

CANDOR: A Disentangled Deep Learning Architecture for Robust and Lightweight Intrusion Detection in Consumer Automotive Electronics

Hannah B. Pasandi ^{1,*}, Poma Panezai ², and Bushra Qayyum ³

¹ Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, CA, USA

² Department of Computer Science, Balochistan University of Information Technology, Engineering and Management Sciences, Quetta, Pakistan

³ Independent researcher

Email: h.pasandi@berkeley.edu (H.B.P.); poma.panezai@buitms.edu.pk (P.P.); bushra.qayyum@yahoo.com (B.Q.)

*Corresponding author

Manuscript received September 17, 2026; accepted October 12, 2026; published October 27, 2026

Abstract—Learned intrusion detectors for the in-vehicle Controller Area Network score well on clean benchmarks, yet a small change to the eight payload bytes of a frame can blind them. In our measurements on the Real Oak Ridge National Laboratory Automotive Dynamometer (ROAD) benchmark, a payload-only perturbation cuts the F1-Score of a recurrent baseline detector from 0.63 to 0.13. We trace this failure to the representation rather than to data or model size. Joint detectors mix frame timing and payload into one feature vector, so an attacker who controls only the payload moves the whole vector. Timing behaves differently, because bus arbitration and each sender's transmission schedule constrain when frames appear. We present CANDOR, a detector that keeps the two feature groups in separate branches and applies adversarial training only to the payload branch, the part an attacker can reach. A short analysis shows that the worst-case output change of this design depends only on the payload encoder and the fusion head. On ROAD, CANDOR keeps 96.4% of its clean F1-Score under a constrained projected-gradient attack, 4.7 times the retention of the strongest baseline, and on the Car-Hacking dataset it keeps 99.3%. The model has 78,432 parameters and classifies a frame in 4.8 microseconds on a Raspberry Pi 4. Masquerade traffic, which preserves the victim's timing, remains the open weakness, and we outline how physical-layer sender fingerprints can close it.

Keywords—controller area network, in-vehicle networks, intrusion detection, adversarial robustness, disentangled representation, embedded network security

I. INTRODUCTION

A passenger car's powertrain and chassis Electronic Control Units (ECUs) exchange commands over a Controller Area Network (CAN) bus that broadcasts every frame to every node and carries no sender authentication [1, 2]. Any node that reaches the bus can therefore inject frames that other ECUs accept as genuine. Koscher *et al.* [3] used such frames to disable the brakes and stop the engine of a production car. Checkoway *et al.* [4] reached the bus remotely through Bluetooth and cellular interfaces, and Miller and Valasek [5] sent CAN messages to an unmodified vehicle over its cellular connection. Learned Intrusion Detection Systems (IDSs) are the usual answer, and several report near-perfect accuracy on public captures [6, 7]. What breaks is the assumption behind those scores, namely that the attacker does not shape frames against the detector. In our measurements on the Real Oak Ridge National Laboratory Automotive Dynamometer (ROAD) benchmark [8], a bounded change to the eight payload bytes of each attack

frame lowers the F1-Score of a stacked Long Short-Term Memory (LSTM) detector from 0.632 to 0.131.

Two changes make this gap pressing now. Detectors are moving from offline analysis into vehicle hardware. Lightweight Intrusion Detection System (LIDS), for example, runs on a node with 520 kB of memory and a 240 MHz processor [9], and connectivity for connected and automated vehicles stays intermittent outside cities [10], so the in-vehicle detector often has to decide alone. Evasion has also moved from images to CAN traffic. Aloraini *et al.* [11] lowered the F1-Score of an in-vehicle detector from 95% to 38% with a black-box attack trained on data read from the diagnostic port. Longari *et al.* [12] injected precomputed evasive payloads at the right moment in bus traffic, and Hasan *et al.* [13] report evasion rates up to 87.9% with protocol-constrained gradient descent in a study that includes CICIoV2024.

The failure has a precise cause. A CAN frame offers two kinds of evidence, and they differ in who controls them. Payload bytes are written by software, so an attacker who can transmit can set them to any value the byte range allows. Timing is different. Arbitration on the shared bus and each ECU's transmission schedule decide when frames appear [14]. An injecting node can add frames, but it cannot delete the legitimate sender's frames, and imitating that sender's cadence and clock skew requires control of a transmitting ECU [15]. Deep CAN detectors such as CANintelliIDS [16] and recurrent models [17] feed both kinds of evidence into one encoder, so a payload-only perturbation moves the entire learned representation, including the part that summarizes timing. The key tension is that payload semantics carry much of the signal for spoofing, yet they are exactly the features the attacker owns.

The obvious fixes do not remove this coupling. Standard adversarial training [18] perturbs every input coordinate, so it spends its budget on timing coordinates that the attacker cannot move, and it pushes clean and adversarial gradients through the same shared weights. Timing-only detectors [19, 20] ignore the payload and miss fabrication that keeps the bus cadence, and a compromised ECU can emulate the clock skew they rely on [15]. Voltage fingerprints identify the true sender [21–23], but they need an analog front end on the monitoring node. Defensive distillation [24] looked promising until stronger optimization attacks defeated it [25].

So, what should the detector do? We separate the evidence

the attacker can reach from the evidence it cannot, and we harden only the reachable part. Fig. 1 contrasts the two designs on one attack frame. In the joint detector of Fig. 1(a), the perturbation enters one shared encoder, every coordinate of the embedding can move, and the label flips. In CANDOR, shown in Fig. 1(b), the timing view sits inside the bus schedule and outside the attacker's reach, so its embedding z_t stays fixed while only the hardened payload embedding z_p absorbs the perturbation. A short analysis (Theorem 1) makes this precise. Under a payload-only attacker, the worst-case change of the classifier output depends only on the Lipschitz constants of the payload encoder and the fusion head. With the constants we estimate on ROAD, the bound for CANDOR is 2.3 times tighter than for a joint detector.

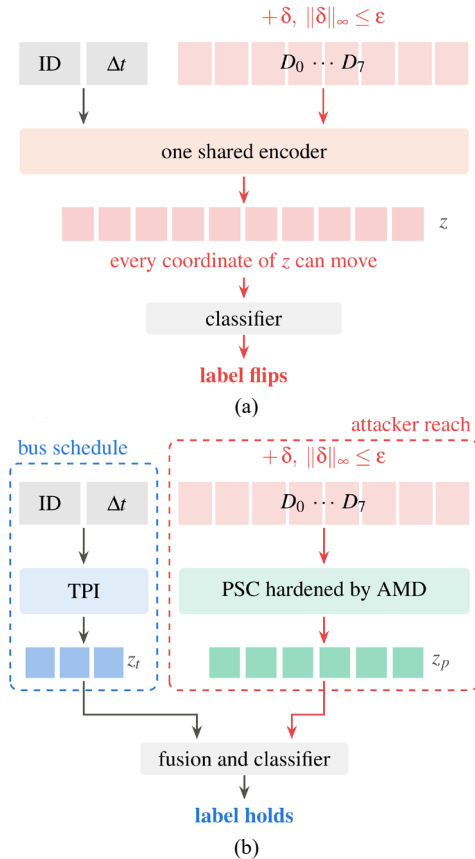


Fig. 1. (a) Joint detector vs (b) CANDOR under the same payload-only perturbation δ . ID = arbitration identifier; TPI = Temporal Pattern Indexer; PSC = Payload Semantic Classifier; AMD = Adversarial Minimax Disentangler.

CANDOR has three parts, drawn in Fig. 2. The Temporal Pattern Indexer (TPI) reads windows of arbitration identifiers and inter-arrival times. The Payload Semantic Classifier (PSC) reads the eight payload bytes with self-attention. The Adversarial Minimax Disentangler (AMD) perturbs payloads during training and stops their gradients before they reach TPI. Two lessons from building it shaped the final design. We first fused the branches with a gated multiplicative layer, which lowered adversarial F1 on ROAD by 1.7 points against plain layer-normalized concatenation, so we kept the simpler fusion. We also learned that the attack must clamp perturbed bytes to the valid range after every step. Without the clamp, adversarial examples left the byte range a real frame can carry, clean F1 rose by about four points, and the robustness numbers became misleading.

We evaluate CANDOR on three public CAN datasets. On ROAD, CANDOR keeps 96.4% of its clean F1-Score under constrained Projected Gradient Descent (PGD), while the strongest baseline keeps 20.7%, a 4.7-fold difference. On the Car-Hacking dataset [26, 27], CANDOR keeps 99.3% against 47.5%. The model has 78,432 parameters and needs a median of 4.8 μ s per frame on a Raspberry Pi 4, far below the 222 μ s that separate back-to-back frames on a saturated 500 kbit/s bus. Our contributions are listed below.

- We identify the joint timing and payload representation as the cause of adversarial brittleness in learned CAN detectors, and we bound the output change of a disentangled detector under a payload-only attacker (Section III).
- We design CANDOR, which confines adversarial training to the payload branch and keeps the timing branch free of adversarial gradients (Section IV).
- We evaluate clean detection, adversarial retention, a simulated cross-vehicle shift, and component ablations on ROAD, Car-Hacking, and CICIoV2024, and we report where CANDOR loses, which is on masquerade traffic (Section V).
- We place the design in the context of in-vehicle network monitoring, covering the frame-time budget, the protocol layering behind the feature split, and a concrete path to masquerade detection through physical-layer fingerprints (Section VI).

We do not claim certified robustness. Our Lipschitz constants are empirical estimates, and our attacks are white-box PGD rather than adaptive attacks designed against the two-branch layout [28]. Our attacker controls payload bytes but not frame timing. An attacker who controls a transmitting ECU can shape timing [15], and the right defense against that attacker is firmware integrity rather than traffic monitoring. All results are frame-level classifications on recorded traffic, not alarms raised on a running vehicle.

II. RELATED WORK

Prior work has measured the CAN threat, built detectors from timing or from frame content, and more recently attacked those detectors. None of it separates the features an attacker controls from those it does not, which is the gap CANDOR addresses.

A. Attacks and Countermeasures on the CAN Bus

Experimental work established that CAN has no built-in defense against injection. Koscher *et al.* [3] showed that an attacker on almost any ECU can override safety-critical functions, and Checkoway *et al.* [4] mapped remote entry points through diagnostic tools, media players, Bluetooth, and cellular links. Bozdal *et al.* [2] tied these attacks to the missing encryption and authentication in the protocol and reviewed countermeasures with a focus on IDSs. Surveys by Wu *et al.* [29] and Lokman *et al.* [30] organize IDS designs by detection approach, deployment point, and in-vehicle constraints. Tavasoli *et al.* [10] show that data communication for connected and automated vehicles faces coverage, reliability, and security gaps outside cities, which strengthens the case for detectors that run on the vehicle itself.

B. Timing and Sender Fingerprints

Early detectors watched message timing. Song *et al.* [19] flagged injection from the time intervals between CAN messages, and Müter and Asaj [31] used the entropy of bus traffic. Marchetti *et al.* [32] found that entropy computed over all messages detects only high-volume injection, while low-volume attacks need one detector per message class. Cho and Shin [20] fingerprinted ECUs by the clock skew in periodic messages, and Sagong *et al.* [15] then showed that a compromised ECU can emulate another ECU's skew by adjusting its transmission delay. Voltage fingerprints resist that emulation because they come from the transceiver's analog behavior [21–23, 33]. CANDOR uses timing only as an anchor that the payload attacker cannot move, and Section V-G returns to voltage as the missing third view.

C. Learned Content Detectors

Kang [34] trained a deep network on simulated CAN packets. Taylor *et al.* [17] predicted the next data word per sender with an LSTM, Seo *et al.* [27] trained a generative adversarial network on normal traffic only, and Song *et al.* [26] applied a deep convolutional network to the Car-Hacking captures. CANet and CANShield work on decoded signals with unsupervised models [35, 36], and X-CANIDS adds explanations at the signal level [37]. CAN-ADF pairs rules with recurrent networks [38], CANintelliIDS combines a convolutional extractor with an attention-based recurrent unit [16], MTH-IDS stacks tree ensembles with clustering for unseen attacks [6], and Lo *et al.* [39] build a spatial-temporal representation of in-vehicle traffic. LIDS shows that a filter in front of a small perceptron fits a 520 kB node [9]. The closest design in spirit is the hybrid detector of Tavasoli *et al.* [40], which combines per-identifier clustering with a byte-sequence transformer for rural vehicles. It weighs the two views by confidence and targets intermittent connectivity, whereas CANDOR fuses learned representations and targets adversarial payloads. AIEisa *et al.* [7] combine LSTM models for in-vehicle and vehicle-to-infrastructure traffic, while detectors for consumer and vehicular networks at the flow level [41, 42] do not model individual frame payloads. None of these designs is evaluated against payload perturbations crafted for the detector.

D. Adversarial Robustness

Goodfellow *et al.* [43] introduced the fast gradient sign method, and Madry *et al.* [18] turned multi-step PGD into an adversarial training recipe. Carlini and Wagner [25] built stronger optimization attacks and used them to break defensive distillation, and Zhang *et al.* [44] formalized the trade-off between robustness and accuracy. Feature squeezing detects adversarial inputs by comparing predictions before and after reducing input precision [45], a test designed for images that treats every input coordinate alike. Pierazzi *et al.* [46] and Apruzzese *et al.* [47] argue that attacks on security classifiers must respect what the attacker can change in the problem domain. In vehicles, Aloraini *et al.* [11], Longari *et al.* [12], and Hasan *et al.* [13], evade CAN detectors, and Aliyu *et al.* [48] detect adversarial

examples statistically in a federated in-vehicle IDS. These defenses harden or filter the whole input. CANDOR instead restricts hardening to the subspace the attacker can reach.

III. THREAT MODEL AND ROBUSTNESS ANALYSIS

We state the attacker first, because each design choice in Section IV follows from what that attacker can and cannot change.

A. Threat Model

Definition 1 (CAN adversary). Let a frame be $x = (x_t, x_p) \in \mathbb{R}^{d_t} \times \mathbb{R}^{d_p}$, where x_t holds timing and identifier features and x_p holds the payload bytes scaled to $[0,1]$. An adversary $\mathcal{A}_\varepsilon$ produces $\tilde{x} = (x_t, x_p + \delta)$ with $\|\delta\|_\infty \leq \varepsilon$ and $x_p + \delta \in [0,1]^{d_p}$.

The definition follows from how the bus works. Timing is set by arbitration and by each ECU's transmission schedule [14], so a node that injects frames cannot remove the legitimate sender's frames or reorder them. Precise imitation of a sender's cadence requires a compromised ECU [15], and we place that attacker out of scope. Payload bytes have no such guard, and the bus carries any value within the byte range. The attacker knows the model and its weights. For classical CAN frames $d_p = 8$, and we use $\varepsilon = 0.3$ on the scaled byte range¹. The datasets also contain masquerade traffic, in which the legitimate sender's frames are suppressed and replaced. We evaluate how well CANDOR detects it on clean traffic, but robustness to a timing-capable attacker is outside Definition 1.

B. Robustness Bound

Why does separation help, beyond intuition? Write the disentangled classifier as $f_\theta(x) = g(h_t(x_t), h_p(x_p))$, where the timing encoder h_t , the payload encoder h_p , and the fusion head g have Lipschitz constants L_t , L_p , and L_g in the ℓ_2 norm. A joint classifier f_j has a single constant L_j over the full input.

Theorem 1 (Disentangled robustness bound). Under $\mathcal{A}_\varepsilon$,

$$\|f_\theta(\tilde{x}) - f_\theta(x)\|_2 \leq L_g L_p \sqrt{d_p} \varepsilon, \quad (1)$$

whereas a joint classifier satisfies

$$\|f_j(\tilde{x}) - f_j(x)\|_2 \leq L_j \sqrt{d_p} \varepsilon. \quad (2)$$

The disentangled bound is tighter whenever $L_g L_p < L_j$.

Proof. The adversary leaves x_t unchanged, so $h_t(x_t)$ is unchanged. Because $\|\delta\|_2 \leq \sqrt{d_p} \|\delta\|_\infty \leq \sqrt{d_p} \varepsilon$, the Lipschitz property of h_p gives $\|h_p(x_p + \delta) - h_p(x_p)\|_2 \leq L_p \sqrt{d_p} \varepsilon$. The input of g changes only in its payload block, so the Lipschitz property of g yields (1). For the joint model, the perturbation $(0, \delta)$ has ℓ_2 norm at most $\sqrt{d_p} \varepsilon$, which yields Eq. (2).

In plain words, both bounds grow with $\sqrt{d_p} \varepsilon$. Separation does not change how the bound scales with the number of attacked bytes. It changes which constants multiply that term.

¹ On the raw byte scale, $\varepsilon = 0.3$ allows a change of up to $0.3 \times 255 \approx 76$ units per byte. We chose this value to cover the largest per-byte deviation we observed in the physical injection traces of ROAD [8].

In CANDOR the constants belong to the payload encoder and the fusion head, which can be regularized without touching the timing encoder, while weight sharing in a joint model ties the payload response to every other feature. With the estimates $L_f = 4.2$ and $L_g L_p = 1.8$ reported in Section V-D, the bounds at $\epsilon = 0.3$ and $d_p = 8$ are 3.56 and 1.53. These constants are empirical estimates rather than certified upper bounds, so Theorem 1 describes the typical sensitivity of the two models, not a worst-case guarantee.

C. Training Objective

CANDOR trains on a clean loss and an adversarial loss restricted to the payload,

$$\min_{\theta} \mathbb{E}_{(x,y)} \left[\mathcal{L}_{\text{CE}}(f_{\theta}(x_t, x_p), y) + \lambda \max_{\delta \in \Delta_{\epsilon}} \mathcal{L}_{\text{CE}}(f_{\theta}(x_t, x_p + \delta), y) \right], \quad (3)$$

where $\mathcal{L}_{\text{CE}}$ is the cross-entropy loss, Δ_{ϵ} is the set of perturbations allowed by Definition 1, and λ weighs robustness against clean accuracy. A K -step PGD with step size $\alpha = \epsilon/K$ solves the inner maximization on x_p alone. Gradients of the adversarial term do not flow into h_t . This is equivalent to projecting every PGD update onto the payload subspace, and the timing encoder trains on clean data only.

Remark 1. Let g_c and g_a be the stochastic gradients of the clean and adversarial terms of Eq. (3) for one parameter block. In joint adversarial training every shared block receives $g_c + \lambda g_a$, whose variance is $\text{Var}(g_c) + \lambda^2 \text{Var}(g_a) + 2\lambda \text{Cov}(g_c, g_a)$. In CANDOR the timing block receives g_c alone. Its update variance is therefore lower whenever $\text{Cov}(g_c, g_a) > -\lambda/2 \text{Var}(g_a)$, that is, unless the two gradients are strongly anti-correlated. The part of the model that the attacker cannot influence then trains without adversarial noise.

IV. DESIGN OF CANDOR

CANDOR is an in-vehicle monitor that receives raw CAN frames and labels each one as benign or as one of the attack classes of the training data. Fig. 2 traces a frame through the system. Stage 1 parses the stream into the identifier, the inter-arrival time Δt , and the payload bytes. Stages 2 to 4 encode the timing view and the payload view in separate lanes, and stage 5 fuses the two embeddings. Stage 6 runs only during training and hardens the payload lane, and Stage 7 is the exported model that runs on the vehicle. The red marker between the lanes shows the rule behind the whole design, namely that adversarial gradients never reach the timing encoder.

A. Temporal Pattern Indexer

TPI receives a sliding window of $W = 32$ consecutive frames. For each frame it uses the inter-arrival time $\Delta t_i = t_i - t_{i-1}$ and maps the arbitration identifier to a learned 16-dimensional embedding. A two-layer LSTM with 64 hidden units reads the window and emits a 48-dimensional embedding z_t . We chose an LSTM over a transformer for this branch because frame order carries physical meaning, since ECUs transmit on fixed cycles and arbitration resolves contention in identifier order. Replacing the LSTM with a two-head transformer encoder lowered clean F1 on ROAD by

3.2%. No perturbation enters this branch during training.

B. Payload Semantic Classifier

PSC receives the eight payload bytes $D_0, \dots, D_7$ of each frame, scaled to $[0,1]$. A four-head self-attention layer with positional encodings produces a 48-dimensional embedding z_p . We use attention rather than a plain perceptron because bytes in one frame are not independent. One signal may span D_0 and D_1 while a related signal sits in D_4 and D_5 , and a perturbation that moves one without the other produces an inconsistency that attention can learn to flag even when every byte stays in range.

C. Adversarial Minimax Disentangler

AMD implements the inner maximization of Eq. (3). For each mini-batch it runs $K = 7$ PGD steps on x_p with $\epsilon = 0.3$ and $\alpha = \epsilon/K$, and it clamps the perturbed payload to $[0,1]$ after every step. The adversarial gradient flows through PSC and is masked before it reaches TPI. We set $\lambda = 0.5$ after a grid search over $\{0.1, 0.3, 0.5, 0.7, 1.0\}$. Values above 0.7 lowered clean F1 noticeably, and values below 0.3 left PSC under-defended.

D. Fusion and Classification

The fusion stage applies layer normalization to z_t and to z_p separately, which corrects for the different scales of the two branches, and concatenates them into a 96-dimensional vector. One fully connected layer with a softmax output maps this vector to the class label. Gated multiplicative fusion and attention-based fusion did not improve adversarial F1 on any dataset, and concatenation adds no parameters.

E. Implementation

The model has 78,432 trainable parameters, 42,880 in TPI and 35,552 in PSC and the fusion head. We export it in the Open Neural Network Exchange (ONNX) format at opset 17 and run it with ONNX Runtime on a Raspberry Pi 4 Model B with 4 GB of memory and an ARM Cortex-A72 processor at 1.5 GHz. The exported file occupies 314 KB.

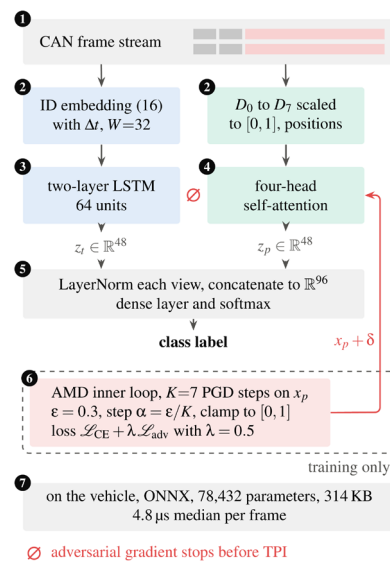


Fig. 2. CANDOR pipeline from raw frames to a class label, with the training-only AMD lane and the on-vehicle deployment. CAN = Controller Area Network; ID = arbitration identifier; LSTM = long short-term memory; TPI = Temporal Pattern Indexer; AMD = Adversarial Minimax Disentangler; PGD = projected gradient descent; CE = cross-entropy; ONNX = Open Neural Network Exchange.

V. EVALUATION

We ask four questions. Does separation cost clean accuracy? How much of the clean F1-Score survives constrained PGD? Does the model hold under a simulated change of vehicle? Which component produces the retention? We then examine the class CANDOR handles worst, masquerade traffic, and its cost on the vehicle.

A. Datasets

Table 1 lists the three datasets and the number of frames

we draw from each. All three contain authentic traffic recorded on real vehicles, and each comes from a single vehicle, so no single dataset tests transfer across makes. ROAD is the hardest because its masquerade captures keep the legitimate timing of the targeted identifier [8]. Car-Hacking injects denial-of-service (DoS), fuzzing, gear, and engine-speed (RPM) spoofing frames into a Hyundai sedan through the diagnostic port [27], and CICIoV2024 records DoS and four spoofing attacks on a stationary 2019 Ford [49].

Table 1. Public CAN datasets used in this study.

Dataset	Vehicle and capture	Attack classes in the release	Frames, benign/attack
ROAD [8]	one undisclosed vehicle, mid-2010s, dynamometer and road driving; masquerade versions made in post-processing	fuzzing, five targeted fabrications, accelerator attack, masquerade versions	25,000/4,000
Car-Hacking [26, 27]	Hyundai YF Sonata, on-road capture; attacks injected through the diagnostic port	DoS, fuzzing, gear spoofing, RPM spoofing	30,000/10,000
CICIoV2024 [49]	2019 Ford with all ECUs intact, stationary during the attacks	DoS, spoofing of gas, RPM, speed, and steering wheel	15,000/8,000

Note: CAN = Controller Area Network; ROAD = Real Oak Ridge National Laboratory Automotive Dynamometer; CICIoV2024 = 2024 Internet of Vehicles dataset of the Canadian Institute for Cybersecurity; ECUs = electronic control units; DoS = denial of service; RPM = revolutions per minute (engine speed).

B. Baselines and Attack

We compare against three detector families. LSTM-IDS is a two-layer stacked LSTM over concatenated frame features, the recurrent design common in CAN detection [17], and we train it with PGD adversarial training on the full input for a fair adversarial comparison. CANintelliIDS combines a one-dimensional convolutional extractor with an attention-based gated recurrent unit [16]. The Random Forest (RF) works on hand-crafted statistics, namely mean inter-arrival time, byte entropy, and identifier frequency. The attack is constrained PGD under Definition 1 with $\epsilon = 0.3$ and $K = 20$ steps, applied to the payload of every test frame. We report F1-Scores and adversarial retention, the ratio of adversarial to clean F1.

C. Clean Detection

Does separation cost accuracy? Table 2 answers no. On Car-Hacking, CANDOR reaches an F1-Score of 0.789 against 0.787 for LSTM-IDS, and on ROAD it reaches 0.657 against 0.632. The random forest trails CANDOR by less than 0.001 on ROAD, which we expect, since its statistics were designed around exactly this kind of traffic. On CICIoV2024, CANDOR reaches 0.533. ROAD scores sit below Car-Hacking scores for both detectors in Table 2, which is consistent with its stealthier masquerade captures. The takeaway is that separating the branches costs no clean accuracy against the deep baselines.

Table 2. Clean F1 and retention under constrained PGD ($\epsilon = 0.3$, $K = 20$)

Detector	Car-Hacking		ROAD	
	F1 $\uparrow$	Retention $\uparrow$	F1 $\uparrow$	Retention $\uparrow$
CANDOR	0.789	99.3%	0.657	96.4%
LSTM-IDS	0.787	47.5%	0.632	20.7%
Ratio		2.1 $\times$		4.7 $\times$

Note: PGD = projected gradient descent; LSTM-IDS = long short-term memory intrusion detection system.

D. Adversarial Retention

Under constrained PGD with $\epsilon = 0.3$ and $K = 20$, LSTM-IDS falls from 0.787 to 0.374 on Car-Hacking and from 0.632 to 0.131 on ROAD, while CANDOR keeps 99.3%

and 96.4% of its clean F1-Score (Table 2). Fig. 3 sets every retention figure side by side. In Fig. 3(a), the two baselines on ROAD keep about a fifth of their clean F1-Score, 20.7% for LSTM-IDS and 20.3% for CANintelliIDS, whereas CANDOR keeps 96.4%. Fig. 3(b) shows the same ordering on Car-Hacking, with a smaller gap because its overt attacks leave strong timing traces that survive any payload perturbation. Sweeping ϵ from 0 to 0.5 on ROAD, both joint baselines collapse once ϵ exceeds 0.1, while CANDOR loses at most 14% of its clean F1 across the whole sweep. The mechanism matches Theorem 1. We estimate the Lipschitz constants empirically on 1,000 random input pairs from the ROAD test set, using power iteration on the Jacobian as in Virmaux and Scaman [50], and obtain $L_f = 4.2$ for the joint model and $L_g L_p = 1.8$ for CANDOR. The perturbation reaches only the payload half of the representation, and the timing half keeps the decision anchored.

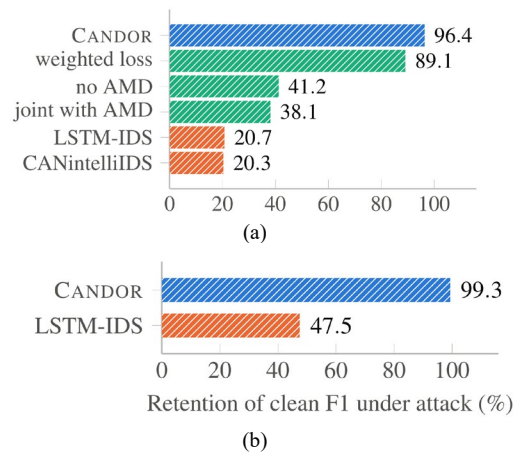


Fig. 3. Retention of clean F1 under constrained PGD, CANDOR in blue, its variants in aqua, baselines in orange. (a) ROAD (b) Car-Hacking. PGD = projected gradient descent; AMD = Adversarial Minimax Disentangler; LSTM-IDS = long short-term memory intrusion detection system.

Fig. 4 plots the bound of Theorem 1 with these two constants. Both bounds grow with ϵ and with $\sqrt{d_p}$, and the joint bound rises 2.3 times faster everywhere. At the operating point of our attack, $\epsilon = 0.3$ with all eight bytes

attacked, the joint model's output may move by up to 3.56 while CANDOR stays within 1.53. The plotted bounds are analytical, so they explain the measured gap in Fig. 3 rather than add evidence to it.

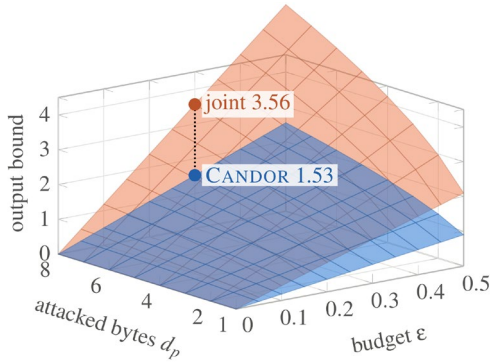


Fig. 4. Theorem 1 bound over budget ϵ and attacked bytes d_p , joint model in orange, CANDOR in blue.

E. Simulated Cross-Vehicle Shift

Fleets span many models, each with its own message dictionary. We simulate cross-vehicle deployment by training on one vehicle profile and testing on another without fine-tuning. CANDOR reaches a Matthews correlation coefficient (MCC) of 0.839 on Car-Hacking and 0.678 on ROAD under this protocol. We attribute much of this to TPI. Periodic transmission and arbitration produce similar inter-arrival statistics across platforms even when payload encodings change, so the timing view transfers better than the payload view. Because the shift is simulated, a test on a second physical vehicle is still needed.

F. Ablation

Table 3 separates the two ingredients. Training a single joint branch with AMD drops retention on ROAD from 96.4% to 38.1%, so adversarial training alone is not enough. Keeping the two branches but removing AMD preserves clean F1 and drops retention to 41.2%, so separation alone is not enough either. Only the combination produces high retention. The lower half of the table records design variants. Two of them, gated fusion and a transformer in TPI, lost accuracy and were dropped. The missing byte clamp inflated clean F1 with adversarial examples no real frame can carry, which is why every result in this paper uses the clamp. The masquerade weighting is the subject of the next subsection.

Table 3. Ablations and design variants of CANDOR on ROAD

Variant	Split	AMD	Effect
CANDOR	✓	✓	retention 96.4%
joint branch with AMD	✗	✓	retention 38.1%
no AMD	✓	✗	retention 41.2%, clean F1 kept
masquerade weighted	✓	✓	retention 89.1%, masquerade F1 0.34 instead of 0.281
gated fusion	✓	✓	adversarial F1 1.7 points lower
transformer in TPI	✓	✓	clean F1 3.2% lower
no byte clamp	✓	✓	clean F1 about 4 points higher, examples not realizable

Note: AMD = Adversarial Minimax Disentangler; TPI = Temporal Pattern Indexer.

² We report the median of 100,000 inference calls after 10,000 warm-up calls, with a batch size of one frame.

G. Masquerade Traffic

Masquerade captures are the hardest class in ROAD, because the attacker's frames replace the legitimate ones at the original cadence. On this class, CANDOR reaches an F1-Score of 0.281, below LSTM-IDS at 0.467 and RF at 0.473. Raising the weight of the masquerade class in the loss lifts its F1-Score to 0.34, but overall retention falls from 96.4% to 89.1% (Table 3), a trade we do not consider worthwhile for most deployments.

Fig. 5 shows why this class is hard for CANDOR and for every detector that reads only bus traffic. We derive each cell from the published attack definitions of the three datasets [8, 27, 49]. DoS, fuzzing, spoofing, and targeted fabrication all add frames, so each leaves a strong trace in timing, the channel TPI reads. Masquerade removes the legitimate frames before injecting its own, so its timing and identifier columns stay at none and its only bus-level trace is a weak payload shift. TPI cannot help by design, and PSC must find a subtle change alone. The last column carries the way out. Every attack in the figure except payload evasion by a compromised sender transmits from a node other than the legitimate one, and that difference appears in the analog voltage of the frame, a channel no branch of CANDOR reads today. Section VI-B turns this observation into a research plan.

	TPI		PSC	unread
DoS flooding	strong	strong	none	strong
fuzzing	strong	strong	strong	strong
spoofing	none	strong	strong	strong
targeted fabrication	none	strong	strong	strong
masquerade	none	none	weak	strong
payload evasion	none	none	weak	weak
	CAN ID	timing	payload	voltage

Fig. 5. Bus channels in which each attack class must leave a trace, with the branch that reads each channel. TPI = Temporal Pattern Indexer; PSC = Payload Semantic Classifier; CAN ID = Controller Area Network identifier; DoS = denial of service.

H. Cost on the Vehicle

Table 4 summarizes the footprint². Is 4.8 μ s fast enough? Fig. 6 compares it with the time that separates two back-to-back frames. A standard frame with eight data bytes occupies 111 bits including the interframe space and up to 135 bits with worst-case bit stuffing [14]³. The dark bars show the shortest possible frame time at each bit rate and the light extensions show the stuffing margin. Even a saturated bus at 1 Mbit/s leaves 111 μ s per frame, 23 times the median latency, and the common 500 kbit/s powertrain rate leaves 46 times. The headroom absorbs operating-system jitter and concurrent work on a shared processor, so CANDOR fits a monitoring node of this class that reads every frame on its segment without sampling. A Raspberry Pi 4 is an application-class processor, however. ECU-class microcontrollers, such as the 240 MHz node that hosts LIDS [9], are far slower, and we have not measured CANDOR on one.

³ The 135-bit figure follows the worst-case transmission time of Davis *et al.* [14] for an 11-bit identifier and eight data bytes, $34 + 64 + 13 + [(34 + 64 - 1)/4] = 135$ bits.

Table 4. Deployment footprint of CANDOR on a Raspberry Pi 4

Quantity	Value
Trainable parameters	78,432
Exported ONNX file	314 KB
Median latency per frame	4.8 μ s
95th percentile latency	6.1 μ s
Frames per second at median latency	about 208,000

Note: ONNX = Open Neural Network Exchange.

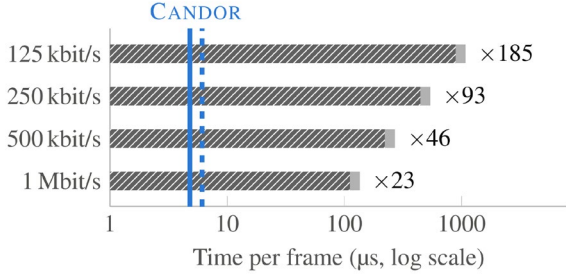


Fig. 6. Frame time of back-to-back 8-byte CAN frames versus CANDOR latency, median solid and 95th percentile dashed. CAN = Controller Area Network.

VI. DISCUSSION

Beyond the benchmark, the results bear on the network, the masquerade gap, attack realism, and fleet deployment.

A. What the Feature Split Means for In-Vehicle Networks

The split in CANDOR follows the layering of the CAN protocol. The arbitration identifier and the transmission time are link-layer and scheduling facts, decided by the communication matrix and by arbitration on the shared medium [1, 14]. The payload is application data. Feature engineering for embedded network IDSs can follow the same rule and group features by the protocol layer that produces them, which is also the rule that decides who can change them. Three network-level consequences follow. First, Theorem 1 scales with $\sqrt{d_p}$, so moving to CAN with flexible data rate, whose frames carry up to 64 payload bytes, raises the payload term from $\sqrt{8} \approx 2.8$ to $\sqrt{64} = 8$, nearly three times more room for the attacker at the same budget. Payload hardening matters more, not less, as payloads grow. Second, CANDOR is a passive monitor. It adds no frames and consumes no payload bytes, unlike authentication schemes that place message codes in the payload [2], so it can run on legacy segments whose ECUs cannot be changed. Third, frame-level F1 is not an alarm rate. At 4,000 frames per second, a false-positive rate of 0.1% produces four alarms per second, so a deployed monitor must aggregate frame decisions per identifier over a time window before alerting. Each bus segment has its own identifiers and schedule, so one model per segment, placed at the gateway that already sees that segment, is the natural deployment point.

B. A Path to Masquerade Detection

Two directions can close the masquerade gap, and both keep the principle of hardening only what the attacker reaches. The first adds a third branch fed by physical-layer sender fingerprints. Voltage-based identification distinguishes ECUs reliably [21–23, 33], and a software attacker cannot change the analog signature of its transceiver, so this branch, like TPI, can stay outside adversarial training. Fig. 5 shows that this channel carries a strong trace for masquerade, the one attack where TPI is blind. Testing it

requires a hardware testbed with an analog tap, because the masquerade captures in ROAD are produced in post-processing and contain no analog traces [8]. The second direction targets the payload itself. A masquerading attacker replaces one identifier's frames, so the injected values lose their consistency with related signals on other identifiers, such as wheel speeds and vehicle speed. Contrastive pretraining on unlabeled fleet telemetry [51] can teach PSC these cross-identifier relations without attack labels, and the correlated-signal attacks of ROAD provide a ready test. We expect the first direction to give the larger gain and the second to be easier to deploy, since it needs no new hardware.

C. Validity of the Perturbation Model

The traffic in all three datasets is authentic, but the perturbations come from PGD, so a fair question is whether real attackers would produce similar changes. Any real attack whose per-byte deviation stays below the bound of Definition 1 is covered by our evaluation, because ϵ was chosen to span the largest deviation in ROAD's physical injection traces. Three limits remain. Non-gradient attacks such as evolutionary search and black-box queries may find perturbations PGD misses. Adaptive attacks that exploit the two-branch layout may do better than plain PGD [28]. The byte-range clamp is also only the coarsest problem-space constraint [46, 47], since many frames carry counters and checksums that a perturbation must preserve to be accepted by the receiving ECU. A constraint set that encodes these fields would make the attacker weaker and the evaluation more realistic.

D. Toward Fleet Deployment

Training on a fleet requires labeled CAN captures, and shipping raw bus data off the vehicle raises privacy concerns. Federated training keeps traffic on each vehicle and shares only model updates, and Aliyu *et al.* have shown a federated forest IDS for in-vehicle networks [52]. Lightweight authentication protocols for vehicular communication, such as the one of Singh *et al.* [53], could also protect the channel that carries model updates. CANDOR suits this setting well. TPI learns timing behavior that transfers across platforms and could be shared fleet-wide, while PSC would need local fine-tuning for each vehicle model's message dictionary.

VII. CONCLUSION

Picture an attack frame arriving at two detectors. The joint detector lets the forged payload pull its whole representation, and its label follows. CANDOR sends the same frame down two paths, where the timing path sits inside the bus schedule and does not move, and the payload path has already seen worse during training. On ROAD that difference is 96.4% retention against 20.7%, from a model of 78,432 parameters that labels a frame in 4.8 μ s. The lesson for in-vehicle network security is to ask, before choosing an architecture, which features the attacker can actually write, and to spend robustness only there. Masquerade traffic remains open, and physical-layer sender fingerprints are the most promising way to close it.

DATA AVAILABILITY

The three datasets analyzed in this study are publicly

available from their maintainers, ROAD through the release described by Verma *et al.* [8], Car-Hacking through the Hacking and Countermeasure Research Lab [27], and CICIoV2024 through the Canadian Institute for Cybersecurity [49].

CONFLICT OF INTEREST

The authors declare no conflict of interest.

AUTHOR CONTRIBUTIONS

H.B.P. conceived the study, designed and implemented CANDOR, ran the experiments, and wrote the paper; P.P. and B.Q. reviewed and edited the paper; all authors had approved the final version.

REFERENCES

- [1] International Organization for Standardization, "Road vehicles, Controller Area Network (CAN), part 1: Data link layer and physical coding sublayer," International Organization for Standardization, Geneva, Switzerland, Standard ISO 11898-1:2024, 2024.
- [2] M. Bozdal, M. Samie, S. Aslam, and I. Jennions, "Evaluation of CAN bus security challenges," *Sensors*, vol. 20, no. 8, 2364, 2020. doi: 10.3390/s20082364
- [3] K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, and S. Savage, "Experimental security analysis of a modern automobile," in *Proc. the 2010 IEEE Symposium on Security and Privacy*, IEEE, 2010, pp. 447–462. doi: 10.1109/SP.2010.34
- [4] S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, K. Koscher, A. Czeskis, F. Roesner, and T. Kohno, "Comprehensive experimental analyses of automotive attack surfaces," in *Proc. the 20th USENIX Security Symposium*. San Francisco, CA, USA: USENIX Association, 2011, pp. 77–92.
- [5] C. Miller and C. Valasek. (Aug. 2015). Remote exploitation of an unaltered passenger vehicle. Presented at Black Hat USA 2015 and DEF CON 23. White paper. [Online]. Available: <http://illmatics.com/Remote%20Car%20Hacking.pdf>
- [6] L. Yang, A. Moubayed, and A. Shami, "MTH-IDS: A multitiered hybrid intrusion detection system for Internet of vehicles," *IEEE Internet of Things Journal*, vol. 9, no. 1, pp. 616–632, 2022. doi: 10.1109/IJOT.2021.3084796
- [7] H. N. AlEisa, F. Alrowais, R. Allafi, N. S. Almalki, R. Faqih, R. Marzouk, M. M. Alnfiar, A. Motwakel, and S. S. Ibrahim, "Transforming transportation: Safe and secure vehicular communication and anomaly detection with intelligent cyber-physical system and deep learning," *IEEE Transactions on Consumer Electronics*, vol. 70, no. 1, pp. 1736–1746, 2024. doi: 10.1109/TCE.2023.3325827
- [8] M. E. Verma, R. A. Bridges, M. D. Iannaccone, S. C. Hollifield, P. Moriano, S. C. Hespeler, B. Kay, and F. L. Combs, "A comprehensive guide to CAN IDS data and introduction of the ROAD dataset," *PLOS ONE*, vol. 19, no. 1, e0296879, 2024. doi: 10.1371/journal.pone.0296879
- [9] Z. Yu, Y. Liu, R. Li, and W. Chang, "LIDS: A lightweight intrusion detection system for controller area network," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 44, no. 9, pp. 3303–3312, 2025. doi: 10.1109/TCAD.2025.3543431
- [10] M. Tavasoli, A. Sarrafzadeh, M. Khaleghi, M. Zakaria, H. B. Pasandi, and A. Karimoddini, "Data communication challenges of connected and automated vehicles in rural areas," *IEEE Access*, vol. 13, pp. 29 220–29 251, 2025. doi: 10.1109/ACCESS.2025.3539598
- [11] F. Aloraini, A. Javed, and O. Rana, "Adversarial attacks on intrusion detection systems in in-vehicle networks of connected and autonomous vehicles," *Sensors*, vol. 24, no. 12, 3848, 2024. doi: 10.3390/s24123848
- [12] S. Longari, P. Cerracchio, M. Carminati, and S. Zanero, "Assessing the resilience of automotive intrusion detection systems to adversarial manipulation," *ACM Transactions on Cyber-Physical Systems*, vol. 9, no. 3, pp. 1–27, 2025. doi: 10.1145/3737294
- [13] M. M. Hasan, R. Islam, Q. Mamun, M. Z. Islam, and J. Gao, "Cross-domain adversarial attacks: Translating network intrusion to CAN bus evasion in V2X environments," in *Proc. the 2025 IEEE 101st Vehicular Technology Conference*, IEEE, 2025, pp. 1–6. doi: 10.1109/VTC2025-Spring65109.2025.11174694
- [14] R. I. Davis, A. Burns, R. J. Bril, and J. J. Lukkien, "Controller Area Network (CAN) schedulability analysis: Refuted, revisited and revised," *Real-Time Systems*, vol. 35, no. 3, pp. 239–272, 2007. doi: 10.1007/s11241-007-9012-7
- [15] S. U. Sagong, X. Ying, A. Clark, L. Bushnell, and R. Poovendran, "Cloaking the clock: Emulating clock skew in controller area networks," in *Proc. the 2018 ACM/IEEE 9th International Conference on Cyber-Physical Systems*, Porto, Portugal: IEEE, 2018, pp. 32–42. doi: 10.1109/ICCPS.2018.00012
- [16] A. R. Javed, S. ur Rehman, M. U. Khan, M. Alazab, and T. R. G., "CANIntelliIDS: Detecting in-vehicle intrusion attacks on a controller area network using CNN and attention-based GRU," *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 2, pp. 1456–1466, 2021. doi: 10.1109/TNSE.2021.3059881
- [17] A. Taylor, S. Leblanc, and N. Japkowicz, "Anomaly detection in automobile control network data with long short-term memory networks," in *Proc. the 2016 IEEE International Conference on Data Science and Advanced Analytics*. Montreal, QC, Canada: IEEE, 2016, pp. 130–139. doi: 10.1109/DSAA.2016.20
- [18] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in *Proc. the 6th International Conference on Learning Representations*, Vancouver, BC, Canada, 2018.
- [19] H. M. Song, H. R. Kim, and H. K. Kim, "Intrusion detection system based on the analysis of time intervals of CAN messages for in-vehicle network," in *Proc. the 2016 International Conference on Information Networking*, IEEE, 2016, pp. 63–68. doi: 10.1109/ICOIN.2016.7427089
- [20] K.-T. Cho and K. G. Shin, "Fingerprinting electronic control units for vehicle intrusion detection," in *Proc. the 25th USENIX Security Symposium*, Austin, TX, USA: USENIX Association, 2016, pp. 911–927.
- [21] K.-T. Cho and K. G. Shin, "Viden: Attacker identification on in-vehicle networks," in *Proc. the 2017 ACM SIGSAC Conference on Computer and Communications Security*. Dallas, TX, USA: ACM, 2017, pp. 1109–1123. doi: 10.1145/3133956.3134001
- [22] M. Kneib and C. Huth, "Scission: Signal characteristic-based sender identification and intrusion detection in automotive networks," in *Proc. the 2018 ACM SIGSAC Conference on Computer and Communications Security*. Toronto, ON, Canada: ACM, 2018, pp. 787–800. doi: 10.1145/3243734.3243751
- [23] W. Choi, H. J. Jo, S. Woo, J. Y. Chun, J. Park, and D. H. Lee, "Identifying ECUs using inimitable characteristics of signals in controller area networks," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 6, pp. 4757–4770, 2018. doi: 10.1109/TVT.2018.2810232
- [24] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, "Distillation as a defense to adversarial perturbations against deep neural networks," in *Proc. the 2016 IEEE Symposium on Security and Privacy*, IEEE, 2016, pp. 582–597. doi: 10.1109/SP.2016.41
- [25] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *Proc. the 2017 IEEE Symposium on Security and Privacy*, IEEE, 2017, pp. 39–57. doi: 10.1109/SP.2017.49
- [26] H. M. Song, J. Woo, and H. K. Kim, "In-vehicle network intrusion detection using deep convolutional neural network," *Vehicular Communications*, vol. 21, 100198, 2020. doi: 10.1016/j.vehcom.2019.100198
- [27] E. Seo, H. M. Song, and H. K. Kim, "GIDS: GAN based intrusion detection system for in-vehicle network," in *Proc. the 2018 16th Annual Conference on Privacy, Security and Trust*. Belfast, UK: IEEE, 2018, pp. 1–6. doi: 10.1109/PST.2018.8514157
- [28] N. Carlini, A. Athalye, N. Papernot, W. Brendel, J. Rauber, D. Tsipras, I. Goodfellow, A. Madry, and A. Kurakin, "On evaluating adversarial robustness," arXiv preprint, arXiv:1902.06705, 2019.
- [29] W. Wu, R. Li, G. Xie, J. An, Y. Bai, J. Zhou, and K. Li, "A survey of intrusion detection for in-vehicle networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 3, pp. 919–933, 2020. doi: 10.1109/TITS.2019.2908074
- [30] S.-F. Lokman, A. T. Othman, and M.-H. Abu-Bakar, "Intrusion detection system for automotive Controller Area Network (CAN) bus system: A review," *EURASIP Journal on Wireless Communications and Networking*, vol. 2019, 184, 2019. doi: 10.1186/s13638-019-1484-3
- [31] M. Muter and N. Asaj, "Entropy-based anomaly detection for in-vehicle networks," in *Proc. the 2011 IEEE Intelligent Vehicles Symposium*, IEEE, 2011, pp. 1110–1115. doi: 10.1109/IVS.2011.5940552
- [32] M. Marchetti, D. Stabili, A. Guido, and M. Colajanni, "Evaluation of anomaly detection for in-vehicle networks through information-theoretic algorithms," in *Proc. the 2016 IEEE 2nd International Forum*

- on Research and Technologies for Society and Industry, IEEE, 2016, pp. 1–6. doi: 10.1109/RTSI.2016.7740627
- [33] P.-S. Murvay and B. Groza, “Source identification using signal characteristics in controller area networks,” *IEEE Signal Processing Letters*, vol. 21, no. 4, pp. 395–399, 2014. doi: 10.1109/LSP.2014.2304139
- [34] M.-J. Kang and J.-W. Kang, “Intrusion detection system using deep neural network for in-vehicle network security,” *PLOS ONE*, vol. 11, no. 6, e0155781, 2016. doi: 10.1371/journal.pone.0155781
- [35] M. Hanselmann, T. Strauss, K. Dormann, and H. Ulmer, “CANet: An unsupervised intrusion detection system for high dimensional CAN bus data,” *IEEE Access*, vol. 8, pp. 58 194–58 205, 2020. doi: 10.1109/ACCESS.2020.2982544
- [36] M. H. Shahriar, Y. Xiao, P. Moriano, W. Lou, and Y. T. Hou, “CANShield: Deep-learning-based intrusion detection framework for controller area networks at the signal level,” *IEEE Internet of Things Journal*, vol. 10, no. 24, pp. 22 111–22 127, 2023. doi: 10.1109/JIOT.2023.3303271
- [37] S. Jeong, S. Lee, H. Lee, and H. K. Kim, “X-CANIDS: Signal-aware explainable intrusion detection system for controller area network-based in-vehicle network,” *IEEE Transactions on Vehicular Technology*, vol. 73, no. 3, pp. 3230–3246, 2024. doi: 10.1109/TVT.2023.3327275
- [38] S. Tariq, S. Lee, H. K. Kim, and S. S. Woo, “CAN-ADF: The controller area network attack detection framework,” *Computers & Security*, vol. 94, 101857, 2020. doi: 10.1016/j.cose.2020.101857
- [39] 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, 100471, 2022. doi: 10.1016/j.vehcom.2022.100471
- [40] M. Tavasoli, A. Sarrafzadeh, A. Karimoddini, M. Khaleghi, and H. B. Pasandi, “Hybrid clustering-transformer IDS for rural CAVs,” in *Proc. the CoNEXT’25 Student Workshop*. New York, NY, USA: ACM, 2025, pp. 7–8. doi: 10.1145/3769700.3771699
- [41] D. Javeed, M. S. Saeed, I. Ahmad, P. Kumar, A. Jolfaei, and M. Tahir, “An intelligent intrusion detection system for smart consumer electronics network,” *IEEE Transactions on Consumer Electronics*, vol. 69, no. 4, pp. 906–913, 2023. doi: 10.1109/TCE.2023.3277856
- [42] F. Ullah, G. Srivastava, S. Ullah, K. Yoshigoe, and Y. Zhao, “NIDS-VSB: Network intrusion detection system for VANET using spark-based big data optimization and transfer learning,” *IEEE Transactions on Consumer Electronics*, vol. 70, no. 1, pp. 1798–1809, 2024. doi: 10.1109/TCE.2023.3328320
- [43] I. J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples,” in *Proc. the 3rd International Conference on Learning Representations*, San Diego, CA, USA, 2015.
- [44] H. Zhang, Y. Yu, J. Jiao, E. Xing, L. E. Ghaoui, and M. Jordan, “Theoretically principled trade-off between robustness and accuracy,” in *Proc. the 36th International Conference on Machine Learning*, 2019, pp. 7472–7482.
- [45] W. Xu, D. Evans, and Y. Qi, “Feature squeezing: Detecting adversarial examples in deep neural networks,” in *Proc. the 2018 Network and Distributed System Security Symposium*. San Diego, CA, USA: Internet Society, 2018. doi: 10.14722/ndss.2018.23198
- [46] F. Pierazzi, F. Pendlebury, J. Cortellazzi, and L. Cavallaro, “Intriguing properties of adversarial ML attacks in the problem space,” in *Proc. the 2020 IEEE Symposium on Security and Privacy*, IEEE, 2020, pp. 1332–1349. doi: 10.1109/SP40000.2020.00073
- [47] G. Apruzzese, M. Andreolini, L. Ferretti, M. Marchetti, and M. Colajanni, “Modeling realistic adversarial attacks against network intrusion detection systems,” *Digital Threats: Research and Practice*, vol. 3, no. 3, pp. 1–19, 2022. doi: 10.1145/3469659
- [48] I. Aliyu, S. Van Engelenburg, M. B. Mu’azu, J. Kim, and C. G. Lim, “Statistical detection of adversarial examples in blockchain-based federated forest in-vehicle network intrusion detection systems,” *IEEE Access*, vol. 10, pp. 109 366–109 384, 2022. doi: 10.1109/ACCESS.2022.3212412
- [49] E. C. P. Neto, H. Taslimasa, S. Dadkhah, S. Iqbal, P. Xiong, T. Rahman, and A. A. Ghorbani, “CICIoV2024: Advancing realistic IDS approaches against DoS and spoofing attack in IoV CAN bus,” *Internet of Things*, vol. 26, 101209, 2024. doi: 10.1016/j.iot.2024.101209
- [50] A. Virmaux and K. Scaman, “Lipschitz regularity of deep neural networks: Analysis and efficient estimation,” in *Advances in Neural Information Processing Systems*, vol. 31, Curran Associates, 2018.
- [51] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, “A simple framework for contrastive learning of visual representations,” in *Proc. the 37th International Conference on Machine Learning*, 2020, pp. 1597–1607.
- [52] I. Aliyu, M. C. Feliciano, S. Van Engelenburg, D. O. Kim, and C. G. Lim, “A blockchain-based federated forest for SDN-enabled in-vehicle network intrusion detection system,” *IEEE Access*, vol. 9, pp. 102593–102608, 2021. doi: 10.1109/ACCESS.2021.3094365
- [53] A. Singh, P. Rani, J. V. N. Ramesh, S. V. Athawale, A. H. Alkhayyat, A. N. Aledaily, T. A. Prola, and R. Sharma, “Blockchain-based lightweight authentication protocol for next-generation trustworthy Internet of vehicles communication,” *IEEE Transactions on Consumer Electronics*, vol. 70, no. 2, pp. 4898–4907, 2024. doi: 10.1109/TCE.2024.3351221

Copyright © 2026 by the authors. This is an open access article distributed under the Creative Commons Attribution License which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited (CC BY 4.0).