Adversarial Diffusion Attacks on Graph-Based Traffic Prediction Models

Lyuyi Zhu, Kairui Feng, Ziyuan Pu, *Member, IEEE*, and Wei Ma, *Member, IEEE*

***Abstract*—Real-time traffic prediction models play a pivotal role in smart mobility systems and have been widely used in route guidance, emerging mobility services, and advanced traffic management systems. With the availability of massive traffic data, neural network-based deep learning methods, especially graph convolutional networks (GCNs) have demonstrated outstanding performance in mining spatio-temporal information and achieving high prediction accuracy. Recent studies reveal the vulnerability of GCN under adversarial attacks, while there is a lack of studies to understand the vulnerability issues of the GCN-based traffic prediction models. Given this, this article proposes a new task—diffusion attack, to study the robustness of GCN-based traffic prediction models. The diffusion attack aims to select and simulate attacks on a small set of nodes to degrade the performance of the traffic prediction models, and it can be used to examine vulnerabilities of the traffic prediction models. We propose a novel attack algorithm, which consists of two major components: 1) approximating the gradient of the black-box prediction model with simultaneous perturbation stochastic approximation (SPSA) and 2) adapting the knapsack greedy algorithm to select the attack nodes. The proposed algorithm is examined with three GCN-based traffic prediction models: 1) ST-GCN; 2) T-GCN; and 3) A3T-GCN on four cities. The proposed algorithm demonstrates high efficiency in adversarial attack tasks under various scenarios, and it can still generate adversarial samples under the drop regularization, such as DROPOUT, DROPNODE, and DROPEDGE. The research outcomes could help to improve the robustness of the GCN-based traffic prediction models and better protect the smart mobility systems.**

***Index Terms*—Adversarial attack, deep learning, graph convolutional network (GCN), intelligent transportation systems (ITSs), traffic prediction.**

Manuscript received 18 April 2022; revised 22 December 2022 and 4 May 2023; accepted 21 June 2023. Date of publication 28 June 2023; date of current version 25 December 2023. This work was supported in part by the National Natural Science Foundation of China under Grant 52102385; in part by the Research Grants Council of the Hong Kong Special Administrative Region, China, under Grant PolyU/25209221 and Grant PolyU/15206322; in part by the Research Institute for Sustainable Urban Development (RISUD) at the Hong Kong Polytechnic University under Project P0038288; and in part by the Otto Poon Charitable Foundation Smart Cities Research Institute (SCRI) at the Hong Kong Polytechnic University under Project P0043552. *(Corresponding author: Wei Ma.)*

Lyuyi Zhu is with the School of Data Science, City University of Hong Kong, Hong Kong, China (e-mail: lyzhu9-c@my.cityu.edu.hk).

Kairui Feng is with the Department of Civil and Environmental Engineering, Princeton University, Princeton, NJ 08544 USA (e-mail: kairuif@princeton.edu).

Ziyuan Pu is with the School of Transportation Engineering, Southeast University, Nanjing 210096, China, and also with the Department of Civil Engineering, School of Engineering, Monash University, Bandar Sunway 47500, Malaysia (e-mail: ziyuanpu@seu.edu.cn).

Wei Ma is with the Department of Civil and Environmental Engineering and the Research Institute for Sustainable Urban Development, The Hong Kong Polytechnic University, Hong Kong, SAR, China, and also with The Hong Kong Polytechnic University Shenzhen Research Institute, Shenzhen 515100, Guangdong, China (e-mail: wei.w.ma@polyu.edu.hk).

Digital Object Identifier 10.1109/JIOT.2023.3290401

I. INTRODUCTION

PEOPLE'S activities and movements in smart cities rely on accurate, robust, and real-time traffic information. With massive data collected in the intelligent transportation systems (ITSs), various methods, such as time series models, state-space models, and deep learning, have been developed to carry out the short-term prediction for traffic operation and management [1]. Among these methods, deep learning methods, especially graph convolutional networks (GCNs), achieve state-of-the-art accuracy and are widely employed in industry-level smart mobility applications. For example, Deepmind has partnered with Google Maps to improve the accuracy of real-time estimated time of arrival (ETA) prediction using GCN [2].

The predicted traffic information plays a critical role in our daily traveling, and travelers take for granted that the predicted results are accurate and trustworthy. However, the robustness and vulnerability issues of these deep learning models have not been investigated for traffic prediction models. Recent studies have shown that neural networks are vulnerable to deliberately designed samples, which are known as adversarial samples. In general, the adversarial samples could be generated by adding imperceptible perturbations to the original data sample. Though the adversarial sample is very similar to its original counterpart, it can significantly change the performance of the deep learning models. Szegedy et al. [3] first discovered this phenomenon on deep neural networks (DNNs), and they found that adversarial samples are low probability but densely distributed. Goodfellow et al. [4] also showed that neural networks are vulnerable to adversarial samples in the sense that it is sufficient to generate adversarial samples when DNNs demonstrate linear behaviors in high-dimensional spaces.

Due to the existence of adversarial samples, potential attackers could take advantage of deep learning models and degrade the model performance. Though related theories and applications have been studied in various areas, such as computer vision [5], social networks [6], traffic signs [7], and recommendation systems [8], few of the studies have investigated the vulnerability and robustness of the traffic prediction systems. It has been shown that industry-level traffic information systems can be "attacked" easily. Recently, a German artist walked slowly with a handcart, which was

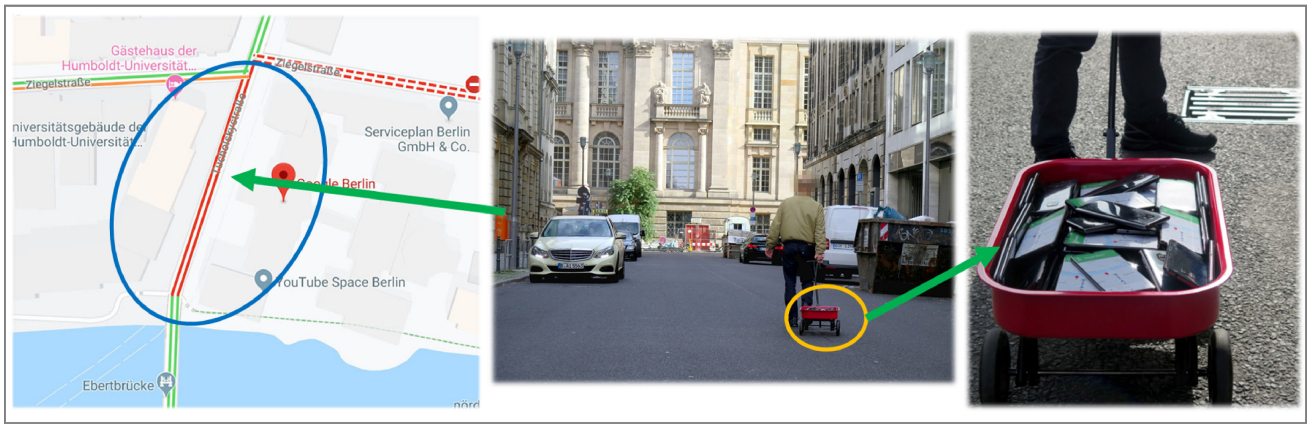


Fig. 1. Google Maps Hacks using 99 cell phones [9].

loaded with 99 smartphones. On each smartphone, the mobile application Google Maps was turned on. The 99 cell phones virtually created 99 vehicles on the road, and all the "vehicles" were slowly moving along the road. As Google Maps estimated and predicted the traffic states based on the data sent back from those cell phones, it wrongly identified an empty street (green) to be a congested road (red), as shown in Fig. 1. Another case occurred in Shanghai, China. A taxi driver also used many phones to create a congested road in the mobile APP, which aims to mislead other taxi drivers, and thus he will receive more orders from the passengers since other drivers always avoid driving to a congested road. Though it is unclear which models Google Maps and the APP are using, these experiments indeed reveal the possibility of adversarial attacks on real-world and industry-level traffic information systems.

Adversarial attacks on traffic prediction models can affect every aspect of smart mobility systems. We summarize the following four scenarios in which adversarial attacks can significantly degrade the performance of the systems.

1) *Smartphone-Based Mobility Applications:* Mobile phone-based mapping services, such as Google Maps and AutoNavi make traffic state estimation and prediction based on the GPS trajectories sent from their users [10]. However, users' mobile phones can be hacked and the information can be deliberately altered to attack the systems [11]. In general, these individual mobility applications are vulnerable to adversarial attacks because the difficulties of hacking users' mobile phones are way lower than hacking the application server.
2) *Connected Vehicle (CV) Systems:* In the CV systems, traffic information is collected by the roadside units (RSUs) and sent to the traffic control center [12]. CVs use information from either RSU or the control center to plan routes and avoid congestion. However, it is possible to hack some of the RSUs to send the adversarial samples to manipulate the predicted traffic information from the control center. Attackers could make use of adversarial attacks to benefit a group of vehicles while causing unexpected delays to other vehicles.
3) *Emerging Mobility Services:* Transportation network companies (TNCs) depend on accurate traffic speed prediction for vehicle dispatching, routing, and relocation on the central platform. It is possible that a group of vehicles collide with each other and falsely report their GPS trajectories and speed to confuse the central platform. By carefully designing the adversarial samples with purpose, this group of vehicles could take advantage of the central platform by receiving more orders and running on less congested roads.
4) *Advanced Traffic Management Systems (ATMSs):* Most of the network-wide transportation management systems [13] rely on user equilibrium (or stochastic user equilibrium) models to depict and predict travelers' behaviors and both models assume that travelers can acquire accurate (or nearly accurate) traffic information. Under adversarial attacks, this assumption no longer holds. The inaccurately predicted traffic information can reduce the effectiveness of the ATMS and degrade the efficiency of the entire network.

Due to the dynamic and complex nature of traffic prediction systems, the task of adversarial attack on these models is unique and distinct from existing literature. The scope of our study in this article is centered on GCN-based traffic prediction models.

The existing literature primarily focuses on attacking static graph-based models, overlooking the dynamic nature of traffic prediction models. Although few works consider the spatio-temporal graph attack method [14], their attack methods only focus on a targeted node. On real-world traffic networks, the impact of only attacking one node is localized. On the other hand, modifying the features on all nodes is challenging and expensive. Thus, our objective is to degrade the network-wide system performance instead of a single node. A practical task is to degrade the overall system performance by perturbing the features on a small subset of nodes, which is a more general attack framework. To this end, we propose a novel concept—diffusion attack, and its definition is presented in Definition 1, and the notations are consistent with those in Sections II and III.

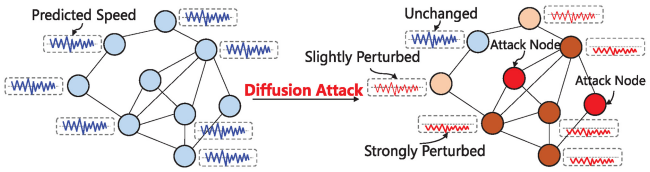


Fig. 2. Illustration of the diffusion attack.

Definition 1 (Diffusion Attack): Considering a GCN-based DNN $f(\mathbf{X}, \mathbf{A})$, the input features $\mathbf{X}$ are associated with all the nodes on the corresponding graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathbf{A}\}$. The diffusion attack aims to modify the input features from $\mathbf{X}$ to $\mathbf{X} + \mathbf{U}$ on a limited set of nodes $\mathcal{P} \subseteq \mathcal{V}$ while keeping the graph topology $\mathbf{A}$ unchanged, and the goal is to maximally change the prediction of the neural network f on all the nodes, i.e., $\max_{\mathbf{U}} \|f(\mathbf{X} + \mathbf{U}, \mathbf{A}) - f(\mathbf{X}, \mathbf{A})\|$.

An illustration of the diffusion attack is presented in Fig. 2. On the left-hand side, the prediction model performs normally and can generate accurate predictions. After the diffusion attack, two nodes in red are attacked, and their nearby neighbors are strongly perturbed, followed by their 2-hop neighbors being slightly perturbed. One can see that the attack effects diffuse from the node being attacked to its neighbors, and later we will develop mathematical proofs and numerical experiments to verify this phenomenon. The purpose of the attackers is to select the optimal attack nodes and to generate adversarial samples to maximize attack effects.

To summarize, the vulnerability issues of the traffic prediction models are critical to smart mobility systems, while the related research is still lacking. Given this, we explore the robustness of the GCN-based traffic prediction models. Based on the unique characteristics of the adversarial attacks on traffic prediction models, we propose the concept of diffusion attack, which aims to examine the vulnerability of the models by simulating attacks on a small set of nodes to degrade the performance of the entire network. On top of that, we develop a diffusion attack algorithm, which consists of two major components: 1) approximating the gradient of the black-box prediction model with SPSA and 2) adapting the knapsack greedy algorithm to select nodes to attack. The proposed algorithm is examined with three GCN-based traffic prediction models: 1) ST-GCN; 2) T-GCN; and 3) A3T-GCN on four cities: 1) Los Angeles; 2) Hong Kong; 3) Bay Area; and 4) Seattle. The proposed algorithm demonstrates high efficiency in the diffusion attack tasks under various scenarios, and the algorithm can still generate adversarial samples under different drop regularization. We further discuss how to improve the robustness of the GCN-based traffic prediction models, and the research outcomes could help to better protect the smart mobility systems in both the cyber and physical world. The contributions of this study are summarized as follows.

1) Different from existing adversarial attack tasks on graphs, we propose a novel task of diffusion attack, which aims to select and simulate attacks on a small set of nodes to degrade the performance of the entire traffic prediction models. This task is suitable for the traffic prediction context, while it is overlooked in the existing literature.
2) To generate the adversarial samples on traffic prediction models, we propose the SPSA algorithm to efficiently approximate gradients of the black-box prediction models.
3) To select the optimal attack nodes, we formulate the diffusion problem as a knapsack problem and then adapt the greedy algorithm to determine the priority of the attack nodes.
4) We conduct extensive experiments to attack the widely-used traffic prediction models in Los Angele, Hong Kong, Bay Area, and Seattle. Different drop regularization strategies (e.g., DROPOUT, DROPEDGE, and DROPNODE) for defending the adversarial attacks are also tested to ensure that the proposed algorithm can still generate effective adversarial samples in various scenarios.

The remainder of this article is organized as follows. Section II reviews the related studies on traffic prediction, GCN models, and adversarial attacks on graphs. Section III rigorously formulates the diffusion attack problem and presents the developed attack algorithms. In Section IV, three traffic prediction models and four real-world data sets are used to examine the proposed attack algorithms. Finally, conclusions and future research are summarized in Section V.

II. RELATED WORKS

In this section, we first overview the traffic prediction tasks and then summarize recent studies on GCN-based traffic prediction models. Robustness issues of the GCN-based prediction models under adversarial attacks are also discussed.

A. Traffic Prediction

The traffic prediction problem has been extensively studied for decades, and various statistical models have been developed to solve the problem, including history average (HA) [15], autoregressive integrated moving average (ARIMA) [16], [17], [18], [19], [20], support vector regression (SVR) [21], clustering [22], and Kalman filtering [23], [24]. In recent years, the data scale becomes large and the spatio-temporal correlation of the data becomes complicated, hence traditional statistical methods reveal their limits in face of the massive and complex data. Instead, neural network models demonstrate potential in traffic prediction with multisource data on large-scale networks. Various neural network models have been used for traffic prediction, including convolutional neural network (CNN) [25], recurrent neural network (RNN) [26], [27], [28], attention [29], [30], and GCN [31], [32], [33], [34].

Traffic prediction tasks can also be categorized into multiple purposes, such as traffic state prediction, demand prediction, and trajectory prediction [35]. Traffic state prediction includes the prediction of traffic flow [29], speed [31], and travel time [36]. Traffic demand prediction aims to make a prediction of the number of users and traffic demand, such as taxi request [37], subway inflow/outflow [38], bike-sharing demand [39], [40], and origin-destination demand [41]. It is

also possible to predict the trajectories of travelers and vehicles, and this task is used for dynamic positioning and resource allocation [42], [43]. Overall, most of the traffic prediction tasks can be carried out by neural network models, and hence it is crucial to study their vulnerability issues.

B. GCN and Its Applications on Traffic Prediction

Traffic data is closely associated with the topological structure of the road networks, and hence it is typical graph-based data. The graph-based data is represented in the non-Euclidean space, and conventional machine learning methods (e.g., multilayer perceptron) overlook the graph-based inter-relationship among data [44]. In this article, we summarize that traffic data consists of the following two types of information.

1) *Spatial Information:* Traffic-related data can be presented on a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{A})$, where $\mathcal{V} = \{1, 2, \ldots, N\}$ represents a set of nodes with $N = |\mathcal{V}|$, and $\mathcal{E}$ denotes a set of edges. In traffic prediction, one way to construct the graph is to make each node v_i represent a road segment, and each edge represents the connectivity relationship between the road segments [31]. We further define the adjacency matrix $\boldsymbol{A} \in \{0, 1\}^{N\times N}$ on the graph $\mathcal{G}$, where $A_{ij} = 1$ when node i connects node j, and 0 otherwise.
2) *Temporal Information:* Traffic data, such as traffic speed, density, and flow, on each node can be viewed as a time series. The traffic data on the graph is represented by a feature matrix $\boldsymbol{X} \in \mathbb{R}^{N\times S}$, where S denotes the number of time intervals in the study period.

GCN demonstrates great capacities in learning graph-based information for various applications. Short-term traffic prediction is one of the most important and practical applications. The GCN can be used to extract the spatial (non-Euclidean) relationship among nodes [31]. As the backbone of a GCN-based model, the graph convolutional layer is defined as follows:

$$\boldsymbol{H}^{(l+1)} = \sigma\left(\hat{\boldsymbol{A}}\boldsymbol{H}^{(l)}\boldsymbol{W}^{(l)}\right)$$

where l is the layer index, $\boldsymbol{H}^{(0)} = \boldsymbol{X}$, and $\hat{\boldsymbol{A}} = \widetilde{\boldsymbol{D}}^{-(1/2)}\widetilde{\boldsymbol{A}}\widetilde{\boldsymbol{D}}^{-(1/2)}$ is the Laplacian matrix. $\widetilde{\boldsymbol{D}}$ is the corresponding degree matrix $\widetilde{\boldsymbol{D}}_{ii} = \sum_j \boldsymbol{A}_{ij}$, and $\widetilde{\boldsymbol{A}} = \boldsymbol{A} + \boldsymbol{I}_N$, where $\boldsymbol{I}_N$ is an $N \times N$ identity matrix. $\sigma(\cdot)$ represents the activation function and $\boldsymbol{W}^{(l-1)}$ are parameters of the lth layer.

An L-layers GCN model can be expressed as following:

$$\boldsymbol{Y} = f(\boldsymbol{X}, \boldsymbol{A}) = g\left(\hat{\boldsymbol{A}} \ldots \sigma\left(\hat{\boldsymbol{A}}\boldsymbol{X}\boldsymbol{W}^{(0)}\right) \cdots \boldsymbol{W}^{(L-1)}\right) \quad (1)$$

where $g(\cdot)$ is a generalized function, $\boldsymbol{Y} \in \mathbb{R}^{N\times T}$ and the parameters $\boldsymbol{W}^{(l-1)}$ for each layer could be learned by minimizing the loss between the estimated $\boldsymbol{Y}$ and true $\boldsymbol{Y}_{\text{true}}$, represented by $\mathcal{L}(\boldsymbol{Y}, \boldsymbol{Y}_{\text{true}})$. GCN can couple with RNNs to learn the spatio-temporal patterns of the traffic data. For example, long short-term memory (LSTM) is widely used with GCN for traffic prediction [45], and the gated recurrent unit (GRU) is also adopted to model the time series on each node separately regardless of the graph structures [31]. The GRU architecture is formulated as follows:

$$\begin{aligned}
\boldsymbol{z}_t &= \sigma(\boldsymbol{W}_z\boldsymbol{x}_t + \boldsymbol{U}_z\boldsymbol{h}_{t-1} + \boldsymbol{b}_z) \\
\boldsymbol{r}_t &= \sigma(\boldsymbol{W}_r\boldsymbol{x}_t + \boldsymbol{U}_r\boldsymbol{h}_{t-1} + \boldsymbol{b}_r) \\
\boldsymbol{c}_t &= \phi(\boldsymbol{W}_c\boldsymbol{x}_t + \boldsymbol{U}_c(\boldsymbol{r}_t \odot \boldsymbol{h}_{t-1}) + \boldsymbol{b}_c) \\
\boldsymbol{h}_t &= (\boldsymbol{1} - \boldsymbol{z}_t) \odot \boldsymbol{h}_{t-1} + \boldsymbol{z}_t \odot \boldsymbol{c}_t
\end{aligned} \quad (2)$$

where $\boldsymbol{z}_t$, $\boldsymbol{r}_t$, and $\boldsymbol{c}_t$ are the update gate vector, reset gate vector, and candidate vector at time t. $\boldsymbol{x}_t$ and $\boldsymbol{h}_t$ are the input and output vector, respectively. $\boldsymbol{W}$ and $\boldsymbol{U}$, and $\boldsymbol{b}$ are the parameter matrices and vector. The GRU units take the hidden status $\boldsymbol{h}_{t-1}$ at time $t-1$ and the input $\boldsymbol{x}_t$ to update the current status $\boldsymbol{h}_t$ at time t, and hence it captures the temporal patterns in time series.

Some emerging models for time-series modeling, such as gated CNN and attention, can also be incorporated into the GCN-based deep learning framework [29], [32], [33], [34]. Readers are referred to [35] for a comprehensive review of the GCN-based traffic prediction methods.

This article will adopt the GCN as the backbone model and study its robustness and vulnerability issues under adversarial attacks.

C. Adversarial Attacks on Graphs

Like other neural networks, GCN is vulnerable to adversarial attacks. Various attack concepts on the graph have been proposed, such as targeted and nontargeted attacks, structure and feature attacks, and poisoning and evasion attacks. Targeted attacks aim to attack a target node [46], while the nontargeted attacks aim to compromise the global performance of a model [47], [48], [49], [50]. Structure attacks modify the structure of the graph, such as adding or deleting nodes/edges [48], [51], [52], and feature attacks perturb the labels/features of nodes without changing the connectivity of graph [14], [47], [53]. Poisoning attacks modify the training data [47], and evasion attacks insert adversarial samples when using the models [14], [48], [49], [54]. To be specific, attacks on the traffic prediction systems are nontargeted, feature, and evasion attacks, which have not been well studied in the existing literature.

The attack algorithms can be further categorized into white-box attacks and black-box attacks. In white-box attacks, the neural network structure and parameters, training methods, and training samples are exposed to attackers. Attackers could utilize the neural network model to generate adversarial samples. Many white-box methods achieve great performance, such as the fast gradient sign method (FGSM) [4], Jacobian-based saliency map approach (JSMA) [55], Carlini and Wagner attacks (CW) [56], and Deepfool [57]. In contrast, in black-box attacks, attackers know little about the internal information of the target neural network model, especially the model structure and parameters, and only the model input and output are exposed to the attackers. For example, if the attackers plan to attack the traffic prediction system, he/she is unlikely to know the internal structure or information of the prediction model.

Though it is more common in the real world, the black-box attack is much more challenging than the white-box attack. Black-box attacks can be achieved through response

surface models [58] and meta-heuristics. For instance, one-pixel attack [5] uses the differential evolution (DE) algorithm and decision-based attack [59] uses covariance matrix adaptation evolutionary strategies (CMA-ESs) to generate and improve the adversarial samples by iteration. It is also possible to approximate the gradient of the target models, and the representative models include zeroth order optimization (ZOO) [60], autoencoder-based ZOO method (AutoZOOM) [61], and natural evolutionary strategies (NESs) [62]. Besides, the semi-black-box attacks are in between, and it is assumed that the information of the prediction models is partially observed [63]. The above attack algorithms mainly focus on the classification task, while attack methods for the traffic prediction models (i.e., regression task) are still lacking.

III. Proposed Works

In this section, we first present the general formulation of the adversarial attack on graphs. Then, a new concept of diffusion attack is proposed for the traffic prediction models. Finally, we propose a new formulation and algorithm to construct the diffusion attack and discuss its implementations.

A. Preliminaries

Traffic prediction on graphs can be regarded as a graph-based regression problem [35]. Using $f : \mathbb{R}^{N\times S} \to \mathbb{R}^{N\times T}$ in (1) as a regression model (e.g., a traffic prediction model) on the graph $\mathcal{G}$, and f contains an L-layer GCN, where S represents the look-back time interval, and T is the number of time intervals to predict. The input feature matrix $\boldsymbol{X} \in \mathbb{R}^{N\times S}$ contains the historical traffic states, and the output $\boldsymbol{Y} \in \mathbb{R}^{N\times T}$ is the traffic states we want to predict. For the traffic prediction problems, $\boldsymbol{Y}$ mainly depends on the $\boldsymbol{X}$ as most of $\boldsymbol{A}$ are fixed [35]. We further denote the $\boldsymbol{X} = (\boldsymbol{x}_1; \cdots ; \boldsymbol{x}_N)$, $\boldsymbol{Y} = f(\boldsymbol{X}) = (\boldsymbol{y}_1; \cdots ; \boldsymbol{y}_N)$, and $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$ are the ith row of $\boldsymbol{X}$ and $\boldsymbol{Y}$, respectively. For node i, $\boldsymbol{x}_i$ is the ith row of the feature matrix $\boldsymbol{X}$, and $\boldsymbol{x}_i$ represents a time series of speed on node i. The prediction model for node i can be written as $\boldsymbol{x}_i = (x_{i1}, x_{i2}, \ldots, x_{iS}) \mapsto \boldsymbol{y}_i = (y_{i1}, y_{i2}, \ldots, y_{iT})$, where $x_{i\cdot}$ is the historical traffic states, and $y_{i\cdot}$ is the future traffic states on node i.

B. Diffusion Attacks on Traffic State Forecasting Models

This article aims to test the robustness of the traffic prediction system by adding perturbations on the feature matrix $\boldsymbol{X}$, to maximally change the prediction results over a selected set of nodes. As discussed above, the graph structure is fixed and cannot be changed easily for the problem of traffic prediction, so we assume that $\boldsymbol{A}$ is fixed throughout this article.

We construct an adversarial sample by adding the perturbation $\boldsymbol{U}$, which is the same size as $\boldsymbol{X}$, to the original input feature matrix $\boldsymbol{X}$, as represented by $\boldsymbol{X}' = \boldsymbol{X}+\boldsymbol{U}$. Consequently, the corresponding output changes from $\boldsymbol{Y} = f(\boldsymbol{X})$ to $\boldsymbol{Y}' = f(\boldsymbol{X}+\boldsymbol{U})$.

We suppose that attackers select a set of nodes $\mathcal{P} \subseteq \mathcal{V}$ to attack. For each node $i \in \mathcal{P}$, $\boldsymbol{u}_i \begin{cases} \neq \boldsymbol{0}, & i \in \mathcal{P} \\ = \boldsymbol{0}, & i \notin \mathcal{P} \end{cases}$, where $\boldsymbol{u}_i$ is the ith row of $\boldsymbol{U}$. Then, the adversarial sample can be expressed as follows:

$$\boldsymbol{X}' = \boldsymbol{X} + \boldsymbol{U} = (\boldsymbol{x}_1 + \boldsymbol{u}_1; \cdots ; \boldsymbol{x}_i + \boldsymbol{u}_i; \cdots ; \boldsymbol{x}_N + \boldsymbol{u}_N).$$

By perturbing the node i, we change the original prediction result on node i (denoted as $\boldsymbol{y}_i$) to $\boldsymbol{y}_i'$. The attack influence function $\phi_i(\boldsymbol{U})$ on node i is defined as follows:

$$\phi_i(\boldsymbol{U}) = \mathcal{L}\big(\boldsymbol{y}_i'(\boldsymbol{X}'), \boldsymbol{y}_i(\boldsymbol{X})\big) \tag{3}$$

where we use $\boldsymbol{y}_i(\boldsymbol{X})$ to indicate that $\boldsymbol{y}_i$ is a function of $\boldsymbol{X}$. $\mathcal{L}(\cdot, \cdot)$ represents loss function between $\boldsymbol{y}_i'$ and $\boldsymbol{y}_i$. Here, the attack influence evaluates the difference between the original prediction (instead of true speed) and perturbed speed. The reason is that we assume the prediction is accurate enough, otherwise there is no need to attack. On the other hand, we could never know the true traffic condition in the future, so it is impossible to perturb the prediction against the true values.

To mathematically characterize the diffusion phenomenon when attacking the GCN, we prove that the following Proposition 1 holds.

Proposition 1: Using the L-layer GCN model presented in (1) for traffic prediction, the effect of perturbation $\boldsymbol{U}$ on each node i, which is denoted as $\phi_i(\boldsymbol{U})$, depends on the perturbations of its L-hop neighbors.

Proof: Using $|\cdot|$ to represent the element-wise absolute value operator, and assuming that $g(\cdot)$ is Lipschitz continuous with constant M, $[\cdot]_i$ is the ith row of a matrix, $\sigma(\cdot)$ is the `Relu` function, and $\mathcal{L}$ represents the mean squared error (MSE), we have

$$\begin{aligned}
\phi_i(\boldsymbol{U}) &= \left\|\boldsymbol{y}_i'(\boldsymbol{X}') - \boldsymbol{y}_i(\boldsymbol{X})\right\|_2^2 \\
&= \left\|\left[g\left(\hat{\boldsymbol{A}}\boldsymbol{H}'^{(L-1)}\boldsymbol{W}^{(L-1)}\right) - g\left(\hat{\boldsymbol{A}}\boldsymbol{H}^{(L)}\boldsymbol{W}^{(L-1)}\right)\right]_i\right\|_2^2 \\
&\le M\left\|\left[\hat{\boldsymbol{A}}\left(\boldsymbol{H}'^{(L-1)} - \boldsymbol{H}^{(L-1)}\right)\boldsymbol{W}^{(L-1)}\right]_i\right\|_2^2 \\
&\le M\left\|\left[\hat{\boldsymbol{A}}\left(|\boldsymbol{H}'^{(L-1)} - \boldsymbol{H}^{(L-1)}|\right)|\boldsymbol{W}^{(L-1)}|\right]_i\right\|_2^2 \\
&\le M\left\|\left[\hat{\boldsymbol{A}}\left(|\boldsymbol{H}'^{(L-1)} - \boldsymbol{H}^{(L-1)}|\right)\right]_i\right\|_2\left\||\boldsymbol{W}^{(L-1)}|\right\|_2^2 \\
&\le MW\left\|\left[\hat{\boldsymbol{A}}\left(|\boldsymbol{H}'^{(L-1)} - \boldsymbol{H}^{(L-1)}|\right)\right]_i\right\|_2^2 \\
&\le MW^2\left\|\left[\hat{\boldsymbol{A}}^2\left(|\boldsymbol{H}'^{(L-2)} - \boldsymbol{H}^{(L-2)}|\right)\right]_i\right\|_2^2 \\
&\le MW^L\left\|\left[\hat{\boldsymbol{A}}^L\left(|\boldsymbol{H}'^{(0)} - \boldsymbol{H}^{(0)}|\right)\right]_i\right\|_2^2 \\
&= MW^L\left\|\hat{\boldsymbol{A}}_{i\cdot}^L|\boldsymbol{U}|\right\|_2^2
\end{aligned}$$

where $\||\boldsymbol{W}^{(l)}|\|_2^2 \le W$ and $\hat{\boldsymbol{A}}_{i\cdot}^L$ represents the ith row of $\hat{\boldsymbol{A}}^L$. Here, we can see that the upper bound of attack influence $\phi_i(\boldsymbol{U})$ associates closely with $\|\hat{\boldsymbol{A}}_{i\cdot}^L|\boldsymbol{U}|\|_2^2$. Denoting $|\boldsymbol{U}|_{\cdot k}$ as the kth column of $|\boldsymbol{U}|$, we can expand $\|\hat{\boldsymbol{A}}_{i\cdot}^L|\boldsymbol{U}|\|_2^2$ as

follows:

$$\left\|\hat{\boldsymbol{A}}_{i\cdot}^{L}|\boldsymbol{U}|\right\|_2^2 = \left\|\left[\hat{\boldsymbol{A}}_{i\cdot}^{L}|\boldsymbol{U}|_{\cdot 1},\ldots,\hat{\boldsymbol{A}}_{i\cdot}^{L}|\boldsymbol{U}|_{\cdot k},\ldots,\hat{\boldsymbol{A}}_{i\cdot}^{L}|\boldsymbol{U}|_{\cdot S}\right]\right\|_2^2 = \sum_{k=1}^{S}\left(\hat{\boldsymbol{A}}_{i\cdot}^{L}|\boldsymbol{U}|_{\cdot k}\right)^2.$$

For each k we can further expand it as the sum of perturbation on different nodes

$$\hat{\boldsymbol{A}}_{i\cdot}^{L}|\boldsymbol{U}|_{\cdot k} = \sum_{h=1}^{N}\hat{\boldsymbol{A}}_{ih}^{L}|\boldsymbol{U}|_{hk}$$

where $|\boldsymbol{U}|_{hk}$ is the absolute value of perturbation added on kth historical traffic state of node h. $\hat{\boldsymbol{A}}_{ih}^{L}$ represents normalized connectivity weight between node i and node h (similar to $\boldsymbol{A}_{ih}^{L}$, which represents number of L-hop paths between i and h, according to the graph theory). If $\hat{\boldsymbol{A}}_{ih}^{L}$ is large, node h will have more impact on $\phi_i(\boldsymbol{U})$. Especially, if node h is out of L-hop neighbors of node i, then $\hat{\boldsymbol{A}}_{ih}^{L} \equiv 0$, which means that the attack effect will not diffuse to node i when attacking h. For most GCN-based traffic prediction models $L \leq 3$, so the effect of $\boldsymbol{U}$ only diffuses to its local neighbors. ■

Proposition 1 holds for both white-box and black-box attack algorithms, which indicates that attacks on each node are localized and hence this could further inspire the development of solution algorithms for the diffusion attack. As discussed in the previous section, this article focuses on the novel concept of diffusion attack, which aims at changing the network-wide prediction results by perturbing a small subset of node features. Mathematically, the diffusion attack problem can be expressed as to find the optimal $\mathcal{P}$ and the corresponding perturbation $\boldsymbol{U}$ such that the following influence function $\Phi(\boldsymbol{U})$ is maximized:

$$\begin{aligned}\max_{\boldsymbol{U},z}\ & \Phi(\boldsymbol{U}) = \sum_{i\in\mathcal{V}} w_i\phi_i(\boldsymbol{U})\\ \text{s.t.}\ & -\varepsilon^{-}z_i\boldsymbol{x}_i \leq \boldsymbol{u}_i \leq \varepsilon^{+}z_i\boldsymbol{x}_i \quad \forall i\in\mathcal{V}\\ & \sum_{i\in\mathcal{V}} b_i z_i \leq B\\ & z_i\in\{0,1\} \quad \forall i\in\mathcal{V}\end{aligned} \tag{4}$$

where z_i indicates whether node i is attacked. To be precise, $z_i = \begin{cases}1, & i\in\mathcal{P}\\ 0, & i\notin\mathcal{P}\end{cases}$. b_i represents the cost of attacking node i, and B is the total budget. The objective function $\Phi(\boldsymbol{U})$ represents attack influence function for the entire network, and w_i is a predetermined importance weight of node i. $\varepsilon^{-}, \varepsilon^{+} > 0$ are used to control the scale of the perturbations.

Formulation (4) is a mixed-integer programming (MIP), and it contains two components: 1) determining $\boldsymbol{U}$ given a fixed $\mathcal{P}$ and 2) determining $\mathcal{P}$. The two components will be discussed in Sections III-C and III-D, respectively.

C. Black-Box Attacks Using SPSA

Given a fixed $\mathcal{P}$, (4) reduces to a continuous optimization problem with box constraints, as shown in the following equation:

$$\begin{aligned}\max_{\boldsymbol{U}}\ & \Phi(\boldsymbol{U}) = \sum_{i\in\mathcal{V}} w_i\phi_i(\boldsymbol{U})\\ \text{s.t.}\ & -\varepsilon^{-}\boldsymbol{x}_i \leq \boldsymbol{u}_i \leq \varepsilon^{+}\boldsymbol{x}_i \quad \forall i\in\mathcal{P}.\end{aligned} \tag{5}$$

Formulation (5) suggests that an ideal perturbation $\boldsymbol{U}^*$ should be small, and meanwhile it maximizes the overall influence function $\Phi(\boldsymbol{U}^*)$. In real-world applications, the internal information about the traffic prediction model is opaque, hence it is proper to consider the traffic prediction model as a "black box." We assume both the input feature $\boldsymbol{X}$ and the prediction results $\boldsymbol{Y}$ are known to the attackers as both matrices represent the true and predicted traffic states in real world, and hence (5) can be viewed as a black-box optimization problem. To solve it, we adopt the SPSA method, which is featured with its efficiency and scalability [64]. In recent years, SPSA has been used for adversarial attacks on the classification problems [65], while it has not been used for the regression problems, in particular, the traffic prediction problem.

The SPSA method uses finite differences between two randomly perturbed inputs $(\mathbf{U}^{+}, \mathbf{U}^{-})$ to approximate the gradient of the objective function. Mathematically, the gradient of Φ can be calculated as follows:

$$\widehat{\nabla\Phi}(\boldsymbol{U}_n) = \frac{\Phi\left(\mathbf{U}^{+}\right) - \Phi\left(\mathbf{U}^{-}\right)}{2c_n\boldsymbol{\Delta}_n} \tag{6}$$

where $\mathbf{U}^{+} = \boldsymbol{U}_n + c_n\boldsymbol{\Delta}_n$, $\mathbf{U}^{-} = \boldsymbol{U}_n - c_n\boldsymbol{\Delta}_n$, n is the index of the iteration, $\boldsymbol{\Delta}_n$ is a random perturbation vector whose elements are sampled from Rademacher distribution (Bernoulli ± 1 distribution with probability $p = 0.5$, and we denote $\text{RAD}_{1\times S} \in \{-1, 1\}^{1\times S}$ as a sample vector that follows Rademacher distribution). We further denote sequences $\{c_n\}$ and $\{a_n\}$ as follows:

$$a_n = \frac{a}{(\eta+n)^{\alpha}} \quad c_n = \frac{c}{n^{\gamma}} \tag{7}$$

where a, c, α, and γ are hyper-parameters for SPSA [64]. Both sequences decrease when the iteration n increases. Then the gradient ascent approach is utilized to maximize $\Phi(\boldsymbol{U})$, as shown in the following equation:

$$\boldsymbol{U}_{n+1} = \boldsymbol{U}_n + a_n\widehat{\nabla\Phi}(\boldsymbol{U}_n) \quad \forall n. \tag{8}$$

We analyze the choice of the hyper-parameter c_n and a_n. Suppose the gradient $\nabla\Phi(\boldsymbol{U})$ is L-Lipschitz continuous, where $\nabla\Phi(\boldsymbol{U})$ is a matrix in $\mathbb{R}^{N\times S}$ and it is flattened to a vector in $\mathbb{R}^{NS}$. Assuming $\alpha\in\mathbb{R}$ and $g(\alpha) = \Phi(\boldsymbol{U}_n + (\alpha-1)\,c_n\boldsymbol{\Delta}_n)$, we have $g(2) = g(0) + \int_0^2 g'(\alpha)d\alpha$. By the Cauchy–Schwarz inequality

$$\begin{aligned}g'(\alpha) - g'(1) &= c_n(\nabla\Phi(\boldsymbol{U}_n + \alpha c_n\boldsymbol{\Delta}_n) - \nabla\Phi(\boldsymbol{U}_n))^{\top}\boldsymbol{\Delta}_n\\ &\leq Lc_n{}^2\alpha\|\boldsymbol{\Delta}_n\|^2\end{aligned}$$

and then

$$\begin{aligned}g(2) - g(0) &= \int_0^2 g'(\alpha)d\alpha\\ &\leq 2\left(Lc_n{}^2\|\boldsymbol{\Delta}_n\|^2 + c_n\nabla\Phi(\boldsymbol{U}_n)^{\top}\boldsymbol{\Delta}_n\right).\end{aligned}$$

Algorithm 1 Determine the Adversarial Sample $\boldsymbol{X}'$ and Optimal Perturbation $\boldsymbol{U}$ Given a Fixed $\mathcal{P}$

Input: Traffic prediction model $f(\boldsymbol{X})$, input feature matrix $\boldsymbol{X}$, attack set $\mathcal{P}$, maximum iteration $\texttt{MaxIter}$.
Output: Adversarial sample $\boldsymbol{X}'$, and optimal perturbation $\boldsymbol{U}$.
Initialize $\boldsymbol{U}_1 \in \mathbb{R}^{N\times S}$
for $n = 1, 2, \cdots, \texttt{MaxIter}$ **do**
  Update a_n and c_n based on Equation (7).
  For $i \in \mathcal{V}$, random sample $\boldsymbol{\delta}_i = \begin{cases} \mathbf{0}_{1\times S} & \text{if } i \notin \mathcal{P} \\ \text{RAD}_{1\times S} & \text{if } i \in \mathcal{P} \end{cases}$
  $\boldsymbol{\Delta} = (\boldsymbol{\delta}_1; \cdots; \boldsymbol{\delta}_i; \cdots; \boldsymbol{\delta}_N)$.
  $\boldsymbol{U}^+ \leftarrow \boldsymbol{U}_n + c_n\boldsymbol{\Delta}$; $\boldsymbol{U}^- \leftarrow \boldsymbol{U}_n - c_n\boldsymbol{\Delta}$.
  Compute $\widehat{\nabla\Phi}(\boldsymbol{U}_n)$ based on Equation (6).
  Compute $\boldsymbol{U}_{n+1}$ based on Equation (8).
  Set $(\boldsymbol{u}_1; \cdots; \boldsymbol{u}_i; \cdots; \boldsymbol{u}_N) \leftarrow \boldsymbol{U}_{n+1}$.
  for $i \in \mathcal{V}$ **do**
    if $i \in \mathcal{P}$ **then**
      $\boldsymbol{u}_i \leftarrow \min(\epsilon^+\boldsymbol{x}_i, \boldsymbol{u}_i)$
      $\boldsymbol{u}_i \leftarrow \max(-\epsilon^-\boldsymbol{x}_i, \boldsymbol{u}_i)$
    else
      $\boldsymbol{u}_i \leftarrow \mathbf{0}$
    end if
  end for
  $\boldsymbol{U}_{n+1} \leftarrow (\boldsymbol{u}_1; \cdots; \boldsymbol{u}_i; \cdots; \boldsymbol{u}_N)$
end for
$\boldsymbol{U} \leftarrow \boldsymbol{U}_{\texttt{MaxIter}+1}$
$\boldsymbol{X}' \leftarrow \boldsymbol{X} + \boldsymbol{U}$
Return $\boldsymbol{X}'$, $\boldsymbol{U}$

The gradient estimator $\widehat{\nabla\Phi}(\boldsymbol{U}_n)$ is bounded by $(Lc_n\|\boldsymbol{\Delta}_n\|^2 + \nabla\Phi(\boldsymbol{U}_n)^\top\boldsymbol{\Delta}_n)(\mathbf{1}/\boldsymbol{\Delta}_n)$, where $(\mathbf{1}/\boldsymbol{\Delta}_n)$ is an element-wise matrix division. If $0 < c_n \le [(|\nabla\Phi(\boldsymbol{U}_n)^\top\boldsymbol{\Delta}_n|)/(L\|\boldsymbol{\Delta}_n\|^2)]$, we have

$$\mathbb{P}\left(\widehat{\nabla\Phi}(\boldsymbol{U}_n)^\top\boldsymbol{\Delta}_n \le 0 \mid \nabla\Phi(\boldsymbol{U}_n)^\top\boldsymbol{\Delta}_n \le 0\right) = 1.$$

We can see that c_n is related to the sign of the bound and a large c_n will increase the randomness of the SPSA method. To explore the entire graph and determine the most critical (sensitive) nodes, it would be better to set larger c_n and a_n in SPSA. To summarize, the adversarial attack with fixed node set $\mathcal{P}$ is presented in Algorithm 1.

D. Node Selection Using Knapsack Greedy

This section focuses on determining the attack set $\mathcal{P}$ with a limited budget B. The cost b_i of attacking node varies depending on their level of difficulty. For instance, a successful attack on certain urban roads with a high volume of traffic flow may require more mobile devices (b_i is high). Additionally, some urban roads may have a more advanced and secure information collection system that can detect and alert potential attacks using anomaly detection methods, while rural roads may be more vulnerable and easier to attack (b_i is low). In contrast, attacks on urban roads usually generate a higher impact on traffic prediction methods because the urban traffic volumes are high. It can be seen that there is a tradeoff between the cost and benefit when selecting the attack set $\mathcal{P}$.

We review the formulation of the diffusion attack in (4), and it is actually similar to the 0–1 knapsack problem, except that the utility of each node i ($\phi_i(\boldsymbol{U})$) is unknown [66], and there is a complex nonlinear combined effect since the diffusion influence of nodes can affect each other. The attack set $\mathcal{P}$ can be viewed as a knapsack with maximum capacity B, and the nodes are items with their weight w_i. Node i is added to $\mathcal{P}$ if $z_i = 1$. Due to the nature of integer programming, there is no provably efficient method to solve (4), which is an NP-hard problem. Real-world networks contain hundreds or thousands of nodes, and hence it is impractical to enumerate all possible integer solutions. In this article, we develop a family of Knapsack Greedy (KG) algorithms to approximately solve for (4), and those algorithms are inspired by the original greedy algorithm for the knapsack problem.

A trivial but insightful observation is that any perturbation $\mathbf{U}$ could reduce the performance of the prediction model. Proposition 2 shows that the convexity exists if we only perturb one node and the perturbation u is small enough, then the objective function in (4) is locally convex.

Proposition 2: Suppose $\phi_i(\boldsymbol{U}) \ge 0$ always holds for $\forall i$. Then, the objective function Φ in (4) is locally convex under small perturbation $\boldsymbol{U}$.

Proof: Given function Φ is smooth and attains global optimal when the perturbation $\boldsymbol{U} = \mathbf{0}$, there exists one region around $\boldsymbol{U} = \mathbf{0}$, in which $\Phi(\boldsymbol{U})$ is convex. ■

Existing literature has shown that the convex separable nonlinear knapsack problems could be approximated with the greedy search framework described by Algorithm 2 [67]. Recalling Proposition 1, the perturbation on GCN-based models would only arise local effect, which means if the attack nodes are selected L-hop away from each other, the objective of (3) would be separable. Besides, it is observed that the combined effect on attack nodes is relatively small. We also noticed that the objective is locally convex given the attack on only one node in Proposition 2. Combing the enlightenment from Propositions 1 and 2, we can consider (4) as an approximate separable knapsack problem. Thus, the greedy search framework would work efficiently for solving this problem. Intuitively, we can determine the effectiveness of the node attack by perturbing each node and examining which node is more sensitive. Such effectiveness measures can be used to develop our attack algorithm in the following sections.

The proposed solution procedure consists of two steps: 1) compute $\hat{\phi}_i$ to approximate ϕ_i for each i and 2) adopt the greedy algorithm for the standard knapsack problem with $\hat{\phi}_i$ as the utility. In step 1, we propose that the utility $\hat{\phi}_i$ can be obtained by SPSA. To be precise, we run Algorithm 1 with K_1 iterations for each node $i \in \mathcal{V}$ with $\mathcal{P} = \{i\}$, and the algorithm outcome is $\boldsymbol{U}_i$. Then, ϕ_i is approximated by $\hat{\phi}_i = \phi_i(\boldsymbol{U}_i)$; in step 2, we initialize the attack set $\mathcal{P}$ as an empty set, then each node is added to the attack set sequentially with the highest utility over budget. Then, we run Algorithm 1 again with K_2 iterations ($K_1 < K_2$) to obtain the perturbation $\boldsymbol{U}$ and the adversarial sample $\boldsymbol{X}'$. Since Algorithm 2 queries the black-box system $2K_1N$ times in

Algorithm 2 KG-SPSA for the Diffusion Attack on Traffic Prediction Models

Input: Traffic prediction model $f(\boldsymbol{X})$, input feature matrix $\boldsymbol{X}$, total budget B, iterations $K_1 < K_2$, and cost of each node $\{b_i\}_{i\in\mathcal{V}}$.

Output: Adversarial sample $\boldsymbol{X}'$, optimal perturbation $\boldsymbol{U}$, and attack set $\mathcal{P}$.

Initialize $\mathcal{P} = \emptyset$.
for $i \in \mathcal{V}$ **do**
  Set $\mathcal{P} = \{i\}$ and run Algorithm 1 with K_1 iterations, return perturbation $\boldsymbol{U}_i$ and adversarial sample $\boldsymbol{X}'_i$.
  Set $\hat{\phi}_i = \phi_i(\boldsymbol{U}_i)$.
end for
Initialize $\mathcal{P} = \emptyset$.
while $\sum_{i\in\mathcal{P}} b_i \le B$ **do**
  Set $\texttt{max\_utility} = -\infty, \texttt{max\_idx} = -\infty$.
  for $i \in \mathcal{V} \setminus \mathcal{P}$ **do**
    if $\frac{\hat{\phi}_i}{b_i} > \texttt{max\_utility}$ **then**
      Set $\texttt{max\_utility} = \frac{\hat{\phi}_i}{b_i}$.
      Set $\texttt{max\_idx} = i$.
    end if
  end for
  if $\sum_{i\in\mathcal{P}} b_i + b_{\texttt{max\_idx}} \le B$ **then**
    $\mathcal{P} = \mathcal{P} \cup \{\texttt{max\_idx}\}$
  end if
end while
Run Algorithm 1 with fixed $\mathcal{P}$ and K_2 iterations, obtain $\boldsymbol{X}'$, $\boldsymbol{U}$.
Return $\boldsymbol{X}'$, $\boldsymbol{U}$, $\mathcal{P}$.

step 1 and $2K_2$ times in step 2, the total time complexity is $O(K_1 N + K_2)$.

The entire procedure is referred to as KG-SPSA, and details of the algorithm are presented in Algorithm 2.

It is possible to adopt other methods to estimate $\hat{\phi}_i$ in step 1, including clustering methods and graph centrality measures. These algorithms will be viewed as baseline algorithms and compared with KG-SPSA in numerical experiments.

IV. EXPERIMENTS

In this section, we evaluate the performance of the proposed diffusion attack algorithm under different scenarios using real-world data.

A. Experiment Setup

We examine the proposed attack algorithm on three traffic prediction models and four data sets, and details are described as follows.

Traffic Data: We consider four real-world traffic data sets.

1) *LA:* The LA data set contains traffic speed obtained from 207 loop detectors in Los Angeles [31], and the data ranges from 1 March 2012 to 7 March 2012. The average degree of the adjacency matrix is 14. The speed data are collected every 5 min, and the average speed is 58 mile/h. The study area is shown in the upper part of Fig. 4.
2) *HK:* The HK speed data is collected from an open data platform initiated by the Hong Kong government, and overall 179 roads are considered in the Hong Kong island and Kowloon area. The data ranges from 1 May 2020 to 31 May 2020. The average degree of the adjacency matrix is 2. The speed data are collected every 5 min, and the average speed is 45 km/h. The study area is shown in the lower part of Fig. 4.
3) *Bay:* The Bay data set is collected by California Transportation Agencies. The data contains 325 sensors in the Bay Area from 1 January 2017 to 30 June 2017. The average degree of the adjacency matrix is 8 and the average speed is 63 mile/h. The study area is shown in the lower part of Fig. 4.
4) *Seattle:* The Seattle data set is collected from the digital roadway interactive visualization and evaluation network (DRIVE Net) system in 2015 and covers four connected freeways. The data contains 323 sensors and the time interval is 5 min [68]. The average degree of the adjacency matrix is 3 and the average speed is 59 mile/h.

The reason for choosing these data sets is that they are publicly available and widely used in existing research papers, so readers can replicate our experimental results using the provided source codes easily.

Traffic Prediction Models: We evaluate the developed diffusion attack framework on three traffic prediction models: 1) T-GCN [31]; 2) ST-GCN [32]; and 3) A3T-GCN [33], which are all based on GCN structures. For each model, we set $S = 12$ and $T = 1$.

To conduct the comprehensive evaluation, we train four variants of each model, which are the original model, the original model with DROPOUT, DROPNODE, and DROPEDGE regularization, respectively, [69], [70]. For different drop regularization strategies, we set the drop probability to 30%. In Appendix A, the Accuracy and root MSE (RMSE) of each model are shown in Tables III and IV, and both measures are defined in [31]. Overall, the average accuracy of the different prediction models is around 90% in testing data.

Attack Settings: The parameter settings for the diffusion attack models and algorithms, as well as the evaluation criterion, are discussed as follows.

1) *Model Specifications:* In (4), we set the constraint $-z_i \boldsymbol{x}_i \le \boldsymbol{u}_i \le 0.5 z_i \boldsymbol{x}_i$, and $w_i = 1$, which means each node is equally important. The cost is defined as $b_i = \text{Degree}(i) = S_D(i)$, where $S_D(i) = \sum_j \boldsymbol{A}_{ij}$. To fairly compare the attack performance on different data sets, we simulate attacks with the same percentage of the nodes on each graph. We set Percentage $= \{0.05, 0.15, 0.2, 0.25, 0.5\}$ and the budget is generated by $B = \textit{Percentage} \times D$, where $D = \sum_i S_D(i)$ and it is the sum of the degree of the graph. Then the optimal target reduces to $\Phi(\boldsymbol{U}) = \sum_{i\in\mathcal{V}} \phi_i(\boldsymbol{U})$, where $\phi_i(\boldsymbol{U}) = y'_i(\boldsymbol{X}') - y_i(\boldsymbol{X})$. The attack algorithm aims to reduce the predicted speed and we intend to generate virtual "congestion" on the network.

2) *Evaluation of Algorithms:* We use the average attack influence (AAI) and AAI ratio (AAIR) to evaluate the effect of the diffusion attacks. We define

$$\text{AAI} = \frac{1}{N}\sum_{i\in\mathcal{V}}|\phi_i(\boldsymbol{U})|$$

$$\text{AAIR} = \frac{1}{N}\sum_{i\in\mathcal{V}}\left|\frac{\phi_i(\boldsymbol{U})}{y_i(\boldsymbol{X})}\right|$$

where AAI represents the average degradation and AAIR represents the average degradation ratio of the prediction on the entire network, respectively.

3) *SPSA Setting:* For Algorithm 1, $a = 0.9, c = 0.1, \alpha = 0.3, \gamma = 0.1$, and $\eta = (n/10)$. For diffusion attack, $K_2 = 30\,000$; for computing the $\hat{\phi}_i$, $K_1 = 100$. Since there are only few iterations for nodes selection, we set a larger step and stronger randomness with $a = 10, c = 1, \alpha = 0.1, \gamma = 0.1$, and $\eta = (n/2)$.

Baseline Algorithms: Because the diffusion attack is a newly proposed task, there are very few existing methods that can be used as baseline algorithms. In addition to the proposed KG-SPSA approach, we modify and develop eight algorithms for comparison. The major difference of each baseline algorithm lies in how to select the attack set $\mathcal{P}$ and whether to use the KG-$\star$ greedy algorithm. DEGREE selects nodes with highest degree $S_D(i)$. K-MEDOIDS selects nodes by clustering the nodes with geo-location features until reaching the total budget B [71], PAGERANK selects nodes with highest pagerank scores $S_{PR}(i) = [(1-\alpha)/N] + \alpha\sum_{j\in\mathcal{N}_i}([S_{PR}(j)]/|\mathcal{N}_j|)$, where $\alpha = 0.85$, and BETWEENNESS chooses nodes with high betweenness scores $S_{Bw}(i) = \sum_{j\neq i,k\neq i}([\text{Path}_{jk}(i)]/\text{Path}_{jk})$, where Path_{jk} is the number of the shortest path between node j and k, and $\text{Path}_{jk}(i)$ is the number of shortest path that passes node i. The RANDOM algorithm just selects the node randomly until meets the budget limit. SPSA selects the highest $\hat{\phi}_i = \phi_i(\boldsymbol{U}_i)$ by running Algorithm 1 with $\mathcal{P} = \{i\}$, and the greedy algorithm is not used in SPSA. When we use the greedy algorithm, it is also possible to use centrality measures, such as pagerank and betweenness to represent $\hat{\phi}_i$. Different from PAGERANK and BETWEENNESS, which determine $\mathcal{P}$ by the highest scores, KG-PAGERANK and KG-BETWEENNESS approximate $\hat{\phi}_i$ by the pagerank and betweenness scores, followed by running Algorithm 2 with different $\hat{\phi}_i$. We also compare our method with other feature attack methods, including DE [5], Powell's method (a conjugate direction method [72], [73]), and Random search [74]. Additionally, we generate attack sets $\mathcal{P}$ by using gradient descent and conduct attacks by surrogate GCN models.

B. Results of Diffusion Attack

In this section, we present the experimental results. We first verify the diffusion effect by simulating an attack on a single node, then the performance of the proposed attack algorithm is evaluated and compared with baseline methods in different scenarios. Finally, we examine the robustness of the proposed algorithm under different drop regularization strategies.

1) Diffusion Effects of Attacks on Single Node: To demonstrate the diffusion phenomenon and verify Proposition 1, we construct an attack on a single node for the three traffic prediction models, and the attack effect of different hops of neighbors is presented in Fig. 3. As can be seen, the attack mainly influences the ego node and its 1–2-hop neighbors, and the influence will diminish as the number of hops increases. As proven in Proposition 1, for an L-layer GCN model, the diffusion only occurs within L-hop neighbors, which is verified in this experiment. The experimental results also indicate that the influence of a single node attack is localized, and a successful diffusion attack algorithm requires a scattered node selection strategy for a small subset of nodes.

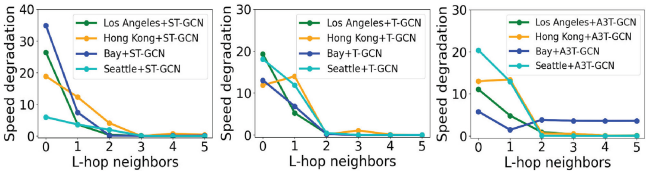


Fig. 3. Diffusion effects when attacking on a single node.

2) Comparisons of Different Attack Algorithms: The proposed algorithm KG-SPSA is compared with different baseline algorithms with *Percentage* = 0.15, and the corresponding AAI is presented in Table I (unit: mile/hour for `LA`, `Bay`, `Seattle`, km/hour for `HK`), and the AAIR table is presented in Appendix B. The attack algorithms are categorized into two types: 1) semi-black-box algorithms that know the graph structure and 2) black-box algorithms that only require inputs and outputs of the prediction models. From Table I one can see, the proposed algorithm KG-SPSA outperforms all the baseline algorithms on `LA`, `HK`, `Seattle`. Comparing with other methods, KG-SPSA is a black-box method, which does not rely on knowledge of the graph structure (i.e., adjacency matrix $\boldsymbol{A}$). In real world, it is challenging to obtain the information of $\boldsymbol{A}$ in the prediction system as the graph can be generated by different configurations, such as sensor layout, network topology, and causal relationship, and this information is hidden to users [35]. Fig. 4 presents the selected nodes by KG-SPSA for different prediction methods with *Percentage* = 0.15, and the color (from green to red) represents the AAIR of each node under the attack. To maximize the attack effect, the selected nodes distribute across the entire network, which is consistent with our previous conjecture.

In addition, it is observed that the robustness of the three prediction models is different. A3T-GCN demonstrates great vulnerability under attack algorithms, while both T-GCN and ST-GCN are more robust. This could be due to the strengthened connection among nodes by the attention layers in A3T-GCN, meanwhile, the strengthened connection can also increase the vulnerability of the prediction models. To further verify the conclusion, we also run the KG-SPSA to attack (with 15% nodes to be attacked) the AR(1) models on each node independently. The result shows that the AAI on the data sets `LA`, `HK`, `Bay`, and `Seattle` are 9.18, 7.54, 9.73, and 7.38, respectively. One can see that the attack influences on the linear models are lower than that on the graph-based models, and this further proves that the joint modeling among nodes by graph-based neural networks makes the prediction models more vulnerable than linear models.

TABLE I
COMPARISON OF DIFFERENT DIFFUSION ATTACK ALGORITHMS IN TERMS OF AAI. (Percentage = 0.15)

Types	Algorithm	LA			HK		
		ST-GCN	T-GCN	A3T-GCN	ST-GCN	T-GCN	A3T-GCN
Semi-black-box	DEGREE	9.62	7.39	9.68	6.01	4.41	5.15
	K-MEDOIDS	11.59	11.41	15.47	12.52	6.75	11.57
	PAGERANK	11.57	10.52	13.05	9.2	5.25	7.22
	BETWEENNESS	11.66	10.08	13.49	16.93	8.55	20.42
	KG-BETWEENNESS	13.22	11.93	15.73	18.6	9.76	30.55
	KG-PAGERANK	19.61	24.11	51.15	23.82	14.46	35.59
Black-box	SURROGATE MODEL	11.35	10.07	18.66	19.92	12.26	34.26
	SPSA	15.23	14.77	33.98	21.88	9.95	30.95
	RANDOMSEARCH	7.27	7.89	11.34	12.21	6.20	18.20
	DE	13.99	16.56	36.68	18.46	12.61	31.49
	POWELL	6.48	7.59	10.68	13.02	4.98	5.30
	KG-SPSA	**20.83**	**26.49**	**65.95**	**26.13**	**17.25**	**37.31**
Types	Algorithm	Bay			Seattle		
		ST-GCN	T-GCN	A3T-GCN	ST-GCN	T-GCN	A3T-GCN
Semi-black-box	DEGREE	4.83	4.5	34.8	9.41	4.76	10.51
	K-MEDOIDS	6.14	6.19	59.77	8.2	5.87	26.47
	PAGERANK	6.14	5.16	42.47	9.61	4.86	10.01
	BETWEENNESS	7.34	6.34	61.2	10.93	5.45	15.19
	KG-BETWEENNESS	9.07	7.18	68.04	10.9	5.56	16.02
	KG-PAGERANK	**14.14**	10.81	89.85	13.22	6.81	22.79
Black-box	SURROGATE MODEL	10.92	4.57	48.28	8.83	5.49	14.9
	SPSA	10.61	9.41	114.63	12.39	6.02	19.32
	RANDOMSEARCH	5.45	4.61	40.14	4.14	2.80	3.66
	DE	10.04	8.31	67.38	8.58	5.22	15.21
	POWELL	3.59	1.93	40.03	4.47	2.51	4.55
	KG-SPSA	13.88	**11.59**	**210.9**	**20.74**	**7.93**	**35.12**

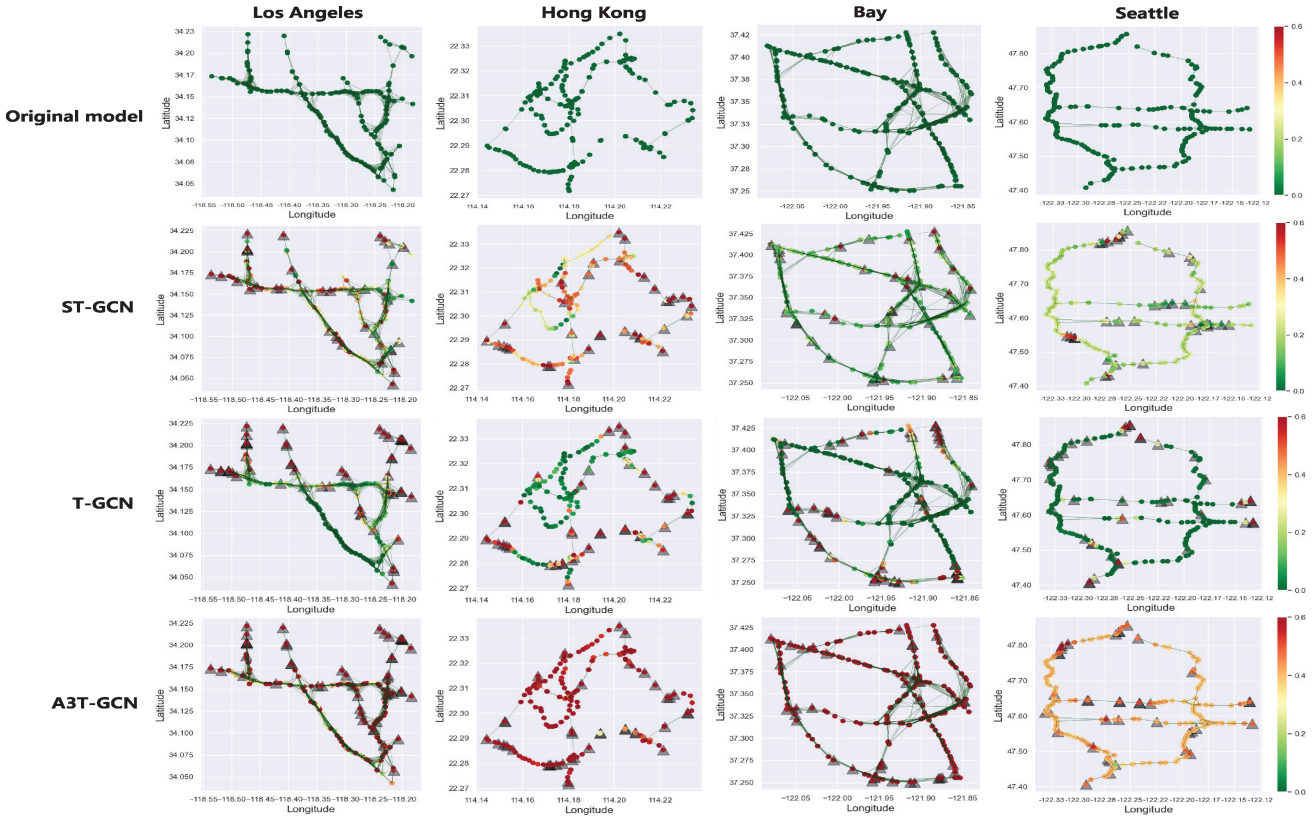


Fig. 4. Distribution of selected nodes and AAIR of each node under the attack algorithm KG-SPSA. (The selected nodes are marked as triangles, and the color represents AAIR).

3) Sensitivity Analysis on the Budget B (Percentage of Attacked Nodes): To study the effect of budget B on the diffusion attack, we run KG-SPSA with $B \in \{0.05, 0.15, 0.2, 0.25, 0.5\} \times D$ for both LA, HK, Bay, and Seattle, and the corresponding AAI is presented in Fig. 5. The attack influence will increase when the total percentage

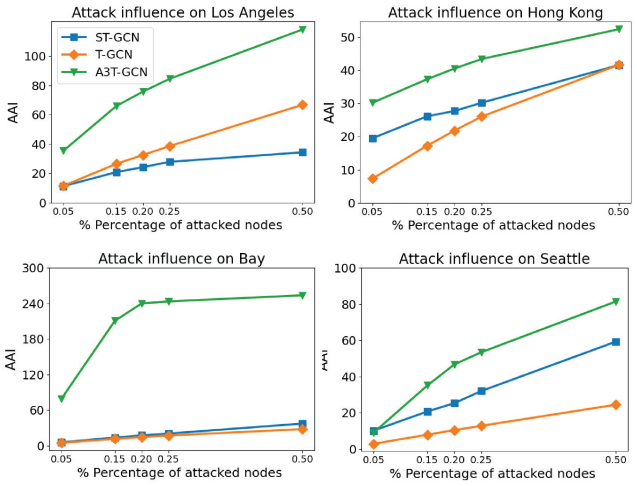


Fig. 5. Attack effects with different percentages of the chosen nodes.

increases for both data sets while the trend is becoming marginal. It is observed that A3T-GCN is the least robust predictions models for both data sets.

4) Performance of the Proposed Algorithm on Drop Regularization Strategies: To better understand the performance of the proposed attack algorithms, we examine the attack effects on the prediction models with drop regularization. Existing studies have widely demonstrated that drop regularization strategies could improve the robustness of the GCN-based model [69], [70], hence it is crucial to show that the performance of the proposed methods remains effective under different drop regularization strategies. To this end, we conduct diffusion attacks with KG-SPSA on the prediction models trained with DROPOUT, DROPNODE, and DROPEGDE. DROPOUT randomly drops rows in feature matrix $\boldsymbol{X}$, DROPNODE randomly drops a subset of the nodes on the graph, and DROPEGDE will randomly drop the edges of the graph for each epoch during the model training. Details of the models are presented in Section IV-A. We run KG-SPSA and KG-PAGERANK for prediction models with different drop regularization on the four data sets, and the algorithm performance is presented in Table II. The reason we choose KG-SPSA and KG-PAGERANK is because both algorithms outperform other semi-black-box and black-box algorithms. For data set LA and HK, KG-SPSA outperforms KG-PAGERANK on all the prediction models, and KG-SPSA is slightly better on Bay and Seattle. In most cases, DROPOUT could degrade the performance of the attack algorithms, while the other two drop regularization strategies do not protect the prediction models. Overall, the proposed diffusion attack algorithms could still generate adversarial samples under various drop regularization strategies.

TABLE II
COMPARISON OF KG-SPSA AND KG-PAGERANK ON THE THREE DEFENSE STRATEGIES IN TERMS OF AAI. (PERCENTAGE = 0.15)

Datasets	Model	Baseline	DROPOUT	DROPNODE	DROPEDGE
KG-SPSA					
LA	ST-GCN	**20.83**	**12.95**	**18.03**	**28.01**
	T-GCN	**26.49**	**19.39**	**19.79**	**15.60**
	A3T-GCN	**65.95**	**16.92**	**50.34**	**31.30**
HK	ST-GCN	**26.13**	**17.14**	**20.43**	**28.20**
	T-GCN	**17.25**	**16.01**	**16.00**	**15.96**
	A3T-GCN	**37.31**	**47.97**	**84.48**	**94.86**
Bay	ST-GCN	13.88	**9.92**	11.82	**16.65**
	T-GCN	**11.59**	**10.28**	**10.64**	**10.79**
	A3T-GCN	**210.9**	**47.96**	**107.72**	**92.59**
Seattle	ST-GCN	**20.74**	**11.31**	**9.99**	**13.66**
	T-GCN	**7.93**	**7.09**	**8.01**	**6.94**
	A3T-GCN	**35.12**	**18.50**	**119.49**	23.58
KG-PAGERANK					
LA	ST-GCN	19.61	10.99	15.56	25.60
	T-GCN	24.11	17.72	18.27	14.29
	A3T-GCN	51.15	15.50	38.51	22.68
HK	ST-GCN	23.82	13.98	17.29	26.51
	T-GCN	14.46	13.69	13.70	13.39
	A3T-GCN	35.59	30.33	29.44	81.31
Bay	ST-GCN	**14.14**	9.49	**13.95**	8.53
	T-GCN	10.81	9.57	9.28	9.96
	A3T-GCN	89.85	38.70	104.84	92.58
Seattle	ST-GCN	13.22	8.26	7.46	10.30
	T-GCN	6.81	5.54	7.12	5.74
	A3T-GCN	22.79	9.13	44.90	**28.30**

TABLE III
ACCURACY OF TRAINED MODELS

Dataset	Model	Baseline	DROPOUT	DROPNODE	DROPEDGE
LA	ST-GCN	92.68%	92.70%	92.77%	88.18%
	T-GCN	90.44%	89.24%	90.01%	90.05%
	A3T-GCN	89.04%	72.71%	89.15%	89.84%
HK	ST-GCN	92.88%	93.20%	93.17%	92.80%
	T-GCN	88.50%	87.70%	86.78%	88.08%
	A3T-GCN	89.47%	87.30%	80.57%	88.12%
Bay	ST-GCN	97.46%	94.51%	95.75%	97.20%
	T-GCN	96.88%	96.20%	96.85%	96.97%
	A3T-GCN	96.70%	92.94%	96.38%	95.34%
Seattle	ST-GCN	91.41%	92.40%	92.34%	92.55%
	T-GCN	90.43%	90.20%	90.62%	90.45%
	A3T-GCN	90.39%	90.41%	89.84%	90.32%

TABLE IV
RMSE OF TRAINED MODELS

Dataset	Model	Baseline	DROPOUT	DROPNODE	DROPEDGE
LA	ST-GCN	4.30	4.28	4.25	6.95
	T-GCN	5.61	6.32	5.86	5.84
	A3T-GCN	6.43	16.02	6.36	5.96
HK	ST-GCN	3.55	3.39	3.41	3.59
	T-GCN	5.74	6.13	6.59	5.95
	A3T-GCN	5.25	6.33	9.68	5.92
Bay	ST-GCN	1.25	2.72	2.11	1.39
	T-GCN	1.55	1.88	1.56	1.50
	A3T-GCN	1.63	3.50	1.79	2.31
Seattle	ST-GCN	3.58	3.17	3.19	3.10
	T-GCN	3.99	4.08	3.91	3.98
	A3T-GCN	4.00	3.99	4.23	4.03

C. Discussions

In this section, we discuss the implications and suggestions for improving the robustness of the traffic prediction models. In the previous section, we carry out numerical experiments to demonstrate the performance of the proposed attack algorithms on different data sets, prediction models, and regularization strategies. Based on the experimental results, we provide the following suggestions to improve the model robustness during different phrases.

1) *Model Selection:* When choosing GCN-based models for speed prediction, RNN-based models are generally more robust than attention-based models. Depending on the data and city scale, it is suggested to choose

TABLE V
COMPARISON OF DIFFERENT DIFFUSION ATTACK ALGORITHMS IN TERMS OF AAIR. (Percentage = 0.15)

Types	Algorithm	LA			HK		
		ST-GCN	T-GCN	A3T-GCN	ST-GCN	T-GCN	A3T-GCN
Semi-black-box	DEGREE	16.03%	18.30%	27.30%	14.20%	10.49%	12.48%
	K-MEDOIDS	19.30%	22.94%	34.21%	26.09%	14.28%	26.22%
	PAGERANK	18.72%	20.25%	29.64%	19.45%	12.06%	16.55%
	BETWEENNESS	20.07%	21.80%	30.31%	32.20%	15.91%	44.99%
	KG-BETWEENNESS	22.43%	25.66%	37.98%	35.34%	17.59%	64.20%
	KG-PAGERANK	32.31%	43.08%	100.00%	44.62%	26.34%	74.55%
Black-box	SURROGATE	18.78%	22.94%	40.58%	38.66%	25.93%	73.55%
	SPSA	25.63%	28.12%	68.47%	41.22%	20.08%	65.08%
	RANDOMSEARCH	11.82%	13.61%	18.66%	22.72%	10.47%	38.44%
	DE	22.76%	29.08%	69.78%	34.58%	22.40%	65.51%
	POWELL	10.40%	13.35%	18.58%	25.20%	9.74%	11.07%
	KG-SPSA	**33.91%**	**45.38%**	**122.27%**	**46.65%**	**27.70%**	**75.82%**
Types	Algorithm	Bay			Seattle		
		ST-GCN	T-GCN	A3T-GCN	ST-GCN	T-GCN	A3T-GCN
Semi-black-box	DEGREE	7.65%	7.44%	54.35%	18.09%	8.79%	50.79%
	K-MEDOIDS	9.57%	10.06%	93.30%	15.89%	10.24%	94.61%
	PAGERANK	9.57%	8.41%	66.32%	18.38%	9.00%	50.52%
	BETWEENNESS	11.29%	10.08%	95.53%	20.83%	10.28%	55.77%
	KG-BETWEENNESS	13.93%	11.39%	106.19%	20.86%	10.48%	58.87%
	KG-PAGERANK	**21.82%**	17.34%	140.28%	27.29%	12.20%	83.98%
Black-box	SURROGATE	16.90%	7.27%	75.06%	17.79%	10.18%	57.08%
	SPSA	16.55%	15.19%	179.04%	26.56%	11.51%	62.19%
	RANDOMSEARCH	8.35%	7.37%	62.59%	8.56%	5.09%	12.09%
	DE	15.44%	13.32%	105.22%	17.61%	9.47%	53.11%
	POWELL	5.51%	3.04%	62.50%	8.61%	4.52%	8.42%
	KG-SPSA	21.54%	**18.49%**	**329.07%**	**41.13%**	**13.48%**	**117.24%**

models with simpler layers, as the complex layers in ST-GCN and A3T-GCN can sometimes degrade significantly under attacks. There is a tradeoff between accuracy and robustness, so it is critical to balance the accuracy and robustness for practical usage.

2) *Model Regularization:* Based on the experimental results, it is suggested to adopt DROPOUT during the training, as the model accuracy remains high while the robustness can be improved after the DROPOUT training. It is also suggested to test different drop regularization strategies before the actual deployment.
3) *Model Privacy:* The graph structure should not be disclosed to the public, as it can significantly improve the efficiency of the attack algorithms. To be specific, it is suggested to not disclose the neural network structure and training data used in the traffic prediction system. It is also suggested to frequently update the prediction model, as the attack models rely on multiple trials and errors on the prediction models. If the prediction model updates frequently, then the robustness of the entire prediction system can be significantly improved.
4) *Active Defending Strategies:* Before actual deployment, it is necessary to comprehensively test the vulnerability of the prediction models and to identify the critical nodes with significant attack influence. For those important nodes, we can enhance the protection by regular patrol in the physical world and consistency checking in the cyber system. For example, if an RSU on a road segment is identified to be critical, then this device should be protected physically [75]. If the attack on this device indeed occurs, the traffic center should spot the anomaly in real time and block the information sent from this device.

V. CONCLUSION

In this article, we explore the robustness and vulnerability issues of graph-based neural network models for the traffic prediction problem. Different from existing graph adversarial attack tasks, adversarial attacks for the traffic prediction problem require to degrade the performance of a spatio-temporal graph-based deep learning model, which is dynamic and more complex. Given this, we propose a novel concept of diffusion attack, which reduces the prediction accuracy of the whole traffic network by perturbing a small number of nodes. To solve for the diffusion attack task, we develop an algorithm KG-SPSA, which consists of the perturbation generation and node selection strategies. The experimental results indicate that the diffusion phenomenon is localized and the combined effect of attack nodes is relatively small. Thus, the optimization problem is approximately separable and can be solved by the greedy algorithm. The results also show that our KG-SPSA algorithm outperforms the baseline algorithms under various scenarios, which demonstrates the effectiveness and efficiency of the proposed method. In addition, the proposed attack algorithms can still generate effective adversarial samples for the traffic prediction models trained with drop regularization. This study could help the public agencies and private sectors better understand the robustness and vulnerability of GCN-based traffic prediction models under adversarial attacks, and strategies to improve the model robustness in different phrases are also discussed. As for the future

TABLE VI
COMPARISON OF KG-SPSA AND KG-PAGERANK ON THE THREE DEFENSE STRATEGIES IN TERMS OF AAIR. (Percentage = 0.15)

Dataset	Model	Baseline	DropOut	DropNode	DropEdge
			KG-SPSA		
LA	ST-GCN	**33.91%**	**21.10%**	**30.23%**	**40.47%**
	T-GCN	**45.38%**	**33.85%**	**33.78%**	**26.39%**
	A3T-GCN	**122.27%**	**39.29%**	**102.36%**	**63.28%**
HK	ST-GCN	**46.65%**	**31.47%**	**34.59%**	**49.71%**
	T-GCN	**27.70%**	25.28%	**25.14%**	**26.64%**
	A3T-GCN	**75.82%**	**92.17%**	**199.48%**	**236.63%**
Bay	ST-GCN	21.54%	**16.44%**	19.03%	**25.76%**
	T-GCN	**18.49%**	**16.34%**	**16.62%**	**17.02%**
	A3T-GCN	**329.07%**	**77.20%**	**172.11%**	**146.32%**
Seattle	ST-GCN	**41.13%**	**21.23%**	**17.76%**	**18.19%**
	T-GCN	**13.48%**	**11.74%**	**13.67%**	**11.64%**
	A3T-GCN	**117.24%**	**134.50%**	**230.54%**	46.22%
			KG-PAGERANK		
LA	ST-GCN	32.31%	18.06%	27.02%	38.51%
	T-GCN	43.08%	32.52%	32.62%	25.42%
	A3T-GCN	100.00%	36.01%	78.11%	47.30%
HK	ST-GCN	44.62%	26.56%	31.55%	48.14%
	T-GCN	26.34%	**25.32%**	24.42%	24.29%
	A3T-GCN	74.55%	56.08%	71.00%	190.35%
Bay	ST-GCN	**21.82%**	15.78%	**22.50%**	13.26%
	T-GCN	17.34%	15.25%	14.60%	15.81%
	A3T-GCN	140.28%	62.49%	167.52%	146.30%
Seattle	ST-GCN	27.29%	14.74%	13.11%	13.39%
	T-GCN	12.20%	10.00%	12.78%	10.37%
	A3T-GCN	83.98%	58.28%	87.64%	**54.21%**

research directions, the proposed attack algorithms could be applied to not only road traffic prediction, but also other traffic modes, such as urban railway transit systems, ride-sourcing services, and parking systems [76], [77]. It is also important to study the effect of adversarial attacks on flow prediction, origin-destination demand prediction, and other tasks relying on GCN-based models. It is also necessary to develop an online learning algorithm to address online attack scenarios. Additionally, how the traffic conditions and neural network structures jointly affect the attack effects can be studied in depth. For the users of the traffic prediction models, it is critical to developing models for defending against adversarial attacks and protecting traffic prediction results. Another way to protect the prediction model is through real-time anomaly detection and filtering of the incoming data stream, which could be a new research direction for improving the robustness of the traffic prediction models under adversarial attacks.

SUPPLEMENTARY MATERIALS

The proposed diffusion attack algorithm and evaluation framework are implemented in Python and open-sourced on GitHub (https://github.com/LYZ98/diffusion_attack).

APPENDIX A
MORE DETAILS ABOUT THE THREE TRAFFIC PREDICTION MODELS AND ATTACK RESULTS

To train the three traffic prediction models, we set the learning rate to be 0.001, batch size to be 32, and the number of epochs to be 300. The four data sets are divided into two parts, in which 80% and 20% are training set and testing set, respectively. MSE is used as the loss function [31], and Adam is adopted as the optimizer. The testing accuracy in terms of accuracy and RMSE of the trained prediction models are presented in Tables III and IV, respectively. Overall, all the prediction models could achieve high prediction accuracy on both data sets.

APPENDIX B
COMPARISONS OF DIFFERENT ATTACK ALGORITHMS IN TERMS OF AAIR. (*Percentage* = 0.15)

Similar to Table I, we evaluate the performance of different attack algorithms in terms of AAIR in Table V. In general, similar arguments could be obtained based on the AAIR, and the proposed KG-SPSA outperforms other baseline models in LA, HK, Seattle, and performs similarly to the semi-black-box algorithms in Bay.

Results in Table VI follow the same patterns as in Table II, and one can observe that the proposed attack algorithms can still generate effective adversarial samples in terms of AAIR.

ACKNOWLEDGMENT

The contents of this article reflect the views of the authors, who are responsible for the facts and the accuracy of the information presented herein.

REFERENCES

[1] E. I. Vlahogianni, J. C. Golias, and M. G. Karlaftis, "Short-term traffic forecasting: Overview of objectives and methods," *Transp. Rev.*, vol. 24, no. 5, pp. 533–557, 2004.
[2] O. Lange and L. Perez. "Traffic prediction with advanced graph neural networks." 2020. [Online]. Available: https://deepmind.com/blog/article/traffic-prediction-with-advanced-graph-neural-networks
[3] C. Szegedy et al., "Intriguing properties of neural networks," 2013, *arXiv:1312.6199*.
[4] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," 2014, *arXiv:1412.6572*.
[5] J. Su, D. V. Vargas, and K. Sakurai, "One pixel attack for fooling deep neural networks," *IEEE Trans. Evol. Comput.*, vol. 23, no. 5, pp. 828–841, Oct. 2019.
[6] K. Zhou, T. P. Michalak, M. Waniek, T. Rahwan, and Y. Vorobeychik, "Attacking similarity-based link prediction in social networks," in *Proc. 18th Int. Conf. Auton. Agents MultiAgent Syst.*, 2019, pp. 305–313.
[7] Y. Li, X. Xu, J. Xiao, S. Li, and H. T. Shen, "Adaptive square attack: Fooling autonomous cars with adversarial traffic signs," *IEEE Internet Things J.*, vol. 8, no. 8, pp. 6337–6347, Apr. 2021.
[8] M. Fang, G. Yang, N. Z. Gong, and J. Liu, "Poisoning attacks to graph-based recommender systems," in *Proc. 34th Annu. Comput. Security Appl. Conf.*, 2018, pp. 381–392.
[9] S. Weckert."Google maps hacks." 2020. [Online]. Available: http://www.simonweckert.com/googlemapshacks.html
[10] A. Bayen, J. Butler, and A. D. Patire, "Mobile millennium: GPS mobile phones as traffic probes, California networked traveler—Safe trip 21 phase II," CCIT Res., Univ. California, Berkeley, CA, USA, Rep. UCB-ITS-CWP-2011-6, 2011.
[11] M. T. Ahvanooey, Q. Li, M. Rabbani, and A. R. Rajput, "A survey on smartphones security: Software vulnerabilities, malware, and attacks," 2020, *arXiv:2001.09406*.
[12] N. Lu, N. Cheng, N. Zhang, X. Shen, and J. W. Mark, "Connected vehicles: Solutions and challenges," *IEEE Internet Things J.*, vol. 1, no. 4, pp. 289–299, Aug. 2014.
[13] B. K. J. Al-Shammari, N. Al-Aboody, and H. S. Al-Raweshidy, "IoT traffic management and integration in the QoS supported network," *IEEE Internet Things J.*, vol. 5, no. 1, pp. 352–370, Feb. 2018.
[14] F. Liu, L. M. Moreno, and L. Sun, "One vertex attack on graph neural networks-based spatiotemporal forecasting," in *Proc. ICLR*, 2021, pp. 1–11. [Online]. Available: https://openreview.net/forum?id=W0MKrbVOxtd
[15] Y. J. Edes, P. G. Michalopoulos, and R. A. Plum, "Improved estimation of traffic flow for real-time control," *Transp. Res. Rec.*, vol. 7, no. 9, p. 28, 1980.

[16] M. S. Ahmed and A. R. Cook, "Analysis of freeway traffic time-series data by using Box-Jenkins techniques," in *Proc. Transp. Res. Rec.*, vol. 722, 1979, pp. 1–9.

[17] M. M. Hamed, H. R. Al-Masaeid, and Z. M. B. Said, "Short-term prediction of traffic volume in urban arterials," *J. Transp. Eng.*, vol. 121, no. 3, pp. 249–254, 1995.

[18] M. Van Der Voort, M. Dougherty, and S. Watson, "Combining Kohonen maps with ARIMA time series models to forecast traffic flow," *Transp. Res. C, Emerg. Technol.*, vol. 4, no. 5, pp. 307–318, 1996.

[19] S. Lee and D. B. Fambro, "Application of subset autoregressive integrated moving average model for short-term freeway traffic volume forecasting," *Transp. Res. Rec.*, vol. 1678, no. 1, pp. 179–188, 1999.

[20] B. M. Williams and L. A. Hoel, "Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: Theoretical basis and empirical results," *J. Transp. Eng.*, vol. 129, no. 6, pp. 664–672, 2003.

[21] C.-H. Wu, J.-M. Ho, and D. T. Lee, "Travel-time prediction with support vector regression," *IEEE Trans. Intell. Transp. Syst.*, vol. 5, no. 4, pp. 276–281, Dec. 2004.

[22] F. G. Habtemichael and M. Cetin, "Short-term traffic flow rate forecasting based on identifying similar traffic patterns," *Transp. Res. C, Emerg. Technol.*, vol. 66, pp. 61–78, May 2016.

[23] I. Okutani and Y. J. Stephanedes, "Dynamic prediction of traffic volume through Kalman filtering theory," *Transp. Res. B, Methodol.*, vol. 18, no. 1, pp. 1–11, 1984.

[24] C. P. I. J. Van Hinsbergen, T. Schreiter, F. S. Zuurbier, J. W. C. Van Lint, and H. J. Van Zuylen, "Localized extended Kalman filter for scalable real-time traffic state estimation," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 1, pp. 385–394, Mar. 2012.

[25] X. Ma, Z. Dai, Z. He, J. Ma, Y. Wang, and Y. Wang, "Learning traffic as images: A deep convolutional neural network for large-scale transportation network speed prediction," *Sensors*, vol. 17, no. 4, p. 818, 2017.

[26] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[27] A. Azzouni and G. Pujolle, "A long short-term memory recurrent neural network framework for network traffic matrix prediction," 2017, *arXiv:1705.05690*.

[28] N. Ramakrishnan and T. Soni, "Network traffic prediction using recurrent neural networks," in *Proc. 17th IEEE Int. Conf. Mach. Learn. Appl.*, 2018, pp. 187–193.

[29] S. Guo, Y. Lin, N. Feng, C. Song, and H. Wan, "Attention based spatial-temporal graph convolutional networks for traffic flow forecasting," in *Proc. 33rd AAAI Conf. Artif. Intell.*, 2019, pp. 922–929.

[30] F. Zhou, Q. Yang, K. Zhang, G. Trajcevski, T. Zhong, and A. Khokhar, "Reinforced spatiotemporal attentive graph neural networks for traffic forecasting," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 6414–6428, Jul. 2020.

[31] L. Zhao et al., "T-GCN: A temporal graph convolutional network for traffic prediction," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 9, pp. 3848–3858, Sep. 2020.

[32] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, 2018, pp. 3634–3640.

[33] J. Zhu, Y. Song, L. Zhao, and H. Li, "A3T-GCN: Attention temporal graph convolutional network for traffic forecasting," 2020, *arXiv:2006.11583*.

[34] B. Yu, Y. Lee, and K. Sohn, "Forecasting road traffic speeds by considering area-wide spatio-temporal dependencies based on a graph convolutional neural network (GCN)," *Transp. Res. C, Emerg. Technol.*, vol. 114, pp. 189–204, May 2020.

[35] J. Ye, J. Zhao, K. Ye, and C. Xu, "How to build a graph-based deep learning architecture in traffic domain: A survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 5, pp. 3904–3924, May 2022.

[36] D. Wang, J. Zhang, W. Cao, J. Li, and Y. Zheng, "When will you arrive? Estimating travel time based on deep neural networks," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, p. 8.

[37] L. Bai, L. Yao, S. S. Kanhere, X. Wang, and Q. Z. Sheng, "STG2seq: Spatial-temporal graph to sequence model for multi-step passenger demand forecasting," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, 2019, pp. 1981–1987.

[38] L. Liu, J. Chen, H. Wu, J. Zhen, G. Li, and L. Lin, "Physical-virtual collaboration modeling for intra- and inter-station metro ridership prediction," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 4, pp. 3377–3391, Apr. 2022.

[39] L. Lin, Z. He, and S. Peeta, "Predicting station-level hourly demand in a large-scale bike-sharing network: A graph convolutional neural network approach," *Transp. Res. C, Emerg. Technol.*, vol. 97, pp. 258–276, Dec. 2018.

[40] J. Yang, B. Guo, Z. Wang, and Y. Ma, "Hierarchical prediction based on network-representation-learning-enhanced clustering for bike-sharing system in smart city," *IEEE Internet Things J.*, vol. 8, no. 8, pp. 6416–6424, Apr. 2021.

[41] K. F. Chu, A. Y. S. Lam, and V. O. K. Li, "Deep multi-scale convolutional LSTM network for travel demand and origin-destination predictions," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 8, pp. 3219–3232, Aug. 2020.

[42] A. Monti, A. Bertugli, S. Calderara, and R. Cucchiara, "DAG-Net: Double attentive graph neural network for trajectory forecasting," in *Proc. 25th Int. Conf. Pattern Recognit.*, 2021, pp. 2551–2558.

[43] A. Mohamed, K. Qian, M. Elhoseiny, and C. Claudel, "Social-STGCNN: A social spatio-temporal graph convolutional neural network for human trajectory prediction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 14424–14432.

[44] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, "Geometric deep learning: Going beyond Euclidean data," *IEEE Signal Process. Mag.*, vol. 34, no. 4, pp. 18–42, Jul. 2017.

[45] X. Ma, Z. Tao, Y. Wang, H. Yu, and Y. Wang, "Long short-term memory neural network for traffic speed prediction using remote microwave sensor data," *Transp. Res. C, Emerg. Technol.*, vol. 54, pp. 187–197, May 2015.

[46] J. Dai, W. Zhu, and X. Luo, "A targeted universal attack on graph convolutional network by using fake nodes," *Neural Process. Lett.*, vol. 54, pp. 3321–3337, Mar. 2022.

[47] D. Zügner and S. Günnemann, "Adversarial attacks on graph neural networks via meta learning," 2019, *arXiv:1902.08412*.

[48] H. Dai et al., "Adversarial attack on graph structured data," in *Proc. 35th Int. Conf. Mach. Learn.*, 2018, pp. 1115–1124.

[49] J. Ma, S. Ding, and Q. Mei, "Towards more practical adversarial attacks on graph neural networks," in *Proc. 34th Int. Conf. Neural Inf. Process. Syst.*, 2020, p. 11.

[50] J. Ma, J. Deng, and Q. Mei, "Adversarial attack on graph neural networks as an influence maximization problem," in *Proc. 15th ACM Int. Conf. Web Search Data Min.*, 2022, pp. 675–685.

[51] K. Xu et al., "Topology attack and defense for graph neural networks: An optimization perspective," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, 2019, pp. 3961–3967.

[52] X. Xu, X. Du, and Q. Zeng, "Attacking graph-based classification without changing existing connections," in *Proc. 37th Annu. Comput. Security Appl. Conf.*, 2020, pp. 951–962.

[53] B. Finkelshtein, C. Baskin, E. Zheltonozhskii, and U. Alon, "Single-node attacks for fooling graph neural networks," *Neurocomputing*, vol. 513, pp. 1–12, Nov. 2022.

[54] B. Wang et al., "Efficient evasion attacks to graph neural networks via influence function," 2020, *arXiv:2009.00203*.

[55] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," in *Proc. IEEE Eur. Symp. Security Privacy*, 2016, pp. 372–387.

[56] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *Proc. IEEE Symp. Security Privacy*, 2017, pp. 39–57.

[57] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "DeepFool: A simple and accurate method to fool deep neural networks," in *Proc. 29th IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 2574–2582.

[58] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, "Practical black-box attacks against machine learning," in *Proc. ACM Asia Conf. Comput. Commun. Security*, 2017, pp. 506–519.

[59] W. Brendel, J. Rauber, and M. Bethge, "Decision-based adversarial attacks: Reliable attacks against black-box machine learning models," 2017, *arXiv:1712.04248*.

[60] P.-Y. Chen, H. Zhang, Y. Sharma, J. Yi, and C.-J. Hsieh, "Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models," in *Proc. 10th ACM Workshop Artif. Intell. Security*, 2017, pp. 15–26.

[61] C.-C. Tu et al., "AutoZOOM: Autoencoder-based zeroth order optimization method for attacking black-box neural networks," in *Proc. 33rd AAAI Conf. Artif. Intell.*, 2019, pp. 742–749.

[62] A. Ilyas, L. Engstrom, A. Athalye, and J. Lin, "Black-box adversarial attacks with limited queries and information," in *Proc. 35th Int. Conf. Mach. Learn.*, 2018, pp. 2137–2146.

[63] N. Akhtar and A. Mian, "Threat of adversarial attacks on deep learning in computer vision: A survey," *IEEE Access*, vol. 6, pp. 14410–14430, 2018.

[64] J. C. Spall, "An overview of the simultaneous perturbation method for efficient optimization," in *Johns Hopkins Apl Tech. Dig.*, vol. 19, 1998, pp. 482–492.
[65] J. Uesato, B. O'Donoghue, P. Kohli, and A. van den Oord, "Adversarial risk and the dangers of evaluating against weak attacks," in *Proc. 35th Int. Conf. Mach. Learn.*, 2018, pp. 5025–5034.
[66] S. Martello, D. Pisinger, and P. Toth, "Dynamic programming and strong bounds for the 0-1 knapsack problem," *Manage. Sci.*, vol. 45, no. 3, pp. 414–424, 1999.
[67] B. Zhang and Z. Hua, "A unified method for a class of convex separable nonlinear knapsack problems," *Eur. J. Oper. Res.*, vol. 191, no. 1, pp. 1–6, 2008.
[68] Z. Cui, R. Ke, Z. Pu, and Y. Wang, "Deep bidirectional and unidirectional LSTM recurrent neural network for network-wide traffic speed prediction," 2018, *arXiv:1801.02143*.
[69] Y. Rong, W. Huang, T. Xu, and J. Huang, "DropEdge: Towards deep graph convolutional networks on node classification," in *Proc. 8th Int. Conf. Learn. Represent.*, 2020, pp. 1–18.
[70] L. Chen, X. Li, and D. Wu, "Enhancing robustness of graph convolutional networks via dropping graph connections," in *Proc. Eur. Conf. Mach. Learn. Knowl. Disc. Databases*, 2020, pp. 412–428.
[71] N. K. Kaur, U. Kaur, and D. D. Singh, "K-medoid clustering algorithm—A review," *Int. J. Comput. Appl. Technol.*, vol. 1, no. 1, pp. 42–45, 2014.
[72] M. J. Powell, "An efficient method for finding the minimum of a function of several variables without calculating derivatives," *Comput. J.*, vol. 7, no. 2, pp. 155–162, 1964.
[73] L. Meunier, J. Atif, and O. Teytaud, "Yet another but more efficient black-box adversarial attack: Tiling and evolution strategies," 2019, *arXiv:1910.02244*.
[74] M. Andriushchenko, F. Croce, N. Flammarion, and M. Hein, "Square attack: A query-efficient black-box adversarial attack via random search," in *Proc. 16th Eur. Conf. Comput. Vis.*, 2020, pp. 484–501.
[75] J. Zhang, Y. Wang, S. Li, and S. Shi, "An architecture for IoT-enabled smart transportation security system: A geospatial approach," *IEEE Internet Things J.*, vol. 8, no. 8, pp. 6205–6213, Apr. 2021.
[76] S. Yang, W. Ma, X. Pi, and S. Qian, "A deep learning approach to real-time parking occupancy prediction in transportation networks incorporating multiple spatio-temporal data sources," *Transp. Res. C, Emerg. Technol.*, vol. 107, pp. 248–265, Oct. 2019.
[77] J. Zhang, H. Che, F. Chen, W. Ma, and Z. He, "Short-term origin-destination demand prediction in urban rail transit systems: A channel-wise attentive split-convolutional neural network method," *Transp. Res. C, Emerg. Technol.*, vol. 124, Mar. 2021, Art. no. 102928.

Lyuyi Zhu received the bachelor's degree in civil engineering from Zhejiang University, Hangzhou, China, in 2021. He is currently pursuing the Ph.D. degree with the School of Data Science, City University of Hong Kong, Hong Kong.

His research interest includes machine learning, optimization, and numerical method.

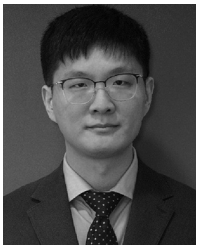
Kairui Feng received the bachelor's degree in civil engineering and mathematics from Tsinghua University, Beijing, China, in 2017, and the Ph.D. degree in civil and environmental engineering from Princeton University, Princeton, New Jersey, USA, in 2023, where he is currently pursuing the Postdoctoral degree with the Department of Civil and Environmental Engineering.

His research interest includes infrastructure system modeling/optimization and climate change using data-driven and numerical approaches.

Ziyuan Pu (Member, IEEE) received the B.S. degree from Southeast University, Nanjing, China, in 2010, and the M.S. and Ph.D. degrees from the University of Washington, Seattle, WA, USA, in 2015 and 2020, respectively.

He is currently a Professor with Southeast University and an Adjunct Senior Research Fellow with Monash University, Melbourne, VIC, Australia. His previous research focused on intelligent transportation systems, traffic sensing, intelligent vehicles, and smart road infrastructures.

Prof. Pu is the recipient of several prestigious awards, including the Outstanding Technical Paper Award presented by ITE Western District in 2022, the Excellence in Research Award (Early Career) presented by Monash University, and the Excellence in Highway Safety Data Research Award presented by FHWA and ITE in 2020. He serves as an Associate Editor of IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS and a Handling Editor of *Transportation Research Record*. He serves as a member of CAV Impacts Committee and AI Committee of ASCE T&DI, the Transportation Research Board Standing Committee on Information Systems and Technology (AED30) and Standing Committee on Artificial Intelligence and Advanced Computing Applications (AED50).



Wei Ma (Member, IEEE) received the bachelor's degree in civil engineering and mathematics from Tsinghua University, Beijing, China, in 2014, and the master's degree in machine learning and civil and environmental engineering and the Ph.D. degree in civil and environmental engineering from Carnegie Mellon University, Pittsburgh, PA, USA, in 2019.

He is currently an Assistant Professor with the Department of Civil and Environmental Engineering, The Hong Kong Polytechnic University, Hong Kong. His research focuses on intersection of machine learning, data mining, and transportation network modeling, with applications for smart and sustainable mobility systems.

Dr. Ma has received the 2020 Mao Yisheng Outstanding Dissertation Award, the Best Paper Award (theoretical track) at INFORMS Data Mining and Decision Analytics Workshop, and the 2022 Kikuchi Karlaftis Best Paper Award of the TRB Committee on Artificial Intelligence and Advanced Computing Applications.