Transportation Research Part C xxx (xxxx) xxx

Contents lists available at ScienceDirect

Transportation Research Part C

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

Data poisoning attacks on traffic state estimation and prediction☆

Feilong Wang [a], Xin Wang [a], Yuan Hong [b], R. Tyrrell Rockafellar [a], Xuegang (Jeff) Ban [a,*]

[a] University of Washington, Seattle, WA 98105, United States
[b] University of Connecticut, Storrs, CT 06269, United States

ARTICLE INFO

Keywords:
Data poisoning attacks
Attack model
Traffic state estimation and prediction
Lipschitz continuity
Semi-derivatives

ABSTRACT

Data has become ubiquitous nowadays in transportation, including vehicular data and infrastructure-generated data. The growing reliance on data poses potential cybersecurity issues to transportation systems, among which the so-called "data poisoning" attacks by adversaries are becoming increasingly critical. Such attacks aim to compromise a system's performance by adding systematic and malicious noises, perturbations, or deviations to the dataset used by the system. Formal investigations of data poisoning attacks are essential for understanding the attacks and developing effective defense methods. This study develops a general data poisoning attack model for traffic state estimation and prediction (TSEP) that is a basic application in transportation. We first formulate data poisoning attacks as a general sensitivity analysis of parameterized optimization problems over parameter changes (i.e., data perturbations) and study the Lipschitz continuity property of the solution with the presence of general (equality and inequality) constraints. Then, we develop attack models that fit a broader spectrum of learning applications (such as TSEP) by extending existing models that only focus on learning problems with no or equality constraints (widely used in the cybersecurity field). Since the solution of such general problems is often continuous but not differentiable with data changes, we apply the generalized implicit function theorem to compute the semi-derivatives that express how the TSEP solution responds to data perturbations. The semi-derivatives enable us to evaluate TSEP models' vulnerability (at each data point) and solve the proposed attack model. We demonstrate the generality and effectiveness of the proposed method on two TSEP models using mobile sensing data.

1. Introduction

Transportation has been and will continue to be under rapid transformations. One transformation is the massive old and new datasets that are now widely collected and becoming more available. The data has revolutionized almost every aspect of transportation, including traffic state estimation and prediction (*TSEP*) (Ban et al., 2011), traffic control (Li and Ban, 2019), safety and behavioral research (Chen et al., 2019), and origin–destination estimation (Chen et al., 2017), among others. Yet, the massively available data and the currently accentuated "data-driven" methods in transportation do come with a price. This includes the so-called "data poisoning" attacks by adversaries to add malicious noises or perturbations to a dataset to produce erroneous results if the dataset

☆ This article belongs to the Virtual Special Issue on "ISTTT25".

* Corresponding author.
E-mail address: banx@uw.edu (X.(J. Ban).

https://doi.org/10.1016/j.trc.2024.104577
Received 5 January 2024; Accepted 18 March 2024

is used for offline training (for machine learning (ML) or deep learning (DL) models) or online decision makings (e.g., traffic signal control, real-time TSEP, etc.). Data attacks, such as poisoning attacks, have been extensively studied in cybersecurity and related fields (El-Rewini et al., 2020). In transportation-related areas, research on data poisoning attacks has been mainly concentrated on vehicular data (e.g., spoofing on GPS and onboard camera/Lidar sensors) (Schmidt et al., 2016) and V2X data (e.g., attacks on communication channels and the recently developed security certificate system management (SCSM) (Hasan et al., 2020)), with limited research efforts on the infrastructure side (Feng et al., 2018). TSEP, e.g., is one of the most basic and extensively studied areas in traffic modeling. It largely relies on data collected at the infrastructure side and produces traffic measures (travel times, volumes, queue lengths, etc.), which are important both in their own merit (e.g., for performance evaluation) and as a critical input to control and optimization applications (e.g., traffic control). However, research on data poisoning-related attacks (and defense) on TSEP is sparse: such attacks have not been formally defined and very few TSEP methods currently consider the risks of potential data attacks.

Often formulated as a learning problem from data, many TSEP models are *optimization-based*, including classical linear/nonlinear optimization problems, and ML/DL-based problems. Examples include Least Squares (LS) models applied to delay pattern estimation of signalized intersections using travel times from mobile sensors (Ban et al., 2011), SVM applied to vehicle trajectories for vehicle classification (Sun et al., 2013) and freight delivery stop identification (Yang et al., 2014), Bayesian Network-based methods for estimating the cycle-by-cycle queue length distribution (Hao et al., 2014), and DL models for traffic state prediction (Li et al., 2020). Therefore, given the reliance of these TSEP models on data, a data attack could compromise TSEP solutions. The TSEP model under attack is also called the *victim* model. A *formal definition* of data poisoning attacks typically consists of multiple components, including adversarial objective/motivation, attack vector, attack model and attacker's capability (Reda et al., 2021). The *adversarial objective* defines what the adversary wants to achieve through the attack. The adversary normally tries to find a set of data perturbations that, after being applied to the pristine dataset, would maximize the negative impact on the TSEP models (e.g., the deviation from the pristine solution) or induce some pre-defined target (e.g., by minimizing the difference between the poisoned solution and the target). An *attack vector* is a means (or path) by which the adversary can gain access and manipulate a victim's system either physically or remotely. *Attack model* is to identify a specific strategy for how the attacker manipulates the dataset to reach the adversarial goal via the attack vector. A sophisticated algorithm is often needed to ensure a stealthy (i.e., small perturbations to data) and effective attack. *Attacker's capability* (or attack budget) specifies the resources that an attacker needs to launch a successful attack. Based on attackers' access and knowledge level of the target model, an attack could be a white-box (full access/knowledge) or black-box attack (no access/knowledge), with the grey-box in the middle. Without suitable resources, carrying out an attack can be difficult. Fig. 1 provides a simple example to help understand the attack components. The two data points and the fitted line represent the pristine traffic data and TSEP model, respectively. Assuming the data are collected wirelessly, the attacker compromises the V2X communication channel (i.e., *attack vector*) to poison data $\bar{x}_a$. The goal is to flatten the (negative) slope of the fitted line as much as possible (i.e., *adversarial objective*). A white-box attack is assumed so that the attacker has full knowledge of the model and data. Nevertheless, to be stealthy, the attack cannot change $\bar{x}_a$ wildly but in a limited region (i.e., *attacker's capability*). Given the limit, we identify an effective algorithm, i.e., the *attack model*, to reach the poison point $x_a^{'}$ that flattens the slope the most.

This study focuses on developing and analyzing the attack model given an adversarial objective under some regular (and common) assumptions of the attack vector and attacker's capability. These assumptions are closely related to real-world problems and are specified in Section 3.2. As noted above, an adversary objective is typically defined as maximizing or minimizing a specific function. Since learning the TSEP solution from data itself is an optimization problem, the attack model can be generally formulated as a *bilevel optimization* problem, where optimizing the adversarial objective and the TSEP model are the upper and lower problems, respectively. The attack model can then be derived by solving the bilevel problem. Despite the many successful attack models developed so far, there are *two major limitations* in most existing attack methods: the ability to deal with constraints especially inequality constraints and the ability to deal with non-differentiability. *Firstly*, there is limited attention on analyzing attack models in the presence of constraints that often exist in TSEP applications. This includes physical and domain-knowledge-related constraints that confine the TSEP solution and its interactions with the data such that the perturbations are carefully bounded to make the attack stealthy and difficult to detect. The

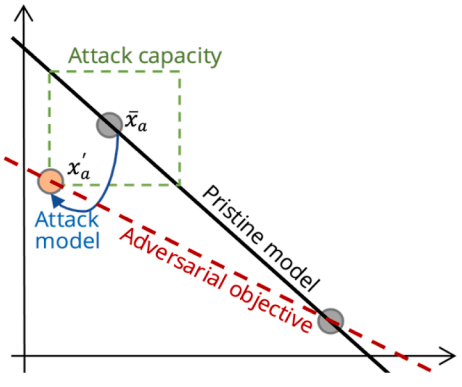

Fig. 1. Illustration of components of data poisoning attack.

existence of constraints renders it challenging to analyze the response of TSEP solutions to data changes, as an explicit mapping from data to solution could no longer be available in this case. *Secondly*, most existing methods assume the adversarial objective with respect to (w.r.t.) the data change is differentiable such that gradient method can be readily applied (Biggio et al., 2013). This may not be the case for TSEP models, whose solutions typically lack differentiability w.r.t. the data change, due to the presence of general constraints. Consequently, existing attack methods are not applicable, especially the commonly used gradient methods. When differentiability does not hold, the problem may only be characterized by properties such as Lipschitz continuity and sometimes combined with one-sided differentiability (e.g., semi-derivatives) (Dontchev and Rockafellar, 2009). Lipschitz continuity and semi-derivatives thus provide a more general methodological framework to define and design poisoning attacks, which however have not been discussed or applied so far.

In this paper, we first formulate data poisoning attacks as a general sensitivity analysis of parameterized optimization problems over parameter changes (i.e., data perturbations) and study the Lipschitz continuity property of the solution with the presence of general (equality and inequality) constraints. Then, we develop attack models that fit a broader spectrum of applications by extending optimization problems with no or equality constraints (widely used in the cybersecurity field) to those expressed as variation inequalities (VI) with general constraints. The generalized implicit function theorem for VI problems is utilized to compute the semi-derivatives that characterize how the TSEP solution responds to data perturbations. The semi-derivatives then enable us to evaluate TSEP models' vulnerability (at each data point) and solve the proposed attack model. Finally, we conduct numerical experiments using two TSEP applications to demonstrate the flexibility, generality and effectiveness of the proposed method.

2. Literature review

2.1. Threat of data poisoning attacks

Transportation systems are increasingly data-driven. On the vehicle side, for example, advanced driving systems rely on deep-learning networks trained on massive datasets to percept surroundings. On the infrastructure side, smart signal control systems utilize data from connected vehicles to optimize signal timing schemes. However, the increasing use of sensor technologies onboard and connectivity amplifies the exposure of potential vulnerabilities. Extensive studies have shown the vulnerability of transportation systems to data poisoning attacks (Almalki and Song, 2020; Feng et al., 2018). Unlike conventional data attacks such as deny-of-service or jamming attacks against sensors, data poisoning attacks can be stealthy as they are often carefully crafted and thus can be challenging to detect and defend (Jagielski et al., 2018).

Studies suggest that *vehicle sensors* are vulnerable to data poisoning attacks (Petit and Shladover, 2015). *GPS spoofing* injects false information into true GPS measurements, aiming to mislead vehicles' trajectories (Schmidt et al., 2016). *Data poisoning attacks* can disrupt the neural network models that camera-based modules depend on to identify objects in the path of vehicles (Xie et al., 2022). In scenarios where vehicles are connected via V2X communication, the effect of data poisoning attacks can go beyond the targeted vehicle and reach its neighboring vehicles. For example, in vehicle *platooning* scenarios, an attacker could break the platooning stability by falsifying vehicular data and broadcasting the falsified data via V2X communication (Dadras et al., 2015). Attacks can also be launched to create false jams with ghost vehicles on the roads, affecting the decisions of neighboring vehicles (Yu, 2021). There are also works demonstrating the threat of data poisoning attacks on vehicular ad hoc networks (Boeira et al., 2018).

Data poisoning-related research on the transportation infrastructure side is relatively sparse. Most existing TSEP models assume benign data (with possible considerations of data errors or even biases) and are thus vulnerable to data poisoning attacks. While pioneering and practically important, some existing data poisoning attacks may be related to but not directly focused on TSEP. They are limited to specific scenarios and applications. For example, attacks on signal control systems exploit the V2X communication and aim to manipulate V2X-based signal time schemes by poisoning trajectories of vehicles approaching an intersection (Feng et al., 2018). The spatial–temporal graph neural network models, in which the network-level traffic forecasting task is often formulated as, could also be vulnerable to slight adversarial perturbations. Liu et al. (2022), for example, proposed an adversarial spatiotemporal attack framework that generates imperceptible perturbations to geo-distributed data for spatiotemporal traffic forecasting attacks. The framework constructs gradient-based adversarial spatiotemporal examples in white-box and black-box settings to perturb the selected time-dependent victim nodes. Zhu et al. (2021) and Liu et al. (2023) devised algorithms to select a limited set of nodes of the graph and modify the input features in order to degrade the overall performance of the graph-based traffic prediction models.

2.2. Modeling data poisoning attack

Data poisoning attacks can be model-specific or domain-specific. Attacks that are not designed for a specific model or algorithm, such as random attacks and bandwidth attacks, have limited success rates. In the following, the two categories of attacks are briefly introduced, focusing on data poisoning attacks related to the transportation system.

Model-specific attacks are optimized for a specific machine-learning model or learning algorithm to achieve effective attack performance. Such attacks have been proposed and demonstrated on popular ML/DL methods, including SVM (Biggio et al., 2013), regression (Jagielski et al., 2018), unsupervised classifiers (Li and Vorobeychik, 2018), and neural networks (Papernot et al., 2016).

For *domain-specific* attacks, they could be more targeted by specifying the domain- or application-related goals and constraints. For example, researchers have developed data poisoning attack models for exploiting the vulnerability of sensing systems and dedicated data-processing or data-learning algorithms. Effective data attacks have been proposed to fail malware and network anomaly detection, spam filtering, computer vision, Portable Document Format (pdf) reader, and recommender systems (Huang et al., 2011; Vorobeychik and Kantarcioglu, 2018), to name a few. Another popular research area is to evaluate the vulnerability of smart power grids facing data poisoning attacks, as smart grid control system relies on an accurate state estimation from sensor measurements, which could be vulnerable (Liu et al., 2011).

Data poisoning attacks are often formulated as a *bi-level optimization problem*, which is commonly solved via *gradient-based* methods. Specifically, the attacker iteratively generates poisoning data points by perturbing data toward the direction that effectively approaches the adversarial objective. Such direction is identified by computing the gradient of the objective in terms of data perturbations. Yet, gradient-based methods have two major limitations: the ability to deal with constraints especially inequality constraints and the ability to deal with non-differentiability. *First*, there is limited attention to analyzing attack models in the presence of general constraints, which is often considered in practical (e.g., TSEP) applications. This includes physical and domain-knowledge-related constraints, which confine the model solution and its interactions with the data such that the perturbations are carefully bounded to make the attack stealthy and thus difficult to detect. The existence of constraints renders it challenging to analyze the response of model solutions to data changes, as an explicit mapping from data to solution could no longer be available. Note that gradient-based methods sometimes do handle constraints, which, however, are typical to bound adversarial noise (added to data) such that the perturbation is imperceptible. Such treatment is also adopted in our study by limiting changes in the data. However, what distinguishes our method from the previous gradient-based methods is that we consider general constraints involving the model solution. Taking LS regression as an example, the attack model may not only limit adversarial noise but also constrains the coefficients learned from data to respect traffic domain knowledge. As we show later, if the constraints involve the TSEP model solution, the calculated gradient may not represent the best direction to perturb the data when certain constraints are active (at a given data point). *Second*, most existing methods (Biggio et al., 2013) assume (implicitly or explicitly) the objective function or constraints are differentiable with respect to data changes, such that gradient methods can be readily applied. This assumption is not true in general: it is well-known that the solution of an optimization model is often continuous but not differentiable with respect to data changes (Luo et al., 1996). Consequently, the existing attack methods, especially the commonly used gradient methods, are not strictly applicable. In this case, one needs to rely on weaker conditions such as the Lipschitz continuity and the existence of one-sided differentiability (e.g., semi-derivatives) (Dontchev and Rockafellar, 2009) to define and analyze attack models.

Many existing poisoning attacks are *objective-driven*, where the attacker has a specified adversarial objective, such as reducing the forecasting/classification accuracy of the attacked model, evaluated using certain benchmark data. Recently, Suya et al. (2021) proposed a *model-targeted* attack. Given a target model (i.e., assuming the adversarial knows where to induce the learning model), the proposed model-targeted attack aims to add a set of poisoning data iteratively to induce the victim model to that target model as close as possible. Each iteration adds a data point that maximizes the loss difference between the target and intermediate models (i.e., the one learned on the clean data and poisoned points from previous iterations). It was proven that the iteration process can eventually induce a model similar to the target model. Such model-targeted attacks have several advantages. It breaks the attack into two steps: finding the target model first and then generating poisoning points to induce the target model. This helps simplify the design of effective attacks for various objectives. It was shown that model-targeted attacks could lead to more effective poisoning attacks than objective-driven models. Existing model-targeted attack models, however, work under strong assumptions by focusing on convex and non-constrained learning problems. This study proposes to generalize existing model-target attacks and addresses the two limitations noted above, by developing new methods/tools that rely on weaker conditions.

3. Methodology

3.1. Overview

This section first formulates data poisoning attacks as the sensitivity analysis of optimization problems with respect to data perturbations (Section 3.2). In TSEP models where the optimization problem is typically constrained or the solution lacks differentiability, studying how the data perturbation affects the solution can be challenging. To address the challenge, a general framework is presented to analyze the Lipschitz properties of the TSEP solution with respect to the data change (Section 3.3). By treating a TSEP as a parameterized version of a nonlinear optimization problem, the framework studies the behavior of the TSEP solution under the perturbation of the parameters (i.e., data change). Due to the above-mentioned challenges, classical gradient-based methods are not appliable. This study shows how the generalized implicit function theorem (Dontchev and Rockafellar, 2009) for constrained optimization can address the challenge and enables us to compute the *semi-derivatives*. Based on the semi-derivatives, we develop an attack algorithm to effectively poison the data (Section 3.4).

3.2. TSEP model and data poisoning attack

Without loss of generality, we model a TSEP problem under perturbations as a parameterized nonlinear program, which can be written as follows.

$$P(x) = \underset{y}{\operatorname{argmin}}\ g_0(x,y)$$
$$s.t.\ g_i(x,y)\begin{cases}\le 0 & \text{for } i \in [1,r]\\ = 0 & \text{for } i \in [r+1,m]\end{cases} \tag{1}$$

Here, $P(x)$ is the TSEP model solution set (e.g., the coefficients of separating hyperplane in SVM). $x \in \mathbb{R}^d$ is the data and modeled as the parameter (e.g., data perturbation). Notice that both inequality and equality constraints are included for generality. The TSEP objective $g_0(x,y)$ is often a loss function.

As illustrated in Fig. 1 (see Section 1), in an adversarial setting, attackers can craft certain poisoned samples, so that input to the TSEP model can be corrupted. Consequently, the estimated/predicted state is compromised and deviates from the one from the pristine data. The attack model carefully crafts a set of perturbations so that the adversarial goal can be achieved while meeting specific constraints. The process of finding such perturbations is formulated as solving an optimization problem. Denoting $G(x,\hat{y}(x))$ the adversarial goal, where $\hat{y}(x) \in P(x)$ is a solution of problem (1), the attack model can be formulated as:

$$\max_{x}\ G(x,\ \hat{y}(x))$$
$$s.t.\quad |x-\bar{x}| < \delta\ \&\ other\ application-specific\ constraints$$
$$\hat{y}(x) \in P(x) \tag{2}$$

Here $\bar{x}$ and x are the pristine and poisoned data, respectively. $|x-\bar{x}| \le \delta$ states that the data perturbations shall be small; the threshold δ would be determined according to the adversary's capability or the desire to confine the magnitude of perturbations for producing a stealthy attack (i.e., not trigger detection algorithms such as outlier-based anomaly detection methods). Other perturbation-related constraints may vary, e.g., only a subset of the data points may be perturbed, or traffic-related constraints (e.g., car-following) or knowledge should be satisfied. Since $\hat{y}(x) \in P(x)$ itself is an optimization problem, the attack model is mathematically a bilevel problem.

The proposed attack model (2) is more flexible than existing data attacks that target at either the learning performance (e.g., loss evaluation on the testing dataset) or achieving a target model (Koh et al., 2021; Suya et al., 2021). If the goal is to maximize the deviation after the attack, we could specify and maximize the objective function $G(x,\hat{y}(x)) = |\hat{y}(x)-\bar{y}|$, with $\bar{y}$ being the pristine solution derived from pristine data. If the goal is to induce the model toward an adversarial target y^*, we could specify and minimize the objective function $G(x,\hat{y}(x)) = -|\hat{y}(x)-y^*|$.

This study assumes that an adversary is capable of gaining full knowledge of training data and model (i.e., white-box attack scenario). Though this assumption may be unrealistic for many scenarios, it allows one to focus on a particular aspect of attacks and is applied in many prior works (Biggio et al., 2013; Koh et al., 2021; Mei and Zhu, 2015; Shafahi et al., 2018). Furthermore, results from such a setting represent the worst-case outcomes, which could be valuable for evaluating learning models' vulnerability. Regarding the attack vector, we focus on attacks *perturbing existing points*, assuming, for example, the V2I communication channel for data transmission in TSEP applications is compromised or certain vehicles or devices are attacked from which data are collected. In Appendix A.10, we show that the proposed method can be adapted for *addition-type attacks* that add erroneous data points to existing data (Suya et al., 2021).

3.3. Lipschitz property of TSEP models

For the attack model introduced above, it is crucial to study the sensitivity of the TSEP solution set $P(x)$ w.r.t. the changes in x. This is the classical (and challenging) sensitivity analysis of optimization problems (Fiacco, 1983). Due to the challenges associated with TSEP models, this study focuses on the Lipschitz continuity property of the solution set. Assuming x is in a δ-neighborhood of $\bar{x}$, i.e., $|x-\bar{x}| \le \delta$, and K (the Lipschitz constant) is a finite value, we have the following definitions (Ω a unit ball):

- If $P(x)$ is single-valued: $P(x)$ is Lipschitz continuous around $\bar{x}$ if $|P(x)-P(\bar{x})| \le K|x-\bar{x}|$.
- If $P(x)$ is set-valued: $P(x)$ is Lipschitz continuous around $\bar{x}$ if $P(x) \subset P(\bar{x}) + K|x-\bar{x}|\Omega$.

Note that the above definitions omit some technicalities to make them more concise; one can refer to Dontchev and Rockafellar (2009) for more precise definitions. Understanding the Lipschitz property of $P(x)$ is fundamental to both data poisoning attacks and defenses. For *data attacks*, if a learning model is not Lipschitz continuous, the solution change with respect to the input data (parameter) change can be explosive (unbounded), which will be disastrous when the model is under data poisoning attacks. For *defenses*, "security by design" is the first critical step to defend poisoning attacks by ensuring that the learning model is Lipschitz continuous so that the solution change is bounded when input data changes; see more discussions on this in Section 5. Here we assume this security by design step has been exercised and the resulting learning model is Lipschitz continuous (see Section 4). However, a

Lipschitz continuous model is not (poisoning) attack free, as shown in the numerical results in Fig. 7 – Fig. 10 in Section 4. This necessitates formal investigations on poisoning attacks on Lipschitz continuous models.

In the following, we briefly introduce the method for analyzing the Lipschitz property, focusing the cases when $P(x)$ is a singleton, i.e., the TSEP model has a locally unique solution.

3.3.1. KKT mapping of the TSEP model

The first-order conditions of the TSEP model defined in Eq (1) can be cast as a variational inequality (VI) as:

$$\begin{cases} F(x,y,\lambda) = \left(\nabla_y L(x,y,\lambda), -\nabla_\lambda L(x,y,\lambda)\right)^\top, \\ E = R^t \times \left[R^r_+ \times R^{m-r}\right]. \end{cases} \tag{3}$$

Here, $L(x,y,\lambda)$ is the Lagrangian function associated with problem (1) with λ being the multipliers. $N_E(y,\lambda)$ is the normal cone operator. The pairs (y,λ) satisfying this VI are the KKT pairs for the problem specified by x in problem (1). If data x is perturbed, we wish to learn more about its influence on the behavior of y by studying the KKT mapping S that is defined by

$$S(x) = \{(y,\lambda) | \text{satisfying the VI in (3)}\} \tag{4}$$

Note that the VI-formed first-order optimality condition in Eq. (3) generalizes optimization problems studied in data poisoning attacks especially those with no or only equality constraints. In the latter case, one can easily verify that Eq. (3) reduces to the classical optimality condition (i.e., $\nabla_y L(x,y,\lambda) = 0,\ -\nabla_\lambda(x,y,\lambda) = 0$), since $N_E(y,\lambda) = \{0\}$.

3.3.2. Auxiliary problem

Directly studying the connection between perturbation in x and changes in y via (3) above can be challenging. It is helpful to consider an auxiliary (approximated) problem with respect to a choice of data $\bar{x}$ and its solution mapping $(\bar{y},\bar{\lambda}) \in S(\bar{x})$. This auxiliary problem is constructed as:

$$\begin{aligned} &\min_{\Delta y}\ \bar{g}_0(\Delta y) - \langle v, \Delta y\rangle \\ &\text{s.t.}\quad \bar{g}_i(\Delta y) + u_i \begin{cases} = 0 \text{ for i} \in \mathrm{I}\backslash\mathrm{I}_0, \\ \leqslant 0 \text{ for i} \in \mathrm{I}_0, \\ \text{unrestricted for i} \in \mathrm{I}_1. \end{cases} \end{aligned} \tag{5}$$

with the three index sets I, I_0 and I_1 defined as,

$$\begin{aligned} &\mathrm{I} = \{\mathrm{i} \in [1,\mathrm{m}] | \mathrm{g_i}(\bar{\mathrm{x}},\bar{\mathrm{y}}) = 0\} \\ &\mathrm{I_0} = \{\mathrm{i} \in [1,\mathrm{r}] | \mathrm{g_i}(\bar{\mathrm{x}},\bar{\mathrm{y}}) = 0 \& \bar{\lambda}_\mathrm{i} = 0\} \subset \mathrm{I} \\ &\mathrm{I_1} = \{\mathrm{i} \in [1,\mathrm{r}] | \mathrm{g_i}(\bar{\mathrm{x}},\bar{\mathrm{y}}) < 0\} \end{aligned}$$

And $\bar{g}_0(\Delta y) = L(\bar{x},\bar{y},\bar{\lambda}) + \langle \nabla_y L(\bar{x},\bar{y},\bar{\lambda}), \Delta y\rangle + \frac{1}{2}\langle \Delta y, \nabla^2_{yy} L(\bar{x},\bar{y},\bar{\lambda})\Delta y\rangle$ is a second-order expansion of L and $\bar{g}_i(\Delta y) = g_i(\bar{x},\bar{y}) + \langle \nabla_y g_i(\bar{x},\bar{y},\bar{\lambda}), \Delta y\rangle$ is a first-order expansion of constraint function g_i, both at the given TSEP solution $(\bar{y},\bar{\lambda})$ for given data $\bar{x}$. (u,v) are two input vectors which need to be specified when defining Eq (5). Δy denotes the perturbation to the solution y that needs to be derived from Eq (5). That is, Eq (5) is a local approximation of the TSEP model around its solution $(\bar{x},\bar{y},\bar{\lambda})$ and serves as a bridge that helps connect the data perturbation in x with changes in the solution $\hat{y}$. The derivation is based on the KKT solution mapping of (5), which is expressed in the *parameterized VI* below.

$$\begin{pmatrix} \nabla_{\Delta y} L_{aux}(\Delta y, z, v, u) \\ -\nabla_{z_i} L_{aux}(\Delta y, z, v, u) \text{for } i \in [1,m] \end{pmatrix} + N_{\bar{E}}(\Delta y, z) \ni \begin{pmatrix} 0 \\ 0 \end{pmatrix}. \tag{6}$$

Here, z contains the multipliers for the constraints in Eq (5). $L_{aux}(\Delta y, z, v, u) = \bar{g}_0(\Delta y) - \langle v, \Delta y\rangle + \sum_{i=1}^m z_i\left[\bar{g}_i(\Delta y) + u_i\right]$ is the Lagrangian function with $\bar{E} = \mathbb{R}^n \times W$. W is the space of the multipliers: $z \ni W \Leftrightarrow z_i \begin{cases} \geq 0 & \text{for } i \in I_0, \\ = 0 & \text{for } i \in I_1. \end{cases}$

The parameterized VI in (6) is equivalent to:

$$\begin{aligned} &\nabla^2_{yy} L(\bar{x},\bar{y},\bar{\lambda})\Delta y + \sum_{i=1}^m z_i \nabla_y g_i(\bar{x},\bar{y}) - v = 0, \\ &\text{with } z_i \begin{cases} \geq 0 & \text{for } i \in I_0 \text{ having } \bar{g}_i(\Delta y) + u_i = 0, \\ = 0 & \text{for } i \in I_0 \text{ having } \bar{g}_i(\Delta y) + u_i < 0 \text{ and for } i \in I_1. \end{cases} \end{aligned} \tag{7}$$

Note that for $i \in I\backslash I_0$, z_i is unrestricted and the first order conditions require $z_i[\bar{g}_i(\Delta y) + u_i] = 0$. Then, Eq (7) specifies *the auxiliary solution mapping* $\bar{S}$:

$$\bar{S}(v,u) = \{(\Delta y, z) | \text{satisfying the requirement in Eq(7)}\} \tag{8}$$

which has $(0,0) \in \bar{S}(0,0)$.

3.3.3. Generalized implicit function theorem

The behaviors of the solution mapping S can be studied with $\bar{S}$ using the generalized implicit function theorem. The theorem directly follows Theorem 2G.9 in (Dontchev and Rockafellar, 2009) and the proof is omitted here.

Theorem 1. *Suppose the optimization model expressed (1) with twice continuously differentiable functions g_i and the following conditions hold:*

i. *The gradients $\nabla_y g_i(\bar{x},\bar{y})$ for $i \in I$ are linearly independent.*

ii. $\langle \Delta y, \nabla^2_{yy} L(\bar{x},\bar{y},\bar{\lambda})\Delta y\rangle > 0$ *for every nonzero $\Delta y \in M^+$, where M^+ is a subspace defined as:*

$$M^+ = \left\{\Delta y \in R^n \middle| \Delta y \perp \nabla_y g_i(\bar{x},\bar{y}) \text{ for all } i \in I \backslash I_0\right\}$$

Then, the solution set of (1) $S(x)$ has a Lipschitz continuous single-valued localization s around $\bar{x}$, and this localization s is semi-differentiable at $\bar{x}$ with semi-derivative given by $Ds(\bar{x},\hat{y}(x))(\Delta x) = \bar{s}(-B\Delta x)$, *with* $B = \begin{pmatrix} \\ \nabla^2_{yx} L(\bar{x},\bar{y},\bar{\lambda}) - \nabla_x g_1(\bar{x},\bar{y}) \vdots - \nabla_x g_m(\bar{x},\bar{y}) \\ \\ \end{pmatrix}$,

where $\bar{s}$ is a Lipschitz continuous single-valued localization around (0, 0) for (0, 0) in mapping $\bar{S}$. Here, B is the Hessian matrix of the Lagrangian and the gradient of the constraints of the learning problem (1) evaluated at $\bar{x}$. When computing the semi-derivative $Ds(\bar{x},\hat{y}(x))(\Delta x)$, we first have matrix $-B$ multiplying Δx (the perturbation in data) to produce the perturbations to the auxiliary objective (v) and constraints (u). Then, the solution of the auxiliary problem (5) with the input (v,u) will yield $(\Delta y, z)$, which is also the semi-derivative as shown in the theorem. Δy then gives the changes in model solution y due to data perturbation Δx: $D\hat{y}(x,\hat{y}(x))(\Delta x)$. This is the exact meaning of how the auxiliary problem serves as a bridge allowing us to evaluate the changes of TSEP solution over data perturbation. Note that Theorem 1 also establishes that given the specific conditions, the TSEP model has a locally unique solution.

3.4. General attack strategy

We develop an attack strategy following the Lipschitz analysis and the computed semi-derivative above. The basic idea is given by Algorithm 1 (Fig. 2). It sequentially applies perturbations to the pristine dataset. In each iteration, we attack one data point by identifying the optimal direction that has the maximum effect in achieving the adversarial objective (Step 1 and Step 2 in Fig. 2) and perturbing it following the identified direction (Step 3 in Fig. 2). The TSEP model is relearned on the perturbed data after each iteration (Step 4 in Fig. 2) until convergence conditions are met. By repeating this process, we can gradually reach the adversarial objective (e.g., either maximizing the deviation of the TSEP solution or obtaining a target TSEP model specified by the attacker). Hereafter, we focus

Algorithm 1. Data Poisoning Attack Algorithm

Input: The clean data $x = [x_i]$ $(i \in [1,\ldots,n])$ and TSEP model.

Repeat:

1. For $i \in N$ at iteration j, evaluate semi-derivative $DG\left(x^j, \hat{y}(x^j)\right)(\Delta x_i)$ given a unit perturbation Δx_i, and find the optimal perturbation direction Δx_i^*
$$\Delta x_i^* = \underset{\Delta x_i}{\operatorname{argmax}}\, DG\left(x^j, \hat{y}(x^j)\right)(\Delta x_i)$$
2. Select data point a that is the most sensitive to the perturbation
$$a = \underset{i}{\operatorname{argmax}}\, DG\left(x^j, \hat{y}(x^j)\right)(\Delta x_i^*)$$
3. Perturb the target point a following direction Δx_a^* by a specific step size η
$$x_a^{j+1} = x_a^j + \eta \Delta x_a^*.$$
4. Re-learn the TSEP model on the poisoned data and evaluate the attack performance.

Until: Improvement in the attack performance is sufficiently small or constraints are violated.

Output: Poisoned data and TSEP model

Fig. 2. Semi-derivative-based Data Poisoning Attack Algorithm.

on maximizing an adversarial objective function for clarity; the algorithm could be easily adapted for minimizing it, if needed. More details are provided below. A theoretical analysis of the convergence behaviors of the proposed attack model can be found in the authors' recent work (X. Wang et al., 2023).The key to Algorithm 1 is to compute $DG(x, \hat{y}(x))(\Delta x)$, i.e., the change of $G(x, \hat{y}(x))$ w.r.t. perturbation Δx. By the chain rule, it can be computed as:

$$DG(x, \hat{y}(x))(\Delta x) = \nabla_x G(x, \hat{y}(x)) \bullet \Delta x + \nabla_{\hat{y}} G(x, \hat{y}(x)) \bullet D\hat{y}(x, \hat{y}(x))(\Delta x) \tag{9}$$

Eq (9) says that the data perturbation impacts the solution *explicitly* via perturbed data x (the first addition term) and *implicitly* via shifting the learned $\hat{y}$ (the second addition term). In scenarios where the outer objective G does not involve data, the first term is dropped. The derivation of $\nabla_x G$ and $\nabla_{\hat{y}} G$ takes advantage of the fact that G is often an explicit function of x and $\hat{y}(x)$. The challenging step of finding how the TSEP solution responds to data changes $D\hat{y}(x, \hat{y}(x))(\Delta x)$ is handled by Theorem 1.

Although similar iterative attack strategies have been widely adopted in previous studies (Biggio et al., 2013; Jagielski et al., 2018; Mei and Zhu, 2015; Xiao et al., 2015a), our attack introduces *multiple contributions*. First, most attack models are designed for a specific objective, while our attack is more general in terms of achieving different types of objectives expressed by $G(x, \hat{y}(x))$. Second, unlike the existing algorithms that assume differentiability and rely on the gradient to find perturbation direction, our method handles problems with general constraints or lacking differentiability and computes the semi-derivative to identify the perturbation direction.

4. TSEP applications

We test the proposed methods by applying them to two specific TSEP models, including the LS regression model (e.g., for queue-length estimation) and support vector machine (SVM). The two models are among the most widely used learning models in science / engineering applications. They also allow us to test the generality of the proposed methods. The former is regression-based, where both the feature values and the responses of data samples are optimized to launch an attack; it is mainly for real-time traffic state estimation with its solution only involving the learned parameters. In contrast, the latter is classification-based, where typically only the feature values are perturbed and optimized to launch a poisoning attack; it is often for traffic state prediction. Though simple, the LS regression model and SVM model are and will continue to be widely applied in transportation applications due to their efficiency, simplicity of use, and effectiveness (Ferrari Dacrema et al., 2019; Tramèr and Boneh, 2021). Therefore, understanding the vulnerability of these fundamental and high-impact models will have important practical significance and shed useful insights on other TSEP models.

For each of the two models, this section first analyzes its Lipschitz properties and then implements the attack model. The former uses mobile sensing data for estimating delay patterns and queue lengths of a signalized intersection, and the latter uses mobile sensing data to classify vehicles (cars vs. trucks) in a traffic stream. These data are real-world datasets and are among the first to apply machine learning (ML) methods on mobile sensing data to estimate key traffic states and performances. The first dataset for linear regression was collected at a signalized intersection (Ban et al., 2011). The data contains travel time information between two consecutive locations at the upstream and downstream of the intersection, including the arrival (start) time of a vehicle at the upstream location and the travel time through the intersection. The second dataset for SVM classification was extracted from real-world vehicle GPS trajectories, including the speed and acceleration-/deceleration-related information. We also conduct additional experiments by applying the method to the Next Generation SIMulation (NGSIM) dataset (DOT, 2018), which is well-known in transportation and contains millions of detailed trajectory data consisting of different types of vehicles, including cars and trucks; detailed results are in Appendix A.9.

We show that applying the proposed general attack model to specific applications involves specifying the adversarial goal, the learning problem of the TSEP model (e.g., regression or classification), and the problem-specific constraints (e.g., those related to traffic and vehicle dynamics). The flexibility of the proposed attack model is tested using these two models under different attack settings: the LS model is attacked to *maximize the deviation* while the SVM model is attacked with a *target-based attack setting*, where a target SVM model is specified as the adversarial objective. For the experiments, we compare the performance of the proposed methods with the state-of-the-art methods. Numerical results are then reported and discussed.

4.1. LS regression model-based queue length estimation

4.1.1. Lipschitz analysis of LS regression model

An LS regression problem can be expressed in the following general form:

$$\begin{cases} \min\limits_{y=(\boldsymbol{w},b)} & \dfrac{1}{2n}\sum\limits_{i=1}^{n}\left(b + \boldsymbol{w}^\top \boldsymbol{x}_i^{ind} - x_i^{dep}\right)^2 \\ \text{s.t} & \boldsymbol{\beta} y - \boldsymbol{\rho} \le 0 \end{cases} \tag{10}$$

where $\boldsymbol{x}_i = (\boldsymbol{x}_i^{ind}, x_i^{dep})$ represents one of the n data points with $\boldsymbol{x}_i^{ind}$ being a vector of independent variables of length k and x_i^{dep} being the dependent variable. The independent variables are feature-wise normalized such that $\frac{1}{n}\sum_{i=1}^{n} x_i^{ind_j} = 0$ (for independent variable $j = 1, \ldots, k$). $\boldsymbol{\beta} \in \mathbb{R}^{m\times(k+1)}$ and $\boldsymbol{\rho} \in \mathbb{R}^m$ specify a set of m linear constraints on y.

Lipschitz analysis Provided the Lagrangian function $L(\boldsymbol{x}, y, \lambda) = \frac{1}{2n}\sum_{i=1}^{n}(b + \boldsymbol{w}^\top \boldsymbol{x}_i^{ind} - x_i^{dep})^2 + \sum_{h=1}^{m}\lambda_h(\boldsymbol{\beta}_h^\top y - \rho_h)$, we have the auxiliary problem below following Eq (5).

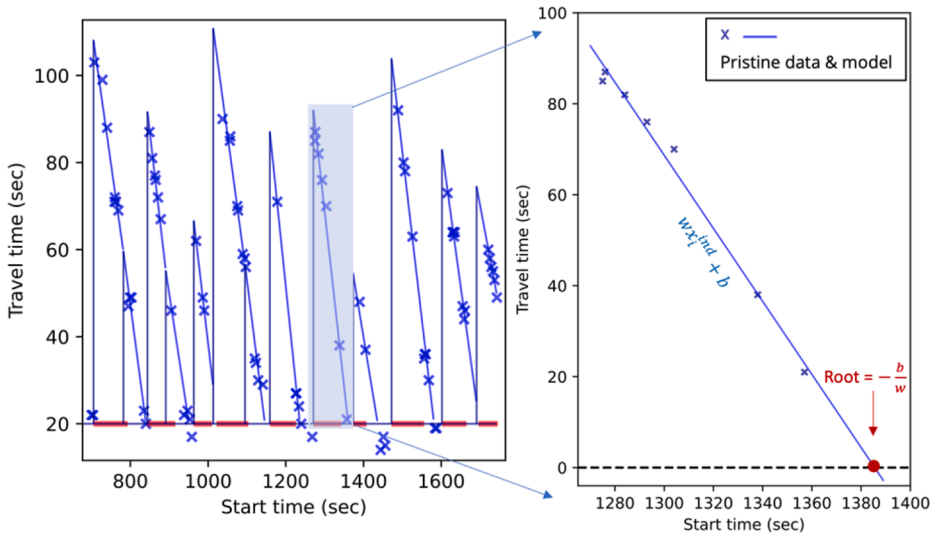

Fig. 3. Illustration of fitting data in each cycle with a line to compute queue length.

$$\min_{\Delta y=(\Delta w,\Delta b)} L(\bar{x},\bar{y},\bar{\lambda}) + \frac{1}{2}\left\langle \Delta y, \begin{bmatrix} \Sigma & 0 \\ 0 & 1 \end{bmatrix} \Delta y \right\rangle - \langle v, \Delta y \rangle$$
$$s.t. \quad \boldsymbol{\beta}_h^\top \bar{y} - \rho_h + \boldsymbol{\beta}_h^\top \Delta y + u_h \begin{cases} = 0 & \boldsymbol{\beta}_h^\top \bar{y} - \rho_h = 0 \,\&\, \bar{\lambda}_h > 0, \\ \le 0 & \boldsymbol{\beta}_h^\top \bar{y} - \rho_h = 0 \,\&\, \bar{\lambda}_h = 0. \end{cases} (h \in [1, \cdots, m]) \tag{11}$$

Here, $\Sigma = \frac{1}{n}\sum_{i=1}^{n} x_i^{ind} x_i^{ind\top}$ is a covariance matrix if we have the data feature-wise normalized (i.e., $\mu_j = \frac{1}{n}\sum_{i=1}^{n} x_i^{ind_j} = 0$, $j \in [1, \cdots, k]$). Then, following Equations (6)-(7), the parameterized VI corresponding to the auxiliary problem leads to the following first-order conditions.

$$\begin{bmatrix} \Sigma & 0 \\ 0 & 1 \end{bmatrix} \Delta y + \sum_{h=1}^{m} z_h \beta_h - v = 0,$$
$$\text{w.t. } z_i \begin{cases} \ge 0 & \text{if } \beta_h^\top \bar{y} - \rho_h = 0 \text{ with } \bar{\lambda}_h = 0 \,\&\, \beta_h^\top \Delta y + u_h = 0, \\ = 0 & \text{if } \beta_h^\top \bar{y} - \rho_h = 0 \text{ with } \bar{\lambda}_h = 0 \,\&\, \beta_h^\top \Delta y + u_h < 0, \text{ or } \beta_h^\top \bar{y} - \rho_h < 0, \\ \text{free} & \text{if } \beta_h^\top \bar{y} - \rho_h = 0 \text{ with } \bar{\lambda}_h > 0 \text{ then from } KKT,\ z_h\left(\beta_h^\top \Delta y + u_h\right) = 0. \end{cases} \tag{12}$$

We then have the auxiliary solution mapping $\bar{S}(v,u) = \{(\Delta y, z) | \text{ satisfying the requirement in Eq (12)}\}$. Following Theorem 1, we examine the two sufficient conditions to check whether the mapping is Lipschitz continuous w.r.t. data perturbations. In Appendix A.1, our analysis suggests that the two conditions will be satisfied if the vectors $\{\boldsymbol{\beta}_h | \boldsymbol{\beta}_h^\top \bar{y} - \rho_h = 0, h \in [1, \ldots, m]\}$ are linearly independent.

Assuming data point a is perturbed, $D\widehat{y}(x, \widehat{y}(x))(\Delta x_a)$ is computed by providing input $-B\Delta x_a$ to Eq (12). Here, $B = -[\boldsymbol{\kappa}_1, \boldsymbol{\kappa}_2, 0]^T$, with $\boldsymbol{\kappa}_1 = \frac{1}{n}\left[\bar{x}_a^{ind}\bar{w}^\top + (\bar{b} + \bar{w}^\top \bar{x}_a^{ind} - \bar{x}_a^{dep})\boldsymbol{I}_k - \bar{x}_a^{ind}\right]$ and $\boldsymbol{\kappa}_2 = \frac{1}{n}[\bar{w}^\top - 1]$.

4.1.2. Attack queue length estimation model

The queue length estimation model uses mobile sensing data collected at a signalized intersection (Ban et al., 2011). Denote x_i^{ind} as the arrival (start) time of vehicle i at the upstream location and x_i^{dep} as the travel time through the intersection (Fig. 3A). To estimate queue lengths using the measured travel time and the arrival time from mobile sensors, we first break data into cycles and estimate the (piecewise) linear delay patterns in each cycle (i.e., $x_i^{dep} \sim b + w x_i^{ind}$). Following the original study (Ban et al., 2009), at each cycle, the estimated queue length Len_{queue} is proportional to the root of the fitted line $-\frac{b}{w}$, i.e., $Len_{queue} \propto -\frac{b}{w}$ (Fig. 3B).

Attack model. Denoting $\widehat{y} = (\widehat{w}, \widehat{b})$ and $\bar{y} = (\bar{w}, \bar{b})$ as the estimates from the poisoned and pristine data, respectively, we can write the attack model as below.

$$\left.\begin{aligned} \min_{x} \quad & G = \left\| \frac{\widehat{b}}{\widehat{w}} - \frac{\bar{b}}{\bar{w}} \right\|_2^2 \\ \text{s.t.} \ & |x - \bar{x}| \le \delta \\ & x_l^{ind} - x_a^{ind} \le hw \\ & x_a^{ind} - x_f^{ind} \le hw \\ & \left(x_l^{ind} + x_l^{dep}\right) - \left(x_a^{ind} + x_a^{dep}\right) \le hw \\ & \left(x_a^{ind} + x_a^{dep}\right) - \left(x_f^{ind} + x_f^{dep}\right) \le hw \end{aligned} \right\} \text{Upper level}$$

$$\left.\begin{aligned} & (\widehat{w}, \widehat{b}) = \underset{(w,b)}{\operatorname{argmin}} \frac{1}{2n} \sum\nolimits_i \left(b + w x_i^{ind} - x_i^{dep}\right)^2 \\ & \text{s.t.} \begin{cases} g_1 : \rho_1 - w \le 0 \\ g_2 : w - \rho_2 \le 0 \end{cases} \end{aligned} \right\} \text{Lower level} \tag{13}$$

That is, the adversarial objective is to maximize the deviation of Len_{queue} by perturbing x. Here, given $Len_{queue} \propto -\frac{b}{w}$, the objective of maximizing the deviation of queue length estimation from the actual queue length $|\widehat{Len}_{queue} - \bar{Len}_{queue}|$ is equivalent to maximizing $|\frac{\widehat{b}}{\widehat{w}} - \frac{\bar{b}}{\bar{w}}|$. Alternatively, we could formulate a target model-based attack by specifying a target queue length and setting the adversarial objective as inducing Len_{queue} to the target. As noted earlier, we test such flexibility on the SVM-based vehicle classification model. g_1 and g_2 in the lower-level problem are to bound w, so that the discharge rate of queues is reasonable for producing a stealthy attack. Assuming vehicle a is under attack, we have constraints on (x_a^{ind}, x_a^{dep}) in the upper level to respect the basic car-following behavior with hw being the headway. Details of deriving these constraints are provided in Appendix A.2. The constraints can certainly be extended to include more physical and/or knowledge-based constraints when constructing the attack model.

Lipschitz continuity and attack algorithm For the queue length estimation model, it can be verified that $\widehat{y}$ is indeed Lipschitz continuous w.r.t. data perturbations. See details in Appendix A.3. In order to implement our attack following Algorithm 1, we can also follow Eq (9) to compute the semi-derivative $DG(x, \widehat{y}(x))(\Delta x_a)$.

$$DG(x, \widehat{y}(x))(\Delta x_a) = \left[\frac{\bar{b}}{\bar{w}^2} - \frac{1}{\bar{w}} \right] \Delta y \tag{14}$$

The solution to Δy depends on the status of constraints at $(\bar{x}, \bar{y}, \bar{\lambda})$. See Appendix A.4 for details. In summary, if no constraint is active, we have:

$$DG(x, \widehat{y}(x))(\Delta x_a) = d^1_{\Delta x}(\bar{x}) \bullet \Delta x_a = \frac{-1}{n\bar{w}} \left(\frac{\bar{b}}{\bar{w}\Sigma} \left[\bar{w} \bar{x}_a^{ind} + \left(\bar{w} \bar{x}_a^{ind} + \bar{b} - \bar{x}_a^{dep} \right) - \bar{x}_a^{ind} \right] + \left[-\bar{w} \ 1 \right] \right)^T \Delta x_a \tag{15}$$

Otherwise, if either g_1 or g_2 is active, there are two groups of semi-derivatives depending on Δx_a, and $DG(x, \widehat{y}(x))(\Delta x_a)$ can be either computed by $d^1_{\Delta x}(\bar{x}) \bullet \Delta x_a$ or $d^2_{\Delta x}(\bar{x}) \bullet \Delta x_a$ below.

$$DG(x, \widehat{y}(x))(\Delta x_a) = d^2_{\Delta x}(\bar{x}) \bullet \Delta x_a = \frac{-1}{n\bar{w}} \left[\bar{b} \ 1 \right]^T \Delta x_a \tag{16}$$

When implementing Algorithm 1 if either g_1 or g_2 is active, we compare the impacts of two unit perturbations (i.e., $\frac{d^1_{\Delta x}(\bar{x})}{|d^1_{\Delta x}(\bar{x})|}$ and $\frac{d^2_{\Delta x}(\bar{x})}{|d^2_{\Delta x}(\bar{x})|}$) and the one leading to the larger impact is applied.

Note that the gradient-based method evaluates Eq (15) only, ignoring the effects of constraints (Jagielski et al., 2018). This could be misleading when constraints are active, leading to less effective attack performance (see discussions below).

4.1.3. Effectiveness of the attack model

We demonstrate the effectiveness of the proposed attack model by sequentially perturbing the pristine data points. Fig. 4(A) shows the optimal semi-derivative for each point together with its impact (indicated by the arrow length) for the first iteration of running Algorithm 1. The results suggest that some points are sensitive to perturbations while others are not (also see Eq (15)): attacking points (vehicles) at the queue's head or tail leads to significantly larger impacts than attacking against other vehicles in the queue.

The proposed attack (referred to as *semi-derivative method* for convenience) is then compared with the state-of-the-art gradient-based attacks[1] and Fig. 4(B) compares their convergence of the adversarial objective. These attacks are also compared with randomly generated noises as perturbations. As discussed in Section 2.2, the gradient method iteratively generates poisoning data points by perturbing randomly selected data toward the direction that is identified by computing the gradient of the objective in terms of data

[1] Inequality constraints in the TSEP model are ignored by such methods in order to compute the gradients.

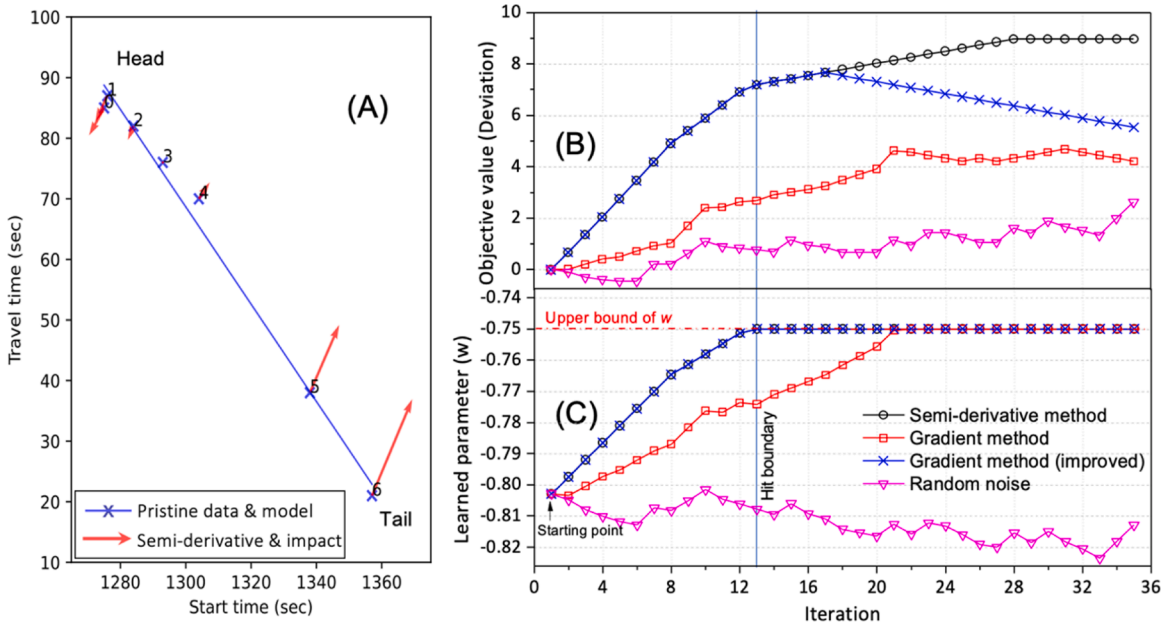

Fig. 4. (A) Semi-derivatives and their impacts; (B) Convergence of adversarial objective when perturbing queue length estimation model; (C) Changes in the learned parameter (w).

perturbations. In general, the classical implementation of the gradient method (referred to as the *classical gradient method*) is different from the semi-derivative method (Algorithm 1) in two aspects: 1) it randomly selects a point as the target; 2) it perturbs the target point following the gradient (Jagielski et al., 2018). Additionally, for a more fair comparison, we improve the classical gradient method so that the implementation is in line with Algorithm 1. Specifically, the *improved gradient method* takes a similar data selection procedure as the semi-derivative method to select the target point with the largest gradient. Then, the only difference lies in the perturbation direction (i.e., following the direction of the gradient instead of the semi-derivative).

It can be observed from Fig. 4(B) that adding noises to data randomly is ineffective to achieve the adversarial objective. Our attack steadily increases the objective value (i.e., the deviation of estimated queue length from the pristine value) at a rate faster than the classical gradient method. The inferior performance of the classical gradient method is due to the random mechanism of selecting target points and the ineffective perturbation direction. By implementing the improved gradient method, the attack performance matches the semi-derivative attack at the early stage of the attack (when no constraint is active). However, when some constraints become active (at the 13th iteration), the semi-derivative method outperforms the improved gradient method. This is because, without considering the effects of constraints, keeping perturbing the data point following the gradient methods (both the classical and the improved ones) is not effective, and even reduces the objective and degrade the attack performance (e.g., around the 17th iteration for the improved gradient method and the 21st iteration for the classical method). These observations demonstrate the advantages of the proposed attack method by applying the Lipschitz continuity concept and the semi-derivative technique.

To help understand the features and advantages of our proposed attack method, Fig. 5 compares the directions of gradients and optimal semi-derivatives together with their impacts (assuming a unit perturbation). The arrows indicate the directions at each point and the arrow lengths are proportional to the impacts. Fig. 5 presents results for the first iteration (Fig. 5(A) and (C)) and after w hits the upper bound (i.e., constraint g_1 is active, Fig. 5(B) and (D)). We can observe that before a constraint becomes active (Fig. 5(A) and 5 (C)), the directions of gradients and semi-derivatives are the same. The directions are intuitive to understand: the slope is to be rotated counterclockwise to deviate the model solution, which is achieved by perturbing points around the queue head "downward" while those around the queue tail "upward". Meanwhile, attacking points (vehicles) at the head or the tail of the queue leads to larger impacts than attacking those in the middle, and clearly, point 6 is targeted for the first iteration of the attack.

However, semi-derivatives change after the constraint becomes active: they are in the same (upward) direction and tend to induce the same impact regardless of the target point (see Eq (16)). In contrast, the gradients are still the same as if the constraint is inactive, since the gradient method ignores constraints. These semi-derivatives seem counterintuitive but can be explained with some examples. First, assuming point 0 is perturbed downwards following the gradient's direction, the fitted line can no longer be rotated counterclockwise due to the constraint but remains the slope unchanged and shifts parallelly towards the left. Such response actually offsets the deviation of model solution.[2] This explains that after selecting the wrong data point and direction, the subsequent attacks by the

[2] Remind that the model solution (queue length estimate) is equivalent to the root of the fitted line (Fig. 3). Shifting the line parallelly towards the left reduces the root, which is initially increasing with the perturbations.

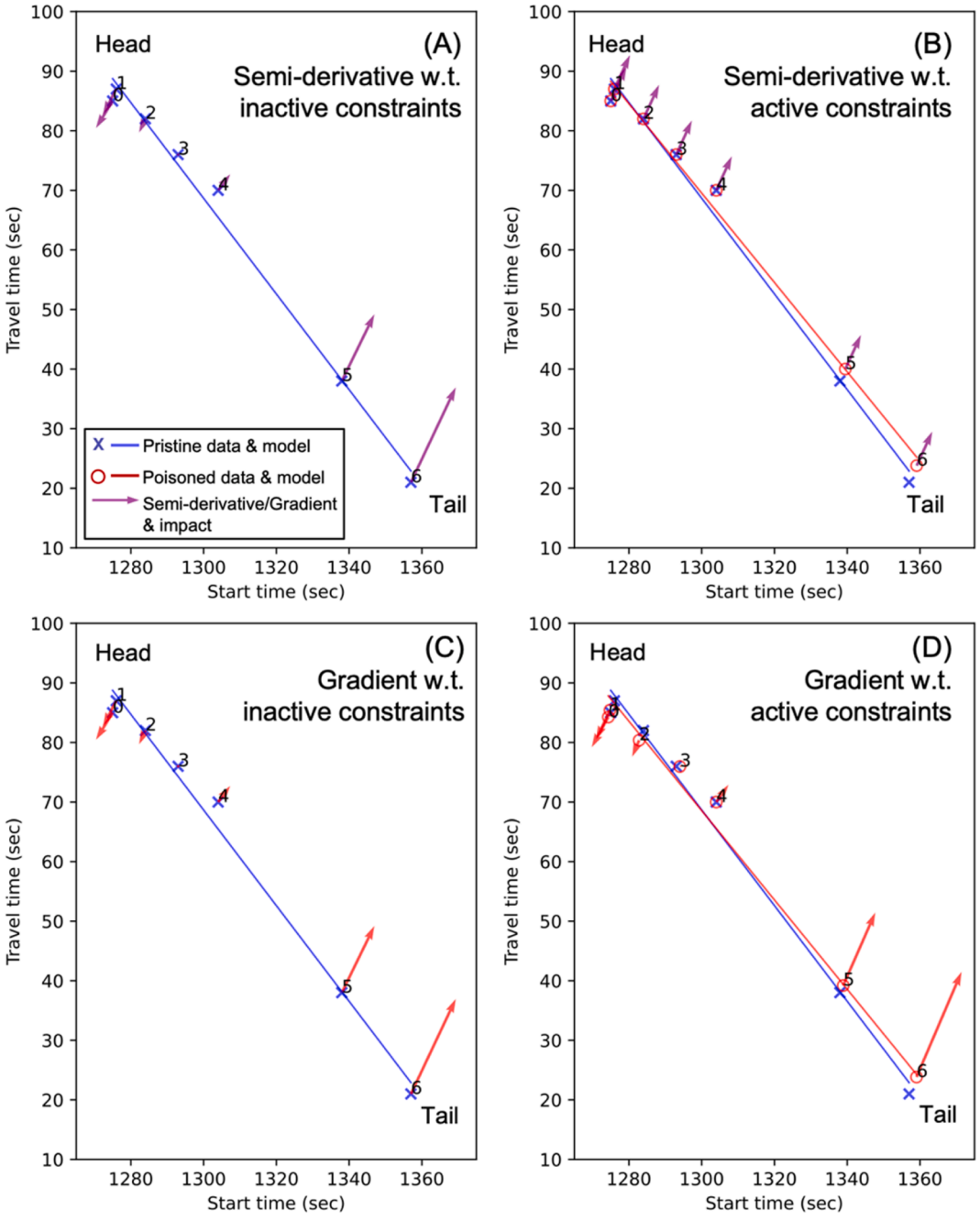

Fig. 5. Compare semi-derivatives with gradients (A) semi-derivatives when the constraint is inactive; (B) semi-derivatives when the constraint is active; (C) gradients when the constraint is inactive; (D) gradients when the constraint is active.

gradient-based method are impacted and become less effective as shown in Fig. 4(B). In contrast, if we perturb point 0 upward following the semi-derivative, the fitted line stays at the same slope and slightly shifts towards the right, leading to further deviation in model solution. Intuitively, deviating model solution by such a parallel shift (towards the right) can be less effective than rotating the line (before constraints become active). This explains the reduction in the convergence rate of our attack after the 13th iteration (Fig. 4B).

Test stealthiness of the attack. In order to validate the stealthiness of the proposed attack, existing defense methods are implemented and tested on the poisoned datasets, including Huber regression (Huber, 1964), RANSAC (Fischler and Bolles, 1981) and TRIM algorithm (Jagielski et al., 2018). Huber regression and RANSAC are two well-known methods in robust statistics; both are robust to outliers in the data. TRIM is a recently developed defense against data poisoning attacks on regression models. It is found that

these existing defense methods could not effectively identify the data poisoned by the proposed attack and correct the queue length estimates, suggesting the stealthiness and effectiveness of the proposed data poisoning attacks.

System-level impacts of the attack. This study focuses on attacking TSEP, since it is not only critical as itself, but also an essential input to many traffic control and management applications. The compromised TSEP, when used for making decisions, would result in *system-level impacts*. Fig. 6 shows the impact of the attack on the total delay over time at a simulated intersection, which is controlled by adaptive, max-pressure-based signal controllers using the estimated real-time queue lengths (Mercader et al., 2020). It can be observed that the attack deviating the queue length estimation (achieved by manipulating traffic of WE approach) result in longer delays under medium traffic demand. On average, the delay increases by 8 %. Our further tests suggest that the impact of attacks on travel delay varies with traffic situations: no change is observed under low demand since queue is not formed, while more delay would be induced (15 %) by poisoning attacks when the queue is long under the high demand.

4.2. SVM-based vehicle classification

4.2.1. Lipschitz analysis of support vector machine

An SVM model can be expressed as follows:

$$\begin{aligned} \min_{y=(\boldsymbol{w},b,\boldsymbol{\xi})} & \frac{1}{2}\|\boldsymbol{w}\|_2^2 + C\sum_i^n \xi_i \\ & g_i^{\mu} = -\xi_i \le 0, \\ \text{s.t.} \quad & g_i^{\mu} = -\xi_i \le 0, \\ & i = 1, \cdots, n. \end{aligned} \tag{17}$$

Here, $x_i = [x_{i,1}, \ldots, x_{i,k}]$ contains k feature values for one of the n data points and l_i gives the label (1 or -1 as one of two classes of vehicles). C is a user-specified hyper-parameter.

Lipschitz analysis. The VI that captures the first-order conditions of Eq (17) is written as,

$$\begin{pmatrix} \nabla_y L(x,y,\alpha,\mu) \\ -\nabla_\alpha L(x,y,\alpha,\mu) \\ -\nabla_\mu L(x,y,\alpha,\mu) \end{pmatrix} + N_E(y,\alpha,\mu) \ni \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}.$$

Here, $L(x,y,\boldsymbol{\alpha},\boldsymbol{\mu}) = \frac{1}{2}\|\boldsymbol{w}\|_2^2 + C\sum_i^n \xi_i + \sum_i^n \left(\alpha_i[1-\xi_i - l_i(\boldsymbol{x}_i^\top \boldsymbol{w} + b)]\right) - \sum_i^n \mu_i \xi_i$ is the Lagrangian function with multipliers α_i, μ_i ($i \in [1, \cdots, n]$), and $E = \mathbb{R}^{k+1+n} \times \left(\mathbb{R}_+^n \times \mathbb{R}_+^n\right)$.

For the convenience of constructing the auxiliary problem, we introduce additional notations in Table 1, including R_0, R_1, V, E_0 and E_1, to group the data points based on the status of constraints (g_i^{α} and g_i^{μ}) and multipliers (α_i and μ_i). Notice that being a multiplier, α_i (μ_i) is always perpendicular with g_i^{α} (g_i^{μ}).

Omitting some details, we can write the auxiliary problem following Eq (5).

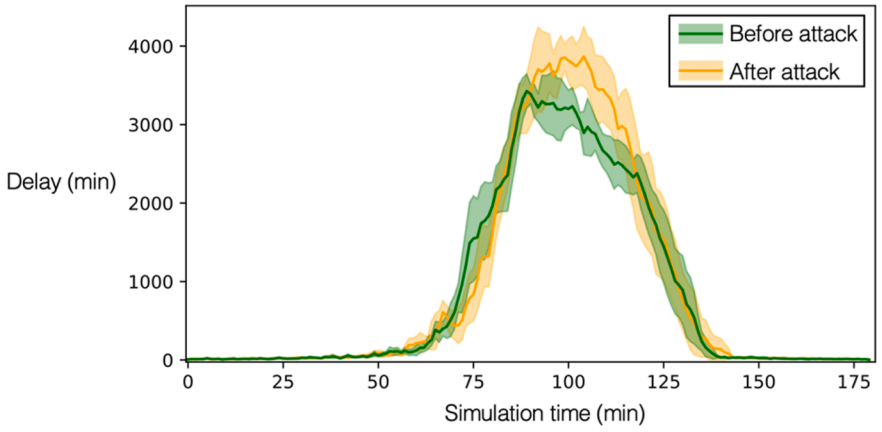

Fig. 6. Total delays before and after attacking queue length estimation (medium demand).

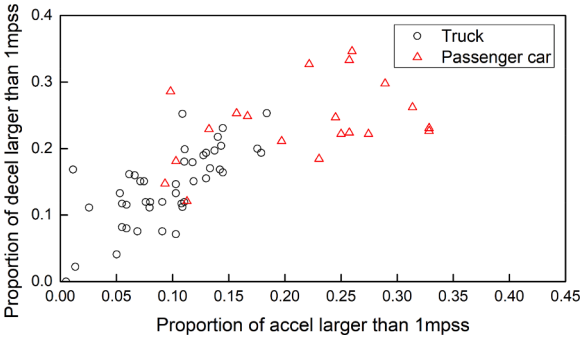

Fig. 7. Data for vehicle classification.

Table 1
Grouping data in the SVM model.

Data groups	$\boldsymbol{\alpha_i}$	$\boldsymbol{\mu_i}$	$\boldsymbol{g_i^{\alpha}}$	$\boldsymbol{g_i^{\mu}}$
$\boldsymbol{R_0}$	$\alpha_i = 0$	$\mu_i = C$	$g_i^{\alpha} = 0$	$g_i^{\mu} = 0$
$\boldsymbol{R_1}$	$\alpha_i = 0$	$\mu_i = C$	$g_i^{\alpha} < 0$	$g_i^{\mu} = 0$
$\boldsymbol{V}$	$0 < \alpha_i < C$	$0 < \mu_i < C$	$g_i^{\alpha} = 0$	$g_i^{\mu} = 0$
$\boldsymbol{E_0}$	$\alpha_i = C$	$\mu_i = 0$	$g_i^{\alpha} = 0$	$g_i^{\mu} = 0$
$\boldsymbol{E_1}$	$\alpha_i = C$	$\mu_i = 0$	$g_i^{\alpha} = 0$	$g_i^{\mu} < 0$

$$\min_{\Delta y=\left(\Delta w,\Delta b,\Delta\xi_{i(i\in N)}\right)} \bar{g}_0(\Delta y) - \langle v, \Delta y\rangle \text{ overall } \Delta y \text{ satisfying}$$
$$\bar{g}_i^{\alpha}(\Delta y) + u_i^{\alpha} \begin{cases} = 0 & \text{if } i \in V \cup E_0 \cup E_1 \\ \le 0 & \text{if } i \in R_0 \\ \text{free} & \text{if } i \in R_1 \end{cases}$$
$$\bar{g}_i^{\mu}(\Delta y) + u_i^{\mu} \begin{cases} = 0 & \text{if } i \in V \cup R_0 \cup R_1 \\ \le 0 & \text{if } i \in E_0 \\ \text{free} & \text{if } i \in E_1 \end{cases} \tag{18}$$

Here $\bar{g}_0(\Delta y) = L(\bar{x}, \bar{y}, \bar{\alpha}, \bar{\mu}) + \frac{1}{2}\Delta w^{\top}\Delta w$, $\bar{g}_i^{\alpha}(\Delta y) = g_i^{\alpha}(\bar{x}_i, \bar{y}) + (-l_i x_i^{\top}\Delta w - l_i\Delta b - \Delta\xi_i)$ and $\bar{g}_i^{\mu}(\Delta y) = g_i^{\mu}(\bar{x}_i, \bar{y}) + (-\Delta\xi_i)$. $N = R_0 \cup R_1 \cup V \cup E_0 \cup E_1$ represents the full dataset. Following Equations (6)-(7), the corresponding first-order conditions for the auxiliary problem are:

$$\begin{bmatrix} \Delta w_j - \sum_i^n z_i^{\alpha} l_i x_{i,j} (j = [1, \ldots, k]) \\ -\sum_i^n z_i^{\alpha} l_i \\ -z_i^{\alpha} - z_i^{\mu} (i \in N) \end{bmatrix} - v = 0, \ with$$
$$z_i^{\alpha} \begin{cases} \ge 0 & \text{if } i \in R_0 \text{ having } -l_i\left(x_i^{\top}\Delta w + \Delta b\right) - \Delta\xi_i + u_i^{\alpha} = 0, \\ = 0 & \text{if } i \in R_0 \text{ having } -l_i\left(x_i^{\top}\Delta w + \Delta b\right) - \Delta\xi_i + u_i^{\alpha} < 0 \,\&\, i \in R \\ \text{free} & \text{if } i \in V \cup E_0 \cup E_1 \end{cases} \tag{19}$$
$$z_i^{\mu} \begin{cases} \ge 0 & \text{if } i \in E_0 \text{ having } -\Delta\xi_i + u_i^{\mu} = 0, \\ = 0 & \text{if } i \in E_0 \text{ having } -\Delta\xi_i + u_i^{\mu} < 0 \,\&\, i \in E_1 \\ \text{free} & \text{if } i \in V \cup R_0 \cup R_1 \end{cases},$$
$$[i = 1, \ldots, n.]$$

Here $z_{i\in N}^{\alpha}$ and $z_{i\in N}^{\mu}$ are the multipliers w.r.t constraints $\bar{g}_{i\in N}^{\alpha}$ and $\bar{g}_{i\in N}^{\mu}$, respectively. We then focus on the auxiliary solution mapping $\bar{S}(v, u) = \{(\Delta y, z) |$ satisfying the requirement in Eq (19)$\}$.

In Appendix A.5, we examine the two sufficient conditions for Lipschitz continuity and show that condition (*ii*) is always satisfied while condition (*i*) depends on the specific dataset. Assuming data point a is perturbed, $D\widehat{y}(x, \widehat{y}(x))(\Delta x_a)$ is computed by inputting

$-B\Delta x_a$ to mapping $\bar{s}$ with

$$B = \begin{pmatrix} \nabla^2_{yx_a} L(\bar{x}_a, \bar{y}, \bar{\alpha}, \bar{\mu}) & -\nabla_{x_a} g_i^{\alpha}(\bar{x}_a, \bar{y}) \text{ for } i \in N & -\nabla_{x_a} g_i^{\mu}(\bar{x}_a, \bar{y}) \text{ for } i \in N \end{pmatrix}. \quad (20)$$

Here, $\nabla^2_{yx_a} L(\bar{x}_a, \bar{y}, \bar{\alpha}, \bar{\mu}) = [-\bar{\alpha}_a l_a \mathbb{I}_k, 0_{(n+1)\times k}]^T$, $\nabla_{x_a} g_i^{\alpha}(\bar{x}_a, \bar{y}) = \left\{ -l_a \bar{w} i = a 0 i \neq a \right\}$, $\nabla_{x_a} g_i^{\mu}(\bar{x}_a, \bar{y}) = 0 (i \in N)$.

4.2.2. Svm-based vehicle classification model

We then apply the proposed attack method to an SVM-based model that classifies vehicles into trucks and passenger cars using GPS data (Sun et al., 2013). Without introducing the details in the original study, the model can be briefly summarized in the following steps:

a) Collect GPS data of passenger cars and trucks.
b) Generate speed and acceleration-/deceleration-related features for each vehicle using GPS data.
c) Train an SVM classifier for vehicle classification and apply the trained classifier to the test dataset.

In this application, feature values extracted from the trajectory of vehicle i give the data x in Eq (17), and vehicle labels give l_i. Note that the step for feature selection in the original study is omitted here; this study takes the top two effective features (i.e., the variations of acceleration x_i^{ac} and deceleration x_i^{dc}) of the multiple features tested in the original study to facilitate readers' understanding of our analysis and visualizations (Fig. 7). The proposed method can be naturally applied to SVM with any number of features. Fig. 8B shows the SVM boundary learned from the pristine data, which gives $\boldsymbol{w} = [w_1, w_2] = [-11.2, -2.5]$ and $b = 6.7$, meaning that the learned SVM *weighs heavily on one feature* (i.e., the acceleration-related feature).

4.2.3. Attack model formulation and solution

The adversarial goal is to mislead the SVM model into believing that the two features are equally important for vehicle classification. As such, we formulate a model-target poisoning attack, where the objective is to induce the model parameter $\boldsymbol{w}$ towards the adversarial goal $\boldsymbol{w}^* = (w_1^*, w_2^*)$ such that the two feature weights are equal. Specifically, the attack model can be written below.

$$\left.\begin{aligned} &\min_x \ G = \|\widehat{w}_1 - \widehat{w}_2\|_2^2 \\ &\text{s.t.} \quad |x - \bar{x}| \le \delta \\ &\qquad \rho_1 \le x_i^{ac} \le \rho_2 \\ &\qquad \rho_3 \le x_i^{dc} \le \rho_4 \end{aligned}\right\} \text{Upper level}$$

$$\left.\begin{aligned} &\widehat{y} = (\widehat{w}_1, \widehat{w}_2) \in \text{solution to Eq (17)} \\ &\qquad i = 1, \ldots, n \end{aligned}\right\} \textit{Lower level} \quad (21)$$

Here, the SVM model (Eq (17))) contains inherent constraints. These constraints complicate the mapping from the data perturbation and TSEP solution, as the constraints themselves are subject to changes due to the data perturbation. The constraints on x_i^{ac} and x_i^{dc} in

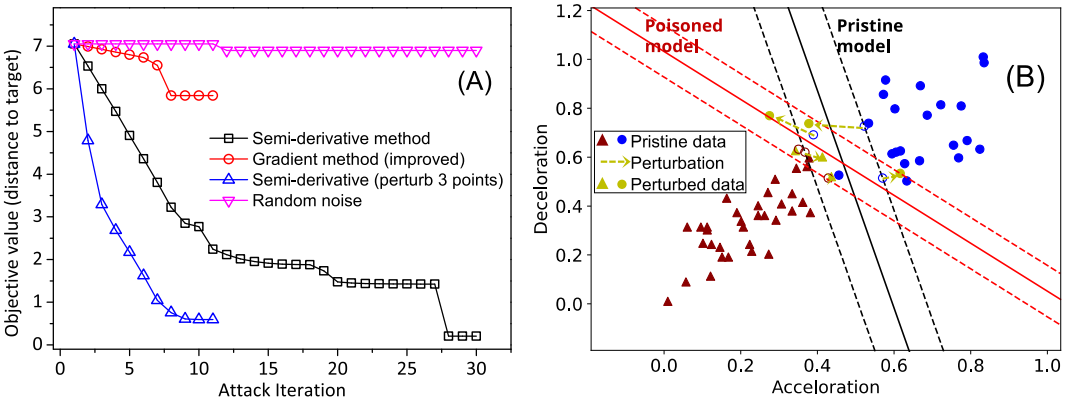

Fig. 8. (A) Convergence of attack models. (B) Poisoned model and data following our attack (pristine and poisoned models are in black and red lines, respectively). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

the upper-level problem suggest that the feature values of vehicles are bounded by the limits of vehicle dynamics (e.g., acceleration and deceleration limits). Details of deriving these constraints can be found in Appendix A.6.

For our SVM model on the vehicle classification dataset, the SVM model solution $\hat{y}$ is indeed Lipschitz continuous and semi-differentiable around $\bar{x}$. See details in Appendix A.7. Then, following Theorem 1, it is straightforward to compute semi-derivatives and implement our attack algorithm. Assuming vehicle a is under attack, we have

$$DG(x, \hat{y}(x))(\Delta x_a) = \begin{bmatrix} -1 \\ 1 \end{bmatrix}^T \Delta w \tag{22}$$

Following the analysis earlier, Δw is computed following Equations (18)-(19). See Appendix A.8.

4.2.4. *Effectiveness of attack*

Fig. 8 shows the convergence of the adversarial objectives following different attack methods. We can observe that our attack steadily reduces the difference (note that the attack objective now is to minimize the difference between the target and the learned model) and consequently converges to the target model. After the convergence, six data points are slightly perturbed and the SVM's decision boundary is significantly shifted (Fig. 8(B)), making the two features equally important to separate cars and trucks. We also test perturbing simultaneously multiple (three) points at each iteration. Fig. 8(A) suggests that the convergence can significantly speed up.

In contrast, adding random noises to data barely affects the SVM model. The gradient attack does not seem to converge toward the target model reliably (Fig. 8(A)) either. In fact, the gradient attack not only converges at a much slower rate than the semi-derivative method but also stops early after only a few iterations. To attack an SVM model that involves equality and inequality constraints, the gradient-based method ignores inequality constraints in the KKT conditions and evaluates gradients using equality constraints only (Biggio et al., 2013, 2011; Xiao et al., 2015b). By ignoring the inequality constraints, the yielded gradients are not accurate but only approximations. With further explorations below, we show that such approximation leads to some drawbacks: the convergence of attacks (to a target model) can be slow, and the attack iteration may stop early.

Fig. 9 compares the directions of gradients and semi-derivatives together with their impacts under a unit perturbation. It is expected to observe from Fig. 9(A) that the gradients are only nonzero at support vectors (including margin and error support vectors; see below). Additionally, these gradients are in parallel with each other, while those associated with the two classes are in opposite directions. To understand this, we simplify the computation of the gradient (Biggio et al., 2013, 2011; Xiao et al., 2015b) and find that the gradient at data x_a is determined by

$$\nabla_{x_a} g = \bar{\alpha}_a \bullet l_a \bullet \Lambda_V \tag{23}$$

Here Λ_V is a vector computed using data group V in Table 1 only. Note that conventionally, the solution structure of an SVM model is characterized by three groups of points based on the dual variable $\bar{\alpha}_a$ (instead of five groups in Table 1), including V ($0 < \bar{\alpha}_a < C$), $E = E_0 \cup E_1$ ($\bar{\alpha}_a = C$) and $R = R_0 \cup R_1$ ($\bar{\alpha}_a = 0$) (Fig. 9). Eq (23) explains the previous observations that $\nabla_{x_a} g = 0$ for $a \in R$ and the opposite yet parallel directions of the gradients. Additionally, it suggests (also observable from Fig. 9(A)) that the impacts of perturbing points in V are smaller than perturbing points in E. Such a difference in the sensitivity of data points to perturbations is part of the reason why the convergence rate of the classical gradient method (attacking SVM by perturbing randomly selected points) is slower than the semi-derivative. Moreover, Eq (23) suggests that computing the gradients relies on a non-empty set of V. If V is empty in certain iteration, the gradient method will fail to proceed as the calculated gradient is zero. Our investigation shows that this is exactly the cause of the

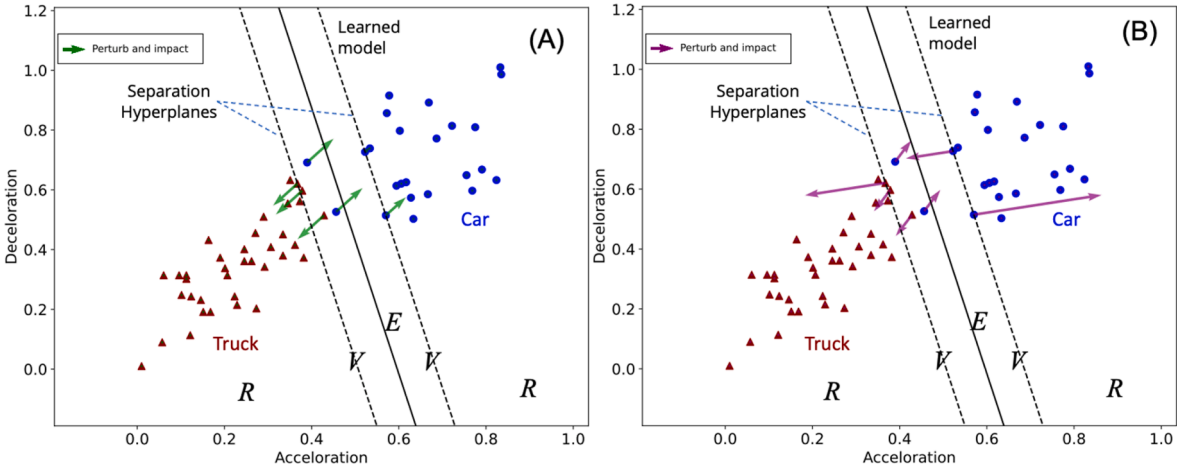

Fig. 9. (A) Gradient and its impact at each data point; (B) semi-derivative and its impact. (Points are in three groups: margin support vectors V, error support vectors E, and reserve points R.)

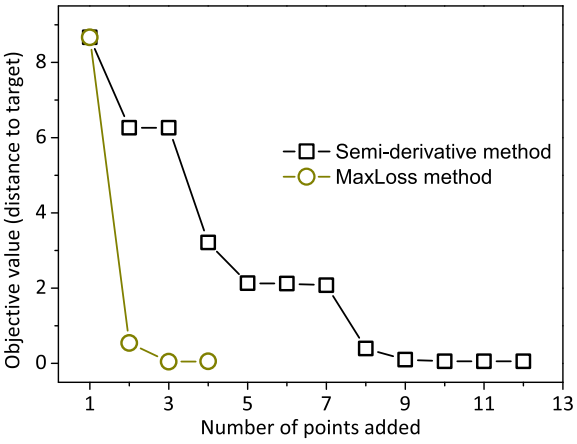

Fig. 10. Convergence to the target SVM model by adding new poisons: comparing our attack with MaxLoss method.

early stop of the gradient method in Fig. 9A.

The semi-derivative and its impact at each data point are shown Fig. 9(B). It shows some similarities but many significant differences with the gradients in Fig. 9(A). In terms of the similarities, there is no impact by perturbing any point in R following the semi-derivative method. And the semi-derivatives in E are in parallel while those at the two classes are in opposite directions. In terms of differences, the semi-derivatives in V are distinct from the gradients. Moreover, the semi-derivatives in V can be different from each other and seem not relevant to those in E. And the impact of perturbing a point in V can be larger than perturbing any point in E, in contrast to the observation from the gradient method above.

Compare with other model-targeted attack methods.

We also compare our attack with the state-of-the-art model-targeted attack (Suya et al., 2021). The latter has been shown effective in inducing the SVM model towards a pre-specified target by sequentially *adding poisoning data*. At each iteration, it adds the poisoning point that has the maximum loss difference between the intermediate model obtained so far and the target model (refer to as the *MaxLoss* method for clarity). It was shown that following the iterations, the attack gradually minimizes the maximum loss difference between the induced intermediate model and the target model, eventually obtaining a model similar enough to the target model. To be comparable with the MaxLoss method, the semi-derivative method also attacks SVM by adding new poisons via modifying Algorithm 1 (see Appendix A.10). Specifically, a new point is initialized by duplicating a (randomly selected) pristine point near the margin lines. Then we perturb it following the semi-derivative until certain conditions/constraints are met before adding another new poison. The algorithm stops when the convergence condition is satisfied.

Fig. 10 shows that the semi-derivative method via adding new poisons can effectively induce the model to the target after adding nine new poisons. The MaxLoss method can achieve the same objective by adding four points. Note that there are some overlapping points in the figure. Comparing the convergence rate of the two methods, it seems that the MaxLoss method is more effective. However, the semi-derivative method can be stealthier than MaxLoss (Fig. 11). It can be observed that the new poisons added by the semi-derivative attack are close to the pristine data and thus harder to be detected as attacks (Fig. 11A). However, the new poisons generated by the MaxLoss method are very different from the pristine data, which could be easily detected by common attack detection methods (see details below).

Test stealthiness of the attack. In order to validate the stealthiness of the proposed attack, common attack detectors are

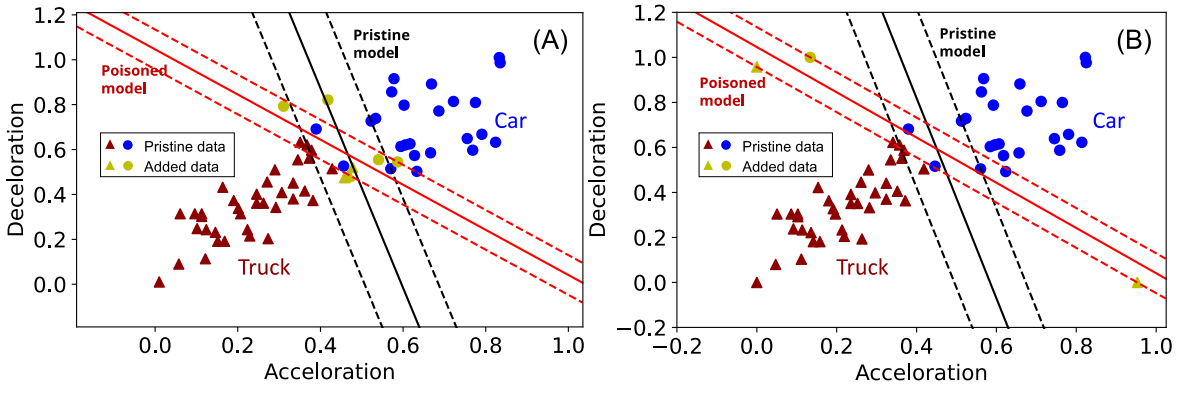

Fig. 11. Results of adding new poisons. (A) poisons added using our attack; (B) poisons added using MaxLoss method.

implemented to check whether the attack could be detected. Specifically, anomaly detectors (e.g., OCSVM and Isolation Forest) are trained as attack detectors using clean datasets. Then the detectors are implemented to detect data poisons. It is found that the data poisons generated by the proposed attack cannot be detected due to their stealthiness. However, the data poisons added by the MaxLoss method can all be identified by the trained detectors, implying that MaxLoss attacks can be relatively easily defended.

System-level impacts of the attack. The attack here shifts the SVM decision boundary for vehicle classification by inducing it to that predetermined by an attacker. Vehicle classification is crucial for traffic state estimation because it provides valuable insights into the composition of traffic on roadways. Different vehicle types have distinct behaviors, speeds, and space occupancy, influencing traffic dynamics. By classifying vehicles, one can better predict congestion, optimize traffic flow, and enhance overall traffic management and control. Therefore, it is expected that attacks against vehicle classification model would lead to system-level impacts.

5. Concluding remarks and discussions

This study investigates the vulnerability of TSEP models under data poisoning attacks, which is an emerging yet urgent issue as transportation systems are increasingly data-reliant. Data poisoning attacks against TSEP models are formulated as a general sensitivity analysis of optimization problems over data perturbations (attacks). Considering that TSEP models are generally constrained and lack differentiability in terms of how TSEP solutions respond to data changes, we study the Lipschitz continuity property of TSEP solutions. We then apply the generalized implicit function theorem for the problem and compute the semi-derivative of TSEP solution over data changes. This enables us to develop attack models to fit a broader spectrum of TSEP applications, especially those with general constraints.

We demonstrate the generality of the Lipschitz analysis and attack methods by tailoring them to the LS regression model and SVM, both of which have been widely used for TSEP applications. Real-world applications are then tested, including queue-length estimation and vehicle classification problems. Specifically, we demonstrate that the proposed attack can be flexible in terms of fitting different types of adversarial objectives. Our results show that the proposed attack model can effectively achieve the adversarial objectives, outperforming the state-of-the-art gradient method. We demonstrate that without considering the impact of constraints, the gradient method can be slow in moving toward the adversarial objectives or even fail to advance when the constraints become active. The study has multiple implications, including promoting the understanding of the vulnerability of data-driven transportation systems and helping to develop effective defense methods.

For future research, several directions can be explored. First is to study attack settings where an attacker does not know the TSEP model (i.e., a grey or black box attack). In these settings, attackers may possess limited or no access to the victim model's internal structure, parameters, or training data. One common approach involves probing the model's behavior through queries and subsequently constructing a *surrogate* model to approximate the original one. This surrogate model can then be targeted using attack methods designed for white-box scenarios (like the one proposed in this paper). For instance, a recent study explored a black-box attack on connected vehicle-based traffic signal control systems. The study initially learned the control logic via queries and subsequently launched attacks based on this acquired knowledge (Feng et al., 2022; Huang et al., 2021). Another promising approach is to explore the *transferability* of attack models that are developed on white-box models. Studies in the cybersecurity domain have reported that attacks developed on public deep-learning models and datasets could still be effective to fail grey or black box models (Xie et al., 2022). Further investigation is necessary to evaluate the effectiveness of these approaches; see more discussions below on data poisoning attacks for DL models.

Second, additional efforts are needed to extend the attack model to statistical learning and deep learning-based TSEP models. The training (and sometimes inference as well) of many *statistical learning models* relies on optimization techniques. For example, Bayesian networks (BN) has been used for (cycle by cycle) vehicle index and queue length (distribution) estimation (Hao et al., 2014), human mobility modeling (Toch et al., 2019), etc. The objective functions of the estimation models in statistical learning can be complex, which may bring difficulty to model attacks and solutions. One promising direction is to investigate methods to approximate complex functions that are from the objective functions or constraints of the attack models. For *deep learning (DL) models*, the target model can be even more complex and often unknown, as the attacker does not necessarily know the key components of the model such as the training data, model structure or parameters. In such cases, a promising idea is to develop attack models based on the transferability of DL models rather than directly working on the target model (Xie et al., 2022). Transferability ensures that the derived attack model from public data and DL structures can be transferred and applied to effectively attack other DL-based TSEP models. For either the statistical learning or DL-based models, multiple solutions may exist. This implies that $P(x)$ in Eq (1) is now a *set-valued map* and the second Lipschitz continuity definition in Section 3.3 should apply. The subsequent analysis and the attack models should also be constructed accordingly.

The third direction is to develop effective defense methods against poisoning attacks. Often the reason we are interested in developing attack models is to better understand the attacks and vulnerability, and then develop effective defense methods against. Detailed discussions of defense methods are beyond the scope of this paper. Here we provide some highlights of the key ideas of the defense methods. Two approaches may be developed to defend against the data poisoning attacks proposed in this study. The first approach is exercising the "security by design" principle to devise data poisoning defenses following the Lipschitz analysis. For a TSEP model, if the model solution turns out to be not Lipschitz continuous with data perturbations, we will explore ways (to reformulate or approximate the objective/constraints or even the structure of the model) to revise/refine the model so that the Lipschitz property may hold. Doing so will ensure that the solution change is bounded (instead of explosive) when input data changes. Appendix A.11 presents an example to show that an LS model with very large Lipschitz constant (practically a non-Lipschitz continuous model) can be extremely sensitive to data perturbations. Such models need to be strengthened first to reduce their Lipschitz constants, for which the

methods and analysis in this paper can provide useful tools and insights. In fact, in the DL literature, research has been conducted to estimate the Lipschitz constants of DL models (Scaman and Virmaux, 2019) to assess the robustness of such models. One can also use the estimated Lipschitz constants to assess the resilience of the model w.r.t. data poisoning attacks. However, it is still an open question to secure an DL model if the model turns out to be not robust (or not resilient).

The second approach is to develop detection and mitigation methods for data attacks. Designed to be stealthy, data poisoning attacks can be challenging to detect and mitigate. Previous studies have shown that conventional detection methods may fail in the context of stealthy GPS spoofing (F. Wang et al., 2023a). One reason is that existing detection methods are often based on anomaly-detection models learned from historical or training data, which could be insecure or contaminated by poisoning data samples. Recent defense methods have been developed based on adversarial training or certified training, i.e., training DL models with both (tagged) pristine data and poisoned data to improve model resilience to attacks (Akhtar and Mian, 2018). However, such models suffer from reduced accuracy (with improved robustness). One alternative way to address the challenge is to leverage secure data from the infrastructure, which allows us to develop "infrastructure-enabled" defense solutions against poisoning attacks. Indeed, such an idea has been successfully demonstrated in our previous study for defending against stealthy attacks on GPS data (F. Wang et al., 2023a), and the attacks on the LS model for queue length estimation (F. Wang et al., 2023b). It is important and interesting to explore "infrastructure-enabled" defense methods against attacks on other TSEP models, as transportation infrastructure is becoming smarter and more equipped (with sensors) to support various advanced transportation applications.

CRediT authorship contribution statement

Feilong Wang: Data curation, Investigation, Methodology, Software, Validation, Writing – original draft, Writing – review & editing. **Xin Wang:** Methodology, Validation, Writing – review & editing. **Hong Yuan:** Investigation, Validation, Writing – review & editing. **R. Tyrrell Rockafellar:** Investigation, Methodology, Validation. **Jeff Ban:** Data curation, Investigation, Methodology, Validation, Writing – review & editing.

Declaration of competing interest

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

Acknowledgement

We thank the two anonymous reviewers for their constructive comments that helped significantly improve the original version of the paper. The research is supported by the National Science Foundation (NSF) grant CMMI-2326340. Any opinions, findings, conclusions, recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of NSF.

Appendix

1.1. Checking Lipschitz continuity of LS regression

In this study, we assume that independent variables of the data are linearly independent, meaning that there exists no feature in the data that is a linear combination of other features (i.e., $\boldsymbol{x}^{ind_j} = \sum_{j \neq j} \boldsymbol{x}^{ind_j}$). This assumption assures that the covariance matrix below

$$\Sigma = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_i^{ind} \boldsymbol{x}_i^{ind\top} = \frac{1}{n}\sum_{i=1}^{n} \left(\begin{bmatrix} x_i^{ind_1} \\ \vdots \\ x_i^{ind_k} \end{bmatrix} \times \begin{bmatrix} x_i^{ind_1} & \cdots & x_i^{ind_k} \end{bmatrix} \right) \tag{24}$$

is positive definite. More specifically, being a covariance matrix, Σ is surely positive semi-definite; it shall be positive definite as long as independent variables of the data are linearly independent.

We can verify that the data used in the queue length estimation model satisfies this assumption.

Following Theorem 1, we check the following two sufficient conditions to ensure that *S has a Lipschitz continuous single-valued s*.

The gradients $\nabla_y g_h(x,y)$ for all active constraints are linearly independent.

Since $\nabla_y g_h(\boldsymbol{x},\boldsymbol{y}) = \boldsymbol{\beta}_h$ in our case, this is to check whether vectors in $\{\boldsymbol{\beta}_h \,|\boldsymbol{\beta}_h^\top \bar{y} - \rho_h = 0, h \in [1,\ldots,m]\}$ are linearly independent. This condition thus depends on the specific problem formulation and the solution $\bar{y}$.

$\frac{1}{2}\langle \Delta y, \nabla_{yy}^2 L(\bar{x},\bar{y},\bar{\lambda})\Delta y\rangle > 0$ii. *for every nonzero* $\Delta y \in M^+$.

We show in the following that

$$\nabla^2_{yy} L(\bar{x}_a, \bar{y}, \bar{\lambda}) = \begin{bmatrix} \Sigma & 0 \\ 0 & 1 \end{bmatrix}, \tag{25}$$

is positive definitive, and this condition is naturally satisfied.

Following our assumption, Σ is positive definitive. According to the property of Schur complement, a matrix in the form of $X = \begin{bmatrix} A & B \\ B^T & C \end{bmatrix}$ possesses a nice property (Zhang, 2006):

If C is invertible, then X is positive definite if and only if C and its Schur complement $X/C = A - BC^{-1}B^T$ are both positive-definite.

Apply this property to our case, we have $C = 1$ that is invertible and positive definite and $X/C = \Sigma - 0 \times 1 \times 0^T = \Sigma$. Thus, $\nabla^2_{yy} L(\bar{x}, \bar{y}, \bar{\lambda})$ is positive definite because Σ is so.

1.2. Problem-specific constraints in the queue length estimation model

Vehicles follow certain behaviors (e.g., car following) on the road, which could help to identify abnormal vehicles. To construct realistic and thus stealthy poisoning samples, we add constraints to express the car-following behavior at the intersection: assuming multiple vehicles travel through the intersection in a sequence, a vehicle who departs from VTL_1 first likely arrives at VTL_2 first, as VTL_1 and VTL_2 are spatially close around the intersection. This is particularly true if the road segments have one lane at both the up- and down-stream of the intersection. If not, we can relax the constraint with a slack variable. Formally, if we have the targeted vehicle $veh_a (a \in (1, \cdots, A))$ travels following vehicle veh_l and leading vehicle veh_f (Figure A1), denote the departure time for the three vehicles being x_l^{ind}, x_a^{ind} and x_f^{ind}, respectively. Similarly, denote the travel time from VTL_1 to VTL_2 for the three vehicles being x_l^{dep}, x_a^{dep} and x_f^{dep}, respectively, we then have arrival time at VTL_2 as $x_l^{ind} + x_l^{dep}$, $x_a^{ind} + x_a^{dep}$ and $x_f^{ind} + x_f^{dep}$. The constraints are:

$$\begin{aligned} & x_a^{ind} - x_l^{ind} \geq hw \\ & x_f^{ind} - x_a^{ind} \geq hw \\ & \left(x_a^{ind} + x_a^{dep}\right) - \left(x_l^{ind} + x_l^{dep}\right) \geq hw \\ & \left(x_f^{ind} + x_f^{dep}\right) - \left(x_a^{ind} + x_a^{dep}\right) \geq hw \end{aligned} \tag{26}$$

Here, *hw* is the minimum headway.

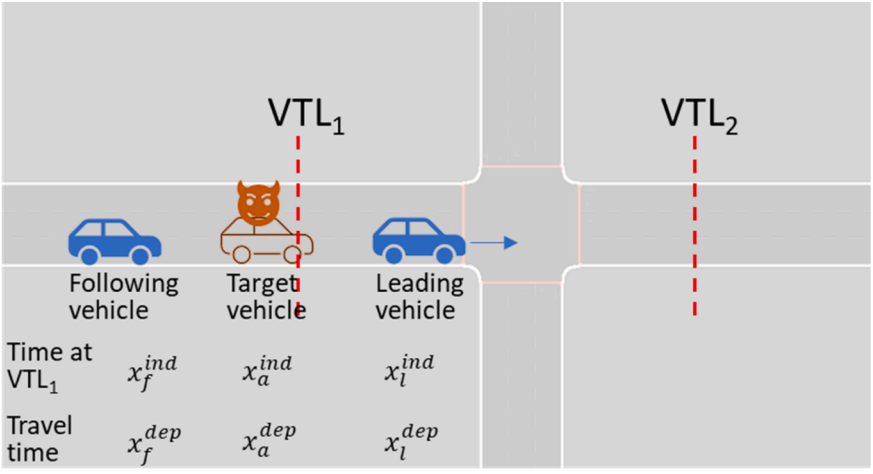

Fig. A1. Illustration of car-following constraint at an intersection.

We can also apply knowledge of traffic flow dynamics as constraints. These constraints assure the dynamic of traffic flow remains respected when crafting poisoning samples. The discharge rate of the queue (i.e., the slope of the delay pattern *w*) is closely related to the dynamics of traffic shock waves, which could be known prior. Therefore, we assume we have a range of *w* as prior knowledge, written as follows.

$$\rho_1 \le w \le \rho_2 \tag{27}$$

In physical meaning, the attack would be easily detected if one finds that the estimated model suggests the queue discharges either too fast or too slow.

1.3. Checking Lipschitz continuity of the queue length estimation model

In the following, we show that the solution mapping S defined for the length estimation model in Eq (13) is indeed Lipschitz continuous. We have proven above that for LS regression model, condition (*ii*) in Theorem 1 will always hold. Condition (*i*) is problem-specific. Regarding the queue length estimation model, the first sufficient condition requires that the gradients $\nabla_y g_i(\bar{x}, \bar{y})$ for all *active* constraints in the lower-level problem Eq (13) are linearly independent. Specifically, assuming all constraints are active, the gradients consist of

$$\nabla_y g_1(\bar{x}, \bar{y}) = [-1,\ 0], \qquad \nabla_y g_2(\bar{x}, \bar{y}) = [1,\ \ 0]. \tag{28}$$

The two constraints cannot be active at the same time, as they represent the upper and lower bounds. Therefore, for the queue length estimation model, the mapping S is Lipschitz continuous and semi-differentiable around $\bar{x}$.

1.4. Solving the semi-derivative of the queue length estimation model

In this section, we give details of computing the semi-derivative for attacking against the queue length estimation model.

First, following Eq (12), we can evaluate $\Delta y = \begin{bmatrix} \Delta w \\ \Delta b \end{bmatrix}$. Giving the input $-B\Delta x_a$ (i.e., (v,u)) to Eq (12) and with some rearrangement, we can obtain Δy by solving the following.

$$\Delta y = -\begin{bmatrix} \Sigma & 0 \\ 0 & 1 \end{bmatrix}^{-1} \left(\begin{bmatrix} \kappa_1 \\ \kappa_2 \end{bmatrix} \Delta x_a + \sum_{h=1}^{m} z_h \beta_h \right),$$
$$\text{with } z_i \begin{cases} \ge 0 & \text{if } \beta_h^T \bar{y} - \rho_h = 0 \text{ with } \bar{\lambda}_h = 0 \,\&\, \beta_h^\top \Delta y = 0, \\ = 0 & \text{if } \beta_h^T \bar{y} - \rho_h = 0 \text{ with } \bar{\lambda}_h = 0 \,\&\, \beta_h^\top \Delta y < 0, \text{ or } \beta_h^{\mathrm{T}} \bar{y} - \rho_{\mathrm{h}} < 0 \\ \text{free} & \text{if } \beta_h^T \bar{y} - \rho_h = 0 \text{ with } \bar{\lambda}_h > 0 \text{ then from } KKT,\ z_h \beta_h^\top \Delta y = 0 \end{cases} \tag{29}$$

For our specific problem of estimating queue length, we then have

$$\Delta y = \begin{bmatrix} \Delta w \\ \Delta b \end{bmatrix} = -\frac{1}{\Sigma} \left(\frac{1}{n} \begin{bmatrix} \kappa_1 \\ \kappa_2 \end{bmatrix} \Delta x_a + z_1 \begin{bmatrix} -1 \\ 0 \end{bmatrix} + z_2 \begin{bmatrix} 1 \\ 0 \end{bmatrix} \right), \text{with}$$
$$z_1 \begin{cases} \ge 0 & \text{if } g_1 \text{ is active with } \bar{\lambda}_1 = 0 \text{ having } [-1, 0]^\top \Delta y = 0 \\ = 0 & \text{if } g_1 \text{ is active with } \bar{\lambda}_1 = 0 \text{ having } [-1, 0]^\top \Delta y < 0 \text{ or } g_1 \text{ is inactive} \\ \text{free} & \text{if } g_1 \text{ is active with } \bar{\lambda}_1 > 0, \text{ then } z_1 [-1, 0]^\top \Delta y = 0 \end{cases} \tag{30}$$
$$z_2 \begin{cases} \ge 0 & \text{if } g_2 \text{ is active with } \bar{\lambda}_2 = 0 \text{ having } [1, 0]^\top \Delta y = 0 \\ = 0 & \text{if } g_2 \text{ is active with } \bar{\lambda}_2 = 0 \text{ having } [1, 0]^\top \Delta y < 0 \text{ or } g_2 \text{ is inactive} \\ \text{free} & \text{if } g_2 \text{ is active with } \bar{\lambda}_2 > 0, \text{ then } z_2 [1, 0]^\top \Delta y = 0 \end{cases}$$

Here, $\Sigma = \frac{1}{n}\sum_{i=1}^{n} \left(x_i^{ind}\right)^2$, $\begin{bmatrix} \kappa_1 \\ \kappa_2 \end{bmatrix} = \frac{1}{n} \left[2\bar{w}\bar{x}_a^{ind} + \bar{b} - \bar{x}_a^{dep} - \bar{x}_a^{ind}\bar{w}\Sigma - \Sigma \right]$.

As noted in Section 4.1.2, the solution to Eq (30) depends on three cases.

1. No constraint is active. The analysis for this case is simple, as $z_1 = 0$ and $z_2 = 0$

$$\Delta y = \begin{bmatrix} \Delta w \\ \Delta b \end{bmatrix} = -\frac{1}{n\Sigma} \begin{bmatrix} \kappa_1 \\ \kappa_2 \end{bmatrix} \Delta x_a$$

Inserting it into Eq (30), we have

$$DG(\bar{x}; \Delta x_a) = \left(-\frac{\bar{b}}{n\bar{w}^2\Sigma} \left[\bar{w}\bar{x}_a^{ind} + \left(\bar{w}\bar{x}_a^{ind} + \bar{b} - \bar{x}_a^{dep} \right) - \bar{x}_a^{ind} \right] + \begin{bmatrix} \frac{1}{n} & -\frac{1}{n\bar{w}} \end{bmatrix} \right) \Delta x_a = d_{\Delta x}^1(\bar{x}) \bullet \Delta x_a \tag{31}$$

Here, the notation $d^1_{\Delta x}(\bar{x})$ is introduced to facilitate our discussion. From Eq (31), $DG(\bar{x};\Delta x_a)$ could be optimized (achieving the maximum impact) by setting Δx_a parallel to $d^1_{\Delta x}(\bar{x})$.

2. g_1 *is active.* In this case, $z_2 = 0$ and Eq (30) is simplified as

$$\Delta y = \begin{bmatrix} \Delta w \\ \Delta b \end{bmatrix} = -\frac{1}{\Sigma}\left(\begin{bmatrix} \kappa_1 \\ \kappa_2 \end{bmatrix}\Delta x_a + z_1 \begin{bmatrix} -1 \\ 0 \end{bmatrix}\right),$$
$$\text{with,} \quad z_i \begin{cases} \geq 0 & \text{if } \bar{\lambda}_1 = 0 \,\&\, \Delta w = 0 \\ = 0 & \text{if } \bar{\lambda}_1 = 0 \,\&\, \Delta w > 0 \\ Free & \text{if } \bar{\lambda}_1 > 0 \end{cases} \tag{32}$$

a) If $\bar{\lambda}_1 = 0$, the solution depends on the perturbation direction Δx_a.

o If $\kappa_1^\top \Delta x_a \geq 0$, we have $\Delta y = \begin{bmatrix} 0 \\ \frac{1}{n}[-\bar{w} \quad 1] \end{bmatrix}\Delta x_a$. Inserting it into Eq (32), we have

$$DG(\bar{x};\Delta x_a) = \frac{-1}{n\bar{w}}[\bar{b}\ 1]\Delta x_a = d^2_{\Delta x}(\bar{x}) \bullet \Delta x_a \tag{33}$$

Here, notation $d^2_{\Delta x_a}(\bar{x})$ is introduced to ease our discussion later.

o If $\kappa_1^\top \Delta x_a < 0$, we have $DG(\bar{x};\Delta x_a) = d^1_{\Delta x_a}(\bar{x}) \bullet \Delta x_a$ following Eq (31).

b) If $\bar{\lambda}_1 > 0$, the solution needs to meet the requirement $z_1[-1, 0]^\top \Delta y = 0$. Since z_1 is free, this means either $z_1 = 0$ or $\Delta w = 0$. The former case leads to $DG(\bar{x};\Delta x_a) = d^1_{\Delta x}(\bar{x}) \bullet \Delta x_a$ following Eq (31), while the latter case leads to $DG(\bar{x};\Delta x_a) = d^2_{\Delta x}(\bar{x}) \bullet \Delta x_a$ following Eq (33).

3. g_2 *is active.* In this case, we have $z_1 = 0$. The analysis is similar to the previous case and the same two groups of semi-derivatives (i.e., $d^1_{\Delta x}(\bar{x}) \bullet \Delta x_a$ and $d^2_{\Delta x}(\bar{x}) \bullet \Delta x_a$) are obtained depending the direction of Δx_a). The only difference with the previous case is that when $\bar{\lambda}_2 = 0$, $d^2_{\Delta x}(\bar{x})$ is obtained when $\kappa_1^\top \Delta x_a \leq 0$ instead of $\kappa_1^\top \Delta x_a \geq 0$.

1.5. Checking Lipschitz continuity of SVM model

Following Theorem 1 introduced earlier, we check two sufficient conditions to ensure that *S has a Lipschitz continuous single-valued s*.
The gradients $\nabla_y g_i(\bar{x},\bar{y})$ for all active constraints are linearly independent
The gradients consist of

$$\nabla_y g_i^\alpha(\bar{x},\bar{y}) = [-l_i \boldsymbol{x}_i \quad -l_i \quad 0 \quad \cdots \quad -1 \quad \cdots \quad 0] \text{for } i \in R_0 \cup V \cup E_0 \cup E_1 \tag{34}$$

$$\nabla_y g_i^\mu(\bar{x},\bar{y}) = [0 \quad 0 \quad 0 \quad \cdots \quad -1 \quad \cdots \quad 0] \text{for } i \in E_0 \cup V \cup R_0 \cup R_1$$

Condition (*i*) depends on the specific dataset. In the following, we discuss that the gradient vectors in Eq (34) are unlikely linearly dependent in practice, based on several observations from real-world modeling. Our experiment in Section 4.2 shows that these gradient vectors specified on the vehicle classification dataset are indeed linearly independent.

o R_0 and E_0 stand for rarely cases (where constraints are active and Lagrangian multipliers are zero) and are often empty in real-world datasets.

o A vector in $\nabla_y g_i^\alpha(\bar{x}_a,\bar{y})$ for $i \in E_1$ and $\nabla_y g_i^\mu(\bar{x}_a,\bar{y})$ for $i \in R_1$ is independent with other vectors (i.e., cannot be expressed as a linear combination of other vectors). Our attention turns to checking whether there is a $j \in V$ having the following linear combination:

$$\nabla_y g_j^\alpha(\bar{x}_a,\bar{y}) = \sum_{i \in V \cup E_1, j \neq i} \beta_i^\alpha \times \nabla_y g_i^\alpha(\bar{x}_a,\bar{y}) + \sum_{i \in V \cup R_1} \beta_i^\mu \times \nabla_y g_i^\mu(\bar{x}_a,\bar{y}) \tag{35}$$

Our focus is to verify whether there is a vector $[-l_j \boldsymbol{x}_j \; -l_j]$ ($j \in V$) can be expressed as linear combination of other vectors in V (i.e., $[-l_i \boldsymbol{x}_i \; -l_i]$ ($i \in V, i \neq j$)), as we cannot use any vectors $[-l_i \boldsymbol{x}_i \; -l_i]$ ($i \in E_1 \cup R_1$) to express $[-l_j \boldsymbol{x}_j \; -l_j]$. Otherwise the vectors from $E_1 \cup R_1$ introduce nonzero elements to the right side of $\nabla_y g_j^{\alpha}(\bar{x}_a, \bar{y})$ in Eq (35) and there is no way to cancel them out.

There is low chance to express $[-l_j \boldsymbol{x}_j \; -l_j]$ using $[-l_i \boldsymbol{x}_i \; -l_i]$ belonging to V, expecially when V contains only a few points (the feature of SVM). Often times, it consists of only two data points, sitting right on the two marginal boundaries, respectively.

In applications, we numerically check the linearly independence. Our test later shows that the vectors are indeed linearly independent on the vehicle classification dataset.

$\frac{1}{2}\langle \Delta y, \nabla^2_{yy} L(\bar{x}, \bar{y}, \bar{\alpha}, \bar{\mu}) \Delta y \rangle > 0$ii. for every nonzero $\Delta y \in M^+$.

Noticing that

$$\frac{1}{2}\left\langle \Delta y, \nabla^2_{yy} L(\bar{x}_a, \bar{y}, \bar{\alpha}, \bar{\mu}) \Delta y \right\rangle = \frac{1}{2} \Delta w^{\top} \Delta w \tag{36}$$

We focus on showing that if Δy is nonzero, Δw shall be nonzero, which will yield $\frac{1}{2}\Delta w^{\top}\Delta w > 0$.

First, in subspace M^+, we need $\Delta y = (\Delta w, \Delta b, \Delta \xi_{i(i \in N)})$ be perpendicular to every gradient as follows:

$$\begin{array}{lccccccc} w & b & \xi_1 & \ldots & \xi_i & \ldots & \xi_n & \\ \nabla_y g_i^{\alpha}(\bar{\boldsymbol{x}}_a, \bar{y}) = & [-l_i \boldsymbol{x}_i & -l_i & 0 & \ldots & -1 & \ldots & 0] \quad \text{for } i \in V \cup E_0 \cup E_1 \quad (1) \\ \nabla_y g_i^{\mu}(\bar{\boldsymbol{x}}_a, \bar{y}) = & [0 & 0 & 0 & \ldots & -1 & \ldots & 0] \quad \text{for } i \in V \cup R_0 \cup R_1 \quad (2) \end{array} \tag{37}$$

With Eq (37–2), we can know that $\Delta \xi_i = 0$ for $i \in V \cup R_0 \cup R_1$.

If $\Delta w = 0$, from (37–1) we will need to require $(\Delta b, \Delta \xi_{i(i \in E_0 \cup E_1)})$ satisfying $\Delta y^{\top} \nabla_y g_i^{\alpha} = -l_i \Delta b = 0$, for every $i \in V$(notice $\xi_{i \in V} = 0$) and $\Delta y^{\top} \nabla_y g_i^{\alpha} = -l_i \Delta b - \Delta \xi_i = 0$,for every $i \in E_0 \cup E_1$. This means $\Delta b = 0$ and $\Delta \xi_i = 0$ for $i \in E_0 \cup E_1$, i.e., $\Delta y = 0$. This also implies that that Δw cannot be zero if Δy be nonzero. Therefore, condition (*ii*) is indeed satisfied.

Putting the observations on condition (*i*) and (*ii*) together, the mapping V for SVM here is most likely to have a Lipschitz continuous single-valued localization s around $\bar{x}$ for $(\bar{y}, \bar{\alpha}, \bar{\mu})$, though condition (*i*) needs to be validated on the specific dataset used by SVM.

1.6. Problem-specific constraints in SVM-based vehicle classification model

The feature values in the data are confined by practical limits such as vehicle movements and dynamics. Specifically, the feature values for vehicle classification are closely related to patterns of vehicle movement (e.g., speed) and dynamics (e.g., acceleration and deceleration). Given that only the variations of accelerations $\boldsymbol{x}^{ac}$ and decelerations $\boldsymbol{x}^{dc}$ are selected as the input, we add constraints by specifying the minimum and maximum of $\boldsymbol{x}^{ac}$ and $\boldsymbol{x}^{dc}$, which are related to vehicle design and driving conditions. In this study, we identify the bounds from the pristine dataset as the minimum and maximum feature values.

$$\begin{aligned} \rho_1 \leq x_i^{ac} \leq \rho_2 \\ \rho_3 \leq x_i^{dc} \leq \rho_4 \end{aligned} \tag{38}$$

1.7. Checking Lipschitz continuity for the vehicle classification model

Since Condition (*ii*) in Theorem 1 always holds, we only need to validate Condition (*i*) by checking that the gradients $\nabla_y g_i(\bar{x}, \bar{y})$ for all active constraints (Eq. (34)) for the vehicle classification dataset are linearly independent.

Our numerical test shows that the vectors are indeed linearly independent of the vehicle classification dataset. Specifically, we perform LU decomposition of the matrix consisting of these vectors and inspect the returned U matrix, finding no null pivot vector. Therefore, for our SVM model on the vehicle classification dataset, the mapping S is Lipschitz continuous and semi-differentiable around $\bar{x}$.

1.8. Solving the semi-derivative of SVM model

We need to evaluate changes in the decision boundary (i.e., $(\Delta w, \Delta b)$), which can be done following Eq (19), where (v, u) is specified

by $-B\Delta x_a$ in Eq. (20). We have

$$\Delta w = \bar{\alpha}_a l_a q + \sum_i^n z_i^\alpha l_i x_i$$
$$\sum_i^n z_i^\alpha l_i = 0$$
$$z_i^\alpha = -z_i^\mu \quad \text{for } (i \in N)$$

$$\text{with } z_i^\alpha \begin{cases} \geq 0 & \text{for } i \in R_0 \text{ having } -l_i\left(x_i^\top \Delta w + \Delta b\right) - \Delta\xi_i + u_i^\alpha = 0 \\ = 0 & \text{for } i \in R_0 \text{ having } -l_i\left(x_i^\top \Delta w + \Delta b\right) - \Delta\xi_i + u_i^\alpha < 0 \text{ and } i \in R_1 \\ \textit{free} & \text{for } i \in V \cup E_0 \cup E_1, \end{cases} \tag{39}$$

where, $u_i^\alpha = -l_a \bar{w} q$ if $i = a$, otherwise $u_i^\alpha = 0$

$$z_i^\mu \begin{cases} \geq 0 & \textit{for } i \in E_0 \text{ having } -\Delta\xi_i = 0 \\ = 0 & \text{for } i \in E_0 \text{ having } -\Delta\xi_i < 0 \textit{ and } i \in E_1 \\ \textit{free} & \text{for } i \in V \cup R_0 \cup R_1 \end{cases}$$

The result in Eq. (39) can be further simplified following observations below.

Note that for $i \in R_1$ we have $z_i^\alpha = 0$, and for $i \in E_1$ we have $z_i^\mu = 0$ and thus $z_i^\alpha = 0$. We have the three groups of points with their z_i^α being nonzero: $V, R_0^{'}, E_0^{'}$. $R_0^{'}, E_0^{'}$ are specified below.

Denote $i \in R_0^{'} \subset R_0$ as those having $-l_i(x_i^\top \Delta w + \Delta b) - \Delta\xi_i + u_i^\alpha = 0$, leading to $z_i^\alpha \geq 0$. For the VI, we have $z_i^\mu \Delta\xi_i = z_i^\alpha \Delta\xi_i = 0$ for $i \in R_0$. Therefore, we have $z_i^\alpha\left(-l_i\left(x_i^\top \Delta w + \Delta b\right) + u_i^\alpha\right) = 0$ for $i \in R_0^{'}$. The VI also states that for $i \in V$, z_i^μ and z_i^α are unrestricted, and $z_i^\mu \Delta\xi_i = 0$ and $z_i^\alpha\left(-l_i\left(x_i^\top \Delta w + \Delta b\right) - \Delta\xi_i + u_i^\alpha\right) = 0$. Thus, $z_i^\alpha\left(-l_i\left(x_i^\top \Delta w + \Delta b\right) + u_i^\alpha\right) = 0$ for $i \in V$.

Denote $i \in E_0^{'} \subset E_0$ are those having $\Delta\xi_i = 0$, leading to $z_i^\mu \geq 0$ and thus $z_i^\alpha \leq 0$. For the VI, we have $z_i^\alpha\left(-l_i\left(x_i^\top \Delta w + \Delta b\right) - \Delta\xi_i + u_i^\alpha\right) = 0$ for $i \in E_0$. Therefore, we have $z_i^\alpha\left(-l_i\left(x_i^\top \Delta w + \Delta b\right) + u_i^\alpha\right) = 0$ for $i \in E_0^{'}$.

Putting these observations together, Eq. (39) is rewritten as:

$$\Delta w = \bar{\alpha}_a l_a q + \sum_{i \in R_0^{'} \cup V \cup E_0^{'}} z_i^\alpha l_i x_i$$

$$\sum_{i \in R_0^{'} \cup V \cup E_0^{'}} z_i^\alpha l_i = 0$$

$z_i^\alpha\left(-l_i\left(x_i^\top \Delta w + \Delta b\right) + u_i^\alpha\right) = 0$ for $i \in R_0^{'} \cup V \cup E_0^{'}$.

If $i = a$, $u_i^\alpha = -l_a \bar{w} \Delta x_a$; else, $u_i^\alpha = 0$. $(\Delta w, \Delta b)$ can be obtained by solving a set of equations.

1.9. Tests on NGSIM dataset

We also conduct additional experiments by applying the proposed attack method to the NGSIM dataset. Figs. A2 and A3 show that the dataset allows us to estimate delay patterns and test vehicle classification, respectively.

We evaluate the proposed semi-derivative-based attack model by testing attacks against both the LS and SVM models using NGSIM data. The results, depicted in Figure A4-(i) and Figure A4-(ii), demonstrate that both the proposed attack model and the gradient-based attack model can deviate the model solution. Similar to Figs. 2 and 3 in the main text, the proposed attack method outperforms the gradient method in terms of deviating the model solution. Notably, Figure A4-(i) clearly demonstrates that the proposed attack still works even after the constraints on the model solution are activated, albeit at a reduced rate. Note that the constraints on the model solution stays *inactive* under the gradient attack method before the algorithm stops (when perturbations added to data are too large). Figure A4-(ii) shows that the gradient-based attack deviates the model solution slower. This is because that the gradient-based attack ignores the inequality constraints in the SVM model, which leads to sub-optimal perturbation directions and further reduces the deviation rate. These findings validate that the proposed attack method and the conclusions drawn in the main text can be generalized to other datasets.

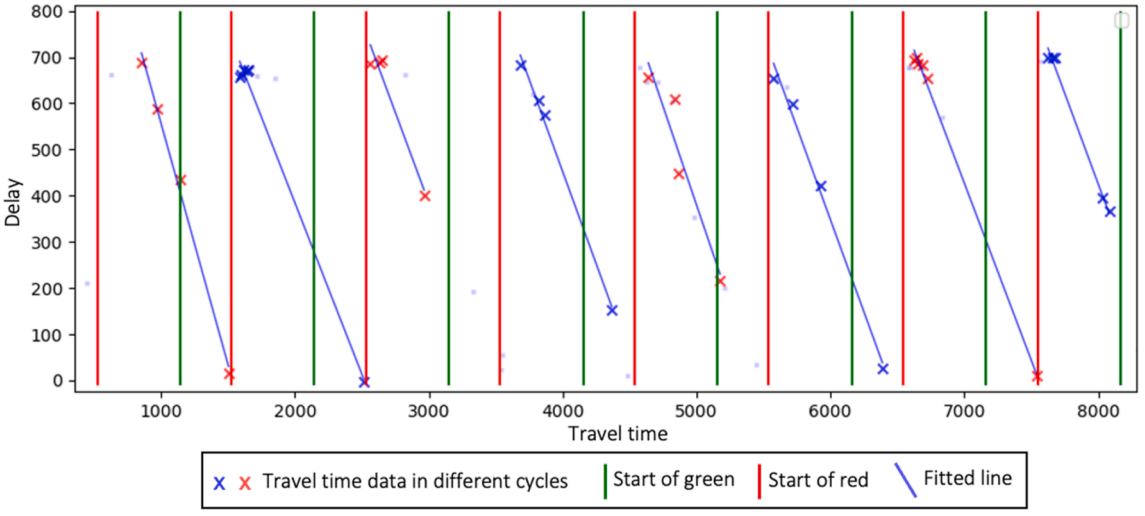

Fig. A2. NGSIM dataset for delay pattern estimation.

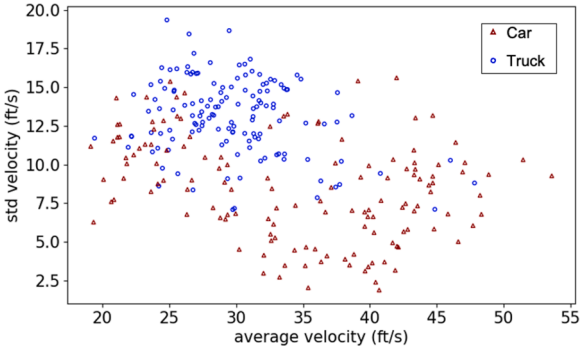

Fig. A3. NGSIM dataset for vehicle classification.

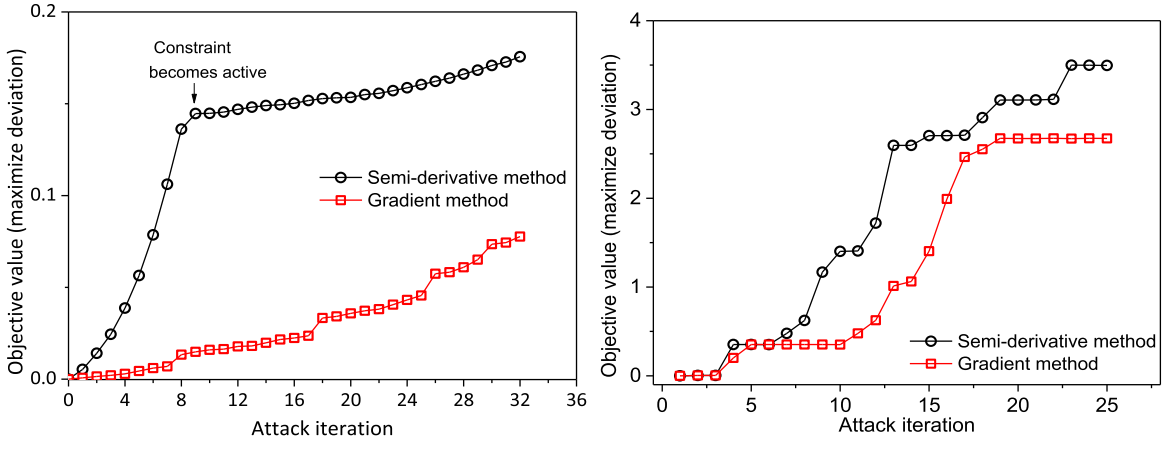

Fig. A4. (i) Attacking against queue length estimation and (ii) vehicle classification using NGSIM dataset.

1.10. Addition-type attack model

The proposed (perturbation-type) attack model can be easily adapted to addition-type attacks by adding (new) poisoning data. The algorithm for addition-type attacks is shown in Figure A5.

Algorithm 2. Data Poisoning Attack Algorithm by Adding New Poisons

Input: The clean data $x = [x_i]$ $(i \in N)$, TSEP model and objective value $G(x, y)$.

Repeat:

1. Initialize a new data point x_a by duplicating a (randomly selected) pristine point near the decision boundary.
2. Optimizing point a

   Repeat:

   Evaluate semi-derivative $DG\big(x \cup x_a, \hat{y}(x \cup x_a)\big)(\Delta x_a)$ given a unit perturbation Δx_a at x_a, and find

   $$\Delta x_a^* = \underset{\Delta x_a}{\operatorname{argmax}}\, DG\big(x \cup x_a, \hat{y}(x \cup x_a)\big)(\Delta x_a)$$

   Move point a following direction Δx_a^* by a specific step size η

   $$x_a = x_a + \eta \Delta x_a^*.$$

   Re-learn the model and evaluate the objective value $G\big(x \cup x_a, \hat{y}(x \cup x_a)\big)$.

   Until: Change in the objective value is sufficiently small or constraint on x_a are met.
3. Expand the dataset $x = x \cup x_a$, re-learn TSEP model on the expanded (poisoned) data, and evaluate the attack performance.

Until: Improvement in the attack performance is sufficiently small.

Output: Poisoned TSEP model and the poisoned dataset

Fig. A5. Semi-derivative-based Data Poisoning Attack Algorithm by Adding New Poisons

1.11. An example model with a large Lipschitz constant

Consider a general LS model

$$\min_{w,b} \sum_{i=1}^{n} \left(wx_i + b - y_i\right)^2 + C\left(w^2 + b^2\right)$$

with data points $(x_i, y_i), i = 1, \cdots, n$. The value of C affects the property of the model, including the sensitivity of model solution w to the perturbation in data. Note that by setting $C = 0$, we have the queue length estimation model investigated in our study. We demonstrate that at specific value of C, the Lipschitz constant of w (model solution) with respect to perturbation of certain data point x_j would be extremely large, resulting in a model that is practically not Lipschitz continuous.

The derivative of w with a single input data x_j has an explicit formula:

$$\frac{\partial w}{\partial x_j} = \frac{\left[(n+C)y_j - \sum y_i\right]}{Z} - \frac{\left[(n+C)\sum x_i y_i - \sum x_i \sum y_i\right]\left[2(n+C)x_j - 2\sum x_i\right]}{Z^2}$$

$$Z = (n+C)\left(\sum x_i^2 + C\right) - \left(\sum x_i\right)^2$$

Then, the Lipschitz constant of w at x_j can be evaluated by $Lipw(x_j) = \left|\frac{\partial w}{\partial x_j}\right|$, which is data-dependent. We simulate 10 random data points from normal distributions to show that $Lipw(x_j)$ could be extremely large under certain C. Specifically, we have $x_i \sim \text{Uniform}(0, 5)$ $(i \in [1, \cdots, 10])$, $y_i \sim \mathcal{N}(0, 1)$ $(i \in [1, \cdots, 8])$ and $y_i \sim \mathcal{N}(200, 1)$ $(i \in [9, 10])$ serving as two outliers.

Our simulation finds that $Lipw(x_9)$ can be large with negative C and the value of w is not stable even for a slight perturbation of x_9. For example, when $C = -90$, $Lipw(x_9)$ surpasses 20000, and a perturbation of x_9 by 1 shifts w from 1960 to -206. Note that a negative C is not practical in real applications but used here to demonstrate the concept that the Lipschitz property is helpful to study the sensitivity of TSEP models and identify the ill-posed/badly-designed ones.

Such instability arises primarily from the non-linear term. Due to the prevalence of non-linear terms in neural networks, there is growing evidence and concern among researchers about the instability of neural network models. Lipschitz continuity is often used to quantify whether such instability exists. For instance, Kim et al. (2021) demonstrated that self-attention, a commonly employed layer in neural networks, does not maintain Lipschitz continuity in the case of unbounded input domains.

References

Akhtar, N., Mian, A., 2018. Threat of Adversarial attacks on deep Learning in computer vision: a survey. IEEE Access 6, 14410–14430. https://doi.org/10.1109/ACCESS.2018.2807385.

Almalki, S.A., Song, J., 2020. A Review on Data Falsification-Based attacks In Cooperative Intelligent Transportation Systems 17.

Ban, X., (Jeff), Hao, P., Sun, Z.,, 2011. Real time queue length estimation for signalized intersections using travel times from mobile sensors. Transportation Research Part C: Emerging Technologies 19, 1133–1156. https://doi.org/10.1016/j.trc.2011.01.002.

Ban, X. (Jeff), Herring, R., Bayen, A.M., Hao, P., 2009. Delay pattern estimation for signalized intersections using sampled travel times. Transportation Research Record, in: Transportation Research Board of the National Academies.

Biggio, B., Nelson, B., Laskov, P., 2013. Poisoning Attacks against Support Vector Machines. arXiv:1206.6389 [cs, stat].

Biggio, B., Nelson, B., Laskov, P., 2011. Support vector machines under Adversarial label noise. In: Proceedings of the Asian Conference on Machine Learning. Presented at the Asian Conference on Machine Learning, pp. 97–112.

Boeira, F., Asplund, M., Barcellos, M.P., 2018. Vouch: A Secure Proof-of-Location Scheme for VANETs, in: Proceedings of the 21st ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems. Presented at the MSWIM '18: 21st ACM Int'l Conference on Modelling, Analysis and Simulation of Wireless and Mobile Systems, ACM, Montreal QC Canada, pp. 241–248. 10.1145/3242102.3242125.

Chen, C., Ban, X. (Jeff), Wang, F., Wang, J., Siddique, C., Fan, R., Lee, J., 2017. Understanding GPS and Mobile Phone Data for Origin-Destination Analysis (No. FHWA-HEP-19-027). Federal Highway Administration.

Chen, J., Xu, H., Wu, J., Yue, R., Yuan, C., Wang, L., 2019. Deer crossing road detection with roadside LiDAR sensor. IEEE Access 7, 65944–65954.

Dadras, S., Gerdes, R.M., Sharma, R., 2015. Vehicular Platooning in an Adversarial Environment, in: Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security. Presented at the ASIA CCS '15: 10th ACM Symposium on Information, Computer and Communications Security, ACM, Singapore Republic of Singapore, pp. 167–178. 10.1145/2714576.2714619.

Dontchev, A.L., Rockafellar, R.T., 2009. Implicit functions and solution mappings. Springer.

DOT, 2018. Next generation simulation (NGSIM) vehicle trajectories and supporting data. US Department of Transportation.

El-Rewini, Z., Sadatsharan, K., Selvaraj, D.F., Plathottam, S.J., Ranganathan, P., 2020. Cybersecurity challenges in vehicular communications. Veh. Commun. 23, 100214 https://doi.org/10.1016/j.vehcom.2019.100214.

Feng, Y., Huang, S., Chen, Q.A., Liu, H.X., Mao, Z.M., 2018. Vulnerability of traffic control system under cyberattacks with falsified data. Transp. Res. Rec. 2672, 1–11. https://doi.org/10.1177/0361198118756885.

Feng, Y., Huang, S.E., Wong, W., Chen, Q.A., Mao, Z.M., Liu, H.X., 2022. On the Cybersecurity of traffic signal control system with connected vehicles. IEEE Trans. Intell. Transp. Syst. 23, 16267–16279. https://doi.org/10.1109/TITS.2022.3149449.

Ferrari Dacrema, M., Cremonesi, P., Jannach, D., 2019. In: Are We Really Making Much Progress? A Worrying Analysis of Recent Neural Recommendation Approaches, in. Association for Computing Machinery, New York, NY, USA, pp. 101–109. https://doi.org/10.1145/3298689.3347058.

Fiacco, A.V., 1983. Introduction to sensitivity and stability analysis in Nonlinear programming. Academic Press.

Fischler, M.A., Bolles, R.C., 1981. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM 24, 381–395. https://doi.org/10.1145/358669.358692.

Hao, P., Ban, X., (Jeff), Guo, D., Ji, Q.,, 2014. Cycle-by-cycle intersection queue length distribution estimation using sample travel times. Transp. Res. B Methodol. 68, 185–204. https://doi.org/10.1016/j.trb.2014.06.004.

Hasan, M., Mohan, S., Shimizu, T., Lu, H., 2020. Securing vehicle-to-everything (V2X) communication platforms. IEEE Trans. Intell. Veh. 5, 693–713. https://doi.org/10.1109/TIV.2020.2987430.

Huang, S.E., Chen, Q.A., Feng, Y., Mao, Z.M., Wong, W., Liu, H.X., 2021. Impact evaluation of falsified data attacks on connected vehicle based traffic signal control systems. In: Presented at the Workshop on Automotive and Autonomous Vehicle Security (AutoSec), p. 25.

Huang, L., Joseph, A.D., Nelson, B., Rubinstein, B.I., Tygar, J.D., 2011. Adversarial machine learning. In: Presented at the Proceedings of the 4th ACM Workshop on Security and Artificial Intelligence, pp. 43–58.

Huber, P.J., 1964. Robust estimation of a location Parameter. Ann. Math. Stat. 35, 73–101.

Jagielski, M., Oprea, A., Biggio, B., Liu, C., Nita-Rotaru, C., Li, B., 2018. In: Manipulating Machine Learning: Poisoning Attacks and Countermeasures for Regression Learning. IEEE, San Francisco, CA, pp. 19–35. https://doi.org/10.1109/SP.2018.00057.

Kim, H., Papamakarios, G., Mnih, A., 2021. The lipschitz constant of self-attention. In: Proceedings of the 38th International Conference on Machine Learning. Presented at the International Conference on Machine Learning, pp. 5562–5571.

Koh, P.W., Steinhardt, J., Liang, P., 2021. Stronger Data Poisoning Attacks Break Data Sanitization Defenses. 10.48550/arXiv, 1811.00741.

Li, W., Ban, X., 2019. Connected vehicles based traffic signal timing optimization. IEEE Trans. Intell. Transp. Syst. 20, 4354–4366. https://doi.org/10.1109/TITS.2018.2883572.

Li, B., Vorobeychik, Y., 2018. Evasion-robust classification on Binary domains. ACM Trans. Knowl. Discov. Data 12, 1–32. https://doi.org/10.1145/3186282.

Li, W., Wang, J., Fan, R., Zhang, Y., Guo, Q., Siddique, C., Ban, X., (Jeff), 2020. Short-term traffic state prediction from latent structures: Accuracy vs. efficiency. Transportation Research Part c: Emerging Technologies 111, 72–90. https://doi.org/10.1016/j.trc.2019.12.007.

Liu, F., Liu, H., Jiang, W., 2022. Practical Adversarial Attacks on Spatiotemporal Traffic Forecasting Models. 10.48550/arXiv, 2210.02447.

Liu, Y., Ning, P., Reiter, M.K., 2011. False data injection attacks against state estimation in electric power grids. ACM Trans. Inf. Syst. Secur. 14 https://doi.org/10.1145/1952982.1952995.

Liu, F., Tian, J., Miranda-Moreno, L., Sun, L., 2023. Adversarial danger identification on temporally dynamic graphs. IEEE Trans. Neural Netw. Learning Syst. 1–12 https://doi.org/10.1109/TNNLS.2023.3252175.

Luo, Z.-Q., Pang, J.-S., Ralph, D., 1996. Mathematical programs with equilibrium constraints. Cambridge University Press.

Mei, S., Zhu, X., 2015. Using machine teaching to identify optimal training-set attacks on machine learners. Presented at the Twenty-Ninth AAAI Conference on Artificial Intelligence.

Mercader, P., Uwayid, W., Haddad, J., 2020. Max-pressure traffic controller based on travel times: an experimental analysis. TransportationResearch Part c: Emerging Technologies 110, 275–290. https://doi.org/10.1016/j.trc.2019.10.002.

Papernot, N., McDaniel, P., Jha, S., Fredrikson, M., Celik, Z.B., Swami, A., 2016. The limitations of deep learning in adversarial settings. Presented at the 2016 IEEE European symposium on security and privacy (EuroS&P), IEEE, pp. 372–387.

Petit, J., Shladover, S.E., 2015. Potential cyberattacks on automated vehicles. IEEE Trans. Intell. Transp. Syst. 16, 546–556. https://doi.org/10.1109/TITS.2014.2342271.

Reda, H.T., Anwar, A., Mahmood, A.N., Tari, Z., 2021. A Taxonomy of Cyber Defence Strategies Against False Data Attacks in Smart Grid. arXiv:2103.16085 [cs, eess].

Scaman, K., Virmaux, A., 2019. Lipschitz regularity of deep neural networks: analysis and efficient estimation.

Schmidt, D., Radke, K., Camtepe, S., Foo, E., Ren, M., 2016. A survey and analysis of the GNSS spoofing threat and countermeasures. ACM Comput. Surv. 48 (64), 1–64. https://doi.org/10.1145/2897166.

Shafahi, A., Huang, W.R., Najibi, M., Suciu, O., Studer, C., Dumitras, T., Goldstein, T., 2018. Poison frogs! Targeted Clean-Label Poisoning Attacks on Neural Networks, in: Advances in Neural Information Processing Systems. Curran Associates Inc.

Sun, Z., Ban, X., (Jeff), 2013. Vehicle classification using GPS data. Transportation Research Part C: Emerging Technologies 37, 102–117. https://doi.org/10.1016/j.trc.2013.09.015.

Suya, F., Mahloujifar, S., Suri, A., Evans, D., Tian, Y., 2021. Model-targeted poisoning attacks with provable convergence. International Conference on Machine Learning. PMLR 10000–10010.

Toch, E., Lerner, B., Ben-Zion, E., Ben-Gal, I., 2019. Analyzing large-scale human mobility data: a survey of machine learning methods and applications. Knowl Inf Syst 58, 501–523. https://doi.org/10.1007/s10115-018-1186-x.

Tramèr, F., Boneh, D., 2021. Differentially Private Learning Needs Better Features (or Much More Data). 10.48550/arXiv.2011.11660.
Vorobeychik, Y., Kantarcioglu, M., 2018. Adversarial machine Learning. Synthesis Lectures on Artificial Intelligence and Machine Learning 12, 1–169. https://doi.org/10.2200/S00861ED1V01Y201806AIM039.
Wang, F., Hong, Y., Ban, X., 2023a. Infrastructure-enabled GPS spoofing detection and correction. IEEE Trans. Intell. Transp. Syst. 24, 13878–13892. https://doi.org/10.1109/TITS.2023.3298785.
Wang, F., Wang, X., Hong, Y., Ban, J., 2023b. Infrastructure-enabled defense solution against data Poisoning attacks on queue length estimation using Mobile sensors. SSRN Electron. J.
Wang, X., Wang, F., Hong, Y., Ban, X., 2023c. Lipschitz continuity and data Poisoning attacks: applications in intelligent transportation systems. SSRN Electron. J.
Xiao, H., Biggio, B., Brown, G., Fumera, G., Eckert, C., Roli, F., 2015a. Is Feature Selection Secure against Training Data Poisoning?, in: Proceedings of the 32nd International Conference on Machine Learning. Presented at the International Conference on Machine Learning, PMLR, pp. 1689–1698.
Xiao, H., Biggio, B., Nelson, B., Xiao, H., Eckert, C., Roli, F., 2015b. Support vector machines under adversarial label contamination. Neurocomput 160, 53–62.
Xie, S., Wang, H., Kong, Y., Hong, Y., 2022. Universal 3-dimensional perturbations for black-box attacks on video recognition systems. IEEE Security Privacy 19.
Yang, X., Sun, Z., Ban, X., Holguín-Veras, J., 2014. Urban freight delivery stop identification with GPS data. Transportation Research Record: Journal of the Transportation Research Board 2411, 55–61. https://doi.org/10.3141/2411-07.
Yu, J.J.Q., 2021. Sybil attack identification for crowdsourced navigation: a self-supervised deep Learning approach. IEEE Trans. Intell. Transport. Syst. 22, 4622–4634. https://doi.org/10.1109/TITS.2020.3036085.
Zhang, F., 2006. The schur complement and its applications. Springer Science & Business Media.
Zhu, L., Feng, K., Pu, Z., Ma, W., 2021. Adversarial Diffusion Attacks on Graph-based Traffic Prediction Models. 10.48550/arXiv.2104.09369.