Journal of the Brazilian Computer Society, 2026, 32:1, doi: 10.5753/jbcs.2026.6210

Natural Language-Based Explainable Intrusion Detection for Vehicular Networks: Handling CAN and Wi-Fi Attack Vectors

Rodnney C. Machado [**Pampa Federal University** | ***rodnneymachado.aluno@unipampa.edu.br***]
Felipe H. Scherer [**Pampa Federal University** | ***felipescherer.aluno@unipampa.edu.br***]
Felipe N. Dresch [**Pampa Federal University** | ***felipedresch.aluno@unipampa.edu.br***]
Silvio E. Quincozes [**Pampa Federal University** | ***silvioquincozes@unipampa.edu.br***]
Diego Kreutz [**Pampa Federal University** | ***diegokreutz@unipampa.edu.br***]
Vagner E. Quincozes [**Fluminense Federal University** | ***vequincozes@id.uff.br***]

AI Horizon Labs, Federal University of Pampa. Ave. Tiaraju 810, Ibirapuitã, Alegrete, RS, 97546-550, Brazil.

Received: 06 July 2025 • **Accepted:** 13 March 2026 • **Published:** 11 June 2026

Abstract. Modern vehicles increasingly rely on interconnected systems, combining internal networks (e.g., CAN) with external interfaces (e.g., embedded Wi-Fi). While this connectivity enables advanced functionalities, it also expands the potential attack surface for cyber threats. Existing intrusion detection solutions often address these layers in isolation, but we identify a need for integrated, explainable, and user-centered approaches. In this work, we propose a conceptual architecture for explainable intrusion detection in connected vehicles. Our solution simultaneously analyzes CAN and Wi-Fi traffic, using supervised learning models (XGBoost) for anomaly detection. To ensure interpretability, we apply SHAP to quantify feature importance and leverage Large Language Models (LLMs) to generate clear, textual explanations from the results. For validation, we conduct experiments using the X-CANIDS and AWID2 datasets, simulating common attacks such as fuzzing, fabrication, Evil Twin, and Hirte. Our results demonstrate that combining XAI and LLMs produces accurate, auditable narratives about attacker behavior, improving transparency in automotive security systems.

Keywords: Explainable Artificial Intelligence (XAI), CAN bus, Wi-Fi attacks, Intrusion Detection System (IDS), SHAP, Large Language Models (LLMs), Connected Vehicles, Cybersecurity, Automotive Networks, AI Interpretability

1 Introduction

Modern vehicles have evolved into highly connected computational platforms due to the increasing digitalization of the automotive industry. While traditional in-vehicle communication relies on the Controller Area Network (CAN), contemporary cars also incorporate external connectivity technologies such as built-in Wi-Fi. These advancements enable functionalities like Over-the-Air (OTA) updates, infotainment, remote diagnostics, and mobile device pairing, but they also introduce new attack vectors that expand the threat surface [Wang *et al.*, 2024; Sharmin *et al.*, 2024; Liu *et al.*, 2025; Sreelekshmi and Aji, 2025].

The CAN network has long been a focus of security research due to its broadcast communication model, lack of encryption, and weak authentication mechanisms between Electronic Control Units (ECUs) [Van Herrewege *et al.*, 2018; Liu *et al.*, 2025]. ECUs are specialized embedded computing nodes that constitute a complex distributed system within the vehicle. In modern automotive architectures, these units effectively transform the vehicle into a 'datacenter on wheels', where each node manages specific mission-critical functions—such as braking, engine control, and steering—while relying on constant, low-latency communication to ensure operational safety. Attacks such as fuzzing, spoofing, and message fabrication have shown that CAN traffic can be manipulated to disrupt critical vehicle functions [Woo *et al.*, 2014; De Vincenzi *et al.*, 2024]. With the addition of Wi-Fi connectivity, risks are no longer confined to the internal network [Wang *et al.*, 2024]. This expanded connectivity allows attackers to target vehicles remotely, often without physical access [Kabilan *et al.*, 2024].

Vehicular Wi-Fi networks are vulnerable to attacks like Evil Twin and Hirte, where attackers impersonate access points or exploit authentication protocols to intercept or inject malicious traffic [Aminanto and Kim, 2017; Kolias *et al.*, 2016; Villain *et al.*, 2024]. These attacks take advantage of weaknesses in the IEEE 802.11 protocol and embedded device configurations, potentially serving as gateways for deeper intrusions into the vehicle's systems. The combination of CAN and Wi-Fi layers creates a complex, multi-layered risk landscape [Kabilan *et al.*, 2024; Wang *et al.*, 2024; Villain *et al.*, 2024; Liu *et al.*, 2025].

Machine Learning (ML) based Intrusion Detection Systems (IDSs) have become essential for identifying and mitigating attacks in vehicular networks. However, many of these models operate as black boxes, providing little transparency into their decision-making processes. This lack of interpretability limits their practical usability, especially in automotive security, where decisions must be clear and auditable [Zhang *et al.*, 2022]. Explainable Artificial Intelligence (XAI) techniques, such as SHapley Additive exPlanations (SHAP) [Lundberg and Lee, 2019], aim to address this challenge by providing human-understandable explanations.

Despite progress in XAI for intrusion detection, two major challenges remain: (i) the need for expert knowledge to make

SHAP outputs more interpretable [Le *et al*., 2023; Jeong *et al*., 2024; Khani *et al*., 2024; Gupta and Seeja, 2024], and (ii) limited research on intrusion detection for both CAN and Wi-Fi networks. Recent efforts have explored using Large Language Models (LLMs) to translate SHAP explanations into natural language narratives [Liu *et al*., 2023; Alawida *et al*., 2023], but these hybrid approaches have yet to be thoroughly tested in automotive contexts. To our knowledge, no existing solution is specifically tailored for CAN and Wi-Fi networks. As noted by [Zytek *et al*., 2024], many current XAI explanations including those based on SHAP are often difficult for domain experts without ML expertise to understand. While initial solutions like XAIstories [Martens *et al*., 2025] have used LLMs to improve explanation accessibility, they remain largely unexplored in vehicular settings.

We address this gap by proposing a novel architecture[1] that integrates intrusion detection systems, XAI techniques, and LLM-based interpretation for the explainable analysis of cyberattacks in connected vehicles. Our approach leverages XAI to identify key intrusion-related features and employs large language models to generate semantically meaningful explanations that support human understanding and analysis.

The remainder of this paper is organized as follows. In Section 2, we examine security threats in vehicular networks, with a focus on vulnerabilities in CAN and Wi-Fi communications. Section 3 introduces the fundamentals of explainable artificial intelligence and discusses the role of LLMs in semantic interpretation. Section 4 reviews related work on intrusion detection and interpretability, highlighting gaps in the existing literature. Section 5 describes the proposed architecture, including its data flow and the integration of XAI and LLM components. Section 6 presents the experimental evaluation, covering classification results, generated explanations, and system performance. Section 7 discusses threats to validity and limitations of the approach, and finally, Section 8 concludes the paper and outlines directions for future research.

2 Security Threats

Modern vehicles rely heavily on communication between internal and external subsystems, with protocols like CAN and wireless interfaces (e.g., Wi-Fi) playing a central role. While this integration enables advanced functionalities, it also expands the attack surface, exposing vehicle systems to various threats. In this section, we examine the primary attack vectors associated with these two critical components.

2.1 Threats in the CAN Network

The CAN protocol serves as the primary in-vehicle communication standard for message exchange between ECUs, including systems for braking, engine control, steering, and airbags. Originally developed in the 1980s to meet automotive industry demands, CAN employs a message-oriented architecture with low latency and high resilience to electrical faults [GmbH, 1991]. However, its design assumes a closed and trusted environment, resulting in the absence of fundamental security mechanisms such as authentication, encryption, and integrity checks [Wang *et al*., 2024].

Attackers exploit these vulnerabilities through various techniques. Fuzzing attacks inject random messages to trigger unexpected system behavior [Marchetti and Stabili, 2017], while message fabrication allows adversaries to spoof legitimate signals from critical ECUs, such as falsifying speed or torque data without detection [Woo *et al*., 2014]. The broadcast nature of the CAN protocol further exacerbates these risks, as any connected ECU can listen to or transmit messages, facilitating sniffing and spoofing attacks in vehicular environments. [Jeong *et al*., 2024].

The lack of authentication enables man-in-the-middle attacks, particularly when physical access is obtained through the OBD-II port [Kabilan *et al*., 2024; Wang *et al*., 2024]. With the increasing use of diagnostic tools and Bluetooth adapters connected to the CAN network, the security risks grow, especially when integrated with externally connected vehicle systems.

2.2 Threats in the Wi-Fi Interface

Wi-Fi connectivity has become standard in modern vehicles, supporting features like in-car hotspots, smartphone pairing, multimedia streaming, OTA updates, and remote management systems. While essential for vehicle modernization, this external communication layer introduces a critical attack surface due to its exposure to remote threats.

Wi-Fi-based attacks are more accessible than those targeting the CAN bus, as they can be executed remotely using low-cost equipment and often without physical access to the vehicle [Kabilan *et al*., 2024]. The most relevant attack types can be summarized as follows:

- **Evil Twin**: A rogue access point mimics a legitimate one to capture traffic from connected devices [Kolias *et al*., 2016]. In vehicular networks, this could target Wi-Fi-enabled ECUs.
- **Hirte**: This attack combines frame fragmentation and packet injection to exploit vulnerabilities in the 802.11 protocol [Aminanto and Kim, 2017].

In automotive environments, Wi-Fi exploitation can serve as an entry point for pivoting into the CAN network. For example, an ECU handling telemetry or infotainment may act as a bridge between Wi-Fi and the CAN bus. Without proper network segregation, cross-layer attacks can compromise critical vehicle functions through initial external exploitation [Kabilan *et al*., 2024; Wang *et al*., 2024].

3 Explainable Artificial Intelligence

Ensuring that stakeholders can understand and verify model decisions is a critical requirement for vehicular security systems, where transparency directly impacts operational trust. Explainable Artificial Intelligence (XAI) addresses this challenge by making machine learning models interpretable, shifting from 'black-box' predictions to auditable results. Among

[1] This work is an extension of "Modelos Interpretáveis com Inteligência Artificial Explicável (XAI) na Detecção de Intrusões em Redes Intra-Veiculares (CAN)" (available at: `https://sol.sbc.org.br/index.php/sbseg/article/view/30042`).

XAI methods, SHAP has emerged as particularly valuable. Based on game theory concepts, SHAP quantifies how each feature contributes to a model's predictions for each instance[Lundberg and Lee, 2017].

In automotive security applications, SHAP has proven useful for both explaining intrusion detection system outputs and supporting attack forensics. For instance, research shows SHAP can identify which vehicle signals - such as `Clu.Odometer` or `SAS.Angle` - most strongly influence a model's classification of fabrication or fuzzing attacks targeting these components [Marchetti and Stabili, 2017].

For Wi-Fi networks, SHAP analysis helps determine the importance of network attributes like `frame.len`, `wlan.duration`, and `wlan.fc.subtype` in detecting malicious traffic. Studies using the AWID2 dataset [Kolias *et al*., 2016] demonstrate these features effectively distinguish between legitimate and malicious network activity [Kolias *et al*., 2016; Aminanto and Kim, 2017].

While SHAP provides valuable insights, interpreting its outputs remains challenging for professionals without ML expertise. This is where LLMs show promise as complementary tools. LLMs can transform technical SHAP visualizations into natural language explanations. For example, an LLM might generate:

> *The increased `wlan.duration` value in this instance suggests abnormal channel occupancy, a pattern commonly associated with Evil Twin attacks.* $GeneratedbyGPT-4$

Recent research highlights LLMs' potential to enhance XAI interpretation, offering contextualized insights that support security system audits [Alawida *et al*., 2023; Martens *et al*., 2025]. In automotive applications, this combined approach could make explainability more accessible across various contexts - from manufacturer interfaces to diagnostic centers and incident response systems.

4 Related Work

Research in automotive network security has evolved to address two key aspects: protecting internal vehicle communications (particularly CAN networks) and securing external connectivity (such as embedded Wi-Fi). While ML approaches have demonstrated strong potential for intrusion detection in both domains, their growing complexity has created new challenges in model interpretability that the research community is actively addressing.

The literature reveals three main research directions in this field. First, numerous studies have focused on detecting CAN bus attacks, with approaches ranging from traditional signature-based methods to advanced deep learning techniques [Marchetti and Stabili, 2017; Woo *et al*., 2014]. Second, researchers have investigated Wi-Fi specific threats in vehicular contexts, developing detection methods for attacks like Evil Twin and Hirte [Kolias *et al*., 2016; Aminanto and Kim, 2017]. Third, and most recently, the field has seen growing interest in making these detection systems more transparent through Explainable AI (XAI) techniques [Lundberg and Lee, 2017; Zhang *et al*., 2022].

4.1 Explainable IDSs for CAN

Research has demonstrated numerous successful implementations of IDSs for CAN networks [Marchetti and Stabili, 2017; Kang *et al*., 2016; Woo *et al*., 2014]. While these models show effectiveness in detection, most lack formal explainability mechanisms in their decision-making processes. Recent work has begun incorporating techniques like SHAP to interpret the importance of CAN message features (e.g., IDs, payload content, periodicity) in anomaly prediction [Gupta and Seeja, 2024; Khani *et al*., 2024]. However, these explanations remain largely technical and inaccessible to non-expert users in real-world deployments.

Jeong *et al*. [2024] made significant progress with their X-CANIDS system, which uses an autoencoder neural network evaluated across six architectures (e.g., FC, CNN, LSTM). Their contribution includes the publicly available X-CANIDS dataset collected from a 2017 Hyundai Sonata, featuring deserialized CAN payloads converted to human-readable signals. While this improves data interpretability, the work does not incorporate XAI techniques to explain model decisions in a transparent manner.

The explainability challenge is further examined by Lundberg *et al*. [2022], who focus specifically on interpreting DNN-based CAN intrusion detection. Their approach faces limitations due to the Survival Dataset [Han *et al*., 2018], where many features lack clear semantic meaning. This restricts the practical utility of their XAI visualizations for real-world automotive security applications.

Several studies encounter similar limitations with existing datasets. Le *et al*. [2023] demonstrate this issue when applying SHAP explanations to XGBoost models trained on the Car-Hacking Dataset [Seo *et al*., 2018]. Like the Survival dataset, its features often lack clear real-world interpretations, limiting the effectiveness of XAI tools. This problem persists in works by Metwaly and Elhenawy [2023], Ding *et al*. [2024], and Wickramasinghe *et al*. [2023], all constrained by dataset interpretability issues.

Wickramasinghe *et al*. [2023] propose an adversarial ML approach that requires generating new samples for explanation. While innovative, this method is model-specific and not easily transferable to other IDS implementations.

The challenges extend to specialized CAN implementations as well. Hong and Yoo [2024] study UAVCAN networks in drones, using SHAP to explain their heterogeneous ensemble model's detection of various attack types. However, dataset imbalance and feature correlation issues potentially compromise both model robustness and explanation reliability.

4.2 XAI-Based IDS for Wi-Fi Networks

Recent research has explored the application of XAI methods to improve intrusion detection in Wi-Fi networks, primarily focusing on model interpretability and feature selection. Table 2 summarizes key contributions in this domain. These studies demonstrate how XAI techniques can enhance understanding of model decisions while revealing persistent challenges in achieving truly accessible explanations.

Gimenes [2024] developed an XAI-based feature selection method for detecting impersonation attacks using the AWID2

Table 1. Related works on intrusion detection in CAN networks with XAI or interpretable signs.

References	Technique/Model	Database	Main Limitations
[Kang *et al*., 2016], [Woo *et al*., 2014]	Random Forest, XGBoost	Car-Hacking	Lacks an interpretable analysis of the decision process.
[Jeong *et al*., 2024]	Autoencoder Neural Networks	X-CANIDS	Absence of XAI.
[Lundberg *et al*., 2022]	DNN with a proprietary explanation method	Survival	Ignores performance metrics.
[Wickramasinghe *et al*., 2023]	Adversarial learning with embedding XAI	Survival	Explanation lacks generalization.
[Le *et al*., 2023]	XGBoost with SHAP	Car-Hacking	Limited dataset (12 attributes, most with unknown meaning).
[Hong and Yoo, 2024]	Heterogeneous ensemble with SHAP	UAVCAN	Data imbalance and risk of overfitting.
[Metwaly and Elhenawy, 2023], [Ding *et al*., 2024], [Wickramasinghe *et al*., 2023]	Multiple ML-based approaches	Survival	Lacks semantic interpretability.

Table 2. Feature selection and explainability in IDSs: Wi-Fi and other XAI-based IDSs.

References	Method	Dataset	Main Contributions
[Gimenes, 2024]	SHAP	AWID2	XAI-based feature selection for Impersonation in Wi-Fi.
[Khani *et al*., 2024]	SHAP	AWID	XAI-based IDS in Wi-Fi attacks.
[Gupta and Seeja, 2024]	SHAP	AWID	XAI-based IDS in Wi-Fi attacks.
[Reyes *et al*., 2020]	SHAP	AWID	WNIDS in Wi-FI attacks.

dataset. Their SHAP-based approach successfully identified critical features for attack classification while providing visual explanations of feature contributions. However, these explanations remain technical in nature, requiring network security and ML expertise for proper interpretation. The study does not bridge the gap to semantic understanding of 802.11 protocol behaviors or attack patterns.

Similar limitations appear in the work of Khani *et al*. [2024], who implemented a CNN-based IDS with SHAP interpretability. While their system effectively revealed feature importance across different attack scenarios using the AWID dataset, the insights are presented through technical visualizations without deeper protocol-level explanations. This restricts the practical utility for security professionals lacking ML expertise during operational analysis.

Gupta and Seeja [2024] demonstrated how SHAP could enhance traditional IDS models for Wi-Fi networks, using the AWID dataset to provide a clearer understanding of model behavior. Their work successfully increased trust in AI-based detection but similarly stopped short of translating technical explanations into more accessible security insights.

A more structured approach comes from Reyes *et al*. [2020], whose WNIDS system combines feature reduction (from 154 to 7-19 features) with SHAP-based explainability. While achieving 99.42% accuracy in detecting flooding, impersonation, and injection attacks on the AWID-CLS-R dataset, their explanations still require specialized knowledge to interpret the SHAP outputs effectively.

These studies collectively reveal three key limitations:

- Over-reliance on SHAP as primarily a visualization tool rather than an explanation generator.
- Lack of semantic interpretation connecting features to protocol behaviors and attack patterns.
- Persistent need for technical expertise to use them.

The current state of research shows promising technical achievements in feature selection and model transparency, but leaves significant room for improvement in making explanations truly accessible to non-expert security practitioners. Future work could benefit from combining SHAP analysis with protocol-aware interpretation layers to bridge this gap in operational settings.

4.3 Integration of XAI and LLMs for IDS

The combination of XAI and LLMs represents an emerging research direction in IDS. While initial studies demonstrate the potential of LLMs to enhance the interpretability of XAI outputs, most existing approaches focus on generic network security domains rather than specialized vehicular networks.

Recent work by Khediri *et al*. [2024] illustrates this trend, using Mistral 7B to interpret SHAP explanations from a Ran-

Table 3. Works related to XAI and Generative Models (LLMs)

Study	**Approach**	**Main Contribution**
[Khediri *et al*., 2024]	SHAP + LLM	SHAP explanations translated into LLM using Mistral 7B
[Ali and Kostakos, 2023]	HuntGPT: ML + SHAP/LIME + GPT-3.5	Interactive IDS dashboard with XAI and LLM integration
[Lim *et al*., 2025]	SHAP/LIME + DeepSeek V3	Technical outputs translated into accessible explanations
[Baral *et al*., 2024]	ML/DL + SHAP/LIME + LLMs	User-level adaptive explanations
[Mavrepis *et al*., 2024]	x-[plAIn]: LLM for XAI	Audience-adaptive explanation generation
[Zytek *et al*., 2024]	LLMs + SHAP	Human-readable narratives using GPT-3.5/4
[Zeng, 2024]	LLM + SHAP	SHAP translated into plain language using Mistral 7B
[Martens *et al*., 2025]	XAIstories with GPT-4	SHAP and CF converted into narrative explanations
[Hsu *et al*., 2024]	ChatGPT + SHAP Global plots	Key data extracted and explained in text
[Alnahdi and Narain, 2024]	LLM + LIME + Domain Rules	Evaluation of XAI alignment with expert knowledge
[Bilal *et al*., 2025]	LLMs and XAI	Review of LLM-based interpretability strategies
[Mumuni and Mumuni, 2025]	XAI, LLMs, and VLMs	Tools and examples for explaining explanations

dom Forest model trained on the CICIDS2017 dataset. Although their approach successfully translates technical feature importance into natural language, it doesn't address the unique requirements of automotive networks like CAN or vehicle Wi-Fi. Similar limitations appear in Ali and Kostakos [2023]'s HuntGPT system, which combines SHAP/LIME with GPT-3.5 Turbo for enterprise network security using the KDD99 dataset as a benchmark.

More specialized applications are beginning to emerge. Lim *et al*. [2025] developed a phishing detection framework that pairs logistic regression with dual XAI methods (LIME and SHAP), using DeepSeek V3 to generate protocol-aware explanations. Their approach achieves 98.4% accuracy while providing clear natural language interpretations of phishing tactics. Similarly, Baral *et al*. [2024] created an IoT security framework that adapts Gemini and OpenAI explanations to different administrator expertise levels.

Several studies specifically address the challenge of making XAI outputs accessible. Mavrepis *et al*. [2024] developed ”x-[plAIn]”, a custom LLM that tailors XAI summaries to audience knowledge levels. Hsu *et al*. [2024] focused on interpreting SHAP global plots through LLM generated narratives, while Alnahdi and Narain [2024] automated XAI evaluation against domain rules using GPT-4.

The most relevant conceptual advances come from Zytek *et al*. [2024] and Zeng [2024], who systematically compared LLM performance in explaining SHAP outputs. Their work demonstrates GPT-4's superior clarity over traditional visualizations while identifying key limitations like prompt sensitivity. Martens *et al*. [2025]'s ”XAIstories” further validates this approach through user studies showing improved understanding with GPT-4-generated narratives.

Comprehensive surveys by Bilal *et al*. [2025] and Mumuni and Mumuni [2025] categorize these approaches, highlighting the growing ”explaining the explanations” paradigm. However, as shown in Table 3, none specifically address the automotive domain's unique requirements for CAN and vehicular Wi-Fi security.

4.4 Discussion

Our literature review reveals several important limitations in current research, as summarized in Table 4. While existing studies have made progress in individual areas, our work represents the first comprehensive integration of CAN, Wi-Fi, XAI, and LLM technologies for vehicular security.

Three key gaps emerge from our analysis. First, most studies examine either Wi-Fi or CAN networks in isolation, neglecting the critical security implications of their interconnection. Modern vehicles increasingly rely on both external interfaces and internal networks operating together, yet research has not adequately addressed this integrated attack surface. Second, while SHAP has advanced model interpretability, its technical outputs remain challenging for non-experts. The visualization-heavy approach common in current work creates accessibility barriers for security professionals without data science training. Third, the potential of LLMs to bridge this interpretability gap through semantic explanations remains largely unexplored in the automotive context in existing literature.

These limitations are particularly concerning given the growing complexity of connected vehicles. Urban environments present dynamic threat scenarios where multiple network interfaces interact, yet current detection systems lack both the holistic perspective and explainability needed for effective security operations. Our proposed framework addresses these gaps by combining multi-layer intrusion detection with LLM enhanced explainability tailored to automotive networks in pratice.

The most promising direction emerging from recent literature is the ”explaining the explanations” paradigm demonstrated in generic domains. However, as shown in Table 4, no existing solution adapts this approach specifically for the unique requirements of vehicular networks. Our work fills this critical gap while maintaining the technical rigor needed for automotive cybersecurity applications.

Table 4. Selected studies on XAI and LLMs for CAN and Wi-Fi

References	CAN	Wi-Fi	XAI	LLM
[Marchetti and Stabili, 2017]	⋆			
[Woo *et al.*, 2014]	⋆			
[Kolias *et al.*, 2016]		⋆		
[Aminanto and Kim, 2017]		⋆		
[Khani *et al.*, 2024]		⋆	⋆	
[Gupta and Seeja, 2024]		⋆	⋆	
[Alawida *et al.*, 2023]			⋆	⋆
[Liu *et al.*, 2023]			⋆	⋆
This Work	⋆	⋆	⋆	⋆

5 Proposed Architecture

This section presents a modular and interpretable architecture for intrusion detection in connected vehicles. The system operates across both internal communication channels, represented by the CAN bus, and external communication channels, represented by the Wi-Fi network. It is organized into four main stages: (i) data processing and classification, (ii) explainability and visualization, (iii) semantic interpretation, and (iv) deployment integration.

Due to the heterogeneity of vehicular data sources and protocols, the classification stage relies on domain-specific models tailored to each communication layer. In contrast, the subsequent stages responsible for explainability and semantic interpretation are designed as a unified pipeline, combining XAI techniques and generative models to produce consistent and integrated explanations across domains. This design enables a holistic explanation workflow that is independent of the underlying classifier while preserving interpretability.

Together, these components form a cohesive pipeline that transforms raw vehicular communication data into explainable intrusion alerts, as illustrated in Figure 1. Each stage of the architecture is described in detail in the following subsections. Additional implementation aspects, including programming languages, libraries, interface design, and other technical details, are discussed in Appendix A.

5.1 Data Processing and Classification

Our architecture processes data from two distinct sources: the CAN bus and the Wi-Fi interface. Each domain produces raw traffic that first undergoes preprocessing, where protocol-specific features are extracted and organized into structured datasets (the *CAN Dataset* and the *Wi-Fi Dataset*).

For the *CAN Dataset*, we collect data via devices connected to the vehicle's OBD-II interface or through simulated CAN messages, as commonly found in datasets such as X-CANIDS. The captured information includes identifiers (IDs), payloads, and timestamps, representing signals from various ECUs such as brakes, steering, engine, and airbags. For the *Wi-Fi Dataset*, we consider IEEE 802.11 packet data from real-world datasets AWID2, which contains message fields such as frame size (`frame.len`), channel occupation time (`wlan.duration`), and packet subtype (`wlan.fc.subtype`) in the dataset.

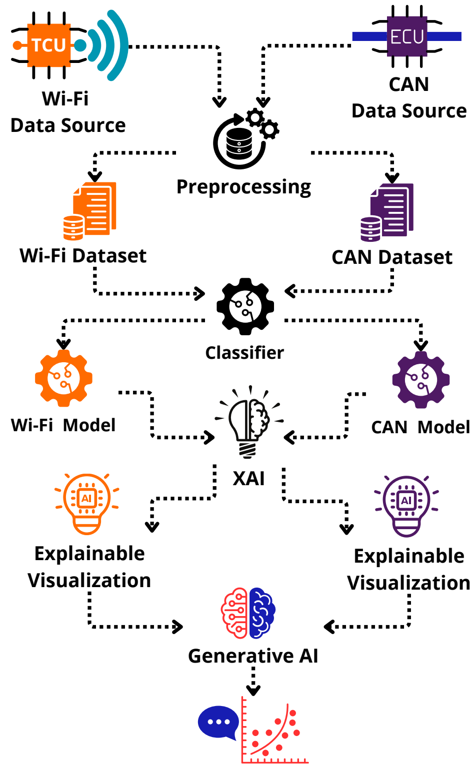

Figure 1. Overview of the IDS architecture: components and data flow.

Both data sources are first passed to a preprocessing module that prepares the inputs for the learning stage. This phase includes noise filtering, feature normalization, categorical encoding using one-hot or ordinal schemes depending on the feature type, class balancing through techniques such as SMOTE or undersampling, and feature selection based on variance analysis, correlation measures, or preliminary estimates of predictive importance.

After preprocessing, the datasets are processed by two separate classification models. Although the same learning algorithm, such as XGBoost, can be employed for both traffic types, each model is trained on a dedicated dataset, one for CAN traffic and another for Wi-Fi traffic, with the objective of identifying anomalies or intrusion events. This methodological separation at the classification layer is essential due to the heterogeneous nature of CAN and Wi-Fi features, as it avoids feature incompatibility and preserves detection fidelity. By isolating the learning process, each model can fully exploit domain-specific characteristics without interference from unrelated traffic patterns, leading to more reliable classification results.

The main integration and methodological contribution of the system occurs in the subsequent explainability and semantic interpretation stages. These stages unify the technical outputs produced by the independent classifiers into a single explanation pipeline based on XAI and generative models, addressing the research gap discussed in Section 4.4. For instance, fields such as `Clu.Odometer` and `EPS.Spd` may indicate spoofing attacks in the CAN domain [Jeong *et al.*, 2024], while features such as `wlan.duration` and `frame.len` are commonly associated with Evil Twin attacks in the Wi-Fi domain [Khani *et al.*, 2024; Gupta and Seeja, 2024].

In addition to detecting intrusions, this stage provides input for explainability and visualization. Both classification models are transferred to the next stage to support these functions for downstream analysis..

5.2 Explainability and Visualization

To enhance interpretability, each classifier is followed by an explainability module that applies post hoc techniques such as SHAP to quantify feature contributions and clarify classification decisions for each instance.

The output of this process is presented through *explainable visualizations*, enabling human operators or automated auditing mechanisms to trace the reasoning behind each alert. These visual summaries make the system's internal behavior transparent and support a fine-grained understanding of model outputs during incident investigation.

While XAI-based tools provide summarized and insightful information, they often require technical knowledge of the application domain, such as understanding Wi-Fi and CAN frame details. As a result, human interpretability at this stage is primarily limited to domain experts.

5.3 Semantic Interpretation

To make AI-based decision-making more accessible, our architecture includes a semantic interpretation layer that generates natural language descriptions of each detection event. This module processes the structured outputs from the explainability stage and reformulates them into concise textual summaries. Instead of requiring analysts to interpret technical plots, the system provides explanations such as:

> *"The communication channel was occupied for an excessive period by beacon-type transmissions, suggesting a potential impersonation attempt via a rogue access point (Evil Twin)."*

This component facilitates communication between the system and its users, particularly in operational contexts where justifications must be recorded, reported, or understood by non-expert stakeholders. By translating technical outputs into natural language, this layer ensures traceability and clarity even for users without deep domain knowledge.

6 Evaluation

We evaluate our approach through four main components: the simulated scenario for our proof of concept (Section 6.1), classification results (Section 6.2), natural language explainability outcomes (Section 6.3), and a discussion on the approach's effectiveness (Section 6.4).

6.1 Scenario and Implementation

Figure 2 shows how we integrated the proposed IDS architecture within a modern vehicle's communication infrastructure. The Telematics Control Unit (TCU) serves as the central hub connecting internal CAN systems with external wireless networks used for connectivity.

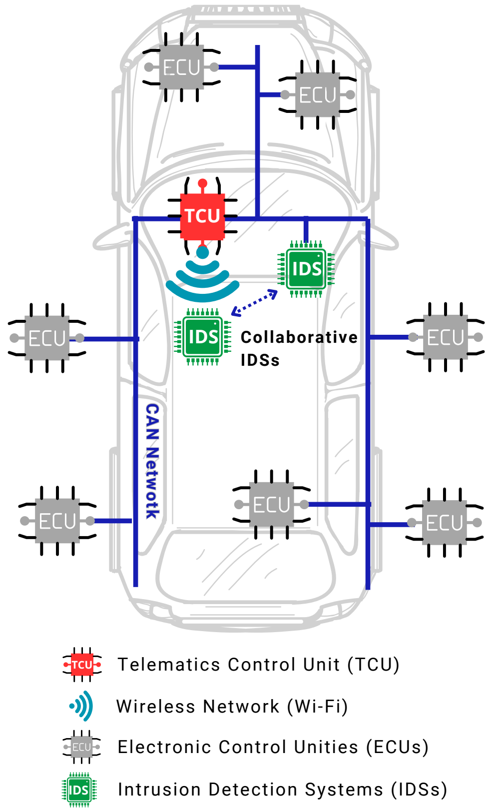

Figure 2. Overview of the IDS architecture showing component interactions and data flow.

To emulate realistic vehicular network traffic, we used two publicly available datasets that represent both external and internal communication channels. For wireless network analysis, we selected samples from the AWID2 dataset [Kolias *et al.*, 2016], including Normal traffic and impersonation-based attacks, namely Cafe Latte, Hirte, and Evil Twin. The selection was intentionally restricted to impersonation attacks and their corresponding benign samples in order to focus the analysis on a well-defined and practically relevant threat class.

To reduce class imbalance and mitigate bias during model training, we applied an undersampling strategy to enforce a strict 1:1 ratio between attack and normal samples, resulting in 20,079 instances per class. This controlled balance prevents the classifier from being biased toward the predominant benign traffic class and enables a fair evaluation of detection performance.

For in-vehicle communication analysis, we used the X-CANIDS dataset [Jeong *et al.*, 2024], which provides representative CAN traffic traces containing Normal, Fuzzing, and Fabrication attack classes. Following the same preprocessing principles, the dataset was balanced by combining 392,387 benign instances with an equal number of malicious instances, yielding a 1:1 macro-class distribution. This strategy supports robust supervised learning while maintaining consistency across both communication domains. Together, these datasets provide a controlled yet representative test environment for evaluating the proposed architecture across internal and external vehicular networks.

The architecture pipeline was implemented using Python as the primary programming language. Data preprocessing and manipulation were performed using the `pandas` and `numpy` libraries, covering tasks such as data cleaning, transformation, temporal aggregation, and categorical encoding. For classification, we employed XGBoost through the `xgboost` library to analyze both CAN and Wi-Fi traffic and to distinguish between benign and malicious samples. XGBoost was selected due to its strong performance and native compatibility with SHAP, which was used to generate feature-level explanations via the `shap` library.

For semantic interpretation, we integrated multiple large language models through their respective APIs, including ChatGPT 4o, DeepSeek V3, and Gemini 2.5 Flash. These models were used to translate SHAP-based feature attributions into human-readable explanations, enabling analysts to interpret detected intrusions in natural language.

6.2 Classification Results

Before conducting explainability analysis, we evaluated model performance on both Wi-Fi and CAN intrusion scenarios to ensure our interpretability techniques would be applied to reliable detection systems. Figure 3 compares four key metrics—Precision, Recall, F1-Score, and Accuracy—for CAN attacks (fuzzing and fabrication) and Wi-Fi attacks (Evil Twin and Hirte) in our evaluation..

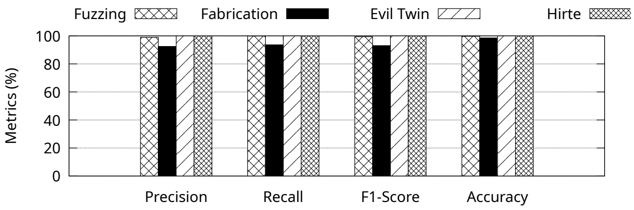

Figure 3. Performance metrics comparison for CAN and Wi-Fi attack detection.

For CAN-based detection, the model showed slightly better performance on fuzzing attacks, with 98.93% Precision, 99.89% Recall, 99.41% F1-Score, and 99.71% Accuracy. Fabrication attacks also achieved strong results with 92.42% Precision, 93.63% Recall, 93.04% F1-Score, and 98.43% Accuracy. The marginal difference suggests the model captures fuzzing patterns more effectively while maintaining robust fabrication detection across scenarios.

Wi-Fi attack detection performed exceptionally well, with both Evil Twin and Hirte attacks exceeding 99% across all metrics. These results demonstrate the model's capability to reliably distinguish malicious wireless communications from normal traffic in pratice.

The high classification performance establishes a trustworthy foundation for our explainability pipeline. When models achieve such consistent results, the subsequent SHAP analysis and LLM generated explanations can more accurately represent the actual decision making process. This reliability is particularly important for security applications where explanation fidelity directly impacts operational trust.

6.3 Explainability Results

We analyze the XAI results generated through SHAP summary plots and their natural language interpretations using multiple LLM prompts. Each prompt examines different aspects of attacker behavior to provide comprehensive insights.

The SHAP summary plots visualize feature importance, where each dot represents a data instance and each row corresponds to a vehicle system feature (CAN) or 802.11 frame field (Wi-Fi). The horizontal position shows the SHAP value, indicating the feature's impact on model decisions: positive values suggest intrusion detection, while negative values indicate normal behavior. Color coding represents feature values, with blue for low values and red for high values.

To extract natural language explanations from the SHAP plots, we used six distinct prompts with three LLMs (ChatGPT 4o, DeepSeek V3, and Gemini 2.5 Flash):

- **#P1**: Describe the attacker's typical behavior based on the most influential SHAP features.
- **#P2**: Identify signs of manipulation or malicious actions in the SHAP plot.
- **#P3**: Determine if certain variables show consistent elevation/reduction during attacks.
- **#P4**: Analyze whether influential variables reveal attack mechanisms or sequences.
- **#P5**: Assess if the attacker attempts behavior camouflage or makes abrupt changes.
- **#P6**: Infer the attacker's likely actions and potential goals.

#P6: Infer the attacker's likely actions and potential goals. Although the generative AI component was configured to produce three independent outputs for each prompt, as described in Appendix A, Listing 5, the results reported in the following tables consolidate these responses into a single, representative narrative for each LLM. This consolidation facilitates a clearer and more focused comparative analysis across models.

The following subsections present and discuss the SHAP visualization results (Figures) together with the corresponding LLM interpretations (Tables) for each attack type. This structured presentation enables systematic comparison of explanatory behavior across different attack scenarios and interpretation strategies.

6.3.1 Fuzzing Attacks

Figure 4 shows the SHAP summary plot for *fuzzing* attacks detected by the XGBoost model trained on CAN data from the X-CANIDS dataset. The most impactful features include `5FA_CR_Wcs_ErrStat`, `042_CR_Datc_RearDrTempDispF`, and `042_CF_Datc_CO2_Warning`, which are related to environmental controls, error states, and internal comfort metrics. These indicators suggest that the attacker manipulates peripheral ECUs to introduce noise, thereby destabilizing the vehicle's internal network.

Table 5 presents the natural language outputs generated by the three LLMs in response to prompts #P1 through #P6. Across prompts #P1 to #P4, a largely consistent interpretation emerges. Both ChatGPT 4o and DeepSeek V3 describe an attack scenario in which the adversary injects a high volume of malformed or arbitrary messages into multiple ECUs. ChatGPT 4o emphasizes the brute-force and randomized nature of the attack, highlighting the absence of clear targeting.

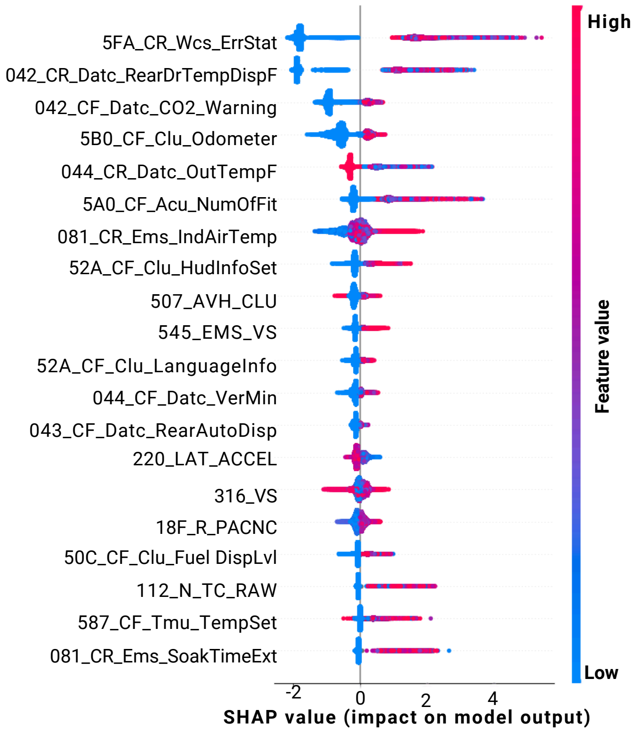

Figure 4. SHAP Values for Fuzzing Attack.

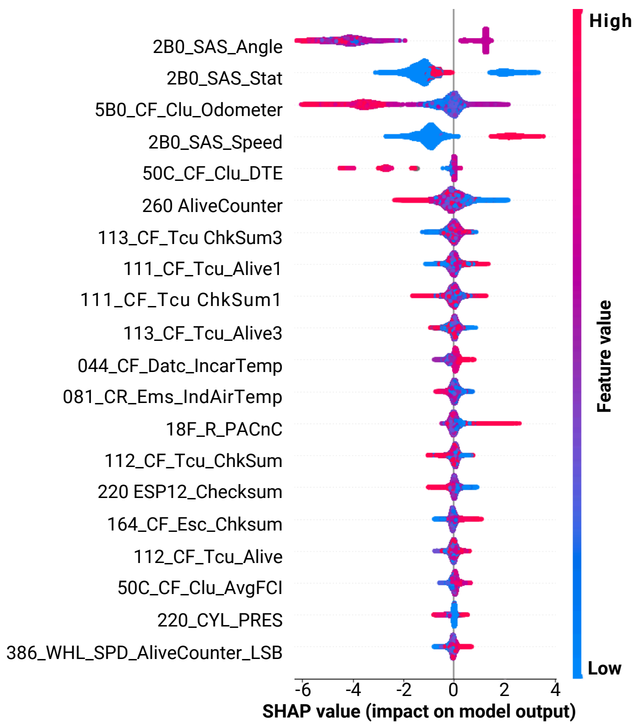

Figure 5. SHAP Values for Fabrication Attack.

DeepSeek V3 complements this view by identifying the affected subsystems, such as temperature and odometer sensors. Gemini 2.5 Flash reinforces these interpretations by focusing on the presence of out-of-range values and the attacker's apparent intent to probe system fault tolerance.

Some explanations provided by the LLMs emphasize features that are not among the most visually prominent in the SHAP summary plot. For example, references to `220_LAT_ACCEL` reflect the models' ability to combine statistical feature attribution with semantic knowledge about the criticality of specific ECUs and their role in potential attack progression. This behavior illustrates how LLM-based interpretation can complement SHAP visualizations by providing context that is not immediately apparent from feature ranking alone.

6.3.2 Fabrication Attacks

We analyze fabrication attacks using the SHAP summary plot in Figure 5, evaluated with our XGBoost model trained on the X-CANIDS dataset. The most influential features include `2B0_SAS_Angle`, `2B0_SAS_Stat`, and `5B0_CF_Clu_Odometer`, which reveal the attacker's strategy of forging messages that imitate legitimate vehicle telemetry, particularly targeting steering and odometer data systems at the ECU level.

Table 6 shows the LLM responses to prompts #P1 through #P6. For prompts #P1 to #P4, we observe that all three LLMs recognize the injection of fabricated messages into the CAN bus. ChatGPT 4o notes the attack's sophistication, highlighting the use of alive counters and checksums to evade detection. DeepSeek V3 specifically identifies steering and stability systems as primary targets, indicating potential safety implications. Gemini 2.5 Flash presents a contrasting view, interpreting the signal changes as obvious and easily detectable, suggesting a more aggressive approach.

The responses to prompt #P5 reveal a clear divergence in interpretation. While ChatGPT 4o and DeepSeek V3 describe a stealthy attack that mimics normal traffic patterns, Gemini 2.5 Flash characterizes the observed alterations as abrupt and easily detectable. This contrast reflects the different analytical emphases adopted by each model. ChatGPT 4o and DeepSeek V3 prioritize the plausibility of spoofing techniques and camouflage strategies, whereas Gemini 2.5 Flash focuses on overt deviations from expected behavior.

Rather than indicating inconsistency, this divergence highlights the complementary nature of a multi-LLM interpretation strategy. By exposing alternative perspectives on the same attack scenario, the architecture supports a more nuanced understanding of attacker behavior, balancing interpretations centered on stealth against those emphasizing detectability.

For prompt #P6, we find consensus among all models that the attacker's primary goal involves altering vehicle behavior through falsified telemetry data. This manipulation could potentially bypass security measures or trigger unexpected responses from ECUs, though the methods of execution differ in their subtlety according to each LLM's interpretation.

6.3.3 Hirte Attacks

We examine the Hirte attack through the SHAP summary plot in Figure 6, generated from our XGBoost model trained on *hirte* and *normal* samples from the AWID dataset. Our analysis reveals three key features that dominate the detection: `radiotap.datarate`, `wlan.duration`, and `wlan.fc.frag`. These parameters, which control transmission rate, frame duration, and fragmentation, indicate the attacker's strategy of manipulating physical and MAC layer characteristics to blend malicious activity with legitimate traffic patterns. These findings support existing research showing Hirte attacks exploit timing and QoS vulnerabilities to evade

Table 5. Natural language explanations for the Fuzzing attack in the CAN communication channel.

Prompt	ChatGPT 4o	DeepSeek V3	Gemini 2.5 Flash
#P1	The attacker floods the CAN bus with random messages targeting unrelated ECUs.	The attacker floods the vehicle network with malformed CAN messages targeting critical sensors (e.g., temperature, odometer).	The attacker's behavior involves manipulating or generating messages related to vehicle status, errors, and sensor readings.
#P2	Yes, widespread anomalies across different signals indicate malicious injection patterns.	Yes, anomalies in features like 220_LAT_ACCEL or 5FA_CR_Wcs_ErrStat suggest deliberate fuzzing to trigger errors.	Yes, strong positive SHAP values for features like error status and sensor readings clearly indicate malicious data manipulation.
#P3	The lack of pattern suggests randomized injections aiming to trigger fault responses.	Variables like 044_CR_Datc_OutTempF or 540_CF_Acu_NumOfFit show extreme values, indicating injected garbage data.	Yes, many key variables show high values systematically elevated during attacks, likely caused by the attacker injecting abnormal or out-of-range data.
#P4	The dispersed impact reveals a non-sequential, brute-force style of attack.	The attacker likely starts with non-critical modules (e.g., climate control) before targeting safety-critical systems (e.g., 220_LAT_ACCEL).	The variables strongly reveal the attack mechanism involves injecting abnormal data values into specific vehicle communication fields, not the temporal order of events.
#P5	The changes are abrupt and clearly deviate from normal traffic, making them easy to detect.	Changes are abrupt (e.g., implausible odometer values), suggesting no effort to mimic legitimate traffic.	The significant impacts and correlation of high feature values with attack prediction suggest the changes are distinct and likely detectable, not camouflaged like normal traffic.
#P6	The attacker is injecting arbitrary CAN frames to disrupt communication and test system resilience.	The attacker disrupts vehicle systems by overwhelming CAN bus messages, possibly to cause malfunctions or crashes.	The attacker is injecting abnormal data into vehicle parameters, aiming to trigger errors or discover vulnerabilities in how the system processes unexpected inputs.

detection [Dresch *et al.*, 2024].

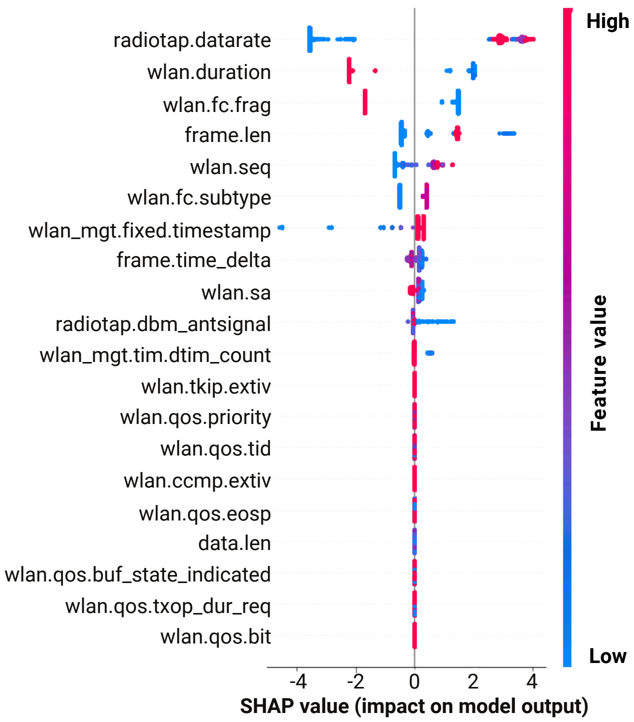

Figure 6. SHAP value distributions for Hirte attack detection.

The SHAP plot demonstrates how both high and low values of features like `frame.len`, `wlan.seq`, and `wlan.fc.subtype` influence model decisions. This pattern reflects the Hirte attack's dual strategy of using fragmentation and sequence spoofing while carefully manipulating control fields. By varying packet sizes and protocol headers, attackers exploit weaknesses in client-side packet processing while maintaining the appearance of normal network activity at the traffic level.

Table 7 presents the LLM-generated explanations of Hirte attack patterns. All three models consistently identify the attack's reliance on low-level parameter manipulation, particularly in Quality of Service fields, timing information, and frame header attributes. This agreement reinforces our observations regarding the attack methodology, which aims to subtly disrupt communication while closely mimicking legitimate traffic patterns.

The emphasis placed by the LLMs on Quality of Service features, even when these attributes do not rank among the most statistically prominent in Figure 6, reflects their semantic understanding of the Hirte attack model. Rather than relying solely on rank-ordered feature importance, the models prioritize protocol-level exploitation patterns, highlighting aspects that are conceptually central to the attack mechanism.

We observe strong alignment across prompts #P1 to #P4 in the LLM responses. Both ChatGPT 4o and DeepSeek V3 highlight the attacker's approach of blending malicious activity with normal network patterns through QoS and timing manipulation. In #P1, they describe how the attacker adjusts QoS fields and timing parameters to appear legitimate, with DeepSeek noting the subtle degradation of communication quality that results. This complements ChatGPT's focus on how these tactics evade detection systems in #P3.

Gemini 2.5 Flash agrees on the targeted features but presents a more detailed enumeration of affected parameters, consistently mentioning data rate, duration, fragmentation, sequence, and timestamp across #P1 and #P3. However, Gemini takes a different position in #P5, arguing that the attack patterns are readily detectable based on SHAP value magnitudes, contrasting with ChatGPT and DeepSeek's characterization of a more subtle infiltration.

For prompt #P4, all models recognize sequencing and timing manipulation as core components of the attack. ChatGPT goes further by suggesting the attacker replicates expected packet ordering, while DeepSeek and Gemini maintain

Table 6. Natural language explanations for the Fabrication attack in the CAN communication channel.

ID	ChatGPT 4o	DeepSeek V3	Gemini 2.5 Flash
#P1	The attacker crafts realistic CAN messages by imitating legitimate patterns like alive counters and checksums.	The attacker forges CAN messages (e.g., steering angle, odometer) to spoof vehicle sensors or controls.	Attacker manipulates parameters (elevating angle/odometer, lowering statistics) to trigger ”Fabrication” detection.
#P2	Yes, the SHAP values show tampering in critical message integrity fields like checksums.	Yes, features like 280_SAS_Angle and checksums (220_ESP12_Checksum) show tampering, indicating message fabrication.	Yes, clear manipulation patterns (elevated angle/odometer, reduced statistics) are linked to ”Fabrication.”
#P3	Crafted values in alive counters and checksums reveal deliberate fabrication of trusted signals.	Values like 580_CF_Clu_Odometer or 280_SAS_Speed are altered to fake readings, suggesting spoofed data injection.	Consistently elevated angle/odometer and reduced statistics indicate intentional data alteration/injection.
#P4	The attacker maintains expected message order and timing to mimic real ECU communication.	The attacker likely targets safety-critical systems (e.g., steering, ESP) first, then alters less critical metrics (e.g., odometer).	Influential variables reveal *what* and *how* is altered, but not the attack’s order or specific mechanism.
#P5	The forged data closely imitates normal patterns, indicating an attempt to evade detection.	Some features (e.g., checksums) mimic valid messages, but others (e.g., implausible steering angles) are clearly malicious.	Changes are abrupt and easily detectable, with no signs of camouflage.
#P6	The attacker is injecting fake but plausible CAN frames to manipulate vehicle behavior without triggering alerts.	The attacker fabricates CAN messages to manipulate vehicle behavior (e.g., false speed readings) or bypass security checks.	Attacker manipulates vehicle parameters to trigger ”Fabrication,” likely aiming to cause errors or exploit system vulnerabilities.

a more conservative interpretation, describing feature alterations without speculating about specific event sequences.

The responses to #P6 show renewed agreement about the attacker’s objectives. All three models identify communication disruption and malicious packet injection as primary goals, with ChatGPT emphasizing covert access, DeepSeek considering denial-of-service possibilities, and Gemini expanding the scope to include exploitation through control and timing manipulation techniques.

6.3.4 Evil Twin Attacks

We analyze Evil Twin attacks using the SHAP summary plot in Figure 7, generated from our XGBoost model trained on *evil twin* and *normal* samples from the AWID dataset. Our examination reveals three critical detection features: `wlan_mgt.fixed.capabilities.preamble`, `radiotap.datarate`, and `frame.len`. These parameters, which control frame transmission characteristics, signal rates, and packet structure, are commonly forged by attackers when creating rogue access points that impersonate legitimate ones. This technique enables man-in-the-middle attacks by deceiving clients into connecting to the malicious AP under attacker control [Dresch *et al.*, 2024].

The SHAP plot demonstrates how both high and low values of these features influence model decisions. This pattern reflects the attacker’s strategy of carefully adjusting signal strength, frame sizes, and management field contents to blend with legitimate network traffic. By maintaining plausible transmission characteristics while subtly altering key parameters, the attack aims to evade detection while maintaining its malicious functionality.

Table 7 presents the LLM generated explanations of Evil Twin attack patterns. While all three models agree on the core impersonation strategy - where attackers mimic legitimate APs by manipulating transmission characteristics - each provides unique insights into the attack methodology. This complementary perspective helps build a more comprehensive understanding of how Evil Twin attacks operate in wireless networks under real conditions..

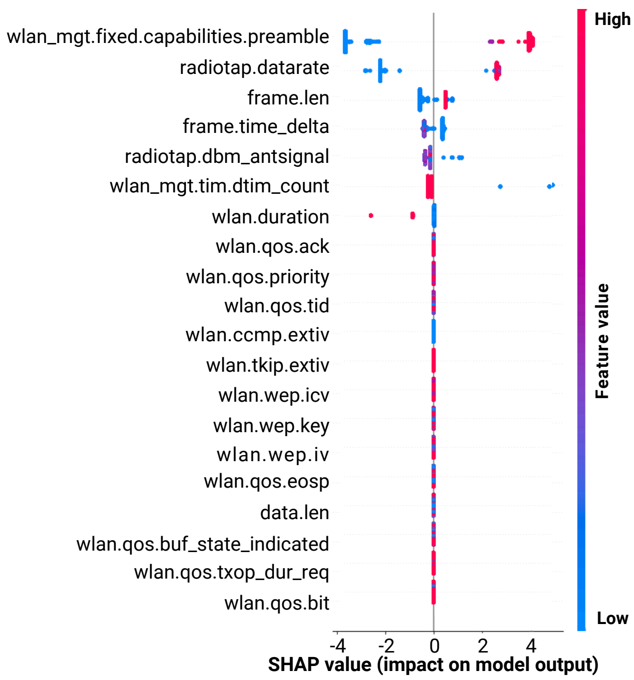

Figure 7. SHAP value distributions for Evil Twin attack detection.

Our analysis of prompts #P1 through #P3 reveals strong agreement about the attacker’s manipulation of physical and QoS parameters to mimic legitimate access points. ChatGPT details the crafting of beacon and QoS fields, while DeepSeek specifically notes adjustments to data rate and signal strength. Gemini contributes additional insight by identifying preamble modifications - a detail overlooked by the other models. When examining persistent value manipulation in #P3, we observe ChatGPT focusing on signal strength anomalies and buffer irregularities, whereas Gemini reiterates its findings regarding elevated preamble and data rate configurations across packets.

The responses to prompt #P2 uncover further distinctions in model interpretations. Both ChatGPT and DeepSeek identify encryption-related anomalies, including WEP IV and QoS buffer fields. Gemini maintains its consistent focus on

Table 7. Natural language explanations for the Hirte attack in the Wi-Fi communication channel.

Prompt	ChatGPT 4o	DeepSeek V3	Gemini 2.5 Flash
#P1	The attacker manipulates QoS fields and timing information to simulate normal traffic while subtly interfering with communication.	The attacker likely manipulates QoS fields and timing-related parameters to mimic legitimate traffic patterns while subtly disrupting communication flow.	The attacker manipulates transmission parameters like data rate, duration, fragmentation, sequence, and timestamp.
#P2	Yes, the prominence of QoS-related features suggests intentional interference with traffic prioritization, typical of targeted disruptions.	Yes, the plot indicates manipulation of QoS and timing features, suggesting a Hirte attack aimed at disrupting Wi-Fi communication.	Yes, abnormal values in data rate, duration, fragmentation, sequence, and timestamp show clear signs of malicious manipulation.
#P3	Yes, systematic changes in fields like 'wlan.qos.priority' and 'frame.time_delta' indicate deliberate timing and priority alterations to mislead detection mechanisms.	Variables like QoS priority and timing parameters show systematic changes, likely due to the attacker intentionally altering packet timing and priority to cause disruptions.	Yes, data rate/duration/frag are high, sequence/timestamp are low, indicating crafted packets exploiting header and timing fields.
#P4	The dominance of timing and sequence features suggests the attack unfolds by mimicking expected packet order while injecting anomalies.	The influential variables suggest the attack involves manipulating packet sequencing and QoS settings to degrade network performance systematically.	The variables indicate the mechanism of packet manipulation (headers, timing), but not the sequence of attack events.
#P5	The attacker seems to camouflage their behavior by adjusting subtle features rather than introducing abrupt anomalies.	The attacker appears to camouflage their actions by subtly altering traffic patterns, making the changes less detectable.	Significant SHAP impacts from specific value ranges suggest the changes are distinct and likely detectable, not camouflaged.
#P6	The attacker appears to replicate legitimate traffic flows to gain access or inject malicious packets without triggering alarms.	The attacker is manipulating QoS and timing parameters to disrupt Wi-Fi communication, possibly aiming to degrade network performance or cause denial of service.	The attacker is sending malicious packets with altered network control and timing data, possibly for disruption or exploitation.

Table 8. Natural language explanations for the Evil Twin attack in the Wi-Fi communication channel.

Prompt	Output		
	ChatGPT 4o	**DeepSeek V3**	**Gemini 2.5 Flash**
#P1	The attacker mimics access point characteristics by forging beacon and QoS parameters.	The attacker likely manipulates data rates and signal strength to mimic a legitimate AP.	The attacker likely manipulates physical layer parameters like preamble and data rate.
#P2	Yes, abnormal values in encryption and buffer QoS fields reveal crafted management frames.	Yes, features like WEP IV and QoS priority show anomalies typical of Evil Twin attacks.	Yes, high values in preamble and data rate characteristics indicate malicious activity detected by the model.
#P3	Repeated anomalies in signal strength and buffer indicators suggest persistent impersonation attempts.	Variables like frame length and signal strength are altered to deceive devices into connecting.	Yes, preamble and data rate appear consistently high, suggesting the attacker is configuring specific network parameters.
#P4	The manipulation of beacon timing and QoS patterns points to a structured impersonation strategy.	The attack likely starts by spoofing beacon frames, then adjusts QoS to prioritize malicious traffic.	The variables reveal changes in packet structure and characteristics, indicating the attack mechanism, not the order.
#P5	The attacker camouflages behavior by aligning with typical frame sizes and QoS structures.	The attacker subtly mimics normal traffic but leaves detectable traces in encryption-related features.	Significant SHAP values suggest changes are impactful and potentially detectable, not camouflaged.
#P6	The attacker is simulating a fake access point to deceive clients into connecting.	The attacker is likely hijacking sessions or stealing credentials via a fake AP.	The attacker is altering network transmission parameters, possibly to impersonate a legitimate network.

physical-layer alterations, demonstrating its particular attention to low-level transmission characteristics throughout the analysis in this section..

For prompt #P4, we find ChatGPT and DeepSeek detecting sequential patterns in the attack strategy. ChatGPT describes coordinated manipulation of beacon timing and QoS patterns as part of a deliberate impersonation scheme. DeepSeek proposes a more defined progression, beginning with spoofed beacons followed by QoS parameter adjustments. Gemini offers a contrasting perspective, acknowledging the altered variables reflect the attack mechanism but questioning whether they reveal any temporal sequence.

Prompt #P5 exposes differing views on attack stealth. ChatGPT and DeepSeek both identify attempts to blend with normal traffic patterns - ChatGPT through typical frame sizes and QoS alignment, DeepSeek through subtle mimicry with detectable encryption artifacts. Gemini presents a divergent assessment, arguing the high SHAP values indicate obvious, uncamouflaged anomalies.

Finally, prompt #P6 shows all models converging on the fundamental attack objective: AP impersonation. ChatGPT frames this as deception to lure client connections, DeepSeek extends the analysis to potential credential theft and session hijacking, while Gemini remains anchored in its examination of transmission parameter manipulation. This multi-faceted interpretation provides comprehensive coverage of both the technical execution and potential consequences of Evil Twin attacks in real deployments.

6.4 Discussion

Our proof of concept demonstrates that combining SHAP-based feature attribution with LLM-generated semantic interpretation enables both high detection performance and human-interpretable explanations across different vehicular communication channels. The XGBoost classifiers achieved near-perfect detection rates for Wi-Fi attacks (`Evil Twin` and `Hirte`) while maintaining F1-scores above 93% for more challenging CAN bus attacks (*fabrication*). These results, obtained using established benchmark datasets (AWID2 and X-CANIDS), support the effectiveness of the SHAP explanation process as well as the ability of LLMs to translate feature attributions into meaningful natural language narratives.

Direct quantitative comparison with related work that emphasizes performance metrics is limited by the lack of semantic interpretability in many legacy datasets, such as the Car-Hacking Dataset, which constrains the meaningful application of XAI techniques. Accordingly, the primary objective of this work is not to pursue marginal improvements in detection metrics over non-interpretable baselines, but to demonstrate explanation fidelity, semantic transparency, and practical interpretability in vehicular intrusion detection.

The SHAP analysis revealed that the most influential detection features correspond to known attack patterns in each domain. For CAN attacks, we found that fabrication attempts primarily targeted critical motion signals like `2B0_SAS_Angle` and `5B0_CF_Clu_Odometer`, consistent with attempts to deceive vehicle safety systems [Dresch *et al.*, 2024]. Fuzzing attacks, by contrast, showed distinctive patterns in cabin sensor frames and error status flags, matching expected noise injection behavior.

In Wi-Fi attacks, we observed that physical layer parameters (`radiotap.datarate`, `wlan.duration`) and fragmentation control (`wlan.fc.frag`) were most significant for detecting Hirte attacks, while management frame characteristics (`wlan_mgt.fixed.capabilities.preamble`) proved crucial for identifying Evil Twin attempts. These findings align with known attack methodologies and provide actionable insights for security analysts.

While SHAP offers mathematically rigorous explanations, its visual representations remain challenging for non-experts to interpret. Our LLM integration addresses this limitation by translating complex SHAP plots into clear natural language explanations. Although we observed some variation in the LLMs' outputs, they consistently converged on core attack characteristics, providing complementary perspectives that balance specificity and clarity. This two-layer approach effectively addresses the interpretability gap identified in recent XAI research for network security [Zytek *et al.*, 2024], making the system more accessible for operational use in industrial environments at scale.

7 Threats to Validity

While this work presents a novel integration of explainable artificial intelligence and large language models for vehicular intrusion detection, several threats to validity may affect the interpretation and generalization of the results. Following established guidelines for empirical research [Wohlin *et al.*, 2012], we discuss these threats in terms of construct validity, internal validity, external validity, and conclusion validity.

7.1 Construct Validity

Construct validity concerns whether the study measures what it intends to measure.

This work relies on two widely used benchmark datasets, X-CANIDS and AWID2, which introduce temporal and contextual limitations. X-CANIDS was collected from a 2017 Hyundai Sonata and may not fully represent modern vehicular architectures that increasingly adopt CAN-FD, Automotive Ethernet, and enhanced ECU security mechanisms. Similarly, AWID2 was captured in 2016 and predates recent Wi-Fi standards such as WPA3 and several important attack classes discovered in subsequent years. As a result, the reported performance metrics should be interpreted as representative of the attack types and conditions present in these datasets, rather than as a comprehensive assessment of current or future vehicular threat landscapes.

The evaluation focuses on a limited set of attack types, namely fuzzing and fabrication in the CAN domain, and Evil Twin and Hirte attacks in the Wi-Fi domain. Although these attacks are well documented and practically relevant, they represent only a subset of possible adversarial behaviors. More complex scenarios such as multi-stage attacks, coordinated cross-layer attacks, zero-day exploits, or stealthy long-lived intrusions are not covered. Therefore, the reported detection performance reflects effectiveness on known attack patterns rather than general robustness against all possible threats.

Explainability is assessed through qualitative analysis of LLM-generated narratives guided by predefined prompts. While this approach enables semantic interpretation of SHAP outputs, it does not include formal or quantitative metrics for explanation quality. Properties such as fidelity to feature attributions, completeness of coverage, and operational usefulness are not measured objectively. Differences observed across LLMs highlight this limitation, as no ground truth annotations or human evaluation benchmarks are available to determine explanation correctness.

To address class imbalance, the datasets were artificially balanced using undersampling to achieve equal proportions of attack and normal samples. Although this is a common practice in intrusion detection research, it does not reflect real-world traffic conditions, where malicious events are rare. This choice may lead to optimistic performance estimates, particularly with respect to false positive rates, and limits the direct applicability of the results to operational environments.

Future work should incorporate human-centered evaluation methods to assess explainability more directly. Controlled user studies with domain experts could measure comprehension, trust, cognitive load, and practical usefulness, allowing explainability to be evaluated as a property of human understanding rather than solely as semantic consistency with model outputs.

7.2 Internal Validity

Internal validity addresses whether observed effects can be attributed to the proposed approach rather than to confounding factors.

The study exclusively employs XGBoost as the classification algorithm, primarily due to its strong performance and compatibility with SHAP. However, no systematic comparison with alternative models, such as deep learning architectures, is conducted. As a result, it is not possible to determine whether the observed performance is specific to XGBoost or influenced by dataset characteristics. Although the proposed explainability architecture is model-agnostic in principle, the lack of algorithmic ablation limits claims about optimal model choice.

The semantic interpretation layer relies on multiple large language models that exhibit non-deterministic behavior. As shown in the results, different LLMs may produce partially divergent interpretations for the same SHAP inputs. This variability arises from inherent randomness in text generation and differences in model training. The current approach does not include automated mechanisms to detect hallucinations, resolve conflicting explanations, or formally verify alignment with feature attributions. Consequently, explanation reliability may depend on expert oversight in practical use.

Model training and evaluation are based on fixed train test splits provided by the original dataset publications, without extensive hyperparameter optimization or cross-validation. While this practice supports reproducibility and fair comparison with prior work, it introduces the risk of sensitivity to specific data partitions. Confidence intervals or repeated evaluations would strengthen statistical claims, although the primary contribution of this work lies in explainability rather than marginal performance gains.

Future research should analyze explanation stability across multiple training runs, random seeds, and parameter settings. Comparing SHAP distributions and generated narratives under controlled variations would help identify stable explanatory patterns and improve reproducibility.

7.3 External Validity

External validity concerns the generalizability of the findings to other contexts and deployment settings.

The CAN-based evaluation relies on data from a single vehicle model produced by one manufacturer. CAN message identifiers, signal encodings, and timing characteristics vary substantially across manufacturers, which limits direct transferability of learned feature patterns. While the underlying attack mechanisms generalize conceptually, the specific features and decision boundaries learned in this study may not apply to vehicles from other vendors. Broader validation would require multi-manufacturer datasets, which remain scarce due to proprietary constraints.

All experiments are conducted on static, pre-recorded datasets in a controlled environment. The proposed pipeline has not been validated under real-world operational conditions that involve strict latency requirements, physical layer noise, mobility effects, and hardware constraints. In particular, the combined latency of classification, SHAP computation, and LLM-based explanation generation is incompatible with real-time safety-critical decision making. The current design is therefore more suitable for post-incident analysis and offline auditing than for real-time prevention.

The evaluation does not account for concept drift, where normal behavior and attack patterns evolve over time due to software updates, changing usage patterns, or adaptive adversaries. Without mechanisms for continuous learning or periodic retraining, model performance may degrade in long-term deployments.

In addition, the work does not demonstrate integration with automotive industry standards such as ISO 26262, AUTOSAR Adaptive, or UNECE R155 and R156. Compliance with these standards is essential for production deployment, particularly given the non-deterministic nature of LLM components. Addressing these aspects would require close collaboration with industry partners.

To strengthen external validity, future work should evaluate the explanation pipeline under naturally imbalanced data distributions and in more realistic operational settings. Such studies would clarify how feature attributions and explanations behave under real-world traffic conditions.

7.4 Conclusion Validity

Conclusion validity relates to the strength of the relationship between the applied methods and the reported outcomes.

The study does not include direct experimental comparison with state-of-the-art intrusion detection systems using identical datasets and evaluation protocols. Differences in preprocessing, data splits, and experimental setups across prior work make fair head-to-head comparison difficult. As a result, claims about performance superiority are not supported. The primary contribution lies in explainability rather than in outperforming existing detectors.

Performance metrics are reported from single experimental runs without statistical significance testing or confidence intervals. Although classification results are deterministic under fixed random seeds, variability in explanation generation is not statistically analyzed. Multiple replications with formal testing would provide stronger evidence of result stability.

Finally, the system is not evaluated against adversarial attacks designed to evade detection or manipulate explanations. Both machine learning models and post hoc explanation methods are known to be vulnerable to such attacks. In high-stakes automotive security contexts, the absence of adversarial robustness evaluation represents an important limitation.

Future work should explore mechanisms to audit and constrain LLM-generated explanations, including stricter grounding to feature attributions, explicit uncertainty signaling, and verification steps. Comparative studies between template-based and generative explanations could further clarify trade-offs between expressiveness, faithfulness, and reliability.

By acknowledging these threats, we contextualize the results appropriately and outline clear directions for future research. These limitations do not diminish the core contribution of this work, which demonstrates the feasibility and practical value of integrating SHAP-based explainability with LLM-driven semantic interpretation for vehicular cybersecurity.

8 Conclusion

This paper proposed a conceptual architecture for explainable intrusion detection in connected vehicles, addressing the joint analysis of two critical communication layers: the internal

CAN bus and the external Wi-Fi interface. By considering these layers together, the work reflects the growing complexity of modern vehicular attack surfaces, where internal and external networks interact and cannot be treated in isolation.

The proposed architecture combines supervised learning for detection, SHAP for feature-level interpretability, and large language models for semantic explanation. XGBoost was employed as the classification backbone, while SHAP was used to quantify the contribution of protocol-level features to intrusion decisions. On top of this, LLMs were integrated to translate technical attribution outputs into natural language descriptions of attacker behavior. This design enables an intrusion detection system that not only achieves high detection performance, but also produces explanations that are accessible and meaningful to human analysts.

Experimental results using the X-CANIDS and AWID2 datasets show that the approach achieves strong classification performance across different attack types in both CAN and Wi-Fi domains. More importantly, the SHAP analysis consistently highlighted features that align with known attack mechanisms, and the LLM-generated explanations converged on coherent descriptions of attacker strategies, objectives, and behavioral patterns. These findings demonstrate the feasibility of combining XAI and generative models to improve transparency and auditability in vehicular intrusion detection.

At the same time, this work positions explainability as a central design goal rather than a secondary visualization step. While SHAP provides mathematically grounded explanations, its outputs remain difficult to interpret for non-experts. The integration of LLMs addresses this gap by enabling semantic interpretation, bridging low-level feature attribution and higher-level security reasoning. However, as discussed, the quality, stability, and reliability of LLM-generated explanations remain open challenges that require further investigation.

As future work, we plan to evaluate the proposed architecture in more realistic and operational settings, including naturally imbalanced traffic, multi-vendor vehicle data, and longitudinal scenarios subject to concept drift. We also intend to explore alternative XAI techniques, different model families, and mechanisms for formally assessing explanation quality through human-centered evaluations. Finally, investigating methods to constrain, validate, and audit LLM-generated explanations will be essential for deploying explainable intrusion detection systems in safety-critical vehicular environments.

Overall, this work demonstrates that integrating SHAP-based explainability with LLM-driven semantic interpretation is a viable and promising direction for advancing transparency in vehicular cybersecurity. Rather than claiming a final solution, the paper establishes a methodological foundation for explainable, auditable, and human-centered intrusion detection in connected vehicles.

Declarations

Authors' Contributions

R.C.M., F.H.S., F.N.D., and V.E.Q. contributed to the conceptualization, implementation, and execution of the experiments. S.E.Q. and D.K. provided crucial methodological insights and supervised the research process. All authors actively participated in the discussion of the results, the drafting of the manuscript, and the final approval of the submitted version.

Competing interests

The authors declare that they have no competing interests.

Acknowledgements

We gratefully acknowledge the financial support provided by the Programa de Desenvolvimento Acadêmico (PDA) at Universidade Federal do Pampa (UNIPAMPA) and Fundação de Amparo à Pesquisa do Estado do Rio Grande do Sul (FAPERGS) through research scholarships and infrastructure funding. Our work benefited significantly from the computational resources made available by UNIPAMPA's Laboratory of Advanced Studies in Computing (LEA). We also extend our sincere appreciation to our colleagues and the anonymous reviewers whose valuable insights and constructive feedback helped strengthen this research.

Funding

This work was supported by the Programa de Desenvolvimento Acadêmico (PDA) at Universidade Federal do Pampa (UNIPAMPA) through research scholarships. Additional funding was provided by the Fundação de Amparo à Pesquisa do Estado do Rio Grande do Sul (FAPERGS) under grants 24/2551-0001368-7 and 24/2551-0000726-1, which supported infrastructure and research development. The authors gratefully acknowledge this financial support that made the research possible.

Availability of data and materials

The datasets and code utilized in this study are available upon request. For further information, please contact the corresponding author.

References

Alawida, M. *et al*. (2023). Explainable ai-enabled cybersecurity: A review of explainability methods in intrusion detection systems. *Computers & Security*, 127:102675. DOI: 10.1016/j.cose.2023.102675.

Ali, T. and Kostakos, P. (2023). Huntgpt: Integrating machine learning-based anomaly detection and explainable ai with large language models (llms).

Alnahdi, A. and Narain, S. (2024). Towards transparent intrusion detection: A coherence-based framework in explainable ai integrating large language models. In *2024 IEEE 6th International Conference on Trust, Privacy and Security in Intelligent Systems, and Applications (TPS-ISA)*, pages 87–96. DOI: 10.1109/TPS-ISA62245.2024.00020.

Aminanto, M. E. and Kim, K. (2017). Detecting impersonation attacks in wi-fi networks using deep learning approach. In *2017 International Workshop on Big Data and Information Security (IWBIS)*, pages 56–61. IEEE. DOI: 10.1007/978-3-319-56549-1_12.

Baral, S., Saha, S., and Haque, A. (2024). An adaptive end-to-end iot security framework using explainable ai and llms. In *2024 IEEE 10th World Forum on Internet of Things (WF-IoT)*, pages 469–474. DOI: 10.1109/WF-IoT62078.2024.10811456.

Bilal, A., Ebert, D., and Lin, B. (2025). Llms for explainable ai: A comprehensive survey.

De Vincenzi, M., Costantino, G., Matteucci, I., Fenzl, F., Plappert, C., Rieke, R., and Zelle, D. (2024). A systematic review on security attacks and countermeasures in automotive ethernet. *ACM Computing Surveys*, 56(6):1–38. DOI: 10.1145/3639433.

Ding, W., Alrashdi, I., Hawash, H., and Abdel-Basset, M. (2024). Deepsecdrive: An explainable deep learning framework for real-time detection of cyberattack in in-vehicle networks. *Inf. Sci.*, 658(C). DOI: 10.1016/j.ins.2023.120057.

Dresch, F. N., Scherer, F. H., Quincozes, S. E., and Kreutz, D. (2024). Modelos interpretáveis com inteligência artificial explicável (xai) na detecção de intrusões em redes intra-veiculares controller area network (can). In *Anais do XXIV Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais (SBSeg)*, pages 445–460, Florianópolis, SC. Sociedade Brasileira de Computação (SBC). DOI: 10.5753/sbseg.2024.236402.

Gimenes, E. H. L. (2024). Uma avaliação das características de redes wi-fi(802.11) para a detecção de ataques de personificação baseada em inteligência artificial explicável – xai. Universidade Federal de Uberlândia.

GmbH, R. B. (1991). *CAN Specification Version 2.0*. Bosch.

Gupta, N. and Seeja, K. R. (2024). Shap-based interpretability for wi-fi intrusion detection. *Journal of Cybersecurity*. DOI: 10.0000/gupta2024shap.

Han, M. L., Kwak, B. I., and Kim, H. K. (2018). Anomaly intrusion detection method for vehicular networks based on survival analysis. *Vehicular Communications*, 14:52–63. DOI: https://doi.org/10.1016/j.vehcom.2018.09.004.

Hong, Y.-W. and Yoo, D.-Y. (2024). Multiple intrusion detection using shapley additive explanations and a heterogeneous ensemble model in an unmanned aerial vehicle's controller area network. *Applied Sciences*, 14(13). DOI: 10.3390/app14135487.

Hsu, C.-C., Wu, I.-Z., and Liu, S.-M. (2024). Decoding ai complexity: Shap textual explanations via llm for improved model transparency. In *2024 International Conference on Consumer Electronics - Taiwan (ICCE-Taiwan)*, pages 197–198. DOI: 10.1109/ICCE-Taiwan62264.2024.10674465.

Jeong, S., Lee, S., Lee, H., and Kim, H. K. (2024). X-CANIDS: Signal-aware explainable intrusion detection system for controller area network-based in-vehicle network. *IEEE Transactions on Vehicular Technology*, 73(3):3230–3246. DOI: 10.1109/TVT.2023.3327275.

Kabilan, N., Ravi, V., and Sowmya, V. (2024). Unsupervised intrusion detection system for in-vehicle communication networks. *Journal of Safety Science and Resilience*, 5(2):119–129. DOI: https://doi.org/10.1016/j.jnlssr.2023.12.004.

Kang, M.-G., Woo, S., and Lee, H. (2016). Intrusion detection system using deep neural network for in-vehicle network security. In *2016 IEEE 2016 15th International Conference on Security and Management (SAM)*, pages 115–120. IEEE. DOI: 10.1371/journal.pone.0155781.

Khani, H. *et al*. (2024). A cnn-based intrusion detection system with shap-based feature analysis for wi-fi networks. *Future Generation Computer Systems*, 143:118–132. DOI: 10.1016/j.future.2023.10.011.

Khediri, A., Slimi, H., Yahiaoui, A., Derdour, M., Bendjenna, H., and Ghenai, C. E. (2024). Enhancing machine learning model interpretability in intrusion detection systems through shap explanations and llm-generated descriptions. In *2024 6th International Conference on Pattern Analysis and Intelligent Systems (PAIS)*, pages 1–6. DOI: 10.1109/PAIS62114.2024.10541168.

Kolias, C., Kambourakis, G., Stavrou, A., and Gritzalis, S. (2016). Intrusion detection in 802.11 networks: Empirical evaluation of threats and a public dataset. *IEEE Communications Surveys & Tutorials*, 18(1):184–208. DOI: 10.1109/COMST.2015.2402161.

Le, T.-T.-H., Suryanto, N., Kim, H., Ji, J., and Heo, S. (2023). Enhancing intrusion detection and explanations for imbalanced vehicle can network data. In *Proceedings of the 12th International Symposium on Information and Communication Technology*, SOICT '23, page 777–784, New York, NY, USA. Association for Computing Machinery. DOI: 10.1145/3628797.3628994.

Lim, B., Huerta, R., Sotelo, A., Quintela, A., and Kumar, P. (2025). Explicate: Enhancing phishing detection through explainable ai and llm-powered interpretability.

Liu, Y., Xue, L., Wang, S., Luo, X., Zhao, K., Jing, P., Ma, X., Tang, Y., and Zhou, H. (2025). Vehicular intrusion detection system for controller area network: A comprehensive survey and evaluation. *IEEE Transactions on Intelligent Transportation Systems*. DOI: 10.1109/TITS.2024.3468537.

Liu, Z., Ahmed, A., Thite, A., Naren□□□□□□, Kamhoua, C., and Ray, I. (2023). LLM-XAI: Large Language Models for Explaining ML Predictions in Cybersecurity. *arXiv preprint arXiv:2304.12345*.

Lundberg, H., Mowla, N. I., Abedin, S. F., Thar, K., Mahmood, A., Gidlund, M., and Raza, S. (2022). Experimental analysis of trustworthy in-vehicle intrusion detection system using explainable artificial intelligence (xai). *IEEE Access*, 10:102831–102841. DOI: 10.1109/ACCESS.2022.3208573.

Lundberg, S. M. and Lee, S.-I. (2017). A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30. DOI: 10.5555/3295222.3295230.

Lundberg, S. M. and Lee, S.-I. (2019). Explainable ai for trees: From local explanations to global understanding.

Marchetti, M. and Stabili, D. (2017). Evaluation of anomaly detection for in-vehicle networks through information-theoretic algorithms. *IEEE Transactions on Vehicular Technology*, 66(7):5860–5876. DOI: 10.1109/TVT.2017.2689320.

Martens, D., Hinns, J., Dams, C., Vergouwen, M., and Evgeniou, T. (2025). Tell me a story! narrative-driven xai with large language models. *Deci-*

sion Support Systems, 191:114402. DOI: https://doi.org/10.1016/j.dss.2025.114402.

Mavrepis, P., Makridis, G., Fatouros, G., Koukos, V., Separdani, M. M., and Kyriazis, D. (2024). XAI for all: Can large language models simplify explainable ai?

Metwaly, A. A. and Elhenawy, I. (2023). Sustainable Intrusion Detection in Vehicular Controller Area Networks using Machine Intelligence Paradigm . *Sustainable Machine Intelligence Journal*, 4:(4):1–12. DOI: 10.61185/SMIJ.2023.44104.

Mumuni, F. and Mumuni, A. (2025). Explainable artificial intelligence (xai): from inherent explainability to large language models.

Reyes, A. A., Vaca, F. D., Castro Aguayo, G. A., Niyaz, Q., and Devabhaktuni, V. (2020). A machine learning based two-stage wi-fi network intrusion detection system. *Electronics*, 9(10):1689. DOI: 10.3390/electronics9101689.

Seo, E., Song, H. M., and Kim, H. K. (2018). Gids: Gan based intrusion detection system for in-vehicle network. In *2018 16th Annual Conference on Privacy, Security and Trust (PST)*, pages 1–6. DOI: 10.1109/PST.2018.8514157.

Sharmin, S., Mansor, H., Abdul Kadir, A. F., and Aziz, N. A. (2024). Benchmarking frameworks and comparative studies of controller area network (can) intrusion detection systems: A review. *Journal of Computer Security*, 32(5):477–507. DOI: 10.3233/JCS-230058.

Sreelekshmi, M. S. and Aji, S. (2025). Real-time intrusion detection in controller area networks: An evaluation of current methods and future directions. *International Journal of Scientific Research in Network Security and Communication*, 13(2):22–34. DOI: 10.26438/ijsrnsc.v13i2.266.

Van Herrewege, A., Singelee, D., and Verbauwhede, I. (2018). A survey of automotive can bus security: Challenges and solutions. *Design Automation for Embedded Systems*, 23:1–24. DOI: 10.1007/s10617-018-9216-4.

Villain, J., Deniau, V., and Gransart, C. (2024). Vehicular wireless communications: Risks and detection of attacks. In Soliman, A.-H. and Ahmed, M. E.-S., editors, *Cooperative Intelligent Transport Systems: Control and Management*, pages 321–344. John Wiley & Sons. DOI: 10.1002/9781394178556.ch13.

Wang, S., Wang, Y., Zheng, B., Cheng, J., Su, Y., and Dai, Y. (2024). Intrusion detection system for vehicular networks based on mobilenetv3. *IEEE Access*. DOI: 10.1109/ACCESS.2024.3437416.

Wickramasinghe, C. S., Marino, D. L., Mavikumbure, H. S., Cobilean, V., Pennington, T. D., Varghese, B. J., Rieger, C., and Manic, M. (2023). Rx-ads: Interpretable anomaly detection using adversarial ml for electric vehicle can data. *IEEE Transactions on Intelligent Transportation Systems*, 24(12):14051–14063. DOI: 10.1109/TITS.2023.3294349.

Wohlin, C., Runeson, P., Höst, M., Ohlsson, M. C., Regnell, B., and Wesslén, A. (2012). *Experimentation in software engineering*. Springer Science & Business Media.

Woo, S., Jo, H., and Kim, D. H. (2014). A practical wireless attack on the connected car and security analysis. In *2014 IEEE 6th International Conference on Software Security and Reliability*, pages 76–85. IEEE. DOI: 10.1109/SERE.2014.17.

Zeng, X. (2024). Enhancing the interpretability of SHAP values using large language models.

Zhang, C., Chen, W., Liu, Y., Xu, Z., and Wang, H. (2022). Explainable ai for cyber security: State of the art and challenges. *IEEE Access*, 10:20738–20761. DOI: 10.1109/ACCESS.2022.3148456.

Zytek, A., Pidò, S., and Veeramachaneni, K. (2024). LLMs for XAI: Future directions for explaining explanations.

A Appendix - Implementation

The XAI Eva Insights[2] is a tool designed to enhance interpretability in IDS for vehicular networks. With an emphasis on transparency and the interpretability of ML model decisions, XAI Eva Insights integrates XAI techniques with LLMs to convert complex outputs into actionable and comprehensible insights. The application was developed using the technologies and versions listed in Table 9.

Table 9. Technologies and Versions of the XAI Eva Insights System

	Technology/Library	Version
Languages	Python	3.11.0rc1
	HTML	5
	CSS	3
	JavaScript	ES6+
LLMs	Groq	0.28.0
	OpenAI	1.86.0
	DeepSeek	API
	Gemini	API
Model	XGBoost	3.0.2
XAI	SHAP	0.48.0
Python Libraries	Flask	3.1.1
	Pandas	2.3.0
	Matplotlib	3.9.4
	Scikit-learn	1.7.0
	Werkzeug	3.1.3
	NumPy	2.2.6
	Requests	2.32.3

Next, we detail the system's architecture, key features, and technological implementation, supported by user interface illustrations throughout this section.

A.1 Architecture and Workflow

The XAI Eva Insights operates through a modular architecture that orchestrates data processing, intrusion detection, interpretability, and semantic translation. The workflow begins with the ingestion of network traffic data, which undergoes preprocessing to ensure consistency and analytical readiness. ML models, specifically based on XGBoost, are employed to classify intrusion attempts. The resulting predictions are passed to an XAI module, which leverages SHAP to quantify the influence of each feature. Finally, a semantic interpretation module—powered by LLMs translates these technical explanations into natural language narratives.

As illustrated in Figure 8, the architecture comprises the following components:

- **Wi-Fi and CAN Data Sources:** Responsible for collecting raw traffic data from vehicular networks.
- **Preprocessing:** Performs data cleaning, normalization, and feature engineering to prepare the data for analysis.

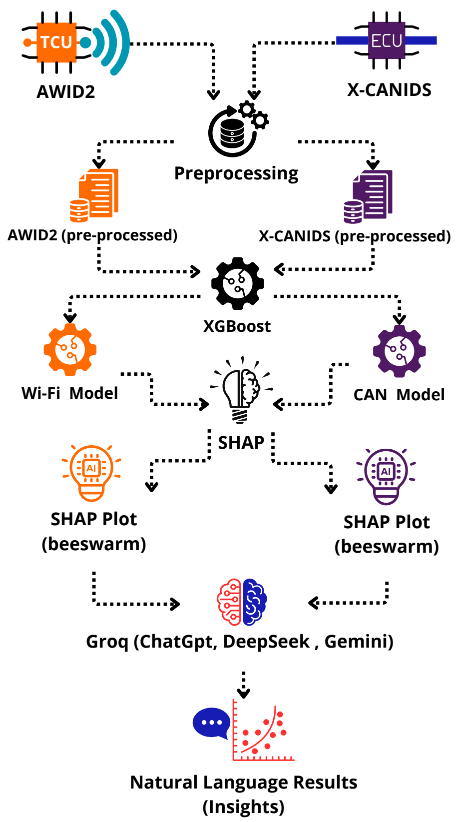

Figure 8. Overview of the XAI Eva Insights Architecture: Components and Data Flow.

- **XGBoost:** The system's core classifier, responsible for detecting intrusion patterns within Wi-Fi and CAN traffic datasets.
- **Wi-Fi and CAN Models:** XGBoost models specifically trained for each network type to optimize detection performance in distinct vehicular contexts.
- **XAI (SHAP):** The explainability module applies the SHAP algorithm to produce local explanations of the model's predictions, highlighting the contribution of each input feature.
- **Explainable Visualization:** Presents SHAP outputs through intuitive graphical representations, facilitating user understanding of feature relevance.
- **Generative AI:** A semantic layer that uses LLMs to transform SHAP's technical insights into descriptive natural language summaries.
- **Natural Language Results:** The final system output, offering clear, human-readable explanations of each intrusion detection instance.

A.1.1 Dataset Processing and Validation

After the user uploads a file, the system validates the dataset to ensure it is suitable for explainable analysis. The `validate_dataset` function performs checks for the presence of numerical columns and verifies whether the last column can serve as the target variable:

```
def validate_dataset(df):
    errors = []
    if df.empty:
        errors.append("The dataset is empty")
    if len(df.columns) < 2:
```

[2]`https://github.com/AI-Horizon-Labs/XAI-Eva-Insights`. The source code will be made available upon acceptance of the manuscript.

```
        errors.append("The dataset must have at
            least 2 columns")
    numeric_cols = df.select_dtypes(include=["
        float64", "int64"]).columns
    if len(numeric_cols) == 0:
        errors.append("The dataset must contain
            at least one numerical column")
    return errors
```

Listing 1: Dataset Validation Before SHAP Analysis

A.1.2 Model Training with XGBoost

Once the dataset passes validation, the system automatically determines whether the classification problem is binary or multi-class. Based on this detection, it trains an appropriate XGBoost model. The following code snippet illustrates this logic in the implementation:

```
if unique_classes == 2:
    model = xgb.XGBClassifier(random_state=42,
        eval_metric="logloss")
else:
    model = xgb.XGBClassifier(random_state=42,
        eval_metric="mlogloss")

model.fit(X, y)
```

Listing 2: Model Training with XGBoost

A.1.3 SHAP Value Generation and Visualization

After training the model, SHAP is employed to compute feature importances. A beeswarm plot is then generated to visually represent the impact of each feature on the model's predictions in aggregate:

```
explainer = shap.Explainer(model)
shap_values = explainer(X)

plt.figure(figsize=(12, 8))
shap.plots.beeswarm(shap_values, show=False,
    max_display=15)
plt.title("SHAP Feature Importance", fontsize=16)
plt.savefig(image_path, dpi=300)
```

Listing 3: SHAP Value Calculation and Plot Generation

The plot is automatically saved and rendered within the application interface, offering users a visual explanation of the model's behavior.

A.1.4 Explanation Generation with LLMs

The system enables users to select a language model for generating explanations based on the key features identified by SHAP. The following snippet illustrates the conditional execution depending on the chosen model:

```
if chosen_model == "openai":
    return call_openai(data)
elif chosen_model == "gemini":
    return call_gemini(data)

# ... other options
```

Listing 4: Conditional Call to the Selected LLM

To generate the explanation, a prompt is automatically constructed with summary information about the dataset and its most influential features:

```
prompt = f"""
DATASET INFORMATION:

Samples: {n_samples}
Features: {n_features}

TOP FEATURES:
{feature_1}: {importance_1}
...

Generate 3 practical insights based on this.
"""
```

Listing 5: LLM Prompt Construction

A.1.5 Web Interface and Results Visualization

Finally, the SHAP plot and the generated natural language explanations are dynamically rendered in a web interface. This is handled via an HTML template, as demonstrated below:

```
return render_template("analysis.html",
                       shap_image=results["
                           shap_image"],
                       insights=results["insights
                           "])
```

Listing 6: Rendering Results in the Web Interface

This interface provides an intuitive way for users to explore and interpret model results, even without advanced expertise in ML or explainability techniques.

A.2 User Interface and Functionalities

The XAI Eva Insights interface was designed to be intuitive and efficient, guiding users from dataset upload to the visualization of interpretable insights. The user experience is organized into clear and sequential steps, facilitating interaction with complex ML and XAI processes.

A.2.1 Home Screen

The home screen, shown in Figure 9, serves as the system's entry point. It provides a concise overview of the XAI Eva Insights value proposition, emphasizing its ability to convert raw data into actionable insights through explainable ML analysis. The *Start Analysis* button initiates the workflow, while the icons at the bottom highlight the platform's core functionalities: CSV upload, SHAP visualizations, and the generation of natural language insights.

A.2.2 Upload and Model Selection

After initiating the analysis, users are directed to the upload screen (Figure 10), which allows for the submission of datasets in CSV format. This data serves as the foundation for the explainability workflow.

A key feature of this screen is the ability to select the language model (LLM) that will be used to generate the textual explanations. The system provides flexibility by offering multiple LLM options—each accompanied by a brief description

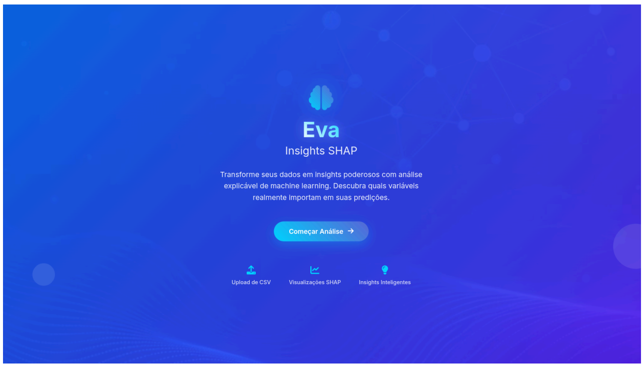

Figure 9. Home screen of XAI Eva Insights.

of its characteristics. Examples include: *Fast and efficient*, *Detailed analysis*, *Advanced insights*, and *Focused on code and logic*. This helps users select the most appropriate model according to their objectives and preferences.

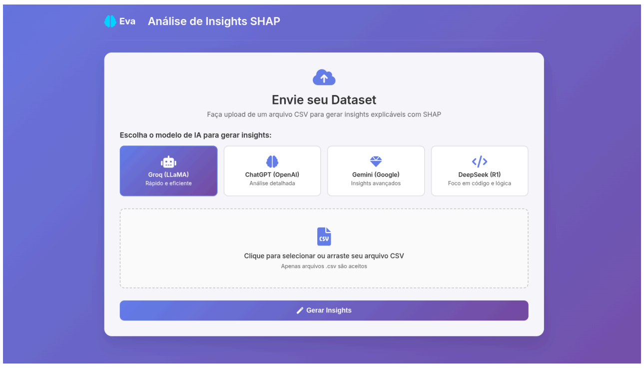

Figure 10. Dataset upload and AI model selection interface.

A.2.3 SHAP Visualization

Once the dataset is processed and the ML models are applied, XAI Eva Insights generates SHAP based visualizations to enhance the interpretability of the predictions. Figure 11 presents a SHAP feature importance plot (beeswarm plot), which helps users understand how each feature in the dataset contributes to the model's decision-making process.

In this plot, each point represents an individual data instance. The horizontal position of each point corresponds to the SHAP value, indicating the magnitude and direction of that feature's impact on the model's output. This visualization enables rapid identification of the most influential features, offering a comprehensive view of the internal logic behind the model's predictions.

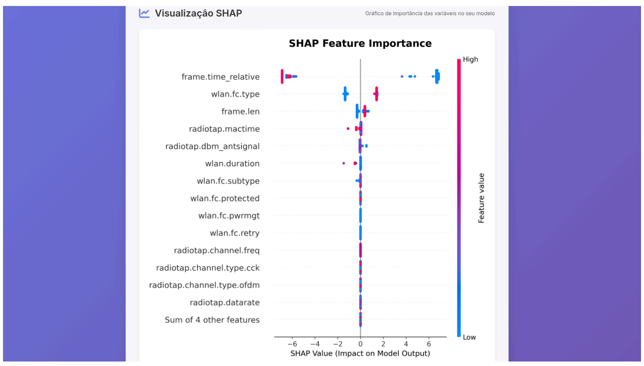

Figure 11. Visualization of SHAP Feature Importance.

A.2.4 Natural Language Generated Insights

A key innovation of XAI Eva Insights lies in its ability to convert complex SHAP outputs into understandable and actionable natural language insights. As shown in Figure 12, the system presents explanations that go beyond numerical values delivering narrative summaries of model behavior.

These insights describe attacker behavior patterns, identify the most influential features for detection, and suggest practical implications for vehicular security. For instance, one explanation may emphasize the importance of a temporal feature, indicating that the sequence of network events is critical for identifying an intrusion. Another may highlight the relevance of a specific Wi-Fi frame type, suggesting the need for further investigation or refined feature engineering.

This functionality significantly lowers the barrier to understanding complex ML models, making interpretability accessible to both domain specialists and non-experts.

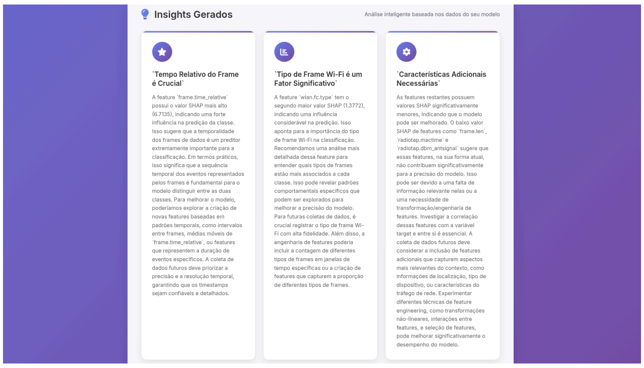

Figure 12. Natural Language Insights Generated by XAI Eva Insights.

A.3 Technologies and Implementation

XAI Eva Insights is built upon a robust and scalable technological stack, with the backend primarily implemented in Python using the Flask framework. This combination supports agile development, ease of deployment, and seamless integration with ML and XAI libraries.

Table 9 outlines the main technologies and their respective versions used in the system.

A.3.1 Backend

The system's backend is implemented in Python, utilizing the Flask microframework. This choice enables a lightweight and flexible structure for API development and HTTP request handling. The `pandas` and `numpy` libraries are employed for efficient data manipulation and preprocessing, which are essential for the data cleaning and feature engineering stages. The `xgboost` library serves as the core component for training and inference of intrusion detection models, while `shap` is responsible for generating explainable AI outputs. Integration with LLMs is achieved via the APIs of `groq`, `openai`, `gemini`, and `deepseek`, allowing the system to flexibly choose the most appropriate model for semantic interpretation of model outputs.

A.3.2 Frontend

The frontend of XAI Eva Insights is developed using standard web technologies: HTML for content structure, CSS for visual styling and layout, and JavaScript for dynamic interactivity. The combination of these technologies results in a responsive, modern, and user-friendly interface that facilitates seamless interaction—from uploading datasets to visualizing SHAP-based explanations and natural language insights.

A.4 Development and Testing Environment

The development and testing of the XAI Eva Insights system were carried out in a high-performance environment to ensure stability, responsiveness, and the computational capability required for processing large datasets and executing ML models under load.

The primary development machine was an Alienware x17 R2 notebook, with the following specifications:

- **Operating System:** Zorin OS 17.3 Core (64-bit)
- **Processor:** 12th Gen Intel® Core™ i9-12900HK (20 threads)
- **RAM:** 64 GB
- **Graphics Card:** NVIDIA GeForce RTX 3080 Ti Laptop GPU / PCIe / SSE2 and Mesa Intel® Graphics (ADL GT2)
- **Storage:** 4 TB

The codebase was developed primarily using the following software tools:

- **Visual Studio Code (VS Code):** Version 1.101.2
- **Sublime Text:** Version 3.2.2

Project dependencies were managed using `pip`, the standard Python package manager. For local testing and execution, the system was run directly via the terminal using the command `python3 app.py`. All command-line operations were conducted through the default terminal of Zorin OS. The web interface was tested and validated using Google Chrome, version 138.0.7204.49 (64-bit), ensuring compatibility and a consistent user experience in a widely adopted browsing environment across platforms.

A.5 Conclusion

XAI Eva Insights represents a meaningful advancement in the application of XAI to vehicular cybersecurity. By integrating the predictive power of ML models with SHAP-based interpretability and the natural language generation capabilities of LLMs, the system delivers a comprehensive solution for understanding and mitigating cyber threats in automotive environments at scale.

Its intuitive interface and the ability to produce clear, actionable explanations even for non-specialists are crucial for fostering trust in AI-based security systems and for enabling informed decision-making in critical contexts. Beyond enhancing intrusion detection capabilities, the tool contributes to the democratization of explainability, providing accessible insights into the reasoning behind detections a foundational step toward the broader adoption and continual refinement of security solutions for connected vehicles.

A.6 Future Work

As a direction for future work, we propose a comprehensive evaluation of the outputs generated by the different LLMs integrated into XAI Eva Insights. The goal is to assess which language model delivers the most effective and accurate insights across various vehicular intrusion scenarios. As the primary direction for future research, we propose the development of a formal framework to quantify the semantic fidelity of LLM-generated explanations. This involves measuring the alignment between the model's textual narratives and the underlying mathematical feature attributions provided by SHAP, ensuring that no information loss or hallucination occurs during the translation process.

Additionally, we plan to conduct human-in-the-loop user studies with security analysts to evaluate the practical utility and cognitive load of our natural language insights in real-world incident response scenarios.

Finally, future iterations will explore the integration of multimodal XAI, combining interactive visualizations with real-time audio alerts for in-vehicle diagnostic systems.