not explicitly detailed	Not explicitly detailed	Explicitly tailored for Non-IID, imbalanced	**Evaluated with Distinct Datasets per Client**
Edge Device Eval	Implied (lightweight)	Yes (Raspberry Pi, Jetson)	Not detailed	Suggests optimization needed	**Yes (Raspberry Pi)**
Dataset Simulation	Diverse driving behaviors	NSL-KDD, Car Hacking	Car Hacking Challenge 2020	Car Hacking Dataset, Ford Transit 500 Dataset, IVN Intrusion Detection Challenge Dataset	**Car Hacking Dataset, Car Hacking Challenge 2020 Dataset, Survival Analysis Dataset**

layer. A softmax activation function is then applied to the output of this linear layer (logits) to produce probabilities for the 4 classes: Normal, DoS, Fuzzy, and Spoofing. This compact architecture was chosen to balance representational power with the computational constraints of edge devices. The model is pre-trained from scratch on a large corpus of normal CAN messages using the Masked Language Modeling (MLM) objective, adopting RoBERTa-style by Liu et al.[20] optimizations such as omitting the Next Sentence Prediction task.

B. FEDERATED LEARNING ARCHITECTURE

Federated Learning (FL) framework is employed to address the privacy requirements and handle the Non-IID nature of distributed vehicular data, illustrated in Fig. 9. This framework operates on a client-server model. A central server coordinates the learning process, while multiple clients (representing individual vehicles, simulated by Raspberry Pi devices in our experiments) perform model training exclusively on their local CAN bus datasets. This ensures that raw, potentially sensitive, in-vehicle data never leaves the client device, inherently preserving data privacy.

The FL process is iterative and occurs over several communication rounds as depicted in the Fig. 10. In each round, the following steps take place:

1) Model Distribution: The central server transmits the current global model parameters to a selection of participating clients;
2) Local Client Training: Upon receiving the global model parameters, it then performs local fine-tuning using its own private dataset for a predefined number of local epochs, optimizing the model based on its specific data characteristics. This local training phase allows each client to contribute its unique knowledge;
3) Model Update Submission: After local training, each client transmits the parameters back to the central server, representing the knowledge gained from its local data;
4) Server-Side Aggregation: The central server then aggregates these individual client updates to produce a new, improved global model. This aggregated model then becomes the starting point for the next communication round.

The central server aggregates the model parameters using Federated Averaging (FedAvg) algorithm [21]. FedAvg was selected as it is the foundational and most widely adopted

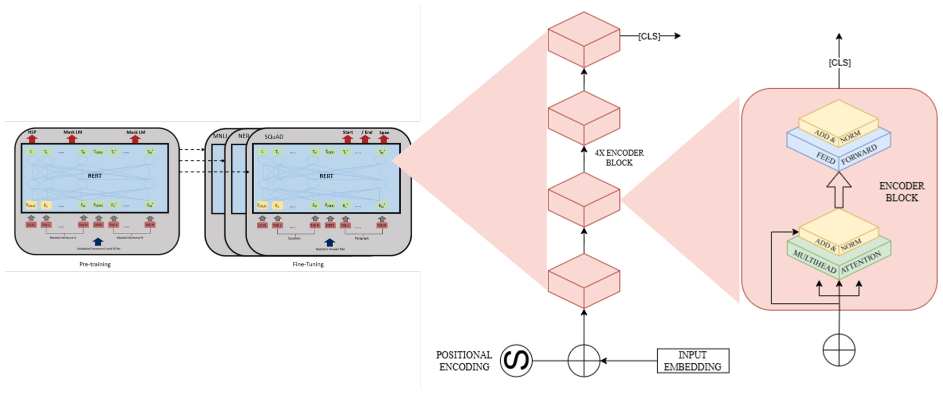


FIGURE 8. **Proposed Lightweight BERT Architecture**

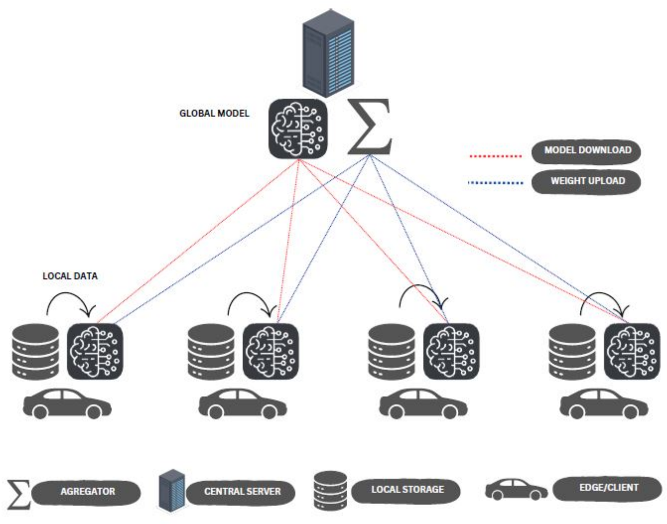


FIGURE 9. **Federated Learning Architecture**

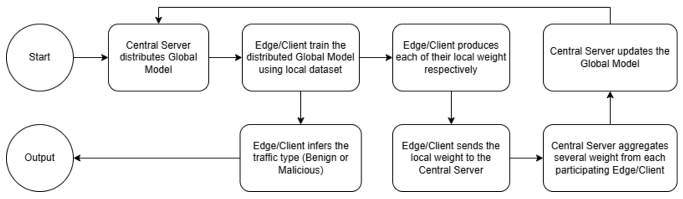

Start
Central Server distributes Global Model
Edge/Client train the distributed Global Model using local dataset
Edge/Client produces each of their local weight respectively
Central Server updates the Global Model
Central Server aggregates several weight from each participating Edge/Client
Edge/Client sends the local weight to the Central Server
Edge/Client infers the traffic type (Benign or Malicious)
Output

FIGURE 10. **Federated Learning Flowchart**

baseline algorithm in federated learning research. Its implementation simplicity and low computational overhead make it an ideal starting point for establishing the fundamental viability of our proposed lightweight model within a federated framework, particularly in a resource-constrained edge environment. This choice allows us to provide a clear performance benchmark and isolate the effectiveness of the model architecture and the FL paradigm itself, without the added complexity of more advanced optimization algorithms.

$$w_{t+1} = \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^{k} \tag{3}$$

w_{t+1} represents the global model parameters at round t, and w_{t+1}^{k} are the parameters from client k (which trained on n_k local samples) after local training in that round, the server updates the global model to w_{t+1}. K is the number of participating clients and $n = \sum n_k$. This iterative process allows the global model to learn collaboratively from the diverse datasets held by all clients without centralized data collection, leading to a more generalized and robust intrusion detection system. Our experiments, detailed in Section 4.3.4, utilize a two-cycle approach with this FL framework, where client dataset assignments are shifted between cycles to further simulate real-world data variability and assess model adaptability.

C. EXPERIMENTAL SETUP

To empirically evaluate the performance and efficacy of our proposed Fed-CALiBER framework, a comprehensive experimental setup was designed. This setup encompasses the implementation environment for our models and federated learning process, the datasets employed to simulate diverse client environments, and the metrics used for performance assessment. Furthermore, we detail a two-cycle experimental design to investigate both initial generalization and the adaptability of the federated model to shifting data distributions.

1) Implementation Details

The Fed-CALiBER framework was implemented using Python, leveraging several key libraries for deep learning and federated learning. The lightweight BERT model and its fine-tuning were built using PyTorch [22] and the Hugging Face Transformers library [23]. For the federated learning component, we utilized the Flower framework [24] to manage client-server communication and model aggregation. Specifically, the Federated Averaging (FedAvg) algorithm was employed as the server-side aggregation strategy.

The experimental environment consisted of a central server simulated on a MSI Vector 16 HX AI A2XW laptop, using Intel Core Ultra 9 275HX, 16GB RAM DDR5-5600 NVidia RTX5080 16GB VRAM, and three distinct clients. Two clients were represented by Raspberry Pi 5 devices with Arm Cortex-A76 CPU 2.4GHz quad-core, 8GB LPDDR4X RAM variant, and Lexar 16GB MicroSD. Due to resource constraint, the third client was simulated on Asus N46VZ Intel Core i5-3210M 2.5 GHz, 16GB RAM DDR4, to represent varied edge device capabilities. All devices operated within the same local network. For the federated learning process, each client performed local training for 1 epoch per communication round using a local batch size of 16 and a learning rate of 3e-5 with the AdamW optimizer. The server was configured to run 5 rounds for the first experimental cycle and another 5 rounds for the second cycle, resulting in a total of ten rounds. Using all 3 clients as a minimum required client to contribute updates before aggregation.

2) Datasets

To simulate a realistic distributed environment with Non-IID data across clients, three distinct, publicly available CAN bus intrusion detection datasets were utilized. The dataset comprised of various samples. For later use in Raspberry Pi, the training and testing dataset were also subsetted with 20.000 rows and 50.000 rows respectively. This reduction was necessary to accomodate the constrained resources in Raspberry Pi.

- Dataset A (Car Hacking Dataset): Sourced from the Hacking and Countermeasure Research Lab (HCRL) [25], this dataset contains CAN traffic from a real vehicle under various injected attacks, including Denial of Service (DoS), Fuzzy, and Impersonation (Spoofing) attacks, alongside normal operational data. This dataset contains 16,5 million rows in total, consists of 14,2 million rows of normal data and 2,3 million rows of attacks.
- Dataset B (Hacking Challenge 2020 Dataset): This dataset was generated during the "Car Hacking: Attack & Defense Challenge" competition in 2020 from a specific target vehicle [25]. It includes normal traffic and four attack types: Flooding (DoS), Spoofing, Replay, and Fuzzy. For consistency with other datasets in this study, Replay attack messages were omitted. This dataset has 3 distinct sets, with each set derived from preliminary training round, premilinary submission round, and final submission round. In total, there are 8,5 million rows in total, with 7,8 million rows of normal data and 873 thousand rows of attacks.
- Dataset C (Survival Analysis Dataset): Also from HCRL [25] this dataset comprises CAN traffic from three different vehicle types under normal conditions and injected attacks including Flooding (DoS), Fuzzy, and Malfunction (Spoofing) scenarios. This dataset was obtained from three different vehicle, which is Hyundai Sonata, Kia Soul and Chevrolet Spark.

For pre-training the lightweight BERT model, a corpus of approximately 6.3 million rows of normal CAN messages was created by merging the normal traffic segments from all three datasets. For the federated fine-tuning experiments, each client was assigned a distinct dataset, with each subset limited to 20,000 rows. The data processing pipeline for all datasets involved converting raw CAN messages into a space-delimited "sentence" format (Timestamp, CAN ID, DLC, 8 Data Bytes), followed by tokenization using a Byte-Level BPE tokenizer (vocabulary size 1000) with a maximum sequence length of 32. Labels were mapped to numerical values (0: Normal, 1: DoS, 2: Fuzzy, 3: Spoofing). Additionally, larger unseen portions (50 thousand rows each) of these datasets were reserved as final test sets for evaluating the global models produced by the FL process.

3) Evaluation Metrics

The performance of the proposed Fed-CALiBER framework and baseline models was evaluated using a comprehensive set of standard classification metrics. For overall performance, we report Accuracy and weighted averages for Precision, Recall, and F1-score. To gain deeper insights into the model's ability to correctly identify each specific class (Normal, DoS, Fuzzy, Spoofing), per-class Precision, Recall, and F1-score.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{4}$$

$$\text{Precision} = \frac{TP}{TP + FP} \tag{5}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{6}$$

$$\text{F1 Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{7}$$

A Confusion Matrix is presented to visually analyze the classification performance and identify specific misclassification patterns between classes. Furthermore, to assess the "lightweight" aspect and suitability for edge deployment, we report model size (in Megabytes), inference speed (in milisecond per sample on client devices), and observed resource usage (CPU and RAM) on the Raspberry Pi clients during local operations.

4) Testing Scenario

As depicted in Fig. 11, our experimental evaluation is structured into two distinct federated learning cycles to assess both initial generalization and adaptability to data shifts: Cycle 1 (Initial Generalization, Rounds 1-5): The FL process starts with the server distributing the base pre-trained Lightweight BERT model. Client 1 trains on Dataset A, Client 2 on Dataset B, and Client 3 on Dataset C. After 5 federated rounds, the aggregated global model ($GMC1$) is saved. This model's performance is then evaluated offline on held-out test portions of Dataset A, Dataset B, and Dataset C to assess its ability to generalize across the initial set of data distributions.

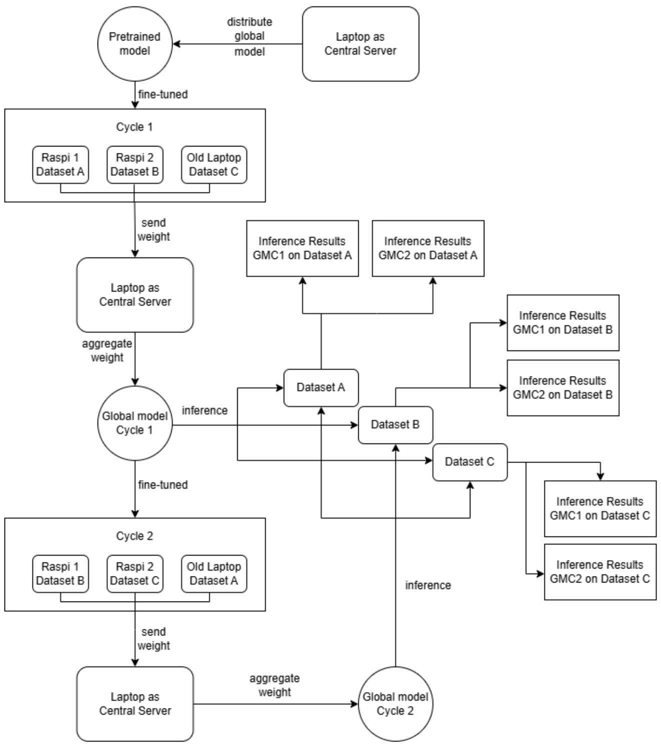


FIGURE 11. Testing Scenario Pipeline

Cycle 2 (Adaptation to Data Shift, Rounds 6 to 10): The FL process resumes, with the server initializing the global model using $GMC1$. For this cycle, the dataset assignments for clients are swapped: Client 1 now trains on Dataset B, Client 2 on Dataset C, and Client 3 on Dataset A. After an additional 5 federated rounds, the new aggregated global model ($GMC2$) is saved. This model is also evaluated offline on the held-out test portions of Dataset A, Dataset B, and Dataset C. This two-cycle design allows for a comparative analysis of the global model's performance before and after clients are exposed to new data distributions, providing insights into the continual learning and adaptation capabilities of the Fed-CALiBER framework.

V. RESULTS AND DISCUSSION

This section presents a comprehensive evaluation of the proposed Fed-CALiBER framework. We begin by establishing the baseline performance of our standalone lightweight BERT model in centralized training scenarios across different hardware platforms and datasets, including an analysis of its cross-dataset generalization capabilities. Subsequently, we detail the performance of the global models generated through our two-cycle federated learning experiment, analyzing their generalization and adaptation to shifting data distributions. This is followed by a comparative analysis against non-federated baselines and relevant existing work. Finally, we discuss the resource usage, efficiency, and privacy implications of Fed-CALiBER.

A. STANDALONE LIGHTWEIGHT BERT PERFORMANCE AND BASELINE COMPARISONS

To validate the effectiveness of the proposed lightweight BERT architecture prior to its integration into the federated learning framework, its performance was first assessed in centralized training scenarios. These experiments aimed to establish its baseline capabilities on individual datasets using both a resource-rich server laptop and a resource-constrained Raspberry Pi.

1) Lightweight BERT Performance on Server Laptop

The proposed lightweight BERT model was initially trained and evaluated on a server laptop, utilizing large subset of each distinct dataset (Dataset A: Car Hacking; Dataset B: Hacking Challenge 2020; Dataset C: Survival Analysis, refers to table 3) to represent its peak potential performance when ample computational resources and data are available. The evaluation metrics is as shown in table 4, 5, and 6, whereas the confusion matrix is shown in fig. 12, 13, and 14. The model achieved exceptionally high accuracy, precision, recall, and F1-scores across all classes for each dataset when trained and tested on its respective data distribution, demonstrating its strong capacity to learn distinguishing features from each CAN traffic corpus. For instance, on Dataset A, an F1-score of 0.9999 was achieved, while for Dataset B, the F1-score was 0.9952, and for Dataset C, it reached 0.9961.

TABLE 3. Large Subset for Central Fine-Tuning

	Car Hacking	Hacking Challenge	Survival Analysis
Normal	3,153,078	2,732,284	1,105,903
DoS	193,749	151,471	88,150
Fuzzy	187,739	76,859	63,742
Spoofing	342,813	34,578	31,422
Total	3,877,379	2,995,192	1,289,217

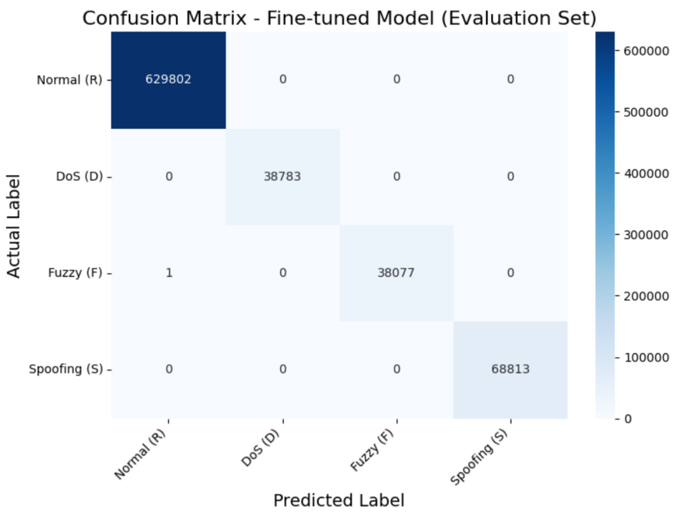


FIGURE 12. Lightweight BERT using Laptop on Car Hacking Dataset Confusion Matrix

TABLE 4. Lightweight BERT using Laptop on Car Hacking Dataset Evaluation Metrics

	Precision	Recall	F1-Score
Normal	0.999998	1.000000	0.999999
DoS	1.000000	1.000000	1.000000
Fuzzy	1.000000	0.999974	0.999987
Spoofing	1.000000	1.000000	1.000000
Weighted	0.999999	0.999999	0.999999

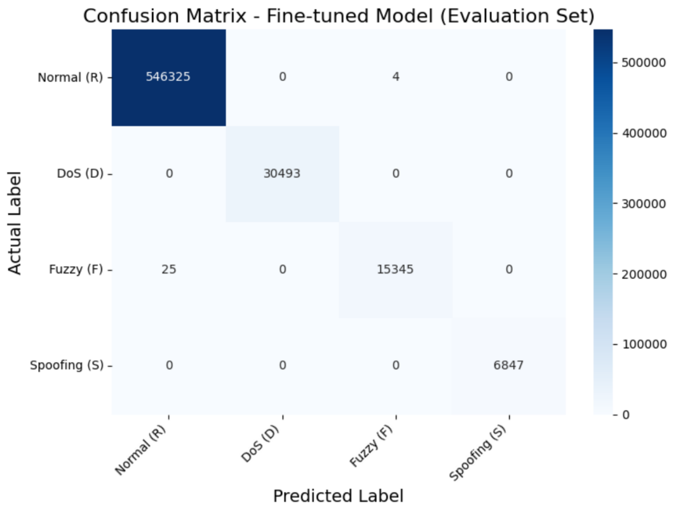


FIGURE 13. Lightweight BERT using Laptop on Hacking Challenge Dataset Confusion Matrix

TABLE 5. Lightweight BERT using Laptop on Hacking Challenge Dataset Evaluation Metrics

	Precision	Recall	F1-Score
Normal	0.999954	0.999993	0.999973
DoS	1.000000	1.000000	1.000000
Fuzzy	0.999739	0.998373	0.999056
Spoofing	1.000000	1.000000	1.000000
Weighted	0.999952	0.999952	0.999952

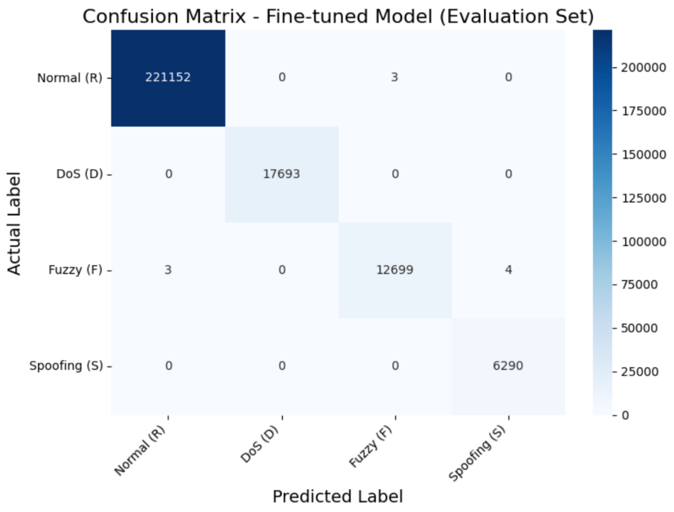


FIGURE 14. Lightweight BERT using Laptop on Survival Analysis Dataset Confusion Matrix

TABLE 6. Lightweight BERT using Laptop on Survival Analysis Dataset Evaluation Metrics

	Precision	Recall	F1-Score
Normal	0.999986	0.999986	0.999986
DoS	1.000000	1.000000	1.000000
Fuzzy	0.999764	0.999449	0.999606
Spoofing	0.999364	1.000000	0.999682
Weighted	0.999961	0.999961	0.999961

2) Lightweight BERT Performance on Raspberry Pi

To simulate deployment on edge devices, the lightweight BERT model was also trained and evaluated directly on a Raspberry Pi 5 using 20,000 row subsets from each primary dataset (A, B, and C). This reduction was necessitated by the device's resource limitations. Nevertheless, studies on few-sample BERT fine-tuning[26][27] suggest that pre-trained Transformer models can achieve strong performance even when fine-tuned on significantly smaller datasets than typically used for larger models, supporting the viability of this approach for our lightweight architecture.

The results, summarized table 7, 8, and 9, while the confusion matrix is shown in fig. 15, 16, and 17, indicate that while there is a performance decrease compared to training on larger datasets on the laptop (e.g., F1-score on Dataset C subset was 0.9985 vs. 0.9999), the model still maintains a high level of detection accuracy. This reduction in dataset size was a necessary compromise to ensure feasible training times and accommodate the memory constraints of the Raspberry Pi, establishing a practical baseline for on-device learning.

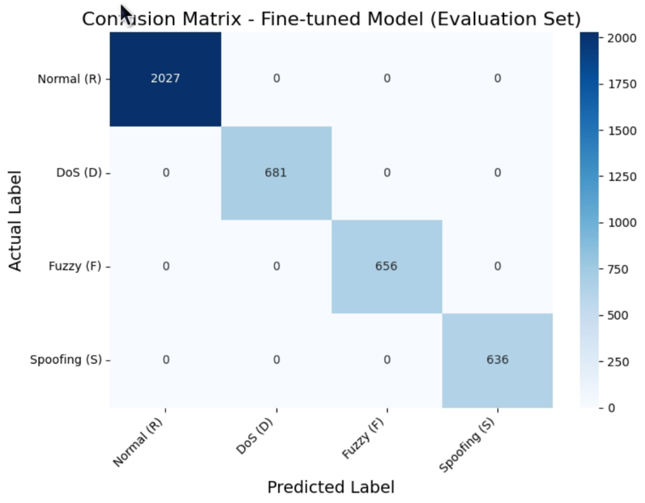


FIGURE 15. Lightweight BERT using Raspberry Pi on Car Hacking Dataset Confusion Matrix

3) Architectural Baseline Comparison

To further justify the selection of our lightweight BERT architecture, its performance on large subset of Dataset A, was benchmarked against two other deep learning architectures: a Convolutional Neural Network (CNN) and a vanilla Transformer Encoder, both trained under similar centralized

TABLE 7. Lightweight BERT using Raspberry Pi on Car Hacking Dataset Evaluation Metrics

	Precision	Recall	F1-Score
Normal	1.000000	1.000000	1.000000
DoS	1.000000	1.000000	1.000000
Fuzzy	1.000000	1.000000	1.000000
Spoofing	1.000000	1.000000	1.000000
Weighted	1.000000	1.000000	1.000000

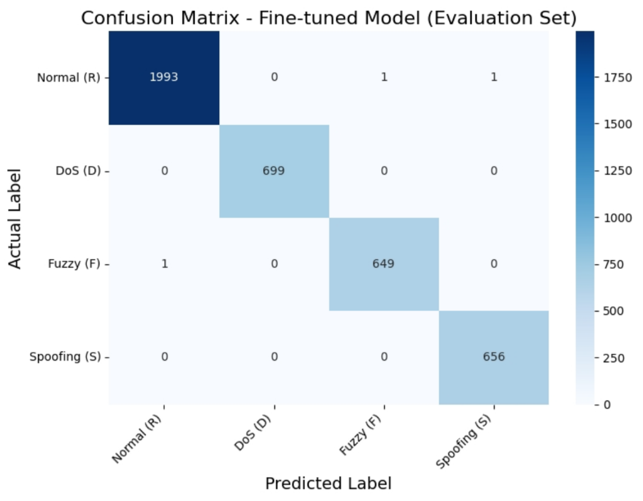


FIGURE 16. Lightweight BERT using Raspberry Pi on Hacking Challenge Dataset Confusion Matrix

TABLE 8. Lightweight BERT using Raspberry Pi on Hacking Challenge Dataset Evaluation Metrics

	Precision	Recall	F1-Score
Normal	0.999498	0.998997	0.999248
DoS	1.000000	1.000000	1.000000
Fuzzy	0.998462	0.998462	0.998462
Spoofing	0.998478	1.000000	0.999238
Weighted	0.999250	0.999250	0.999250

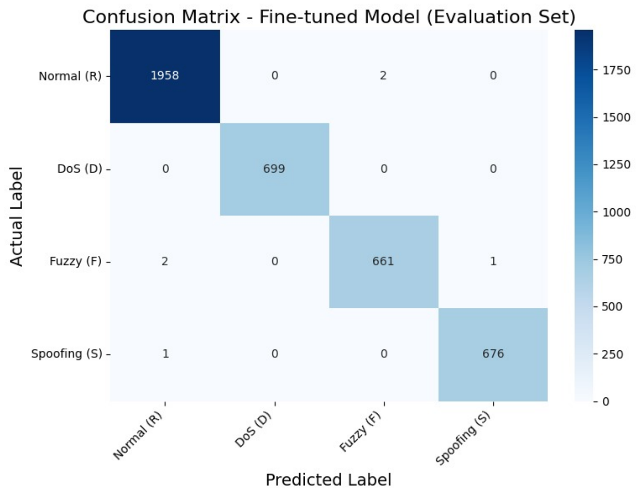


FIGURE 17. Lightweight BERT using Raspberry Pi on Survival Analysis Dataset Confusion Matrix

TABLE 9. Lightweight BERT using Raspberry Pi on Survival Analysis Dataset Evaluation Metrics

	Precision	Recall	F1-Score
Normal	0.998470	0.998980	0.998725
DoS	1.000000	1.000000	1.000000
Fuzzy	0.996983	0.995482	0.996232
Spoofing	0.998523	0.998523	0.998523
Weighted	0.998500	0.998500	0.998500

conditions on the server laptop. As presented in table 10 and fig. 18, by comparing the weighted/average of the metrics, the proposed lightweight BERT model demonstrated its superiority compared to the CNN and vanilla Transformer baselines, underscoring its effectiveness in capturing relevant features from the CAN message.

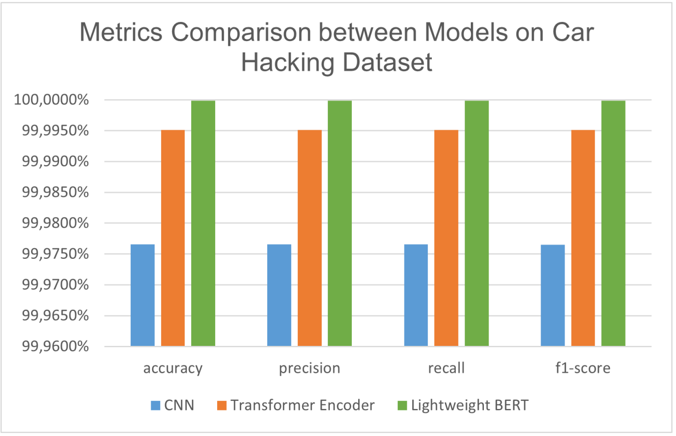


FIGURE 18. Baseline Model Comparison Graph

TABLE 10. Baseline Model Comparison Evaluation Metrics

	Accuracy	Precision	Recall	F1-Score
CNN	99.9765%	99.9765%	99.9765%	99.9765%
Transformer Encoder	99.9951%	99.9951%	99.9951%	99.9951%
Lightweight BERT	99.9999%	99.9999%	99.9999%	99.9999%

4) Discussion of Standalone Model Performance

The standalone evaluations confirm that the proposed lightweight BERT architecture is a potent model for CAN bus intrusion detection, achieving near-perfect results when trained and tested on specific datasets with sufficient resources. While performance on the Raspberry Pi with reduced data is, as expected, slightly lower than on the server laptop with more extensive data, it remains high, validating its potential for edge deployment. The architectural comparisons further establish its advantages over standard CNN and vanilla Transformer approaches for this task. However, the subsequent cross-dataset testing (detailed in Section 5.2.5 as part of the FL comparison) revealed significant performance degradation when these centrally trained standalone models were applied to unseen dataset distributions, motivating the exploration of federated learning.

B. FED-CALIBER: FEDERATED LEARNING RESULTS AND ANALYSIS

This subsection details the experimental results of our proposed Fed-CALiBER framework. We evaluate the performance of the global models generated at the end of each of the two federated learning cycles, where clients contributed updates from distinct and subsequently shifted local datasets. The objective is to assess both the initial generalization achieved through federated averaging and the model's adaptability to evolving data distributions across the clients.

1) Performance of Global Model after Cycle 1 (GMC1)

The following 5 rounds of federated learning in Cycle 1, where pre-trained model on Client 1 trained using Dataset A, Client 2 on Dataset B, and Client 3 on Dataset C, the aggregated global model $GMC1$ was evaluated. Table 11, 12, and 13 present the classification performance of $GMC1$ when tested on held-out, 50 thousand rows of Dataset A, Dataset B, and Dataset C, respectively. These results provide insights into the initial generalization capability of the federated model after learning collaboratively from diverse, specialized client data from the first cycle.

TABLE 11. Global Model Cycle 1 on Car Hacking Dataset (Dataset A)

	Precision	Recall	F1-Score
Normal	0.999919	0.999959	0.999939
DoS	1.000000	1.000000	1.000000
Fuzzy	0.999898	0.999796	0.999847
Spoofing	1.000000	1.000000	1.000000
Weighted	0.999940	0.999940	0.999940

TABLE 12. Global Model Cycle 1 on Hacking Challenge Dataset (Dataset B)

	Precision	Recall	F1-Score
Normal	0.998947	0.995317	0.997128
DoS	1.000000	1.000000	1.000000
Fuzzy	0.989336	0.994811	0.992066
Spoofing	0.994737	0.999540	0.997133
Weighted	0.996713	0.996700	0.996702

TABLE 13. Global Model Cycle 1 on Survival Analysis Dataset (Dataset C)

	Precision	Recall	F1-Score
Normal	0.997872	0.995002	0.996435
DoS	0.999005	1.000000	0.999502
Fuzzy	0.833305	0.996636	0.907682
Spoofing	1.000000	0.790721	0.883131
Weighted	0.966189	0.959780	0.959365

2) Performance of Global Model after Cycle 2 (GMC2)

Cycle 2 commenced with $GMC1$ as the starting global model, and clients were assigned shifted datasets (Client 1 on B, Client 2 on C, Client 3 on A) for an additional 5 federated learning rounds. The resulting aggregated global model, $GMC2$, was then evaluated on the same held-out test portions of Dataset A, Dataset B, and Dataset C. The performance metrics are detailed in table 14, 15, and 16. These results illustrate the model's performance after adapting to the new data distributions introduced by the clients.

There are also a noticable performance discrepancy in the global models on Dataset C, which was consistently lower than on Datasets A and B. Dataset C (Survival Analysis) is the most heterogeneous of the three, as it was collected from three distinct vehicle models (Hyundai Sonata, Kia Soul, Chevrolet Spark). This makes it harder for the global model to achieve the same near-perfect scores. We also hypothesize that attack patterns in this mixed-vehicle context, particularly those labeled as "Spoofing," may have more subtle manifestations, as reflected by the lowest F1-score result compared to other class. This result highlights the challenge that extreme data heterogeneity poses even for federated learning and underscores the importance of more advanced aggregation strategies in a highly diverse fleet.

TABLE 14. Global Model Cycle 2 on Car Hacking Dataset (Dataset A)

	Precision	Recall	F1-Score
Normal	0.999837	0.999837	0.999837
DoS	1.000000	1.000000	1.000000
Fuzzy	0.999592	0.999592	0.999592
Spoofing	1.000000	1.000000	1.000000
Weighted	0.999840	0.999840	0.999840

TABLE 15. Global Model Cycle 2 on Hacking Challenge Dataset (Dataset B)

	Precision	Recall	F1-Score
Normal	0.999233	0.999435	0.999334
DoS	1.000000	1.000000	1.000000
Fuzzy	0.998727	0.995157	0.996939
Spoofing	0.997020	1.000000	0.998508
Weighted	0.998881	0.998880	0.998879

TABLE 16. Global Model Cycle 2 on Survival Analysis Dataset (Dataset C)

	Precision	Recall	F1-Score
Normal	0.998069	0.999189	0.998633
DoS	0.999403	1.000000	0.999701
Fuzzy	0.856479	0.996331	0.921127
Spoofing	1.000000	0.817552	0.899619
Weighted	0.970899	0.966300	0.965926

3) Impact of Data Shift and Continual Learning (GMC1 vs GMC2)

A direct comparison between $GMC1$ and $GMC2$ displayed in table 17, 18, and 19 reveals the impact of the second training cycle with shifted data. For Datasets B and C, $GMC2$ showed a notable improvement in F1-score from 99.6702% to 99.8879% and from 95.9365% to 96.5926% respectively, suggesting successful adaptation and knowledge integration from the new data exposures. A consistent finding across all evaluations is that models generally achieved a slightly lower F1-score on Dataset C (Survival Analysis). This is attributable to Dataset C being the most internally heterogeneous, as it was sourced from three different vehicle models. The resulting diversity in its "normal" traffic baseline and potential variations in attack manifestations make it an inherently more challenging classification task compared to Datasets A and B, which were each sourced from a single vehicle.

Interestingly, a different dynamic appears during the federated aggregation process, particularly in Cycle 2. The slight performance degradation of the global model (GMC2) specifically on Dataset A (Car Hacking). Looking at the training result of only using dataset A, which produced a really good result, it's possible that Dataset A may be the most statistically unique or distinct "outlier" relative to the other two datasets, making the patterns in Dataset A are very specific and don't transfer well to B and C. While Dataset C is internally diverse, its overall characteristics and those of Dataset B may share more commonalities and are helpful for each other, and even somewhat helpful for Dataset A, but not enough to overcome the loss of specialization from Cycle 1. In Cycle 2, Client 3 (training on A) is trying to contribute "specialized A-knowledge" to a model that is simultaneously being pulled towards a "general B/C-knowledge" state by two other clients, where their gradient updates likely pushed the global model towards a feature space that was mutually optimal for both B and C, as they may share more statistical similarities. The general knowledge wins out, improving B and C, but hurting A's specialized performance.

TABLE 17. Metrics Comparison of GMC1 and GMC2 on Car Hacking Dataset

Global Model	Accuracy	Precision	Recall	F1-Score
Cycle 1 (GMC1)	99.9940%	99.9940%	99.9940%	99.9940%
Cycle 2 (GMC2)	99.9840%	99.9840%	99.9840%	99.9840%

TABLE 18. Metrics Comparison of GMC1 and GMC2 on Hacking Challenge Dataset

Global Model	Accuracy	Precision	Recall	F1-Score
Cycle 1 (GMC1)	99.6700%	99.6713%	99.6700%	99.6702%
Cycle 2 (GMC2)	99.8880%	99.8881%	99.8880%	99.8879%

TABLE 19. Metrics Comparison of GMC1 and GMC2 on Survival Analysis Dataset

Global Model	Accuracy	Precision	Recall	F1-Score
Cycle 1 (GMC1)	95.9780%	96.6189%	95.9780%	95.9365%
Cycle 2 (GMC2)	96.6300%	97.0899%	96.6300%	96.5926%

4) Comparison with State-of-the-Art Federated IDS

To position Fed-CALiBER within the existing landscape of federated IDS for vehicular networks, we compare its performance against the results reported by Zhang et al.[16], who also employed a federated Transformer-based approach. Table 20 and fig. 19 presents this comparison, considering relevant metrics and dataset characteristics. The comparison was based on the Car Hacking Dataset, as it is one of the datasets used in both experiments.

TABLE 20. Metrics Comparison between Fed-CALiBER and existing method on Car Hacking Dataset

	Accuracy	Precision	Recall	F1-Score
Zhang et al. (2025)	99.7552%	99.7645%	99.7552%	99.7596%
Fed-CALiBER	99.9840%	99.9840%	99.9840%	99.9840%

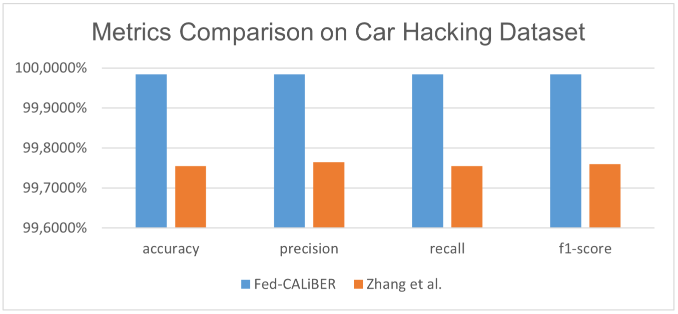


FIGURE 19. Fed-CALiBER and existing method Comparison Graph

5) Cross-Dataset Generalization Analysis of FL Models vs Standalone

To further quantify the benefits of federated learning for generalization, table 21 to 23 and fig. 20 to 22 compares the performance of the global model $GMC2$ against standalone lightweight BERT models (trained using Raspberry Pi on only one specific dataset) when evaluated on all three test datasets (A, B, and C). For instance, when tested on Dataset A, $GMC2$ achieved an F1-score of 99.9840%, whereas standalone models trained only on Dataset B and Dataset C achieved 77.5061% and 93.1838% respectively. Similar trends were observed for tests on Datasets B and C. These results consistently show that the federated global model $GMC2$ exhibits superior generalization across diverse, unseen dataset distributions compared to models trained in isolation on a single data source, highlighting FL's strength in creating more robust and widely applicable IDS solutions.

TABLE 21. Cross Dataset Testing on Car Hacking Dataset (Dataset A)

	Accuracy	Precision	Recall	F1-Score
Global Model	99.9840%	99.9840%	99.9840%	99.9840%
Model B	83.9060%	72.3761%	83.9060%	77.5061%
Model C	93.7000%	94.3298%	93.7000%	93.1838%

TABLE 22. **Cross Dataset Testing on Hacking Challenge (Dataset B) Dataset**

	Accuracy	Precision	Recall	F1-Score
Global Model	99.8880%	99.8881%	99.8880%	99.8879%
Model A	78.6820%	65.8354%	78.6820%	71.5313%
Model C	78.3780%	65.8948%	78.3780%	71.3972%

TABLE 23. **Cross Dataset Testing on Survival Analaysis Dataset (Dataset C)**

	Accuracy	Precision	Recall	F1-Score
Global Model	96.6300%	97.0899%	96.6300%	96.5926%
Model A	90.5580%	92.7991%	90.5580%	90.6849%
Model B	79.2380%	66.1628%	79.2380%	71.8611%

C. RESOURCE USAGE AND EFFICIENCY OF FED-CALIBER

A key motivation for Fed-CALiBER is the development of a lightweight and efficient IDS. The lightweight BERT model has a disk size of approximately 13MB (for .safetensors, huggingface files), reflecting its compact parameterization. Inference speed evaluations conducted on a Raspberry Pi 5 client demonstrated an average throughput of 3-4ms per sample with a batch size of 16. During federated local training rounds on the Raspberry Pi clients, CPU utilization typically ranged between 60% and 70%, with the Python process consuming an average of 0.6 GB of RAM, well within the device's 8GB capacity. The average duration for a full experimental run of two cycles of FL (which consists of 10 communication round), including local client training for an epoch and server aggregation, was approximately 7.5 minutes. This rapid round time makes the system highly viable for frequent, periodic model updates (such as multiple times per day or daily) to quickly adapt to new data, while simultaneously having no need to be instantaneous but should occur regularly to maintain security posture.

D. DISCUSSION

The experimental results presented in this study demonstrate the viability and effectiveness of the proposed Fed-CALiBER framework for CAN bus intrusion detection. Our

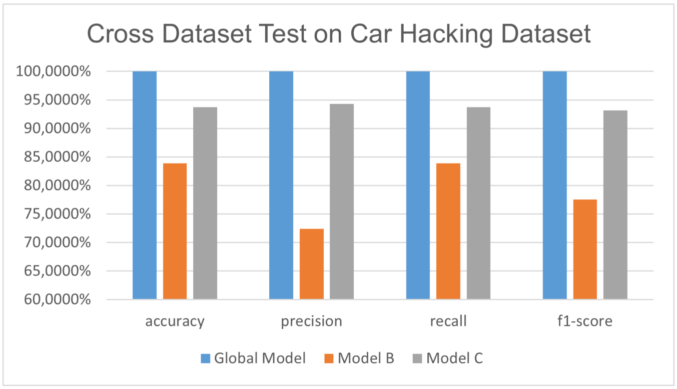


FIGURE 20. **Cross Dataset Testing on Car Hacking Dataset (Dataset A) Comparison Graph**

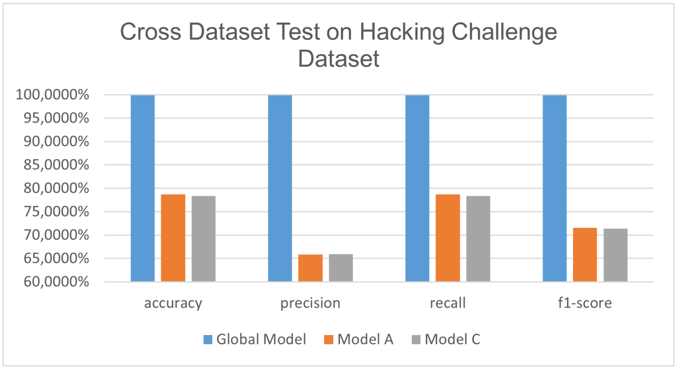


FIGURE 21. **Cross Dataset Testing on Hacking Challenge Dataset (Dataset B) Comparison Graph**

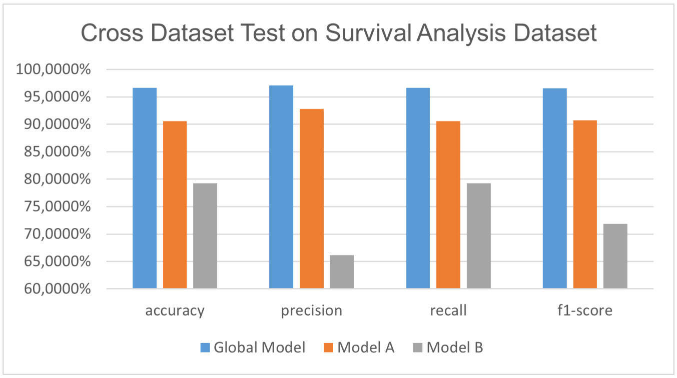


FIGURE 22. **Cross Dataset Testing on Survival Analysis Dataset (Dataset C) Comparison Graph**

proposed lightweight BERT model, when trained in a centralized manner, established a strong baseline, achieving near-perfect detection scores on individual datasets when ample resources (server laptop, full dataset portions) were available. Performance remained high even when trained on resource-constrained Raspberry Pi devices using significantly smaller 20,000-row data subsets, although a predictable performance dip was observed compared to laptop-based training with more data. Interestingly, for Dataset A, the Raspberry Pi standalone model occasionally outperformed its laptop-trained counterpart, potentially due to the smaller training subset inadvertently excluding particularly noisy or challenging samples. However, the crucial limitation of these standalone models was their significantly reduced cross-dataset generalization, underscoring the inherent challenges posed by the Non-IID nature of real-world vehicular data.

The introduction of Federated Learning via Fed-CALiBER, particularly through our two-cycle experimental design, showcased clear benefits. The global model after the first cycle ($GMC1$) demonstrated an initial ability to generalize by learning from diverse client specializations. The second cycle, which involved clients training on shifted datasets, led to the $GMC2$ model. $GMC2$ generally exhibited improved robustness and superior average generalization across Datasets B and C compared to $GMC1$, indicating successful adaptation and knowledge integration from the varied data exposures. For instance, the F1-score on Dataset C improved from 95.9365% to 96.5926%. Notably, while $GMC2$ showed broad improvements, its performance on Dataset A was slightly lower than $GMC1$'s, a phenomenon potentially attributable to catastrophic forgetting of some Dataset A specifics as the model adapted to the other datasets, or the inherent complexities of achieving uniform peak performance across all three distinct distributions simultaneously. Nevertheless, $GMC2$ consistently outperformed standalone models in cross-dataset evaluations, highlighting FL's strength in mitigating the negative impacts of Non-IID data.

The resource usage metrics further validate the "lightweight" aspect of Fed-CALiBER. With a model size of approximately 13MB and efficient inference speeds observed on Raspberry Pi clients, the system demonstrates suitability for edge deployment. Furthermore, the federated approach inherently provides a foundational layer of data privacy by ensuring raw CAN messages remain localized on client devices.

It is acknowledged, however, that this study has specific scope limitations. The proposed lightweight BERT model is fundamentally specialized for the CAN bus protocol, as its entire knowledge base was built during pre-training on normal CAN messages. While the architectural design is hardware-agnostic and portable to other capable embedded platforms, applying this methodology to other IVN protocols like Automotive Ethernet or FlexRay would require a new, dedicated pre-training phase with data from those specific protocols.

The Raspberry Pi was chosen in this study to act as a representative proxy for a resource-constrained edge device, confirming the feasibility of Fed-CALIBER's lightweight design on low-power hardware. While the current simulation with three clients and three distinct datasets provides valuable insights into FL's potential for CAN IDS, scaling these experiments to a larger, more diverse fleet of hundreds or thousands of vehicles introduces further challenges including the increased communication burden on the central server, the need for robust client management and scheduling, and handling real-world issues like intermittent client connectivity (client dropouts) and varying network latency, making it a promising avenue for future investigation to further validate the framework's practical applicability. These considerations are critical for the transition from a simulated environment to a production-level system and are highlighted as key directions for future work.

While a foundational privacy benefit of FL is data localization, it is important to acknowledge that the transmission of model updates is not without risks. The framework, in its current form, is vulnerable to advanced attacks. For instance, an adversary could potentially perform inference attacks by analyzing a client's model updates to deduce information about the private training data used. Furthermore, the conducted experiment assumes "honest" participants (no rogue, compromised client) and does not explicitly defend against model poisoning attacks, where a malicious client could intentionally send corrupted updates to degrade the global model's performance or insert a backdoor.

To address these vulnerabilities, future work could integrate established Privacy-Enhancing Technologies (PETs). Secure Aggregation protocols could be applied during the server-side aggregation step, using cryptographic techniques to allow the server to compute the sum of client updates without decrypting any individual update. On the client-side, Differential Privacy could be implemented by adding a precisely calibrated amount of statistical noise to model parameters before transmission, providing mathematical guarantees against inference attacks.

To enhance the defense even further, Secure Aggregation could be applied during the server-side aggregation step. In this scheme, clients would encrypt their model updates, and the server would only be able to compute the sum of these updates without decrypting individual contributions. Differential Privacy could also be applied on the client-side. By implementing it, client would add a precisely calibrated amount of statistical noise to its parameters before the client transmits its model update to the server to be aggregated.

The observed performance dynamics in our experiments, particularly the catastrophic forgetting on Dataset A during Cycle 2, are characteristic of client drift, a known challenge when applying standard Federated Averaging (FedAvg) to Non-IID data. Client drift occurs as local models diverge towards their skewed local optima, leading to a sub-optimal aggregated global model. More robust FL algorithm such as FedProx and SCAFFOLD should be explored to address this matter. The FedProx algorithm [28] addresses this by adding a proximal term to the client loss function, which regularizes local training and keeps client models closer to the global consensus, directly counteracting the 'tug-of-war' effect. Alternatively, Scaffold [29] uses control variates to correct for this drift by estimating the update directions of both the client and server, leading to more stable convergence in heterogeneous settings. Implementing and comparing these advanced algorithms against our FedAvg baseline is a suggested next step for developing a more robust and consistently high-performing federated IDS.

VI. CONCLUSION

In this paper, we introduced Fed-CALiBER, a federated learning framework leveraging a novel lightweight BERT model for intrusion detection in automotive CAN bus networks. Our approach successfully addresses critical challenges of data privacy by keeping raw data localized, utilize resource constrained edge devices through its compact architecture, and tackles the Non-IID nature of data from distributed vehicles via collaborative learning. Experimental results demonstrated that the proposed lightweight BERT achieves high performances, and the federated learning process, particularly through our two-cycle design, enables significant model improvement and adaptation to shifting data distributions. This resulted in a global model with enhanced robustness and superior average performance across diverse datasets compared to standalone models. The system's efficiency and suitability for edge deployment were validated on Raspberry Pi clients, which exhibited low resource consumption and practical inference speeds.

The primary contributions of this work include the design and evaluation of a compact Transformer architecture specifically tailored for CAN messages, combined with a unique two-cycle federated learning experimental setup that demonstrates both initial generalization and continual learning capabilities when client data distributions evolve. Fed-CALiBER thus offers a promising direction for developing scalable, privacy-preserving, and resource-efficient IDS for autonomous vehicle in-vehicle networks.

Future work will focus on several key advancements to build upon the foundation established by Fed-CALiBER. A primary direction is the extension of this methodology to other critical In-Vehicle Network (IVNs) protocols. While the current model is an expert on CAN bus, new pre-training and fine-tuning cycles could be conducted on datasets from Automotive Ethernet or FlexRay to create specialized lightweight models for those domains, eventually leading to a multi-protocol federated IDS. Secondly, the security and robustness of the federated process itself can be enhanced by exploring advanced FL algorithms such as FedProx or Scaffold, to better handle statistical heterogeneity, and by integrating privacy-enhancing technologies such as Secure Aggregation Protocol on server-side, or Differential Privacy on client-side, to defense against model poisoning attacks from potential rogue clients. A crucial step towards real-world application involves scaling the experiments to a larger, more heterogeneous fleet of clients and, ultimately, validating the end-to-end framework on automotive-grade hardware integrated directly into a physical vehicle. Additionally, to better contextualize the performance gains, a future comparative analysis should include strong, non-deep-learning baselines. Evaluating the proposed model against traditional machine learning models such as Gradient Boosting (e.g., XGBoost) or Support Vector Machines (SVM), trained on the same feature sets, to provide a more complete picture of its performance within the broader machine learning landscape.

REFERENCES

[1] A. Anwar, A. Anwar, L. Moukahal, and M. Zulkernine, "Security assessment of in-vehicle communication protocols," *Vehicular Communications*, vol. 44, p. 100 639, 2023. DOI: 10.1016/j.vehcom.2023.100639.

[2] S. Lee, W. Choi, I. Kim, G. Lee, and D. Hoon Lee, "A comprehensive analysis of datasets for automotive intrusion detection systems," *Computers, Materials & Continua*, vol. 76, no. 3, pp. 3413–3442, 2023. DOI: 10.32604/cmc.2023.039583.

[3] M. L. Ali, K. Thakur, S. Schmeelk, J. Debello, and D. Dragos, "Deep learning vs. machine learning for intrusion detection in computer networks: A comparative study," *Applied Sciences*, vol. 15, no. 4, p. 1903, 2025.

[4] W. Lo, H. Alqahtani, K. Thakur, A. Almadhor, S. Chander, and G. Kumar, "A hybrid deep learning based intrusion detection system using spatial-temporal representation of in-vehicle network traffic," *Vehicular Communications*, vol. 35, p. 100 471, 2022. DOI: 10.1016/j.vehcom.2022.100471.

[5] M. Althunayyan, A. Javed, and O. Rana, "A robust multi-stage intrusion detection system for in-vehicle network security using hierarchical federated learning," *Vehicular Communications*, vol. 49, p. 100 837, 2024. DOI: 10.1016/j.vehcom.2024.100837.

[6] S. Corrigan, *Introduction to the controller area network (CAN) (rev. b)*, 2002. [Online]. Available: https://www.rpi.edu/dept/ecse/mps/sloa101.pdf.

[7] B. Lampe and W. Meng, "Can-train-and-test: A curated CAN dataset for automotive intrusion detection," *Computers & Security*, vol. 140, p. 103 777, 2024. DOI: 10.1016/j.cose.2024.103777.

[8] L. Zhang, X. Yan, and D. Ma, "A binarized neural network approach to accelerate in-vehicle network intrusion detection," *IEEE Access*, vol. 10, pp. 123 505–123 520, 2022. DOI: 10.1109/ACCESS.2022.3208091. [Online]. Available: https://ieeexplore.ieee.org/document/9895417/.

[9] H. J. Jo and W. Choi, "A Survey of Attacks on Controller Area Networks and Corresponding Countermeasures," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 6123–6141, Jul. 2022. DOI: 10.1109/TITS.2021.3078740. [Online]. Available: https://ieeexplore.ieee.org/document/9439954/.

[10] V. Cobilean, H. S. Mavikumbure, C. S. Wickramasinghe, B. J. Varghese, T. Pennington, and M. Manic, "Anomaly Detection for In-Vehicle Communication Using Transformers," in *IECON 2023- 49th Annual Conference of the IEEE Industrial Electronics Society*, Singapore, Singapore, Oct. 2023, pp. 1–6. DOI: 10.1109/IECON51785.2023.10311788.

[11] E. Nwafor and H. Olufowobi, "CANBERT: A Language-based Intrusion Detection Model for In-vehicle Networks," in *2022 21st IEEE International Conference on Machine Learning and Applications (ICMLA)*, Nassau, Bahamas, Dec. 2022, pp. 294–299. DOI: 10.1109/ICMLA55696.2022.00048.

[12] H. Jo and D.-H. Kim, "Intrusion Detection Using Transformer in Controller Area Network," *IEEE Access*, vol. 12, pp. 121 932–121 946, 2024. DOI: 10.1109/ACCESS.2024.3452634.

[13] N. Alkhatib, M. Mushtaq, H. Ghauch, and J.-L. Danger, "CAN-BERT do it? Controller Area Network Intrusion Detection System based on BERT Language Model," in *2022 IEEE/ACS 19th International Conference on Computer Systems and Applications (AICCSA)*, Abu Dhabi, United Arab Emirates, Dec.

2022, pp. 1–8. DOI: 10.1109/AICCSA56895.2022.10017800.

[14] M. Fu, P. Wang, M. Liu, Z. Zhang, and X. Zhou, "IoV-BERT-IDS: Hybrid Network Intrusion Detection System in IoV Using Large Language Models," *IEEE Transactions on Vehicular Technology*, pp. 1–13, 2024. DOI: 10.1109/TVT.2024.3402366.

[15] Q. Guan et al., "Transformer model with multi-type classification decisions for intrusion attack detection of track traffic and vehicle," in *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2024, pp. 4510–4514. DOI: 10.1109/ICASSP48485.2024.10447002. [Online]. Available: https://ieeexplore.ieee.org/document/10447002/.

[16] Y. Zhang, J. Song, Y. Sun, Z. Gao, Z. Hu, and M. Sun, "Federated two-stage transformer-based network for intrusion detection in non-iid data of controller area networks," *Cybersecurity*, vol. 8, no. 1, p. 29, 2025.

[17] A. Taneja and G. Kumar, "Attention-CNN-LSTM based intrusion detection system (ACL-IDS) for in-vehicle networks," *Soft Computing*, vol. 28, no. 23-24, pp. 13 429–13 441, Dec. 2024. DOI: 10.1007/s00500-024-10313-0.

[18] M. Driss, I. Almomani, Z. E Huma, and J. Ahmad, "A federated learning framework for cyberattack detection in vehicular sensor networks," *Complex & Intelligent Systems*, vol. 8, no. 5, pp. 4221–4235, 2022. DOI: 10.1007/s40747-022-00705-w.

[19] M. H. Bhavsar, Y. B. Bekele, K. Roy, J. C. Kelly, and D. Limbrick, "FL-IDS: Federated learning-based intrusion detection system using edge devices for transportation IoT," *IEEE Access*, vol. 12, pp. 52 215–52 226, 2024. DOI: 10.1109/ACCESS.2024.3386631.

[20] Y. Liu et al., "RoBERTa: A robustly optimized BERT pretraining approach," no. arXiv:1907.11692, Jul. 26, 2019. DOI: 10.48550/arXiv.1907.11692. arXiv: 1907.11692[cs]. [Online]. Available: http://arxiv.org/abs/1907.11692.

[21] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*, PMLR, 2017, pp. 1273–1282.

[22] A. Paszke, "Pytorch: An imperative style, high-performance deep learning library," *arXiv preprint arXiv:1912.01703*, 2019.

[23] T. Wolf et al., "Huggingface's transformers: State-of-the-art natural language processing," *arXiv preprint arXiv:1910.03771*, 2019.

[24] D. J. Beutel et al., "Flower: A friendly federated learning research framework," *arXiv preprint arXiv:2007.14390*, 2020.

[25] Hacking and C. R. Lab, *Hcrl datasets*, Accessed: 25 November 2024, 2024. [Online]. Available: https://ocslab.hksecurity.net/Datasets.

[26] H. Zhang et al., "Effectiveness of pre-training for few-shot intent classification," *arXiv preprint arXiv:2109.05782*, 2021.

[27] T. Zhang, F. Wu, A. Katiyar, K. Q. Weinberger, and Y. Artzi, "Revisiting few-sample bert fine-tuning," *arXiv preprint arXiv:2006.05987*, 2020.

[28] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *Proceedings of Machine learning and systems*, vol. 2, pp. 429–450, 2020.

[29] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, "Scaffold: Stochastic controlled averaging for federated learning," in *International conference on machine learning*, PMLR, 2020, pp. 5132–5143.

HAMMAN ARYO BIMMO received the bachelor's degrees in informatics engineering from Telkom University, Bandung, in 2017. He is currently pursuing master's degree with the School of Electrical Engineering and Informatics, Bandung Institute of Technology (ITB). His current research interests include cybersecurity and autonomous vehicle.

BUDI RAHARDJO (Member, IEEE) received the B.S. degree in electrical engineering from Bandung Institute of Technology (ITB), Indonesia, in 1986, and the M.S. and Ph.D. degrees in computer science from the University of Manitoba, Winnipeg, MB, Canada, in 1990 and 1997, respectively. He is currently a Researcher and a Lecturer with the School of Electrical Engineering and Informatics, ITB. His current research interests include computer security and cryptography. He is the Founder and the Chairperson of Indonesia Computer Emergency Response Team (ID-CERT). He is one of the founders of Asia–Pacific Computer Emergency Response Team (AP-CERT).

• • •