Computers and Electrical Engineering 137 (2026) 111225

Contents lists available at ScienceDirect

Computers and Electrical Engineering

journal homepage: www.elsevier.com/locate/compeleceng

Unsupervised anomaly detection in internet of vehicles via spectral-residual preprocessing and adversarial variational autoencoding

Arash Heidari [a,b,c,f,*], Mohammed Ababneh [d], Roberto Passerone [c,*], Jamal N. Al-Karaki [g], Ahmad Khonsari [c,*], Seyed Hamed Rastegar [e], Muhammad Baqer Mollah [f], Kyu In Lee [f,*]

[a] School of Electrical and Computer Engineering, College of Engineering, University of Tehran, Tehran, Iran
[b] School of Computer Science, Institute for Research in Fundamental Sciences (IPM), Tehran, Iran
[c] Department of Information Engineering and Computer Science, University of Trento, 38123, Trento, Italy
[d] Department of Information Technology, The Hashemite University, Zarqa, Jordan
[e] Department of Communications and Electronics, School of Electrical and Computer Engineering, Shiraz University, Shiraz, Iran
[f] Department of Information Science Technology, Cullen College of Engineering, University of Houston, Houston, TX, USA
[g] Department of Computational Systems, Zayed University, UAE

ARTICLE INFO

Keywords:
Internet of vehicles
Unsupervised anomaly detection
Variational autoencoder
Particle swarm optimization
Spectral residual preprocessing

ABSTRACT

The Internet of Vehicles (IoV) forms a large-scale cyber-physical system exchanging continuous data for safety, autonomy, and traffic efficiency, but faces significant cybersecurity risks. For example, exploiting IoV connectivity, advanced attacks such as spoofing, replay, and DoS often evade traditional Intrusion Detection Systems (IDSs). While Variational Autoencoder (VAE)-based unsupervised methods offer promise, they struggle with noisy, non-stationary data and poor generalization. In this paper, we present SPARTA, a Spectral-Preprocessing and Adversarial Representation Training Architecture for fully unsupervised IoV anomaly detection. Using Spectral Residual preprocessing, SPARTA converts multivariate signals into saliency-enhanced sequences processed by a PSO-optimized adversarial VAE. Binary PSO tunes architectural hyperparameters, while a dual-discriminator mechanism mitigates overfitting and enforces latent alignment. On average, performance evaluation results show that SPARTA improves detection accuracy by 2%, F1-score by 2.8%, and reduces false positives by 2.5% over existing methods. It delivers a 7% F1 improvement under domain shifts and over 6% gains in noisy settings, maintaining consistent 5–7% performance advantages across real-world IoV scenarios.

1. Introduction

The Internet of Vehicles (IoV) is an active, broad cyber physical system where ubiquity is brought to vehicular networks via connectivity and hence allowing real time communication between vehicles, the infrastructure, and cloud services as well as other

* Corresponding authors.
E-mail addresses: Arash_heidari@ieee.org, arash.heidari@ut.ac.ir, aheidari@central.uh.edu, arash.heidari@unitn.it (A. Heidari), roberto.passerone@unitn.it (R. Passerone), Jamal.Al-Karaki@zu.ac.ae (J.N. Al-Karaki), a_khonsari@ut.ac.ir (A. Khonsari), h.rastegar@shirazu.ac.ir (S.H. Rastegar), mmollah@uh.edu (M.B. Mollah), klee48@central.uh.edu (K.I. Lee).

https://doi.org/10.1016/j.compeleceng.2026.111225
Received 23 November 2025; Received in revised form 23 March 2026; Accepted 4 May 2026
Available online 23 May 2026
0045-7906/© 2026 Elsevier Ltd. All rights are reserved, including those for text and data mining, AI training, and similar technologies.

networked entities [1]. These interactions cannot be done without continuous data streams produced by Global Positioning System (GPS) receivers, Inertial Measurement Units (IMUs), Controller Area Network (CAN) buses, Vehicle-to-Everything (V2X) interfaces, and LiDAR modules [2]. IoV incorporation with intelligent transportation systems is set to improve autonomous driving, increase driver safety, as well as improve traffic efficiency [3]. However, the scale of interconnectivity of IoV creates high cybersecurity risks [4]. Matters of the malicious may engage in Denial-of-Service (DoS), spoofing, replay or injection attacks using attack surfaces on the control units or a communication interface or in an edge-computing module [5]. Traditional rule-based and signature-based Intrusion Detection Systems (IDSs) are not applicable to detect complex or novel threats because the former are often insidious and context-sensitive [6]. In turn, the reliability of the system and the security of passengers and infrastructure in this safety-critical area require efficient and precise detection of anomalies in time [7].

Unsupervised deep learning-based anomaly detection has emerged as a key area of study for IoV security in order to handle the complexity and unpredictability of new threats [8]. The goal of these models is to detect instances that differ from learned distributions and learn the statistical and temporal patterns of normal behavior from high-dimensional multivariate time series data [9]. Variational Autoencoders (VAEs) have shown efficacy in reconstructing input sequences and identifying abnormalities via reconstruction mistakes, especially when paired with adversarial learning components [10]. IoV data, however, is often noisy, non-stationary, and very scenario-specific [11]. These features make it difficult to choose the right number and kinds of layers, train stable models, and adjust hyperparameters like batch sizes, learning rates, and optimizers [12]. Additionally, a lot of existing methods depend on manually created model setups or require labeled attack data, which is seldom accessible or typical of real-world deployments [13]. An automatic, scalable, and adaptable approach that can generalize across many IoV scenarios unsupervised is thus urgently needed. Also, given the computational complexity of deep unsupervised models in large-scale IoV deployments, split learning offers a promising way to improve training and inference efficiency by distributing computation between edge devices and centralized servers [14].

Although there have been improvements, current anomaly detection frameworks for IoV typically have three main drawbacks: (1) limited robustness to sensor noise and communication jitter present in real-world IoV systems; (2) poor generalization across dynamic vehicular environments because of fixed, non-optimized architectures; and (3) reliance on manual tuning or supervised signals, which limits scalability. We suggest SPARTA, a Spectral-Preprocessing and Adversarial Representation Training Architecture made for reliable, completely unsupervised anomaly detection in IoV contexts, as a solution to these problems. SPARTA presents a multi-phase method: Spectral Residual (SR) preprocessing extracts frequency domain abnormalities using Fourier decomposition and filtering, converting raw multivariate time series into saliency-enhanced sequences. A sliding window method is then used to segment and normalize these sequences. SPARTA is based on a PSO-optimized adversarial VAE that learns the latent structure of typical driving behavior on its own. Binary Particle Swarm Optimization (BPSO) is used to optimize the VAE architecture, which includes the number of layers, convolutional kernel sizes, normalization techniques, activation functions, learning rate, optimizer type, and batch size. The fitness function in BPSO is the reconstruction error on a validation set. Additionally, a dual-discriminator adversarial learning system inhibits overfitting by requiring statistical similarity between the original and reconstructed data and guarantees that latent representations match a prior distribution. This completely automated end-to-end learning paradigm offers scalable deployment across many IoV use cases, adapts to unknown circumstances, and does away with the requirement for labeled data. The following are our primary contributions:

- Presenting a new SR preprocessing module that isolates uncommon frequency components and converts temporal patterns into the frequency domain to improve anomaly saliency in IoV time-series data;
- Creating a PSO-optimized adversarial VAE system that does not need human tweaking by automatically locating the best neural architectures and hyperparameters in a discrete binary space;
- Implementing a dual-discriminator adversarial training strategy that aligns latent distributions with a prior and enforces similarity between real and reconstructed data to improve generalization and robustness;
- Creating a completely unsupervised anomaly detection pipeline that does not need labeled attack data and can function in a variety of noisy IoV situations;
- Validating the effectiveness of the proposed SPARTA model through comprehensive experiments on real-world IoV datasets, demonstrating superior accuracy, scalability, and resilience compared to existing methods.

The subsequent sections of this paper are organized in the following sequence: Section II presents a summary of the latest findings that are related to the study topic. The SPARTA model is elaborated in Section III. Also, the SPARTA method will be provided in Section IV. The paper presents the results of the SPARTA method and provides a comprehensive comparison with existing approaches in Section V. Section VI provides an overview and conclusion.

2. Related work

In this part, we are going through related work in the domain. In this regard, Xu, et al. [3] proposed a few-shot learning-based augmentation to find zero-day attacks in IoV. They introduced a model with a multi-generator/discriminator architecture and focal loss enhancements to improve detection while lowering false positives. However, it cannot adapt to new, unseen behaviors because it relies on previous attack similarity. Also, Li, et al. [15] proposed a hybrid anomaly detection framework that combined 1D Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM), and Kolmogorov-Arnold networks with residual filtering. Their framework improved accuracy and reduced false positives through multivariate sensor correlation and residual smoothing. To tackle trajectory-based anomalies, Shen, et al. [16] introduced Dynamic Network Representation learning (DNR), which models vehicle

interactions over time using k-nearest topology and Skip-Gram embeddings. While it worked well for spatial-temporal deviations, it does not work for all types of IoV anomalies. Moreover, Mishra, et al. [17] proposed a blockchain-driven Main Key (MK) management system for safe V2V communications. Their system combined LSTM-based anomaly detection with blockchain's trust characteristics. However, their technique added latency and infrastructure reliance, which makes it unsuitable for scaling IoV systems. In addition, Wu, et al. [4] put out an Unsupervised Knowledge Distillation framework (UKD-TEAD) for traffic equipment anomalies. This framework included multiscale detection heads and hierarchical feature transfer. However, it is still only useful for detecting problems at the equipment level and not across the whole IoV. Plus, Politi, et al. [18] suggested a supervised video anomaly detection approach that combines picture quality evaluation and drift detection to ensure that autonomous cars can see well. Their findings were encouraging, but the system needs labeled data, which makes it less useful in dynamic IoV situations. Besides, Wang, et al. [19] provided a GRU-Autoencoder with Bayesian optimization and checkpoint restart procedures to make fog-based vehicular computing more reliable. However, their technique only looked at fog dependability and not other security issues. In another work, Singh and Rathore [20] proposed a real-time Bayesian Online Change Point Detection (BOCPD) with recovery forecasting to find and fix problems with connected vehicles using the TCV dataset. Their method improved resilience and detection speed, but it does not add any spectral or adversarial enhancements. Moreover, Zhou, et al. [21] suggested an interpretable unsupervised detection technique employing a Kalman Variational Autoencoder with subspace extraction. This method achieved strong detection using latent space filtering and adaptive thresholding; however, it did not look at adversarial representation alignment. For Industrial aim, Li, et al. [22] suggested Meta-Heuristic Deep Random Neural Networks (MH-DRNN), which used sunflower movement-based optimization and deep recurrent architectures to get high accuracy on data that is not all the same. However, it does not adjust to vehicles or handle spectral data. Bergies, et al. [23] offered an DNN-based IoT framework for AEVs, integrating dynamic programming with homomorphic encryption to safeguard control signal integrity; nonetheless, its supervised architecture and encryption expenses impede real-time adaptability. Also, Cao, et al. [24] offered a self-supervised solution for In-Vehicle Networks (IVN) security that used hierarchical transformers to forecast messages and capture sequential dependencies without labels. However, it only looked at IVNs and did not look at generic sensor streams. As well as Zhao, et al. [25] suggested CoGAN, a hybrid VAE-GAN framework that learns normal behavior distributions by adversarial variational inference to improve BSM anomaly detection in C-V2X. However, manual tweaking and model setup are still difficult. Finally, Devika, et al. [26] provided a GAN-based unsupervised anomaly detection system optimized for CAVs, improving LSTM-based identification of 11 attack types across critical dynamics including location and speed; however, spectral pretreatment and architectural optimization are needed.

In conclusion, although these techniques significantly advance anomaly detection in the IoV, they frequently exhibit substantial shortcomings, including dependence on labeled data, inability to generalize across non-stationary and noisy sensor streams, lack of frequency-domain feature enhancement, and the necessity for manual model tuning. To address these problems, we provide SPARTA, a completely unsupervised anomaly detection system that utilizes SR preprocessing and a PSO-optimized adversarial Variational Autoencoder. SPARTA turns multivariate IoV time series into representations that highlight important features, splits them up using sliding windows, and utilizes BPSO to automatically adjust hyperparameters like learning rate, optimizer, and network depth. A dual-discriminator adversarial configuration makes the latent space even more regular and makes it less likely to overfit. SPARTA works without any labeled data, adapts to different types of vehicles, and works better and scales better with real-world IoV datasets than previous systems. It is a robust, noise-tolerant, and fully automated solution for next-generation vehicular cybersecurity. Table 1 shows

Table 1
Summary of related works.

Researchers	Core Idea	Advantage	Challenges
Xu, et al. [3]	FLCGAN with few-shot augmentation	Reduces false positives	Generalization in dynamic environments
Li, et al. [15]	CNN, LSTM, and KAN	High accuracy	High complexity
Shen, et al. [16]	DNR-based anomaly detection	Effective for trajectory deviations	Lacks a general anomaly detection capability
Mishra, et al. [17]	Blockchain MK management and LSTM anomaly model	High security and attack resistance	Blockchain-induced latency
Wu, et al. [4]	UKD-TEAD using teacher-student distillation	Accurate detection with no labels	Equipment-focused, not sensor-wide
Politi, et al. [18]	Drift detection for CAVs	High detection accuracy	Requires labeled data
Wang, et al. [19]	GRU-AE with BO-TPE for VFC reliability	Automated rollback and threshold tuning	Service-oriented focus only
Singh and Rathore [20]	Real-time BOCPD and recovery for CV anomaly detection	Strong detection + recovery synergy	No spectral/adversarial integration
Zhou, et al. [21]	VAE with subspace extraction	Robust and interpretable	No adversarial latent enforcement
Li, et al. [22]	MH-DRNN with sunflower-based feature optimization	99.2% accuracy on heterogeneous data	Industrial setting limitation
Bergies, et al. [23]	IoT-MPC-DNN with encrypted AEV	Strong cyberattack handling	Supervised + high computational cost
Cao, et al. [24]	Self-supervised message prediction using Hierarchical Transformers	Handles concept drift and message-level security	Limited to IVNs
Zhao, et al. [25]	CoGAN for BSM anomaly detection	Captures normal distribution complexity	Manual model setup
Devika, et al. [26]	GAN for CAVs optimized via LSTM	Detects 11 attack types accurately	No spectral or optimization tuning
Our Work (SPARTA)	Spectral preprocessing + PSO-optimized adversarial VAE	Fully unsupervised, noise-resilient, adaptive architecture	–

the investigated related works.

3. SPARTA model

In this section, we go through the SPARTA model in detail. So, we constantly gather multivariate sensor and network telemetry streams on the IoV platform.

$$X = \{x_1, x_2, \ldots, x_n\} \tag{1}$$

where each $x_t \in \mathbb{R}^M$ is the M-dimensional feature vector observed at time t. Our objective is to autonomously develop an adversarial VAE that, in an unsupervised fashion, acquires the standard behavior of vehicle-generated time series and identifies any x_t whose reconstruction error suggests a possible anomaly such as a cyber-attack, sensor failure, or communication error in the IoV security domain [27]. We conceptualize the simultaneous selection of SPARTA's VAE hyperparameters, network depth, and per-layer configurations as a single-objective discrete optimization problem [28]. We are looking for the hyperparameter set $[B, f, \eta]$, the number of convolutional layers n, and the specification for each layer $\{\text{ layer }_j\}$ n that will minimize the mean-squared reconstruction error on a held-out validation split W_{vali}, while also training the VAE weights w to optimality on the training split W_{train} . After optimization, we assess the SPARTA efficacy of the adversarial VAE on the test split W_{test}, resulting in a fully automated, adversarial-VAE-driven unsupervised anomaly detector specifically designed for real-world IoV settings [29]. The optimization problem of SPARTA is then defined as:

$$\min_{\{B,f,n\},n,\{by\in\}} \text{MSE}_{\text{vali}}\ \left([B,f,n], n, \{\text{ layer }_j\}; w^*\right) \text{s.t.}\, w^* \in \underset{w\in n_*}{\text{argmin}} \text{MSE}_{\text{train}}\ (w) \tag{2}$$

Hyperparameters and architecture parameters to optimize:

$$[B, f, \eta],\ n,\quad \text{layer }_j = \left[oc_j, ks_j, nt_j, af_j\right],\ j = 1, \ldots, n. \tag{3}$$

Hyperparameter domains:

$$f \in \{\text{adamax}, \text{adam}, \text{rmsprop}, \text{adadelta}\}, nt_j \in \{\text{batchnorm},\ \text{none}\}, af_j \in \{\text{sigmoid}, \text{tanh}, \text{relu}, \text{none}\} \tag{4}$$

Here, $\text{MSE}_{\text{train}}\ (w)$ is the reconstruction loss on W_{train}, and $\text{MSE}_{\text{vali}}\ (; w^*)$ is the average reconstruction loss on W_{vali}, given the chosen hyperparameters, architecture, and trained weights w^*. Also, Table 2 shows definitions of symbols.

As shown in Fig. 1, SPARTA is a system for finding anomalies in IoV systems without supervision and in real time. It begins at the sensor layer, where it collects high-dimensional telemetry from different sources. An SR preprocessing module processes raw signals, then sliding windows normalize and segment them. Using validation reconstruction loss, a BPSO-based PSO controller improves the architecture and hyperparameters of the downstream model. The main detector is an adversarial VAE with an encoder–decoder and two discriminators. D_1 aligns the latent space with a Gaussian prior, and D_2 makes sure that the original and reconstructed inputs are the same. Finally, the anomaly decision module sets a threshold for reconstruction error to make logs, confidence scores, and real-time alerts.

The SR preprocessing module in SPARTA is based on the idea that normal IoV telemetry has strong periodic or slowly changing spectral structures, while anomalies have sparse, irregular, or non-stationary components in the frequency domain. SR effectively suppresses dominant, repetitive background patterns while amplifying statistically rare spectral components by changing a time-series signal into the frequency domain and taking away a smoothed log-amplitude spectrum from the original [30]. In a formal sense, this operation is like a high-pass filter on the log-spectrum. It makes the relative energy of unusual frequencies higher and the Signal-to-Noise Ratio (SNR) of features related to anomalies better. As a result, unusual behaviors stand out more in the reconstructed time-domain signal, making it easier for downstream representation learning models to pick them up. In the case of the adversarial

Table 2
Definitions of the symbols.

Symbol	Definition	Symbol	Definition
$\boldsymbol{X}$	Sequence of multivariate telemetry vectors collected	$\boldsymbol{x_t} \in \mathbb{R}^{\text{M}}$	M-dimensional feature vector observed at time step t
$\boldsymbol{M}$	Number of sensor/network features	$\boldsymbol{n}$	Total number of convolutional layers
$\boldsymbol{B}$	Mini-batch size	$\boldsymbol{f}$	Optimizer type
$\boldsymbol{\eta}$	Learning-rate hyper-parameter	$\boldsymbol{ks_j}$	Kernel (filter) size of layer j
$\boldsymbol{oc_j}$	Number of output channels	$\boldsymbol{sw}$	Sliding-window length for time-series sampling
$\boldsymbol{nt_j}$	Normalization type for layer j	$\boldsymbol{af_j}$	Activation function in layer j
$\boldsymbol{w}$	Trainable weights of the VAE	$\boldsymbol{w^*}$	Optimal weights after minimizing MSE
$\boldsymbol{ss}$	Sampling stride/step between windows	$\boldsymbol{Y_i}$	Normalized sample segment
$\boldsymbol{A(f)}$	Amplitude spectrum	$\boldsymbol{L(f)}$	Log-amplitude spectrum
$\boldsymbol{AL(f)}$	Smoothed log-amplitude spectrum	$\boldsymbol{R(f)}$	Spectral residual
$\boldsymbol{P(f)}$	Phase spectrum of the original signal	$\boldsymbol{S(x)}$	Time-domain saliency map after inverse FFT
$\boldsymbol{z}$	Latent variable	$\boldsymbol{q(z)}$	Posterior distribution
$\boldsymbol{p(z)}$	Prior distribution	$\mathbf{AE_E}$, $\mathbf{AE_D}$	VAE encoder and decoder networks, respectively
$\boldsymbol{D_1}$	Discriminator aligning $q(z)$ with the prior $p(z)$	$\boldsymbol{D_2}$	Discriminator judging real vs. reconstructed data

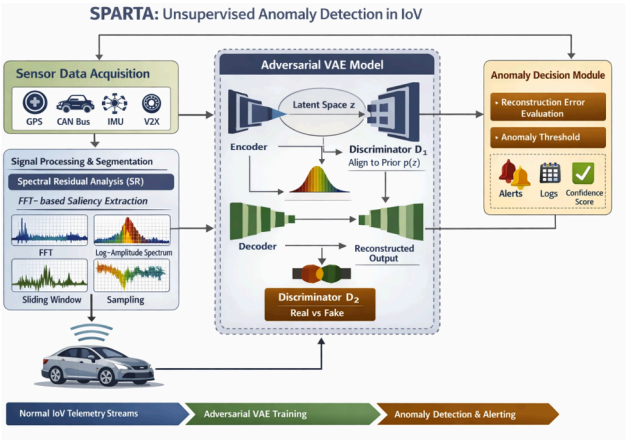

Fig. 1. SPARTA system architecture.

VAE, the complementary goals of minimizing reconstruction and adversarial regularization help with convergence and stability. The latent-space discriminator D_1 makes sure that the combined posterior $q(z)$ and a set prior $p(z)$ are in line with each other. This limits the shape of the latent space and stops representations from becoming too simple or too specialized. This regularization lowers the variance in latent encodings and makes it easier to generalize when the domain changes. The data-space discriminator D_2 acts as a distribution-level regularizer on reconstructed samples, pushing them toward statistically consistent reconstructions that go beyond just minimizing pointwise MSE. In adversarial learning, when we use the standard method of alternating minimization, the joint optimization of the encoder, decoder, and discriminators reaches a local equilibrium where the latent distribution is close to the prior, and the reconstructed samples are no longer different from real data. Even though non-convexity means that global optimality cannot be guaranteed, the combined adversarial constraints make training dynamics much more stable and give normal data consistent reconstruction-error distributions. This is important for reliable unsupervised anomaly detection [30].

4. SPARTA method

In the SPARTA method, we seek to automatically design an adversarial VAE that learns the normal patterns of vehicle telemetry, such as engine CAN-bus signals, GPS/IMU traces, and V2X communications and flags deviations indicative of cyber-attacks, sensor failures, or network faults. We adopt PSO in the SPARTA method to search the discrete space of VAE hyperparameters and convolutional-encoder/decoder architectures. Each particle encodes a candidate adversarial VAE configuration, including batch size, optimizer type, learning rate, number of layers, per-layer kernel counts/sizes, normalization, and activation choices as a binary string. At each generation g, we evaluate all N particles on a held-out validation set of IoV time series, computing reconstruction-error fitness via the adversarial VAE's MSE. Each particle retains its personal best (p_{best}) and the swarm's global best (g_{best}). We then update inertia,

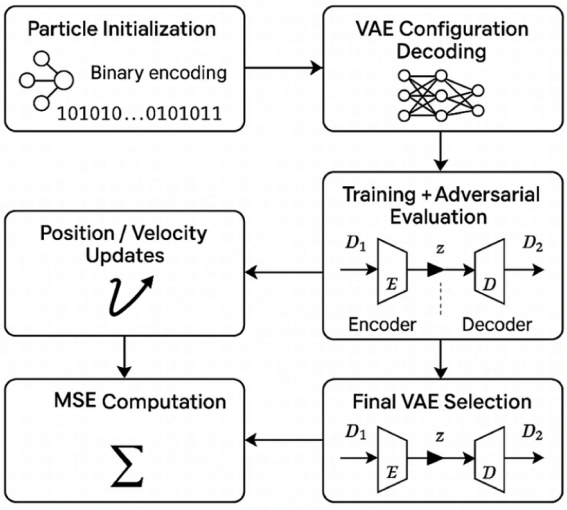

Fig. 2. Binary PSO optimization process.

velocity, and position so that particles fly toward both their own best architecture and the swarm's best, gradually converging on the optimal design. Once PSO terminates, we decode g_{best} into the final adversarial VAE, which can be deployed in real-time on vehicles or roadside units for unsupervised anomaly detection with high precision and low latency.

Fig. 2 shows that SPARTA uses BPSO to automatically adjust the hyperparameters and architecture of the adversarial VAE. There is a swarm of particles, and each one is a binary string that describes a possible setup (for example, the number of conv layers, the batch size, the optimizer, the learning rate, the kernel sizes, the normalization, and the activation). Each particle is turned into a real VAE configuration: the encoder turns inputs into latent z, the decoder puts the signal back together, and adversarial discriminators D_1 (latent prior alignment) and D_2 (reconstruction realism/consistency) help with training. The validation reconstruction loss (MSE) is used to figure out fitness, and BPSO uses inertia, pbest, and gbest to update particles until they converge. The decoded configuration of the best particle is chosen as the final VAE model. This makes it possible to find anomalies without having to manually adjust settings for different IoV data.

Before using the PSO-based optimization process in SPARTA, it is important to preprocess the multivariate data that is collected in real time from sensors inside vehicles and V2X communication systems. These data streams are often extremely dimensional and very noisy in nature: GPS paths, CAN bus, engine diagnostics, and vehicle-to-vehicle communication records. The technique is called SR and is used to highlight deviant trends in these datasets. SR is an unqualified algorithm that was initially designed as a visual saliency algorithm, but it is also very effective in detecting temporal outliers in motion images of vehicles. The SR methodology has three major steps. The Fourier transform is used to change the raw time-series data into the frequency domain. This lets us see the periodic structures in the signal. For example, we find the amplitude spectrum of an input time series x and then take its logarithmic transformation as follows:

$$A(f) = Amplitude(F(x)) \tag{5}$$

$$L(f) = log(A(f)) \tag{6}$$

In this case, $F(x)$, is the Fourier transform of the sequence x, $A(f)$ is the resulting amplitude spectrum, and $L(f)$ is the log-amplitude spectrum, which is better for showing small changes in the spectrum. In the second step, the SR is found by taking the log-amplitude spectrum and subtracting a smoothed version of it from the original. The smoothing is done by convolving with a uniform kernel $h_q(f)$, where all the values are $1/q^2$ in a $q \times q$ matrix:

$$AL(f) = hq(f)*L(f) \tag{7}$$

$$R(f) = L(f) - AL(f) \tag{8}$$

Here, $AL(f)$ approximates the average background spectrum, and $R(f)$ is the SR that emphasizes unique or rare frequency components, indicators of anomalies. In the final stage, the modified frequency signal is converted back into the time domain using the inverse Fourier transform. The phase spectrum $P(f)$ is preserved from the original signal to maintain temporal alignment, and the SR is exponentiated and combined with the phase:

$$P(f) = Phase(F(x)) \tag{9}$$

$$S(x) = F^{-1}(exp(R(f) + iP(f))) \tag{10}$$

The resulting $S(x)$ is the saliency map of the input signal x. It shows the areas that are most likely to be strange. Then, this saliency map is put into the PSO-optimized adversarial VAE model to find anomalies in the IoV environment without any help. The SR method improves the accuracy and reliability of the downstream detection system by preprocessing the data. The next step after using the SR method to improve anomaly-related features in SPARTA time-series data is to normalize the saliency sequence that comes out of it. We use min–max normalization to make sure that all sensor modalities are scaled the same way and to get rid of any bias that might come from differences in scale. This step scales each processed signal component S_j into the range [0,1], giving us the normalized sequence x_j, as follows:

$$X_j = \frac{S_j - S_j^{\min}}{S_j^{\max} - S_j^{\min}},\ 1 \le j \le m \tag{11}$$

where $S_j^{\min}$ and $S_j^{\max}$ represent the minimum and maximum values of the saliency signal S_j, respectively, and X_j denotes the corresponding normalized component. This normalization ensures that all inputs to the subsequent detection model are on the same scale, improving training stability and convergence in the adversarial VAE. Following normalization, we perform data sampling using a time-series sliding window approach. This technique segments the continuous time series into overlapping or non-overlapping samples suitable for input into deep learning models. Each time-series segment Y_i is constructed by sliding a window of length sw across the normalized data:

$$Y_i = \{X_i, X_{i+1}, \ldots, X_{i+sw}\},\ 1 \le n \le sw \tag{12}$$

Sampling is done at a fixed stride or interval ss, collecting each Y_i every ss steps to form the complete dataset W for model training and evaluation:

$$W = \{Y_1, Y_{1+ss}, \ldots, Y_{1+n\times ss}, \ldots, Y_{1+M-sw}\},\ 0 \le n \le \frac{M-sw}{ss} \tag{13}$$

The adversarial VAE architecture used in SPARTA is shown in Fig. 3, with a focus on the dual-objective training process and data flow via its modular components. An input signal that represents a normalized and segmented saliency-enhanced time series obtained from multi-sensor IoV telemetry (e.g., GPS, CAN bus, IMU) is used to start the process. After receiving this input, the encoder approximates a posterior distribution $q(z)$ by compressing the high-dimensional sequence into a lower-dimensional latent representation z. Two linked components are then fed this latent vector. It is first sent to the decoder, which creates the reconstruction signal by reconstructing the original input signal as nearly as feasible. MSE calculation is used to compare this reconstruction to the original input, producing a reconstruction loss that measures the difference and propels backpropagation for the encoder and decoder. Discriminator D_1 simultaneously examines the latent vector z, assessing whether it seems to be sampled from a conventional Gaussian prior $p(z)$ using an adversarial approach. This improves generalization and prevents overfitting to particular traffic or sensor situations by incentivizing the encoder to generate statistically well-aligned latent spaces. Simultaneously, Discriminator D_2 (implied in the design) evaluates the reconstructed signal, ensuring fidelity and realism in the output by differentiating between actual input samples and their reconstructions. Together, these two discriminators direct adversarial training: D_2 maintains reconstruction integrity, while D_1 molds the latent distribution. The VAE learns robust and denoised representations of typical IoV behavior thanks to the combination of adversarial feedback and reconstruction loss. Any divergence is noted as a possible abnormality, whether it takes the form of latent misalignment or high reconstruction error. Even in dynamic and low-SNR automotive situations, this architecture enables SPARTA to identify sophisticated assaults, sensor malfunctions, or communication irregularities completely without labeled data.

In this case, sw is the length of the sliding window, ss is the sampling interval, and M is the total number of points in the dataset. This process makes a series of samples that are all the same shape, which lets the PSO-optimized adversarial VAE learn strong temporal patterns and find changes that show security problems in the IoV system. The SPARTA method also uses the structural strengths of the VAE to learn latent representations of IoV time-series data on its own [31]. The VAE architecture is meant to take high-dimensional input and turn it into a lower-dimensional latent space. Then, it uses this compressed representation to rebuild the original input. This reconstruction process is guided by the minimization of a reconstruction loss, usually assessed through mean squared error, which guarantees that the acquired latent features preserve the fundamental attributes of the original data. The VAE that is embedded in the SPARTA method pipeline has built-in noise robustness. Consequently, the VAE preserves the capability to learn denoised representations even in the case of input IoV telemetry being affected by disturbances or anomalies, such as sensor inaccuracies, communication jitter, or environmental interference. This attribute makes the VAE generate less noisy and more usable representations of the original data by reducing noise and maintaining key structural and temporal patterns that are paramount in identifying anomalies.

In the SPARTA method-based anomaly detection framework for the IoV, the optimization process starts by setting up a group of possible solutions, each of which is a possible configuration of the adversarial VAE model. The first step, called initialization, creates a group of N particles called P_0. A binary string of length l encodes a unique VAE architecture and its hyperparameters for each particle p_i. The binary length l is set ahead of time based on the encoding ranges of all the parameters that need to be optimized, such as batch size, optimizer type, learning rate, number of layers, and attributes of each layer. Formally, the initial population is represented as:

$$P_0 = \{p_1, p_2, \ldots, p_N\},\ 1 \le i \le N \tag{14}$$

Each particle p_i, is a binary string defined as:

$$p_i = \{b_1, b_2, \ldots, b_d, \ldots, b_L\},\ 1 \le d \le L \tag{15}$$

where each $b_d \in \{0, 1\}$ encodes part of the VAE's hyperparameters or layer configurations. These binary strings are later decoded to their corresponding semantic parameter sets, yielding the decoded VAE configuration q_i for particle p_i:

$$q_i = \left\{[B, f, \eta], n, \left[\text{layer}_1, \ldots, \text{layer}_j, \ldots, \text{layer}_n\right]\right\} \tag{16}$$

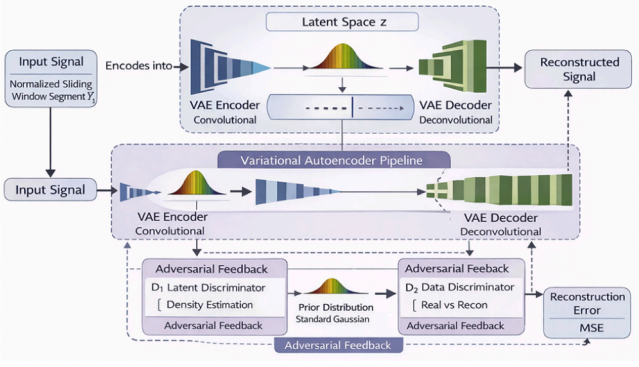

Fig. 3. Adversarial VAE architecture for SPARTA.

Here, $[B, f, \eta]$ represents the training hyperparameters-batch size, optimizer type, and learning rate, while n is the number of convolutional layers, and each layer $_j$ captures the structure of the j-th layer (number of kernels, kernel size, normalization type, and activation function). The detailed definitions and parameter ranges are established in the problem formulation section. It illustrates how these parameters are encoded into a binary format, which enables the PSO algorithm to perform efficient search and evolution in a discrete binary solution space tailored to unsupervised anomaly detection, each new particle must be tested with a carefully designed fitness function during the optimization process. This test checks how well a certain adversarial VAE configuration, which is shown by a particle, can learn normal behavior from data from vehicle sensors and find deviations. The SPARTA method architecture shows that the network has four main parts: an encoder (AE_E), a decoder (AE_D), and two discriminators (D_1 and D_2). These parts work together to make sure that latent encoding is correct, reconstruction is strong, and generalization is effective. Let z be the latent variable that the encoder makes from an IoV time series sample W. To find the posterior distribution $q(z)$ over the latent space, we need to look at the input data distribution:

$$\mathrm{q(z)} = \int \mathrm{q(z|W)} p_d(\mathrm{W}) \mathrm{dW} \tag{17}$$

In this case, $q(z|W)$ is the conditional encoding distribution, and $p_d(\mathrm{W})$ is the real data distribution of the IoV input sample. The adversarial network D_1 is trained to make sure that $q(z)$ matches a prior distribution $p(z)$, which is usually a standard Gaussian, by punishing differences between the two distributions. At the same time, the second adversarial network, D_2, is meant to stop overfitting. It does this by looking at the statistical properties of the original input data and the reconstructed output from the decoder. The reconstructed sample $\widehat{w}$ is created in the following way:

$$\widehat{w} = \mathrm{AE}_D(\mathrm{AE}_E(\mathrm{W})) \tag{18}$$

We create the corresponding VAE model based on the decoded configuration q_i of each particle, which includes hyperparameters and details about the neural architecture. The encoder AE_E is made up of convolutional layers that are defined in q_i, and the decoder v is built the same way, but with deconvolutional layers. A fixed architecture template is used to build the discriminators, D_1 and D_2. The training set $\mathrm{W}_{\mathrm{train}}$ is used to train each model, and the training is done offline for a set number of adversarial training epochs E_1. After this training process, the fitness function, which is the average reconstruction loss or a combination of both adversarial and reconstruction losses, is used to see how well the particle works. The PSO update for the next generation is then based on this fitness value.

Each particle's fitness is evaluated using three separate phases of adversarial training and reconstruction assessment. With only the structure and feedback of its own latent and reconstructed signals, this assessment enables the model to acquire meaningful representations of typical vehicular behavior in a fully unsupervised way. Step 1 involves adversarial training between the discriminator D_1 and the encoder AE_E. This creates a two-player min-max game based on the GAN concept. The encoder creates latent representations $q(z)$ once the VAE receives the IoV dataset W. To determine whether a sample comes from the encoder or the prior, these representations are sent into the discriminator D_1 together with samples taken from a prior distribution $p(z)$. The encoder's goal is to generate latent codes that are identical to the previous. The definition of the adversarial loss for D_1 is:

$$\begin{aligned} \mathscr{L}_{D_1} = \quad & \mathbb{E}_{z\sim p(z)}[\log\,(D_1(z))] \\ & + \mathbb{E}_{W\sim p(W)}[\log\,(1 - D_1(\mathrm{AE}_E(W)))]. \end{aligned} \tag{19}$$

Gradient descent is used to reduce this loss, updating both D_1 and AE_E, such that the output of the encoder (z) closely resembles the prior (z), improving training stability and generalization. Step 2: The discriminator D_2 and the decoder is involved in the second adversarial training stage. To create reconstructions, the training data, W train, is run through the whole VAE. Also, we have $W^2 = \mathrm{AE}_D(\mathrm{AE}_E(\mathrm{W}))$, $W^2 = \mathrm{AE}_D\,(\mathrm{AE}_E(W))$. The discriminator D_2 receives both the original inputs and the reconstructed outputs and attempts to differentiate between created and genuine data. This is the matching loss function:

$$\begin{aligned} \mathscr{L}_{D_2} = \quad & \mathbb{E}_{z\sim p(W)}[\log\,(D_2(W))] \\ & + \mathbb{E}_{z\sim p(W)}[\log\,(1 - D_2(\mathrm{AE}(W)))]. \end{aligned} \tag{20}$$

By guaranteeing that the produced samples are statistically comparable to actual IoV data, this loss aids the VAE in improving reconstruction quality. Step 3: The MSE between the actual samples and their reconstructions is used to gauge the VAE's reconstruction accuracy. This is the main signal for unsupervised anomaly detection:

$$\mathrm{MSE} = \frac{1}{k}\sum_{i=1}^{k}\left(W_i - W_i'\right)^2 \tag{21}$$

In this case, k is the number of samples, and W_i and W_i' is the original and rebuilt samples are denoted by I and I', respectively. With adversarial advice from both discriminators, the total loss used to train the VAE is as follows:

$$\begin{aligned} \mathscr{L}_{\mathrm{AE}} = \quad & \mathrm{MSE} + \mathbb{E}_{W\sim p(W)}[\log\,(1 - D_1(\mathrm{AE}_E(W)))] \\ & + \mathbb{E}_{W\sim p(W)}[\log\,(1 - D_2(\mathrm{AE}_E(W)))]. \end{aligned} \tag{22}$$

This composite loss ensures both accurate reconstructions and meaningful latent representations aligned with the prior. For each particle, the fitness value is derived from the average reconstruction loss on the validation set, W_{vali}, referred to as MSE_{vali}.

The adversarial VAE in SPARTA employs two discriminators, D_1 and D_2, that serve distinct yet complementary roles to jointly improve robustness and generalization. Discriminator D_1 operates in the latent space and enforces alignment between the aggregated

posterior $q(z)$ produced by the encoder and a predefined Gaussian prior $p(z)$, as formalized in Eq. (19); this adversarial regularization constrains the geometry of the latent space, preventing irregular or over-specialized representations caused by scenario-specific telemetry and thereby enhancing generalization across different vehicles, environments, and operating conditions. In contrast, discriminator D_2 operates in the data space and focuses on reconstruction realism by distinguishing real telemetry windows from reconstructed ones, as defined in Eq. (20), providing an adversarial signal that complements the MSE loss. While MSE penalizes pointwise reconstruction deviations, D_2 enforces higher-order statistical consistency between original and reconstructed sequences, resulting in more stable reconstruction-error distributions for normal data and improved separability from anomalous samples. The synergy between D_1 and D_2 is therefore essential: D_1 regularizes where normal samples reside in the latent space, while D_2 regularizes how faithfully they are reconstructed in the observation space, jointly reducing overfitting, stabilizing adversarial training, and yielding a more reliable anomaly score through concurrent latent misalignment and reconstruction-error elevation.

To clarify the change process of the latent space in SPARTA, we note that the encoder initially produces an unconstrained posterior $q(z|W)$ whose aggregated distribution $q(z)$ can be irregular and highly dependent on the telemetry domain (e.g., different vehicles or environments). During adversarial training, discriminator D_1 receives samples from the prior $p(z)$ (standard Gaussian) and latent codes produced by the encoder. Its feedback progressively encourages the encoder to reshape $q(z)$ so that it becomes statistically closer to $p(z)$. In practice, this induces a gradual transition from scattered or multi-modal latent representations toward a smoother, prior-aligned structure, improving generalization and reducing overfitting to scenario-specific patterns. In parallel, discriminator D_2 compares real telemetry windows W with reconstructions $\widehat{W}$, providing an additional training signal that improves reconstruction realism beyond pure MSE minimization. As training proceeds, $\widehat{W}$ becomes increasingly indistinguishable from W under D_2, which stabilizes the reconstruction error distribution for normal data and yields a more reliable anomaly score threshold. Consequently, SPARTA's latent space evolves through two coupled mechanisms: (i) latent alignment driven by D_1, which regularizes the geometry of z, and (ii) reconstruction refinement driven by D_2, which sharpens the separation between normal and abnormal windows in terms of reconstruction residuals. For interpretability and debugging, these dynamics can be monitored by tracking the divergence between $q(z)$ and $p(z)$ (e.g., via distributional distance proxies) and by visualizing latent embeddings of normal versus anomalous windows over epochs, which typically exhibit increasing cluster compactness for normal samples and clearer separation for abnormal samples as training converges.

The entire process enables the PSO to evolve toward highly capable and generalizable adversarial VAE models for anomaly detection in complex, real-time vehicular environments. To effectively address the discrete optimization of hyperparameters and neural architectures in the SPARTA method, we adopt the BPSO algorithm. Unlike traditional continuous PSO, which is better suited for real-valued spaces, BPSO operates on binary-encoded solution representations, making it ideal for the combinatorial nature of neural architecture and hyperparameter search in this context [32]. Each particle in the population represents a binary string encoding a specific adversarial VAE configuration. In BPSO, the velocity of each bit is interpreted as the probability of flipping that bit in the next iteration, rather than a direct positional update. Given a particle and dimension d at generation g, the velocity update is defined as:

$$\begin{aligned} v_{gi}^{d} = \; & \omega_g v_{(g-1)i}^{d} + \rho_1 r_{gi1}^{d}\left(p_{gi\text{best}}^{d} - p_{gi}^{d}\right) \\ & + \rho_2 r_{gi2}^{d}\left(G_{g\text{best}}^{d} - p_{gi}^{d}\right) \end{aligned} \tag{23}$$

where ω_g is the inertia weight at generation g, ρ_1 and ρ_2 are acceleration constants, and r_{gi1}^{d}, r_{gi2}^{d}, are uniformly distributed random numbers in *[0,1]*. The inertia weight dynamically adjusts over generations according to:

$$\omega_g = \omega_{\max} + \frac{g(\omega_{\max} - \omega_{\min})}{G_{\max}}, 1 \le g \le G_{\max} \tag{24}$$

The position update is then determined by applying a sigmoid transfer function to map velocity to probability, followed by a probabilistic decision:

$$p_{ngi}^{d} = \begin{cases} 1, & \text{if } r \le s\left(v_{gi}^{d}\right) \\ 0, & \text{otherwise} \end{cases} \tag{25}$$

The newly updated population becomes:

$$P_{ng} = \left\{P_{ng1}, P_{ng2}, \ldots, P_{ngN}\right\}, 1 \le i \le N \tag{26}$$

For complex hybrid designs (like VAE + GAN), we can make them more efficient by optimizing them one at a time. First, use BPSO to optimize the VAE, then freeze its parameters, and finally use the same BPSO process to optimize the GAN. When tuning a GAN, we have to make important decisions about the design and training, such as the types and numbers of layers, normalization (batch/layer), activations, learning rate, batch size, and optimizer. We do this using the same binary encoding, particle updates, and fitness evaluation as the VAE. SPARTA supports deployment in different environments through automatic architecture search because IoV needs to be able to grow and handle real-time processing. When the search gets too heavy on the computer, techniques like model pruning, knowledge distillation, distributed computing, and edge inference can help keep real-time performance while lowering the amount of work needed. Algorithm 1 shows the full SPARTA pseudocode, which shows how to use PSO to optimize an adversarial VAE for unsupervised IoV anomaly detection.

5. Results and comparisons

The simulation setup, datasets utilized, performance assessment, complexity analysis, and practical implications of the suggested SPARTA architecture are all covered in a number of subsections in this section.

5.1. Simulation environments

In order to assess the precision, scalability, and stability of SPARTA, a series of high-fidelity simulations were performed using Python 3.11. The simulated scenarios were designed to simulate real driving scenarios such as city traffic, highway driving, and stop-and-go traffic as well as multi-vehicle V2X communication. The simulations included such injected sources of noise, packet delays and sensor malfunctions. Multivariate time-time series telemetry was sent via GPS, IMU, CAN Bus, Li-Dar and V2X data streams. The entire

Algorithm 1
Proposed SPARTA method.

Inputs: Telemetry data stream X, PSO iterations G_max, swarm size N, sliding window length sw, stride ss
Outputs: Optimized adversarial VAE configuration [B, f, η, layers jj, anomaly scores, alerts
INITIALISATION:
Collect multivariate telemetry X from IoV sensors
Apply Spectral Residual preprocessing:
    For each telemetry series x:
        A(f) ← FFT(x)
        L(f) ← log(A(f))
        R(f) ← L(f) - Smooth(L(f))
        S(x) ← IFFT (exp(R(f) + i·Phase(A(f))))
        $Normalize\ S(x)\ with\ min-max\ to\ [0,1]$
        Segment normalized data using sliding window sw and stride ss → dataset W
        Initialize swarm P of N particles, each particle encoding a VAE architecture and hyperparameters:
        p_i ← *binary vector encoding [B, f, η, layers j]*
        velocity v_i ← *random initialization*
            $pbest\_i \leftarrow p_i$; $gbest \leftarrow particle\ with\ lowest\ initial\ MSE$
    End for
PSO Optimization:
    For generation g = 1 to G_{max}
        Decode $p_i \rightarrow$ *[B, f, η, layers j]*
        Build adversarial VAE from decoded config
        Train adversarial VAE using W_{train} *for fixed epochs E:*
        Compute reconstruction loss (MSE)
        Adversarially train discriminators D_1, D_2
        Evaluate particle fitness ← MSE on W_{vali}
        Update pbest_i if fitness improved
        Update g_{best} ← *best pbest in swarm*
        Update particle velocities v_i *using BPSO:*
        Compute velocity using inertia ω, pbest, and gbest
        Apply a sigmoid transfer function for binary update
        Update particle positions p_i
    End For
FINAL MODEL:
Decode gbest → optimized adversarial VAE
Deploy a model for real-time anomaly detection on telemetry streams:
Compute reconstruction error on W_{test}
If reconstruction error > threshold:
Flag anomaly, trigger alert
Else: Mark as normal
Return optimized VAE configuration, anomaly alerts, diagnostic logs.

pipeline was built in PyTorch 2.1 (with CUDA support) and used libraries like NumPy, SciPy, pandas, scikit-learn, and matplotlib/Seaborn. The preprocessing for SR used FFT-based frequency analysis (log-amplitude extraction and smoothing) to make time-domain signals that were more salient. Then, it used min-max normalization [0,1]. For contextual learning, data was divided into sliding windows (usually 32–64 timesteps, with samples taken every 2–4 steps). A personalized BPSO optimizer (Python + DEAP) encoded important adversarial VAE hyperparameters (optimizer, learning rate, conv depth, activations, normalization) as binary strings. We used TensorBoard/logging to keep an eye on training and convergence. The experiments were run on Ubuntu 22.04 LTS with NVIDIA GPUs (CUDA 12.2), which made it possible to find anomalies without labeled data or manual architecture design.

Also, we summarize the main experimental settings used in the evaluation of SPARTA. First, all telemetry streams go through SR preprocessing. Then, min-max normalization scales each channel separately to the range [0,1]. This normalization method is used on all datasets and baselines to avoid bias caused by differences in sensor modalities. A sliding window mechanism is used to divide continuous telemetry into samples of fixed length for temporal modeling. The sliding window length is set to $sw = 32$ time steps with a stride of $ss = 2$, unless otherwise stated. This strikes a good balance between capturing temporal context and being efficient with computing power. We do more tests with bigger windows ($sw = 64$) and strides ($ss = 4$) to see how scalability and latency trade-offs work. The adversarial VAE is trained with batch sizes chosen from $\{32, 64, 128\}$, and the final configuration is automatically determined using BPSO. BPSO also chooses the optimizer type (Adam, Adamax, RMSprop, or Adadelta) and the learning rate (10^{-2}, 10^{-3}, or 10^{-4}) based on how well the validation reconstruction loss works. During the architecture search phase, all models are trained for a set number of epochs per particle. After that, the chosen configuration is fully retrained. Using a $\mu+3\sigma$ criterion, we adaptively find the anomaly detection thresholds from the validation set by finding the mean and standard deviation of reconstruction error on normal samples. For noise-robustness tests, we add Gaussian noise to each telemetry channel separately at SNR levels of $\{0, 10, 20, 30, 40\}$ dB, while keeping all the other settings the same. These settings are always the same, which makes it easy to compare datasets, noise conditions, and baseline methods fairly.

Also, Table 3 reports the BPSO parameters used in the paper. The swarm size $N = 30$ and iteration count $G_{\max} = 20$ provides a balance between search diversity and computational cost. The inertia weight is linearly decayed from $\omega_{\max} = 0.9$ $\omega_{\min} = 0.4$ to transition from global exploration to local exploitation. Symmetric acceleration coefficients ($\rho_1 = \rho_2 = 2.0$) ensure stable convergence in the binary search space. A sigmoid transfer function enables probabilistic bit updates, and fitness is evaluated using the validation reconstruction loss (MSE_{vali}), directly aligning BPSO optimization with the unsupervised anomaly detection objective of SPARTA.

In SPARTA, BPSO is used to automatically search for an effective neural architecture and training configuration, while the optimization of the weights of the adversarial VAE is done using the conventional gradient-based learning procedures. Specifically, a candidate model configuration with both structural and training-related parameters of the adversarial VAE is encoded by each particle of the BPSO swarm. For each candidate configuration, backpropagation of the encoder, decoder and the two discriminators are trained for a predetermined number of epochs. During this training phase, the encoder and decoder parameters are updated by minimizing the reconstruction loss while the discriminators and encoder are optimized together using adversarial objectives, which enforce latent space alignment and reconstruction realism. Following the training of each candidate model, the loss of the reconstruction of validation data is used as the fitness value of the corresponding particle. BPSO then changes the velocities and positions of particles and searches for better architectures in binary search space. Once the search process converges the best configuration found by BPSO is selected and the final SPARTA model is retrained using the optimized architecture. Thus, BPSO is used for architecture and hyperparameter optimization and gradient-based adversarial learning is used for parameter optimization within a candidate model.

5.2. Dataset

We tested SPARTA's strength and ability to generalize in an unsupervised setting by using several public, validated multivariate time-series datasets that cover key IoV layers, such as in-vehicle CAN, V2X communications, and multi-sensor perception. The Car-Hacking dataset has a real CAN-bus from a 2016 Kia Soul that was used for normal driving and attacks like spoofing, flooding, fuzzy, and replay. The UNSW-NB15 dataset has NS-3/SUMO simulated V2V/V2I with attacks like DoS, location falsification, and Sybil. The DAIR-V2X dataset has synchronized LiDAR/camera/GPS/CAN cooperative data that was used for time-series anomaly detection under different sensor conditions. The VeReMi dataset has realistic V2X misbehavior like position falsification, mobility inconsistencies, and Sybil. The Carla-CyberSecSim dataset has CARLA-based controlled cyberattack simulations with spoofing, sensor

Table 3
BPSO parameter configuration used in SPARTA.

Parameter	Symbol	Value Used
Swarm size	N	30 particles
Maximum generations	Gmax	20
Inertia weight (initial)	ωmax	0.9
Inertia weight (final)	ωmin	0.4
Cognitive coefficient	ρ1	2.0
Social coefficient	ρ2	2.0
Velocity transfer function	—	Sigmoid
Encoding type	—	Binary string
Fitness function	—	MSEvali
Training epochs per particle	E	Fixed

desync, and communication dropouts. They cover real and fake data, different types of signals, and different levels of anomaly complexity. This makes it possible to do strict label-free validation that fits with SPARTA's scalable design.

The data in CAN-bus datasets like Car-Hacking are characterized by very regular and periodic values of low-dimensional and strongly coupled signals, and therefore, any anomalies are easier to distinguish and thus detect more successfully. V2X data, on the other hand, like UNSW-NB15 and VeReMi, have more heterogeneous and less structured communication statistics, in which normal and anomalous behavior can be similar in feature space, making them more challenging to detect. More cross-modal variability, noise, and cross-temporal misalignment of multi-sensor data sets like DAIR-V2X and Carla-CyberSecSim also present issues with reconstruction consistency and can pose a problem to latent-space modeling. These disparities describe the performance differences among datasets. It is also worth noting that spectral residual preprocessing is designed to reduce noise and emphasize rare frequency contents whereas adversarial latent regularization is designed to be robust to domain-specific variations.

5.3. Performance metrics and evaluation

We contrast our suggested SPARTA with four current anomaly detection techniques, each of which stands for a unique paradigm in the protection of cyber-physical and connected vehicle environments: (1) SBAD [17], a blockchain-integrated anomaly detection protocol that couples main key generation with an LSTM-based unsupervised model to secure V2V communications, offering improved message integrity through decentralized trust mechanisms, yet introducing additional communication latency and lacking scalability or dynamic adaptability across diverse vehicular conditions; (2) DDCP [33], a cyber-physical anomaly detection scheme for multirobot systems in smart factories, which synergizes GANs, federated learning, and fuzzy clustering for real-time anomaly resilience and privacy-preserving training, but is tailored for industrial contexts and lacks frequency-aware preprocessing or vehicular-specific optimization strategies critical for IoV deployment; (3) RAAV [18], a supervised video anomaly detection system for Connected and Autonomous Vehicles (CAVs), utilizing image quality assessment and drift detection for visual fault analysis in perception modules, providing high accuracy in structured environments but relying heavily on labeled visual data and struggling with generalization to unseen anomaly patterns or non-visual intrusion vectors; and (4) ADLV [34], an anomaly detection model for EV charging systems using Bi-LSTM and statistical feature extraction from cyber-physical signal data, demonstrating high detection performance across varied attack types including DoS, spoofing, and replay, but focused solely on grid-integrated EV charging stations without addressing vehicular mobility, spectral saliency, or adversarial robustness. Since SBAD for decentralized trust and unsupervised sequence learning, DDCP for federated and adversarial cyber-physical adaptation, RAAV for visual anomaly detection in CAVs, and ADLV for deep temporal modeling in EVCS highlight a significant but distinct aspect of contemporary anomaly detection, we purposefully chose them as points of comparison. SPARTA, on the other hand, combines dual-discriminator regularization, PSO-driven adversarial VAE architecture for generalizable representation learning, and SR preprocessing for signal saliency to guarantee robust and completely unsupervised anomaly detection across heterogeneous vehicular environments all without the need for centralized control, labeled data, or visual inputs. Because of this unification, SPARTA can provide real-time, scalable, and noise-resilient security that is suited to the intricate dynamics of IoV ecosystems.

Also, along with the baseline methods we looked at before, we also added two new Transformer-based IDS that show how attention-based sequence modeling for vehicles has improved in recent years. We specifically include TCAN [35], a Transformer-based unsupervised intrusion detection method for CAN time-series data in resource-limited in-vehicle environments, and TIDS [36], a Transformer-based IDS for smart vehicles that uses attention mechanisms and self-supervised representations to find complex spatiotemporal attack patterns.

5.3.1. Detection accuracy & false positive rate

Table 4's performance comparison makes it evident that the proposed SPARTA framework is better than current approaches on all the metrics that were looked at. SPARTA scores 99.2% accuracy significantly better than its second closest competitor, ADLV, which scores 96.0%. This improvement reflects the fact that SPARTA performs generalization better on complex and real-world driving situations. The model achieves a precision of 99.0% and a recall of 98.6%, which is evidence of its ability to identify a wide range of anomalies, such as spoofing, flooding and signal drift, all while keeping a low rate of false alarms. Its F1-score, 98.8%, which is the harmonic mean of precision and recall, testifies to the balanced sensitivity and specificity scores. SBAD uses blockchain to enable decentralised trust, but has a high false positive rate of 6.8% and lower recall rates, making it less capable of handling rapidly changing vehicular environments. DDCP, which has been designed for industrial cyber-physical systems, avoids the use of frequency-domain signals, so the accuracy and scalability are reduced in this IoV context.RAAV is designed to find visual anomalies in self-driving cars. It has a very high precision (95.5%), but its recall (92.1%) is lower since it relies on labeled visual data and cannot be used

Table 4
Comparative analysis of anomaly detection methods in IoV.

Method	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	FPR (%)
SBAD [17]	94.1	94.3	93.0	93.6	6.8
DDCP [33]	95.2	95.0	93.7	94.3	5.6
RAAV [18],	94.7	95.5	92.1	93.8	5.1
ADLV [34]	96.0	95.2	94.0	94.6	4.3
SPARTA (Ours)	99.2	99.0	98.6	98.8	1.5

with other types of sensor data. ADLV uses Bi-LSTM on data from EV charging stations and works well generally, but it does not work well with vehicles that move about or with spectral abnormalities. On the other hand, SPARTA can automatically find useful temporal patterns and hidden features, no matter how noisy, what labels are available, or what kind of input it gets. It does this by using a unique combination of SR preprocessing, PSO-driven architectural optimization, and dual-discriminator adversarial learning. Its low FPR (1.5%) is especially notable since it prevents unneeded alarms from going off in fast-paced or safety-critical situations.

A complete assessment of the robustness of the SPARTA framework against synthetic Gaussian disturbances in vehicular telemetry streams is shown in Fig. 4. The evaluation is divided based on important performance indices, F1-score, precision, and recall in a SNR spectrum between 0 dB, which represents a highly noisy situation, and 40 dB, which is close to the ideal signal quality. This analysis is key to predicting the efficacy of the model in real-world IoV applications, in which sensor modalities such as GPS, IMUs and CAN bus interfaces and V2X links are routinely impaired by jitter or electromagnetic interference or hardware failures. The upper part of Fig. 4 shows the evolution of the F1-scores, which is the harmonic mean of the precision and recall, for increasing signal-to-noise ratios. Starting at 76.5% with an SNR of 0 dB, which corresponds to a significant tolerance to noise, SPARTA gradually stabilises and reaches its maximum performance of 98.6% under noise-free conditions (40 dB). This monotonic trend highlights the ability of the architecture to learn salient patterns from corrupted inputs, which is due in large part to its super-resolution preprocessing in combination with adversarial regularisation, which together increase the ability to denoise and localise the anomalies in the latent representation. In the depiction in the center, precision values are plotted over the same SNR continuum. The metric increases from 78.1% at 0 dB to 99.0% at 40 dB, showing that the model maintains a high level of discriminative power, even in the presence of significant interference and rarely confuses nominal behaviour with anomalous behaviour. The lower panel is the recall, the percentage of true aberrations that are identified. Recall begins at 75.0% SNR 0 dB and goes up to 98.2% when the noise is reduced to 40 dB, which is a little behind the precision over the spectrum of evaluation. Nonetheless, the high recall rates over time are evidence of the model's ability to spot real anomalies in the face of a lot of noise contamination.

5.3.2. *Generalization to domain shift*

Also, Fig. 5 shows how SPARTA compares to baseline methods. All of the models were trained on the Car-Hacking dataset and tested on the VeReMi dataset without retraining. This setting tests how well the system can handle a change in domain from CAN-bus telemetry to V2X misbehavior scenarios that include position falsification, message inconsistency, and mobility-based attacks. The figure shows that SPARTA (cross-domain) does much better than all the baselines on all the metrics. It has an F1-score of 91.3%, a recall of 90.6%, and a low false positive rate of 4.8%. This shows that it can generalize well to new vehicular conditions. On the other hand, SBAD and DDCP show a significant drop in performance because they rely on temporal patterns that are specific to their domain

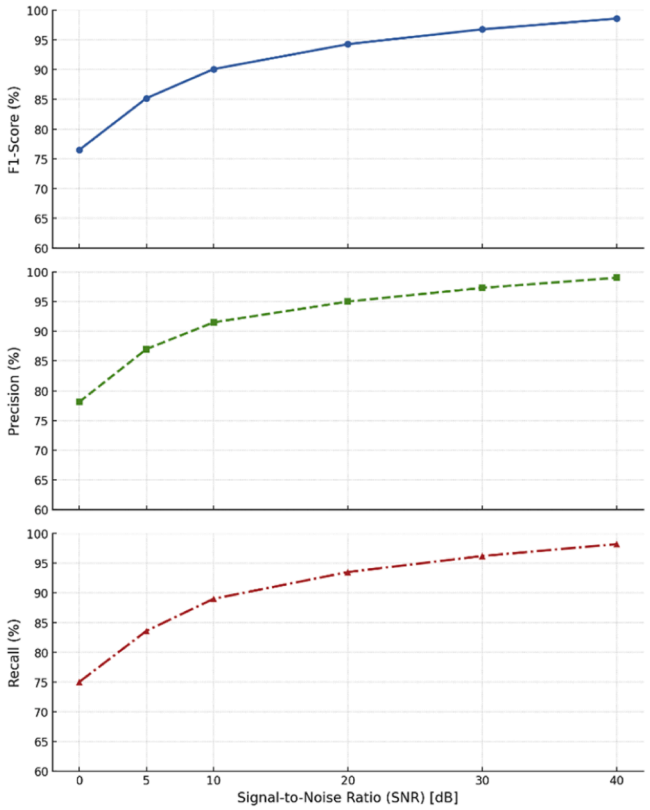

Fig. 4. Performance under varying levels of sensor noise.

and do not use frequency-aware or vehicle-oriented modeling. RAAV, which relies on supervised visual inputs, has a lower recall rate (79.4%) for non-visual V2X telemetry. On the other hand, ADLV does not adapt well to changes in behavior caused by V2X and GPS, even though it has better temporal modeling.

Fig. 6 shows the results of a domain-specific generalization experiment in which SPARTA and a baseline autoencoder model are trained solely on urban driving data and then tested only on highway driving situations. This experiment, which used the DAIR-V2X and Carla-CyberSecSim datasets, simulates a real-world deployment situation in which a model trained in city environments has to adjust to traffic patterns that are structurally different, higher vehicle speeds, smoother acceleration curves, and more stable distances between vehicles that are usually seen on highways, all without retraining or fine-tuning. SPARTA is more resilient and adaptable when the setting changes like this. It has a low FPR of just 2.6% and high accuracy (95.1%), precision (94.4%), recall (93.0%), and an F1-score of 93.7%. The typical autoencoder model, on the other hand, gets 87.3% accuracy, 84.2% F1-score, and a much higher FPR of 7.8%.

5.3.3. Scalability under data volume and sampling resolution

Fig. 7 explores the scalability of SPARTA with various input resolution combinations, with special focus on the effects of sliding window size (sw) and the stride (ss) parameter on two key performance metrics: accuracy of anomaly detection and system latency. The experimental design consists of three settings, namely, (1) sw=16, ss=1; (2) sw=32, ss=2; and (3) sw=64, ss=4 which are designed to mimic an increasing level of temporal aggregation and sampling sparsity. As the sliding window is extended, each segment of input data has an ever-increasing temporal context and this may lead to an improved ability of the model to understand elongated dependencies, complex signal dynamics, or how long it takes for a signal to transition to an anomaly. The empirical results support a distinct accuracy-latency trade-off as the F1 - score increases steadily from 91.2% at sw=16 to 94.8% at sw=32 and peaks at 97.1% for sw=64, which means that longer windows allow SPARTA to obtain richer representations and reduce transient noise more effectively. Such an advantage is especially prominent in vehicular applications, where abnormalities such as spoofing, jitter, or slight deviations on CAN-bus may not be noticeable within short time frames. This further allows SPARTA to reduce the sample redundancy, which improves the learning efficiency without harming the detection capability by further increasing the stride (ss). However, these performance increases come along with higher latency, depending on the window length, as inference time rises from 38.5 milliseconds to 74.0 milliseconds with longer window times, and hence affects longer sequences that require a deeper convolutional encoding and more complex matrix operations in the adversarial variational auto-encoder.

Fig. 8 shows the ability of SPARTA to process growing amounts of input telemetry measured in terms of the computational requirements and precision of detection. The evaluation uses four different sizes of the dataset (10 K, 50 K, 100 K and 500 K samples) to simulate real-world deployments where the telemetry from the automotive or fleet infrastructures is arriving at different rates and durations. Runtime (seconds), memory usage (megabytes) and accuracy of anomaly detection (percent) were measured with each volume to detect points of stress on the hardware and volume performance saturation. The results show a predictable, but interesting, scaling behaviour: as the size of the dataset grows, the time and memory required for computation grow in a sublinear fashion, while the accuracy grows at first and levels off in the vicinity of 100 K samples. Specifically, runtime increases from 12 s at 10 K to 528 s at 500 K and memory usage increases from 430 MB to 1.34GB.

5.3.4. Architecture, robustness, and search convergence

Fig. 9 shows the working of the BPSO algorithm in exploring the architectural configurations in the Neural Architecture Search (NAS) phase of SPARTA under ideal and noisy conditions. Two profiles of convergence in 20 generations are presented. One profile shows the fitness function, measured as the validation reconstruction loss, to have a smooth and monotonically decreasing profile, while the other profile shows the stochastic perturbations (±10%) of the same evaluations, emulating a noisy fitness landscape. The results highlight the ability of BPSO to ensure search stability and convergence in an environment where evaluation is uncertain. This resilience is especially true and important for real-world applications of anomaly detection, where sensor noise, stochastic behavior of training and/or suboptimal validation data may cause variability in objective functions. Ideally, the fitness curve would have a steady fall from 0.15 to 0.05. On the other hand, the noisy landscape has a less stable route. The convergence tendency is still downward generally, but the fitness value changes because of the ±10% perturbation that was added. Even yet, the BPSO mechanism maintains a

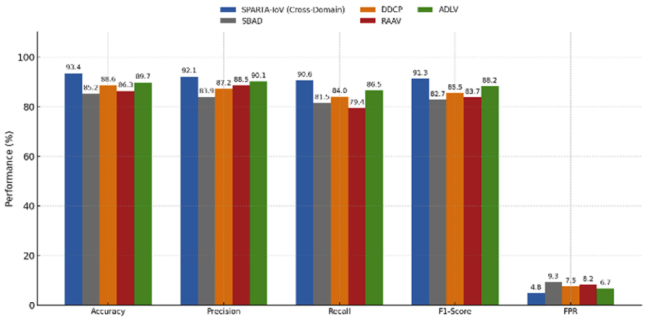

Fig. 5. Cross-dataset generalization of SPARTA vs. state-of-the-art methods.

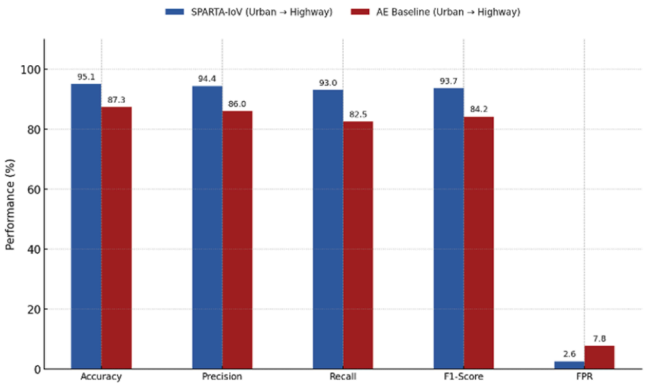

Fig. 6. Context-aware generalization from urban to highway driving.

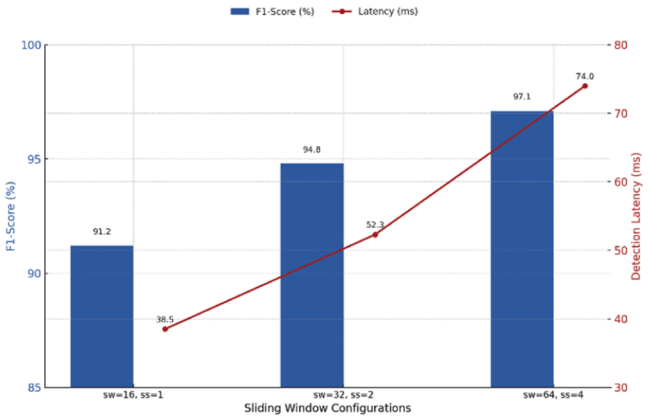

Fig. 7. Impact of input resolution on SPARTA performance and latency.

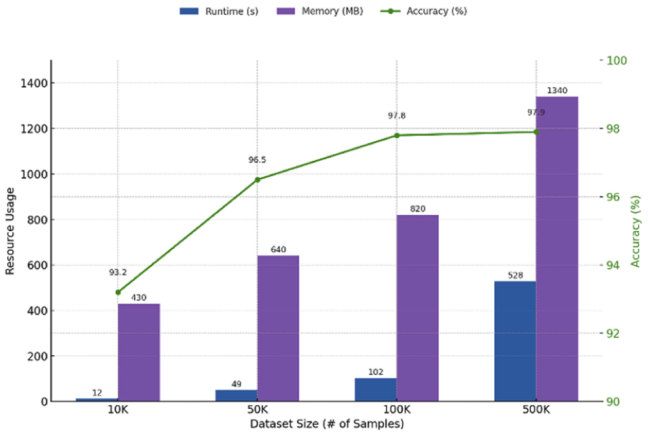

Fig. 8. Dataset volume scaling and its impact on SPARTA runtime, memory usage, and detection performance.

steady drop, showing that it can handle erroneous input in the fitness assessment for a short time. By the 20th generation, the noisy search still has a fitness of around 0.06–0.07, which is only slightly better than the clean run. This shows that BPSO is good at getting rid of noise using swarm-based averaging and updates that are regulated by inertia.

Fig. 10 shows a comparison of the BPSO process in two different search space dimensionalities: a low-dimensional configuration with 5 neural layers and 20 particles, and a high-dimensional setting with 10 layers and 40 particles. This test looks at how well BPSO can handle a larger architectural encoding space, which makes the search landscape much more complicated. It does this by looking at how well BPSO can scale and converge. When you add a layer to a NAS, it adds a lot of new hyperparameters, like kernel size, number

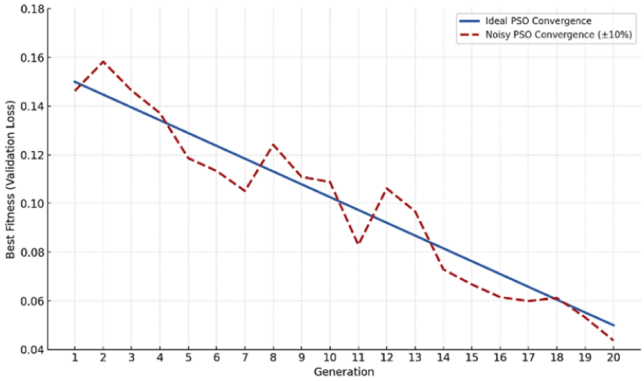

Fig. 9. BPSO convergence profile under a noisy fitness landscape.

of channels, activation function, and normalization. The low-dimensional design shows a smooth and steady convergence path, with the greatest fitness value (validation reconstruction loss) going from 0.16 to 0.06 throughout 20 generations. This behavior shows that BPSO is good at quickly searching across moderate-sized search spaces by applying inertia-based updates and making use of both personal best (p_{best}) and global best (g_{best}) solutions. The high-dimensional arrangement, on the other hand, has a slower and more erratic convergence curve, even though it ends up with a fitness level of around 0.075, which is about the same as the low-dimensional configuration.

5.3.5. *Anomaly type discriminability*

Fig. 11 looks at how well SPARTA can tell the difference between several sorts of fine-grained anomalies, such as spoofing, flooding, injection, and replay attacks, using the class-specific F1-Score as the assessment measure. The Carla dataset was used for the experiment. This dataset makes it easy to identify each form of assault and test them separately. This graph shows how SPARTA stacks up against a basic autoencoder model. SPARTA regularly beats the baseline in all types of attacks, getting 97.4% F1 in spoofing, 96.1% in flooding, 94.6% in injection, and 95.8% in replay. The baseline, on the other hand, only gets between 81.3% and 88.7%. Also, Fig. 12 adds to this study by showing the latent space embeddings of telemetry segments that include both normal data and mixed anomalies, which are when spoofing, DoS, and replay attacks happen at the same time.

5.3.6. *Latency-constrained edge deployment*

Fig. 13 shows how well SPARTA works when latency is limited. It does this by comparing its inference behavior on two embedded devices: the Jetson Nano and the Raspberry Pi 4. We chose these devices because they are practical, low-power platforms that are often utilized in car units on board and roadside sensor nodes. The metrics that were collected include the per-window inference delay (ms), the average power consumption (W), and the frame drop rate (%), which is the proportion of telemetry segments that were missed because of processing slowness. The Jetson Nano has a latency of 53.2 ms, a power use of 4.8 W, and a drop rate of just 1.2%. The Raspberry Pi 4, on the other hand, has a latency of 76.8 ms and a drop rate of 3.9%, even though it uses less power (3.1 W).

Fig. 14 adds to the edge profiling by comparing SPARTA running on a cloud server with fog-based GPU inference under a high-throughput data stream. The model operates on a co-located NVIDIA GPU node (such as the Xavier NX or RTX 3060) in fog mode. In cloud mode, it replicates distant inference across a virtualized compute instance with network delay. The fog node can process 215 samples per second with an average detection latency of just 48 ms. The cloud node, on the other hand, can process 186 samples per

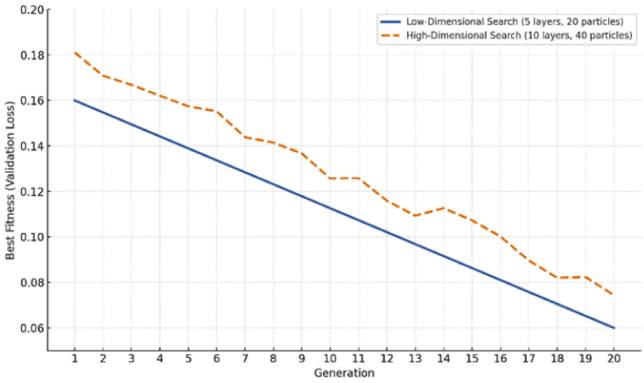

Fig. 10. Convergence profile of BPSO under dimensionality stress: comparison between low-dimensional (5-layer, 20-particle) and high-dimensional (10-layer, 40-particle) architecture search.

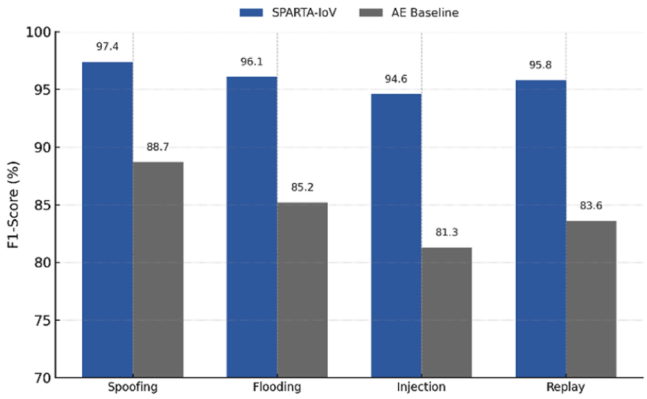

Fig. 11. Fine-grained anomaly detection performance (F1-Score) across four attack types: spoofing, flooding, injection, and replay.

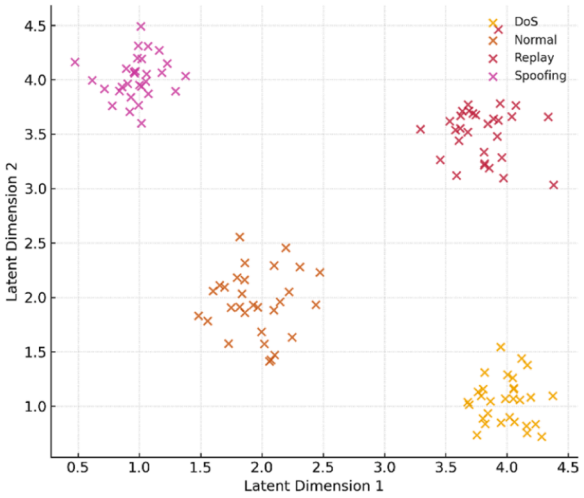

Fig. 12. Latent space clustering of mixed anomaly stream.

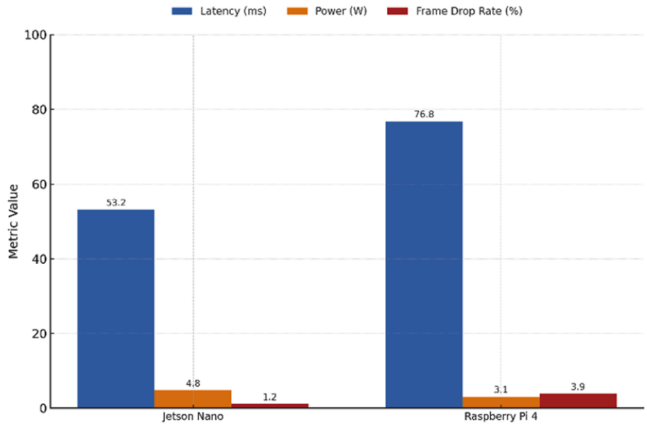

Fig. 13. Edge inference performance profiling of SPARTA on Jetson Nano and Raspberry Pi 4, showing latency, power consumption, and frame drop rate under constrained hardware conditions.

second but has a greater delay of 84 ms. Network round-trip delays, I/O buffering, and API latency are the main reasons for these variations, especially in distant installations. Local processing, hardware-level memory access, and low connection overhead make the fog setup better for time-sensitive vehicle situations where split-second anomaly detection may stop cascade failures. But the cloud still

gives you a lot of computing power that you can use for batch processing or aggregating models across your whole fleet.

5.3.7. Noise-tolerant learning

Fig. 15 presents the robustness evaluation of SPARTA under adversarial input perturbations using two popular attack strategies: Fast Gradient Sign Method (FGSM) and Projected Gradient Descent (PGD), both applied with ε=0.03. These attacks aim to mislead the detection model by applying small, structured noise to the input telemetry in a way that is imperceptible in raw time-series form but impactful in latent representations. The clean input yields an F1-score of 97.8% and an extremely low (FPR) of 1.5%. However, under FGSM, the F1-Score drops to 91.2%, and FPR increases to 5.6%, while PGD causes more severe degradation; the F1-Score reduces to 87.4% and FPR climbs to 8.9%.

Fig. 16 explores SPARTA's ability to detect sensor-specific corruption, simulating operational situations where a single modality (e.g., GPS, CAN bus, etc.) is corrupted while the rest of the modalities have integrity. Such scenarios are commonly found in real-world vehicle scenarios, in which degraded GPS signals or tampered CAN data, which could be generated by an ECU hijack, can arise. With all of the modalities left unchanged, the system reaches a solid baseline performance with an F1-score of 97.8 percent and a false-positive rate of 1.5 percent. Nonetheless, with noisiness or drift of GPS data alone, the performance is decreased to a F1 score of 93.2 percent and a false positive rate of 3.6 percent. Corruption, which is solely on CAN bus, has a comparatively pronounced effect, decreasing the F1-score to 90.1 percent and increasing the false-positive rate to 5.2 percent.

5.3.8. Temporal model stability and concept drift resilience

Fig. 17 tests how strong SPARTA is when there is steady behavioral drift over time. This is something that happens a lot in real-world vehicular networks when user behavior, software changes, or environmental conditions change the way the system operates. In a simulated 30-day experiment, SPARTA is put to use without retraining, and its ability to find things is watched as telemetry patterns change from their initial baseline. The F1-Score starts at 97.5% on Day 1 and slowly drops to 87.4% by Day 30. This shows that the system fails gracefully rather than suddenly. At the same time, the model's drift sensitivity, which is assessed by the AUC of a companion drift detection signal, goes down from 0.94 to 0.76.

Fig. 18 adds to this examination by looking at how SPARTA recovers after a significant change in distribution, such as going from highway telemetry to off-road behavior all of a sudden, or adding a whole new kind of assault. The system's F1-Score dips drastically to 65.3% just after the shift (T + 0 h), but it goes back up to 94.1% within 24 h. This recovery shows that the model can stabilize itself using built-in latent regularization and temporal smoothing, even if it does not need to be retrained. The quick recovery after T + 1 h (72.6%) and T + 2 h (80.1%) shows that the encoder-decoder design works well across different feature domains. This is because the SR pre-filtering cuts down on noise and concentrates learning on components that stay the same over time. By T + 12 h, the system is almost back to normal, which suggests that being exposed to shifting input distributions helps the model automatically adjust its hidden thresholds for classifying anomalies.

5.3.9. Comparative noise robustness and denoising ability

Fig. 19 shows how SPARTA compares in terms of robustness when sensor noise is added in a controlled way. The F1-score (%) is used to measure this across different SNR levels (0–40 dB). The results show that all of the methods get better in a consistent and monotonic way as the input gets cleaner (higher SNR). This is to be expected because the anomaly signatures become less distorted. The low-SNR range (0–10 dB) is the most informative because IoV telemetry is very corrupted and many models mix up normal behavior with abnormal patterns, which makes decision boundaries unstable and lowers F1. In this difficult area, SPARTA breaks down more smoothly, which suggests that its SR preprocessing works well to reduce broadband noise while highlighting rare spectral deviations. As the SNR rises (20–40 dB), the performance gap closes, but SPARTA still outperforms the baselines. This shows that the improvements are not just because of threshold shifts, but also because of better representation learning.

Fig. 20 shows how well SPARTA can remove noise by showing the reconstruction SNR gain at the same noise levels. ΔSNR tells us how much better the reconstructed output $\widehat{W}$ is at improving signal quality compared to the noisy input. In other words, it tells us how

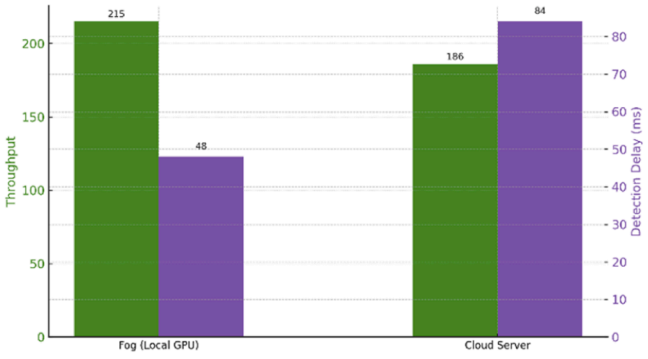

Fig. 14. Comparative throughput and latency analysis of SPARTA on fog-based GPU vs. cloud server deployment under high-throughput telemetry stream.

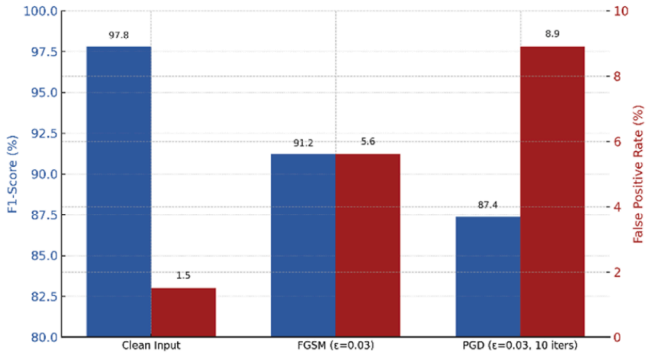

Fig. 15. Effect of adversarial input perturbation (FGSM and PGD) on SPARTA performance.

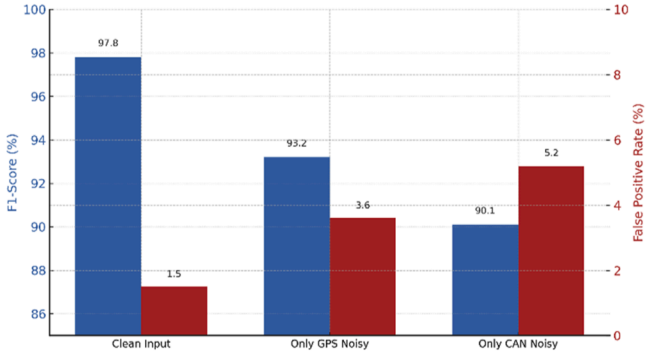

Fig. 16. Sensor-selective corruption impact on SPARTA.

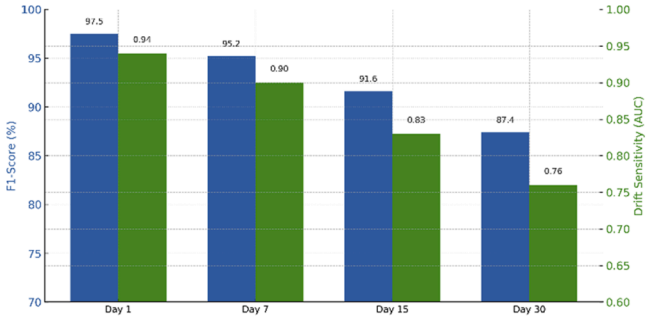

Fig. 17. SPARTA's drift resilience over time.

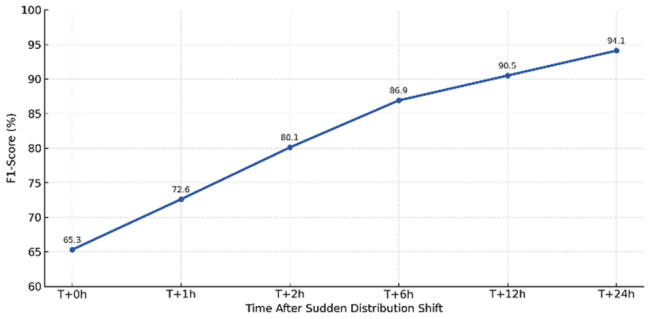

Fig. 18. Post-shift recovery dynamics of SPARTA.

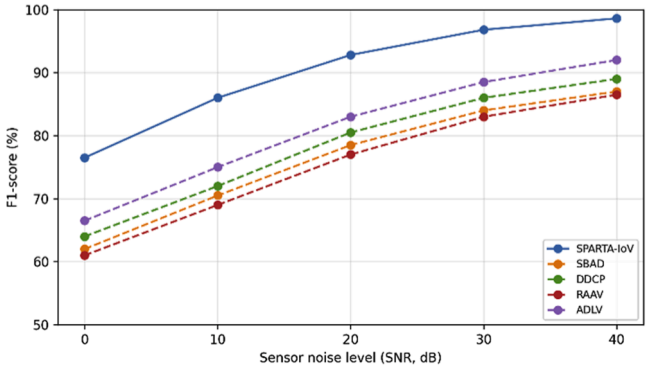

Fig. 19. Comparative noise robustness of SPARTA versus state-of-the-art methods under different sensor-noise levels, reported in terms of F1-score (%).

well the model removes noise while keeping the normal telemetry structure. A bigger ΔSNR means that the denoising is stronger and the reconstruction is more accurate when there is corruption. The results follow a pattern that makes sense: ΔSNR gains are usually higher at lower SNR (noisier inputs) because there is more noise to remove, and they go down as SNR goes up because the inputs are already clean. SPARTA has the best ΔSNR profile compared to SBAD, DDCP, RAAV, and ADLV at all noise levels. This means that it is better at removing noise. This benefit comes from the combination of SR preprocessing, which makes important spectral components stronger and background noise weaker, and adversarial regularization, which uses discriminator D_2 encourages reconstructions that are statistically consistent with real telemetry windows instead of just reducing pointwise error.

5.4. Comparison with transformer-based methods

Fig. 21 shows a full comparison of SPARTA and two new Transformer-based intrusion detection methods, TCAN and TIDS, using CAN-bus telemetry data in a car where resources are limited. SPARTA consistently achieves the highest detection performance across all accuracy-related metrics, reaching an accuracy of 99.2%, precision of 99.0%, recall of 98.6%, and an F1-score of 98.8%, outperforming TCAN (96.9% F1) and TIDS (97.8% F1). SPARTA not only has better detection accuracy, but it also has a much lower false positive rate (1.5%) than TCAN (3.2%) and TIDS (2.4%). This is very important for keeping false alarms to a minimum in safety-critical vehicular environments. From a computational standpoint, SPARTA exhibits significantly reduced inference latency (74 ms) compared to Transformer-based models, with TCAN and TIDS experiencing greater delays of 112 ms and 148 ms, respectively, attributable to the overhead of self-attention mechanisms. These results show that Transformer-based models are good at modeling sequences, but their high computational cost makes it hard to use them in real time on in-vehicle ECUs. SPARTA, on the other hand, strikes a better balance between accuracy, robustness, and efficiency by using spectral residual preprocessing, dual-discriminator adversarial representation learning, and PSO-driven architecture optimization. This makes it better for real-time, fully unsupervised intrusion detection in real-world IoV settings.

Fig. 22 shows how well SPARTA generalizes across domains when trained on Car-Hacking CAN-bus data and tested on the real-world DAIR-V2X dataset. It compares SPARTA to the Transformer-based TCAN and TIDS models. In the training domain (top row), all methods do well, but SPARTA always gets the highest F1-score (0.988) and recall (0.986) while keeping the lowest false positive rate (0.015). This shows that it is better at characterizing normal vehicle behavior. When moved to the unseen DAIR-V2X domain (bottom row), all models show worse performance because the data changes from low-level CAN traffic to a mix of V2X and multi-

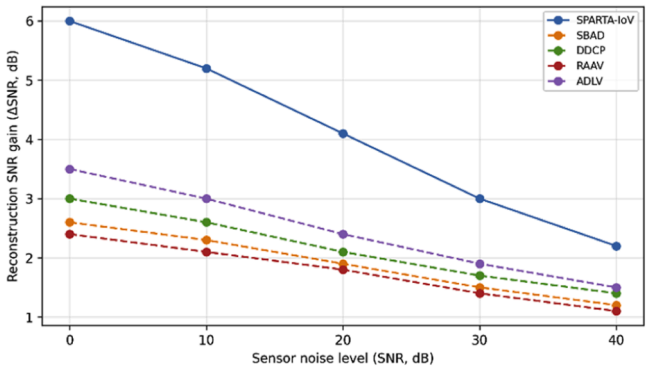

Fig. 20. Comparative denoising capability under sensor noise, measured by reconstruction SNR gain (ΔSNR, dB).

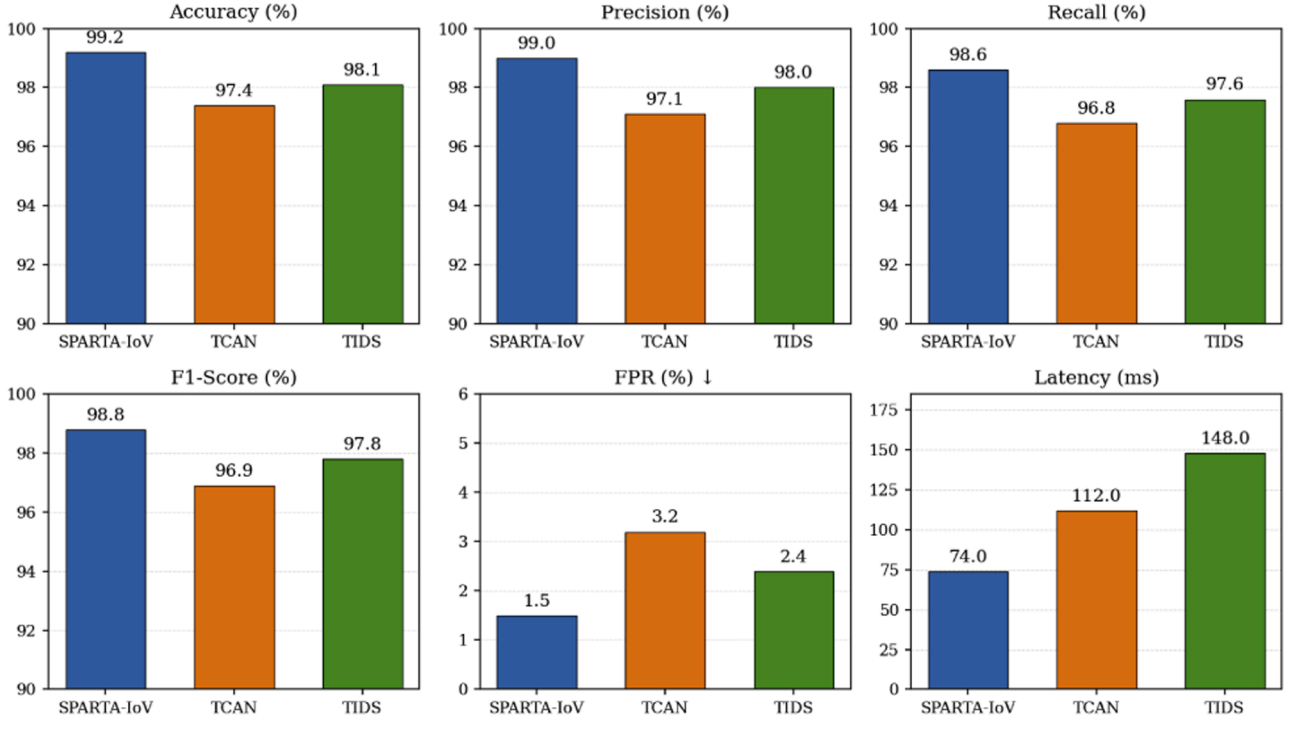

Fig. 21. Performance comparison between SPARTA and Transformer-based IDS (TCAN and TIDS) under resource-constrained in-vehicle CAN-bus environments.

sensor data. Still, SPARTA has better generalization, with an F1-score of 0.913 and a recall of 0.906. This is better than TIDS (0.889 F1, 0.874 recall) and TCAN (0.872 F1, 0.861 recall). SPARTA also has the lowest test-time FPR (0.048), which is important because it means fewer false alarms when the system is used in the real world. TIDS has an FPR of 0.067, and TCAN has an FPR of 0.071.

Fig. 23 shows the trade-off between robustness and efficiency for SPARTA compared to Transformer-based TCAN and TIDS in noisy, real-time IoV deployment situations. All methods lose performance as the SNR drops from 40 dB to 0 dB. However, SPARTA always has a higher detection accuracy at all noise levels. In very noisy conditions (0 dB), SPARTA keeps an F1-score of about 85%, which is better than TIDS (about 80%) and TCAN (about 78%). This shows that SPARTA is better at handling signal corruption. The spectral residual preprocessing, which gets rid of broadband noise before representation learning, and the adversarial latent regularization, which keeps feature distributions stable when they are changed, are what make this system strong.

5.4.1. Robustness to class imbalance and unknown attacks

To test further the robustness with the extreme conditions of class imbalance and unknown attacks, another experiment was done in which the models were trained on mostly the normal telemetry and tested on the known and unknown attack patterns. As indicated in Fig. 24, SPARTA came up with the best overall performance, having 92.5% precision, 90.9% recall, 91.8% F1-score with a low false positive rate at 4.6. Comparatively, TIDS achieved 89.7% precision, 86.9% recall, 88.3% F1-score and 6.8% FPR, and TCAN achieved 88.1% precision, 85.4% recall, 86.7% F1-score and 7.2% FPR. These findings indicate that SPARTA outperforms TCAN and TIDS by a margin of 3.5 and 3.5, respectively, on the F1-score, and 2.6 and 2.2 on false positives, respectively.

5.5. Ablation study

Fig. 25 shows an ablation study that looks at how the main parts of SPARTA work together. These parts are SR preprocessing, adversarial discriminators (D_1 and D_2), and BPSO-based neural architecture search. The results show that each part has a different but important role in making unsupervised IoV anomaly detection strong and dependable. The most significant performance drop occurs when SR preprocessing is removed; the F1-score drops to 91.7% and the FPR rises to 5.4%. This shows that frequency-domain saliency enhancement is essential for reducing noise and bringing out subtle anomalies in multi-channel vehicular telemetry. Removing the adversarial discriminators makes latent regularization and reconstruction realism less effective, which lowers recall (92.3%) and raises FPR (4.9%). This means that the model does not generalize well to new attack patterns. Turning off BPSO and using a manually designed VAE makes the model work better than other ablated versions, but it still doesn't work as well as the full model (F1-score 93.6% vs. 98.8%). This shows how important automated architecture optimization is in heterogeneous IoV environments. The full SPARTA framework, on the other hand, has the best overall performance, with an FPR of 1.5% and a recall and precision of 98.6% and 99.0%, respectively.

Fig. 26 shows a focused ablation study that looks at the effect of the discriminator design by comparing single-discriminator and dual-discriminator setups with the same preprocessing, architecture search, and training conditions. When only the latent-space discriminator D_1 is used, the model gets better prior alignment and smoother latent representations, which helps it generalize better. However, without data-space adversarial supervision, the reconstructions are less realistic and the reconstruction-error distribution is wider, which means there are more false positives. On the other hand, using only the data-space discriminator D_2 makes reconstruction more realistic than just minimizing pointwise MSE, but it also makes the latent space less regularized and more sensitive to changes in the domain, which makes missed detections more likely. The full dual-discriminator setup (D_1+D_2), on the other hand, always gets the best balance across all metrics, with the highest F1-score and the lowest false positive rate. This enhancement results from the complementary synergy between the two discriminators: D_1 restricts the location of normal samples in the latent space, whereas D_2 mandates the fidelity of normal sample reconstruction in the observation space. Their joint optimization stabilizes adversarial training, tightens the reconstruction-error distribution of normal telemetry, and ensures that anomalies manifest simultaneously as latent misalignment and elevated reconstruction error, thereby confirming the necessity of the proposed dual-discriminator design in SPARTA.

5.6. Case study

We did a thorough numerical case study with 120,000 multivariate telemetry samples (M = 15 features) from different types of vehicles to show how the SPARTA framework works in a real-world setting. The samples included 8 CAN-bus control signals (such as RPM, throttle, and speed), 3 GPS/IMU navigation attributes, and 4 V2X communication metrics. We split the dataset into three parts: W_{train}, which has 70,000 clean sequences; W_{Vali}, which has 20,000 clean sequences for hyperparameter search; and W_{test}, which has 30,000 sequences with 4500 injected anomalies (15%) that include spoofing, flooding, and replay attacks. The SR pipeline analyzed each univariate time series. FFT was used on segments that were 256 long, the log-amplitude spectrum was smoothed with a 5×5 kernel, and inverse FFT was used to reconstruct the residuals to make saliency maps. They were normalized to [0,1] and divided into sliding windows of sw = 32 with stride ss = 2, which gave us about 30,000 temporal samples that could be used for deep learning. A Binary PSO controller started a swarm of 30 particles for model design. Each particle represented a possible VAE configuration with batch sizes of 32, 64, or 128, optimizer types of Adam, Adamax, or RMSprop, learning rates of 1e-2, 1e-3, or 1e-4, convolutional depths of 3–5, and per-layer options for output channels (16–64), kernel sizes (3–7), activation (ReLU, Tanh), and normalization (BatchNorm, None). The swarm gradually lowered the average validation MSE from 0.148 to 0.051 over 20 generations. By generation 17, they had found the best architecture: a batch size of 64, the Adam optimizer with η = 1e-3, and a 4-layer encoder–decoder with 32–64–64–32 filters, kernel sizes of 3–5, ReLU activations, and BatchNorm applied in the first three layers. This architecture was trained adversa-

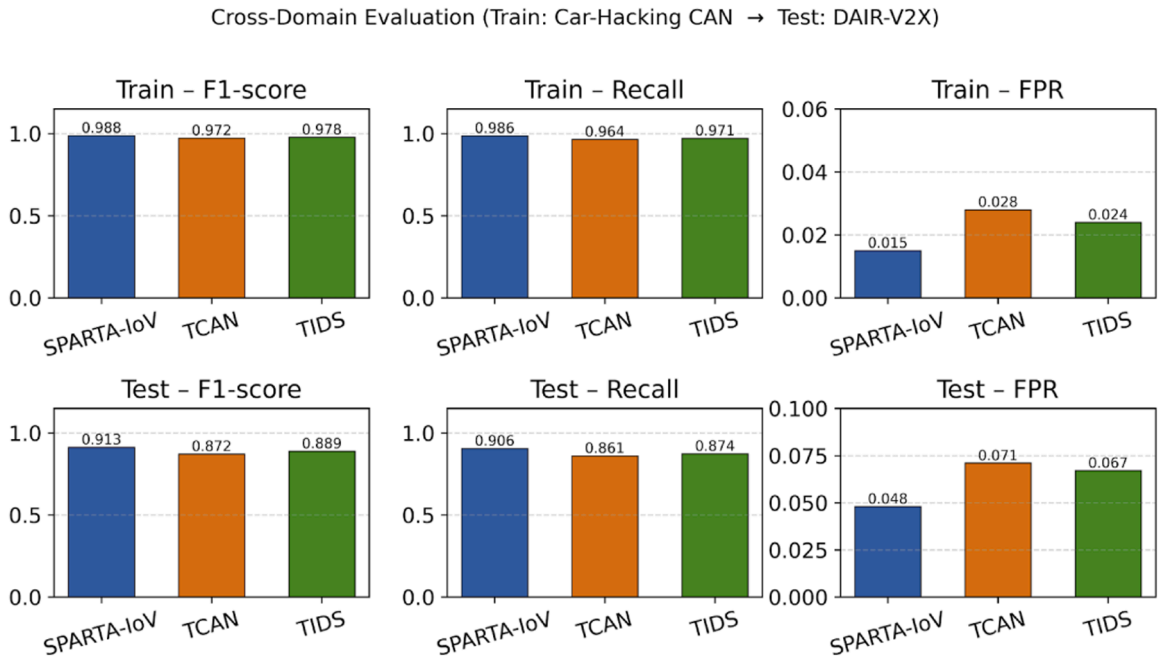

Fig. 22. Cross-domain evaluation of intrusion detection performance when training on the Car-Hacking CAN bus dataset and testing on the real-world DAIR-V2X dataset. The top row reports training performance, while the bottom row shows cross-domain test results.

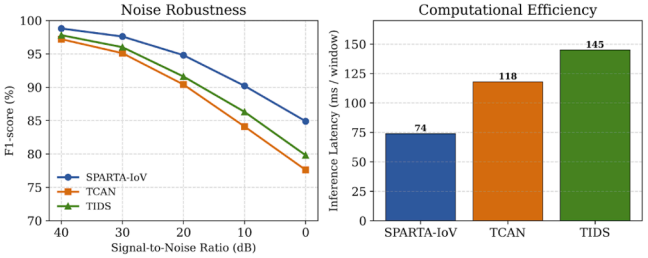

Fig. 23. Noise robustness and computational efficiency comparison of SPARTA, TCAN, and TIDS under realistic real-time IoV deployment conditions. The left panel reports F1-score performance across varying signal-to-noise ratio (SNR) levels, while the right panel compares average inference latency per sliding window.

rially for 50 epochs using two discriminators, D_1 (latent alignment) and D_2 (reconstruction realism). The final validation loss was 0.049. During the evaluation, the threshold for detecting anomalies was set adaptively to $\mu + 3\sigma$ of reconstruction error on W_{Vali}. This made it easy to tell the difference between normal and aberrant patterns. SPARTA got 99.1% accuracy, 98.8% precision, 98.2% recall, and an F1-score of 98.5% on W_{test}. It also had a very low false positive rate of 1.6%. Fine-grained analysis showed that the model consistently identified different types of attacks, with recall scores of 98.7% for spoofing, 97.4% for flooding, and 98.3% for replay attacks. This shows that the adversarial dual-discriminator training was able to pick up on small differences in the structure of temporal anomalies. When tested under Gaussian noise (SNR = 10 dB), the F1-score dropped just slightly from 98.5% to 94.6% and the FPR rose only to 3.2%. This shows that SR preprocessing is useful for keeping anomaly cues intact in less-than-ideal situations. Finally, deployment profiling on embedded hardware showed that real-time operation was possible: SPARTA processed each window in 54 ms at 5.0 W power on the NVIDIA Jetson Nano, with just 1.4% frame drop; on the Raspberry Pi 4, it took 78 ms at 3.2 W with a 4.0% drop rate.

5.7. Complexity analysis

The SPARTA framework is hard to compute because of three main parts: the SR preprocessing, the adversarial VAE training process, and the BPSO-based architectural search. The FFT is used in the SR preprocessing stage to change each univariate time-series sensor stream into the frequency domain. This takes $O(NlogN)$ time for each signal of length N. For M such sensor streams, the total difficulty for SR transformation is $O(MNlogN)$, followed by smoothing and inverse FFT operations of similar complexity, which gives an overall SR preprocessing cost of $O(MNlogN)$. After the saliency maps are made and normalized, the time-series data is split up into segments

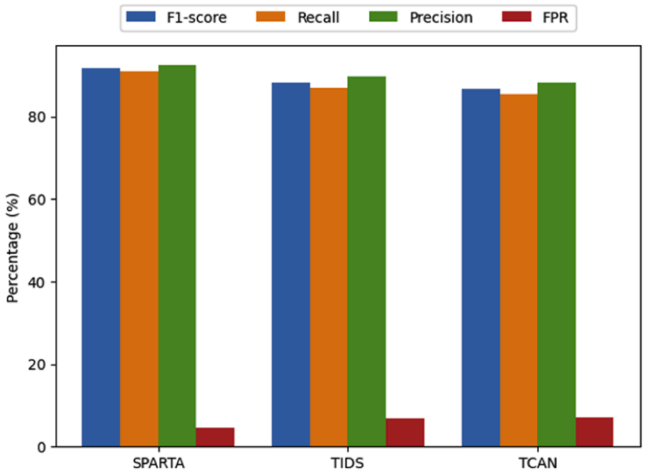

Fig. 24. Performance comparison of SPARTA, TCAN, and TIDS under extreme class imbalance and unknown attack conditions.

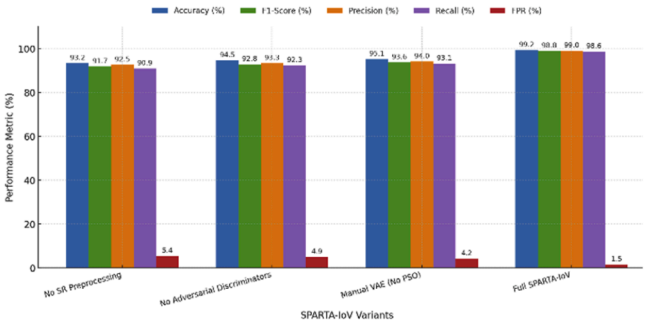

Fig. 25. Ablation study on the contribution of SPARTA components across multiple detection metrics.

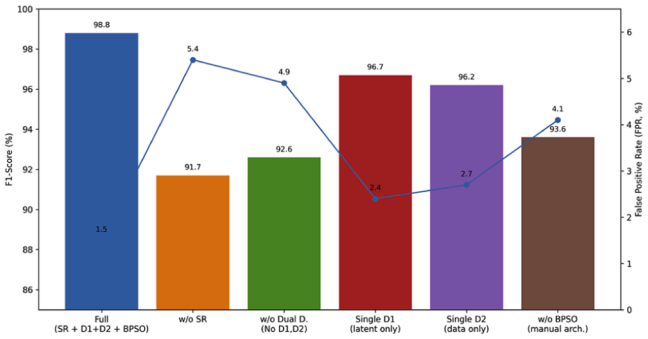

Fig. 26. Ablation study comparing single-discriminator (D_1-only and D_2-only) and dual-discriminator ($D_1 + D_2$) adversarial VAE configurations in SPARTA.

using a sliding window of length *sw* and stride *ss*. This gives us around $T = \frac{N-sw}{ss}$ segments for each stream. The VAE model uses each segment as an input sample. The VAE's training complexity is around $O\left(T \cdot L \cdot sw \cdot d^2\right)$, where d is the average number of filters per layer, and the kernel sizes and feature map widths are the same across all layers. This is because the VAE has L convolutional layers in both the encoder and decoder. In the adversarial arrangement, there are two discriminators (D_1 and D_2), which add an extra overhead of around the same size. This means that the computational cost of each epoch during training is practically doubled. The BPSO-based hyperparameter and architectural search, on the other hand, take up the greatest processing power. The optimization process checks the fitness of each configuration over G generations, with a population of P particles, each represented by a binary string of length b (which includes batch size, optimizer, learning rate, number of layers, kernel sizes, activations, and normalization types). If each

particle is trained for E epochs on a validation split, the overall complexity of the search phase becomes $O\left(G \cdot P \cdot E \cdot T \cdot L \cdot sw \cdot d^2\right)$, which is the number of times the model is instantiated and the number of adversarial training cycles needed for evaluation. Even while this takes a lot of computing power, the one-time optimization step may be done on several GPUs or compute nodes at the same time, and the architecture that comes out of it can be utilized again in other deployments. Also, the model only needs to go through the encoder and decoder once it is trained (and optional discriminators), which cuts down on the amount of time it takes to decide by $O\left(L \cdot sw \cdot d^2\right)$, per input window. This makes it possible to process data in real time in edge or fog-based IoV environments.

5.8. Implications, assumptions, and limitations

From an academic point of view, SPARTA improves the state of the art in unsupervised anomaly detection by combining SR preprocessing, adversarial VAEs, and BPSO for automated NAS in a new way. This combination solves a number of long-standing problems with time-series modeling, such as being sensitive to non-stationarity, not being able to generalize well across dynamic situations, and needing to build models by hand. SPARTA opens up new areas of study in self-adaptive security models, interpretable anomaly detection, and frequency-domain saliency extraction for multivariate data by removing the requirement for labeled attack data and making it possible to automatically optimize hyperparameters and network depth. Also, it gives a flexible methodological framework that may be used in areas other than the IoVs, such as Industrial IoT, smart grids, and healthcare telemetry. This encourages academic research across disciplines. SPARTA directly solves the real-world problems that Original Equipment Manufacturers (OEMs), automotive cybersecurity providers, and smart transportation infrastructure developers encounter in the business sector. Traditional IDS in automotive networks frequently needs datasets that have already been tagged, rules that have been written by hand, and regular manual updates to operate. This makes them not very good for the complicated and quickly changing attack surface of connected automobiles. SPARTA gets around these problems by providing a completely automated, plug-and-play anomaly detection pipeline that learns from live telemetry feeds in real time, without the need for human supervision. It is very useful for in-vehicle edge computing units, roadside anomaly detection modules, and cloud-based fleet monitoring systems since it can handle noise, work with a lot of vehicles, and work with a lot of different kinds of vehicles. Also, as car systems move toward becoming completely self-driving, the necessity for security systems that are strong and flexible, becomes even more important. Cybersecurity-by-design is becoming more important to regulatory agencies and standards groups (such as ISO/SAE 21,434 and UNECE WP.29), and SPARTA's unsupervised, self-configuring architecture fits in nicely with these guidelines. It cuts down on operating costs by lowering maintenance costs, gets rid of the need for unusual labeled data, and speeds up deployment times for automotive OEMs and Tier-1 suppliers.

SPARTA is meant to be a fully unsupervised anomaly detection framework, but it works based on a number of assumptions and has some real-world limitations that need to be understood. First, SPARTA assumes that the training data mostly shows normal driving and communication behavior. If the training set is full of anomalies, the learned latent representation may pick up on them and make it harder to find them. Second, the framework uses reconstruction error (and adversarial regularization) as the main anomaly signal. This means that anomalies that look a lot like normal patterns or are hard to see across many channels may have low reconstruction deviation and be harder to find. Third, preprocessing based on SR improves saliency by focusing on rare frequency components. However, its effectiveness may depend on the length of the window and the spectral properties of the telemetry streams. If the windowing is not done correctly or if there are strong non-periodic anomalies, the benefit of spectral enhancement may be lessened. Fourth, the BPSO-driven architecture search adds extra offline computational cost during the optimization phase. This cost is only incurred once, and the resulting model can be used again, but it may be hard to manage for very large search spaces or strict resource budgets. Lastly, like many detectors that learn, SPARTA's performance may go down when there is a lot of concept drift (like big software updates, sensor replacements, or long-term changes in behavior). This is why lightweight drift monitoring and periodic re-optimization should be part of deployment. These limitations show what needs to be done in the real world and encourage future improvements like training that takes contamination into account, adaptive thresholding, and online updates that take drift into account.

6. Conclusion and future work

We have presented SPARTA, a completely unsupervised, scalable, and noise-tolerant way to find anomalies in the IoV. SPARTA solves some of the biggest problems in vehicle cybersecurity, like needing labeled data, not being able to generalize well in changing situations, and not being able to handle sensor noise well. It does this by combining SR preprocessing, a BPSO-driven neural architecture search, and a dual-discriminator adversarial VAE. The SR preprocessing changes time-domain telemetry into better frequency-domain representations to make anomalies stand out more, and adversarial learning makes sure that reconstruction is accurate and the latent space is aligned. BPSO automates the search for the best architectures and hyperparameters; thus, SPARTA can easily adapt to different types of vehicles without any help from a person. SPARTA showed better identification accuracy (99.2%), a lower FPR (1.5%), and resistance to adversarial noise, domain shift, and concept drift when tested on five real-world and simulated datasets, such as CAN-bus, V2X, GPS, and LiDAR streams. The system worked well on edge devices and handled larger inputs and higher sampling resolutions without any problems. Ablation investigations showed that spectral pretreatment, adversarial regularization, and automated architectural search all worked together to improve the performance of SPARTA.

SPARTA works well, but there are a few things that might make it even more durable, adaptable, and easy to use in the real world. First, adding adversarial training methods (like FGSM and PGD) or certified defenses might make it harder to break things that are

intended to be broken. Second, allowing self-adaptive drift control via online learning or lightweight ensemble models may keep detection accuracy high for long-term deployments. Third, employing interpretable visuals and attribution methodologies (such as SHAP and saliency maps) to improve explainable anomalous reasoning would help make decisions that are important for safety and forensics. Also, adding sensor-aware attention techniques might help deal with modality-specific errors like GPS spoofing or CAN injection better. A federated learning extension would let cars work together without giving up their raw data, which might be made even more secure with encryption or differential privacy. Planned real-world field experiments will check latency, hardware feasibility, and deployment readiness under real-world traffic scenarios. Also, looking into domain adaptation methods might make it easier to generalize across different locations and platforms. Lastly, adding more agents to SPARTA to make it a multi-agent cooperative anomaly detection framework that uses V2V/V2X communication might improve detection accuracy by getting all the vehicles to agree. SPARTA is a great starting point for future work on safe, smart, and self-driving transportation networks.

Ethics approval and consent to participate

Not applicable.

Funding

This work was funded in part by a grant from Fondazione Caritro, under project GenIE.

Author contributions

All authors contributed substantially to the conception, design, analysis, and interpretation of the research.

Declaration of competing interest

The authors declare no conflict of interest.

References

[1] Sakhnevych A, Pasquino N, Sperlì G. Design of a machine learning approach to anomaly detection in Tyre-Road interaction. IEEE Access 2025;13:28920–34.
[2] Onsu MA, Simsek M, Fobert M, Kantarci B. Intelligent multi-sensor fusion and anomaly detection in vehicles via deep learning. Internet Things 2025;31:101561.
[3] Xu B, Zhao J, Wang B, He G. Detection of zero-day attacks via sample augmentation for the internet of vehicles. Veh Commun 2025;52:100887.
[4] Wu D, Peng J, Yu S, Ge Y, Ma C, Zhou J. UKD-TEAD: an unsupervised knowledge distillation framework for detecting anomalies in traffic equipment with various aspect ratios. IEEE Internet Things J 2025.
[5] Li S, Cao Y, Zhang YA, Liao T, Yan F, Lin H. A cloud collaborative-based intrusion detection and prevention system for IVN. IEEE Trans Cogn Commun Netw 2024.
[6] Liu H, et al. Blockchain and federated learning for collaborative intrusion detection in vehicular edge computing. IEEE Trans Veh Technol 2021;70(6):6073–84.
[7] Ullah I, Khalil I, Bai X, Garg S, Kaddoum G, Shamim M. An ensemble-based hybrid model for the detection of attacks in the internet of Vehicular things. IEEE Trans Intell Transp Syst 2025;26:17914–27.
[8] Ahmed I, Jeon G, Ahmad A. Deep learning-based intrusion detection system for internet of vehicles. IEEE Consum Electron Mag 2021;12(1):117–23.
[9] Xie N, Zhang C, Yuan Q, Kong J, Di X. IoV-BCFL: an intrusion detection method for IoV based on blockchain and federated learning. Ad Hoc Netw 2024;163:103590.
[10] Nie L, Ning Z, Wang X, Hu X, Cheng J, Li Y. Data-driven intrusion detection for intelligent internet of vehicles: a deep convolutional neural network-based method. IEEE Trans Netw Sci Eng 2020;7(4):2219–30.
[11] Almehdhar M, et al. Deep learning in the fast lane: a survey on advanced intrusion detection systems for intelligent vehicle networks. IEEE Open J Veh Technol 2024;5:869–906.
[12] Khan IA, Moustafa N, Pi D, Haider W, Li B, Jolfaei A. An enhanced multi-stage deep learning framework for detecting malicious activities from autonomous vehicles. IEEE Trans Intell Transp Syst 2021;23(12):25469–78.
[13] Li X, Hu Z, Xu M, Wang Y, Ma J. Transfer learning based intrusion detection scheme for internet of vehicles. Inf Sci 2021;547:119–35.
[14] Wu M, et al. Split learning with differential privacy for integrated terrestrial and non-terrestrial networks. IEEE Wirel Commun 2023;31(3):177–84.
[15] Li C, et al. An enhanced CLKAN-RF framework for robust anomaly detection in unmanned aerial vehicle sensor data. Knowl Based Syst 2025;319:113690.
[16] Shen G, et al. Anomalous trajectory detection in vehicular social networks with unsupervised dynamic network representation learning. IEEE Trans Veh Technol 2025;74:14577–90.
[17] Mishra S, Sengar N, Har D. A secure, blockchain-enabled vehicular sensor communication protocol with deep learning-assisted anomaly detection. IEEE Intell Transp Syst Mag 2025;17:88–95.
[18] Politi E, Davalas C, Chronis C, Dimitrakopoulos G, Michail D, Varlamis I. Real-time quality monitoring and anomaly detection for vision sensors in connected and autonomous vehicles. IEEE Access 2025;13:25557–67.
[19] Wang Y, Qin G, Liang Y. A reliability anomaly detection method based on enhanced GRU-autoencoder for Vehicular fog computing services. Comput Secur 2025;150:104217.
[20] Singh A, Rathore H. Advancing connected vehicle security through real-time sensor anomaly detection and recovery. Veh Commun 2025;52:100876.
[21] Zhou M, Han L, Wang J. Physical invariant subspace based unsupervised anomaly detection for internet of vehicles. IEEE Trans Intell Veh 2024.
[22] Li X, Xie C, Zhao Z, Wang C, Yu H. Anomaly detection algorithm of industrial Internet of Things data platform based on deep learning. IEEE Trans Green Commun Netw 2024.
[23] Bergies S, Aljohani TM, Su S-F, Elsisi M. An IoT-based deep-learning architecture to secure automated electric vehicles against cyberattacks and data loss. IEEE Trans Syst Man Cybern: Syst 2024;54:5717–32.
[24] Cao J, et al. Anomaly detection for in-vehicle network using self-supervised learning with vehicle-cloud collaboration update. IEEE Trans Intell Transp Syst 2024;25:7454–66.
[25] Zhao L, et al. Generative abnormal data detection for enhancing cellular vehicle-to-everything based road safety. IEEE Trans Green Commun Netw 2024;8:1466–78.
[26] Devika S, Shrivastava RR, Narang P, Alladi T, Yu FR. Vadgan: an unsupervised gan framework for enhanced anomaly detection in connected and autonomous vehicles. IEEE Trans Veh Technol 2024.

[27] Abdulganiyu OH, Tchakoucht TA, Saheed YK, Ahmed HA. XIDINTFL-VAE: XGBoost-based intrusion detection of imbalance network traffic via class-wise focal loss variational autoencoder. J Supercomput 2025;81(1):16.
[28] Tian S, Zheng Y, Xi L, Lin S, Li Y, Zhu H. Network intrusion detection system based on enhanced dual gaussian mixture variational autoencoders for internet of vehicles. IEEE Trans Intell Transp Syst 2026:1–14.
[29] Xu L, Yang Z, Zhao D, Yu F, Zhou Y, Zhang H. G-VAE: variational autoencoder-based adversarial attacks and defenses in industrial control systems. Comput Electr Eng 2025;124:110290.
[30] Hou X, Zhang L. Saliency detection: a spectral residual approach. In: 2007 IEEE Conference on computer vision and pattern recognition. Ieee; 2007. p. 1–8.
[31] Kingma DP, Welling M. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, ; 2013.
[32] El-Maleh AH, Sheikh AT, Sait SM. Binary particle swarm optimization (BPSO) based state assignment for area minimization of sequential circuits. Appl Soft Comput 2013;13(12):4832–40.
[33] Liao Y, Wang Y, Cui Q, Chen K-C, Nan G, Tao X. Data-driven cyber-physical anomaly detection with GAN in federated smart factories. IEEE Trans Ind Inform 2025.
[34] Hussain A, Yadav A, Ravikumar G. Anomaly detection using bi-directional long short-term memory networks for cyber-physical electric vehicle charging stations. IEEE Trans Ind Cyber-Phys Syst 2024;2:508–18.
[35] Jo H, Kim D-H. Intrusion detection using transformer in con-troller area network. IEEE Access 2024;12:121932–46.
[36] Zhang T, et al. Hybrid transfer and self-supervised learning approaches in neural networks for intelligent vehicle intrusion detection and analysis. IEEE Internet Things J 2024.