Machine Learning for Security Resiliency in Connected Vehicle Applications

Srivalli Boddupalli, Richard Owoputi, Chengwei Duan, Tashfique Choudhury, and Sandip Ray

1 Resiliency Needs and Challenges in CAV Applications

Automotive systems have evolved over the last two decades from primarily mechanical and electro-mechanical systems into complex cyber-physical systems with a wide range of communication and sensory capabilities. In addition to a variety of sensors (e.g., Radar, Lidar, etc.), a modern vehicle is equipped with various interfaces for Internet connectivity, and vehicular communications (V2X) technology (e.g., Digital Short Range Radio) to interact with other vehicles (V2V), components of transportation infrastructure (V2I), or other electronic devices connected to the Internet (V2IoT). The combination of sophisticated sensors and communication enables *connected and autonomous vehicle* (CAV) applications, i.e., applications that exploit cooperative information sharing among vehicles and infrastructures for streamlining traffic movement, improving road safety, and efficient infrastructure utilization. CAV applications being developed today include platooning [5], cooperative dynamic route management [2, 3], intersection management [12], etc. With increasing proliferation of connectivity and autonomy of vehicles, the trend is towards increasing sophistication of such applications and the consequent potential to bring in transformative impact on road safety, passenger comfort, and environmental sustainability.

However, one critical challenge with CAV applications is their vulnerability to a spectrum of cyber-attacks. An adversary can easily compromise the sensory and communication inputs to disrupt traffic movement, cause catastrophic accidents, and bring down the transportation infrastructure. A key problem with these attacks is that an adversary no longer needs to actually hack into the hardware or software

S. Boddupalli · R. Owoputi · C. Duan · T. Choudhury · S. Ray (✉)
Department of ECE, University of Florida, Gainesville, FL, USA
e-mail: bodsrivalli12@ufl.edu; rowoputi@ufl.edu; duan.c@ufl.edu; choudhury.t@ufl.edu; sandip@ece.ufl.edu

V. K. Kukkala, S. Pasricha (eds.), *Machine Learning and Optimization Techniques for Automotive Cyber-Physical Systems*, https://doi.org/10.1007/978-3-031-28016-0_16

of the vehicle that is the target of the cyber-attack: sending misleading or even malformed V2X messages or sensory data is often sufficient to disrupt the connected car ecosystem.

The focus of this chapter is the problem of *real-time resiliency* in CAV applications, against adversaries that target compromising V2X or sensory inputs. We refer to these inputs as "perception inputs" or "perception channels". The impact of a successful compromise can be a perturbation of some (subset of) perception channels involved in the application, such that the inputs received would be different from actual. For instance, in Cooperative Adaptive Cruise Control (CACC), a vehicle $\mathcal{E}$ receives the velocity, relative position, and acceleration of its preceding vehicle $\mathcal{P}$; during an attack, the values received by $\mathcal{E}$ would be perceived to be different from ground truth. The focus of *real-time resiliency* is to augment the application functionality so that $\mathcal{E}$ can perform safely and efficiently, even during attack.

1.1 Constraints

Designing real-time resiliency for practical CAV applications is a challenging proposition. A viable solution must address the following key issues (among others).

How Can We Identify a Suitable Threat Model for a Given Application? The security requirements vary from one cooperative driving application to the other based on the application objectives. Consequently, the relevant adversaries to defend against, and the impact of a given attack on the target vehicle also vary from one application to the other. For instance, an eavesdropping attack may be considered unimportant for cooperative collision detection application. However, for a routing service application it may be paramount to protect private navigation data of the target vehicles from an unauthorized entity. While it is important to consider a realistic threat model that can account for the relevant attack orchestrations, an all-powerful adversary with limitless capabilities cannot be defended against by any security solution. For instance, consider a CAV application where a vehicle computes driving decisions based on the sensory data specifying the states (position, velocity, acceleration, etc.) of all the vehicles in the vicinity. If an adversary collusively corrupts all the sensory data in a way that all kinematics equations remain valid but the values are different from ground truth, then it is impossible for the victim vehicle to determine if the values it receives are ground reality or corrupted. For instance, an multi-channel adversary could replace the velocity of the preceding vehicle with a different value while adjusting the position and acceleration accordingly so that the laws of kinematics are satisfied. Such an adversary is clearly "all powerful" in the sense that it is impossible to defend against. Therefore it becomes essential to strike the right balance between identifying a threat model that is practical while also accounting for the most relevant adversaries compromising the application objectives.

How Can We Make Sure that the Resiliency Is Viable? A viable solution should address the diverse spectrum of attacks on CAVs, while obeying the safety requirements and automotive platform constraints including limited computational resources, strict timing requirements, stringent cost and time to market constraints, etc. At the same time, it is required that the security solution should be capable of handling unknown attack scenarios.

How Can We Validate the Solution? Testing and validation are crucial in developing a security solution for CAV systems. However, real-world testing is not a feasible option due to road safety concerns. This may require the use of simulation environments for validation instead. Unfortunately, most automotive simulators available for the research community are not sophisticated enough to provide a flexible simulation environment to validate the solution in realistic attack scenarios. The diverse and evolving attack spectrum further complicates the validation process.

1.2 REDEM*: Vision for ML-Based Resiliency*

In recent work [6], we have put forward a generic approach, which we call REDEM (for "Real-time Detection and Mitigation"), to address real-time resiliency requirements in CAV applications to protect adversaries compromising perception inputs. REDEM is not a specific architecture: after all, note from above that a resiliency solution must be customized for different CAV applications and different target adversaries. Instead, REDEM represents a systematic methodology for architecting, tuning, and validating a resiliency solution. At the heart of REDEM is the idea that it is possible for a vehicle to detect adversarial actions through a machine learning model designed (and tuned) to predict normal behavior pattern when engaged in the targeted CAV application. The REDEM infrastructure includes a configurable, flexible "architectural skeleton" (described below) to realize this vision, together with recipes for (1) configuring the skeleton into an architectural solution for a given CAV application against a specific adversary model, and (2) providing a comprehensive validation of such architectures.

1.3 Overview of the Chapter

In this chapter, we provide the vision of REDEM and its realization in a fundamental but representative CAV application, Cooperative Adaptive Cruise Control (CACC). Our goal for this chapter is not necessarily to advocate REDEM as an instrument for designing CAV resiliency. Furthermore, we eschew rehashing technical results about the various REDEM incarnations, except as necessary for the completeness of the chapter or to explain the intuition behind a specific design choice; the readers interested in a more technical treatment of the REDEM architecture for

specific CAV applications and their validation are referred to previous publications on the subject [7, 9–11]. Instead, in this chapter we endeavor to elucidate the thinking behind the many architectural decisions, challenges encountered, and the approaches taken to address them. We believe the lessons from REDEM could carry over to other applications of ML targeting real-time security resiliency in various critical infrastructures, particularly under computational resource limitations.

The remainder of the chapter is organized as follows. Section 2 introduces the high-level design of REDEM and explains the relevance (and requirements) of ML-based resiliency for CAV applications. Sections 3 and 4 present a variety of challenges involved in making such a solution work, and REDEM's approach to address these challenges. In Sect. 5 we demonstrate the efficacy of REDEM in an illustrative, foundational CAV application. We conclude in Sect. 6.

2 REDEM Basics

At the level of usage, REDEM can be envisioned as a vehicular service for connected vehicles. A vehicle can subscribe to the service as long as it includes a certain on-board architecture for ML-based anomaly detection described below. Figure 1 shows the overall setup of REDEM. We refer to the subscribing vehicle as the *ego vehicle*,"$\mathcal{E}$", and all of REDEM analysis is done from the point of view of this vehicle. Data from all subscribing vehicles is periodically uploaded to a trusted cloud server for progressively refining ML models used by the on-board hardware; $\mathcal{E}$ periodically updates the on-board system by downloading the latest ML models. The communication with cloud is performed when $\mathcal{E}$ is connected to Internet through a

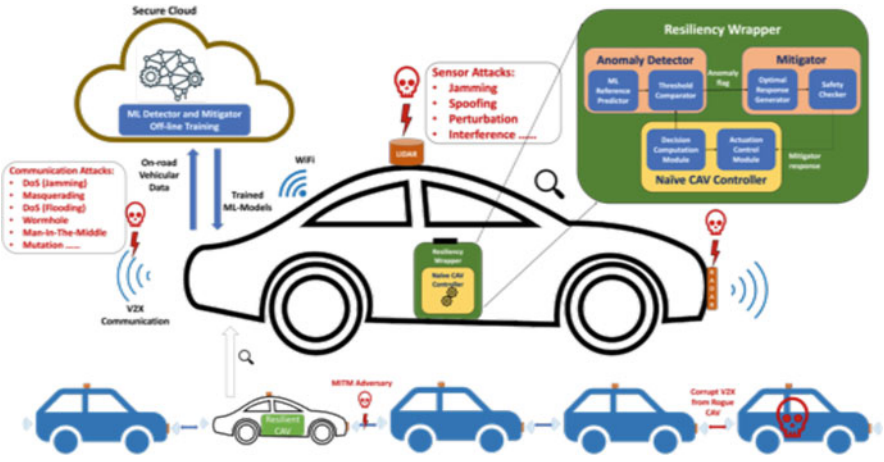

Fig. 1 REDEM-augmented CAV engaging in cooperative autonomous driving with neighbouring CAVs

trusted network, e.g., when stationary at the owner's residence; on-road connectivity with cloud is not necessary. During driving operations, the on-board hardware automatically detects anomalies using the trained ML model installed in $\mathcal{E}$, and performs mitigation.

2.1 Architecture

The key insight behind REDEM on-board design is that the architecture of most CAV applications follow a standard template with two major components: (1) *Decision Computation Module* and (2) *Actuation Control Module*. Given the sensory and V2X inputs pertaining to the application, Decision Computation Module computes the desired actions of the vehicle, and Actuation Control Module generates the control commands for the actuators. Correspondingly, REDEM augments this template with the following two resiliency components to defend against adversarial attacks.

1. *Anomaly Detector* is responsible for detecting suspicious communication or sensory inputs.
2. *Mitigator* is responsible for applying the appropriate alternate action to the vehicle in response to a detected anomaly.

The role of ML in REDEM is in the design of the Anomaly Detector and Mitigator components. More precisely, anomaly detection is implemented through deployment of an ML-based *predictor* model that is trained to learn the normal behavior of Decision Computation Module. The output of Predictor is compared against the (real) output Decision Computation Module. A deviation beyond a predefined threshold is classified as an anomaly. If no anomaly is detected, the output of Decision Computation Module is applied to the vehicle; otherwise, Mitigator is triggered.

2.2 Appropriateness of ML-Based Solution

CAV applications represent a domain where safety requirements are paramount. Given that the resiliency solution influences the driving behavior of a vehicle, safety requirements obviously extend to the resiliency solution as well. In particular, any driving decision generated from an automated source must not increase the risk of accident. This applies particularly to any system that performs real-time mitigation in response to detected anomalies: road safety should not be compromised by the mitigating action irrespective of whether the response is to an input classified as anomalous in the context of a real attack or simply due to the imprecision/inaccuracy in the detection algorithm. Given the criticality of safety requirement, it is natural to ask why one would consider ML-based solutions to address the resiliency

question in CAV applications. After all, machine learning approaches are inherently probabilistic: even the best ML solution would incur errors in some cases. Would it not be more appropriate to consider a technology that would provide a more deterministic safety guarantee?

Unfortunately, the answer is "no". To understand the reason, note that a key requirement for resiliency solutions is that they must enable protection against a spectrum of attacks. In particular, it is infeasible to have a different solution for each individual attack. Aside of the fact that the number of potential attack mechanisms already available today is prohibitively large, we can anticipate several more to be discovered during the long life time of the vehicle. Since resiliency is a design solution, it will be difficult (and sometimes impossible) to patch the design in field in response to each new attack discovered after deployment. A corollary is that the resiliency solution must be equipped with mechanisms to address the so-called *zero-day attacks*, i.e., attacks not known at design time but subsequently discovered when the vehicle is in field. To our knowledge, machine learning is one of the only few known technologies that enable potential prediction and analysis of previously unknown scenarios, based on the similarity of the new scenario with those the model has been trained for. Furthermore, the need for zero-day attack resiliency undermines any argument of deterministic (or complete) protection against the spectrum of attacks: after all, if an attack is not known at design time one cannot directly guarantee that the resiliency solution protects against that attack.

Nevertheless, it is non-trivial to actually create a practically viable resiliency solution using the technology. In Sects. 3 and 4, we consider some of the challenges and considerations involved. For each of the challenges discussed, we briefly mention the REDEM approach to addressing the challenge. Note that the goal is not to specifically advocate the REDEM approach itself but to provide a sense of the kind of thinking that one has to carry on to make an ML-based resiliency solution viable for a safety-critical multi-agent cyber-physical application domain.

There has been significant research in developing resiliency solutions for CAVs against adversaries compromising the perception systems. In addition to machine learning approaches, there has been work on control-theoretic approaches for detecting attacks on CAV applications [1, 13, 17]. Control-theoretic solutions enable a more deterministic analysis than machine learning. However, these solutions indeed suffer from the problem of being point solutions to specific vulnerabilities alone. For instance, control-theoretic solutions proposed to defend Cooperative Adaptive Cruise Control against Denial-of-Service attacks on V2X communications or spoofing attacks on sensor systems are tightly coupled to specific attack mechanisms. Consequently, an adversary can easily evade these protections by tweaking the attack mechanisms to break the assumptions made in the solution design. Correspondingly, while ML-based solutions have been devised before to detect anomalies in cooperative connected vehicle applications [4, 14], they did not account for *real-time resiliency*. Rather, these techniques are used to detect a compromised execution off-line through post-analysis of the communications or sensory inputs provided to the vehicle vis-a-vis ground truth.

3 Architectural Considerations

Coming up with a resiliency architecture requires addressing a variety of challenges. While ML is the central component of a viable real-time resiliency, it is not the only thing. The resiliency solution must define a system that incorporates the ML prediction together with other components (e.g., the original application, mitigation, anomaly source identification, etc.). In this section, we consider the thought process behind coming up with the architecture of this overall system, the exploration challenges, and the REDEM approach for addressing them. Implementation, tuning, and validation of the ML components in particular will be discussed in Sect. 4.

3.1 Small Data Problem

The efficacy of any ML-based system depends upon the availability of high-quality data. So a critical question task is: how do we get copious high-quality data necessary to make the ML-based predictions viable? Note that in traditional applications of ML (e.g., recommendation systems) this problem is addressed simply by collecting data for a longer duration. Unfortunately, that does not work for a domain like cyber-security, since finding one (or a few) security vulnerabilities generally triggers a mitigation response (possibly through patching, point fixes, or sometimes design overhaul) resulting in the previous vulnerabilities being obsolete and possibly making ways for newer attacks and compromises. The lack of data represents a vexing problem in security and is known to be a bottleneck in the application of ML in cyber-security solutions.

To address the small data problem in REDEM, our key insight is that while the data on security attacks is indeed limited, normal behavior data is in fact plentiful. Furthermore, normal behavior data follows the typical characteristic of standard ML domains: more data can be obtained by simply collecting data for a longer duration. REDEM makes use of this observation by defining the resiliency problem in terms of *anomaly detection* (capturing deviations from normal behavior), rather than as *classification* (categorizing an input into normal or attack classes). The formulation as anomaly detection implies that the ML models need to be trained to predict only the *normal behavior*; attack or adversarial data is not necessary. Furthermore, data collector in REDEM enables progressive improvement of the ML model through continuous real-world data collection.

However, the small data problem does have repercussions on parameter tuning and validation, which must be done before the application is deployed in field. We discuss those challenges in Sect. 4.

3.2 Resource Constraints

Vehicular systems are resource-constrained in terms of performance and power. Although automotive systems can be considered relatively high-performance compared to many other Internet-of-Things devices, operations with high computational complexity are infeasible. The situation is exacerbated in the case of resiliency solutions because of the need for real-time response: if the response of a resiliency solution depends on the result of a computation, viability of the solution relies on the feasibility of carrying out the computation within limited computing resources under a tight upper bound on time. Indeed, resource constraints preclude traditional hardware security mitigations such as high-overhead cryptography-based approaches or authentication techniques. Resource limitations affect the choice of ML as a resiliency solution as well, given the computational needs of ML.

Addressing the resource limitation problem in REDEM requires a more careful dissection of the source of computational overhead in ML. Roughly, there are two sources of computational overhead in ML-based systems. First is the cost of training through a substantially large set of examples to ensure sufficient prediction accuracy. Second is the cost of inference (or prediction) of an input in field as normal or anomalous. The way REDEM architecture ameliorates the training cost is to separate the training from in-field inference. Training in REDEM is performed offline in the cloud, and no real-time communication is required with the trained model during prediction in-field: the trained model is downloaded periodically and deployed into the on-board architecture. Optimizing inference cost is more tricky. Inference has to be done in real time using in-vehicle electronics: reliance on a cloud-based infrastructure for this activity would result in a requirement of continuous connectivity which may not be viable for various terrains and geographical regions. Reduction of inference cost therefore requires reducing the complexity of the ML model itself: the more elaborate the model, the more likely that the inference entails increasingly sophisticated computation and consequently higher inference cost. On the other hand, prediction accuracy does require the ML model to be sufficiently elaborate both in terms of the sophistication of the underlying algorithm and in the number of features/parameters incorporated in the model. REDEM addresses this conundrum by making the trade-off between cost and accuracy explicit and providing the user the ability to tune their model to customize for the trade-off target for the application. The framework itself is agnostic to the specifics of the underlying ML model. Rather, the user chooses a target for prediction accuracy (in terms of metrics like precision, recall, and f1-score) and can select the simplest ML model that addresses that accuracy need.

3.3 Multi-Channel Adversary

In typical CAV applications, more than one perception channel can be compromised. For instance, in CACC application, there are three perception channels that correspond to the velocity, position, and acceleration of the preceding vehicle. Generally acceleration is communicated through V2X messages and velocity and position data computed by the follower vehicle through its on-board sensors. Different adversary models would be interesting for different implementations or even a specific instance of the application, e.g., it may be appropriate for some CACC implementation to consider an adversary to corrupt only acceleration information (V2X corruption), or velocity and position information (corruption of sensor data), or some combination thereof.[1] When an adversary can corrupt multiple channels, one crucial requirement for ML-based resiliency is *source identification*, i.e., determining which channels are "actually" corrupted. Considering the CACC example above, suppose the adversary actually corrupts the acceleration information. From the perspective of the following vehicle, however, all that can be perceived is that pattern of acceleration values and velocity/position values received from the preceding vehicle are mutually inconsistent based on standard kinematics equations. Without some contextual information about the environment (e.g., what acceleration values are feasible under a specific driving condition), it is not possible to derive which ones the acceleration or velocity/position channels correspond to ground truth and which ones are anomalous. Furthermore, if we want to enumerate all subsets of potentially compromised channels, we will quickly run into combinatorial explosion. For instance, consider a platooning scenario consisting of five non-lead vehicles following a leader to create a platoon string. Suppose each vehicle receives three inputs (e.g., position, velocity, acceleration) from the leader and the vehicle immediately preceding it in the platoon. Assuming that at most three of the six inputs each non-lead vehicle receives can be corrupted by an adversary, there are 15 possibly compromised channels in the platoon at any point. Consequently, the total number of possible subsets of corrupt channels will be 2^{15}. Clearly, a naive approach of systematically examining each subset of channels for possible anomalies would be computationally prohibitive.

REDEM addresses the problem of multi-channel adversaries through a process of source identification that exploits *selective sensitivity*. The key insight is that the same anomaly can affect behavior of different functions in different ways. For instance, an anomalous value of a preceding vehicle acceleration would not affect a machine learning model in the following vehicle that is trained to predict based on only the values of the velocity and position of the preceding vehicle. REDEM source identification creates a number of ML models with selective sensitivity to different

[1] When doing this, care has to be taken so that we are still considering an adversary against whom it is possible to have a viable defense, e.g., if the adversary can collusively corrupt all the perception channels of the ego vehicle it is easy to see that no resiliency solution is possible.

parameters which can then be used cumulatively to narrow down the number of adversarial channels.

3.4 Error Control and Recoverability

The error control and recovery challenges arise from the imperfections in ML-based systems. The ideal case for a CAV resiliency is that the resilient system, when provided with any input whether benign or malicious, **always** behaves the way that the application is targeted to behave when all inputs receive ground truth, i.e., with no adversarial action. However, since ML techniques can only provide accuracy with a certain probability, it is important for any ML-based resiliency solution to account for the situations when ML would perform misprediction. There are two different ways in which the misprediction can impact the application. One is the *direct* way, where the impact would be a risk to safety or efficiency of the application. Another, more subtle way is the *indirect* effect on the vehicle state after the attack is completed. Under the latter, consider a platooning application where a vehicle computes its acceleration at each instance based on the acceleration, velocity, and position of the preceding vehicle. Consider an attack in which the position and velocity channels are collusively corrupted, i.e., the values of these channels are changed such that the kinematics equations are satisfied. The upshot of this attack will be that the victim vehicle would receive values of velocity and position that are mutually consistent, but inconsistent with the acceleration values. By analysis of the inconsistency alone, the victim vehicle would have no reason to deduce a vel-pos attack instead of the acceleration attack. (Indeed, if a vehicle does in fact deduce this then it would likely mis-predict the complementary scenario where acceleration is the channel being corrupted and vel-pos channels provide the ground truth.) A good mitigation would likely be conservative and ensure safe operation irrespective of the channels corrupted. Nevertheless, the vehicle's "perception" of its environment would be different depending on whether it correctly identifies the source of corruption. Furthermore, if the source identification is erroneous, it is possible that a subsequent benign (ground-truth) input would then be deduced as malicious. In the platooning example, after having wrongly deduced that the acceleration value received is corrupted and the velocity-position values are ground truths, the resiliency system would have a perception of the preceding vehicle's state velocity, position, and acceleration which is different from reality. Consequently, when it receives benign (ground truth) values of these three parameters it might wrongly consider (any subset of) them anomalous.

From the discussion above, we see that a resiliency solution must have the property of *recoverability*, i.e., the ability to return to a state in which when it is provided benign inputs when its perception of the environment is not too far from ground truth. We also observe that in particular with multi-channel adversary, recoverability may be difficult to ensure.

REDEM introduces a variety of techniques to reduce inaccuracies for solutions abating the errors resulting from the imperfections in ML systems. This includes additional rule-based validation steps in each decision-making cycle, which can control errors from previous time steps to propagate and accumulate. In addition to checks, REDEM "exploits" adversary assumptions to address recoverability issues, e.g., the adversaries handled by REDEM are constrained in terms of the degree of bias they can introduce during a corruption at different time steps and the number of channels that can be corrupted at the same time. This permits REDEM solutions to correct mistakes, e.g., the scenario where velocity and position are continually corrupted can be precluded by the requirement that an adversary during one episode of continuous attack can only select one untrusted channel. We put the word "exploits" in quote, since in practice a resiliency solution clearly cannot get to choose the adversaries against which to defend; the quality and power of the adversary ought to be defined by the characteristics of the application and deployment. Nevertheless, as we argued in Sect. 1.1, it is impossible (in principle and practice) to develop resiliency against an adversary that is all powerful. Consequently, it is fair to constrain the set of adversaries that can be handled by a specific resiliency solution. Nevertheless, we must still ensure that the adversary is realistic, i.e., it is worth developing a resiliency solution to focus specifically on the adversary for a threat model. For each incarnation of REDEM for different applications, we define a threat model constraining adversary power and argue why it is a realistic adversary.

4 Design, Implementation, Tuning, and Validation of ML Component

The considerations discussed in the preceding section pertained to the design of the overall resiliency system. In this section we delve a bit more into the ML component of the system. Some representative questions we need to address here include: (1) *Which ML architecture should we choose?* (2) *How should we train it?* (3) *Where do we find valid data to train it?* (4) *How can we perform validation?* We discuss some of these issues here, and the methodologies developed in REDEM to address them.

4.1 Architecture Selection and Tuning

One of the key activities to enable ML application is to identify the appropriate ML model for the task and determine its parameters. In case of ML-based resiliency, selection and tuning of ML model incurs several interesting challenges. First, the complexity of the ML model itself is constrained by the available computation

resource as discussed in Sect. 3.2. Second, accuracy of different ML models depends on the specifics of the scenario, e.g., a model may be more accurate under benign conditions or a specific type of corruption. Navigating this space of models to identify the accurate one for a target application is highly challenging. The small data problem discussed in Sect. 3.1 exacerbates the problem: given the very low amount of data available, it is difficult for any ML model to learn the features to make it generalizable for all target applications, with a real danger of over-fitting.

To address this problem, a key insight is that the goal of an ML-based resiliency is to ideally behave like the naive application (i.e., application with no resiliency introduced) when provided data corresponding to the ground reality. In other words, the efficiency (and accuracy) of the solution would be determined primarily by the prediction accuracy under benign scenario. With that in mind, the following steps provide a recipe for selecting the ML architecture.

1. Identify a set of candidate ML architectures that can be deployed under the resource constraints. The constraints preclude overtly complicated ML systems, and generally permit only a small set of simple candidate architectures. It is generally possible to effectively navigate the space of architectures left through quick sampling.
2. Train the candidate architectures with benign data, discarding ones with unacceptable prediction accuracy under benign conditions.
3. Among the architectures with acceptable prediction accuracy under benign scenarios, select the one with the highest prediction accuracy under malicious conditions.

Obviously, the above steps should be used as a guideline, not a procedure cast in stone. For instance, determining the accuracy entails tuning the right set of hyperparameters, which in turn requires trade-offs between time, cost, computation capability, and many others.

4.2 Data Preprocessing and Feature Selection

CAV application anomalies are contextual, i.e., determining whether a specific input is anomalous requires understanding of the driving environment. For instance, a speed of 70 Mph is normal in a rural highway during a clear summer evening but perhaps not in a snowy winter morning or during rush hours near a big city. To be able to accurately qualify some input as normal or anomalous, the ML system ideally must have access to a large quantity of fine-grained data (sampled at high frequency), together with sufficient context. Unfortunately, real-world datasets are generally incomplete and inadequate. For instance, HighD Dataset [15], which provides trajectory data corresponding to real vehicles driving in German highways, has individual vehicle trajectory data encompassing approximately 15 s. Data from real datasets also include noise and inconsistency, arising from errors in collection and measurement from the physical environment. One can augment this with

synthetic data, collected from a variety of simulators. However, for synthetic data it is critical to ensure that data characteristics are consistent with what the application is expected to encounter in field. Note that the accuracy of ML can be harmed by having irrelevant features and lack of diversity in the data. Insufficient diversity in data may lead to overfitting. Poor feature selection may also affect generalizability. So, feature selection and feature engineering must be done, and it completely depends upon the purpose of the ML model used in the CAV application resiliency.

Data preprocessing is the idea of preparing the raw data to make it suitable for consumption by ML algorithms. This includes several components. Data *cleaning* entails filling in missing values, smoothing or removing noisy data and outliers, and resolving inconsistencies. Data *integration* involves integrating data from multiple sources such as databases, data cubes, files, etc. To solve the complexities arising due to feature selection, the input features should be selected such that they have actual impact on the learning behavior. In such cases redundant features and features having no importance for the prediction objective of the ML target in question should be ignored while training to boost up the performance. At the same time new features can be engineered from existing features in order to get better results.

REDEM addresses the data preprocessing issues by using "realistic synthetic data". In particular, REDEM uses a *physical* simulator platform, RDS1000® [16]. This platform can be used to acquire data as follows. The system permits configuration of various different driving environment, and an immersive environment for a human to have the experience of driving in the programmed environment. We can record the actions of humans as they perform driving, and use that as a proxy for what a vehicle does under similar situation. It also includes autonomous driving modules that can be used to study reaction of autonomous driving algorithms under similar situations. We curated an extensive dataset of vehicular behavior under 24 different driving conditions by first acquiring the data and then performing cleaning, reformatting and validation. This setup can produce fine-grained and real-time data. However, it leaves open the issue whether the environments programmed (and the vehicular behavior recorded) do in fact correspond to reality. To address this question, we show that the real dataset snippets in fact match in pattern with our curated dataset. The real dataset only includes short snippets of time as mentioned above. Nevertheless, if these snippets match the synthetic data for the corresponding environment, we can gain confidence that the synthetic data is indeed realistic.

Finally, note the apparent dichotomy in the discussion above and the discussion in Sect. 3.1. We argued that normal behavior data is in fact plentiful, and it is the anomaly data that is limited. The discussion here at cursory glance would appear in contradiction to that statement, *is that right*?

Actually, it is wrong: there is no contradiction. Normal behavior data is in fact plentiful once the application is deployed and can collect such data in field. However, when determining the parameter set and performing feature engineering, the application is not yet deployed in field, so we have to depend on synthetic data or available real-vehicle datasets.

4.3 Decision Threshold Selection

Given that ML in CAV resiliency is targeted specifically for anomaly detection, an important issue is to determine the anomaly threshold. A high threshold may result in reduced detection accuracy, whereas a low threshold may result in more false reports in detection. An ideal threshold is one that would provide both safety and efficiency under adversarial scenarios while incurring minimal performance overhead in benign conditions. A more subtle impact of the choice of the threshold is the robustness of the resiliency system to *subversion attacks*. The idea of subversion attacks is for the adversary to create anomalous data that is nevertheless accepted as normal by the detector, thereby bypassing any mitigation against the attack. A high anomaly threshold can make the CAV application vulnerable to subversion attacks, impacting the safety, efficiency, and recoverability of the resiliency solution.

To address this problem, REDEM includes systematic methodology for identifying and tuning anomaly threshold. REDEM accounts for the fact that the choice of threshold also depends on the operating environment and may require reconfiguration as the driving conditions change during the application engagement. In REDEM methodology, the threshold is determined by analyzing the distribution of test-set error incurred by the ML prediction model under benign conditions as well as a finite set of representative attacks. This is achieved in two steps. First, the threshold is coarsely tuned to minimize false positives and false negatives under benign conditions determining a ball-park range. Subsequently, a series of special subversion attacks are orchestrated to fine-tune the threshold that can balance the trade-off between the conflicting design goals of minimizing inference cost, achieving required detection accuracy, and minimizing overhead due to false-positives/false-negatives. While achieving the ideal outcome for all the design goals simultaneously is impractical, REDEM identifies the tolerable imperfections that can still ensure overall resiliency (guaranteed safety and optimal efficiency at all times) for CAVs. The threshold selection then accounts for the re-defined practical design goals carefully allowing for a small amount of inference cost, false-positives, and false-negatives, while achieving the required detection accuracy.

4.4 Validation

Validation is crucial for a CAV resiliency system, since it targets highly safety-critical applications. However, the unique nature of ML-based resiliency makes it highly challenging to achieve effective validation. Obviously, validation is a broad topic with many different facets. Here we provide a very quick summary of the challenges involved, and our approach to address these challenges. The reader interested in a fuller discussion of validation in REDEM is referred to our companion publication [9] that provides an exclusive treatment of the subject.

Roughly, validation of ML-based prediction involves addressing three critical problems as described below.

1. **Inadequacy of data.** The challenge with data for validation is similar to that for parameter estimation and tuning discussed in Sect. 4.2: before the application is deployed, how can we obtain copious amount of realistic data to validate an ML system?
2. **Validation against zero-day attacks.** This issue arises from the fact that it is inadequate for a resiliency solution to only provide protection against a specific set of known attacks. New attacks not considered during resiliency design can become feasible during the life-time of the application after technology (and hence sophistication of attack) advances in ways not necessarily anticipated at deployment. This results in a conundrum for security validation: how can we ensure that the resiliency system is indeed effective, not only against known attacks but against a spectrum of attacks that are unknown at deployment time?
3. **Validation challenges for an inherently probabilistic system.** This challenge is the verification counterpart of the design challenge we discussed in Sect. 3.4. Since no ML system is accurate in 100% of cases, we must be able to verify that, either (1) no matter what attack is instigated, the victim vehicle's perception is always within tolerable limits of reality; or (2) if the perception of the victim vehicle deviates significantly from reality then its response still ensures safe and efficient operation, and after the attack is over it eventually returns to a state in which benign inputs are treated as benign.

REDEM addresses the first two problems discussed above through new validation techniques. The third problem (probabilistic system challenge) is relegated to design (and validation) of resiliency solutions with the property of recoverability as discussed in Sect. 3.4. We address the problem of data for validation in the same way we did for model training, e.g., by creation of realistic synthetic data from a physical simulator. The uniqueness of REDEM validation is in how it addresses the second problem, i.e., validating resiliency against unknown adversary. The key insight is that it is possible to develop a resiliency system that accounts for attacks based on its manifestation features, stealth, and impact rather than detailed attack mechanism. Furthermore, it is possible to comprehensively classify the spectrum of attacks in this manner simply from the threat model. For instance, consider a CACC application where a vehicle follows its preceding vehicle by maintaining a specific time headway. If the adversary is confined to V2V communications, the only choices for the adversary are to (1) mutate an existing message, (2) fabricate a new message, and (3) prevent the delivery of a message. Going through this argument enables us to create a taxonomy of V2X attacks. Note that if our validation covers attack space defined by the taxonomy then the above argument suggests that we indeed comprehensively cover the space of all attacks defined by the threat model, including unknown attacks. REDEM additionally includes an automated evaluation framework for systematically generating attacks from the adversary taxonomy [8].

5 REDEM Case Study: Resilient Cooperative Adaptive Cruise Control

We have instantiated REDEM to incorporate resiliency on platooning applications [6, 7, 10]. Here we quickly summarize the instantiation of REDEM architecture on Cooperative Adaptive Cruise Control (CACC). We also present a summary of results showing the overall efficacy of the REDEM resiliency for CACC.

5.1 CACC Overview

In CACC, the following vehicle autonomously adapts its velocity in accordance to the acceleration of the vehicle in front (received through V2V communication), as well as the relative velocity and inter-vehicle gap (obtained from the ranging sensor readings). CACC enables improved road safety and efficiency (e.g., a much smaller headway) compared to its non-cooperative counterpart, Adaptive Cruise Control (ACC). Figure 2a depicts vehicles engaged in CACC. Figure 2b demonstrates the high-level functionality of a CACC decision computation module that implements constant time headway policy. Following vehicle receives the preceding vehicle's instantaneous acceleration as a V2V message. It utilizes this information in addition to the on-board ranging sensor readings providing the relative position and velocity of the preceding vehicle. Consequently, the following vehicle efficiently adapts its velocity in accordance with the acceleration of the preceding vehicle achieving improved efficiency and safety. CACC forms the basis for several connected car applications such as multi-vehicle platooning, cooperative on-ramp merging, etc.

A vehicle engaging in CACC can be exploited by an adversary that is capable of manipulating the V2V communication or a malicious preceding vehicle that shares false information. For instance, a malicious preceding vehicle can report a fake acceleration value that is greater than its true value. This can mislead the vehicle to accelerate at a higher value than desired that can lead to an increased risk of a collision. Similarly, a Man-In-The-Middle (MITM) adversary can mutate the messages from the preceding vehicle by adding a negative bias. The vehicle receives false acceleration value and fails to maintain the optimal space gap. This leads to loss in efficiency or can cause string instability in the traffic.

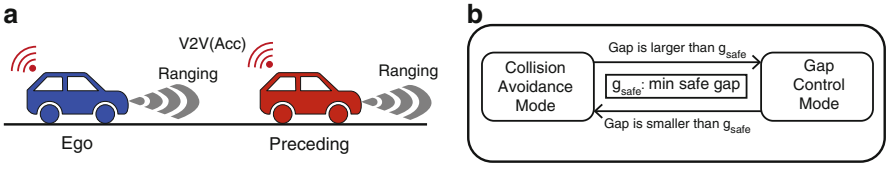

Fig. 2 (**a**) Two vehicles engaged in CACC; (**b**) Modes of operation of a conventional CACC decision computation module

5.2 *Evaluation of* REDEM *Resiliency on CACC*

We extended CACC with REDEM architecture to generate a resilient CACC solution which we call RACCON (for "Resilient Cooperative Adaptive Cruise Control). Our evaluation of RACCON also represents one of the most comprehensive resiliency evaluations performed on a connected vehicle application up to date. As pointed out in Sects. 3 and 4, this means consideration of a number of different factors. In summary, the evaluation of RACCON included the following components.

1. **Data Validation:** We validated that the vehicular driving patterns reflected in our simulation data conform to real-world patterns from a public dataset.
2. **Identification of Appropriate ML Model:** We developed a systematic evaluation methodology for identifying and tuning the optimal ML architecture.
3. **Attack Impact Analysis:** The viability of attack orchestration framework for RACCON evaluation depends on the quality of the orchestrated attacks themselves. We developed a methodology to analyze attacks, in terms of stealth and impact.
4. **Anomaly Detection Threshold:** A key factor in the effectiveness of RACCON is the identification of *anomaly threshold*, i.e., the extent of deviation from normal behavior pattern that would be classified as a potential threat. Selecting an appropriate threshold involves balancing the trade-off between maximizing attack detection accuracy and minimizing false alarms. We present a series of experiments to compute the optimal threshold, achieving the balance between maximizing attack detection accuracy and minimizing false alarms.
5. **V2V Attack Resiliency:** The central component of our evaluation shows the robustness of RACCON against various V2V attacks.
6. **Resiliency Against Detector Subversion:** We designed a set of experiments to address evaluating the robustness of RACCON against detector subversion, and tune anomaly threshold accounting for the trade-off between robustness to subversion and minimizing false alarms.

Here we show some representative plots from our experiments to give a flavor of the evaluation and the extent and quality of REDEM resiliency for CACC. Figure 3 shows a representative plot for the scenario ⟨highway, windy, day⟩, for discrete, cluster, and continuous attacks. The frequency of the malicious activity and the magnitude of deviation between the false V2V message received and the ground truth, determine the stealth of the attack. The detection system is capable of capturing attacks of varying stealth as can be seen from the figure. Figures 4, 5, and 6 show the conclusions from our experiments on Mitigator efficacy under collision-causing, efficiency-degrading, and delivery prevention attacks. Under each category, different types of attacks for discrete, cluster, and continuous adversaries are simulated. Mitigation guarantees safety while keeping the efficiency optimal. This is reflected in the time headway values achieved by the mitigation that closely resemble the ideal values.

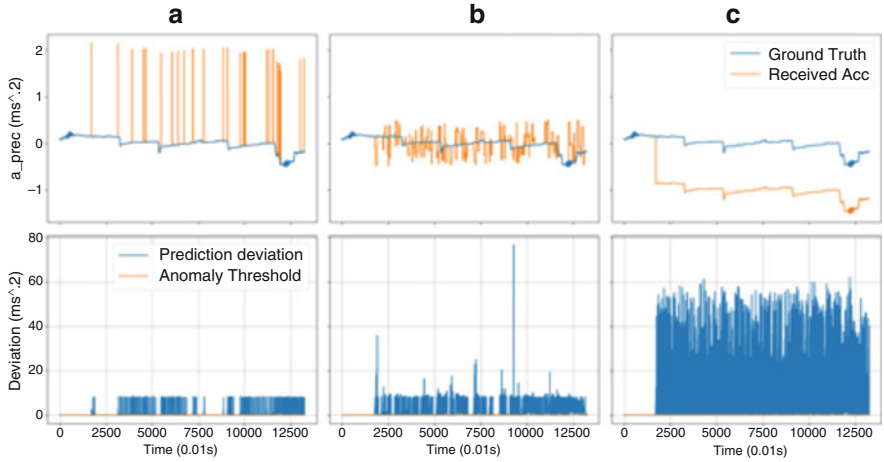

Fig. 3 Anomaly detection analysis: (**a**) Discrete attack (constant bias +2.0); (**b**) Cluster attack (bias [−0.5, +0.5]); (**c**) Continuous attack (constant bias −1.0)

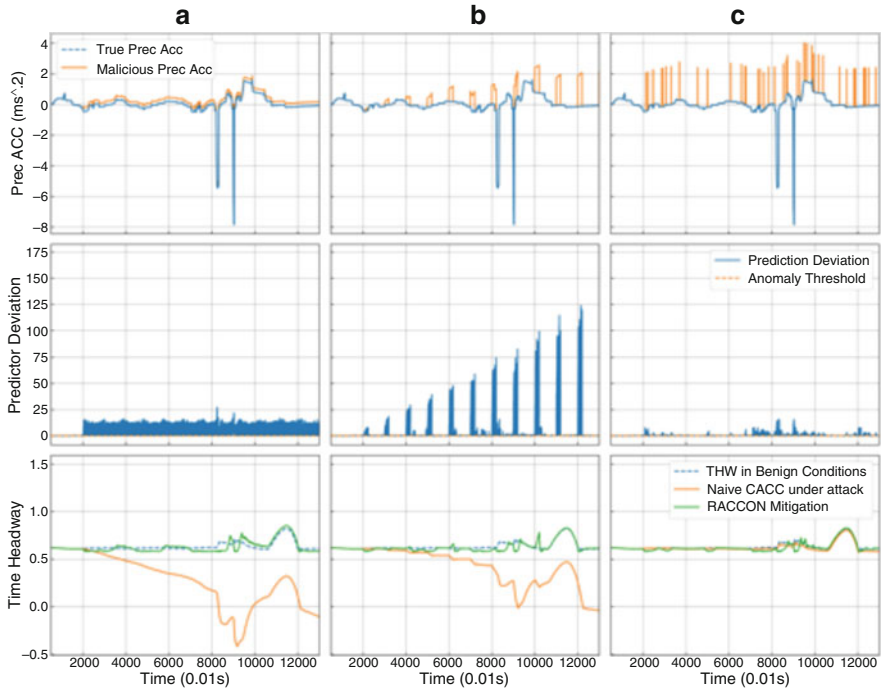

Fig. 4 Evaluation under collision attacks: Comparison of resultant time headway for resiliency augmented CACC and naive CACC with no resiliency; (**a**) Continuous attack (constant bias +0.25); (**b**) Cluster attack (linear bias +0.1t); (**c**) Discrete attack (constant bias +2.5)

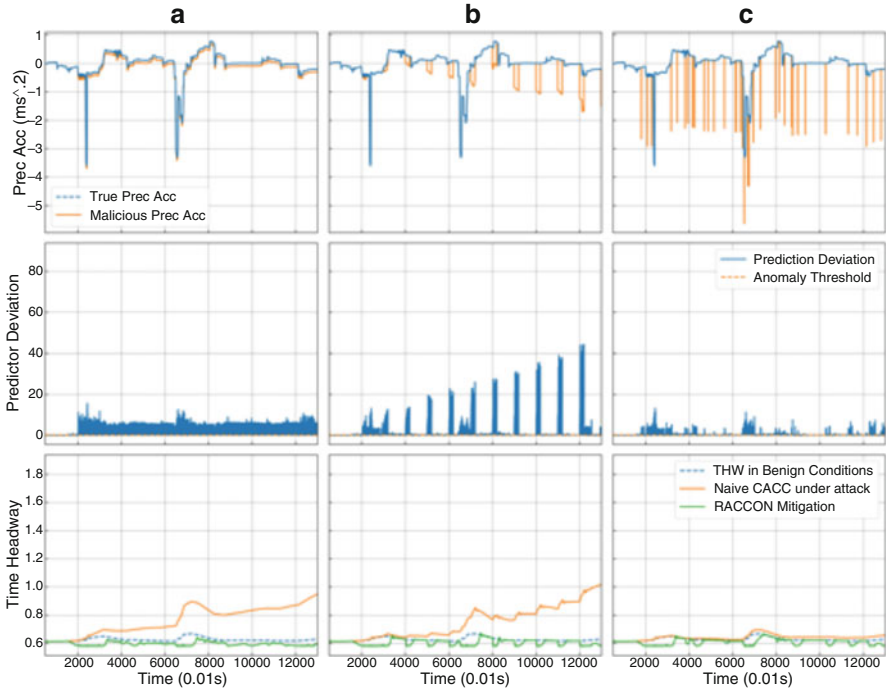

Fig. 5 Evaluation under efficiency degradation attacks: comparison of resultant time headway for resiliency augmented CACC and naive CACC with no resiliency; (**a**) Continuous attack (constant bias -0.1); (**b**) Cluster attack (linear bias $-0.06t$); (**c**) Discrete attack (constant bias -2.5)

6 Conclusion

We have considered the problem of introducing resiliency in CAV applications against attacks on perception inputs. With increasing proliferation of connectivity and autonomy in vehicles, perception inputs can create a large and highly vulnerable attack surface that can be easily exploited with catastrophic consequences. We discussed the promise and challenges in adopting ML-based solutions to achieve resiliency in this domain. We also discussed one effective framework, REDEM, to achieve this resiliency, and explained REDEM's approach to address the various challenges in system design, architecture, and validation. The efficacy of REDEM was demonstrated in Cooperative Adaptive Cruise Control application.

REDEM is very much a work in progress, and is under active development. What we presented is representative of our thinking at the time of this writing, but the thinking will inevitably evolve as we extend REDEM for newer applications. Indeed, it is important to try REDEM on applications of diverse flavors to identify weakness in the current line of thinking and determining how to expand the methodology to incorporate new challenges. Some critical applications that can provide such

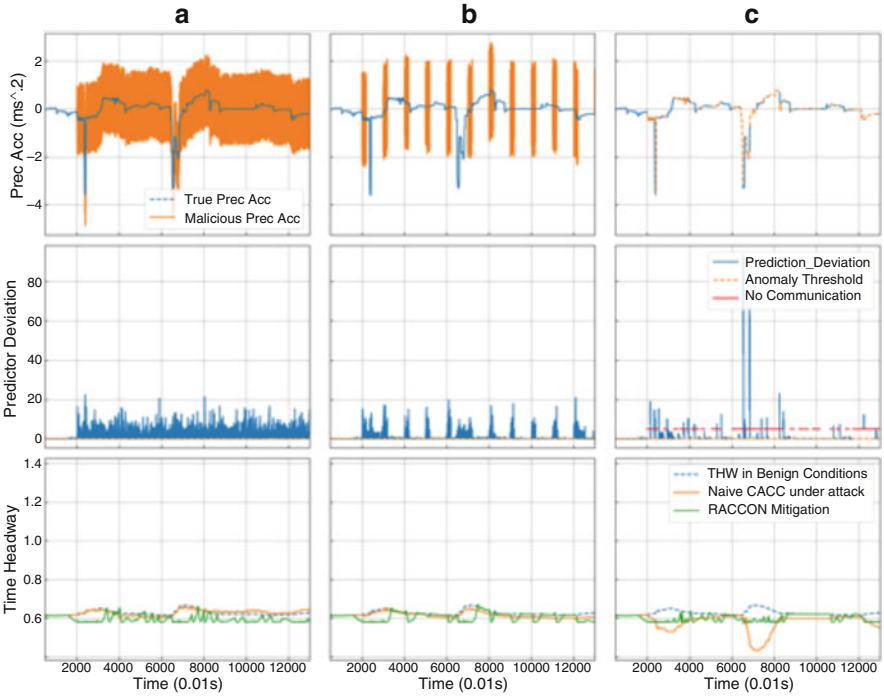

Fig. 6 Evaluation under random mutation and delivery prevention attacks: comparison of resultant time headway for resiliency augmented CACC and naive CACC with no resiliency; (**a**) Continuous attack (random bias −1.5, 1.5); (**b**) Cluster attack (random bias −2.0, 2.0); (**c**) Intermittent communication

challenges include Distributed Cooperative Collision Detection, Cooperative Route Management, etc. These applications are different from the current platooning applications in that they involve communication of perceived environment in addition to the state of the communicating vehicle. We will explore security challenges in perception of such scenarios and investigate the applicability of REDEM.

References

1. Abdollahi Biron, Z., Dey, S., Pisu, P.: Real-time detection and estimation of denial of service attack in connected vehicle systems. IEEE Trans. Intell. Transp. Syst. **19**(12), 3983–3902 (2018)
2. Adler, J.L., Blue, V.J.: A cooperative multi-agent transportation management and route guidance system. Transp. Res. C Emerg. Technol. **10**(5–6), 433–454 (2002)
3. Adler, J.L., Satapathy, G., Manikonda, V., Bowles, B., Blue, V.J.: A multi-agent approach to cooperative traffic management and route guidance. Transp. Res. B Methodol. **39**(4), 297–318 (2005)

4. Alotibi, F., Abdelhakim, M.: Anomaly detection for cooperative adaptive cruise control in autonomous vehicles using statistical learning and kinematic model. IEEE Trans. Intell. Transp. Syst. **22**(6), 1–11 (2020)
5. Bergenhem, C., Shladover, S., Coelingh, E., Englund, C., Tsugawa, S.: Overview of platooning systems. In: Proceedings of the 19th ITS world congress, Oct 22–26, Vienna (2012)
6. Boddupalli, S., Ray, S.: Redem: real-time detection and mitigation of communication attacks in connected autonomous vehicle applications. In IFIP international Internet of Things conference, pp. 105–122. Springer (2019)
7. Boddupalli, S., Hegde, A., Ray, S.: Replace: real-time security assurance in vehicular platoons against v2v attacks. In: 2021 IEEE international intelligent transportation systems conference (ITSC), pp. 1179–1185 (2021)
8. Boddupalli, S., Chamarthi, V.S.G., Lin, C.-W., Ray, S.: CAVELIER: Automated security evaluation for connected autonomous vehicle applications. In: 25th IEEE international conference on intelligent transportation (ITSC 2022) (2022)
9. Boddupalli, S., Owoputi, R., Duan, C., Choudhury, T., Ray, S.: Resiliency in connected vehicle applications: challenges and approaches for security validation. In: Proceedings of 22nd great lakes symposium on VLSI 2022 (GLSVLSI 2022) (2022)
10. Boddupalli, S., Rao, A.S., Ray, S.: Resilient cooperative adaptive cruise control for autonomous vehicles using machine learning. IEEE Trans. Intell. Transp. Syst., **23**(9), 15655–15672 (2022)
11. Casaca, A., Katkoori, S., Ray, S., Strous, L.: Internet of Things. In: A confluence of many disciplines: second IFIP international cross-domain conference, IFIPIoT 2019, Tampa, FL, USA, October 31–November 1, 2019, Revised Selected Papers, vol. 574. Springer Nature (2020)
12. Dresner, K., Stone, P.: A multiagent approach to autonomous intersection management. J. Artif. Intell. Res. **31**, 591–656 (2008)
13. Dutta, R.G., Yu, F., Zhang, T., Hu, Y., Jin, Y.: Security for safety: a path toward building trusted autonomous vehicles. In: ICCAD (2018)
14. Jagielski, M., Jones, N., Lin, C.-W., Nita-Rotaru, C., Shiraishi, S.: Threat detection for collaborative adaptive cruise control in connected cars. In: Proceedings of the 11th ACM conference on security & privacy in wireless and mobile networks, pp. 184–189. ACM (2018)
15. Krajewski, R., Bock, J., Kloeker, L., Eckstein, L.: The highd dataset: a drone dataset of naturalistic vehicle trajectories on German highways for validation of highly automated driving systems. In: 2018 21st International conference on intelligent transportation systems (ITSC), pp. 2118–2125 (2018)
16. Realtime-Technologies. Physical automotive simulator. See https://www.faac.com/realtime-technologies/products/rds-1000-single-seat-simulator
17. van Nunen, E., et al.: Robust model predictive cooperative adaptive cruise control subject to v2v impairments. In: IEEE 20th international conference on intelligent transportation systems (ITSC) (2017)