Transportation Research Part C 128 (2021) 103196

Contents lists available at ScienceDirect

Transportation Research Part C

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

A data-driven method for falsified vehicle trajectory identification by anomaly detection

Shihong Ed Huang [a], Yiheng Feng [b,*], Henry X. Liu [a]

[a] Department of Civil and Environmental Engineering, University of Michigan, 2350 Hayward Street, Ann Arbor, MI 48109, United States
[b] Lyles School of Civil Engineering, Purdue University, 550 Stadium Mall Drive, West Lafayette, IN 47907, United States

ARTICLE INFO

Keywords:
Connected vehicles
Cybersecurity
Falsified data attack
Misbehavior detection
Trajectory embedding

ABSTRACT

The vehicle-to-infrastructure (V2I) communications enable a wide range of new applications, which bring prominent benefits to the transportation system. However, malicious attackers can potentially launch falsified data attacks against V2I applications to jeopardize the traffic operation. To ensure the benefits brought by the V2I applications, it is critical to protect the applications from those cyber-attacks. However, existing literature on the defense solution that protects the V2I applications is very limited. This paper aims to fill this research gap by proposing a data-driven method to identify falsified trajectories generated by compromised connected vehicles (CVs). A trajectory embedding model, inspired by the word embedding model from the natural language processing (NLP) community, is developed. The proposed trajectory embedding model generates vector representations of vehicle trajectories that can be used to compute the similarities between trajectories. The proposed method consists of two steps. In the first step, historical trajectory data are used to train a neural network and obtain the vector representations of trajectories. The second step computes a distance matrix between each pair of trajectories and identifies falsified trajectories using a hierarchical clustering algorithm. Simulation experiments show that the proposed method has a very high detection rate (>97.0%) under different attack goals with varying CV penetration rates from 100% to 25%, while the false alarm rate remains low. It has great potential to be implemented in a wide range of trajectory-based CV applications such as traffic state estimation and traffic signal control, to safeguard the CV system from cyber threats.

1. Introduction

Vehicles and transportation infrastructure are being connected through advanced wireless communication technologies, such as Dedicated Short-Range Communications (DSRC) and cellular networks. Transportation infrastructure can greatly leverage real-time trajectories reported by connected vehicles (CVs) through vehicle-to-infrastructure (V2I) communications for a wide range of applications such as traffic state estimation and traffic signal control. Comparing with traditional traffic data collected from fixed-location infrastructure-based sensors (e.g., loop-detectors), CV trajectory data provide richer information across both temporal and spatial domains, from which more accurate traffic information can be inferred. Besides all the benefits brought by CVs, vehicle-infrastructure connectivity also opens a new door to cyber attacks. Malicious attackers can potentially launch cyber attacks against V2I applications

* Corresponding author.
E-mail addresses: edhuang@umich.edu (S.E. Huang), feng333@purdue.edu (Y. Feng), henryliu@umich.edu (H.X. Liu).

https://doi.org/10.1016/j.trc.2021.103196
Received 10 October 2020; Received in revised form 28 April 2021; Accepted 29 April 2021
Available online 26 May 2021

to jeopardize the operation and cause safety and mobility concerns. The benefits of trajectory-based applications can be realized only if the system is secure in cyberspace.

To guarantee the benefits brought by the V2I applications, the received CV trajectory data have to be authentic. However, due to the fact that the CV trajectories are normally self-reported (e.g., in the form of Basic Safety Messages (BSMs)), they can be falsified and do not represent true vehicle status (e.g., location and speed). To enhance the security of the CV system, the Security Credential Management System (SCMS) (Whyte et al., 2013; Brecht et al., 2018) is being implemented by USDOT. The SCMS requires that the sender signs each CV message with a digital certificate, after which the receiver verifies the signature before extracting information from the message. However, the attacker may use a legitimate communication device to transmit falsified data with valid digital certificates. In this case, the attacker does not spoof the sender's identity but only modifies the contents of a message (e.g., speed and location data). Falsified CV data can be signed properly to pass an SCMS identity check. It has been demonstrated that a malicious vehicle owner can hack into her personal vehicle by exploiting software vulnerabilities through the Electronic Control Units (ECU) (Koscher et al., 2010) or the infotainment system (Mazloom et al., 2016). Because the vehicle owner has arbitrary access to her own vehicle, launching such falsified trajectory attacks is achievable in practice.

Recent studies have shown that falsified vehicle trajectory data could be leveraged to attack the transportation system and affect a wide range of trajectory-based applications. For instance, routing decisions in navigation systems (e.g., Google Maps, Waze) (Jeske, 2013; Sinai et al., 2014.) and traffic signal control system at signalized intersections (Chen et al., 2018; Feng et al., 2018). Therefore, it is critical to protect CV-based applications from falsified trajectories, or trajectory spoofing attacks. One straightforward approach to identify falsified trajectories involves using other data sources to cross-validate suspicious trajectories. For instance, Canepa and Claudel (Canepa and Claudel, 2013; Canepa and Claudel, 2013) formulated a mixed-integer linear feasibility problem to detect falsified trajectories using loop-detector data. Traffic states were modeled using the LWR model. Loop detector data provided initial and boundary conditions for this model. The information (e.g., average speed) obtained from falsified trajectories could influence the traffic state estimation, making the original mixed-integer linear problem infeasible. Shoukry et al. (Shoukry et al., 2018) also used legacy loop detectors to estimate macroscopic traffic states. A set of honest vehicles were then identified, whose velocity values were consistent with macroscopic traffic states. Yan et al. (Yan et al., 2008) presented a system that used onboard radars to detect the presence of neighboring vehicles and verify their coordinates. However, the applicability of these methods is limited by the need to acquire additional onboard or infrastructure-based sensors, which may not exist at all times and locations.

Another approach for falsified trajectory identification is to model the problem as a misbehavior detection problem, which is also the focus of this study. According to different attack models, existing studies can be further classified into three categories. The first category is mostly focused on network-level behaviors such as route spoofing. Usually, the goal of the attacker or the malicious user is to maximize her personal rewards. This type of anomalous trajectory is commonly observed in the context of the taxi or ride-hailing fraud. Greedy taxi or ride-hailing drivers may take unnecessary detours and overcharge passengers (Zhu et al., 2015). One popular defense approach is to apply network models to identify anomalous routes. For example, (Zhang et al., 2011; Chen et al., 2013) divided the road network into grid cells of the same size. The key idea was to identify anomalous trajectories that traversing different cells from normal trajectories with the same origin and destination (OD). In (Lv et al., 2017) a clustering-based approach was applied to find popular routes given an OD pair. Then finding anomalous trajectories was considered as the outlier detection problem. A distance metric such as the edit distance used in (Zhu et al., 2015), is usually applied to represent the trajectory similarities in such approaches. In recent years, learning-based approaches are becoming more and more popular in anomalous route detection. Different learning structures are applied. (Oh and Iyengar, 2019) applied an inverse reinforcement learning (IRL) framework to learn the normal routing behaviors (i.e., the reward function) from demonstrations. Then the normality score was used to determine whether the observed trajectories have low normality scores (i.e., anomalous). Unlike other transportation applications where ground truth data is available (i.e., normal trajectories), most of the attack models do not have benchmarking falsified trajectories. To address this challenge, a Generative Adversarial Network (GAN) integrated with Gaussian mixture models was proposed in (Smolyak et al., 2020), in which falsified trajectory generation and detection could be performed simultaneously. Other studies considered various aspects in the trajectory generation process for anomaly detection. Wang et al. (Wang et al., 2020) included the geographical information in terms of road network topology constraints and used Recurrent Neural Network (RNN) and Convolutional Neural Network (CNN) to represent these features. Yao et al. (Yao et al., 2017) proposed to use Long Short Term Memory (LSTM) networks to capture the spatiotemporal variations in the trajectory clustering.

In the second category, the attacker tries to spoof trajectory data in vehicular communication networks and compromise vehicle-to-vehicle (V2V) applications. The main objective of the attacker is to induce improper vehicle maneuvers and cause safety risks. Vehicle location and speed are two common data elements that are spoofed and cooperative adaptive cruise control (CACC) is the most studied V2V application (Amoozadeh et al., 2015; Singh et al., 2018; Cui et al., 2018; Dadras et al., 2015; Alipour-Fanid et al., 2017), among others. Both simulation platforms (Amoozadeh et al., 2015; Singh et al., 2018; Cui et al., 2018) and control models (Dadras et al., 2015; Alipour-Fanid et al., 2017) were proposed to investigate the impact of trajectory data spoofing attacks as well as other types of cyber attacks (e.g., DoS, Sybil) on the safety performance of a single vehicle or the entire platoon (Wang et al., 2020). Defense solutions mainly focus on detecting misbehaviors at different layers of the application. Physical layer misbehavior detection analyzes the physical signal properties, such as received signal strength indicator (RSSI) (So et al., 2019; Nguyen et al., 2019), angle-of-arrival (AoA), and Doppler speed (Sun et al., 2017), to verify the sender's location. However, this method is not very accurate, especially considering that signals may bounce off between vehicles and other obstacles (Bißmeyer et al., 2010). The application layer misbehavior detection models analyze the content of the data and perform consistency and plausibility check. However, most of the existing studies only consider single vehicle behaviors such as consistency in location data (Ruj et al., 2011), consistency between location and speed data (So et al., 2018), or vehicle dynamics model (Yavvari et al., 2017), which ignores the surrounding traffic environment. A sophisticated attacker can easily penetrate the defense by incorporating vehicle dynamics constraints in the falsified trajectory generation process (Huang, 2020).

In the third category, the attacker tries to spoof trajectory data in the context of V2I applications. For example, Zhao et al. (Zhao et al., 2021) evaluated the impact of two trajectory spoofing strategies in highway cooperative merging scenario while another group of studies systematically investigated the vulnerabilities and impact of trajectory data spoofing attacks on both actuated traffic signal control system (Feng et al., 2018) and CV-based adaptive traffic signal control system (Chen et al., 2018; Huang et al., 2021; Yen et al., 2018). As for the defense solution, physical layer misbehavior detection models can still be applied, since V2V and V2I applications share the same communication network. However, in the application layer, there is very limited research in the detection and prevention of trajectory data spoofing attacks for V2I applications, especially associated with signalized intersections. Note that the SCMS does consider misbehavior detection of vehicles, but the current focus is on V2V applications. For V2I applications with different attack models, the current detection methods may not work. For example, the falsified CV data can be carefully designed so that they are complied with vehicle dynamics and relations with other vehicles (e.g., none-overlapping), which may not trigger the misbehavior detection mechanism but still greatly influence the V2I applications (See Section 5.1 for details). Therefore, the V2I applications demand new misbehavior detection strategies besides those currently in the SCMS. This study aims to fill this gap.

In this paper, a data-driven method for identifying falsified trajectories at signalized intersections is proposed. This method can be used to complement existing misbehavior detection mechanism in the SCMS for V2I applications. A trajectory embedding model, inspired by the word embedding model from the natural language processing (NLP) community, is developed. This model generates vector representations of vehicle trajectories, which can be used to compute the similarity between a pair of trajectories. The concept of trajectory embedding has been explored in previous studies such as (Yao et al., 2017; Song et al., 2018; Li et al., 2018; Wang et al., 2020; Cheng et al., 2019). However, most of them can only be applied to the route level, since they do not consider microscopic driving behaviors (e.g., car following). Microscopic driving behaviors are essential in identifying anomalous trajectories at the intersection level because such behaviors are key to many trajectory-based CV applications (Wang et al., 2020).

The proposed method consists of two steps. In the first step, trajectory data points are encoded based on three microscopic traffic features: range, range rate, and vehicle speed, which include not only the status (i.e., speed) of the ego vehicle but also relations with its leading vehicle (i.e., range and range rate). Then historical trajectory data are used to train a neural network and obtain the vector representations of trajectory points. The second step computes a distance matrix between each pair of trajectories and identifies falsified trajectories using a hierarchical clustering algorithm. Simulation results show that the proposed method has a very high detection rate while the false alarm rate remains low. This method can be applied under varying penetration rates of CVs and does not require additional data from any other sensors.

The rest of the paper is organized as follows. Section 2 describes the falsified trajectory identification problem and provides an overview of the proposed method. Section 3 details the trajectory embedding model that generates vector representations. Section 4 explains how to identify falsified trajectories using vector representations. Numerical examples are presented in Section 5 to demonstrate the effectiveness of the proposed method. Finally, Section 6 summarizes this paper.

2. Problem description and methodology overview

The proposed method is designed to be deployed at the infrastructure side in a mixed traffic condition, where not all vehicles are CVs. CV trajectories within the communication range (including the falsified trajectory) can be observed by the infrastructure (e.g., a roadside unit), but the trajectories of regular vehicles (non-CVs) are not observable. A time–space diagram with all vehicle trajectories at a signalized intersection is shown in Fig. 1. The solid blue curves represent observed CV trajectories and the dashed blue curves represent unobservable non-CV trajectories. The red curve represents a falsified trajectory. The objective of the proposed method is to identify the red curve from the solid blue curves. Note that the falsified trajectories can be generated in different ways, which depends

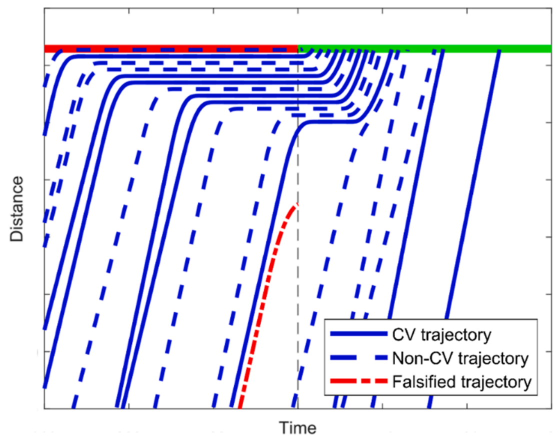


Fig. 1. Time-Space Diagram with Vehicle Trajectories.

on the attacker's goals (e.g., to cause safety issues, to reduce mobility performance, etc.) (Huang, 2020). In this particular example, the falsified trajectory intentionally slows down although the leading vehicle is still far away. More details about the attack trajectory generation will be introduced in the experiment section.

The problem is grounded on two basic assumptions. First, we assume that the majority of the observed trajectories are normal, while only a small portion of the trajectories are generated by the attacker, as shown in Fig. 1. This is a realistic assumption in real-world scenarios because of the existence of the SCMS deployed in the vehicular communication network. The attacker needs to acquire legitimate devices or vehicles and get digital certificates to send falsified data to avoid being detected by the SCMS, which limits the attacker's ability due to high costs. Second, the driving behaviors of falsified trajectories are different from normal trajectories in order to achieve certain attack goals. This behavioral difference is also supported by our previous studies (Chen et al., 2018; Huang et al., 2021), which showed that an attacker can launch effective attacks towards a CV-based traffic control system by generating anomalous vehicle speed and location data in the BSMs.

An overview of the methodology is shown in Fig. 2. This method includes two key parts: trajectory embedding and anomalous trajectory identification. A trajectory is composed of multiple trajectory data points and each trajectory data point reveals certain driving behaviors (i.e., the choice of range, range rate, and speed). Inspired by a word embedding model from the NLP community, a trajectory embedding model is developed. Each trajectory data point is converted into a word via trajectory pre-processing. The trajectory embedding model then encodes each word into a vector. The trajectory embedding model is a neural network with a single hidden layer. It is trained on both positive samples (correct context-target word pairs) and negative samples (incorrect context-target word pairs). The weights of the hidden layer are the vector representation of a word. The trained neural network implies that words that occur in a similar context would have similar vector representations (similar hidden layers). In the trajectory embedding model, this means that the trajectory data points that are consecutively observed in the temporal or spatial dimension would have similar vector representations. The vector representations enable the computation of trajectory similarity. Then falsified trajectories can be identified based on their similarity to other trajectories. The similarity between two trajectory data points is defined by a distance metric, which is calculated using the Euclidean distance between the two vectors that represent the two trajectory data points. The similarity between two trajectories is then calculated as the average distance of all trajectory data points over a common time window. A similarity matrix is then obtained by computing the similarity between all pairs of trajectories. Next, a hierarchical clustering algorithm is adopted to merge similar trajectories into clusters. A predefined threshold is used to terminate the merging process. Trajectories in the largest cluster are marked as normal, whereas other trajectories are considered anomalous (i.e., falsified) because the normal trajectories are the majority. The logic behind the proposed method is that falsified trajectories have a larger distance to the normal trajectories due to the behavioral differences. The hierarchical clustering algorithm merges normal trajectories into one single cluster, leaving the falsified trajectories being identified.

3. Trajectory embedding model

The trajectory embedding model is inspired by a word embedding model called word2vec (Mikolov et al., 2013a, 2013b, 2013c), a popular tool primarily used in the NLP community. Word2vec uses real-valued vectors to represent words. Each word is mapped into a

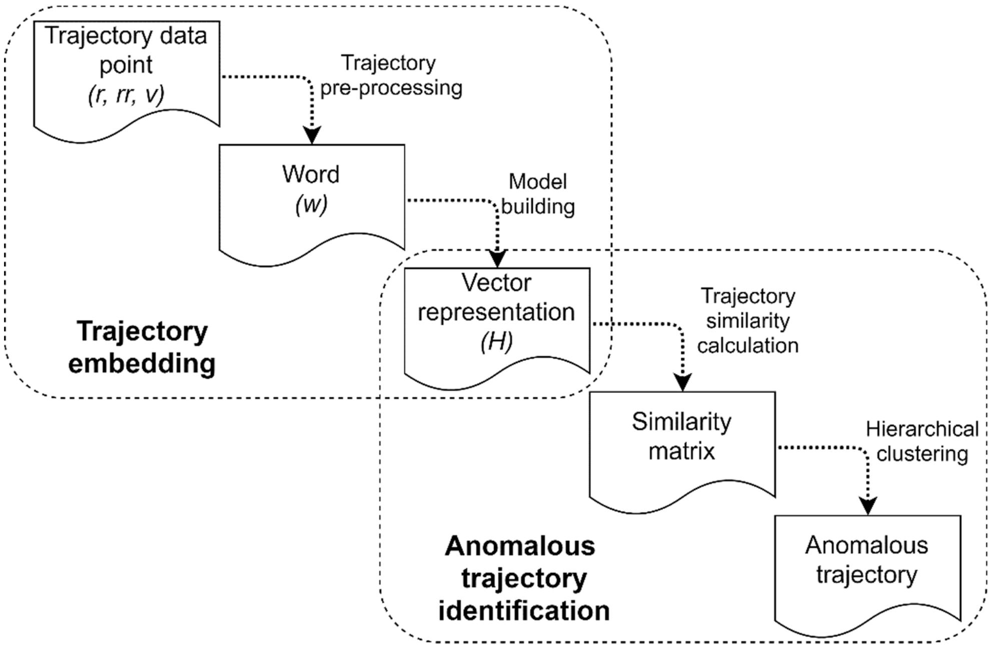


Fig. 2. Methodology Overview.

low dimensional vector (by “low” the dimension of the vector is compared to the size of the vocabulary of words). Words with similar meanings end up with similar vector representations. The vector representation carries semantic information and can be used for various downstream NLP applications. Just like a sentence is composed of multiple words, a trajectory is composed of multiple trajectory data points. Each trajectory data point can be considered an analog to a word in a sentence. The goal of the trajectory embedding model is to generate vector representations of trajectory data points. This section includes two parts: trajectory pre-processing and embedding model building. The first part describes how to convert a trajectory data point into a word. The second part explains how to train the model and generate vector representations of words (i.e., trajectory data points).

3.1. *Trajectory pre-processing*

Each trajectory data point contains the location and speed of the vehicle with a timestamp. For each trajectory data point, car-following information such as range (the space gap between the ego CV and its leading CV) and range rate (the speed difference between the ego CV and its leading CV) can be derived directly based on the trajectories of the two CVs. Note that when the CV market penetration rate is not 100%, the leading CV is not necessarily the immediate leading vehicle, as shown in Fig. 1. In other words, there may be non-CVs in between the CVs.

Range (r), range rate (rr), and speed (v) are considered as features to represent driving behaviors in this study. The three parameters are then mapped into a three-letter word using a mapping function $g(r, rr, v)$. The three letters correspond to the values of range, range rate, and speed, respectively. The mapping function $g(r, rr, v)$ discretizes the parameter space. In this study, range r has 26 possible intervals with 2 m per interval: [0, 2 m), [2 m, 4 m),…, [48 m, 50 m), [50 m, $+\infty$). Range rate rr has 13 possible intervals with 2 m/s per interval: ($-\infty$, -18 m/s), [-18 m/s, -16 m/s),…, [2 m/s, 4 m/s), [4 m/s, $+\infty$). Speed v has 11 possible intervals with 2 m/s per interval: [0, 2 m/s), [2 m/s, 4 m/s),…, [18 m/s, 20 m/s), [20 m/s, $+\infty$). The upper and lower boundaries of each parameter can be obtained from the historical trajectory data. Each interval corresponds to a letter alphabetically. For example, the driving behavior ($r = 1$ m/s; $rr = -17$ m/s; $v = 5$ m/s) corresponds to the word “*abc*”. With this mapping function, any trajectory data point can be mapped into a word. In this study, the total number of unique words (i.e., vocabulary) is $26 \times 13 \times 11 = 3{,}718$. With a different number of features, boundaries, and discretization resolutions, the vocabulary can be constructed based on different datasets and study objectives.

3.2. *Embedding model building*

The trajectory embedding model is adapted from the word2vec model (one input word and one output word) with the negative sampling approach. For a detailed explanation of the word2vec model, please refer to (Goldberg and Levy, 2014; Rong, 2016). The embedding model generates vector representations of words based on the distribution of word co-occurrences in a training dataset. As shown in Fig. 3, the embedding model is a neural network with three layers: an input layer, an output layer, and a hidden layer. The model has an internal prediction task: predicting a target word given a context word. The model parameters are updated through training. However, the prediction task itself is not the goal of this model. The goal is to learn the weights of the hidden layer. Eventually, each word will have its unique weights in the hidden layer and these weights are the vector representation of the word, which can be considered as features that describe this word.

Denote V as the vocabulary and M as the number of unique words (i.e., the size of the vocabulary). The input layer is a one-hot encoded vector with dimension M. Given a context word $w_i \in V$ (here i indicates its location in the vocabulary), the input to the embedding model is $X_{w_i} = [0, 0, \cdots 0, 1, 0, \cdots, 0]^T$, where only the i^{th} element is 1 and all other elements are 0. The weight between the input layer and the hidden layer is a $M \times N$ matrix denoted as W. Row i of W is denoted as $v_{w_i}^T$ with dimension N, which is the input

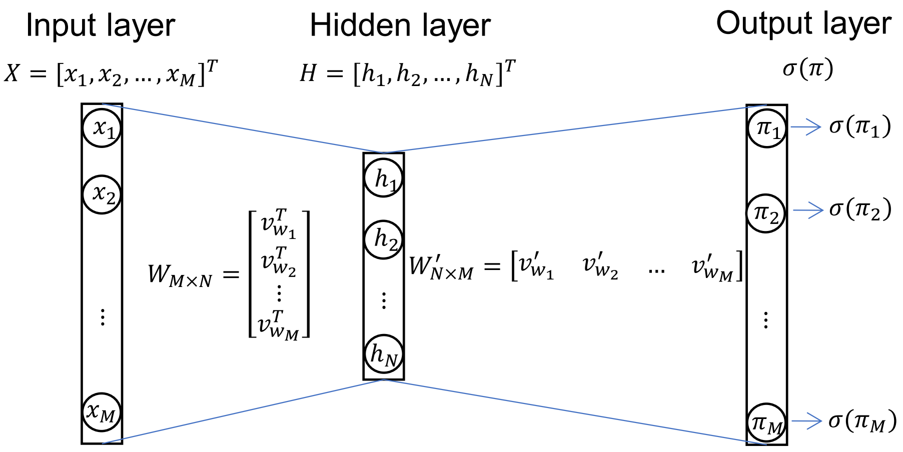

Input layer
$X = [x_1, x_2, \ldots, x_M]^T$
Hidden layer
$H = [h_1, h_2, \ldots, h_N]^T$
Output layer
$\sigma(\pi)$
x_1
x_2
x_M
$W_{M\times N} = \begin{bmatrix} v_{w_1}^T \\ v_{w_2}^T \\ \vdots \\ v_{w_M}^T \end{bmatrix}$
h_1
h_2
h_N
$W'_{N\times M} = [v'_{w_1} \quad v'_{w_2} \quad \ldots \quad v'_{w_M}]$
π_1
π_2
π_M
$\sigma(\pi_1)$
$\sigma(\pi_2)$
$\sigma(\pi_M)$

Fig. 3. Illustration of the Embedding Model.

vector of the input word w_i. The dimension of the hidden layer is N, which is a predefined value (N is significantly smaller than M). In this study, $N = 20$. Given a word $w_i \in V$, the hidden layer can be computed as $H = W^T X_{w_i} = v_{w_i}$. The weight between the hidden layer and output layer is a $N \times M$ matrix denoted as W' (W' is different from W). Column j of W' is denoted as v'_{w_j} with dimension N, which is the output vector for the output word w_j. The dimension of the output layer is M. For each output word w_j in the vocabulary, the score π_j is computed as $\pi_j = {v'_{w_j}}^T H = {v'_{w_j}}^T v_{w_i}$. The probability of predicting word w_j given the input word w_i is expressed by a logistic function $\sigma(\pi_j) = \sigma\left({v'_{w_j}}^T v_{w_i}\right) = \frac{1}{1+\exp(-{v'_{w_j}}^T v_{w_i})}$.

Denote w_{i^*} as the target word of w_i, i.e., the positive sample and $D(w_i)$ as the set of negative samples. $D(w_i)$ contains K “wrong” words (i.e., any word other than the target word) that are randomly drawn from the vocabulary V (thus the name negative sampling). In this study, K is set to 5, the recommended value for a small training dataset (Mikolov et al., 2013b). The loss function is defined as:

$$E = -log\sigma\left({v'_{w^*}}^T v_{w_i}\right) - \sum_{w_j \in D(w_i)} log\sigma\left(-{v'_{w_j}}^T v_{w_i}\right) \tag{1}$$

Take the derivative of E with respect to v'_{w_j}:

$$\frac{\partial E}{\partial v'_{w_j}} = \left[\sigma\left({v'_{w_j}}^T v_{w_i}\right) - t(w_j)\right] v_{w_i} \tag{2}$$

where

$$t(w_j) = \begin{cases} 1, w_j = w_{i^*} \\ 0, w_j \in D(w_i) \end{cases} \tag{3}$$

Therefore, using stochastic gradient descent, the update rule for the weight v'_{w_j} in matrix W' is:

$${v'_{w_j}}^{k+1} = {v'_{w_j}}^{k} - \alpha \frac{\partial E}{\partial v'_{w_j}} \tag{4}$$

where $\alpha > 0$ is the learning rate and k is the iteration number.
Similarly, take the derivative of E with respect to v_{w_i}:

$$\frac{\partial E}{\partial v_{w_i}} = \sum_{w_j \in \{w_{i^*}\} \cup D(w_i)} \left[\sigma\left({v'_{w_j}}^T v_{w_i}\right) - t(w_j)\right] v'_{w_j} \tag{5}$$

The corresponding update rule for the weight v_{w_i} in matrix W is

$$v_{w_i} k+1 = {v_{w_i}}^{k} - \alpha \frac{\partial E}{\partial v_{w_i}} \tag{6}$$

For each input word w_i, Equation (4) is applied to $w_j \in \{w_{i^*}\} \cup D(w_i)$ and updates the corresponding v'_{w_j} in matrix W' and Equation (6) is applied to update v_{w_i} in matrix W. In the proposed method, $\sigma\left({v'_{w_j}}^T v_{w_i}\right)$ is the probability of the correct prediction and $t(w_j)$ is the expected output (1 if w_j is a positive sample, and 0 if w_j is a negative sample). If $\sigma\left({v'_{w_j}}^T v_{w_i}\right) \rangle t(w_j)$, it means that the word w_j is not the correct context word for w_i. Equation (4) subtracts a proportion of v_{w_i} from v'_{w_j}, and moves v'_{w_j} further away from v_{w_i}. The step size of the

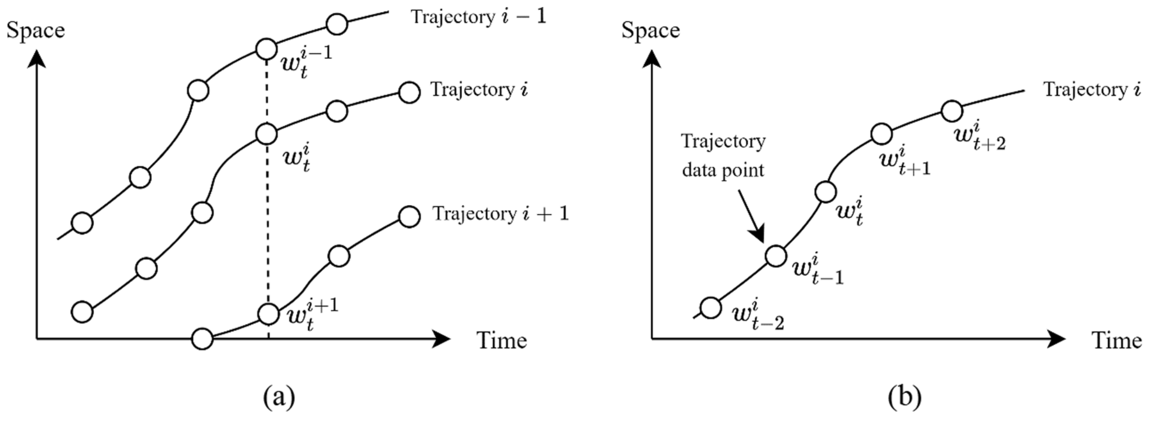


Fig. 4. Construction of Positive Samples in Training Data (a) Multiple trajectories (spatial dimension) (b) Single trajectory (temporal dimension).

movement is determined by the prediction error $\sigma\left(v'^{T}_{w_j} v_{w_i}\right) - t(w_j)$. Similarly, Equation (6) subtracts a proportion of v'_{w_j} from v_{w_i}, making v_{w_i} further away from v'_{w_j}. If $\sigma\left(v'^{T}_{w_j} v_{w_i}\right) \langle t(w_j)$, the equations move v_{w_i} and v'_{w_j} to be closer. For each iteration, one goes through all the context words from a training dataset and updates the weight matrix W and W'. After the training is completed, given an arbitrary word $w_i \in V$, its vector representation is the vector of its hidden layer H (i.e., v_{w_i}).

3.3. Data for model training

The trajectory embedding model is trained with positive and negative samples. As discussed in the previous section, for each positive sample, K negative samples are randomly chosen from the vocabulary. In this study, there are three types of positive samples.

- Between-trajectory (spatial dimension): As illustrated in Fig. 4**(a)**, at time t, a snapshot is taken to capture all CV trajectories. The trajectory data points of all CVs are converted into words. For trajectory i, its trajectory data point is denoted as w_t^i. w_t^i is the positive sample of its leading vehicle's word w_t^{i-1} and its following vehicle's word w_t^{i+1}. This type of positive samples ensures that adjacent trajectory data points in the spatial dimension have similar vector representations.
- Single-trajectory (temporal dimension): As illustrated in Fig. 4**(b)**, in a complete trajectory i, each trajectory data point is converted into a word. Given a word w_t^i, its immediate next word w_{t+1}^i is considered as a positive sample. This type of positive samples ensures that adjacent trajectory data points in the temporal dimension have similar vector representations.
- Related words: Related words are defined as the words that only differ from one interval in the mapping function. For example, "*aaa*" and "*aab*" are related words and are positive samples for each other. This type of positive samples ensure that similar driving behaviors have similar vector representations. Moreover, this type of positive samples would enumerate all possible words in the vocabulary and therefore can prevent out-of-vocabulary words (i.e., driving behavior that does not appear in a training data set but appears in a testing data set).

4. Falsified trajectory identification

The trained trajectory embedding model provides vector representations of the trajectory data. Therefore, the similarity (distance) between any pair of trajectories can be computed using the vector representations. Then a clustering algorithm is applied to group similar trajectories and identify anomalous ones.

Fig. 5 shows two trajectories in a time–space diagram. Each trajectory is composed of multiple trajectory data points. t_s and t_e define the time window that both trajectories are received by the infrastructure. At any time t within this time window, the distance between two trajectory points is calculated using the Euclidean distance of the two vector representations (i.e., $\|H_t^i - H_t^j\|_2$). Then, the similarity between the two trajectories is defined as the average distance over the time window, as shown in Equation (7).

$$d(i,j) = \frac{1}{10(t_e - t_s) + 1} \sum_{t=t_s}^{t_e} \|H_t^i - H_t^j\|_2 \tag{7}$$

where H_t^i and H_t^j are the vector representations for trajectory i and j at time t. $10(t_e - t_s) + 1$ denotes the number of data points within the time window.

For a total number of N observed trajectories, an N by N similarity matrix can be constructed by computing the similarity between each pair of the trajectories.

With the similarity matrix, hierarchical clustering is used to classify the trajectories. Hierarchical clustering is an unsupervised learning method in machine learning (Rokach and Maimon, 2005). It groups similar objects into clusters such that objects in the same cluster are similar to each other than the objects in other clusters. In this study, a bottom-up approach in hierarchical clustering is

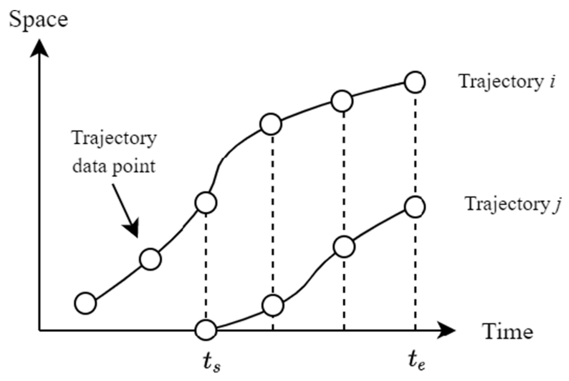


Fig. 5. Illustration of Computing the Similarity Between Two Trajectories.

adopted. Each trajectory is treated as one cluster at the beginning. The similarity between clusters c_p and c_q is calculated as the average distance between all pairs of trajectories in the two clusters, i.e., $D(c_p, c_q) = \frac{1}{|c_p||c_q|}\sum_{i\in c_p, j\in c_q} d(i,j)$. For each iteration, the most similar clusters are merged together to form one single cluster. The merging process stops when the clusters are not close to each other (i.e., the similarity exceeds a predefined threshold ϵ) or all trajectories have been merged into the same cluster. Finally, the trajectories in the largest cluster are marked as normal; otherwise anomalous. The pseudo-code of the falsified trajectory identification algorithm using hierarchical clustering is shown below.

Algorithm 1. (*Falsified Trajectory Identification*)

```
Initialization: Assign each trajectory into a cluster (i.e., $|C| = N$, where $C$ is the set of clusters)
    Compute $D_{min} = \min_{c_p, c_q \in C} D(c_p, c_q)$;
While $D_{min} \le \epsilon \& |C| > 1$ do
  Merge cluster $c_p^*$ and $c_q^*$ into one cluster, where $D(c_p^*, c_q^*) = D_{min}$;
  If $|C| > 1$ then
    Compute $D_{min} = \min_{c_p, c_q \in C} D(c_p, c_q)$
  Else
    break while;
  End if
End While
Output: The trajectories in the largest cluster are marked as normal; otherwise anomalous
```

5. Numerical experiments

In this section, numerical experiments are provided to demonstrate the effectiveness of the proposed trajectory embedding method using microscopic simulation. Before presenting the experiment results, we will first introduce how the falsified trajectories are generated. Then we will show examples of training results as well as the identification results with sensitivity analysis and discussions.

5.1. Falsified trajectory generation

The falsified trajectory generation is formulated as an optimization problem **P1**.

$$\min_{a(t)} \sum_{t=t_s}^{t_e} \left(d_l(t-\tau^w) - d^w - d(t)\right)^2 \tag{8}$$

s.t.

$$g(d(t), v(t), a(t)) = 0 \tag{9}$$

$$v(t) \times \Delta t = d(t) - d(t-\Delta t); a(t) \times \Delta t = v(t) - v(t-\Delta t) \tag{10}$$

$$0 \le v(t) \le v_f; a_{min} \le a(t) \le a_{max} \tag{11}$$

$$d_f(t) + d_0 \le d(t) \le d_l(t) - d_0 \tag{12}$$

$$v(t_s) = v_s; d(t_s) = d_s \tag{13}$$

A falsified trajectory can be represented by its location $d(t)$, speed $v(t)$, and acceleration rate $a(t)$ at time t. The objective of the optimization problem is to minimize the cumulative square error in car-following behavior. In this case, Newell's first-order car following model (Newell, 2002) is selected where τ^w and d^w are time and distance displacement in the model. $d_l(t)$ are the locations of the leading vehicle. t_s and t_e are the start and end time of the falsified trajectory generation. The objective function tries to mimic a normal car-following behavior to reduce the probability of being detected. Newell's model is chosen as an example. In reality, the attacker could use the received BSM data to calibrate a human-like car-following model as the objective function. Equation (9) represents a general attack goal. The function $g(\cdot)$ could take different forms with different attack goals, which would be introduced next. Equations (10) and (11) represent the vehicle dynamics and boundaries of speed v and acceleration a. Equation (12) guarantees that the falsified trajectory keeps a safe distance d_0 from the leading vehicle (i.e., $d_l(t)$) and following vehicle (i.e., $d_f(t)$) at any time t. Finally, Equation (13) is the initial condition where the falsified trajectory enters the intersection area from a certain distance (e.g., communication range) with a certain speed (e.g., free-flow speed).

The reason for generating the falsified trajectory as the solution to **P1** is to increase the difficulty of the defense model. The optimization problem tries to generate a trajectory as realistic as possible, which makes the identification problem more challenging. As mentioned before, attackers may have different objectives, which are reflected in the attack goal (i.e., Equation (9)). In this study, we adopt results from our previous studies (Chen et al., 2018; Huang et al., 2021), which investigated the vulnerability of a CV-based traffic signal control system (i.e., I-SIG) (Feng et al., 2015). Two types of cyber attacks are found: ETA (estimated time of arrival) attack and phantom queue attack. The ETA attack generates a fake slow-moving vehicle with a predefined ETA at a certain time point (e.g., beginning of the green interval) while the phantom queue attack generates a fake stopped vehicle at a certain location and certain time point. Both attacks are able to influence the signal optimization model to generate sub-optimal signal timing plans and effectively increase the total delay of the intersection. With the two attack goals, Equation (9) can be formulated as Equations (14) and (15) for ETA attack and phantom queue attack respectively.

$$v(t_e) > v_{stop}; \frac{(d_s - d(t_e))}{v(t_e)} = ETA \tag{14}$$

$$v(t_e) = 0; (d_s - d(t_e)) \times k_j = Queue \tag{15}$$

where v_{stop} is a speed threshold (2.24 m/s), below which the vehicle is considered as a stopped vehicle; d_s is the location of the stop bar; and k_j is the jam density. The value of *ETA* and *Queue* can be determined by the maximum green time and saturation flow rate (Chen et al., 2018). Note that in this study, the end time t_e is set to be the beginning of the green interval, as shown in Fig. 1. This setting is also adopted from our previous study because the particular signal control system under study executes the signal optimization algorithm at this time point. As a result, the attacker has no incentive to keep generating falsified trajectories after t_e because the new data points will not be utilized to influence the signal timing, but increase the chances of been detected. However, for other applications, the attack model can be adapted to generate falsified trajectories with any duration.

5.2. Experiment setup

VISSIM is used as the simulator to generate “normal” trajectories. In order to include uncertainties in driving behaviors, the simulation uses stochastic values for safety distance, desired acceleration, and desired deceleration. The free-flow speed of each vehicle follows a uniform distribution, with a lower bound of 13.89 m/s (50 km/h) and an upper bound of 19.44 m/s (70 km/h). The traffic signal implements a fixed-time plan, with 60 s of green time and 90 s of red time for every cycle. The traffic volume is set to 500 vehicles per hour per lane. The time displacement and space displacement in **P1** are set to 1.5 s and 6.2 m. The safety distance is 6.2 m. The free-flow speed is 16.67 m/s (60 km/h). The minimum and maximum acceleration rates are -2 m/s^2 and 2 m/s^2. The initial speed of the falsified trajectory is the same as the free-flow speed and the initial position is at the boundary of the approach (i.e., 300 m).

The first simulation lasts for 10 simulation hours. 300 signal cycles of vehicle trajectory data are recorded. The simulation resolution is 10 Hz, where the trajectory data is recorded every 0.1 s. These 10 h of trajectories are used as training data for the trajectory embedding model. Then nine case studies are conducted and each test also lasts for 10 simulation hours. The nine case studies consist of three types of attack goals (ETA attack, phantom queue attack, and no-goal attack) and three CV penetration rates (100%, 50%, and 25%). In the case studies of no-goal attack, Equation (9) is simply removed from **P1**, which means that the attacker tries to generate a falsified trajectory with normal behaviors. In reality, the attacker may not have the incentives to generate such falsified trajectories. These case studies are added just for comparison purposes. It is interesting to see whether the proposed model can still identify falsified trajectories even they appear to be normal. In the case studies, for each signal cycle, one falsified trajectory is inserted by solving **P1**.

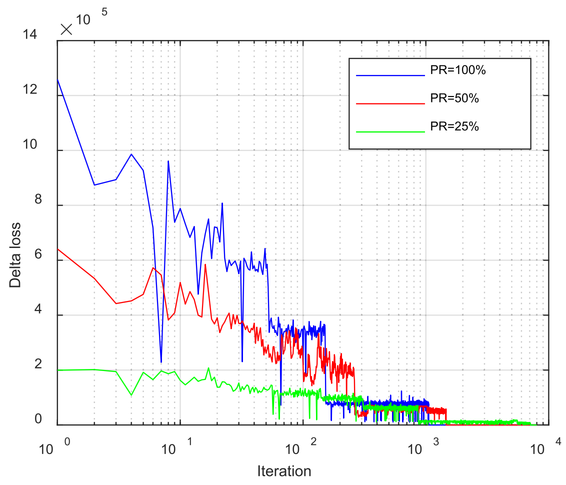


Fig. 6. Training Results of the Trajectory Embedding Model.

5.3. Trajectory embedding model training

Gensim (Rehurek, 2010), a Python package for word embedding, is used for training the trajectory embedding model. Three trajectory embedding models are trained based on different CV penetration rates. The delta loss (the difference of loss between two iterations) is shown in Fig. 6. The training converges at 1056, 1463, and 7060 iterations when the CV penetration rate is 100%, 50%, and 25% respectively. It is expected that when the penetration rate is low, it takes a longer time to converge because the range and range rate values are more diverse.

Table 1 shows a few examples of similar words to the word *aka* from the trajectory embedding model under 100% CV penetration rate. Since a word is represented by a vector, the Euclidean distance between two vectors is utilized to quantify the similarity. In this example, the maximum Euclidean distance is 17.44 and the median is 12.20. Therefore, all three example words (*aja, zkh,* and *bja*) are considered similar to *aka*. The words *bja*, *aja*, and *aka* represent queueing state. In the training data set, we could constantly observe *bja*, *aja*, and *aka* in one vehicle's trajectory trace. These words are positive samples in the single-trajectory data category. They reflect the relations between consecutive trajectory data points for a single vehicle over time (temporal dimension). Therefore, the trajectory embedding model considers the words *aja* and *bja* being similar to *aka*. It is counterintuitive that the word *zkh* is similar to *aka*, because *zkh* represents the free-flow state. However, these two words (w1 and w3 in Fig. 7) are both positive samples of w2 in the between-trajectory data category, which represents the spatial relation. Therefore, they are similar. Actually, the fact that *aka* has a small Euclidean distance to *zkh* is one of the reasons why the model performs well. The training dataset contains frequent positive samples of w1, w2, and w3 in the same timestamp, which represents a typical traffic scenario: transition from queuing state to free flow state. The proposed method considers them as a normal driving behavior, which essentially improves the detection rate and reduces the false alarm rate.

5.4. Identification results

Examples of falsified trajectories and corresponding clustering results are shown in Fig. 8. In Fig. 8 (a), a falsified trajectory is generated with the attack goal ETA = 60 s under 100% CV penetration rate. The falsified vehicle follows the leading vehicle based on Newell's car-following model at first and slows down to achieve the attack goal. Fig. 8(b) shows the clustering result. For clarity, the falsified trajectory is always labeled as 1 in all cases. The threshold in the clustering algorithm ϵ is set 6, which is highlighted by a red dashed line. Fig. 8(b) shows that normal trajectories (trajectory 2 to 19) form one cluster, while the falsified trajectory forms the second cluster. Because of the unusual behavior (slowing down when the front vehicle is still far), the distance between the falsified trajectory and other trajectories is significant. Therefore, the clustering method is able to identify the falsified trajectory. Similarly, in Fig. 8(c), a falsified vehicle is generated with the attack goal Queue = 30 vehicles under 50% CV penetration rate. Again, the falsified vehicle follows the leading vehicle based on Newell's car-following model at first and stops far from the intersection at the beginning of the green interval. This unusual behavior is successfully detected by the proposed method, as shown in Fig. 8(d). Interestingly, in Fig. 8(e) when a falsified trajectory is generated without any attack goal, it can still be identified as anomalous as shown in Fig. 8(f). This is because the car-following model for generating the falsified trajectory (Newell's car-following model) is different from the car-following behavior of VISSIM's internal driving model (Wiedemann's model). The difference becomes more prominent when the falsified vehicle slows down at the end. However, the small differences in car-following behaviors may not be captured every time. As a result, the detection rate in this scenario is significantly reduced, as shown in Table 3.

Detection rate and false alarm rate are used to measure the effectiveness of the proposed method. The two measurements can be calculated using the coincidence matrix as shown in Table 2. They are calculated as

$$\text{Detection Rate} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$\text{False Alarm Rate} = \frac{\text{FP}}{\text{FP} + \text{TN}}$$

Table 3 shows a summary of the results for the proposed method. Each case includes 300 signal cycles (i.e., 300 chances to generate falsified trajectories). However, if there is no feasible solution to **P1**, then no falsified trajectory is generated in this cycle. As shown in the table, when there is a specific attack goal, the proposed method can identify the majority of the falsified trajectories with a detection rate greater than 97% in all cases. As shown in Fig. 8, such falsified trajectories usually have anomalous behaviors such as slowing down when the range is still large or stopping far away from the intersection. Such anomalous driving behaviors are rarely observed in the training data set. Therefore, their vector representations are highly different from normal behaviors. When there is no

Table 1
Similar Words Identified by the Trajectory Embedding Model.

Euclidean Distance	Range (m)	Range Rate (m/s)	Speed (m/s)	Word
–	[0,2)	[0,2)	[0,2)	*aka*
0.86	[0,2)	[-2,0)	[0,2)	*aja*
1.83	[50,+∞)	[0,2)	[14,16)	*zkh*
2.33	[2,4)	[-2,0)	[0,2)	*bja*

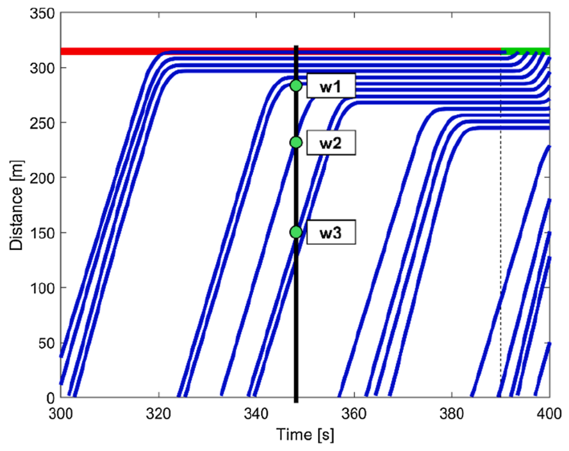


Fig. 7. Illustration of Similar Words in the Training Data Set.

attack goal, an attacker generates falsified trajectories that simply follow a certain car-following model. Such falsified trajectories seem reasonable because they satisfy vehicle dynamics and car-following relations. Therefore, the detection rate is reduced. This confirms that the attack goal is indeed the reason why the falsified trajectories are anomalous and also validates the proposed misbehavior detection method. The proposed method is still effective when the penetration rate is not 100%. The detector rate remains high while a slightly increased false alarm rate is observed. The increased false alarm rate is due to more uncertainties in observed CV trajectories. However, the false alarm rate is less concerned than the detection rate. A 8% false alarm rate in the 25% penetration rate case means a total of 2% (25%*0.08) normal trajectories are identified as falsified and will be discarded. The actual penetration rate is slightly dropped to 23%, which is expected to have a small impact on the performance of the V2I application under protection. Overall, the results indicate that the proposed identification model is robust and performs well under varying penetration rates.

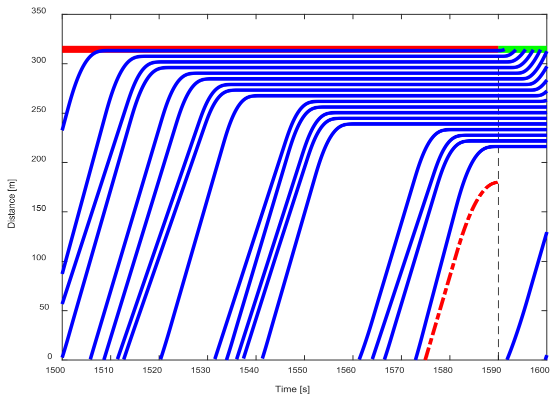
Note that when the penetration rates are 50% and 25%, the proposed method is able to identify considerable falsified trajectories even if there is no attack goal (48.8% and 61.2%). There are two reasons to explain this phenomenon. First, as mentioned previously, the car-following model of the falsified trajectories is different from the normal trajectories. Second, when the penetration rate is lower, the probability of observing a CV closely following another CV decreases, which becomes a rare event in the training dataset (thus anomalous).

5.5. *Sensitivity analysis*

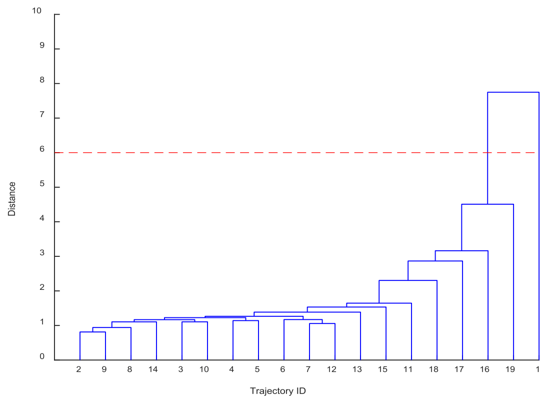
One important hyperparameter of the proposed model is the threshold $\in$ of the clustering algorithm, which has an impact on both detection rate and false alarm rate. A sensitivity analysis is conducted to reveal its impact, as shown in Fig. 9. The case study with the ETA attack under 100% CV penetration rate is considered in the sensitivity analysis. A larger threshold indicates more tolerance for anomalous behaviors. Therefore, the proposed method would fail to identify falsified trajectories. When the threshold is 9, the detection rate drops to below 20%. The false alarm rate drops to near zero. A smaller threshold permits less anomalous behaviors. When the threshold is 5, the detection rate increases to 100%, and the false alarm rate also increases to 3.6%. Both the detection rate and false alarm rate decrease as the threshold increases. The results imply that there is a trade-off between the detection rate and false alarm rate. A cost function can be formulated to evaluate the cost of cyber-attacks and the system cost with less available data. This interesting problem will be left for further study.

6. Conclusions

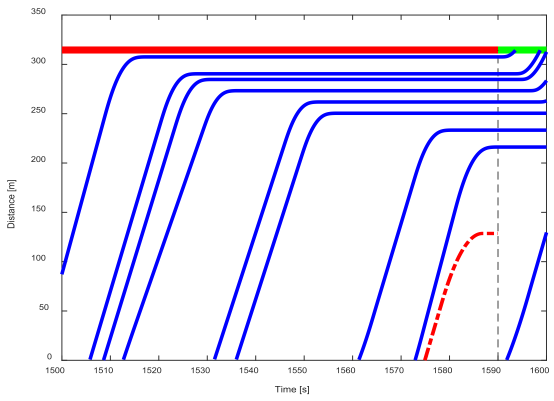
To protect the infrastructure-based V2I applications from falsified data attacks, this paper proposed a data-driven method to identify falsified vehicle trajectories. It is assumed that falsified trajectories need to achieve a certain attack goal. As a result, they are behaviorally distinct from normal trajectories. The problem of identifying falsified trajectories is hence considered a misbehavior detection problem. A trajectory embedding model was developed to generate vector representations of trajectory data points. The similarity (distance) between trajectories was computed based on vector representations. Hierarchical clustering was applied to identify anomalous (i.e., falsified) trajectories. A series of case studies were then conducted to evaluate the effectiveness of the proposed method. In the case studies, falsified trajectories were generated every signal cycle under different CV penetration rates (i.e., 100%, 50%, and 25%) with different attack goals (i.e., ETA attack and phantom queue attack). Findings indicated that the proposed method could successfully identify the majority (over 97% in general) of falsified trajectories while maintaining low false alarm rates. One future research direction is to apply attention mechanisms (Bahdanau et al., 2016) to improve the model. Another interesting direction is to investigate the impact of the false alarm rate on different CV applications and how to further reduce it.



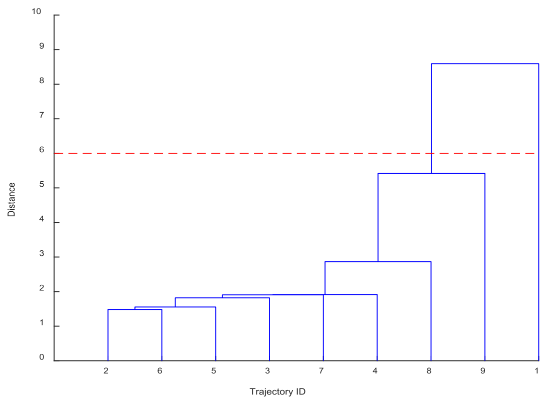
(a)



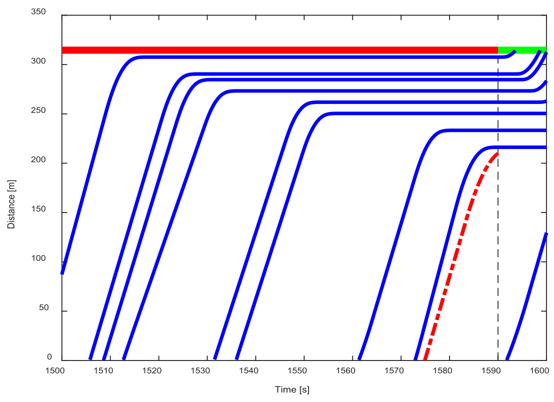
(b)



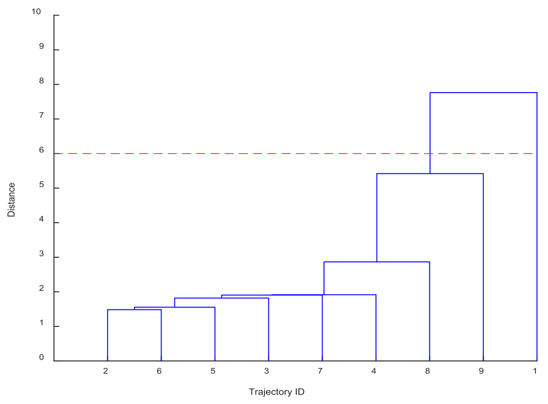
(c)



(d)



(e)



(f)

Fig. 8. Falsified Trajectory Generation and Identification Results: (a) example of a falsified trajectory with ETA attack (100% CV penetration rate); (b) hierarchical clustering results of (a); (c) example of a falsified trajectory with phantom queue attack (50% CV penetration rate); (d) hierarchical clustering results of (c); (e) example of a falsified trajectory with no-goal attack (50% CV penetration rate); (f) hierarchical clustering results of (e);

Table 2
A simple coincidence matrix.

Predicted Class	True Class	
	Positive	Negative
Positive	True Positive Count (TP)	False Positive Count (FP)
Negative	False Negative Count (FN)	True Negative Count (TN)

Table 3
Results Summary of Falsified Trajectory Identification.

Penetration Rate	Attack Goal	TP	TN	FP	FN	Detection Rate	False Alarm Rate
100%	ETA	183	4445	43	5	97.3%	1.0%
	Phantom Queue	129	4457	31	0	100.0%	0.7%
	No Goal	24	4472	16	240	9.1%	0.4%
50%	ETA	242	2115	79	1	99.6%	3.6%
	Phantom Queue	205	2125	69	0	100.0%	3.1%
	No Goal	144	2114	80	151	48.8%	3.6%
25%	ETA	229	1002	90	2	99.1%	8.2%
	Phantom Queue	211	998	88	1	99.5%	8.1%
	No Goal	161	1003	93	102	61.2%	8.5%

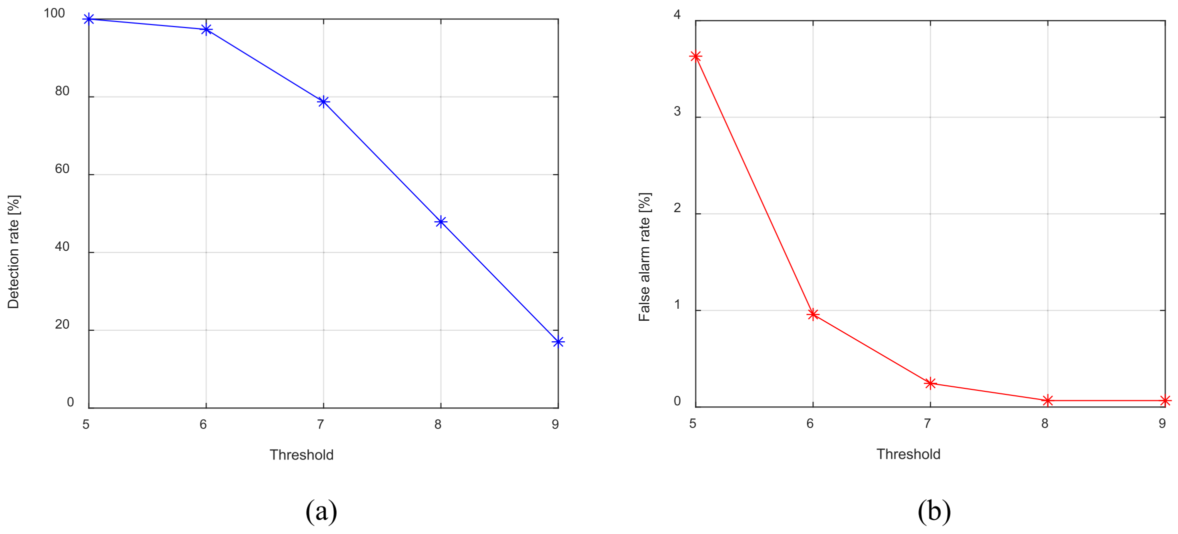


Fig. 9. Sensitivity Analysis on the Identification Threshold: (a) Detection rate; and (b) False alarm rate.

CRediT authorship contribution statement

Shihong Ed Huang: Methodology, Formal analysis, Validation, Investigation, Data curation, Writing - original draft. **Yiheng Feng:** Conceptualization, Methodology, Investigation, Data curation, Funding acquisition, Writing - review & editing. **Henry X. Liu:** Conceptualization, Funding acquisition, Supervision, 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

This research is supported in part by the U.S. National Science Foundation through Grant SaTC #1930041 and Mcity at the University of Michigan for financial support. The views presented in this paper are those of the authors alone.

References

Alipour-Fanid, A., Dabaghchian, M., Zhang, H., Zeng, K., 2017. String stability analysis of cooperative adaptive cruise control under jamming attacks. Presented at the 2017 IEEE 18th International Symposium on High Assurance Systems Engineering (HASE).

Amoozadeh, M., Raghuramu, A., Chuah, C., Ghosal, D., Zhang, H.M., Rowe, J., Levitt, K., 2015. Security vulnerabilities of connected vehicle streams and their impact on cooperative driving. IEEE Commun. Mag. 53 (6), 126–132. https://doi.org/10.1109/MCOM.2015.7120028.

Bahdanau, D., Cho, K., Bengio, Y., 2016. Neural Machine Translation by Jointly Learning to Align and Translate. arXiv:1409.0473 [cs, stat].

Bißmeyer, N., Stresing, C., Bayarou, K.M., 2010. Intrusion detection in VANETs through verification of vehicle movement data. In: 2010 IEEE Vehicular Networking Conference. https://doi.org/10.1109/VNC.2010.5698232.

Brecht, B., Therriault, D., Weimerskirch, A., Whyte, W., Kumar, V., Hehn, T., Goudy, R., 2018. A security credential management system for V2X communications. IEEE Trans. Intell. Transp. Syst. 19 (12), 3850–3871. https://doi.org/10.1109/TITS.2018.2797529.

Canepa, E.S., Claudel, C.G., 2013. A framework for privacy and security analysis of probe-based traffic information systems.

Canepa, E.S., Claudel, C.G., 2013. Spoofing cyber attack detection in probe-based traffic monitoring systems using mixed integer linear programming. Presented at the 2013 International Conference on Computing, Networking and Communications (ICNC).

Chen, Q.A., Yin, Y., Feng, Y., Mao, Z.M., Liu, H.X., 2018. Exposing data spoofing attacks on CV-based traffic signal control. Presented at the 25th Network and Distributed System Security Symposium (NDSS).

Chen, C., Zhang, D., Castro, P.S., Li, N., Sun, L., Li, S., Wang, Z., 2013. IBOAT: isolation-based online anomalous trajectory detection. IEEE Trans. Intell. Transp. Syst. 14 (2), 806–818. https://doi.org/10.1109/TITS.2013.2238531.

Cheng, Y., Wu, B., Song, L., Shi, C., 2019. Spatial-Temporal Recurrent Neural Network for Anomalous Trajectories Detection. Cham.

Cui, L., Hu, J., Park, B.B., Bujanovic, P., 2018. Development of a simulation platform for safety impact analysis considering vehicle dynamics, sensor errors, and communication latencies: assessing cooperative adaptive cruise control under cyber attack. Transp. Res. Part C: Emerg. Technol. 97, 1–22. https://doi.org/10.1016/j.trc.2018.10.005.

Dadras, S., Gerdes, R.M., Sharma, R., 2015. Vehicular Platooning in an Adversarial Environment. New York, NY, USA.

Feng, Y., Head, K.L., Khoshmagham, S., Zamanipour, M., 2015. A real-time adaptive signal control in a connected vehicle environment. Transp. Res. Part C: Emerg. Technol. 55, 460–473. https://doi.org/10.1016/j.trc.2015.01.007.

Goldberg, Y., Levy, O., 2014. Word2vec explained: deriving Mikolov et al.'s negative-sampling word-embedding method. arXiv:1402.3722 [cs, stat].

Feng, Y., Huang, S., Chen, Q.A., Liu, H.X., Mao, Z.M., 2018. Vulnerability of traffic control system under cyberattacks with falsified data. Transport. Res. Rec. 0361198118756885.

Huang, S., 2020. Cyber Security of Traffic Signal Control Systems with Connected Vehicles. University of Michigan.

Huang, S.E., Feng, Y., Wong, W., Chen, Q.A., Mao, Z.M., Liu, H.X., 2021. Impact evaluation of falsified data attacks on connected vehicle based traffic signal control systems. In: Workshop on Automotive and Autonomous Vehicle Security (AutoSec), No. 2021, pp. 25.

Jeske, T., 2013. Floating car data from smartphones: what google and waze know about you and how hackers can control traffic. In: Presented at the BlackHat Europe.

Koscher, K., Czeskis, A., Roesner, F., Patel, S., Kohno, T., Checkoway, S., McCoy, D., Kantor, B., Anderson, D., Shacham, H., Savage, S., 2010. Experimental security analysis of a modern automobile. In: Presented at the 2010 IEEE Symposium on Security and Privacy.

Li, X., Zhao, K., Cong, G., Jensen, C.S., Wei, W., 2018. Deep representation learning for trajectory similarity computation. Presented at the 2018 IEEE 34th International Conference on Data Engineering (ICDE).

Lv, Z., Xu, J., Zhao, P., Liu, G., Zhao, L., Zhou, X., 2017. Outlier Trajectory Detection: A Trajectory Analytics Based Approach. Cham.

Mazloom, S., Rezaeirad, M., Hunter, A., McCoy, D., 2016. A security analysis of an in vehicle infotainment and app platform. Berkeley, CA, USA.

Mikolov, T., Chen, K., Corrado, G., Dean, J., 2013a. Efficient Estimation of Word Representations in Vector Space. arXiv:1301.3781 [cs].

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., Dean, J., 2013b. Distributed Representations of Words and Phrases and Their Compositionality. Lake Tahoe, Nevada.

Mikolov, T., Yih, W., Zweig, G., 2013c. Linguistic regularities in continuous space word representations. In: Presented at the NAACL-HLT 2013, Atlanta, Georgia.

Newell, G.F., 2002. A simplified car-following theory: a lower order model. Transp. Res. Part B: Methodol. 36 (3), 195–205. https://doi.org/10.1016/S0191-2615(00)00044-8.

Nguyen, V., Lin, P., Hwang, R., 2019. Physical signal-driven fusion for V2X misbehavior detection. Presented at the 2019 IEEE Vehicular Networking Conference (VNC).

Oh, M., Iyengar, G., 2019. Sequential anomaly detection using inverse reinforcement learning. In: Presented at the KDD '19: The 25th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Anchorage AK USA.

Rehurek, R., Sojka, P., 2010. Software Framework for Topic Modelling with Large Corpora.

Rong, X., 2016. Word2vec Parameter Learning Explained. arXiv:1411.2738 [cs].

Rokach, L., Maimon, O., 2005. Clustering methods. In: Maimon, O., Rokach, L. (Eds.), Data Mining and Knowledge Discovery Handbook. Springer US, Boston, MA, pp. 321–352.

Ruj, S., Cavenaghi, M.A., Huang, Z., Nayak, A., Stojmenovic, I., 2011. On data-centric misbehavior detection in VANETs. Presented at the 2011 IEEE Vehicular Technology Conference (VTC Fall).

Shoukry, Y., Mishra, S., Luo, Z., Diggavi, S., 2018. Sybil attack resilient traffic networks: a physics-based trust propagation approach. Presented at the 2018 ACM/IEEE 9th International Conference on Cyber-Physical Systems (ICCPS).

Sinai, M.B., Partush, N., Yadid, S., Yahav, E., 2014. Exploiting Social Navigation. arXiv:1410.0151 [cs].

Singh, P.K., Tabjul, G.S., Imran, M., Nandi, S.K., Nandi, S., 2018. Impact of security attacks on cooperative driving use case: CACC platooning. Presented at the TENCON 2018–2018 IEEE Region 10 Conference.

Smolyak, D., Gray, K., Badirli, S., Mohler, G., 2020. Coupled IGMM-GANs with applications to anomaly detection in human mobility data. ACM Trans. Spatial Algorithms Syst. 6 (4) https://doi.org/10.1145/3385809, p. 24:1–24:14.

So, S., Petit, J., Starobinski, D., 2019. Physical Layer Plausibility Checks for Misbehavior Detection in V2X Networks. New York, NY, USA.

So, S., Sharma, P., Petit, J., 2018. Integrating plausibility checks and machine learning for misbehavior detection in VANET. Presented at the 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA).

Song, L., Wang, R., Xiao, D., Han, X., Cai, Y., Shi, C., 2018. Anomalous Trajectory Detection Using Recurrent Neural Network. Cham.

Sun, M., Li, M., Gerdes, R., 2017. A data trust framework for VANETs enabling false data detection and secure vehicle tracking. Presented at the 2017 IEEE Conference on Communications and Network Security (CNS).

Wang, H., Feng, J., Sun, L., An, K., Liu, G., Wen, X., Hu, R., Chai, H., 2020. Abnormal trajectory detection based on geospatial consistent modeling. IEEE Access 8, 184633–184643. https://doi.org/10.1109/ACCESS.2020.3028847.

Wang, X., Shen, S., Bezzina, D., Sayer, J.R., Liu, H.X., Feng, Y., 2020. Data infrastructure for connected vehicle applications. Transp. Res. Rec. 2674 (5), 85–96. https://doi.org/10.1177/0361198120912424.

Wang, P., Wu, X., He, X., 2020. Modeling and analyzing cyberattack effects on connected automated vehicular platoons. Transp. Res. Part C: Emerg. Technol. 115, 102625 https://doi.org/10.1016/j.trc.2020.102625.

Whyte, W., Weimerskirch, A., Kumar, V., Hehn, T., 2013. A security credential management system for V2V communications. Presented at the 2013 IEEE Vehicular Networking Conference.

Yan, G., Olariu, S., Weigle, M.C., 2008. Providing VANET security through active position detection. Comput. Commun. 31 (12), 2883–2897. https://doi.org/10.1016/j.comcom.2008.01.009.

Yao, D., Zhang, C., Zhu, Z., Huang, J., Bi, J., 2017. Trajectory clustering via deep representation learning. Presented at the 2017 International Joint Conference on Neural Networks (IJCNN).

Yavvari, C., Duric, Z., Wijesekera, D., 2017. Vehicular dynamics based plausibility checking. Presented at the 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC).

Yen, C.-C., Ghosal, D., Zhang, M., Chuah, C.-N., Chen, H., 2018. Falsified data attack on backpressure-based traffic signal control algorithms. Presented at the 2018 IEEE Vehicular Networking Conference (VNC).

Zhang, D., Li, N., Zhou, Z.-H., Chen, C., Sun, L., Li, S., 2011. IBAT: Detecting Anomalous Taxi Trajectories from GPS Traces. New York, NY, USA.

Zhao, X., Abdo, A., Liao, X., Barth, M.J., Wu, G., 2021. Evaluating the cybersecurity risks of cooperative ramp merging in a mixed traffic environment. In: Presented at the Transportation Research Board 100th Annual Meeting, Washington, D.C.

Zhu, J., Jiang, W., Liu, A., Liu, G., Zhao, L., 2015. Time-dependent popular routes based trajectory outlier detection. In: Presented at the Wang, J. et al. (Eds.) Web Information Systems Engineering – WISE 2015.