Safeguarding Federated Learning-based Road Condition Classification

Sheng Liu
Networked Systems Security Group
KTH Royal Institute of Technology
Stockholm, Sweden
shengliu@kth.se

Panos Papadimitratos
Networked Systems Security Group
KTH Royal Institute of Technology
Stockholm, Sweden
papadim@kth.se

***Abstract*—Federated Learning (FL) has emerged as a promising solution for privacy-preserving autonomous driving, specifically camera-based Road Condition Classification (RCC) systems, harnessing distributed sensing, computing, and communication resources on board vehicles without sharing sensitive image data. However, the collaborative nature of FL-RCC frameworks introduces new vulnerabilities: Targeted Label Flipping Attacks (TLFAs), in which malicious clients (vehicles) deliberately alter their training data labels to compromise the learned model inference performance. Such attacks can, e.g., cause a vehicle to mis-classify slippery, dangerous road conditions as pristine and exceed recommended speed. However, TLFAs for FL-based RCC systems are largely missing. We address this challenge with a threefold contribution: 1) we disclose the vulnerability of existing FL-RCC systems to TLFAs; 2) we introduce a novel label-distance-based metric to precisely quantify the safety risks posed by TLFAs; and 3) we propose FLARE, a defensive mechanism leveraging neuron-wise analysis of the output layer to mitigate TLFA effects. Extensive experiments across three RCC tasks, four evaluation metrics, six baselines, and three deep learning models demonstrate both the severity of TLFAs on FL-RCC systems and the effectiveness of FLARE in mitigating the attack impact.**

***Index Terms*—federated learning, road condition classification, data poisoning attacks, transportation safety**

I. Introduction

Road Condition Classification (RCC) [1], encompassing tasks such as unevenness detection, friction estimation, and surface material identification, is important for intelligent transportation. It directly influences vehicle control, traffic safety, and passenger comfort. Based on the significant and growing deployment of on-board vehicle cameras and thanks to the rapid development of Artificial Intelligence (AI), Deep Neural Network (DNN) models can be efficiently trained and then deployed on vehicles to support RCC tasks based on images captured by onboard cameras [2].

High-performance RCC models demand extensive training data covering diverse scenarios (e.g., varying weather, road types, lighting conditions) to ensure robustness [3]. Nonetheless, escalating concerns on user privacy, e.g., GDPR[1] in Europe, CCPA[2] in United States, and PIPL[3] in China, render the traditional **centralized learning paradigm** impractical for training and updating RCC models in the long term. Moreover, adversaries can easily exploit vehicle or even passenger private details: e.g., the recently proposed GeoSpy[4] can predict the location of photos within seconds.

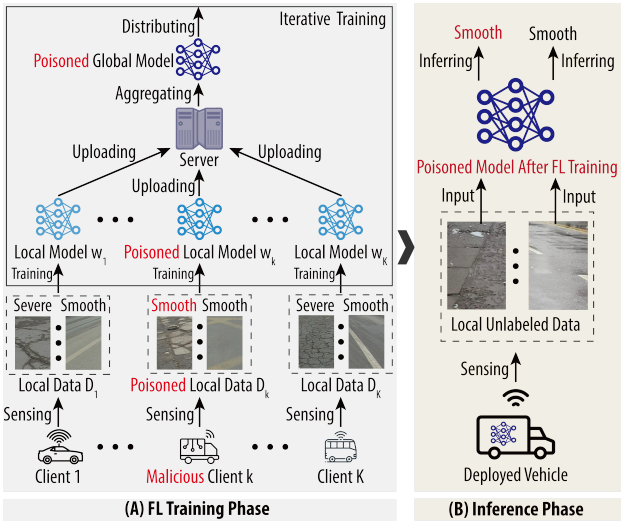

Fig. 1. Illustration of TLFAs in FL-RCC: (A) training phase poisoning global model; and (B) inference phase threatening transportation safety.

As a promising distributed solution, **Federated Learning (FL)** [4] has recently attracted significant attention from the autonomous driving community too [5], [6]. By iteratively interacting updated local model parameters and aggregated global model parameters (rather than transmitting sensitive and likely high-rate data) between clients and the server, FL can leverage both privately, locally, kept data and the massive distributed computation resources of smart, connected vehicles. This collaborative and privacy-preserving approach has recently been explored in the RCC domain [7].

However, the cooperative nature of FL introduces new vulnerabilities. **Targeted Label-Flipping Attacks (TLFAs)** [8] only require simple replacement of labels (rather than operating on image pixels or features); thus being efficient and practical to launch with compromised vehicle resources. Fig.

[1]https://gdpr-info.eu/
[2]https://www.oag.ca.gov/privacy/ccpa
[3]http://en.npc.gov.cn.cdurl.cn/2021-12/29/c_694559.htm
[4]https://geospy.ai/

1 provides an example of TLFAs, in which the adversary flips local data labels regarding road unevenness level from severe-uneven (the **source class**, i.e., the true class) to smooth (the **target class**, i.e., the falsified class) during the FL training. The resultant model may erroneously classify in the inference phase actual severe-uneven conditions as smooth; an underestimation of hazardous conditions, jeopardizing transportation safety and increasing the chance of an accident. To the best of our knowledge, no prior research has specifically addressed the vulnerability of FL-RCC systems to such attacks.

Moreover, existing studies [1] on RCC predominantly employ conventional **evaluation metrics**, such as accuracy and recall, overlooking the transportation safety ramifications of prediction errors. For instance, misclassifying the road unevenness level from severe-uneven to smooth (label distance = 2) is more dangerous than predicting labels from slight-uneven to smooth (label distance = 1); the former mis-classification could lead to inadequate or no vehicle adjustment (e.g., speed). Nevertheless, traditional metrics assign equal weights to both error types, failing in properly quantifying the TLFA influence.

Although there are countermeasures against general Data Poisoning Attacks (DPAs) [9], **defensive mechanisms** have not been tailored to TLFAs or FL-RCC systems. Moreover, the road images are inherently Non-Independent and Identically Distributed (Non-IID), as distinct spatial and temporal vehicle driving patterns lead to diverse data distributions. This intrinsic heterogeneity complicates the task of distinguishing malicious updates from benign ones, as legitimate updates can exhibit significant variance. This is why popular countermeasures that typically rely on coarse-grained features (e.g., *whole model parameters* [10] and *output layers* of DNNs [9]) cannot effectively detect poisoned models for FL-RCC. Thus, a need arises for defense mechanisms with *fine-grained features* specific to TLFAs. Several novel poisoning attack mitigation methods have been recently proposed; however, they primarily focus either on backdoor attacks [11] that change both original data and labels or untargeted attacks [12] that degrade the overall accuracy; they are not specifically designed for TLFAs.

To close this research gap, we propose a new metric based "label distance" to improve measuring the attack influence on FL-RCC. Based on a specific *neuron-wise analysis* of the output layer, we also design a new defensive mechanism, FLARE, which can 1) identify *source and target neurons* (output neurons that predict source/target classes), then filter out poisoned models during the global aggregation process; and 2) maintain a client (vehicle) blacklist, based on the number of times it was recognized as malicious, for each client for the client selection process in each round. According to our extensive evaluation based on three RCC tasks (friction, material, and unevenness classification) and three RCC models (ResNet-18 [13], EfficientNet-B1 [14], and Deit-Tiny [15]), our proposed defensive mechanism outperforms six state-of-the-art general countermeasures (FedAvg [4], Krum [10], Trimmed Mean (TMean) [16], Median [16], FoolsGold [9], and FLAME [17]). This being the first work to safeguard FL-RCC, we establish that this remains a challenging problem, with further improvement required before deployment.

In brief, our main contributions are:

1) The first TLFA analysis for FL-based RCC systems.
2) A new metric for poisoning attack influence capturing transportation safety.
3) A new countermeasure, FLARE, that can detect TLFA poisoned FL-RCC models based on neuron-wise analysis, and prohibit the identified adversaries from further participating in the process.
4) An extensive evaluation of DNN models, RCC tasks, and defensive mechanisms, demonstrating the vulnerability of current FL-RCC systems and showing the improved effectiveness of our FLARE countermeasure.

The rest of this paper is organized as follows: Section II summarizes recent intersection research between RCC, FL, and DPAs. Section III introduces the system model, threat model, and the new evaluation metric. Our scheme is described in detail in Section IV. The described attacks, methods, and current countermeasures are evaluated in Section V. Section VI concludes this paper and indicates future research directions.

II. Background and Related Work

This section first explains the motivation of the emerging research on FL-RCC; then, highlights how TLFAs are effective and a practical threat to FL-RCC; finally, introduces available countermeasures and summarizes their limitations.

A. FL-based RCC

The weather, e.g., snow, rain, or hailstone, may significantly change the road surface *friction*, thus affecting vehicle braking and steering, and ultimately increasing the probability of accidents [18]. Given weather changes can take place locally or unpredictably, it's critical to have reliable detection in real-time. The road *material* type, e.g., concrete or mud, is related to the suspension behavior and passenger comfort. Awareness of material enables vehicles to adjust driving configuration, improving driving experience and safety. The road *unevenness* level also influences ride smoothness, fuel efficiency, and vehicle degradation. Thus, RCC can be multiply beneficial.

Recently, the RCC research community started paying more attention to FL because of its advantages in preserving user privacy and utilizing dispersed resources. Specifically, FedRD [3] implements FL and DNNs for hazardous road damage detection with good performance. Moreover, with individualized differential privacy with pixelization FedRD provides better privacy protection. Then, FLRSC [19] and FedRSC [7] also incorporate FL into the multi-label RCC based on edge-cloud frameworks. The focus of existing FL-RCC proposals is basically on feasibility and privacy; in contrast, *there is no investigation on and no proposals to secure FL-RCC*.

B. DPAs in FL

Although FL is a promising approach to simultaneously bridge data silos and unleash the power of edge AI, by training and deploying AI models on edge devices (e.g., vehicles

and roadside units) with real-time inference, it is inherently vulnerable to poisoning attacks [20]. Malicious clients can either manipulate their local model parameters [21] (i.e., model poisoning attacks) or falsify their local data [22] (i.e., DPAs), so that the resultant poisoned updates affect the global aggregation results in each round.

One type of targeted DPAs, TLFAs [22], change local data labels from source class to target class, while data features remain unchanged. Such attacks are simple yet effective to be mounted by compromised vehicles: 1) they only involve a flipping operation, done before model training, requiring little prior knowledge from participants; and 2) have a great negative impact on source class identification while the changes in other classes are insignificant. However, to the best of our knowledge, there is no investigation of specific DPAs (including TLFAs) targeting FL-RCC.

C. Defensive Mechanisms for DPAs in FL

Various countermeasures in the literature can defend against DPAs, without any consideration of FL-RCC. E.g., Krum [10], TMean [16], Median [16], FoolsGold [9], and FLAME [17]. All the above-mentioned methods either use the ensemble of model parameters or extract the output layer of DNN that predicts labels for poisoning attack analysis. However, in the context of TLFAs, *model parameters directly connected to source and target neurons* are more critical for poisoned model detection, due to contradictory objectives of malicious clients and benign clients [23]. Defensive mechanisms designed for FL-RCC should utilize such **neuron-wise** character for both higher detection accuracy and better computation efficiency.

Only few mechanisms are specifically designed for vehicles, e,g, LFGurad [8], RoHFL [24], and OQFL [25]. Both RoHFL and OQFL are evaluated on *general* classification datasets such as Fashion-MNIST. However, current vehicle research mainly focuses on classical traffic sign classification tasks [8] or steering angle regression [26], and RCC remains untouched. Moreover, the above-mentioned FL countermeasures filter out poisoned *models* as they may be detected in each round, but do not thwart adversarial *clients* participating in the FL.

III. System Model and Threat Model

A. System Model

Cryptographic security and privacy protocols: We consider an FL system with one trusted server and numerous vehicles. All vehicles are registered as legitimate clients, having credentials provided by certification authorities. Each vehicle leverages standardized, state-of-the-art V2X security and privacy [27], notably vehicular public-key infrastructure (VPKI) [28] and short-lived pseudonyms [29] to achieve unlinkability, authenticity, integrity, and non-repudiation. With cloud-based VPKIs [30], vehicles can efficiently obtain pseudonyms, while detection of misbehavior, initially attributed to one or multiple pseudonyms, can lead to efficient revocation and eviction from the system [31]. Each client establishes an individual secure communication channel with the server, using its current pseudonym and the TLS protocol [32] - to ensure end-to-end confidentiality (along with authentication) for the client model contributions. Privacy is enhanced by pseudonymous authentication and other methods, e.g., mix-zones [33], [34].

Task Initialization: If there is a need to train a new RCC model or improve an existing RCC model, the server will release an FL task, made known to vehicles in the system. Similar to the existing machinery for the distribution of revocation list data, a publish-subscribe approach can facilitate this [31]. Based on their willingness and available resources, vehicles can choose to declare their readiness to contribute to the task. We do not dwell on approaches to select among those, it can be random or it can be within a region. Each client must use its credentials and is authenticated (pseudonymously) thus accountably identified from this point on and in the event it is selected for the FL procedure by the server. Assume that eventually K available and preselected vehicles/clients, $V = \{v_1, v_2, \ldots, v_K\}$, form a cluster for this training task. Each vehicle, v_k, owns a private RCC dataset with n_k training samples, $D_k = \{(x_k^1, y_k^1), (x_k^2, y_k^2), \ldots, (x_k^{n_k}, y_k^{n_k})\}$, where x_k^i is a road image captured by the camera on vehicle v_k and y_k^i denotes the corresponding label. In practice, y_k^i can be obtained through driver feedback or annotation tools [35].

FL Procedure: In each communication round, t, first, the server distributes the newest global model, ω^t, to the P participants in the randomly selected set of clients for this round V^t, where $V^t \subseteq V$, $|V^t| = P$, $P \leq K$. Such a selection strategy can balance computation/communication overhead among P vehicles and preclude adversarial clients from dominating the training process over time (the adversaries cannot influence the selection by the server). Second, each participant $V_i^t \in V^t$ updates the received model to ω_i^t using its dataset D_i based on Equation 1 and sends the local model ω_i^t back to the server once the local training is finished, where β is the learning rate, and $\mathcal{L}_i$ is the loss function of vehicle v_i, e.g., cross-entropy.

$$\omega_i^t = \omega^t - \beta \nabla_{\omega^t} \mathcal{L}_i(\omega^t, D_i) \tag{1}$$

Third, after receiving all updated local models from participants, the server executes the aggregation process to form the newest global model ω^{t+1} according to Equation 2,

$$\omega^{t+1} = \sum_{V_i^t \in V^t} q_i \times \omega_i^t \tag{2}$$

where q_i is the aggregation weight (normalized data size in FedAvg [4]). The above processes execute iteratively until the global model converges. Finally, the learned model will be then deployed to the vehicles to support real-time RCC.

B. Threat Model

We assume that among the K clients, there are M malicious ones, where $M < \frac{K}{2}$. Within the general adversary model for V2X ([36]–[38]), here we focus on internal adversaries: vehicles that are part of the system yet compromised and can be pre-selected to contribute to FL-RCC (granted, one can combine attacks from other fronts, e.g., jamming or DoS,

and internal attacks, but this would effectively remove some of the benign/honest contributions). Then, we focus here on adversarial behavior relevant to FL-RCC; considering security and not privacy (e.g., model inversion attacks). Assume there are L classes for the RCC task; and malicious vehicles sort these classes with hazard level into an ordered sequence $\mathcal{S} = \{l_1, l_2, ..., l_L\}$, in which the larger index number indicates a more dangerous class. For instance, a friction classification task with $L = 6$ could set $\mathcal{S}$ as $\{dry, wet, water, fresh - snow, melted - snow, ice\}$.

Capability: Each adversary can only access its local image data. All adversaries agree on the attack goal (i.e., the source and target classes), at each round. Adversaries cannot influence the random selection of clients by the server at each round of FL. They cannot compromise the trusted server or benign clients among the K pre-selected ones. We also assume that the resources of adversaries allow them to flip the original label l_{sr} (source class) to another label l_{tr} (target class), without additional knowledge or computation overhead while keeping the images/features unchanged. Once the simple label-flipping operations are finished, the malicious clients update local models based on poisoned data and then transmit these poisoned local model parameters to the server for global aggregation. The newest global model parameters are thus poisoned and distributed to the participants in the next round.

Objective: The M malicious clients aim to reduce the global model RCC performance regarding a source class through TLFA, rather than reduce the overall performance for all classes. As transportation safety is a priority compared to traffic efficiency, we assume the former is the attack goal. Thus, the adversary should flip a source class, l_{sr}, to a target class, l_{tr}, during the model training process, where $sr > tr$, e.g., flipping from l_6 (e.g., ice) to l_1 (e.g., dry), denoted as $l_6 \rightarrow l_1$. In this way, the adversary tries to convince the model to consider the real source (e.g., ice) condition as the false target (e.g., dry) condition. Thus, the vehicle underestimates its current transportation safety level, which may cause an accident. Note that if the attack target of the adversary were traffic efficiency, the flipping direction is the opposite, which would mislead the vehicle to overestimate its transportation safety situation and, e,g., slow down or even pull over; those attacks merit a separate future investigation.

Capturing TLFA Influence on Transportation Safety: Consider the deployed model misclassifying l_{sr1} or l_{sr2} to l_{st} because of TLFAs, where $sr1 > sr2 > st$; the traditional metrics on model performance, such as accuracy and recall, treat the two situations as equivalent. However, the misclassification regarding l_{sr1} poses a greater risk to transportation safety, as it leads to a significant or even larger underestimation of the true hazard level. Another example is the misclassification of l_{sr} to l_{st1} or l_{st2}, where $sr > st1 > st2$; although the second prediction (l_{st2}) is more dangerous than the first one, traditional metrics again treat them equally, neglecting the practical consequences of the misclassification. This implies the need to design a new metric based on "label distance" in the context of $\mathcal{S}$ to reflect the influence of the attack on RCC.

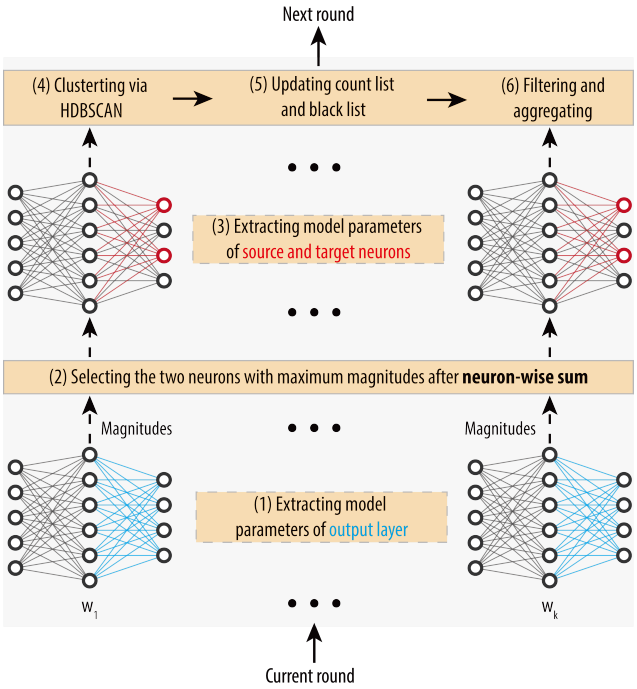

Fig. 2. Workflow of FLARE in a round. Steps (1) and (3) are executed for each model, while the rest steps are implemented based on a models cluster.

To do so, first, we calculate a weighted distance, $d_{i,j}$[5], between the true label, l_i, and the predicted label, l_j, according to an exponential function defined by Equation 3, where e is Euler's number. This way, higher label distance suggests higher danger level due to mis-classification; the distance is 1 for correct predictions (i.e., $i = j$). For instance, let l_1, l_2, and l_3 be severe-uneven, slight-uneven, and smooth, respectively; then $d_{1,2} = 1.359$ while $d_{1,3} = 1.847$.

$$d_{i,j} = (\frac{e}{2})^{|i-j|} \tag{3}$$

We include the Equation 3 distance in the confusion matrix correction to get a weighted matrix X, with elements $X[i,j] = d_{i,j} \times n_{i,j}$, where $n_{i,j}$ denotes the number of testing samples whose true labels are i but are predicted as j. Based on X, we calculate $Error = 1 - Acc$ through Equation 4. We can also compute new attack success rate (ASR), etc., with the weighted confusion matrix. Here, we choose $Error$ because it comprehensively involves all $n_{i,j}$. Generally speaking, higher $Error$ indicates worse attack influence, implying, in turn, that the resultant false predictions are more dangerous.

$$Acc = \frac{\sum_{i=1}^{L} n_{i,i}}{\sum_{i=1}^{L}\sum_{j=1}^{L} d_{i,j} \times n_{i,j}} \tag{4}$$

[5]Note that $d_{i,j}$ is not chosen to be linear exactly in order to "amplify" the value as conditions (and thus labels) become more "remote"/distinct.

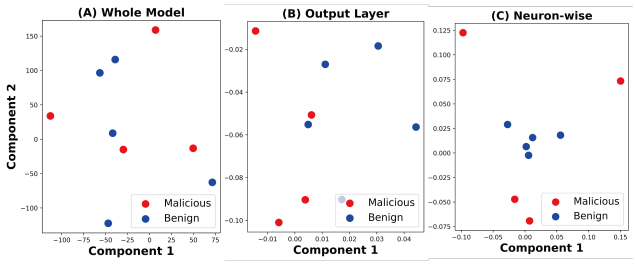

Fig. 3. Comparison between model, layer, and neuron-wise levels.

Algorithm 1 Protocol of FLARE

1: Initialize black list $BL = \emptyset$ and count list $CL = \{i{:}\ 0$ for i in range$(K)\}$
2: **for** each round $t \in [1, T]$ **do**
3: $V^t \leftarrow$ random set of P clients from $V - BL$
4: The server sends ω^t to all clients in V_t
5: **for** each client $V_i^t \in V^t$ in parallel **do**
6: Update local model ω_i^t according to Equation 1
7: Send ω_i^t back to the server
8: **end for**
9: The server receives updates from V^t
10: **for** each ω_i^t the server **do**
11: $\Delta_{i,S}^t = \{\omega_{i,l}^t - \omega_l^t | l \in S\}$ ▷ Output layer changes
12: Calculate magnitudes $||\Delta_{i,l}^t||$ for $l \in S$
13: **end for**
14: $\{||\Delta_{l_1}^t||, ..., ||\Delta_{l_{|S|}}^t||\} \leftarrow$ neuron-wise magnitudes
15: $l_{sr}, l_{tr} \leftarrow$ Top-2$(\{||\Delta_{l_1}^t||, ..., ||\Delta_{l_{|S|}}^t||\})$ ▷ $sr > tr$
16: $U^t \leftarrow \{\Delta_{i,l}^t | V_i^t \in V^t, l \in \{l_{sr}, l_{tr}\}\}$
17: V_{out}^t = HDBSCAN(U^t)
18: $\omega^{t+1} = Aggeregate\{\omega_i^t | V_i^t \notin V_{out}^t\}$
19: **for** $V_i^t \in V_{out}^t$ **do**
20: $CL[i]$+=1
21: **if** $CL[i] > Threshold\ And\ v_i \notin BL$ **then**
22: Add v_i in BL
23: **end if**
24: **end for**
25: **end for**
26: **return** ω^{T+1}

IV. FLARE: Our FL Scheme against TLFA in RCC with Efficiency

Scheme Overview: To thwart TLFAs on FL-RCC, we design FLARE, achieving the following three objectives: 1) identify source and target classes; 2) recognize and filter out poisoned model parameters maintaining benign ones for global aggregation; and 3) remove promptly malicious clients from V during the FL procedure. We depict FLARE workflow in Fig. 2, which consists of six consecutive steps in each round.

The FLARE protocol is further illustrated in Algorithm 1. First (*line 1*), we initialize a blacklist and a count list for filtering. Second (*lines 2-8*), in each round, the server randomly selects clients not on the blacklist for local training. The server notifies the selected clients leveraging their authenticated declaration of availability; the selection of the server is authenticated by the client (identified by its pseudonym certificate), establishes a secure channel with the server. Third (*lines 9-15*), the server receives updated model parameters over each of the client-server secure channels, and executes the neuron-wise analysis to identify source and target classes. Fourth (*lines 16-17*), HDBSCAN [39] is utilized to recognize outliers based on parameters connected to the two classes. Each of the detected outliers, sent over each of the secure channels, is non-repudiably sent by a specific client (its current pseudonym) - thus the client can be excluded by the FL procedure. Finally (*lines 18-24*), the global model, count list, and blacklist are updated.

Intuition: FLARE is based on the observation [23] that benign and poisoned model parameters have distinct differences for those parameters directly connected to source and target neurons. Intuitively, adversaries mounting TLFAs and honest clients have contradicting objectives for local model training, reflected on these parameters. This indicates that we can detect TLFAs more efficiently by focusing on those more critical parameters; instead of utilizing the whole model parameters (e.g., Krum [10]) or the whole output layer of DNN (e.g., FoolsGold [9]). Fig. 3 visualizes the comparison between malicious and benign parameters using different features.

Design Details: From the defense perspective, first, we need to recognize the potential source and target classes for TLFAs. To do so, for each participant V_i^t, we calculate the changes in model parameters connected to the output layer neurons as $\Delta_{i,S}^t = \{\omega_{i,l}^t - \omega_l^t | l \in S\}$. Then, we aggregate the neuron-wise change magnitudes among V^t into the vector $\{||\Delta_{l_1}^t||, ||\Delta_{l_2}^t||, ..., ||\Delta_{l_{|S|}}^t||\}$, i.e., we sum $|V^t|$ weight value changes for each particular neuron. We identify the two neurons with the highest magnitudes as source and target classes. This is because, for other neurons, the malicious clients and the benign clients share the same training objectives, leading to smaller change magnitudes; on the other hand, source and target neurons are expected to produce larger magnitudes in view of inconsistent training objectives.

FLARE filters out poisoned model parameters based on parameter changes of the two recognized source and target neurons. Specifically, we utilize HDBSCAN [39] to cluster the extracted parameter changes, and those changes identified as outliers will be regarded as changes from malicious clients. Thus, the corresponding model parameters can straightforwardly be kept out of the current round of global aggregation to avoid TLFAs. We adopt HDBSCAN because it is an advanced clustering method based on density without the need to pre-define the number of clusters. If we choose other traditional clustering technologies, e.g., K-Means, we would need to define the number of clusters as 2 in advance: one for poisoned model parameters and the other for benign ones. However, for some FL rounds, there may be no poisoned model parameters, as only a subset of clients (not all the clients) are selected to participate in the current round.

FLARE maintains a blacklist by recording the number of times each client was associated with poisoned model parameters during the entire FL process. If this count reaches a threshold, the corresponding client is excluded from the rest of the rounds. Such a blacklist can reduce the detection burden and improve the efficiency of defense, as we do not need to filter out the poisoned model parameters provided by adversaries again and again. Any such exclusion of a client, in practice, its credential, constitutes evidence of misbehavior and is reported by the server to the VPKI; which in turn, can revoke

TABLE I
DEFAULT TRAINING CONFIGURATIONS USED IN THIS PAPER.

Term	Value
Loss Function	Cross-Entropy
Batch Size	64
Learning Rate (LR)	0.03
Momentum for LR	0.5
Optimizer	SGD
Local Epoch	3
Total Round	60

TABLE II
SUMMARY OF MODEL INFORMATION OF FRICTION TASK.

Model Type	Model Size (MB)	Inference Time (ms)	Memory Usage (MB)
ResNet-18	21.32	2.18	150.52
EfficientNet-B1	12.44	9.16	237.30
DeiT-Tiny	10.54	4.38	97.55

the current and possibly all client credentials ([28], [31]), and further protect the system having evicted adversaries.

Complexity Analysis: Assume the dimensionalities of the whole DNN model, the output layer of DNN, and one neuron in the output layer are d_w, d_o, and d_e, respectively. Note that $d_w \gg d_o \gg d_e$. The computation overhead of FLARE in each round includes the following parts: 1) $\mathcal{O}(Pd_o)$ to calculate the output layer changes of P clients; 2) $\mathcal{O}(PLd_e)$ to compute the neuron-wise magnitudes of L output neurons and P clients; 3) $\mathcal{O}(L\,log\,L)$ to identify the source and target neurons; 4) $\mathcal{O}(Pd_e)$ to cluster parameters using HDBSCAN; and 5) $\mathcal{O}(P)$ to maintain blacklist and count list. Such that, the overall complexity of FLARE is $\mathcal{O}(Pd_o)$. Compared to other countermeasures based on the entire model, the output layer, or K-Means, e.g., Median ($\mathcal{O}(P\,log\,Pd_w)$), TMean ($\mathcal{O}(P\,log\,Pd_w)$), Krum ($\mathcal{O}(P^2d_w)$), and FoolsGold ($\mathcal{O}(P^2d_o)$), in brief, FLARE is computation-efficient.

Practical Considerations: The RCC model should be lightweight and trade-off cost for classification performance, so that vehicle resources can support local model training, parameter transmitting, and real-time inference. We leverage standardized state-of-the-art vehicular security and privacy technologies. The use of pseudonymous authentication ensures that the participation of the same client in two FL processes cannot be linked as long as it uses two different pseudonyms (thus public/private key pair) and as mandated a new network (IP) address. This unlinkability and the client-server encryption of the contributed model parameters (confidentiality) reinforce the FL-provided privacy motivation. Equally important, leveraging a VPKI allows this while maintaining accountability and adversary exclusion and eviction. All these aspects, clearly feasible and standard compliant, are not discussed and analyzed in further detail due to the paper's focus on FL targeting attacks. Our scheme can easily adapt to and safeguard other DNN-based autonomous driving tasks beyond RCC, as the neuron-wise analysis is designed for TLFAs and can work for any DNN, no matter its specific classification task.

V. EVALUATION

A. Experiment Settings

We simulate 40 clients/vehicles and one server based on PyTorch, using an NVIDIA A100 GPU with 40GB memory and an Icelake CPU with 16 cores and 128GB memory for the entire system. The server will randomly select 8 out of 40 clients as participants in each round. Each experiment is repeated four times, and the averaged results are provided. The default training configurations are summarized in Table I. We adopt code in [22] as the basic FL framework. Note that as a snapshot of the system operation, the participants are a subset: $K = 40$ preselected clients corresponds to one instantiation of the protocol, e.g., those selected in an urban area after an event (e.g., abrupt change of weather conditions) that makes re-training necessary or valuable.

We utilize the Road Surface Classification Dataset (RSCD) [1] for our evaluation. It contains 1 million real-world samples captured by cameras on vehicles. We resize image size to $224 \times 224 \times 3$ for efficiency. We create three sub-datasets based on RSCD for three crucial RCC tasks, classification on a specific dimension: 1) **RCC @ Friction**, which contains 58,800 training samples, 14,550 testing samples, and 6 labels (dry, wet, water, fresh-snow, melted-snow, and ice); 2) **RCC @ Material**, which includes 57,000 training images and 15,000 testing images with 4 categories (asphalt, concrete, mud, and gravel); and 3) **RCC @ Unevenness**, which consists of 57,542 training pictures and 18,000 testing pictures labeled by smooth, slight-uneven, or severe-uneven.

Similar to the literature, e.g., [23], this paper adopts the Dirichlet distribution to create Non-IID training data for each client and the default α is 1.0 in the experiments. By default, there are 12 malicious clients among the 40 preselected (30% poisoned rate). If included in the FL process, they execute the TLFAs by shifting the labels in their training datasets from *water* to *dry* for Friction, from *gravel* to *asphalt* for Material, and from *severe-uneven* to *smooth* for Unevenness. As for testing datasets, they remain unchanged during the training and are only used for inference.

To evaluate the generality and compatibility of each method, we train three popular DNN models under FL for the RCC tasks: ResNet-18 [13], EfficientNet-B1 [14], and Deit-Tiny [15]. They are all lightweight versions of DNN thus suitable for vehicles, and the detailed model information is summarized in Table II. To evaluate current countermeasures against TLFAs in RCC, we compare the following seven methods in the experiments: FedAvg [4] (not really a defense) Krum [10], Trimmed Mean [16], Median [16], FoolsGold [9], FLAME [17], and FLARE. The following four evaluation metrics quantify different performance aspects: Global Accuracy (GAcc), Source Recall (SRec), ASR, and $Error$.

B. TLFA Impact on FL-RCC

By comparing the results with and without TLFAs, we can analyze the influence of the attack on FL-RCC. Note that NA denotes No Attack, and other rows without this symbol

TABLE III
THE OVERALL RESULTS WITHIN DEFAULT CONFIGURATIONS. ALL VALUES ARE RATIOS IN %.

Model	Method	RCC @ Friction				RCC @ Material				RCC @ Unevenness			
		GAcc↑	SRec↑	ASR↓	Error↓	GAcc↑	SRec↑	ASR↓	Error↓	GAcc↑	SRec↑	ASR↓	Error↓
ResNet-18 [13]	FedAvg-NA‡ [4]	86.03	76.74	6.40	21.73	80.60	80.66	5.73	27.16	74.30	69.24	11.49	33.15
	FedAvg [4]	81.95	50.77	45.59	27.68	75.34	54.48	50.23	35.81	69.14	46.74	50.34	39.51
	Krum [10]	70.91	40.01	65.20	43.07	61.50	48.69	59.70	51.34	48.36	32.44	76.51	61.07
	TMean [16]	82.04	50.55	52.92	27.46	75.74	56.95	44.49	35.20	69.45	**50.25**	37.09	39.28
	Median [16]	79.15	44.50	35.86	31.54	68.26	41.55	57.47	44.11	55.95	22.08	63.41	53.81
	FoolsGold [9]	82.08	53.24	57.78	27.50	74.72	52.68	61.69	36.74	**69.56**	48.88	40.67	**39.17**
	FLAME [17]	82.17†	52.69	38.38	27.66	67.62	56.26	60.14	35.73	65.92	46.99	38.03	44.14
	FLARE (Ours)	**82.84***	**56.62**	**31.16**	**26.27**	**76.73**	**60.09**	**35.83**	**33.60**	68.87	47.76	**35.41**	40.06
EfficientNet-B1 [14]	FedAvg-NA	87.37	80.18	3.72	19.82	81.36	80.64	4.01	26.50	74.45	72.31	9.68	32.97
	FedAvg	84.35	60.13	33.16	37.57	78.43	65.21	39.27	31.17	70.26	52.73	37.07	38.34
	Krum	74.82	41.86	41.39	24.38	63.20	42.74	55.11	49.84	53.80	25.94	79.25	56.12
	TMean	85.63	68.76	26.01	**22.31**	78.65	65.94	31.03	31.00	69.23	53.69	39.18	39.67
	Median	84.70	69.52	21.00	24.05	76.10	57.52	38.75	34.27	64.16	37.21	57.25	45.46
	FoolsGold	85.23	66.45	36.71	25.39	76.13	56.83	56.63	34.81	70.34	52.70	28.70	38.25
	FLAME	82.24	56.09	11.72	25.66	77.01	66.85	32.57	33.92	69.71	55.74	30.43	38.91
	FLARE (Ours)	**86.20**	**74.28**	**6.76**	23.20	**78.95**	**69.18**	**31.02**	**30.45**	**71.58**	**56.91**	**28.42**	**36.73**
Deit-Tiny [15]	FedAvg-NA	87.86	82.90	4.18	21.72	81.29	80.95	5.66	26.39	74.06	69.86	8.10	33.48
	FedAvg	83.79	58.02	49.34	25.06	74.33	51.99	56.92	37.50	67.86	44.11	49.13	40.83
	Krum	70.59	45.83	71.41	44.60	55.44	40.72	68.12	57.52	52.09	31.69	70.90	57.48
	TMean	81.84	48.66	69.80	28.19	75.88	57.49	44.92	35.03	68.44	46.28	44.23	40.31
	Median	**84.43**	**62.29**	**34.59**	**24.23**	74.83	55.09	44.43	36.22	68.17	48.44	**36.76**	40.68
	FoolsGold	83.86	61.60	49.41	25.04	76.56	58.76	37.20	34.11	67.76	45.42	39.45	41.12
	FLAME	81.44	59.02	36.16	25.02	75.93	60.39	38.95	33.62	68.51	48.84	49.48	41.81
	FLARE (Ours)	84.37	62.10	40.35	24.41	**77.40**	**61.64**	**30.05**	**32.42**	**68.87**	**53.61**	38.33	**39.66**

* **Bold numbers are the best performance.**
† Numbers with underline are the second best values.
‡ NA denotes No Attack. Others without this symbol are all under attack.

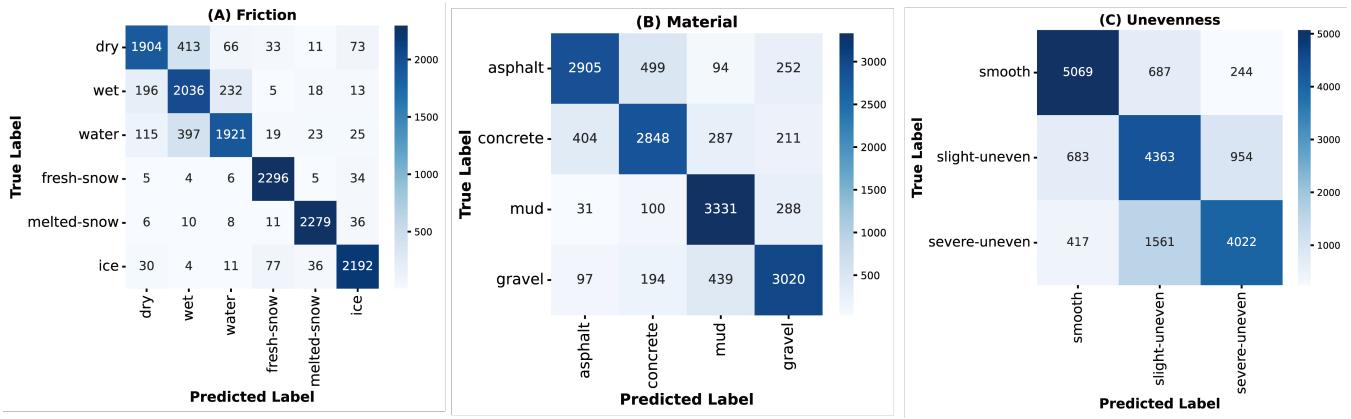

Fig. 4. The confusion matrices of FLARE based on EfficientNet-B1 in three RCC tasks: (A) Friction, (B) Material, and (C) Unevenness.

are all under attack. The performance of FedAvg, which is not actually a defense measure, degrades severely compared with FedAvg-NA for the same models and the same RCC tasks. According to Table III, we can make the following *assessments*: 1) average reductions caused by the TLFAs to the GAcc among the three models are 3.72%, 5.05%, and 5.18% for Friction, Material, and Unevenness, respectively; 2) the average decrease on SRecs is even more significant, achieving 23.63%, 23.52%, and 22.61% for the three tasks respectively; 3) ASRs of FedAvg increased by 37.93%, 43.67%, and 35.76% compared to FedAvg-NA for the three tasks, respectively; and 4) Errors regarding Friction, Material, and Unevenness grow by 9.01%, 8.14% and 6.37%, respectively. All in all, TLFAs primarily influence ASR and SRec, as the goal of TLFAs is to misguide prediction involving source and target classes.

The influence of the *poisoned rate* can be analyzed based on Table IV (20% and 40% poisoned rate) and EfficientNet-B1's results (30% poisoned rate) in Table III: The higher the poisoned rate, the worse the performance; GAcc, SRec, ASR, and $Error$ all deteriorate gradually when the poisoned rate increases, especially for ASR; whose values degenerate from 3.72% to 48.60% for Friction, from 4.01% to 67.63% for

TABLE IV
RESULTS AS A FUNCTION OF POISONED RATES BASED ON EFFICIENTNET-B1. ALL VALUES ARE RATIOS IN %.

Poisoned Rate	Method	RCC @ Friction				RCC @ Material				RCC @ Unevenness			
		GAcc ↑	SRec ↑	ASR ↓	Error ↓	GAcc ↑	SRec ↑	ASR ↓	Error ↓	GAcc ↑	SRec ↑	ASR ↓	Error ↓
-	FedAvg-NA	87.37	80.18	3.72	19.82	81.36	80.64	4.01	26.50	74.45	72.31	9.68	32.97
20%	FedAvg	85.41	67.99	25.11	22.65	79.33	69.41	27.20	29.78	71.18	60.46	30.68	37.21
	Krum	72.70	64.03	9.01	40.96	65.82	47.39	59.21	47.03	56.31	31.98	57.13	53.19
	TMean	86.12	72.49	16.32	21.64	79.15	69.42	27.82	30.08	71.14	56.12	28.94	37.27
	Median	85.01	69.42	23.49	23.67	78.29	67.24	17.67	30.95	70.86	**62.29**	**22.14**	37.29
	FoolsGold	85.61	69.99	25.03	22.41	79.85	**73.17**	20.82	28.93	**71.78**	58.31	24.15	36.43
	FLAME	86.02	71.44	13.13	22.15	78.46	71.12	18.92	29.29	70.81	59.42	25.75	36.82
	FLARE (Ours)	**86.75**	**77.00**	**6.72**	**20.52**	**80.21**	72.16	**14.39**	**28.47**	**72.19**	61.52	23.77	**35.88**
40%	FedAvg	83.43	54.80	48.60	25.48	74.41	49.54	67.63	37.59	64.53	36.36	63.33	45.36
	Krum	73.47	51.37	65.37	40.00	63.07	31.29	70.96	50.56	48.38	15.09	85.76	62.11
	TMean	82.77	51.35	44.28	26.42	75.11	49.76	50.49	36.61	68.58	**52.24**	44.58	40.24
	Median	82.87	58.84	47.00	26.66	74.67	50.31	51.44	36.50	61.80	29.96	75.64	48.26
	FoolsGold	83.61	57.06	43.18	25.37	77.09	57.46	35.76	33.51	66.17	40.27	45.49	43.50
	FLAME	83.51	56.39	34.73	25.29	77.66	62.76	32.95	34.46	65.43	43.43	46.79	42.22
	FLARE (Ours)	**84.38**	**62.37**	**16.80**	**23.98**	**79.57**	**72.83**	**21.12**	**29.71**	**68.79**	45.61	**31.62**	**40.17**

Material, and from & 9.68% to 63.33% for Unevenness.

From these results, we can further have the following *observations*: 1) in the view of RCC tasks, TLFAs have a more significant impact on Unevenness, which may be because the number of labels in this task is limited while the learned model parameters are more sensitive; 2) from the model standpoint, TLFAs are general and have adverse effects on all evaluated models; and 3) in terms of influence factors, higher poisoned rates one can naturally expect leads to worse performance. In conclusion, current FL-RCC systems are vulnerable to TLFAs, and we should pay more attention to such powerful attacks.

C. FLARE Defense Effectiveness

As per Table III, the performance of *Krum* is worse than FedAvg in most cases, which may be caused by the fact that Krum simply chooses one local model as the newest global model. Even if Krum can choose the benign one in each round, the advantage of collaboration in FL is gone, thus the performance will degrade. Moreover, once the selected model is the malicious one, the adverse effect is significant. *TMean, Median, FoolsGold, and FLAME* perform better than FedAvg in some cases, especially for SRec and ASR. Considering SRec, the highest values among FedAvg, TMean, Median, FoolsGold, and FLAME are 69.52% for Friction, 66.85% for Material, and 55.74% for Unevenness. However, these values are not satisfactory and unstable. In other words, simply adapting existing countermeasures cannot be effective for the heterogeneous FL-RCC scenarios under TLFAs.

In most cases (88.89%), *FLARE* can achieve the best or second-best performance compared to the baseline schemes. Specifically, the best ASR values of FLARE are 6.76% for Friction, 30.05% for Material, and 28.42% for Unevenness; the lowest $Error$ values are 23.20% (Friction), 30.45% (Material), and 36.73% (Unevenness). Compared to the best baseline results, on average, 2.23% and 0.76% improvements are achieved for ASR and $Error$, respectively. This superiority by FLARE indicates that paying attention to source and target neurons rather than coarsely analyzing the entire model or output layer can indeed improve defense effectiveness.

Based on Table IV, FLARE is more robust and performs better than other baselines when the *poisoned rate* increases. Specifically, 1) the ASR of FLARE in Friction ranges from 6.72% to 16.80%, while the ASR of the best baseline schemes ranges from 9.01% to 34.73%; and 2) FLARE can achieve 72.83% SRec in Material with 40% poisoned rate, while the values of baseline schemes range from 31.29% to 62.76%. These results suggest that the poisoned rate can influence performance severely, while FLARE is more robust when TLFAs are serious. Fig. 4 provides detailed prediction results of FLARE via *confusion matrices*. FLARE can predict most samples correctly, even for those with source classes (water, grave, and severe-uneven). Moreover, most of the misclassified results are concentrated around the true labels, indicating the influence of the poisoning attack is largely mitigated.

Discussion: Although FLARE can achieve overall the best performance compared to baselines for the FL process, two limitations are observed from the results: 1) there is still a huge gap between FLARE and FedAvg-NA; and 2) the number of misclassified samples is not small enough, according to Fig. 4. These observations indicate that 1) TLFAs are powerful attacks on FL-RCC systems, very difficult to defend against; and 2) filtering out potentially malicious model parameters in aggregation is not enough, and mitigating the deterioration of already poisoned global model could also be beneficial.

VI. CONCLUSIONS

This paper investigates and reveals the vulnerability of current FL-RCC systems to TLFAs. We propose an evaluation metric based on label distance to better reflect transportation safety and a scheme named FLARE based on neuron-wise analysis to better quantify and address such a vulnerability. Experimental results based on various RCC tasks, models, and baselines show the significance of TLFAs and the relative

effectiveness of FLARE. Nonetheless, the stability and performance of current countermeasures can be further enhanced.

Ongoing work: We will investigate TLFA influence across multiple FL-RCC tasks and assess FLARE effectiveness, also measuring latency with embedded hardware. Second, we will further improve FLARE performance, exploring machine unlearning and knowledge distillation for poisoned model correction. Finally, we will consider adaptive colluding adversaries.

ACKNOWLEDGMENT

This work was supported in parts by WASP, VR, and in kind by the KAW Foundation granting access to Berzelius at the National Supercomputer Centre.

REFERENCES

[1] T. Zhao, J. He, J. Lv, D. Min, and Y. Wei, "A comprehensive implementation of road surface classification for vehicle driving assistance: Dataset, models, and deployment," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 8, pp. 8361–8370, 2023.

[2] M. Otoofi, L. Laine, L. Henderson, W. J. B. Midgley, L. Justham, and J. Fleming, "FrictionSegNet: Simultaneous semantic segmentation and friction estimation using hierarchical latent variable models," *IEEE Transactions on Intelligent Transportation Systems*, vol. 25, no. 12, pp. 19 785–19 795, 2024.

[3] Y. Yuan, Y. Yuan, T. Baker, L. M. Kolbe, and D. Hogrefe, "FedRD: Privacy-preserving adaptive federated learning framework for intelligent hazardous road damage detection and warning," *Future Generation Computer Systems*, vol. 125, pp. 385–398, 2021.

[4] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *AISTATS*, 2017, pp. 1273–1282.

[5] S. Liu, L. You, R. Zhu, B. Liu, R. Liu, H. Yu, and C. Yuen, "AFM3D: An asynchronous federated meta-learning framework for driver distraction detection," *IEEE Transactions on Intelligent Transportation Systems*, vol. 25, no. 8, pp. 9659–9674, 2024.

[6] L. You, S. Liu, B. Zuo, C. Yuen, D. Niyato, and H. V. Poor, "Federated and asynchronized learning for autonomous and intelligent things," *IEEE Network*, vol. 38, no. 2, pp. 286–293, 2024.

[7] I. V. Vondikakis, I. E. Panagiotopoulos, and G. J. Dimitrakopoulos, "FedRSC: A federated learning analysis for multi-label road surface classifications," *IEEE Open Journal of Intelligent Transportation Systems*, vol. 5, pp. 433–444, 2024.

[8] K. Sameera, P. Vinod, R. R. KA, and M. Conti, "LFGurad: A defense against label flipping attack in federated learning for vehicular network," *Computer Networks*, vol. 254, p. 110768, 2024.

[9] C. Fung, C. J. Yoon, and I. Beschastnikh, "The limitations of federated learning in sybil settings," in *RAID*, 2020, pp. 301–316.

[10] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," in *NeurIPS*, 2017, p. 118–128.

[11] H. Fereidooni, A. Pegoraro, P. Rieger, A. Dmitrienko, and A.-R. Sadeghi, "FreqFed: A frequency analysis-based approach for mitigating poisoning attacks in federated learning," in *NDSS*, 2024, pp. 1–16.

[12] N. Wang, Y. Xiao, Y. Chen, Y. Hu, W. Lou, and Y. T. Hou, "FLARE: Defending federated learning against model poisoning attacks via latent space representations," in *ASIACCS*, 2022, p. 946–958.

[13] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016, pp. 770–778.

[14] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *ICML*, 2019, pp. 6105–6114.

[15] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, "Training data-efficient image transformers & distillation through attention," in *ICML*, 2021, pp. 10 347–10 357.

[16] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *ICML*, 2018, pp. 5650–5659.

[17] T. D. Nguyen, P. Rieger, R. De Viti, H. Chen, B. B. Brandenburg, H. Yalame, H. Möllering, H. Fereidooni, S. Marchal, M. Miettinen *et al.*, "FLAME: Taming backdoors in federated learning," in *USENIX Sec*, 2022, pp. 1415–1432.

[18] F. Malin, I. Norros, and S. Innamaa, "Accident risk of road and weather conditions on different road types," *Accident Analysis & Prevention*, vol. 122, pp. 181–188, 2019.

[19] I. V. Vondikakis, I. E. Panagiotopoulos, and G. J. Dimitrakopoulos, "An adaptive federated learning framework for intelligent road surface classification," in *ITSC*, 2023, pp. 4121–4126.

[20] X. Zhang, Q. Liu, Z. Ba, Y. Hong, T. Zheng, F. Lin, L. Lu, and K. Ren, "FLTracer: Accurate poisoning attack provenance in federated learning," *IEEE Transactions on Information Forensics and Security*, vol. 19, pp. 9534–9549, 2024.

[21] V. Shejwalkar and A. Houmansadr, "Manipulating the byzantine: Optimizing model poisoning attacks and defenses for federated learning," in *NDSS*, 2021, pp. 1–18.

[22] V. Tolpegin, S. Truex, M. E. Gursoy, and L. Liu, "Data poisoning attacks against federated learning systems," in *ESORICS*, 2020, pp. 480–501.

[23] N. M. Jebreel, J. Domingo-Ferrer, D. Sánchez, and A. Blanco-Justicia, "LFighter: Defending against the label-flipping attack in federated learning," *Neural Networks*, vol. 170, pp. 111–126, 2024.

[24] H. Zhou, Y. Zheng, H. Huang, J. Shu, and X. Jia, "Toward robust hierarchical federated learning in internet of vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 5, pp. 5600–5614, 2023.

[25] W. Yamany, N. Moustafa, and B. Turnbull, "OQFL: An optimized quantum-based federated learning framework for defending against adversarial attacks in intelligent transportation systems," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 1, pp. 893–903, 2023.

[26] S. Wang, Q. Li, Z. Cui, J. Hou, and C. Huang, "Bandit-based data poisoning attack against federated learning for autonomous driving models," *Expert Systems with Applications*, vol. 227, p. 120295, 2023.

[27] T. ETSI, "ETSI TS 102 941 v1. 1.1-Intelligent Transport Systems (ITS); security; trust and privacy management, standard, TC C-ITS," 2021.

[28] "IEEE standard for wireless access in vehicular environments–security services for application and management messages," *IEEE Std 1609.2-2022 (Revision of IEEE Std 1609.2-2016)*, pp. 1–349, 2023.

[29] M. Khodaei, H. Jin, and P. Papadimitratos, "SECMACE: Scalable and robust identity and credential management infrastructure in vehicular communication systems," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 5, pp. 1430–1444, 2018.

[30] M. Khodaei, H. Noroozi, and P. Papadimitratos, "SECMACE+: Upscaling pseudonymous authentication for large mobile systems," *IEEE Transactions on Cloud Computing*, vol. 11, no. 3, pp. 3009–3026, 2023.

[31] M. Khodaei and P. Papadimitratos, "Scalable & resilient vehicle-centric certificate revocation list distribution in vehicular communication systems," *IEEE Transactions on Mobile Computing*, vol. 20, no. 7, pp. 2473–2489, 2021.

[32] E. Rescorla, "The transport layer security (TLS) protocol version 1.3," Tech. Rep., 2018.

[33] P. Papadimitratos, "Mix-Zones in Wireless Mobile Networks," in *Encyclopedia of Cryptography, Security and Privacy*, S. Jajodia, P. Samarati, and M. Yung, Eds. Springer Nature Switzerland, 2025, pp. 1555–1559.

[34] M. Khodaei and P. Papadimitratos, "Cooperative location privacy in vehicular networks: Why simple mix-zones are not enough," *IEEE Internet Of Things Journal*, vol. 8, no. 10, pp. 7985–8004, 2021.

[35] Y. Zhou, L. Cai, X. Cheng, Q. Zhang, X. Xue, W. Ding, and J. Pu, "OpenAnnotate2: Multi-modal auto-annotating for autonomous driving," *IEEE Transactions on Intelligent Vehicles*, pp. 1–13, 2024.

[36] P. Papadimitratos, V. Gligor, and J.-P. Hubaux, "Securing vehicular communications-assumptions, requirements, and principles," in *ESCAR*, 2006, p. 5–14.

[37] P. Papadimitratos, L. Buttyan, T. Holczer, E. Schoch, J. Freudiger, M. Raya, Z. Ma, F. Kargl, A. Kung, and J.-P. Hubaux, "Secure vehicular communication systems: design and architecture," *IEEE Communications Magazine*, vol. 46, no. 11, pp. 100–109, 2008.

[38] H. Jin, Z. Zhou, and P. Papadimitratos, "Future-proofing secure V2V communication against clogging dos attacks," in *ARES*, 2024, pp. 1–8.

[39] L. McInnes, J. Healy, and S. Astels, "hdbscan: Hierarchical density based clustering," *Journal of Open Source Software*, vol. 2, no. 11, p. 205, 2017.