Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Few-Shot Learning for Adaptive Sybil Attack Detection in the Internet of Vehicles

Rohit Kumar

profroot@iitkgp.ac.in

Indian Institute of Technology Kharagpur

Chittaranjan Mandal

Indian Institute of Technology Kharagpur

Arun Raj Kumar P

National Institute of Technology Calicut

Research Article

Keywords:

Posted Date: November 10th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11313033/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Few-Shot Learning for Adaptive Sybil Attack Detection in the Internet of Vehicles

Rohit Kumar[a,b], Chittaranjan Mandal[a], Arun Raj Kumar P[c]

[a] *Department of computer Science and Engineering , IIT Kharagpur, Kharagpur, INDIA*
[b] *Department of computer Science and Engineering , CMR University, Bengaluru, INDIA*
[c] *Department of computer Science and Engineering , NIT Calicut, , INDIA*

Abstract

The Internet of Vehicles (IoV) enables safety-critical cooperative driving applications through continuous exchange of Basic Safety Messages (BSMs). However, these networks remain vulnerable to Sybil attacks, in which a single malicious node fabricates multiple identities to disrupt traffic information, manipulate voting protocols, and create phantom congestion. Existing machine learning defences rely on large labelled datasets and must be fully retrained whenever attackers develop new variants—a response time incompatible with the dynamic nature of vehicular environments. This paper presents FSL-Sybil, a Prototypical Network-based few-shot learning framework that detects novel Sybil attack variants with as few as two labelled samples, without retraining the underlying model. We construct a realistic simulation environment using OMNeT++ 6.1, the veins_inet framework, and SUMO 1.20.0 on a real-world road network of Indiranagar, Bengaluru (India), and implement three distinct Sybil attack types: random position fabrication (SA-RND), trajectory fabrication (SA-TRJ), and coordinated sophisticated attacks (SA-CRD). We extract 20 discriminative features across BSM, mobility, and network-layer domains and evaluate the proposed approach against Random Forest, SVM, and k-NN baselines across 10 scenarios (65,087 samples total). Under a novel attack adaptation scenario in which SA-CRD attacks are withheld from training, our method achieves 99.88% detection recall with only one support sample per class, while the naive baseline Random Forest recall drops to 82.23%. Computational profiling on CPU hardware demonstrates that the proposed model occupies only 70 KB and sustains 1.4 million inference samples per second—1.7× faster than Random Forest and 26× faster than k-NN—making it suitable for deployment on resource-constrained Road Side Units. We've also identified a critical vulnerability: while RSSI-based features are highly discriminative under ideal conditions they represent a single point of failure in adaptive spoofing attacks.

Keywords: Internet of Vehicles, Sybil attack detection, Few-shot learning, Prototypical Networks, VANET Security, Edge computing for IoV, OMNeT++ Simulations, SUMO Simulations, veins_inet

Email addresses: profroot@iitkgp.ac.in (Rohit Kumar), chitta@iitkgp.ac.in (Chittaranjan Mandal), park@nitc.ac.in (Arun Raj Kumar P)

1. Introduction

The Internet of Vehicles (IoV) transforms transportation infrastructure by enabling vehicles, Road Side Units (RSUs) and cloud services to exchange information in real time. Contreras-Castillo et al. [1] describe the IoV as a layered architecture in which vehicle-to-vehicle, vehicle-to-infrastructure and vehicle-to-cloud links jointly support cooperative driving, traffic optimisation and safety-critical decision-making. An overview of this architecture is shown in Fig. 1.

One of the most important components of this ecosystem is the continuous broadcast of Basic Safety Messages (BSMs). The medium-access and channel-coordination rules for these broadcasts are defined by the IEEE 1609 WAVE family of standards [2], while the content and encoding of each BSM are specified in the SAE J2735 message set dictionary [3]. Kenney [4] reviews how these standards combine in the Dedicated Short-Range Communications (DSRC) stack, in which BSMs are transmitted at 10 Hz over the 5.9 GHz IEEE 802.11p channel and announce a vehicle's position, speed, heading, acceleration and other kinematic state. Although BSMs are crucial for collision avoidance and trajectory prediction, their unauthenticated semantic content also makes the network vulnerable to identity-based attacks. Our own earlier comparative study of generative deep learning models for malicious node detection in the IoV [5] showed that detectors trained on synthetic attack data transfer poorly to new attack behaviour, which motivates the present work.

The security requirements of vehicular networks have been surveyed extensively. Raya and Hubaux [6] provided one of the first systematic threat analyses of vehicular ad hoc networks (VANETs) and proposed a public-key infrastructure with pseudonymous certificates. Engoulou et al. [7] classified VANET attacks by the security service they violate, and Mejri et al. [8] examined the cryptographic solutions available for each class. Hasrouny et al. [9] later extended these surveys and grouped Sybil defences into cryptography-based authentication, location verification and anomaly analysis. Krishnan and Kumar [10] further highlighted the tension between security and privacy in VANETs, since strong identity binding conflicts with the pseudonymity that drivers expect.

Among these threats, the Sybil attack is particularly damaging. Douceur [11] first formalised the attack in peer-to-peer systems and proved that, without a logically centralised authority, a single entity can always present multiple identities. Newsome et al. [12] subsequently showed how the attack undermines voting, data aggregation and routing in wireless sensor networks. In the IoV, a single malicious vehicle (insider or outsider) generates multiple fake identities and broadcasts spurious BSMs as if each identity corresponded to a separate physical vehicle. This immediately injects false information into the local traffic view of nearby vehicles. An attacker could create phantom congestion to reroute emergency services such as ambulances, manipulate voting-based cooperative protocols, or disrupt intersection management. Given the safety-critical nature of decisions made from BSMs, robust and low-latency Sybil detection is essential for any realistic IoV deployment.

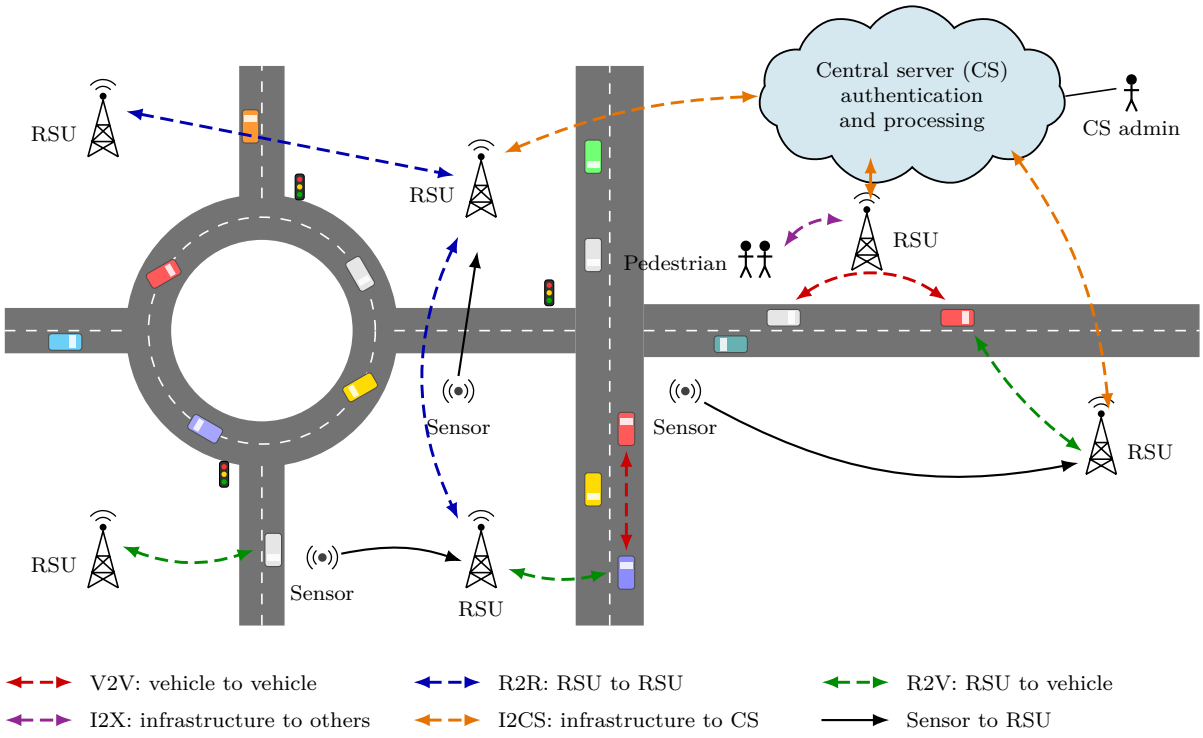


Figure 1: Schematic diagram of the Internet of Vehicles network architecture. RSU: roadside unit; CS: central server.

1.1. Evolution of Sybil Detection in Vehicular Networks

Cryptography-based approaches. Lu et al. [13] proposed ECPP, an efficient conditional privacy preservation protocol in which RSUs issue short-lived anonymous keys that can be traced by a trusted authority if misuse is detected. Zhou et al. [14] proposed P2DAP, in which roadside boxes passively overhear the pseudonyms that sign each reported event, flag events signed by pseudonyms sharing the same coarse-grained hash value, and forward them to a trusted authority that confirms the attacker with a secret fine-grained hash; its evaluation reports communication overhead and detection latency (about 30 s in the adaptive variant) rather than detection accuracy. Park et al. [15] addressed the initial deployment stage, when few RSUs are available, by using RSU-issued timestamps to identify vehicles that present similar motion trajectories. These schemes provide strong identity assurance, but they depend on a trusted infrastructure and add computational and certificate-management overhead to every message.

Position verification approaches. Xiao et al. [16] were among the first to verify claimed positions against the signal strength measured by witnesses in the opposite traffic flow, using hypothesis tests over an observation period; in simulation they reported a detection rate of at least 95% with a false positive rate of at most 5% when more than five witnesses were available. Yu et al. [17] extended this scheme with a RANSAC-based position estimator that is robust to fabricated witness readings and a Presence Evidence System based on roadside certificates, and reported a detection rate above 99% for position deviations of at least 10 m

in NS2 and GrooveNet simulations on a Detroit road map. Grover et al. [18] proposed a fully distributed alternative that needs no roadside infrastructure: vehicles periodically exchange and intersect their neighbour lists, and identities that share the same neighbours for longer than a time threshold are flagged, giving a false positive rate of about 0.1% with an 80 s threshold. Position verification is effective under ideal conditions, but it relies on honest witnesses in opposite-direction traffic and on roadside infrastructure, and it cannot separate Sybil identities claimed within about 10 m of the attacker. Krishnan and Kumar [19] proposed a collaborative strategy in which RSUs, vehicles and a central server combine received-signal-strength and Euclidean-distance checks with a reward system to detect and evict both the Sybil attacker and its fabricated nodes, reporting a detection rate of up to 99.89% in NS2 and SUMO simulations.

Behavioural and machine learning approaches. Golle et al. [20] introduced the idea of checking received data for consistency against a model of the physical world, which underlies most behavioural detectors. Van der Heijden et al. [21] released the VeReMi dataset to enable comparable evaluation of such misbehaviour detectors, and Kamel et al. [22] extended it with additional attack types, including Sybil-style ghost-vehicle attacks. So et al. [23] used plausibility checks as the feature vector of k-NN and SVM classifiers on VeReMi and improved detection precision by more than 20% over the plausibility checks alone, while Gyawali and Qian [24] proposed a cooperative machine-learning misbehaviour detection system that verifies false alerts with the Greenshields traffic-flow model and detects position-falsification attacks on VeReMi, where a bagging classifier reached a precision of 0.98 and a recall of 0.94. Azam et al. [25] combined k-nearest neighbours, naive Bayes, decision tree, SVM and logistic regression classifiers through majority voting and reported 95% detection accuracy, and Gu et al. [26] represented each vehicle by the two largest eigenvalues of its driving-pattern matrix and classified Sybil nodes with an RBF-kernel SVM, reaching a true positive rate of 97.6% at a false positive rate of 2% in near-capacity urban traffic. These methods are attractive because they rely only on observable traffic and need no additional hardware. However, they share two limitations. First, they need large volumes of labelled attack data to reach high accuracy, and such data are rarely available for real Sybil attackers; most datasets are simulated or synthetically generated. Second, whenever attackers change tactics, the model has to be retrained from scratch, which is time-consuming and again requires a sufficient number of labelled examples of the new variant. These constraints are particularly severe in the IoV, where deployments are distributed, infrastructure is resource-limited and response times are critical.

Few-shot learning. Few-shot learning (FSL) addresses exactly this setting by training models that generalise from a handful of labelled examples. Wang et al. [27] survey the field and attribute the difficulty of FSL to an unreliable empirical risk minimiser, and Hospedales et al. [28] frame FSL within the broader meta-learning paradigm. Koch et al. [29] learned a pairwise similarity function with Siamese networks, Vinyals et al. [30] introduced Matching Networks that attend over a support set, and Finn et al. [31] proposed MAML, which learns an initialisation that adapts in a few gradient steps. Sung et al. [32] replaced the fixed

distance with a learned relation module. Prototypical Networks [33] learn class prototypes, i.e., centroids in an embedding space, and compare new queries to them using Euclidean distance; they are simple, stable to train and particularly effective in structured, low-dimensional feature spaces. Contrastive representation learning [34] offers a complementary route to label-efficient embeddings.

FSL has recently been applied to network security. Yu and Bian [35] trained an intrusion detector on less than 1% of the NSL-KDD training set and still outperformed classical classifiers trained on far more data. Xu et al. [36] formulated few-shot intrusion detection as a meta-learning problem over pairs of benign and malicious traffic, and Yang et al. [37] proposed the FS-IDS framework for detecting rare attack classes. Bilot et al. [38] combined a self-supervised graph neural network with a few-shot decoder and showed that a single labelled attack per attack family suffices to reach a macro F1-score of 97.44% on NF-CSE-CIC-IDS2018-v2, exceeding fully supervised graph baselines. A recent PRISMA-guided review by Roszeitis et al. [39] retained 21 few-shot intrusion detection studies and found that meta-learning and convolutional networks dominate, while heterogeneous evaluation settings hinder comparison.

FSL has also started to reach vehicular security. Dina et al. [40] combined self-supervised pre-training, a Prototypical Network and transfer learning in FSProtoTransfer, and with only 20% of the labelled data outperformed fully supervised models on two VANET intrusion datasets by 41%, 78% and 70% in precision, recall and F1-score. Shafiq et al. [41] coordinated five specialised few-shot agents in MAFSID and evaluated them on NSL-KDD, CICIDS2017 and in-vehicle data. Khan et al. [42] meta-trained a topology-aware encoder with a Reptile-style loop on the VeReMi dataset and fine-tuned small adapter modules to new driving environments in under one second, reporting an average AUC of 0.70 and an F1-score of 0.62. Xu et al. [43] addressed zero-day attacks in the IoV by augmenting the few available attack samples with a few-shot conditional generative adversarial network evaluated on the F2MD misbehaviour-detection platform. For the in-vehicle Controller Area Network (CAN), Ahmad et al. [44] proposed a stochastic Prototypical Network that reached 99.17% accuracy in a 15-shot configuration, Duy et al. [45] obtained an F1-score of 0.9994 on the Car Hacking dataset with 2% of the training data, and Ben Atia et al. [46] combined MAML with federated learning to adapt an LSTM detector to previously unseen attacks. Datta et al. [47] embedded a lightweight FSL module in an IoV authentication protocol to classify traffic conditions from few labelled samples.

The two few-shot studies closest to Sybil detection address attacks that falsify positions or safety messages. Vasan et al. [48] proposed an Uncertainty-Aware Prototypical Network that detects novel GNSS spoofing and jamming patterns in intelligent transportation systems from one to three labelled examples, without any training or adaptation at deployment, and reached 80.6% accuracy with one example and 88.3% with three on GPS, BeiDou and GLONASS data. It is the prior work closest to ours in method, since it also classifies unseen attack patterns by their distance to class prototypes without retraining, but it targets signal-level GNSS attacks rather than fabricated vehicle identities in BSMs. Gyawali et al. [49] converted

BSM kinematic and peer-report sequences into text and used few-shot prompting and LoRA fine-tuning of large language models to separate benign sensor faults from cyber-attacks on the Tampa connected-vehicle pilot data. These studies show that FSL is viable in vehicular settings, but they target generic intrusion, misbehaviour, GNSS spoofing or in-vehicle CAN attacks and report no Sybil-specific detection results. FSL for Sybil detection under evolving attack behaviour therefore remains unexplored, which the systematic search in Section 1.2 confirms.

1.2. Systematic Literature Review (PRISMA)

To position this work objectively, we conducted a systematic literature search following the PRISMA 2020 reporting guideline [50]. Scopus was searched on 28 September 2026 and IEEE Xplore on 30 September 2026 with the search strings listed in Table 1. Both strings combine three concept blocks joined by AND: Sybil terms (Sybil, fake identity, ghost vehicle, identity spoofing), vehicular-network terms (VANET, vehicular ad hoc network, Internet of Vehicles, IoV, V2X, vehicular network, vehicular communication, connected vehicle) and detection terms (detect*, identif*, classif*). Scopus was searched in the title, abstract and keyword fields and IEEE Xplore in its All Metadata field; both searches were restricted to journal articles and conference papers published between 2010 and 2025, and the Scopus search additionally to English. Fig. 2 summarises the identification, screening and inclusion process. Of the 793 records identified (318 from Scopus and 475 from IEEE Xplore), 162 duplicates were removed by matching the DOI or the normalised title and publication year, and 631 records were screened on title and abstract. At this stage, 443 records were excluded because Sybil identities were not addressed (215) or were mentioned only in passing (120), because they were reviews or surveys (55), because they were not set in vehicular networks (40), because they proposed only authentication or cryptographic prevention without a detection mechanism (9), or because they analysed the attack without proposing a detector (4). All 188 remaining reports were retrieved and assessed for eligibility against three inclusion criteria: the work is peer-reviewed, it proposes or evaluates a Sybil or ghost-vehicle detection method for VANET/IoV, and it reports an evaluation of detection performance. Eligibility was judged from the abstract and bibliographic record of each report. A further 33 reports were excluded because they described a prevention or authentication scheme without evaluating detection (11), reported no detection-performance evaluation (20), or were not set in vehicular networks (2), leaving 155 studies in the review.

Grouped by their primary detection approach, the 155 included studies rely on machine or deep learning (55), signal-strength or position verification (32), cryptography, certificates or pseudonyms (23), mobility, trajectory or traffic-behaviour analysis (16), trust, reputation or blockchain mechanisms (14), or other techniques (15). The complete list, with the group assigned to each study, is given in Supplementary Material S1. Citing all 155 studies individually is neither practical nor informative, so Section 1.1 discusses representative studies from each group (the seminal and most-cited works together with recent ones), and

Table 7 compares them quantitatively. The review shows that detection research is increasingly data-driven, yet none of the 155 studies uses few-shot learning or meta-learning. A targeted search confirms this gap: adding few-shot, meta-learning, prototypical-network, one-shot and zero-shot terms to the Sybil and vehicular blocks returned no record in Scopus and four records in IEEE Xplore, none of which applies few-shot learning to Sybil detection. Two of the four address generic misbehaviour detection, one classifies traffic conditions within an authentication protocol [47], and one uses a stacked meta-learner ensemble.

Few-shot learning has, however, been applied to other attacks on vehicular networks. To capture this related work, we ran a supplementary search that combines the few-shot and meta-learning terms with the vehicular-network terms and general attack terms (intrusion, attack, Sybil, misbehaviour, anomaly, malicious, spoofing, threat), without an upper year limit. It returned 92 records in Scopus and 99 in IEEE Xplore (153 unique records), of which 27 apply few-shot, meta- or zero-shot learning to vehicular security. None of these records mentions Sybil attacks in its title or abstract, so none of them meets the inclusion criteria of the review. The vehicular few-shot studies discussed in Section 1.1, such as those of Dina et al. [40] and Khan et al. [42], come from this supplementary search. They address generic intrusion, misbehaviour or in-vehicle CAN attacks, which is why few-shot learning is established for vehicular intrusion detection but not, to date, for Sybil detection. The remaining works cited in this paper also lie outside the review's inclusion criteria and are not counted in Fig. 2: foundational Sybil and VANET-security studies published before 2010, the few-shot learning methods on which FSL-Sybil builds, and the datasets, standards and simulation tools used in this work.

Table 1: Search strings used for the systematic literature search (2010–2025; journal and conference papers).

Database	Search string	Records
Scopus	`TITLE-ABS-KEY ((sybil OR "fake identit*" OR "ghost vehicle*" OR "identity spoofing") AND (vanet* OR "vehicular ad hoc network*" OR "vehicular ad-hoc network*" OR "internet of vehicles" OR iov OR v2x OR "vehicular network*" OR "vehicular communication*" OR "connected vehicle*") AND (detect* OR identif* OR classif*)) AND PUBYEAR > 2009 AND PUBYEAR < 2026 AND (LIMIT-TO (DOCTYPE , "ar") OR LIMIT-TO (DOCTYPE , "cp")) AND (LIMIT-TO (LANGUAGE , "English"))`	318
IEEE Xplore	`("All Metadata":sybil OR "All Metadata":"fake identit*" OR "All Metadata":"ghost vehicle*" OR "All Metadata":"identity spoofing") AND ("All Metadata":vanet* OR "All Metadata":"vehicular ad hoc network*" OR "All Metadata":"internet of vehicles" OR "All Metadata":iov OR "All Metadata":v2x OR "All Metadata":"vehicular network*" OR "All Metadata":"vehicular communication*" OR "All Metadata":"connected vehicle*") AND ("All Metadata":detect* OR "All Metadata":identif* OR "All Metadata":classif*)`; filters: 2010–2025, Journals, Conferences	475

Fig. 3 plots the number of records identified per publication year in each database. Research on Sybil detection in vehicular networks grew slowly until 2018 and has accelerated since: Scopus indexed 2 such articles in 2010 and 70 in 2025, and IEEE Xplore 5 and 100, respectively, while the number of unique

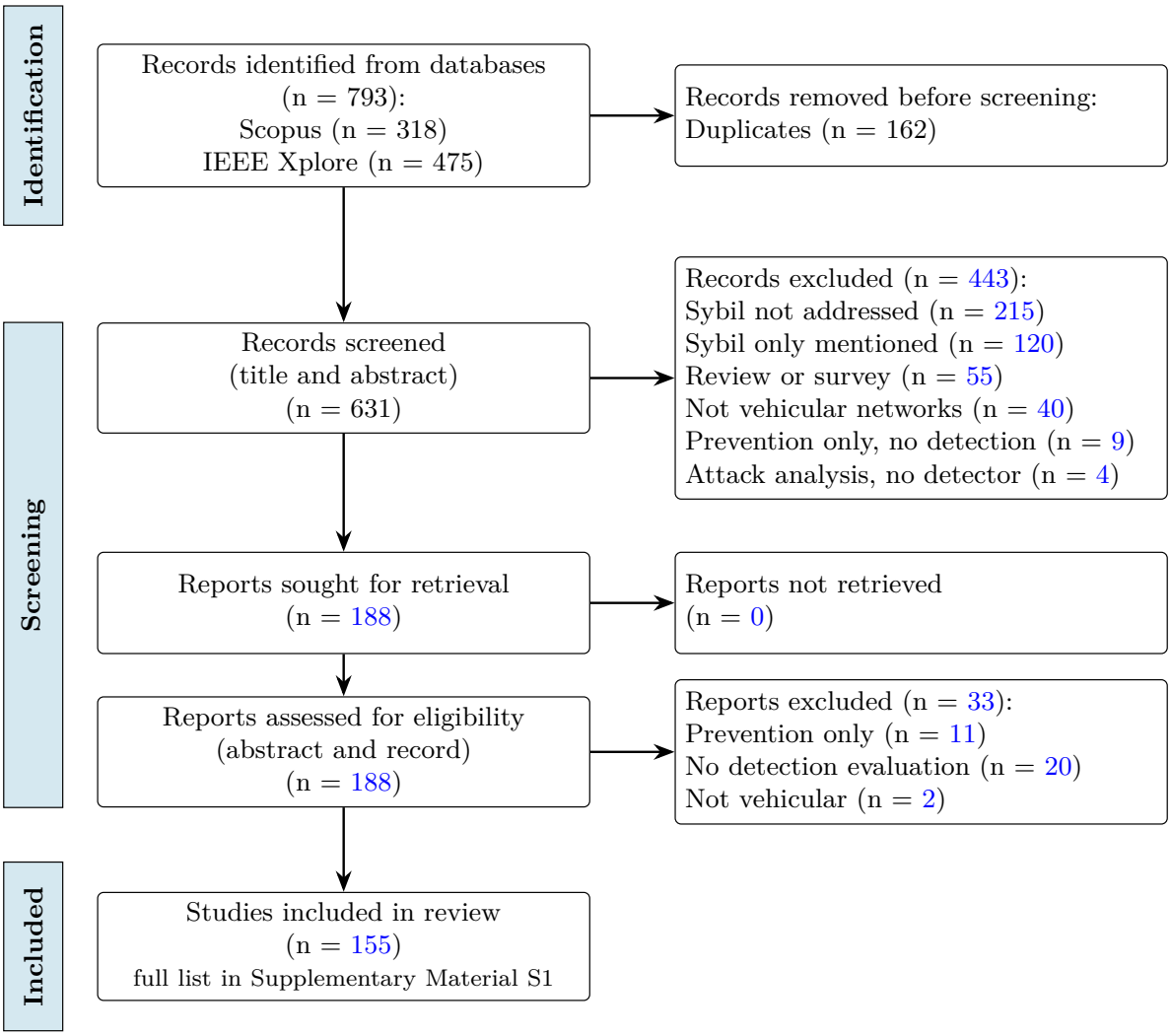


Figure 2: PRISMA 2020 flow diagram of the systematic literature search on Sybil detection in VANET/IoV (Scopus and IEEE Xplore, 2010–2025). Eligibility was assessed on the abstract and bibliographic record of each report. Blue counts are provisional until the screening decisions are confirmed by the authors.

records after de-duplication rose from 7 in 2010 to 128 in 2025. Machine-learning-based detection appears only from 2016 onwards but accounts for much of the recent growth, rising to 32 (Scopus) and 46 (IEEE Xplore) articles in 2025 alone. Few-shot and meta-learning approaches, in contrast, are absent from the Sybil-detection literature, as shown above. This trend confirms both the growing attention to the problem and the scarcity of label-efficient detection methods, which the present work addresses.

1.3. Research Gap and Novelty Justification

The review above exposes a clear gap. Existing machine-learning detectors, such as those of So et al. [23] and Gyawali and Qian [24], assume that every attack variant is well represented in the training data, and in

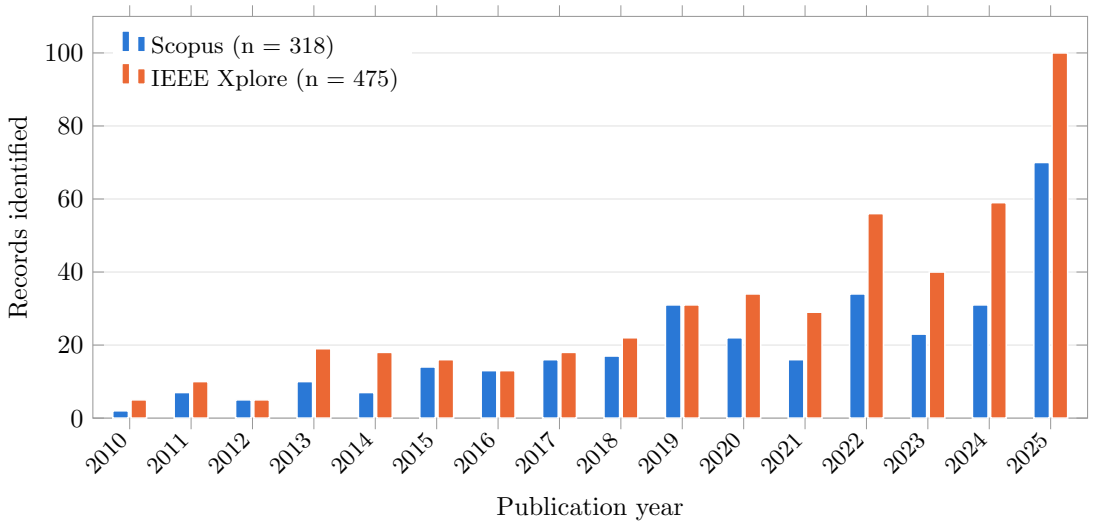


Figure 3: Number of records on Sybil detection in VANET/IoV identified per publication year in Scopus ($n = 318$) and IEEE Xplore ($n = 475$), before de-duplication (search strings in Table 1).

the systematic search (Section 1.2) we found no Sybil detector whose title or abstract reports how quickly a deployed model can adapt to a variant it has never seen. Few-shot methods that address this problem have been demonstrated for generic intrusion detection by Yu and Bian [35], Xu et al. [36] and Bilot et al. [38], and, more recently, for vehicular intrusion and misbehaviour detection by Dina et al. [40], Khan et al. [42] and Ahmad et al. [44], but not for BSM-level Sybil detection, where the feature space, the attack semantics and the deployment hardware are all different. This paper closes that gap with FSL-Sybil, a Prototypical Network-based detector that is trained on two common attack variants (random position fabrication and trajectory fabrication) and then adapts to a coordinated attack variant using as few as two labelled support samples (one per class; see the worked example in Section 6.2). Because adaptation only replaces the support set, no retraining is needed, and the model reaches 99.88% detection recall on the unseen variant. To the best of our knowledge, this is the first demonstration of retraining-free adaptation to an unseen Sybil variant in the IoV.

The novelty of the evaluation also lies in the data. Publicly available misbehaviour datasets such as VeReMi [21] and its extension [22] are built on European road layouts and do not separate Sybil variants by sophistication. We therefore construct a new benchmark on a real-world road network of Indiranagar, Bengaluru (India), with a vehicle mix that includes auto-rickshaws and buses, three increasingly sophisticated Sybil attack types and ten simulation scenarios, yielding 65,087 labelled BSM feature samples. This allows the difficulty of each attack variant to be controlled explicitly, which existing benchmarks do not permit.

Prior studies seldom report whether a detector can actually run on roadside or on-board hardware. We fill this gap with a CPU-only efficiency study, showing that FSL-Sybil occupies 70 KB and processes 1.4 million

samples per second, which is smaller and faster than the Random Forest, SVM and k-NN baselines. Finally, whereas position-verification schemes such as those of Xiao et al. [16] and Yu et al. [17] treat received signal strength as a trustworthy signal, our analysis shows that RSSI-based features dominate classification under ideal conditions but become a single point of failure under adaptive spoofing. We quantify this effect for every model and derive design guidance for detectors that remain robust when RSSI cannot be trusted.

1.4. Paper Organisation

The remainder of the paper is organised as follows. Section 2 defines the threat model and the three Sybil attack variants. Section 3 describes the reported baseline models and the proposed FSL-Sybil model. Section 4 presents the simulation environment, the Indiranagar road network and the feature extraction pipeline. Section 5 summarises the overall methodology with a flowchart and the experimental setup. Section 6 reports the simulated and experimental results, including a comparison with the state of the art. Section 7 discusses limitations and threats to validity, and Section 8 concludes the paper.

2. Sybil Attack Model

We formalise the Sybil attack in IoV and define three attack variants of increasing sophistication. Each variant captures a realistic capability of a practical adversary and motivates different defensive mechanisms.

2.1. Threat Model

We assume an adversary that controls one or more physical vehicles, each equipped with a standard IEEE 802.11p radio [2] and a GPS receiver. The adversary can fabricate multiple identities and inject BSMs into the network with arbitrary claimed position, speed, heading, and acceleration. The adversary does not compromise the cryptographic stack; we assume pseudonymous certificates are either present or orthogonal to our defence. The adversary knows the structure of the BSM and can schedule its transmission, but cannot change the position of its antenna. This means that no matter how many spoofed identities are created, all radio transmissions will be sent from the physical position of the attacker's device. Anomaly-based detection is based on a single fixed physical attribute.

2.2. Attack Variants

SA-RND (Random Position Fabrication). The simplest attack, in which each Sybil identity broadcasts BSMs with positions sampled uniformly within a $500\,\mathrm{m} \times 500\,\mathrm{m}$ region around the attacker, and speeds and headings drawn from broad ranges. Fake trajectories are temporally inconsistent: positions do not match claimed speeds over time, and speed–position consistency is violated. This variant is expected to be easy to detect from short-window mobility features.

SA-TRJ (Trajectory Fabrication). A more sophisticated attack in which each Sybil identity maintains a plausible but fabricated trajectory with physically realistic speeds, bounded acceleration, and rare but present heading changes. Positions are updated consistently with claimed kinematics. This variant defeats simple position-jump detectors but may still leave traces in neighbour stability, trajectory smoothness, and RSSI-versus-claimed-distance consistency.

SA-CRD (Coordinated and Sophisticated). The strongest attack, In this scenario, Sybil identities move in a coordinated formation around the real attacker whose offset radius smoothly varies over time. Each Sybil mirrors the attacker's speed heading and acceleration with small random perturbations. This variant maximises local behavioural plausibility and is most difficult to detect purely from kinematics.

Table 2 summarises the characteristics and expected detectability of the three variants.

Table 2: Characteristics and expected detectability of the three Sybil attack variants.

Attack	**Position Model**	**Temporal Consistency**	**Expected Difficulty**
SA-RND	Random within 500 m box	Violated	Easy (mobility features)
SA-TRJ	Plausible straight trajectory	Partially satisfied	Medium (needs RSSI or neighbour features)
SA-CRD	Formation around attacker	Satisfied	Hard (cross-feature reasoning)

3. Reported and Proposed Models

This section first describes the reported models, i.e., the supervised classifiers that have been widely used for Sybil and misbehaviour detection in the literature and that serve as baselines in this study. It then presents the proposed FSL-Sybil model.

3.1. Reported Models

Supervised classifiers operating on hand-crafted BSM features are the most frequently reported detectors for vehicular misbehaviour. So et al. [23] used SVM and k-nearest neighbours on top of plausibility checks, Gyawali and Qian [24] evaluated five supervised classifiers for position falsification, and Azam et al. [25] combined five of them in a majority-voting ensemble. We therefore re-implement the three most commonly reported model families on exactly the same 20-dimensional feature vectors (Section 4.4) and data splits as the proposed model, so that differences in performance can be attributed to the learning paradigm rather than to the features.

Random Forest (RF). Random Forest [51] is an ensemble of decision trees, each trained on a bootstrap sample of the data with random feature sub-sampling at every split; the final prediction is a majority vote. It handles heterogeneous features without scaling and is the strongest baseline in most misbehaviour-detection studies.

Support Vector Machine (SVM). The SVM [52] finds the maximum-margin hyperplane between classes in a kernel-induced feature space. We use a radial basis function (RBF) kernel with standardised inputs.

k-Nearest Neighbours (k-NN). The k-NN classifier [53] assigns a query the majority label of its k closest training samples. We use $k = 5$ with Euclidean distance on standardised inputs. Because k-NN stores the entire training set, its memory footprint grows with the amount of labelled data.

All reported models are implemented in scikit-learn with fixed hyperparameters that are identical in every experiment; they were not tuned for individual scenarios. The RF uses 100 trees with a maximum depth of 15 and at least two samples per leaf. The SVM uses an RBF kernel with $C = 1.0$ and γ set to `scale`, i.e. $1/(d\,\sigma_X^2)$ for $d = 20$ features with variance σ_X^2, and, because its training cost grows super-linearly with the number of samples, it is fitted on a random subset of 20,000 training samples. The k-NN classifier uses $k = 5$ neighbours with Euclidean distance. All inputs are standardised to zero mean and unit variance with statistics computed on the training split only, the random seed is fixed to 42, and every model is evaluated on the full test set. They are evaluated in two regimes: a *naive* regime, in which the trained model is applied unchanged to new attack variants, and a *retrained* regime, in which newly labelled samples are appended to the training set and the model is retrained from scratch (Section 6.2).

3.2. Proposed Model: FSL-Sybil

We build on the Prototypical Network architecture of Snell et al. [33] because it is simple, stable to train, and fast at inference. In this subsection, we describe the embedding network, the episodic training procedure, and our adaptations for Sybil detection.

3.2.1. Embedding Network

We use a three-layer feed-forward network with ReLU activations and dropout for regularisation. The architecture is: $\text{Linear}(20, 64) \to \text{ReLU} \to \text{Dropout}(0.3) \to \text{Linear}(64, 128) \to \text{ReLU} \to \text{Dropout}(0.3) \to \text{Linear}(128, 64) \to \text{ReLU}$, as shown in Fig. 4. The output is a 64-dimensional embedding. The three linear layers contain 1,344, 8,320 and 8,256 trainable parameters, 17,920 in total, which corresponds to 70 KB of storage at 32-bit precision—small enough to deploy on embedded processors. Dropout is active only during training; at inference the network is a fixed sequence of three matrix multiplications and ReLU activations.

3.2.2. Episodic Training

Training proceeds in episodes. Each episode samples $N = 2$ classes (legitimate and Sybil) and for each class samples K support examples and 15 query examples. The embedding network f_θ maps all examples into embedding space. Class prototypes are computed as the mean of support embeddings:

$$\mathbf{c}_k = \frac{1}{|S_k|} \sum_{(\mathbf{x}_i, y_i) \in S_k} f_\theta(\mathbf{x}_i) \tag{1}$$

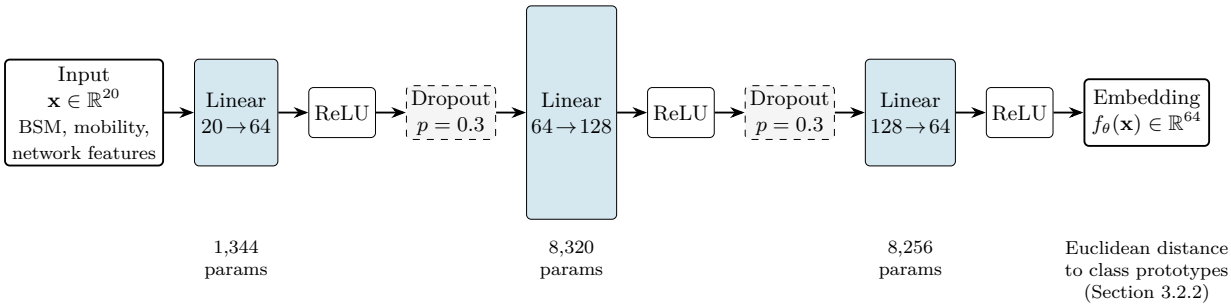


Figure 4: Embedding network f_θ of FSL-Sybil. The heights of the linear-layer boxes are proportional to their output width; the three linear layers hold 17,920 parameters (70 KB at 32-bit precision). Dropout layers (dashed) are active only during training.

For each query example, we compute negative squared Euclidean distance to each prototype, apply softmax to obtain class probabilities, and minimise cross-entropy loss:

$$\mathcal{L}(\theta) = -\frac{1}{|Q|} \sum_{(\mathbf{x}_q, y_q) \in Q} \log \frac{\exp(-\| f_\theta(\mathbf{x}_q) - \mathbf{c}_{y_q} \|^2)}{\sum_k \exp(-\| f_\theta(\mathbf{x}_q) - \mathbf{c}_k \|^2)} \tag{2}$$

We train with Adam at a learning rate of 0.001 for up to 1,000 episodes and halve the learning rate every 200 episodes. Every 50 episodes, the model is validated on 200 episodes drawn from the validation split with dropout disabled; the checkpoint with the highest validation accuracy is kept, and training stops early after five consecutive validations without improvement. We train separate models for $K \in \{1, 3, 5, 10\}$, where K is the number of support samples per class used at inference time. All models share the same architecture but develop different embeddings that are optimised for the specific amount of support information available. Algorithm 1 summarises the training procedure.

3.2.3. Deployment and Inference

At deployment, the trained embedding network is fixed. Given any K new labelled support examples per class, the defender computes the two class prototypes on-the-fly and classifies incoming BSM feature windows by nearest prototype. This is the key mechanism enabling rapid adaptation: when a new attack variant appears, the defender only needs to update the support set, not retrain the network. Algorithm 2 gives the procedure. Adapting to a new variant costs one forward pass over the $2K$ support examples and no gradient update, and the Sybil support set may consist solely of examples of the new variant, as in the novel-attack experiment of Section 6.2.

4. Simulation Environment and Dataset

This section describes how the labelled dataset used to train and evaluate all models was generated. Section 4.1 introduces the co-simulation stack of OMNeT++ 6.1, veins_inet and SUMO 1.20.0, and Section 4.2

Algorithm 1 Episodic training of the FSL-Sybil embedding network

Require: Training split $\mathcal{D}_{\text{tr}}$ and validation split $\mathcal{D}_{\text{val}}$ of labelled BSM feature vectors (class 0: legitimate, class 1: Sybil); support size K; query size $M = 15$ per class; learning rate $\eta = 0.001$; maximum number of episodes $E = 1000$

Ensure: Trained embedding network f_{θ^*}

Initialise θ; best validation accuracy $a^* \leftarrow 0$; patience counter $p \leftarrow 0$
for $e = 1$ **to** E **do**
  for each class $k \in \{0, 1\}$ **do**
    Sample $K + M$ examples of class k from $\mathcal{D}_{\text{tr}}$ and split them into a support set S_k (K examples) and a query set Q_k (M examples)
  end for
  Compute the prototypes $\mathbf{c}_0, \mathbf{c}_1$ from S_0, S_1 with Eq. (1)
  Compute the loss $\mathcal{L}(\theta)$ on $Q = Q_0 \cup Q_1$ with Eq. (2)
  $\theta \leftarrow \text{Adam}(\theta, \nabla_\theta \mathcal{L}, \eta)$
  if $e \bmod 50 = 0$ **then**
    $a \leftarrow$ mean query accuracy over 200 episodes sampled from $\mathcal{D}_{\text{val}}$ (dropout disabled, no gradient)
    if $a > a^*$ **then**
      $a^* \leftarrow a; \quad \theta^* \leftarrow \theta; \quad p \leftarrow 0$
    else
      $p \leftarrow p + 1$
      if $p = 5$ **then**
        break {early stopping}
      end if
    end if
  end if
  if $e \bmod 200 = 0$ **then**
    $\eta \leftarrow \eta / 2$
  end if
end for
return f_{θ^*}

Algorithm 2 Retraining-free adaptation and inference of FSL-Sybil at an RSU

Require: Trained embedding network f_{θ^*} (parameters frozen, dropout disabled); labelled support sets S_0 (K legitimate examples) and S_1 (K Sybil examples, possibly of a newly observed variant only); stream of incoming BSM feature vectors

Ensure: Label $\hat{y} \in \{\text{legitimate}, \text{Sybil}\}$ for every incoming feature vector

$\mathbf{c}_k \leftarrow \frac{1}{K} \sum_{\mathbf{x}_i \in S_k} f_{\theta^*}(\mathbf{x}_i)$ for $k \in \{0, 1\}$ {Eq. (1); no gradient step}
for each incoming feature vector $\mathbf{x}$ **do**
  $\mathbf{z} \leftarrow f_{\theta^*}(\mathbf{x})$
  $\hat{y} \leftarrow \arg\min_{k \in \{0,1\}} \|\mathbf{z} - \mathbf{c}_k\|^2; \quad P(y = k \mid \mathbf{x}) \leftarrow \text{softmax}_k\left(-\|\mathbf{z} - \mathbf{c}_k\|^2\right)$
  Report $\hat{y}$ and its probability
  if K labelled examples of a new attack variant become available **then**
    $S_1 \leftarrow$ the new examples; recompute $\mathbf{c}_1$ as in line 1 {θ^* is not changed}
  end if
end for

presents the real-world road network of Indiranagar, Bengaluru. Section 4.3 defines the ten simulation scenarios in which the three Sybil attack variants are injected, Section 4.4 explains how 20 features are extracted from BSM, mobility and network-layer observations, and Section 4.5 summarises the resulting dataset of 65,087 labelled samples.

4.1. Simulation Stack

OMNeT++ [54] is a modular, event-driven discrete simulation framework widely used for network research. When it is combined with SUMO [55] for microscopic traffic simulation and with Veins [56] (or its INET-based successor veins_inet) as the bridging framework, it enables high-fidelity co-simulation of vehicular communication at the MAC, PHY and application layers. We adopt this toolchain because it is the community standard and because results are directly reproducible by other researchers.

Our simulation environment uses OMNeT++ 6.1 [54] for discrete-event simulation, INET 4.5 for the IPv4 and Ieee80211Dimensional radio models, the veins_inet framework [56] for TraCI-based coupling with SUMO, and SUMO 1.20.0 [55] for microscopic traffic simulation. The IEEE 802.11p radio [2] operates at 5.9 GHz with 20 mW transmit power, a 300 m nominal communication range, and 10 Hz BSM broadcast rate [3]. We implement a custom `SybilDetectionApp` module extending `veins::VeinsInetApplicationBase`, which handles legitimate BSM transmission, attack logic, BSM reception, and per-window feature extraction to CSV for offline analysis.

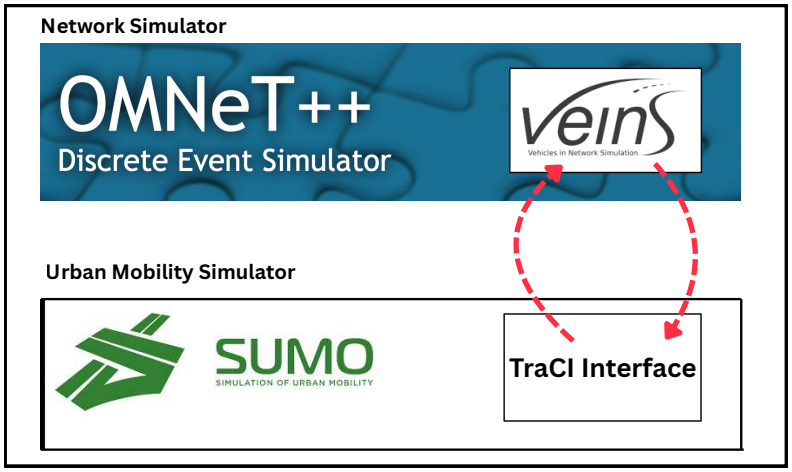


Figure 5: System architecture: OMNeT++ + veins_inet + SUMO co-simulation with custom SybilDetectionApp module.

4.2. Road Network: Indiranagar, Bengaluru

We use a real-world road network extracted from OpenStreetMap covering Indiranagar, a dense urban area in Bengaluru, India. It's 2.5 km × 2.5 km and includes major arterials like 100 Feet Road and CMH Road,

inner lanes, signalised intersections and one-way streets. Using the standard SUMO `netconvert` pipeline, we create a drivable network with realistic topology. Vehicle routes are generated using `randomTrips.py` and include standard cars (80%), buses (10%) and auto-rickshaws (10%). These vehicle types have parameters calibrated for Indian urban traffic, such as a 2.6 m length and a lower top speed for auto-rickshaws.



Figure 6: Indiranagar, Bengaluru road network imported from OpenStreetMap and rendered in SUMO.

4.3. Scenario Design

We simulate ten scenarios (labelled S1 through S10) varying along three axes: attack type (SA-RND, SA-TRJ, SA-CRD), the number of attackers (3, 5 or 10) and the number of Sybil identities per attacker (5 or 10). Scenarios S9 and S10 also incorporate intermittent attacks, where the attack alternates on and off every 30 seconds. Each scenario runs for 600 simulated seconds with a BSM interval of 0.1 seconds and is repeated five times with different random seeds for statistical validity. The full configuration is shown in Table 3.

4.4. Feature Extraction

For each receiver, we calculate a 20-dimensional feature vector over a 5-second sliding window with a 2.5-second stride. These features encompass three semantic categories.

Table 3: The 10 simulation scenarios. The Remarks column summarises the distinguishing property of each scenario.

Scenario	Attackers	Sybil/Atk	Attack Type	Remarks
S1	3	5	SA-RND	Low attacker density
S2	3	5	SA-TRJ	Low attacker density
S3	3	5	SA-CRD	Low attacker density
S4	5	5	SA-RND	Moderate attacker density
S5	5	5	SA-TRJ	Moderate attacker density
S6	5	5	SA-CRD	Moderate attacker density
S7	5	10	SA-CRD	Dense Sybil
S8	10	10	SA-CRD	Heavy attack ratio
S9	5	5	SA-CRD	Intermittent (30 s)
S10	10	10	SA-CRD	Heavy + Intermittent

BSM features (10): These include message frequency, the average claimed position (x, y), the average and variance of claimed speed, the average heading, the heading change rate, the average and variance of claimed acceleration and the proportion of braking events.

Mobility features (6): These cover distance travelled within the window, the number of physically implausible position jumps, the maximum observed jump, speed–position consistency, trajectory smoothness (measured by mean jerk) and the number of neighbours.

Network features (4): These include the mean approximated RSSI, the consistency between RSSI and claimed distance, the regularity of timestamps, and the consistency between sender and receiver distance.

4.5. Dataset Summary

After merging all per-node, per-scenario output CSV files removing the 30-second warm-up window and dropping rows with infinities or missing values we have a final dataset of 65,087 labelled samples. This includes 49,563 legitimate observations (76.1%) and 15,524 Sybil observations (23.9%). All three attack variants are represented: SA-RND (1,315), SA-TRJ (1,299) and SA-CRD (12,910). We then partition the dataset stratifying by both attack type and class label into 65% training (42,306), 15% validation (9,763) and 20% test (13,018).

5. Methodology

This section brings the preceding components together into a single experimental workflow. Section 5.1 presents a flowchart of the complete methodology, from data generation and feature extraction to training the reported and proposed models, evaluating them and deploying the detector. Section 5.2 then specifies the experimental setup used for all experiments in Section 6.

5.1. Flowchart of the Methodology

Fig. 7 summarises the end-to-end methodology. In the *data generation* stage, the Indiranagar road network is imported from OpenStreetMap into SUMO, and OMNeT++ with veins_inet co-simulates legitimate and Sybil BSM traffic for the ten scenarios of Table 3. In the *feature engineering* stage, each receiver extracts a 20-dimensional BSM, mobility and network feature vector over sliding windows; the resulting samples are cleaned and split into stratified training, validation and test sets. In the *model learning* stage, the proposed Prototypical Network is trained episodically for $K \in \{1, 3, 5, 10\}$ while the reported models (RF, SVM and k-NN) are trained in the conventional supervised way on the same split. In the *evaluation* stage, all models are compared under clean conditions, novel-attack adaptation, CPU-only computational profiling, adaptive RSSI spoofing and RSSI degradation. Finally, the trained embedding is deployed with a support set that can be updated whenever a new attack variant is observed, without retraining.

5.2. Experimental Setup

All experiments are run on a MacBook Pro with an Apple M-series SoC, using the PyTorch [57] MPS backend for FSL training and scikit-learn 1.3 for the reported models. Computational efficiency measurements are performed on CPU only (single thread) to reflect realistic edge deployment. Each reported result is the mean of five random seeds, and confidence intervals are given as ± 1 standard deviation. For FSL evaluation, predictions are averaged over 100 to 500 episodes to achieve stable estimates.

6. Simulated and Experimental Results

This section reports the results of the experiments described in Section 5. Section 6.1 compares all methods on the standard test set under ideal conditions, and Section 6.2 evaluates the central scenario of this work, adaptation to a Sybil variant unseen during training. Section 6.3 profiles the computational cost of each model on CPU hardware, Section 6.4 analyses feature importance and the dependence on RSSI, and Section 6.5 measures how detection degrades when RSSI features are corrupted. Finally, Section 6.6 compares FSL-Sybil with the state of the art.

6.1. Standard Evaluation

We firstly evaluate all methods on the standard test set under ideal conditions. Table 4 shows the key metrics; all methods achieve high accuracy (>99%) on this dataset demonstrating the strong discriminative power of RSSI-based features. Interestingly, all methods cluster near 100%. This is arguably the most important observation of the baseline experiment: under ideal simulation conditions Sybil detection isn't a difficult machine learning problem. While accuracy on clean test data is important, it's not everything. What truly matters is how methods perform under realistic constraints like limited labelled data for novel attacks and resource-constrained deployment hardware. The remaining experiments focus on these constraints.

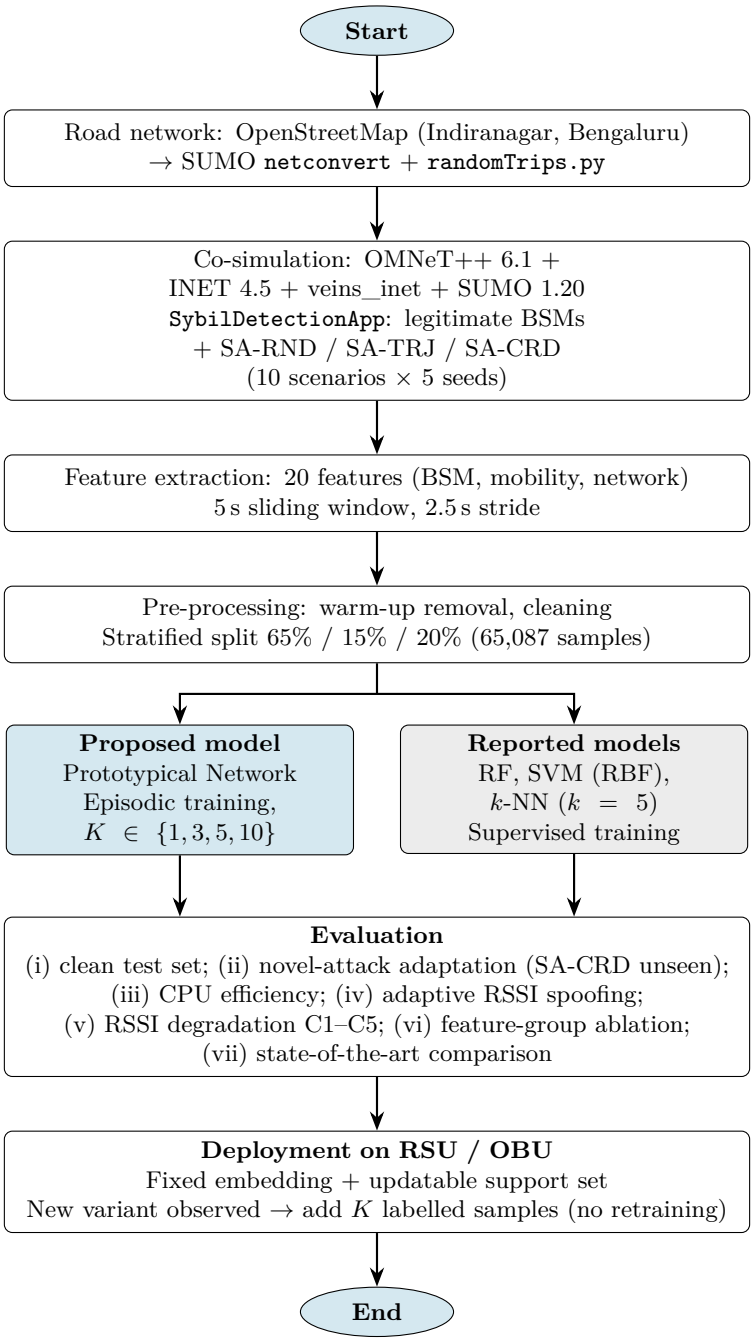


Figure 7: Flowchart of the proposed methodology, from road-network import and co-simulation to feature extraction, model learning, evaluation and deployment.

Table 4: Test set performance under clean conditions. N is the number of labelled training samples used

Method	N	**Accuracy**	**Precision**	**Recall**	**F1**
FSL $K=1$	2	0.9972	0.9885	1.0000	0.9942
FSL $K=5$	10	0.9995	0.9978	1.0000	0.9989
FSL $K=10$	20	0.9995	0.9978	1.0000	0.9989
Random Forest	42,306	1.0000	1.0000	1.0000	1.0000
SVM (RBF)	42,306	1.0000	1.0000	1.0000	1.0000
k-NN ($k=5$)	42,306	0.9991	0.9962	1.0000	0.9981

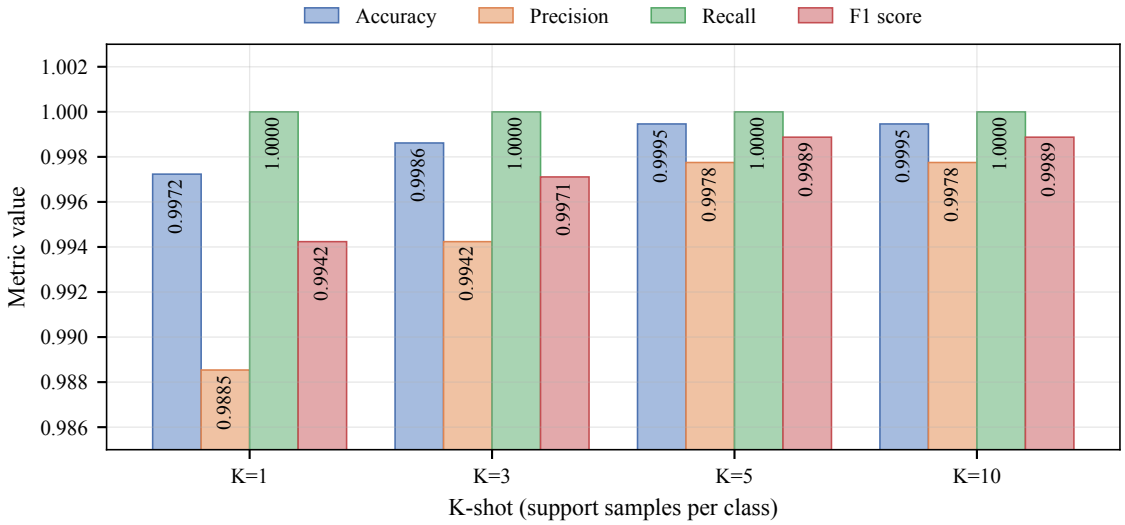


Figure 8: Shows the prototypical network's performance across various K-shot values ($K = 1, 3, 5, 10$) for accuracy, precision, recall and the F1 score.

6.2. Novel Attack Adaptation

We now consider the core scenario that motivates few-shot learning: the deployed detector is trained only on SA-RND and SA-TRJ attacks, and a novel SA-CRD variant emerges in the field. We ask: with K newly-labelled SA-CRD samples, how well can each method adapt?

For the proposed FSL model, adaptation means simply updating the support set to include K SA-CRD samples and K legitimate samples. No retraining occurs. For baselines, we consider two regimes: (i) *naive*, in which the model is not updated and must classify SA-CRD using the knowledge of SA-RND and SA-TRJ only; (ii) *retrained*, in which K SA-CRD samples and K legitimate samples are appended to the training set and the model is retrained from scratch. Results are in Table 5.

Three observations are noteworthy. First, FSL with $K=1$ reaches 99.88% recall on the novel SA-CRD attack without any model update. This figure is a majority vote over 100 one-shot episodes, each with a

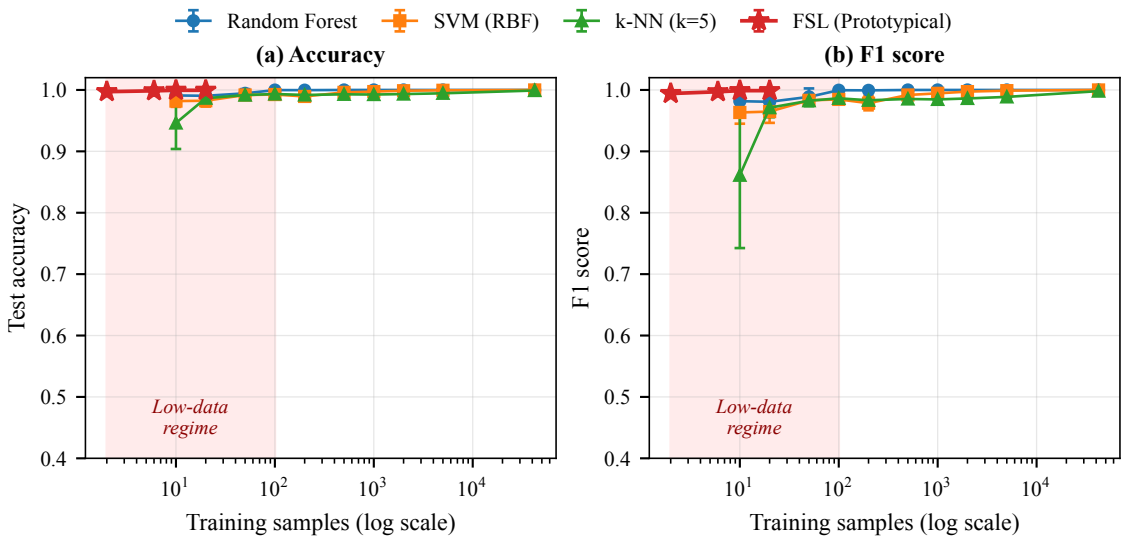


Figure 9: Data efficiency comparison: accuracy and F1 score as a function of training set size (log scale). FSL maintains high accuracy in the low-data regime where baselines degrade.

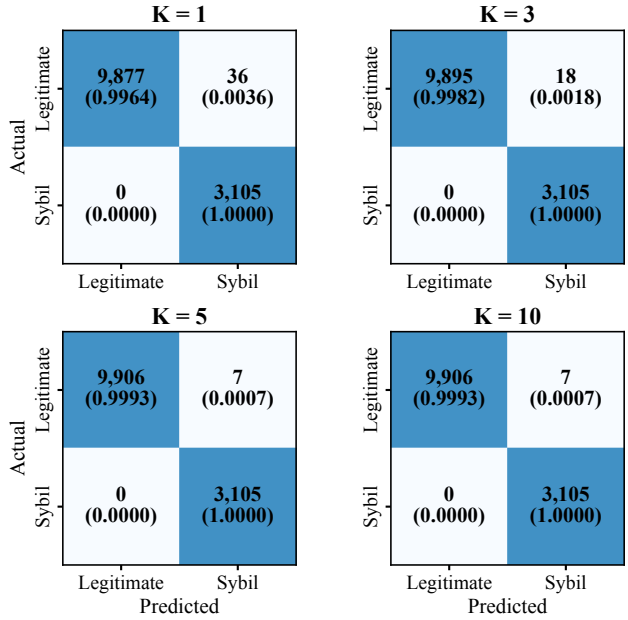


Figure 10: Confusion matrices of the Prototypical Network on the test set for $K = 1, 3, 5, 10$ (counts with row-normalised rates in parentheses). False positives decrease from 36 ($K=1$) to 7 ($K=5$), while false negatives remain at 0.

support set of two samples, and the worked example below analyses a single support set. Second, the naive Random Forest recall drops to 82.23%, confirming that traditional classifiers extrapolate poorly to novel variants without retraining. Third, SVM and k-NN recover only partially even after retraining: SVM with $K=10$ reaches 78.24% recall, and k-NN reaches 55.44%.

Table 5: Novel attack adaptation on SA-CRD. Training uses only SA-RND and SA-TRJ. Recall measures SA-CRD detection rate.

Method	**Testing samples**	**Acc.**	**F1**	**SA-CRD Recall**
FSL $K=1$ (no retrain)	2	0.9993	0.9982	**0.9988**
FSL $K=5$ (no retrain)	10	0.9972	0.9930	0.9968
FSL $K=10$ (no retrain)	20	0.9984	0.9960	0.9972
RF (naive)	0	0.9641	0.9025	0.8223
RF (retrained, $+K=5$)	10	0.9995	0.9988	0.9976
RF (retrained, $+K=10$)	20	0.9998	0.9994	0.9988
SVM (naive)	0	0.8969	0.6573	0.4895
SVM (retrained, $+K=5$)	10	0.9385	0.8203	0.6954
SVM (retrained, $+K=10$)	20	0.9561	0.8779	0.7824
k-NN (naive)	0	0.9013	0.6766	0.5113
k-NN (retrained, $+K=5$)	10	0.9034	0.6854	0.5214
k-NN (retrained, $+K=10$)	20	0.9100	0.7133	0.5544

Worked example: adapting with two labelled samples. Consider a single adaptation step at an RSU after SA-CRD first appears. The operator labels exactly two feature windows: one SA-CRD window and one legitimate window, taken as the first of each class in the held-out support pool. The trained embedding network maps each window to a 64-dimensional embedding. With $K=1$ these two embeddings serve directly as the prototypes $\mathbf{c}_1$ and $\mathbf{c}_0$ (Algorithm 2, line 1), and no weight changes. Each of the 12,295 evaluation windows (9,813 legitimate and 2,482 SA-CRD) is then assigned to the nearer prototype. None of these windows belongs to the support pool. With this support set of two samples, the model detects 2,462 of the 2,482 SA-CRD windows (recall 99.19%) and raises 5 false alarms on the 9,813 legitimate windows (accuracy 99.80%).

The result does not depend on a favourable pair. We scored all 10,000 two-sample support sets that the 100 + 100 support pool allows, each on its own and without voting. Across these pairs the median SA-CRD recall is 99.84% and the mean is 98.77%. Of the 10,000 pairs, 94.2% reach at least 95% recall, although the worst pair reaches only 68.17%. The 99.88% in Table 5 is the majority vote of 100 such one-shot episodes, which suppresses the occasional poor pair. Because voting over several draws uses more than two labelled windows in total, the median of 99.84% is the figure that applies to a single two-sample support set.

We note that retrained Random Forest matches FSL's performance with $K=5$ labelled SA-CRD samples. This is a fair comparison that we report transparently. The operational advantage of FSL is not that it is

more accurate than a retrained Random Forest, but that it achieves this performance *without retraining*, which eliminates an entire operational pipeline: stopping the model, collecting labelled data, retraining, validating, and redeploying.

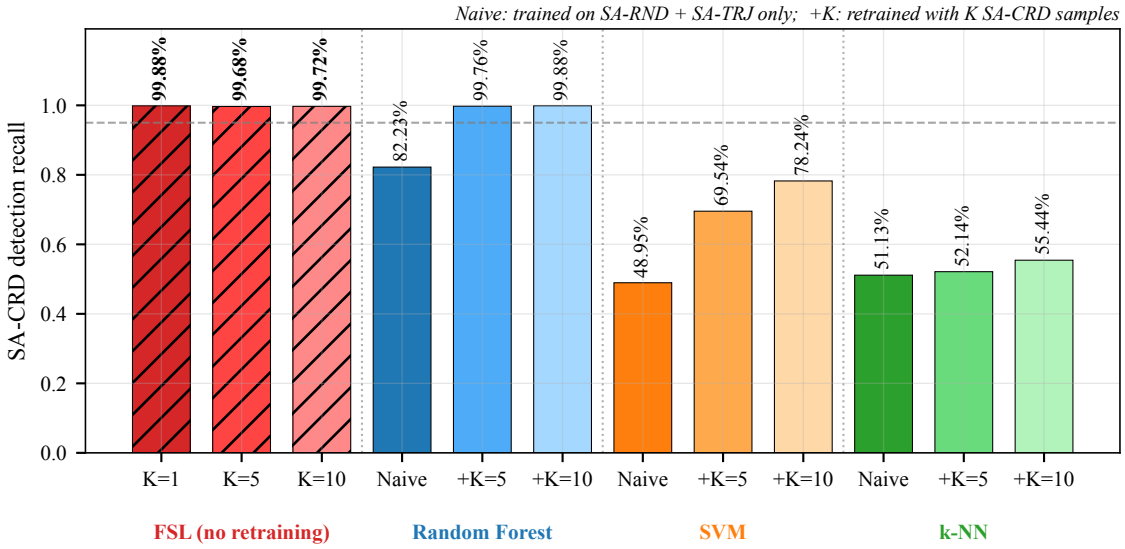


Figure 11: Novel attack adaptation comparison. FSL achieves near-perfect recall with $K = 1$, while naive baselines fail and retrained SVM/k-NN do not recover.

6.3. Computational Efficiency

We now turn to computational properties, which are critical for edge deployment on RSUs and on-board units. All measurements are performed on CPU (single thread) to simulate realistic embedded-hardware constraints. Results are in Table 6.

Table 6: Computational efficiency on CPU (single thread). FSL model size is constant across K values.

Method	**Train (s)**	**Size (KB)**	**Inf. (s)**	**Samp./sec**
FSL $K = 1$	0.25	70.0	0.009	1,416,417
FSL $K = 5$	0.19	70.0	0.010	1,330,931
FSL $K = 10$	0.20	70.0	0.009	1,414,106
Random Forest	1.59	174.5	0.016	811,531
SVM (RBF)	1.84	55.2	0.127	102,426
k-NN ($k = 5$)	0.00	6,941.5	0.243	53,483

FSL achieves 1.4 million samples per second, which is 1.75× faster than Random Forest, 14× faster than SVM, and 26× faster than k-NN. The FSL model size is 70 KB—roughly half of Random Forest and a

hundredth of k-NN (which must store the entire training set). The k-NN model size of 6.9 MB is impractical for RSU deployment.

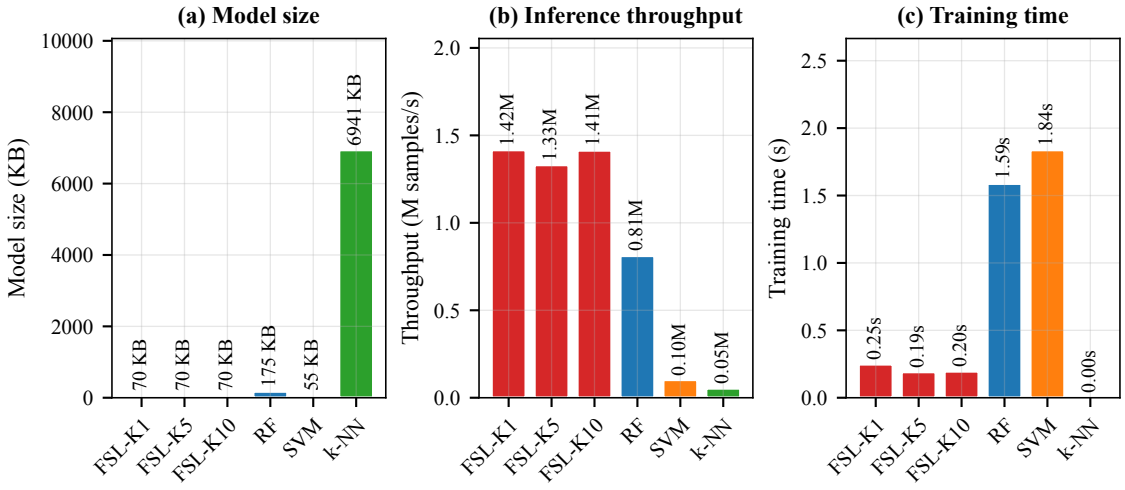


Figure 12: Computational efficiency comparison on CPU: (a) model size, (b) inference throughput, (c) training time.

6.4. Feature Importance and the RSSI Vulnerability

To understand what drives detection performance, we compute per-feature mutual information with the label and perform feature-group ablation using a Random Forest proxy. The top 5 features are `rssi_mean` (MI = 0.549), `sender_recv_dist_consistency` (0.525), `rssi_consistency` (0.525), `max_pos_jump` (0.514), and `traj_smoothness` (0.512). All four network-layer features are strongly discriminative.

This extreme dependence on RSSI motivates an important follow-up experiment: what happens if an adaptive attacker spoofs RSSI features to match the legitimate distribution? We implement this attack at test time by sampling Sybil RSSI feature values from the legitimate class distribution (with small residual noise). Under this adaptive spoofing (Level 3), Random Forest trained on clean data drops to 76.21% accuracy, SVM to 80.63%, and k-NN to 83.78%, while FSL $K = 5$ reaches 94.63%.

When RSSI features are removed entirely from both training and test, Random Forest actually recovers to 100% using only kinematic features, indicating that the problem is still solvable without RSSI if the system is designed for it. The practical implication is that a defensible production system should not rely solely on RSSI-based features, regardless of which classifier is used.

6.5. Robustness Under RSSI Degradation

To further evaluate robustness, we test all models under five RSSI degradation conditions: C1 (clean baseline), C2 (Gaussian noise with $\sigma = 5\,\text{dB}$), C3 (30% of RSSI observations masked to simulate packet loss), C4 (attacker adjusts TX power to spoof consistency features), and C5 (all RSSI features removed entirely). Models are trained on clean data and evaluated on corrupted test sets.

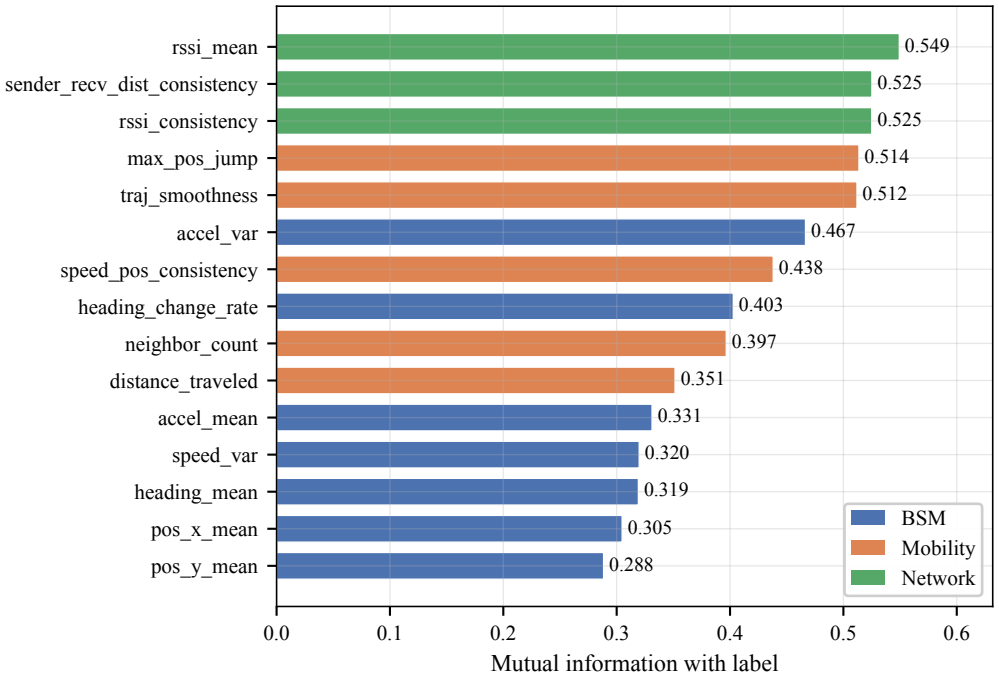


Figure 13: Top-15 features by mutual information, coloured by feature group (BSM, Mobility, Network).

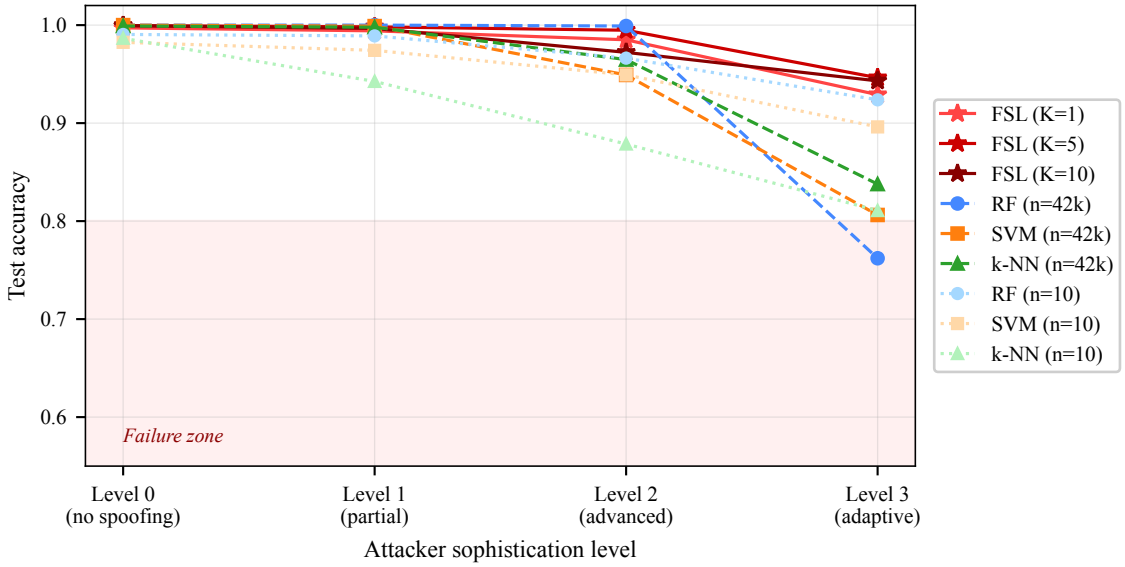


Figure 14: Detection accuracy under increasing attacker sophistication (Level 0: no spoofing through Level 3: adaptive spoofing from legitimate distribution).

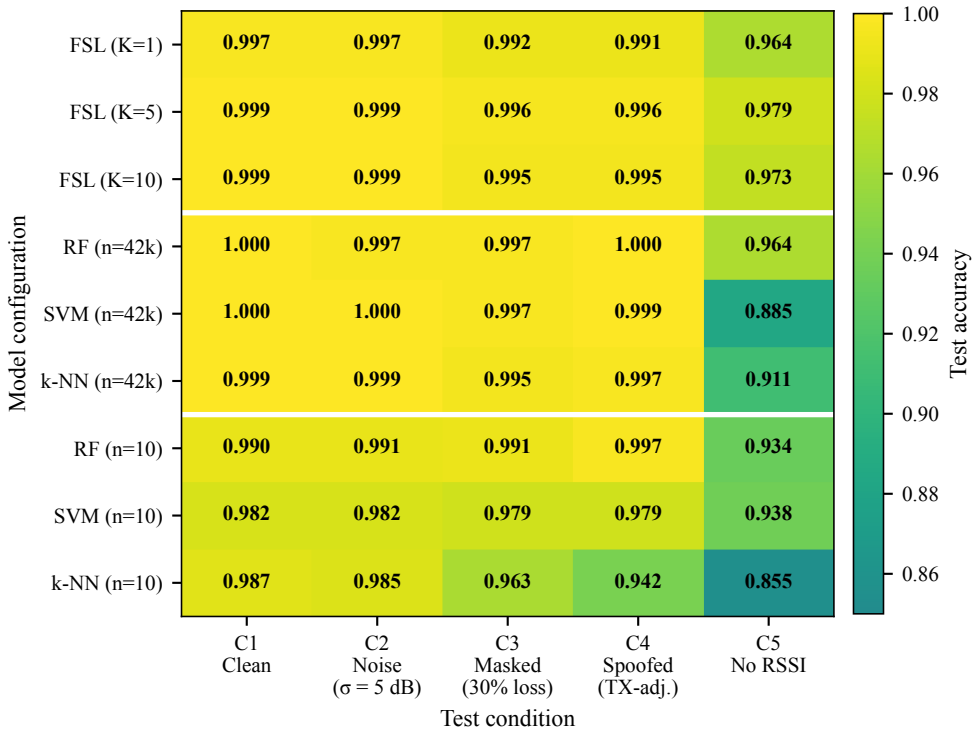


Figure 15: Robustness heatmap showing accuracy across 9 model configurations and 5 RSSI degradation conditions. Brighter (yellow) cells indicate higher accuracy and darker (purple) cells lower accuracy; exact values are printed in each cell. White lines separate FSL models, fully supervised baselines ($n \approx 42$k) and data-limited baselines ($n = 10$).

Figure 15 reveals that FSL degrades more gracefully than baselines as conditions worsen. Under C5 (no RSSI), FSL $K = 5$ maintains 97.87% accuracy while SVM drops to 88.48% and k-NN to 91.12%. Random Forest trained on 42,306 samples reaches 96.39%, which is close to FSL but required three orders of magnitude more labelled data.

Finally, we examine which feature groups sustain detection under Level 3 adaptive spoofing with clean training data and spoofed test data. Figure 17 shows that when RSSI features are removed (F5 configuration), all methods recover to near-perfect accuracy, confirming that RSSI is the vulnerability rather than the solution. The mobility-only configuration (F2) achieves 97.54% for FSL $K=5$ and 99.98% for Random Forest.

6.6. Comparison with the State of the Art

Table 7 compares FSL-Sybil with representative Sybil and misbehaviour detection methods reported in the literature. Because most prior works use different datasets, road layouts and attack definitions, the reported figures are not directly comparable in absolute terms; the table is intended to contrast what each method requires (infrastructure, labelled data, retraining) and what it achieves. The position-verification schemes of Xiao et al. [16] and Yu et al. [17] report detection rates of 95–99%, but only with several honest

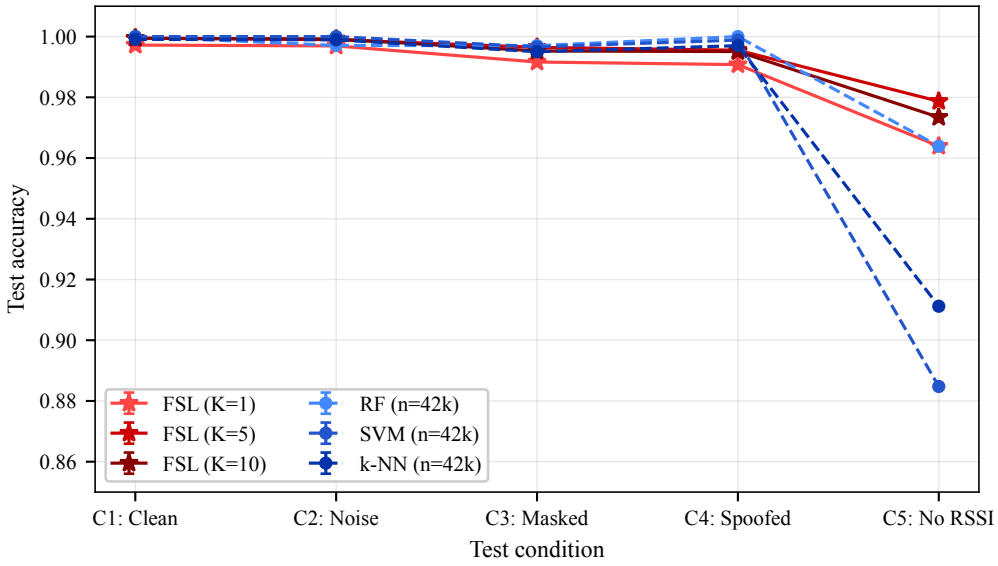


Figure 16: Accuracy degradation from clean (C1) to no-RSSI (C5) conditions. FSL (solid lines) degrades more gradually than baselines (dashed lines).

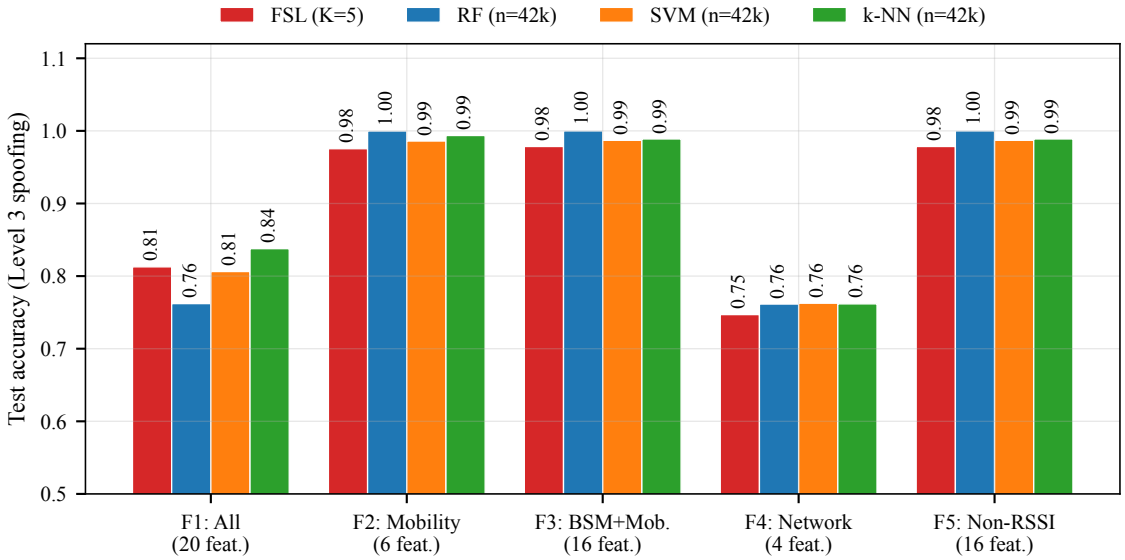


Figure 17: Feature ablation under Level 3 adaptive spoofing. F1: all features; F2: mobility only; F3: BSM+mobility; F4: network only; F5: non-RSSI.

witnesses in opposite-direction traffic, roadside certificates and position deviations of at least 10 m. The supervised detectors of Gyawali and Qian [24], Gu et al. [26] and Azam et al. [25] reach 95–98%, but they assume that every attack variant is present in the training data. The few-shot detectors of Yu and Bian [35] and Bilot et al. [38] reduce the labelling burden, but they have so far been evaluated only on generic network-flow datasets. The vehicular few-shot detectors of Dina et al. [40], Khan et al. [42], Ahmad et al. [44] and

Duy et al. [45] are closest to our setting, but they detect generic intrusions, VeReMi misbehaviour or in-vehicle CAN attacks rather than Sybil identities. Unlike every reported method, FSL-Sybil adapts to an attack variant unseen during training using only two labelled samples and no retraining, while remaining small and fast enough for RSU deployment.

7. Discussion

This section interprets the results reported in Section 6. Section 7.1 discusses the operating conditions under which few-shot learning offers an advantage over the reported models, Section 7.2 examines the limitations of the study and the threats to its validity, and Section 7.3 describes how the experiments can be reproduced.

7.1. When Does Few-Shot Learning Help?

Our results suggest a nuanced answer. Under standard evaluation with abundant clean data, few-shot learning does not outperform Random Forest; both achieve near-perfect accuracy, and Random Forest is slightly ahead. The advantages of few-shot learning emerge in two specific operational regimes. First, when novel attack variants appear and labelled examples are scarce, FSL adapts instantly by updating its support set, whereas baselines either fail (naive) or require full retraining (retrained). Second, for resource-constrained edge deployment, FSL offers the smallest model and highest inference throughput, which matters on RSU hardware. Researchers and practitioners considering few-shot learning for this domain should evaluate it specifically under these two regimes, not just on clean test accuracy.

7.2. Limitations and Threats to Validity

We discuss threats to validity openly because the problem benchmarking field often under-reports them. First, the near-perfect clean-condition accuracy indicates the simulation does not capture the full difficulty of real-world Sybil detection. Real deployments face GPS noise, packet loss, multi-path channel effects, and environmental interference that our simulation abstracts away. Second, our SA-CRD attack is an approximation of a sophisticated attacker; a real attacker with adjustable TX power could potentially spoof RSSI directly. Third, the evaluation is limited to simulated data. Deploying in a real urban environment would require field data collection and likely adaptation of the feature set.

We plan to address these limitations in follow-up work by integrating GPS jitter models, empirical packet loss distributions, and a realistic adaptive attacker that actively learns and mimics legitimate distributions during the simulation itself.

Table 7: Comparison with the state of the art. All values are taken from the original publications.

Ref.	Method	Performance	Remarks
Cryptography and infrastructure based			
[14]	Coarse/fine-grained hashing of pseudonyms overheard by roadside boxes (P2DAP)	Detection latency ≈30 s at ≈580 packets of authority overhead (τ-P2DAP, 10 attackers; ns-2)	Detection accuracy not reported; requires roadside boxes and a trusted authority
Position verification			
[16]	RSSI-based position verification by opposite-flow witnesses with hypothesis testing	Detection rate $\geq$95%, FPR $\leq$5% (>5 witnesses, 10-unit observation; simulation)	Needs roadside base stations and honest witnesses; cannot resolve Sybils within ≈10 m
[17]	RANSAC-based cooperative verification + Presence Evidence System + statistical test	Detection rate >99% (deviation $\geq$10 m, observation $\geq$10 s; NS2 + GrooveNet, Detroit map)	Fails on one-way roads or without opposite traffic; RSSI dependent
[18]	Periodic exchange and intersection of neighbour lists; identities sharing neighbours beyond a time threshold flagged	FPR ≈0.1% (threshold 80 s, 500 m range; MMTS traces, 10–25% attackers)	Fully distributed, no RSU; attackers active for less than the threshold evade detection; more false positives in small networks
[19]	RSS and Euclidean-distance checks with an RSU feedback and reward system; detects attacker and Sybil nodes	Detection rate up to 99.89% (500 m) and 97.88% (250 m); FPR 0.023% (500 m, 10 s observation; NS2 + SUMO, Clermont-Ferrand)	Needs RSUs and a central VANET server; RSS dependent
Supervised machine learning			
[23]	Plausibility checks as features for SVM / k-NN	Precision >20% higher than plausibility checks, recall within 5% (VeReMi)	Position falsification; full retraining for new attacks
[26]	Driving-pattern-matrix eigenvalues + RBF-kernel SVM	TPR 97.6%, FPR 2.0% ($\sigma = 0.3$; SUMO urban scenario)	Needs RSU and near-capacity traffic (26–42 veh/km)
[24]	Cooperative ML detector (bagging) for false alerts and position falsification	Precision 0.98, recall 0.94, F1 0.96 (VeReMi, position falsification)	Supervised; unknown attacks left to future work
[25]	Majority-voting ensemble of k-NN, NB, DT, SVM and LR	Accuracy 95%	Fully supervised; new variants require retraining
Few-shot learning (generic network intrusion)			
[35]	Metric-based FSL with centre loss (CNN embedding)	Accuracy 92.34% (NSL-KDD Test+) with <1% of training data	Generic network IDS; not vehicular
[38]	Self-supervised GNN + few-shot decoder (FEAE)	Macro-F1 97.44% (NF-CSE-CIC-IDS2018-v2), 92.64% (NF-UNSW-NB15-v2), 1 label per attack family	Network-flow graphs; not vehicular
Few-shot and meta-learning (vehicular)			
[40]	Self-supervised pre-training + Prototypical Network + transfer learning (FSProtoTransfer)	With 20% of labels, precision, recall and F1 41%, 78% and 70% above fully supervised models (two VANET datasets)	Generic VANET intrusion detection; not Sybil-specific
[42]	Reptile-style meta-learning with a topology-aware encoder and adapter modules	Average AUC 0.70, F1 0.62; adaptation in under 1 s (VeReMi)	Misbehaviour detection; adapts by fine-tuning adapters
[44]	Stochastic Prototypical Network on a Random Neural Network	Accuracy up to 99.17% (15-shot)	In-vehicle CAN bus, not V2X
[45]	Few-shot convolutional anomaly transformer	F1 0.9994 with 2% of training data (Car Hacking); 0.9918 (CAN-ML)	In-vehicle CAN bus, not V2X
[48]	Uncertainty-aware Prototypical Network, Euclidean–Mahalanobis distance	Accuracy 80.6% ($K = 1$), 88.3% ($K = 3$) on unseen GNSS attacks (GPS, BeiDou, GLONASS)	Retraining-free like ours, but GNSS spoofing/jamming, not Sybil
This work	Prototypical Network (FSL-Sybil), retraining-free adaptation	Accuracy 99.95% (clean, $K = 5$); SA-CRD recall 99.88% (unseen, $K = 1$); 70 KB, 1.4M samples/s	Two labelled samples to adapt; no retraining; CPU-deployable

7.3. Reproducibility

The raw and processed datasets generated in this simulation, the trained checkpoints and the training and evaluation scripts are publicly available in the repository `https://github.com/bashkid/fslSybil-data`. Every table and data figure in Section 6 comes from a named script that loads the committed checkpoints and data splits, so the reported numbers can be regenerated without retraining. The repository README lists the script and results file behind each table and figure. The simulation scenarios are deterministic given the reported random seeds.

8. Conclusion

We presented FSL-Sybil, a Prototypical Network-based few-shot learning framework for Sybil detection in the Internet of Vehicles. We showed that under clean conditions the problem is essentially solved by any reasonable classifier, but that the advantages of few-shot learning emerge in two practically important operational regimes. First, FSL adapts to novel attack variants with two labelled samples, achieving 99.88% recall where naive Random Forest drops to 82.23% and only retraining-based recovery matches FSL's performance. Second, FSL offers the smallest model (70 KB) and the highest inference throughput (1.4M samples/sec) among the methods evaluated, making it well-suited for resource-constrained edge deployment. We also reported a critical security finding that RSSI-based features provide near-perfect detection in idealised conditions but create a single point of failure under adaptive spoofing, and we provided design guidance accordingly.

We view this work as a first step towards adaptive, edge-deployable Sybil defence for real IoV deployments. Ongoing work is extending the simulation to include GPS noise, packet loss, and adaptive attackers, which we believe will widen the practical gap between few-shot and traditional approaches and enable evaluation under more realistic conditions.

Authorship Contribution Statement

Rohit Kumar: Conceptualisation, Methodology, Simulation, Comparative analysis, Visualisation, Writing – original draft. **Chittaranjan Mandal:** Supervision, Conceptualisation, Validation, Writing – review & editing. **Arun Raj Kumar P:** Validation, Resources, Writing – review .

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Ethics, Consent to Participate, and Consent to Publish

Ethics, Consent to Participate, and Consent to Publish declarations: not applicable.

Clinical Trial Number

Clinical trial number: not applicable.

Data Availability

The dataset is made publicly available at `https://github.com/bashkid/fslSybil-data` (see Section 7.3).

Acknowledgements

The authors sincerely thank Dr. Brijesh Mishra for his suggestions during the preparation of this manuscript. The authors also gratefully acknowledge CMR University, Bengaluru, and the Indian Institute of Technology Kharagpur for providing the research facilities and academic support that made this work possible.

References

[1] J. Contreras-Castillo, S. Zeadally, J. A. Guerrero-Ibañez, Internet of vehicles: Architecture, protocols, and security, IEEE Internet of Things Journal 5 (5) (2018) 3701–3709.

[2] IEEE standard for wireless access in vehicular environments (WAVE), Tech. Rep. IEEE Std 1609.4, IEEE (2016).

[3] SAE International, Dedicated short range communications (DSRC) message set dictionary, Tech. Rep. SAE J2735 (2020).

[4] J. B. Kenney, Dedicated short-range communications (DSRC) standards in the United States, Proceedings of the IEEE 99 (7) (2011) 1162–1182.

[5] R. Kumar, P. Basak, J. C. Patni, Comparative study of generative deep learning for malicious node detection in iov, in: 2025 IEEE International Conference on Blockchain and Distributed Systems Security (ICBDS), IEEE, 2025, pp. 1–7.

[6] M. Raya, J.-P. Hubaux, Securing vehicular ad hoc networks, Journal of Computer Security 15 (1) (2007) 39–68.

[7] R. G. Engoulou, M. Bellaïche, S. Pierre, A. Quintero, VANET security surveys, Computer Communications 44 (2014) 1–13.

[8] M. N. Mejri, J. Ben-Othman, M. Hamdi, Survey on VANET security challenges and possible cryptographic solutions, Vehicular Communications 1 (2) (2014) 53–66.

[9] H. Hasrouny, A. E. Samhat, C. Bassil, A. Laouiti, VANet security challenges and solutions: A survey, Vehicular Communications 7 (2017) 7–20.

[10] R. Krishnan P., A. R. Kumar P., Security and privacy in vanet : Concepts, solutions and challenges, in: 2020 International Conference on Inventive Computation Technologies (ICICT), 2020, pp. 789–794. doi:10.1109/ICICT48043.2020.9112535.

[11] J. R. Douceur, The Sybil attack, in: Peer-to-Peer Systems (IPTPS 2002), Lecture Notes in Computer Science, Vol. 2429, Springer, 2002, pp. 251–260.

[12] J. Newsome, E. Shi, D. Song, A. Perrig, The Sybil attack in sensor networks: Analysis & defenses, in: Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks (IPSN), 2004, pp. 259–268.

[13] R. Lu, X. Lin, H. Zhu, P.-H. Ho, X. Shen, ECPP: Efficient conditional privacy preservation protocol for secure vehicular communications, in: Proceedings of IEEE INFOCOM, 2008, pp. 1229–1237.

[14] T. Zhou, R. R. Choudhury, P. Ning, K. Chakrabarty, P2DAP—Sybil attacks detection in vehicular ad hoc networks, IEEE Journal on Selected Areas in Communications 29 (3) (2011) 582–594.

[15] S. Park, B. Aslam, D. Turgut, C. C. Zou, Defense against Sybil attack in the initial deployment stage of vehicular ad hoc network based on roadside unit support, Security and Communication Networks 6 (4) (2013) 523–538.

[16] B. Xiao, B. Yu, C. Gao, Detection and localization of Sybil nodes in VANETs, in: Proceedings of the Workshop on Dependability Issues in Wireless Ad Hoc Networks and Sensor Networks (DIWANS), 2006, pp. 1–8.

[17] B. Yu, C.-Z. Xu, B. Xiao, Detecting Sybil attacks in VANETs, Journal of Parallel and Distributed Computing 73 (6) (2013) 746–756.

[18] J. Grover, M. S. Gaur, V. Laxmi, N. K. Prajapati, A Sybil attack detection approach using neighboring vehicles in VANET, in: Proceedings of the 4th International Conference on Security of Information and Networks (SIN '11), ACM, 2011, pp. 151–158.

[19] R. Krishnan P., A. R. Kumar P., A collaborative strategy for detection and eviction of Sybil attacker and Sybil nodes in VANET, International Journal of Communication Systems 34 (3) (2021) e4621. doi:10.1002/dac.4621.

[20] P. Golle, D. Greene, J. Staddon, Detecting and correcting malicious data in VANETs, in: Proceedings of the 1st ACM International Workshop on Vehicular Ad Hoc Networks (VANET), 2004, pp. 29–37.

[21] R. W. van der Heijden, T. Lukaseder, F. Kargl, VeReMi: A dataset for comparable evaluation of misbehavior detection in VANETs, in: Security and Privacy in Communication Networks (SecureComm 2018), Springer, 2018, pp. 318–337.

[22] J. Kamel, M. Wolf, R. W. van der Heijden, A. Kaiser, P. Urien, F. Kargl, VeReMi extension: A dataset for comparable evaluation of misbehavior detection in VANETs, in: Proceedings of IEEE International Conference on Communications (ICC), 2020, pp. 1–6.

[23] S. So, P. Sharma, J. Petit, Integrating plausibility checks and machine learning for misbehavior detection in VANET, in: Proceedings of the 17th IEEE International Conference on Machine Learning and Applications (ICMLA), 2018, pp. 564–571.

[24] S. Gyawali, Y. Qian, Misbehavior detection using machine learning in vehicular communication networks, in: Proceedings of IEEE International Conference on Communications (ICC), 2019, pp. 1–6.

[25] S. Azam, M. Bibi, R. Riaz, S. S. Rizvi, S. J. Kwon, Collaborative learning based Sybil attack detection in vehicular AD-HOC networks (VANETS), Sensors 22 (18) (2022) 6934. doi:10.3390/s22186934.

[26] P. Gu, R. Khatoun, Y. Begriche, A. Serhrouchni, Support vector machine (SVM) based Sybil attack detection in vehicular networks, in: Proceedings of IEEE Wireless Communications and Networking Conference (WCNC), 2017, pp. 1–6.

[27] Y. Wang, Q. Yao, J. T. Kwok, L. M. Ni, Generalizing from a few examples: A survey on few-shot learning, ACM Computing Surveys 53 (3) (2020) 63:1–63:34.

[28] T. Hospedales, A. Antoniou, P. Micaelli, A. Storkey, Meta-learning in neural networks: A survey, IEEE Transactions on Pattern Analysis and Machine Intelligence 44 (9) (2022) 5149–5169.

[29] G. Koch, R. Zemel, R. Salakhutdinov, Siamese neural networks for one-shot image recognition, in: ICML Deep Learning Workshop, 2015.

[30] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, D. Wierstra, Matching networks for one shot learning, in: Advances in Neural Information Processing Systems, Vol. 29, 2016.

[31] C. Finn, P. Abbeel, S. Levine, Model-agnostic meta-learning for fast adaptation of deep networks, in: Proceedings of the 34th International Conference on Machine Learning (ICML), 2017, pp. 1126–1135.

[32] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. S. Torr, T. M. Hospedales, Learning to compare: Relation network for few-shot learning, in: Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2018, pp. 1199–1208.

[33] J. Snell, K. Swersky, R. Zemel, Prototypical networks for few-shot learning, in: Advances in Neural Information Processing Systems, Vol. 30, 2017, pp. 4077–4087.

[34] T. Chen, S. Kornblith, M. Norouzi, G. Hinton, A simple framework for contrastive learning of visual representations, in: Proceedings of International Conference on Machine Learning (ICML), 2020.

[35] Y. Yu, N. Bian, An intrusion detection method using few-shot learning, IEEE Access 8 (2020) 49730–49740. doi:10.1109/ACCESS.2020.2980136.

[36] C. Xu, J. Shen, X. Du, A method of few-shot network intrusion detection based on meta-learning framework, IEEE Transactions on Information Forensics and Security 15 (2020) 3540–3552.

[37] J. Yang, H. Li, S. Shao, F. Zou, Y. Wu, FS-IDS: A framework for intrusion detection based on few-shot learning, Computers & Security 122 (2022) 102899.

[38] T. Bilot, N. El Madhoun, K. Al Agha, A. Zouaoui, Few edges are enough: Few-shot network attack detection with graph neural networks, in: Advances in Information and Computer Security (IWSEC 2024), Lecture Notes in Computer Science, Springer, 2024, arXiv:2501.16964.

[39] A. Roszeitis, V. Jüttner, E. Buchmann, Few-shot learning for network intrusion detection: Methods, datasets, and performance, arXiv preprint arXiv:2609.11275 (2026).

[40] A. S. Dina, C. J. Edell, K. O. Elish, A. Sarker, FSProtoTransfer: Synergizing few-shot, prototypical networks, and transfer learning for intrusion detection in VANETs, in: Proceedings of the International Florida Artificial Intelligence Research Society Conference (FLAIRS), Vol. 38, 2025. doi:10.32473/flairs.38.1.138952.

[41] S. Shafiq, H. M. Awan, A. A. Khan, A. Hussain, I. Javed, MAFSID: Multi-agent few-shot intrusion detection for VANETs through rapid collaborative learning, Transactions on Emerging Telecommunications Technologies 36 (11) (2025) e70285. doi:10.1002/ett.70285.

[42] A. Khan, A. Junaid, G. Husnain, K. J. Alzahrani, H. K. Alkahtani, An efficient intrusion detection system using domain-aware meta-learning with adapter-based few-shot adaptation in vehicular ad-hoc networks (VANETs), IET Intelligent Transport Systems 20 (1) (2026) e70182. doi:10.1049/itr2.70182.

[43] B. Xu, J. Zhao, B. Wang, G. He, Detection of zero-day attacks via sample augmentation for the Internet of Vehicles, Vehicular Communications 52 (2025) 100887. doi:10.1016/j.vehcom.2025.100887.

[44] J. Ahmad, S. Latif, D. Djenouri, F. Ullah, M. S. Khan, M. M. Saad, R. H. Jhaveri, P. Verma, A stochastic prototypical network for few-shot intrusion detection in CAN-based IoV network, IEEE Open Journal of the Communications Society 6 (2025) 6421–6436. doi:10.1109/OJCOMS.2025.3595980.

[45] N. T. M. Duy, T. H. B. Huy, P. V. Phu, T.-D. Le, D. Kim, Multi-class intrusion detection system for in-vehicle networks using few-shot learning and convolutional anomaly transformer network, Knowledge-Based Systems 330 (2025) 114436. doi:10.1016/j.knosys.2025.114436.

[46] O. Ben Atia, M. Al Samara, I. Bennis, A meta learning framework for intrusion detection in connected vehicles, in: International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), 2025. doi:10.1109/WiMob66857.2025.11257565.

[47] S. Datta, S. Namasudra, M. R. Reddy, A. K. Sangaiah, S. Kumari, A lightweight few-shot learning-based traffic classification system for secure Internet of Vehicles, IEEE Transactions on Intelligent Transportation SystemsEarly access (2025). doi:10.1109/TITS.2025.3596139.

[48] D. Vasan, M. Hammoudeh, C. Firth, Proactive threat hunting and anomaly detection in consumer electronic ecosystems, IEEE Transactions on Consumer Electronics (2026). doi:10.1109/TCE.2026.3703197.

[49] A. Gyawali, R. Das, H. Rathore, Can llms classify vehicular basic safety messages anomalies?, in: Proceedings of the International Florida Artificial Intelligence Research Society Conference, FLAIRS, Vol. 39, 2026. doi:10.32473/flairs.39.1.141767.

[50] M. J. Page, J. E. McKenzie, P. M. Bossuyt, I. Boutron, T. C. Hoffmann, C. D. Mulrow, et al., The PRISMA 2020 statement: An updated guideline for reporting systematic reviews, BMJ 372 (2021) n71.

[51] L. Breiman, Random forests, Machine Learning 45 (1) (2001) 5–32.

[52] C. Cortes, V. Vapnik, Support-vector networks, Machine Learning 20 (3) (1995) 273–297.

[53] T. Cover, P. Hart, Nearest neighbor pattern classification, IEEE Transactions on Information Theory 13 (1) (1967) 21–27.

[54] A. Varga, R. Hornig, An overview of the OMNeT++ simulation environment, in: Proceedings of SIMUTools, 2008.

[55] P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, E. Wießner, Microscopic traffic simulation using SUMO, in: Proceedings of IEEE International Conference on Intelligent Transportation Systems (ITSC), IEEE, 2018, pp. 2575–2582.

[56] C. Sommer, R. German, F. Dressler, Bidirectionally coupled network and road traffic simulation for improved IVC analysis, IEEE Transactions on Mobile Computing 10 (1) (2011) 3–15.

[57] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al., PyTorch: An imperative style, high-performance deep learning library, in: Advances in Neural Information Processing Systems, Vol. 32, 2019.