Title

Towards Efficient and Secure Intelligent Transportation Services: AI-driven Traffic Light Controller and Privacy-Preserving Mobility Data Generation

Author

Haydari, Ammar

Publication Date

2024

Peer reviewed|Thesis/dissertation

Towards Efficient and Secure Intelligent Transportation Services: AI-driven Traffic Light Controller and Privacy-Preserving Mobility Data Generation

By

AMMAR HAYDARI

DISSERTATION

Submitted in partial satisfaction of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in

Electrical and Computer Engineering

in the

OFFICE OF GRADUATE STUDIES

of the

UNIVERSITY OF CALIFORNIA

DAVIS

Approved:

Chen-Nee Chuah, Chair

Michael Zhang, Co-Chair

Dipak Ghosal

Committee in Charge

2024

To my beloved parents, Ahmet and Fatma, whose unwavering support have been my constant source of strength and inspiration, to my dear wife, Sayre, and my son, Hamza, whose presence in my life fills me with boundless joy and gratitude, and to my brothers, Huseyin and Servet, who have always been there for me, I am blessed to have such an amazing family.

Contents

Abstract

The widespread adoption of artificial intelligence (AI) and Intelligent Transportation Systems (ITS) technologies has led to the increasing application of AI-based ITS controllers, with the Traffic Signal Controller (TSC) being a prominent example. Reinforcement learning (RL) models have shown promising results for adaptively adjusting traffic light schedules in urban environments through RL-based TSCs (RL-TSCs). The real-world deployment of RL-TSCs involves three key aspects: performance, security, and data privacy. In terms of performance, RL-TSC models need to be designed with consideration for various metrics, such as fair traffic scheduling and air quality impact. To address this, our approach takes into account a multi-objective constrained learning formulation to optimize performance. However, the use of RL-TSCs for automation, by leveraging external inputs, introduces security concerns that require active research to mitigate. We address these security challenges by introducing an innovative defense mechanism. Additionally, the training of RL-TSCs relies on real-world mobility datasets, necessitating the protection of data privacy at different levels of granularity. To minimize the constraints associated with limited real data availability or privacy concerns, we introduce two distinct directions: synthetic trajectory data generation using recent generative AI methods, and location privacy models for raw mobility datasets based on differential privacy, which safeguard individual trajectories and aggregated mobility datasets. This research provides a valuable tool for evaluating the practical deployment of RL-TSCs, particularly in real-world settings where the last mile of implementation and security is paramount. By addressing the key challenges of performance, security, and data privacy, this work aims to facilitate the successful real-world deployment of AI-powered ITS controllers.

Acknowledgments

I am incredibly grateful and fortunate to have had Professor Chen-Nee Chuah as my advisor, whose vast knowledge and wealth of experience have served as a constant source of inspiration during my academic journey. I will always be grateful for the countless hours she spent reviewing my drafts, providing constructive feedback, and challenging me to push beyond my limits. Beyond her role as an advisor, Professor Chuah has been an invaluable mentor, and I am also deeply grateful for the many wonderful colleagues and collaborators I have had the pleasure of working with during my time in her lab, RubiNet.

Also, I would like to express my gratitude to my co-advisor, Professor Michael Zhang, for his invaluable guidance, support, and encouragement throughout my Ph.D. journey. His expertise, insight, and dedication to intelligent transportation systems have been instrumental in shaping the direction of my research and helping me to navigate the challenges that arise in any scientific pursuit. I am deeply grateful for the opportunity to work with him and for the many valuable lessons and insights he has shared with me along the way. Thank you for your unwavering support and for always pushing me to strive for excellence.

I would like to express my appreciation to Dr. Sean Peisert, Dr. Jane Macfarlane and Professor Dipak Ghosal, for their invaluable contributions to my research. Their insightful comments, constructive feedback, and critical evaluation have been instrumental in shaping the final outcome of this work. Thank you for sharing your expertise, devoting your time, and providing unwavering support in helping me achieve my academic goals.

I would like to extend my deepest thanks to my parents and brothers, who have been perfect role models for me throughout my academic journey. Their tireless support, encouragement, and guidance have been instrumental in shaping the person I am today. They have provided me with the freedom to pursue my interests and passions, and I am grateful for every opportunity they have given me. Thank you for your unconditional love and support and for the countless sacrifices you have made to help me achieve my goals.

I am also grateful for the love and support of my loving wife Sayre and our wonderful son Hamza. Throughout this challenging journey of completing my thesis, you have been my unwavering source of support, inspiration, and motivation. Your presence has brought balance and joy to my life, and

I am grateful for the sacrifices you have made to support me on this journey. This achievement is as much yours as it is mine, and I look forward to creating a brighter future together as a family. I am blessed to have you in my life.

Last but not the least, I would also like to thank my lab mate Dr. Zhengfeng (Jeff) Lai, and other colleagues from RubiNet research lab and Dr. Chia-Cheng (Jerry) Yen for our many research discussions and fruitful collaborations.

CHAPTER 1

Introduction

1.1. Overview

Intelligent transport systems (ITS) integrate information and communication technologies (ICT) with transportation applications that increase the efficiency and security of traffic for all the participants, such as pedestrians and vehicles. Latest technological improvements have improved the quality of transportation. New data-driven approaches bring out a new research direction for all control-based systems, e.g., transportation, robotics, IoT, and power systems. Combining data-driven applications with transportation systems plays a key role in recent transportation applications.

Artificial intelligence (AI) tries to control systems with minimal human intervention. AI-based control mechanisms in ITS, such as traffic signal control (TSC) systems, take action based on real-time data from the environment for online updating. There are several reasons why authorities want to implement data-driven autonomous controllers in ITS, such as time-saving for drivers, energy-saving for the environment, and safety for all participants. Coordinated and connected traffic management systems can save travel time with the help of TSCs. Spending more time in traffic increases fuel consumption with environmental and economic impacts. Another reason why human intervention is tried to be minimized using AI-based controllers is the unpredictable nature of human behavior. It is expected that autonomous ITS controllers will decrease traffic accidents and increase the quality of transportation [**48**]. For the above reasons, there is a high demand for various aspects of autonomous and adaptive TSCs in ITS. One popular approach is to use experience-based learning models, similar to human learning.

Reinforcement learning (RL) is conceived to increase the traffic efficiency in ITS by enabling a learning structure that interacts with the environment. While many RL-based traffic optimization methods are presented for different ITS applications, the majority of these applications are concentrated on TSCs. RL-TSCs have the potential to offer a solution by decreasing the travel delay and increasing the traffic efficiency. Nonetheless, security vulnerabilities and possible security mechanisms of RL-based TSC are research questions that must be addressed. A security breach of RL-TSC in the case of either compromising the learning model or participating device such as a vehicle is a challenge.

Training RL-TSCs mainly rely on traffic simulators, which can generate traffic flow using model-based travel demands or real traffic trajectories showing the travel path of a user. These collected trajectories can be used to design a traffic model or used to validate ITS control algorithms such as RL-TSCs. However, the convenience of trajectory datasets comes with a cost of privacy because user trajectories contain personally identifiable information such as home and office addresses. Due to the privacy issue, trajectory datasets are not easily accessible. The datasets should be synthesized or privatized before being made available to the public.

This thesis delves into three critical challenges within the ITS: the performance and security of RL-based TSCs, and the privacy concerns surrounding vehicular mobility datasets. To underscore the rationale for adopting learning-based TSCs, we initiate our exploration by assessing the impact of RL-TSCs on traffic delay and air pollution, drawing insights from a real traffic dataset sourced from downtown San Francisco. Subsequently, we propose a constrained RL model to promote fairness and environmentally responsible RL-TSCs. Furthermore, our investigation extends to the security aspects of RL-TSCs, examining a range of threat models and potential defense mechanisms. Beyond the performance and security challenges of RL-TSCs, this research also addresses the privacy and data availability challenges inherent in vehicular mobility datasets. The privacy constraint has been addressed through the application of differential privacy techniques. To overcome the limited availability of diverse real-world mobility data, we propose using synthetic data generation methods leveraging generative AI models.

1.2. RL-based TSCs

RL is a general learning tool where an agent interacts with the environment to learn how to behave without prior knowledge by learning to maximize a numerically defined reward (or to minimize a penalty). After taking an action, RL agent receives feedback from the environment at each time step t about the performance of its action. Using this feedback (reward or penalty) it iteratively updates its action policy to reach an optimum control policy. RL learns from experiences with the environment, exhibiting a trial-and-error kind of learning, similar to human learning [**194**].

In a general RL model, an agent controlled with an algorithm, observes the system state s_t at each time step t and receives a reward r_t from its environment/system following the action a_t. After taking an action based on the current policy π, the system transitions to the next state s_{t+1}. At every interaction, RL agent updates its knowledge about the environment. Fig 1.1 depicts the schematic of the RL process.

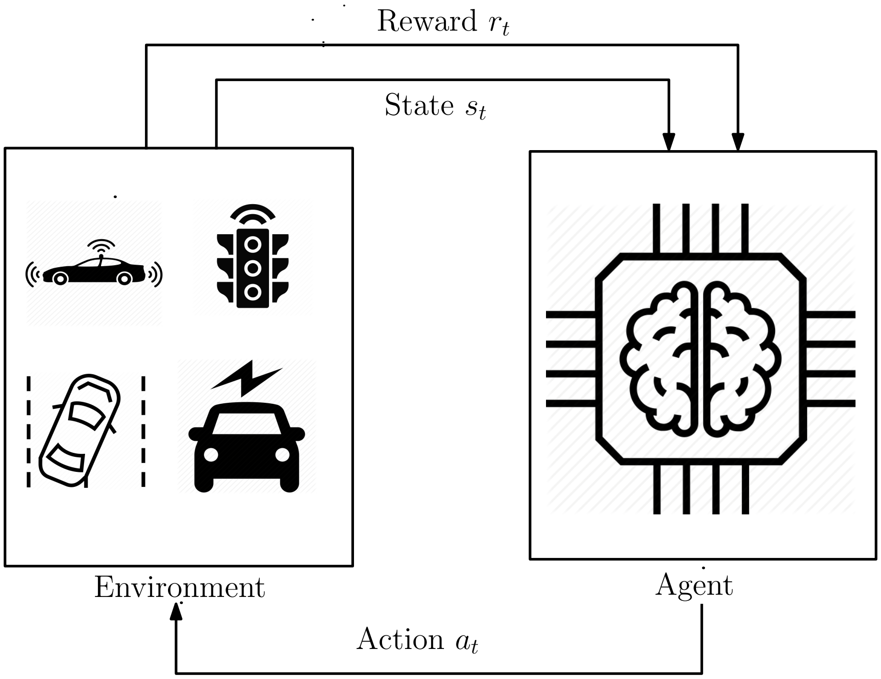


FIGURE 1.1. Reinforcement learning control loop.

Standard RL algorithms cannot efficiently compute the value or policy functions in high-dimensional state spaces. Although some linear function approximation methods are proposed for solving the large state space problem in RL, their capabilities are still up to a certain point. In high-dimensional and complex systems, standard RL approaches cannot learn informative features of the environment. However, this problem can be easily handled by deep learning-based function approximators, in which deep neural networks are trained to learn the optimal policy or value functions. Different neural network structures such as convolutional neural network (CNN) and recurrent neural network (RNN) are used for training RL algorithms in large state spaces [**124**].

In the case of RL-TSCs, RL agent is implemented in the TSC center to control traffic signals adaptive to the traffic flow. First, the control unit collects the state information, which can be in different formats such as queue length, position of vehicles, speed of vehicles etc., and then control unit takes an action based on the current policy of RL method. Finally, the agent (control unit) gets a reward with respect to the action taken. By following these steps, the agent tries to find an optimal policy to minimize the congestion on the intersection.

1.3. Vehicular Mobility Datasets

Vehicular mobility datasets are a great source of information for understanding and extracting knowledge from dynamic human mobility. The mobility datasets can be at different scales from an individual level to the collective level of populations such as trajectories, origin-destination travels, or aggregated level of user movements. Human mobility using real datasets has been studied in many fields, such as geography, transportation, physics, and public health. The data sources of mobility datasets are very broad, such as census data & surveys, mobile devices, GPS devices, and online data collection sources [**13**].

In transportation, different applications require different scales of information. For example, user stay point analysis cannot be performed on an aggregated level because the problem requires analysis of individual user stay points. On the other hand, major route identification or traffic zoning problems can be studied with aggregated mobility datasets since routes can be extracted from aggregated traffic networks.

Due to the high correlation between user characteristics and group behaviors, there is a high demand for human mobility studies. However, the mobility datasets have two main challenges: privacy concerns and a lack of publicly available mobility data.

- **Data Privacy**: The data collected from users can reveal private lifestyle patterns, such as home and office addresses and user point of interest locations. Applying privacy protection tools to data before sharing it with third parties is required to relieve such concerns. The privacy issues are valid for both trajectory datasets and aggregated traffic mobility datasets.

While privacy issues on individual trajectories are well known, it is already proven that aggregated mobility datasets also have some privacy issues. The authors were able to re-identify user trajectories from raw aggregated trajectories [**220**]. User locations need to be privatized at the individual trajectory and aggregated level.

- **Data Availability**: Limited mobility datasets can have significant negative impacts on transportation systems, from biased and inaccurate AI/ML models to reduced safety and inefficiencies in transportation systems. Enhancing the availability of mobility datasets provides additional benefits to the transportation sector, such as improved safety, better efficiency, and increased innovation.

1.4. Contributions

1.4.1. Performance of RL-TSC. Traffic signal controller (TSC) has a crucial role in managing traffic flow in urban areas. We designed an on policy multi-agent RL model assuming a vehicular network-based communication environment. We first analyzed the performance of on policy RL-TSC model in the case of a small-scale synthetic TSC network and real TSC network from the San Francisco downtown in terms of traffic delay and air quality in [**92**] using SUMO traffic simulatior. The results show that RL-TSCs are prone to high air pollution compared to standard TSCs for urban traffic control if it does not carefully designed.

Although the proposed RL-TSC model performed well, these RL-TSC still need to be improved for real-world deployment due to limited exploration of different performance metrics such as fair traffic scheduling or air quality impact. Furthermore, we introduce a constrained multi-objective RL model that minimizes multiple constrained objectives while achieving a higher expected reward. Our proposed RL strategy integrates the peak and average constraint models into the RL problem formulation with maximum entropy off-policy models. We applied this strategy to different networks of TSCs. As part of this constrained RL-TSC formulation, we discuss fairness and air quality parameters as constraints for the close-loop control system optimization model at TSCs called *FAirLight*. Our experimental analysis shows that the proposed *FAirLight* achieves a good traffic flow performance in terms of average waiting time while being fair and environmentally friendly.

Our method outperforms the baseline models and allows a more comprehensive view of RL-TSC regarding its applicability to the real world.

Our relevant publications are listed below:

- A. Haydari, M. Zhang, C. N. Chuah, & D. Ghosal, "Impact of Deep RL-based Traffic Signal Control on Air Quality." In 2021 IEEE 93rd Vehicular Technology Conference (VTC2021-Spring) (pp. 1-6). IEEE, (2021, April).
- A. Haydari, V. Aggarwal, M. Zhang , C. N. Chuah, "Constrained Reinforcement Learning for Fair and Environmentally Efficient Traffic Signal Controller ", Under Review, ACM Journal on Autonomous Control, 2023

1.4.2. Security of RL-TSCs. Controlling ITS components with learning-based model opens up a new attack surface for adversaries [**127**, **203**, **222**]. Misleading the behavior of ITS controllers with adversarial samples may result in life-threatening conditions. One of the main application areas of learning based controller models is TSC with RL. Therefore, the security analysis of RL-TSCs needs to be investigated. We identified two main threat models for RL-TSCs: injecting minimal random perturbations to the learning controller directly or sending falsified information to the TSC using Sybil or compromised vehicles. We investigate the RL-TSC security in adversarial attacks with threat models and a novel statistical detection mechanism in Chapter 4. The proposed security model reaches average 98% of detection performance. The relevant publication for this work is below:

- A. Haydari and C.N. Chuah, M. Zhang, Adversarial attacks and defense in deep reinforcement learning (DRL)-based traffic signal controllers. IEEE Open Journal of Intelligent Transportation Systems, 2021.

1.4.3. Privacy and Availability of Vehicular Mobility Datasets. The widespread adoption of location-based services and smart GPS devices (smartphones/watches) make continuous monitoring of human mobility both desirable and feasible. Such mobility data can enable different smart urban planning and other applications, such as training learning-based ITS controllers. However, real mobility data can reveal private lifestyle patterns (e.g., home/office addresses, points of interest). Removing personal identifiers from the dataset does not adequately provide privacy because attackers can still re-identify users [50]. These privacy concerns inhibit free sharing of mobility or CAVs data across multiple entities.

ITS research can face limited progress due to insufficient training data. Since existing publicly available datasets [36, 123, 167] are not rich enough for large-scale evaluations, researchers resort to simulating realistic human trajectories with generative models. Although there are several attempts at mobility data generation, their capabilities are limited to specific data types.

In this thesis, we investigate two primary directions in the realm of privacy preservation and mobility data generation. The First direction focuses on privacy protection mechanisms for different granularity levels of vehicular mobility datasets, including individual and aggregated mobility datasets, using differential privacy (DP), which protects the query output against inference attacks regardless of background knowledge. We proposed a DP-based map-matching algorithm, called DPMM, that generates link-level location trajectories in a privacy-preserving manner to protect users' origin destinations (OD) and travel paths. OD privacy is achieved by injecting Planar Laplace noise to the user OD GPS points. Travel-path privacy is provided with randomized travel path construction using exponential DP mechanism [93]. Furthermore, we extended this to aggregated datasets using map-matching to protect the origin destination of users at aggregated mobility networks [90]. This is achieved by injecting Planar Laplace noise to the user origin and destination GPS points. The noisy GPS points are then transformed into a link representation using a map-matching algorithm. The injected noise level is selected using the Sparse Vector Mechanism. This DP selection mechanism considers the link density of the location and the functional category of the localized links.

The second direction delves into developing synthetic data generation tools utilizing transformer-based models [201], specifically on generating individual trajectories. By leveraging the capabilities of transformers, this approach addresses the challenge of data availability by synthesizing realistic mobility data, which can be crucial in scenarios where access to real-world data is limited or restricted [89].

Through these complementary directions, we aim to contribute to the advancement of mobility data generation techniques while addressing critical privacy concerns in the context of ITS and beyond.

Our relevant publications are listed below:

- A. Haydari, C. N. Chuah, M. Zhang, J. Macfarlane, & S. Peisert (2022, December). Differentially private map matching for mobility trajectories. In Proceedings of the 38th Annual Computer Security Applications Conference (pp. 293-303).
- A. Haydari, C. N. Chuah, M. Zhang, J. Macfarlane, & S. Peisert (2023). Differential Privacy in Aggregated Mobility Networks: Balancing Privacy and Utility. arXiv preprint arXiv:2112.08487.
- A. Haydari, D. Chen, Z. Lai, & C.N. Chuah (2024). MobilityGPT: Enhanced Human Mobility Modeling with a GPT model. arXiv preprint arXiv:2402.03264.

CHAPTER 2

Impact of Deep RL-based Traffic Signal Control on Air Quality

2.1. Introduction

Air pollution becomes a very problematic issue in urban areas due to the rise of the number of motor vehicles. In the US, transportation accounts for the 28% of greenhouse gas emissions in which 97.2% of source of emission is CO_2 via consumption of fuels [**3**]. Vehicular emission depends on several circumstances such as traffic condition, vehicle characteristics, and driver behaviors. Traffic intersections play a key role in managing mobile air pollution since frequent vehicles' speed changes and stop-and-go traffic result in increased fuel consumption and CO_2 emissions.

Machine learning-based control mechanisms in intelligent transportation systems (ITS), such as traffic signal control (TSC) systems, take action based on real-time data from the environment for online updating. Today, popular learning-based controller approaches combine deep neural networks (DNN) with RL, referred to as DRL, in which policy estimation is performed by DNNs. One good example application of such methods in ITS is developing the optimal traffic signal schedules. In general, learning-based TSCs perform better than standard dynamic TSCs in terms of delay and throughput for multi-intersection settings [**91**]. However, it remains an open research question how such learning based TSCs affect local mobile emissions near the surface streets.

In this context, we investigate the emission and fuel consumption produced by DRL controlled intersections. To assess the impact of such controllers in terms of the emissions, we consider policy-gradient-based advantage actor-critic (A2C) DRL algorithm with multi-agent settings and simulate the following: (i) grid-like 4-intersection TSC scenario, and (ii) the San Francisco Downtown road network. We run all our experiments on the SUMO traffic simulator where pollutant emission and fuel consumption models are derived from the HBEFA application database [**87**]. SUMO collects fuel consumption and pollutant emission results from each vehicle individually based on the speed and acceleration parameters. These emission statistics are examined using different type of traffic network settings.

The contributions of this work are as follows:

- We quantify fuel consumption and CO_2 emission rates with multi-agent DRL controller methods using a simple delay-based penalty function. Our results show that the pollution levels are highly correlated with the total travel time in intersections and reducing the travel times spent in intersection also lowers the CO_2 emission.
- In addition to simulation study of a synthetic 2x2 grid network (Fig. 2.2), we train and test our DRL controller on the San Francisco downtown network with real data consisting of 24 hours traffic flow. To study the effect of peak and off-peak hours, we also perform different trace-driven simulations with 3 hours in the morning and 3 hours in the afternoon traffic flow, respectively.
- Although DRL-based TSCs perform the best on the synthetic network, they do not outperform the rule-based TSC method (max pressure control) in the San Francisco downtown network in terms of CO2 emissions and fuel consumption. DRL-based TSCs outperforms both fixed-time and queue-based vehicle-actuated TSCs.

The rest of the chapter is organized as follows. Section 2.2 discusses related work while Section 2.3 provides background for DRL learning agents and TSC settings. We discuss our simulation results in Section 2.4. Section 2.5 concludes the chapter.

2.2. Related Work

Learning-based TSC control mechanisms have good performance compared to classic TSC approaches. One such approach leverages different DNN settings, RL settings and traffic network structures referred to as DRL [**91**]. In general, the performance of learning based TSCs are better than standard TSC controllers in terms of delay and total waiting time [**68**]. Existing DRL based TSC approaches may differ from one another in terms of problem definitions [**196**], neural network structures [**155**] and applied algorithms [**28**]. While some studies control multiple intersections with a centralized agent [**232**], some others assign different agents for different intersections with multi-agent models [**39**].

Emission and fuel consumption increases in the urban areas due to high load of traffic and congestion [**19**]. Authors in [**120**] evaluates the impact of TSCs on air pollution based on VT-Micro microscopic fuel and emission estimation model [**4**]. Another team studies the effects of coordinated and non-coordinated TSC on emission rates with different emission models [**42**]. The work in [**64**] examines the roundabout effects on air pollution on a microscopic traffic simulator by comparing the results with standard fixed-time TSCs. A recent review discusses impact of different traffic management systems such as lane management, speed management and traffic flow control strategies on air pollution [**19**].

There are not many studies investigating the effects of learning based TSCs on air quality. In this work, we examine the effects of learning based TSCs on CO_2 emission and fuel consumption on both a synthetic network and the San Francisco downtown network with the SUMO microscopic traffic simulator.

2.3. DRL-based Traffic Signal Controllers

2.3.1. Deep Reinforcement Learning. Reinforcement learning (RL) is a trial-and-error based learning algorithm where agent interacts with the environment and takes action to maximize cumulative reward. Mathematical formulation of RL is based on Markov Decision Process (MDP). In general, an RL agent interacts with the environment and receives a numerical reward (or penalty if it is negative). Continuously observing the environment called state s_t, receiving feedback from the environment r_t and taking action a_t, an RL agent learns an action policy which defines how to behave by computing action value function $Q(s_t, a_t)$ after each iteration [**194**].

2.3.2. Advantage Actor-Critic DRL. In a general DRL model, DNNs extract the features from data with multi-layered neural networks [**149**]. Actor-critic-based DRL models consist of policy estimation and value function estimation algorithms applying to an advantage function (Fig 2.1). While the actor is responsible for determining the actions to take, based on the current state, by learning a policy function that maps states to actions, the critic evaluates the actions taken by the actor and provides feedback on how good or bad those actions were. The critic learns a value function that estimates the expected future reward for a given state and action. Actor-critic reinforcement learning offers several key advantages over traditional Q-learning approaches, particularly in tasks with continuous or high-dimensional action spaces, where the actor network can directly output the optimal actions rather than requiring a costly search, leading to more stable and efficient learning. Additionally, actor-critic methods can learn stochastic policies, enabling better exploration-exploitation trade-offs, and scale more effectively to high-dimensional state spaces by leveraging the compact policy representation learned by the actor network, making them a more powerful and flexible reinforcement learning technique for a wide range of real-world applications compared to Q-learning.

Actor-critic models update both actor and critic networks synchronously. There are several synchronous and asynchronous actor critic models in literature [**130**]. Asynchronous advantage actor-critic (A3C) models estimate both actor and critic networks in parallel asynchronously, which increases the computation time. Since there is not much performance difference between synchronous and asynchronous actor-critic models, we used synchronous actor-actor critic method know as A2C in on-policy settings, which utilize the most recent interaction data to update the policy and value function, leading to more efficient use of samples.

2.3.3. Deep Reinforcement Learning for TSC. In this work, the states of A2C agents are value vectors for each incoming lane of intersection. For one intersection, we created two value vectors for each lane: one is average speed and the other is total number of vehicles. Position and speed of each vehicle can be collected from individual vehicles via vehicle-to-infrastructure (V2I) communication to calculate the average speed and number of vehicles. Using the formed state input, the DRL agent in TSC selects a green phase from among possible green phases: North-South Green, East-West Green, North-South Advance Left Green, East-West Advance Left Green. Each selected green phase is executed after a yellow phase transition. With the objective of maximizing cumulative reward, a scalar reward is computed for penalizing or rewarding each taken action. There are several reward/penalty definitions for TSC settings such as vehicle waiting time, cumulative delay, and queue length. Although there are more complicated reward designs in literature, the authors in [**232**] demonstrated that in general, simpler state and reward definitions are superior to the complex reward functions. For this reason, in our DRL-based TSCs, we choose a simpler reward function namely the change in the waiting time at an intersection for one green phase.

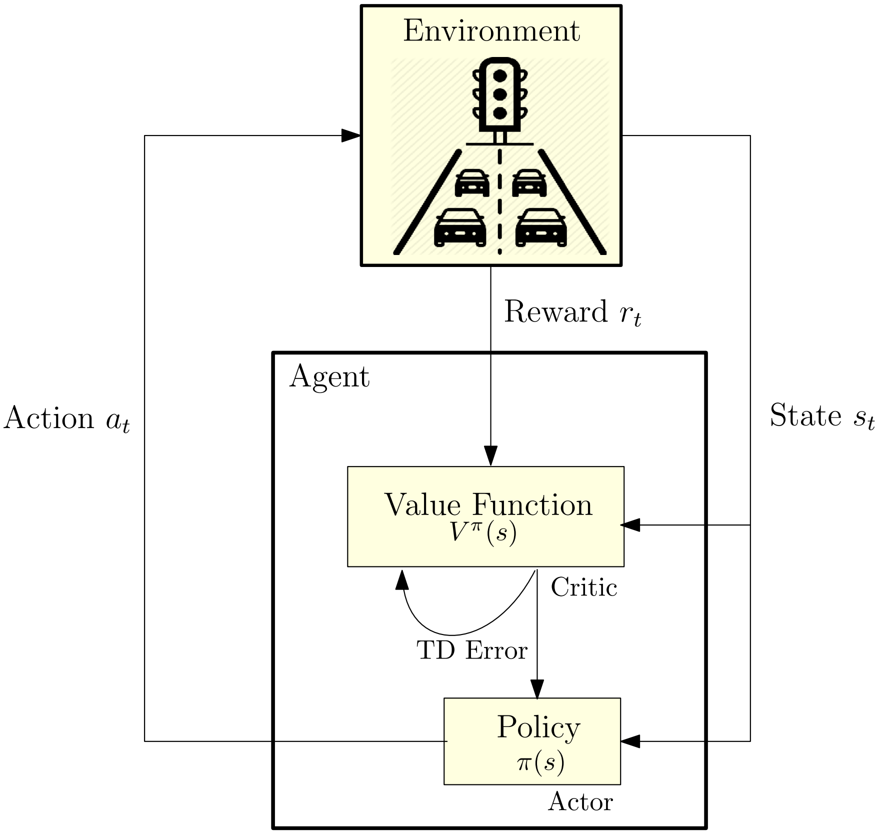


FIGURE 2.1. Actor Critic RL model for a TSC

For DRL models, designing a DNN structure for better performance is another critical step. In this work, we used multi-layer perceptron with 5 layers for both actor and critic, with "relu" and "softmax" activation functions for policy estimations of learning agents. In multi-agent RL settings, interaction with the nearest agents is necessary to reach a global optimum. In our experiments, each agent updates its policy by including the current traffic condition of neighbor TSCs as well to decrease the overall traffic delay. The global state is found with concatenation of the local states of neighboring intersections and the reward is generated by summing the local rewards of neighboring intersections.

2.3.4. Fuel Consumption and Emission Models. There are several vehicle acceleration-based emission estimation models such as HBEFA [**94**], MODEM [**224**]. We adopted the HBEFA emission estimation model in our experiments, which is widely used in Europe, with SUMO traffic simulator providing a variety of tools for collecting statistics from the simulation. The parameter called relative positive acceleration (RPA) is a key component determining the emission rates for driving cycles. RPA value is calculated using the equation:

$$RPA = \frac{1}{\lambda} \sum a_i * v_i * \Delta t \tag{2.1}$$

where λ is the total traveled distance, a_i is positive acceleration value, v_i is speed for the sample i, and Δt is the time interval between sample i and $i - 1$.

The latest version of HBEFA is v4.1 released in August 2019. Although HBEFA includes a large amount of source data for different sort of pollutants, SUMO only allows its users to simulate a few of them such as fuel consumption, CO_2, CO, HC. In our experiments, we only measured the rates of fuel consumption and CO_2 since we know that 97.2% of emission in traffic is only CO_2. SUMO also enables the use of different vehicle classes for simulating such parameters. Some of them are passenger cars, buses, heavy duty vehicles with gas driven and diesel driven types. In this work, we only simulated one type of vehicle that releases the same amount of gas to the air and consumes the same amount of fuel for all the vehicles.

2.4. Experimental Evaluation

In this section, we experimented the impact of DRL-based TSCs on fuel consumption and CO_2 emission statistics using SUMO [**138**] microscopic vehicular traffic simulator with Tensorflow Python API for controlling multi-agent A2C agents. Both synthetic and real networks are trained on the same agent parameters with 2000 experience replay memory size, discount factor $\gamma = 0.95$, as well as, 0.00001 and 0.000005 learning rates for actor and critic networks, respectively.

All our experiments compare the performance of DRL TSCs with three baselines. One of the baselines is standard fixed time TSC where green light times are allocated to each direction with pre-defined duration. We also compared our method with two adaptive control methods: queue-based vehicle-actuated TSC [**214**], and max-pressure-based TSC [**200**]. Maximum phase duration for the vehicle-actuated controller and the max-pressure controller and DRL controller is set to be 40 seconds.

2.4.1. Results from the Synthetic Network. In this section, we perform experiments on a multi-intersection environment with A2C DRL-based TSCs using 4 connected intersections (see Fig. 2.2). One traffic intersection has only 3 incoming roads while the other three intersections have 4 incoming roads. The roads connecting the different intersections are 1000 meters long, while the roads on the edges are 500meters long. 1 hour traffic flows on the synthetic traffic network constitutes one episode. The traffic is generated one vehicle per second by selecting the origin and destinations randomly. We trained our DRL agent on synthetic network for 20 episodes.

Due to space limitations we do not include comparison results with other DRL methods here but our previous experiments show that multi-agent A2C model achieves the best performance among other DRL models. In this study, we only showed the impact of multi-agent A2C (MA2C) model on fuel consumption and CO_2 emission rate in addition to total network waiting time. Fig. 2.4 shows the air pollution statistics and total vehicle waiting time with established baselines fixed-time, actuated and max-pressure TSCs throughout the simulation for the synthetic network. Fig. 4(a) exhibits the learning curve of multi-agent A2C agents in terms of total travel waiting time. Fig.

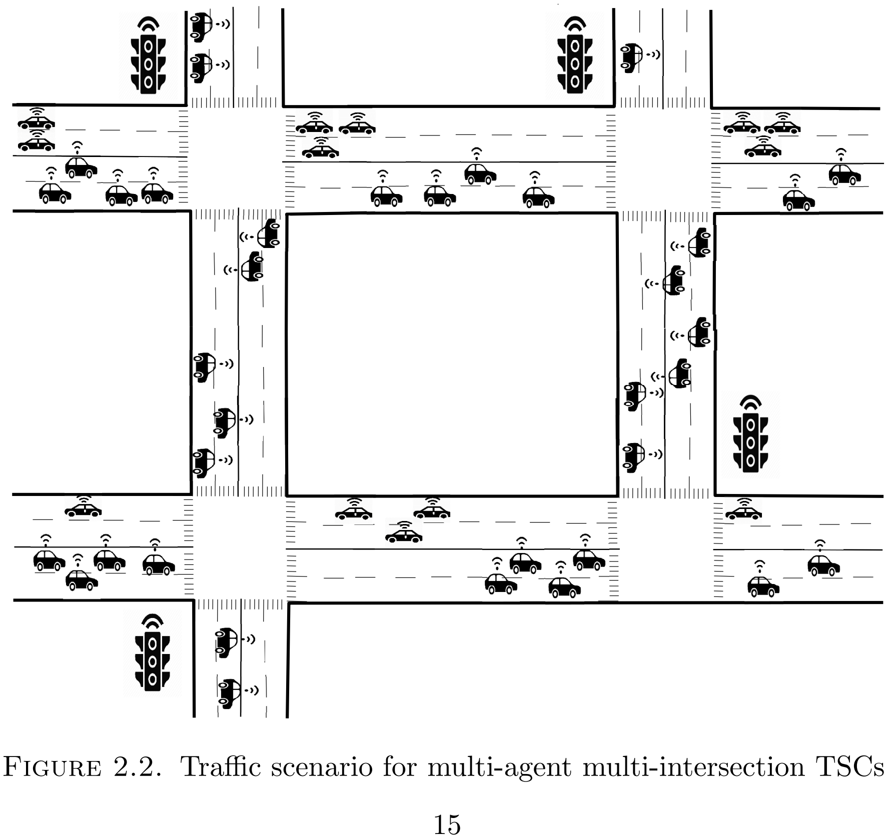


Figure 2.2. Traffic scenario for multi-agent multi-intersection TSCs.

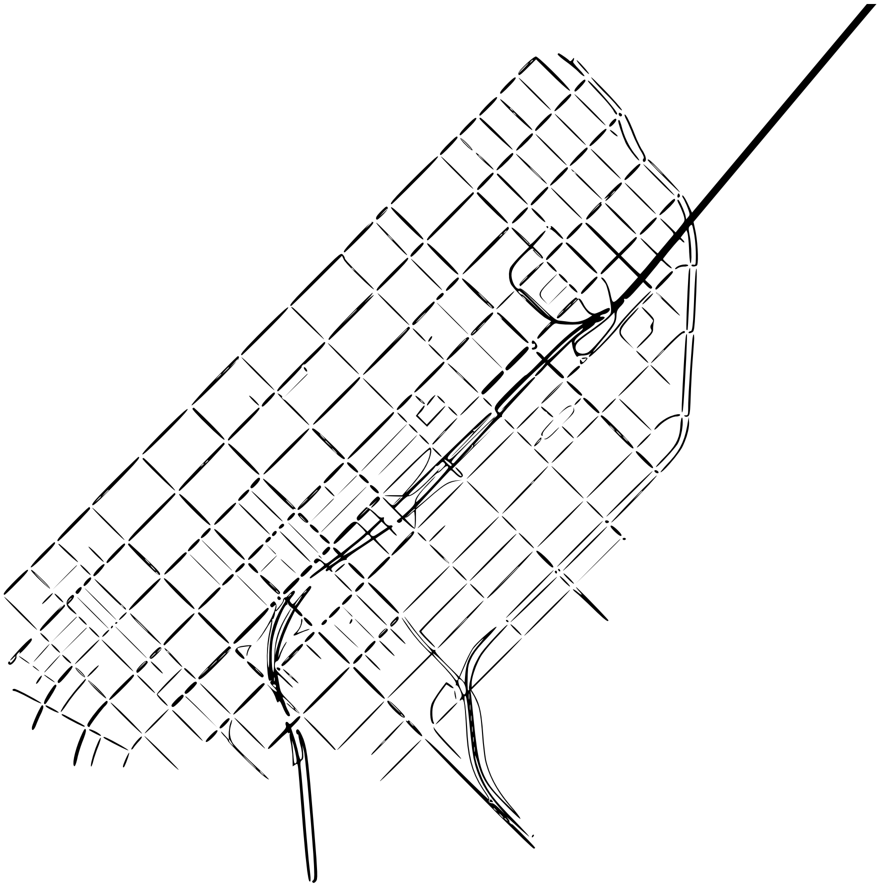
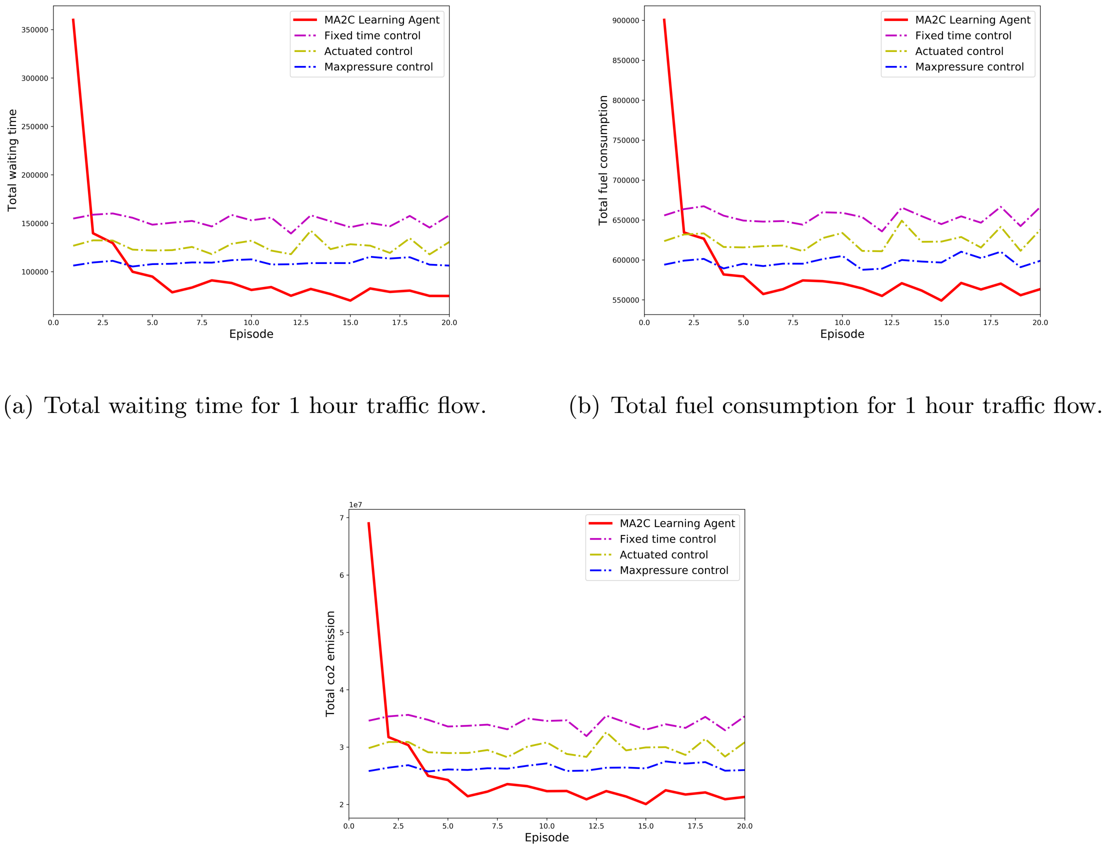


FIGURE 2.3. San Francisco downtown traffic network

4(b) and Fig. 4(c) show the total fuel consumption rate and total CO_2 emission rate. Results in Fig. 2.4 shows that DRL based TSC can achieve the minimal fuel consumption and CO2 emission rate, along with the total waiting time. In general, as the total vehicles travel time decreases, fuel consumption and CO_2 emission rate also decrease proportionally.

2.4.2. Results from the Real Network. In addition to simulating the synthetic road network, we evaluated the DRL-based traffic controllers and state-of-the-art conventional TSC controllers using a real dataset on San Francisco downtown road network, which follows a grid structure. The traffic from the bay bridge is also a part of the traffic flow in San Francisco downtown, where the bridge is merged with the main downtown traffic network. Fig. 2.3 shows the downtown San Francisco traffic network with 115 signalized intersections in total. Since it is not practical to control all the signalized intersections, we trained and tested only 10 neighboring intersections in the lower



(a) Total waiting time for 1 hour traffic flow.

(b) Total fuel consumption for 1 hour traffic flow.

(c) Total CO_2 emission rate for 1 hour traffic flow.

FIGURE 2.4. Waiting time, fuel consumption and CO2 emission rate results compared with standard fixed time and actuated controller models including max-pressure control

central downtown area. In addition, we tested our DRL agent with 4 neighboring intersections on the real road network, closer to the synthetic network. The results of such a 4 intersection controller have similar results with 10 intersections. Hence, in this study, we only present the 10 intersection controller model results below. We trained our DRL-based TSC controller with a 24 hours replicated traffic route file where similar timely traffic patterns are preserved. Cumulative CO_2 emission and fuel consumption rates are collected at around the signalized intersections. We presented real network test results in separate tables for three scenarios: 24-hour all-day traffic, 8am-11am morning traffic, and 5pm-8pm evening traffic.

First, the all-day simulation results for the San Francisco traffic network is shown in Table 2.1. Although multi-agent A2C achieves the highest performance in the synthetic network, it performs slightly worse than Max-pressured-based TSC in the San Francisco network. Among the four TSC models we evaluated, DRL-based TSC controller achieves the second best performance in terms of total vehicle waiting time, total fuel consumption and total CO_2 emission.

TABLE 2.1. Comparison of different TSC controllers using 24 hours traffic flow on San Francisco downtown network

TSC	Waiting time (sec)	Fuel (liter)	CO_2 (gram)
Max-pressure	**658656**	**1658.5**	**128443.5**
MA2C	783140	1835.2	146028.4
Actuated	845829	1925.4	159295.3
Fixed-time	1453968	2543.5	254762.2

Next, we studied the San Francisco network with 3 hours traffic flow for two groups of time periods: 8am-11am and 5pm-8pm. The purpose of this analysis is to identify how learning agents behave in different time periods of the day. SUMO runs traffic flow with a given route file. Since we have only one all-day dataset, we need to train the network with replicated traffic flow route files before testing learning agent with the actual traffic conditions. We randomly sampled traffic routes and replaced some of the routes with sampled routes for creating a replicated route file. This way, we preserved the same traffic behaviors for the given time period. However, we observe that training with the 3-hour dataset with one replicated route file does not provide sufficient learning for the DRL agent. Therefore, we generated 10 different route files with the same traffic behaviours and trained the DRL agent 10 episodes. Then we tested real traffic routes with DRL agent. Tables 2.2 and 2.3 summarize our results.

TABLE 2.2. Comparison of different TSC controllers using 3 hours traffic flow on San Francisco downtown network between 8am and 11am

TSC	Waiting time (sec)	Fuel (liter)	CO_2 (gram)
Max-pressure	**94762**	**285.6**	**19594.0**
MA2C	110485	310.1	21789.2
Actuated	141265	341.7	27024.0
Fixed-time	218525	421.1	38889.0

We begin with presenting the morning simulation results in Table 2.2. Similar to the all-day results in Table 2.1, the max-pressure TSC performs best in lowering traffic congestion, fuel consumption and CO2 emissions than other controllers, with MA2C comes second.

TABLE 2.3. Comparison of different TSC controllers using 3 hours traffic flow on San Francisco downtown network between 5pm and 8pm

TSC	Waiting time (sec)	Fuel (liter)	CO_2 (gram)
Max-pressure	**40537**	**146.7**	**8941.3**
MA2C	61584	170.6	11972.0
Actuated	69748	181.2	13495.7
Fixed-time	117731	227.5	20824.9

Next we present the results for the evening period shown in Table 2.3. Compared with the morning period, the evening period has lower congestion, fuel consumption, and CO_2 emissions, largely due a difference in traffic demand between the two peak commuting periods. Among all the control methods, the max pressure controller still performs the best, with MA2C being the second best. But the performance of MA2C is closer to that of the actuated controller in the evening period than in the morning period.

2.5. Conclusion

This chapter investigated the effectiveness of learning based TSCs in reducing fuel and emissions, as compared with other state-of-the-art conventional TSCs, on both a synthetic and a real road network. The main findings are (i) there is a high correlation between the CO2 emission and fuel consumption rates and the total waiting time, (ii) learning based TSC controllers are not universally more effective than other types of controllers in our application context. While the multi-agent A2C controller achieves the best performance on the synthetic network, it was outperformed by the max pressure traffic controller on the San Francisco downtown network in all three testing scenarios. Nevertheless, the DRL controller still performs the second best in these cases. Several factors influence the ability of DRL controllers to learn and generalize, one of which is the reward function. Our current study used a simple reward function based on vehicle waiting time only. We will explore other forms of reward functions including the emission in our future work to see if the performance of the DRL controller can be further improved.

CHAPTER 3

Constrained Reinforcement Learning for Fair and Environmentally Efficient Traffic Signal Controllers

3.1. Introduction

Traffic signals at intersections play a vital role in urban traffic management. The currently deployed Traffic Signal Controllers (TSCs) are predominantly fixed-time or vehicle-actuated controllers that are rule-based. These TSCs often provide sub-optimal control under rapidly changing and/or heavy traffic load.

Reinforcement learning (RL) has been shown to be a promising control strategy for TSCs with the advantage of adaptive learning in response to different traffic conditions [**91**, **129**, **213**]. RL learns how to act based on the feedback from the environment for online updating. RL-based TSCs can efficiently assign green phases and their durations to improve performance. Although the current TSC methods with RL made progress in regulating traffic flow dynamically, in general, prior works target the only one performance metric, such as increasing the traffic flow or decreasing the delay. However, this could result in poor traffic scheduling, causing substantial delays for vehicles or traffic flows with low traffic rates While a TSC that provides a high traffic flow is desirable, it may lead to other undesirable traffic conditions, such as longer vehicle waiting times on side roads (which raises fairness concerns) or increased air pollution.

A well-rounded TSC should, therefore, not only address efficiency as measured by delay or throughput, but also take into account its effect on fairness and greenhouse gas emissions. To this end, applying RL-based TSCs to the real world requires proven efficiency in multiple metrics. This research presents a novel constrained multi-objective RL approach formulation to circumvent the aforementioned restraints by combining traditional objective functions such as minimizing the delay along with fair scheduling and air pollution constraints.

Fair traffic scheduling is an essential performance aspect of TSCs. Prolonging the green phase time for one road direction may result in an unfairly higher waiting time for other traffic flow directions. The aim is to allocate proper time for all traffic directions (north-south, east-west) without exceeding the maximum green time for each phase in the TSC optimization performance. For instance, allocating disproportionate green time to the arterial road would cause higher waiting time for the side road entering vehicles. In this work, we consider an intersection level fairness in terms of maximum-green time, where the goal is to schedule traffic fairly for different approaching traffic flows to the TSC. We customize RL controller optimization with the maximum green time constraints that maintain more efficient traffic scheduling.

The efforts on TSC optimization considering air quality metrics (e.g., emission or fuel consumption) focus on offline methods. However real-time adaptive TSC mechanisms are rarely optimized with such metrics [**6**]. One well-studied objective function is minimizing delay and unnecessary stops at intersections for finding a signal plan to minimize the emission at intersections [**5**]. However, since the RL-based TSCs have proven to outperform the fixed-time controllers in heterogeneous and dynamic traffic conditions, optimizing the learning-based TSC by considering emission metrics is necessary. A relatively recent research studies the mixed linear integer optimization approach for dynamic TSC control with emission constraints [**83**], which accomplishes optimization with a set of cycle length durations. Our work, on the other hand, optimizes TSCs considering the CO_2 vehicle emissions with RL models in real-time.

Dealing with different objectives with an RL controller and integrating these different objectives into RL formulation in the form of constraints is challenging. In general, RL models focus on either instantaneous performance with *peak constraints* or long-term average performance with *average constraints*. In this work, we consider instantaneous constraints as the upper bound of air pollution and average constraints as the maximum green time with multi-objective form of RL formulation for TSCs. The goal is to achieve scheduling fairness in terms of maximum-green time violations and air quality in terms of lower CO_2 emission rate at the signalized intersections.

Several prior works studied the fairness [**176**] and air pollution aspects [**117**] of RL-based TSCs separately in different studies. Unfortunately, such studies cannot address multi-objective TSCs. Hence, the RL model should be designed considering different objectives in advance to reach an optimal control performance. In this work, we incorporate the fairness and air quality constraints to the RL formulation, strengthening the applicability of such learning-based TSCs to the real world. Specifically, we designed *FAirLight*, a multi-objective model-agnostic RL module using constrained RL formulation. Our key contribution is designing an RL framework with *peak* and *average* RL constraints. We conduct experiments on a network of signalized intersections with real datasets. Our model has improved fairness in terms of maximum green threshold by 68% compared to second best RL controller and achieves lowest CO_2 emission rates with around 6% improvement.

Our summary of contributions are as follows:

- We propose a generalized RL formulation with multiple objectives that maximizes the cumulative reward subject to average and peak constraints. Although RL has been studied with either average or peak constraints, but not both, this work combines the two constraints and each targets a different objective of signalized traffic control.
- We formulate the air quality-related constraints (CO_2 emission rate) with a linear regression model with respect to the vehicle occupancy rate and introduce an upper bound for emission rates. The emission rate is conditioned with a separate constraint value network, known as average constraints.
- We develop a constraint model of fair traffic scheduling with a maximum-green threshold term and incorporate a penalty function into the RL controller. This maximum-green threshold is a peak constraint, which learns a constraint on the reward function.
- We conduct simulation with real and synthetic traffic networks to evaluate the performance of our proposed approach

3.2. Related Works

In recent years, there has been growing interest in the research community towards developing effective strategies for controlling signalized traffic signals. One main direction is applying RL techniques to TSC optimization. Unlike traditional rule-based TSC methods like SOTL [**40**], learning-based TSC approaches have shown superior performance in terms of optimizing specific objectives such as traffic flow or delay reduction [**145**]. However, despite the advancements in RL-based TSC optimization, there is still limited exploration of multi-objective TSC optimization using RL models. While existing studies have primarily focused on optimizing a single objective, such as maximizing traffic flow or minimizing delay, the consideration of multiple objectives in TSC optimization remains an ongoing challenge.

Reward engineering for improving the performance of RL-TSCs has been explored [**215**, **221**]. Recently several researchers studied RL-TSC formulation with multiple objectives, including fair traffic scheduling and safety. One type of reward engineering is to add multiple objectives to the reward definition with weighted averaging. One of the earlier works utilized multiple traffic-related parameters in reward definition for RL-TSCs in [**112**]. In another study [**176**], authors proposed two fairness variants of rewards definitions with a delay-based and a throughput-based approach for RL-TSCs. The proposed strategy weighted averages the fairness objectives with the traffic flow. Thus it requires fine hyper-parameter tuning and does not guarantee convergence. However, modeling the objectives on Q values of the RL model tends to converge better with stronger guarantees. A hierarchical multi-objective RL model with a maximum green threshold for fairness property is presented in [**100**] . While RL is responsible for scheduling at a higher level, the lower-level module optimizes the control parameters. Safety is another objective considered in RL-TSC optimizations in a few studies [**51**, **73**]. Compared to prior works, in this research, we consider constrained RL formulation for fairness objective using average constraints formulations with separate Q function estimation.

Several earlier studies focused on air quality and fuel consumption at signalized intersections [**5**, **126**]. However, they generally target fixed-time or rule-based adaptive TSCs. This line of research aims to penalize vehicle stops since deceleration and acceleration cause more air pollution. An optimization study considering air pollution for rule-based TSCs using mixed linear integer programming is presented on a fairly simple traffic network in [**117**]. The authors proposed a new reward function to lower the air pollution at RL-based signalized intersections. In our previous work [**92**], we empirically evaluated the impact of RL-TSCs on air quality at intersections using the San Francisco downtown area with a real traffic dataset without constrained optimization. This work proposes a novel RL approach that integrates the air quality constraints into the RL objective using peak constraint formulation and further reduces the CO_2 emission rate at intersections by around 6%.

3.3. Preliminaries

3.3.1. Partially Observable Markov Decision Process. A Markov Decision Process (MDP) is a mathematical framework used to describe sequential decision-making processes. It consists of a tuple $(\mathcal{S}, A, T, R, \rho_0, \gamma)$, where $\mathcal{S}$ represents the state space, A represents the action space, $T : \mathcal{S} \times A \rightarrow \Delta(\mathcal{S})$ is the stochastic transition function, $R : \mathcal{S} \times A \rightarrow \mathbb{R}$ is the reward function, $\rho_0 : \mathcal{S} \rightarrow \Delta(\mathcal{S})$ is the initial state distribution, and $\gamma \in [0, 1]$ is a discount factor.

We frame the scenario in the TSC context as a Partially Observable Markov Decision Process (POMDP). When the environment is not entirely observable by the agent, an observation function Ω maps a state $s \in \mathcal{S}$ to an observation $o \in O$, where O represents the observation space. Thus, the agent's knowledge is based on these observations, allowing for decision-making in a partially observable environment.

The objective of the RL in this POMDP framework is to search for an optimal policy π^* that maximizes the expected cumulative reward over time. The goal is to find a policy that leads to the highest expected total reward. Mathematically, this can be represented as $\max_\pi \mathbb{E}_{(s,a)\sim\rho_\pi}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)\right]$. Here, π represents a policy, and $\rho\pi$ represents the state-action distribution induced by the policy. The reward function $R(s_t, a_t)$ provides immediate feedback after taking action a_t in state s_t. The agent aims to make effective decisions in the TSC domain by learning from these rewards.

3.3.2. Constrained Reinforcement Learning. Constrained RL aims to maximize the expected reward while satisfying some constraints in which the MDP environment is modeled by additional limitations such as peak [**65**] and average [**47**, **66**] constraints. While peak constraints limit the immediate reward function, $c_p(s_t, a_t) \geq 0$, average constraints target long-term limitations, $\mathbb{E}_{(s,a)\sim\rho_\pi}\left[\sum_t \gamma^t c_a(s_t, a_t)\right]$. Both peak and average constraints can be unknown functions with known return values at each time step. Additionally, the constraint functions $c_a(s_t, a_t)$ and $c_p(s_t, a_t)$ evaluate whether the constraints are satisfied under the current (s_t, a_t).

Most RL research direction focuses on either peak or average constraint formulations. When the problem domain has different constraint objectives with different functions, finding a Pareto optimality on a reward function is hard. Therefore, in this work, we implement multi-objective constraint learning to the RL formulation with both peak and average constraints to find a feasible policy while satisfying the constraints. The problem with peak and average constraints can be formulated as:

$$\max_{\pi} \ \mathbb{E}_{(s,a)\sim\rho_\pi}\left[\sum_t \gamma^t R(s_t, a_t)\right], \tag{3.1}$$

$$\text{s.t. } \mathbb{E}_{(s,a)\sim\rho_\pi}\left[\sum_t \gamma^t c_a(s_t, a_t)\right] \geq 0, \tag{3.2}$$

$$c_p(s_t, a_t) \geq 0, \tag{3.3}$$

where optimal policy is found in Eq. 3.1 by maximizing the expected cumulative reward, Eq. 3.2 and Eq. 3.3 satisfy average and peak constraints in long-term and immediate returns, respectively.

3.4. Constraints on RL-TSCs

The motivation behind this research is to develop a policy, denoted as π, that enhances the efficiency of Traffic Signal Controllers (TSCs) while simultaneously reducing emissions and ensuring fairness through constrained optimization. To achieve this, we introduce two constraints: a peak constraint, denoted as $c_p(s_t, a_t)$, which represents the maximum threshold for a green phase duration, and an average constraint, denoted as $c_a(s_t, a_t)$, which represents the total vehicular emission in a green phase.

3.4.1. Emission as a Constraint. As early studies stated [**83**], total emission on the road is closely related to the number of vehicles passing by in a given period. Assuming a linear relationship between emission and the number of vehicles, we can formulate the emission rate as an average constraint. Figure 3.1 depicts the linear correlation between the number of vehicles and the total CO_2 emission rate based on one-hour traffic simulation in a traffic network using a single vehicle type. More vehicle ‘stop and go’ will cause more traffic congestion, leading to higher emissions. A regression analysis can evaluate this relationship in closed form. In this work, we employ a linear relationship formulation of total emission vs the number of vehicles running on a road from [**83**]. Since we are using microscopic simulators for experiments, we use a discretized version of the emission estimation.

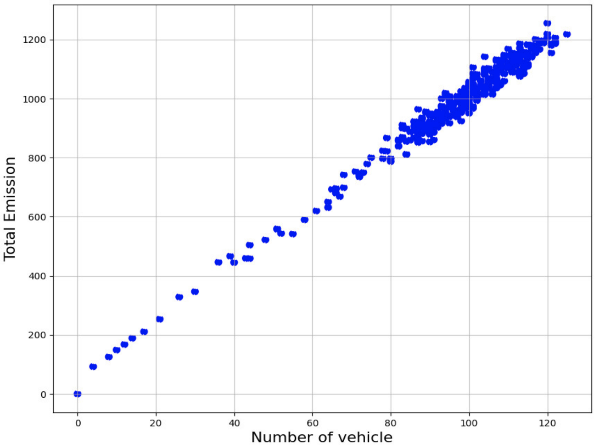


FIGURE 3.1. Relation of the number of vehicles vs total emission.

Let x_t be the number of vehicles at time t and L be a polynomial degree. The total emission ψ at a road link can be represented with:

$$\psi_t = \sum_{l=0}^{L} w x_t^l = \mathbf{w}^T \mathbf{x}_t \tag{3.4}$$

where $\mathbf{w}$ is the vector of coefficients and $\mathbf{x}_t$ is the vector of the number of vehicles on different orders of l. The coefficient vector $\mathbf{w}$ can be estimated using linear regression models. Considering the linear relationship, the maximum total emission on a road network at time t should satisfy the peak emission constraint $c_p(s_t, a_t)$. The coefficients $\mathbf{w}$ for a controller can be learned by pretraining, which will be used for constraining the total emission while RL training. We can define the peak constraint in terms of the total emission as:

$$\psi_t - d_t(s, a) \geq k \tag{3.5}$$

where ψ_t is the predicted total emission using coefficients $\mathbf{w}$, $d_t(s, a)$ is the estimated total emission during RL training and k is the desired threshold that RL agent tries not to exceed.

3.4.2. Maximum Green Time as a Constraint. Fixed-time traffic signal controllers (TSCs) operate with predetermined phase durations, making them suitable for consistent traffic volumes and patterns. In contrast, RL controllers typically operate on TSCs with discrete action spaces, where the RL agent selects a green phase from a set of available options. In situations with high traffic demand in a particular direction, the RL agent is expected to repeatedly choose the same action to prolong the green duration in that phase. However, this can lead to unfair traffic flow, as it favors one direction over others.

To achieve fair traffic scheduling at TSCs, the TSC must have both maximum and minimum green phase duration. During RL training, the minimum green phase duration is fixed, but the RL agent may exceed the maximum green threshold and allocate more time to a specific green phase. While a simple approach would be to force the RL agent to change the phase once it surpasses the maximum threshold, an ideal RL-TSC agent should learn a policy that respects such constraints.

Existing literature on learning-based TSCs often needs to look more closely at fairness criteria when optimizing TSC policies. Fairness, in terms of equitable traffic distribution among different directions, is an important aspect that should be considered for efficient traffic management. Consequently, there is a need to address fairness concerns in RL-based TSC approaches and develop policies that strike a balance between optimizing traffic flow and ensuring fairness.

We incorporate the maximum green threshold to the RL-TSCs as a peak constraint $c_p(s_t, a_t)$. The agent learns a policy π by minimizing the maximum green time violations,

$$\sum_t \gamma^t (g_{max} - g_t) \geq 0 \tag{3.6}$$

where g_{max} is the maximum green threshold for given traffic direction and g_t is the total green time of the current phase assigned by the RL-TSC agent.

3.4.3. Multi-objective Problem Setting. The aim of the proposed algorithm is to find a feasible policy while minimizing the peak constraint violations of Eq. 3.5 along with the average constraint violations of Eq. 3.6. Instead of dealing with two constraints separately in this work, we absorbed the peak constraints into reward and average constraints functions. Formally, the proposed strategy adds a penalty $c_p(s_t, a_t)$ to the global reward $r(s_t, a_t)$ and average constraint function $c_a(s_t, a_t)$ if the peak constraint is violated. Otherwise, peak constraint $c_p(s_t, a_t)$ is equal to zero.

By incorporating the peak constraints into the reward function and separate average constraint functions, the proposed algorithm aims to find a feasible policy that minimizes both the violations of peak constraints and average constraints simultaneously. This approach offers a unified framework to handle multiple constraints, allowing for more effective optimization of TSCs. By penalizing peak constraint violations within the reward function, the algorithm encourages the RL agent to prioritize actions that maintain compliance with the peak constraints, leading to improved overall performance and adherence to traffic regulations.

Using the multi-objective constraint formulation, the objective policy search problem can be expressed with average constraints as follows:

$$\max_{\pi} \ \mathbb{E}_{(s,a)\sim\rho_\pi}\left[\sum_t \gamma^t (r(s,a) + c_p(s,a))\right] \tag{3.7}$$

$$\text{s.t. } \mathbb{E}_{(s,a)\sim\rho_\pi}\left[\sum_t \gamma^t (c_a(s,a) + c_p(s,a))\right] \leq d \tag{3.8}$$

where expected value is computed with respect to the (s, a) pairs generated by policy π and the expectation of long-term costs generated by the policy π is less than or equal to d. In the remaining of the chapter, we represent expectation without $(s, a) \sim \rho_\pi$ to save space.

In literature, several research groups studied designing multi-objective reward designs for TSCs [**134**]. However, this work is the first initiative in this domain considering constrained and multi-objective RL for TSCs.

3.4.4. Conservative Constraints. In our research, we have made significant progress in advancing the constrained multi-objective reinforcement learning (RL) formulation by introducing conservative constraints. These conservative constraints have proven to be highly effective in reducing constraint violations and improving the performance of RL agents in complex environments.

To implement these conservative constraints, we have chosen to lower-bound the original constraints with tighter thresholds. By doing so, we establish more stringent limits that decrease the likelihood of constraint violations occurring during the RL process. This allows us to achieve lower rates of constraint violations compared to the traditional approach of using original inequality limits (as seen in equations 5 and 6).

The integration of conservative constraints has an important impact on the RL agent's learning process. By imposing stricter limits, the agent is encouraged to develop a policy that adheres more closely to the imposed constraints. As a result, the agent becomes more skilled at navigating the environment while staying within the specified limitations.

It is worth mentioning that our conservative approach is a point-wise lower bound, which has been commonly employed in achieving zero constraint violations in constrained reinforcement learning. This approach has been previously explored and validated in the works of Bai et al. (2022) [**8**] and Bai et al. (2023) [**9**]. We specify the values of constraint limits in the section of experimental analysis (Section 3.6)

3.5. Proposed Solution

3.5.1. Maximum Entropy RL. An RL model aims to maximize the expected return from the environment while exploring the environment. A well-known RL method for policy gradients with stochastic policies is based on the maximum entropy principle that controls the entropy of the policy with an additional term aiming to improve exploration [**56**]. We study a known maximum entropy RL model, Soft-Actor Critic (SAC), to learn optimum policies [**82**]. The objective function for SAC with entropy function is

$$\mathbb{E}\left[\sum_t \gamma^t r(s,a) + \kappa \mathcal{H}(\pi_\phi(.|s))\right] \tag{3.9}$$

where κ is the temperature parameter for adjusting the relative importance of the entropy versus reward, and $\mathcal{H}(\pi_\phi(.|s))$ is the entropy of policy π at state s_t, which is calculated as $\mathcal{H}(\pi_\phi(.|s)) = -\log \pi_\phi(.|s)$ to regularize the policy gradient objective. While the original implementation of SAC is for continuous action space, in this work, we implement the SAC in a discrete action space where each action refers to a green phase in a set of green phases.

The policy evaluation step computes the value of a state for policy π, and SAC defines a soft state value function with entropy term as:

$$V(s) \coloneqq \pi_\phi(s)^T[Q_\theta(s) - \kappa \log(\pi_\phi(s))] \tag{3.10}$$

Using the soft state value function from Eq. 3.10, policy evaluation updates the soft Q-value with Bellman backup ($Q : \mathcal{S} \times A \to \mathbb{R}$) as:

$$Q_\theta(s,a) \coloneqq r(s,a) + \gamma \mathbb{E}[V(s')] \tag{3.11}$$

where s' indicates the next-state after taking action a. Following the update on the soft Q-function with Bellman residuals, we can define the policy improvement as follows:

$$J_\pi(\phi) = \mathbb{E}[\pi_\phi(s)^T[\kappa \log(\pi_\phi(s)) - Q_\theta(s)]] \tag{3.12}$$

These steps and equations highlight the iterative process of policy evaluation and improvement in the SAC algorithm, where the soft state value function and the soft Q-value are updated to optimize the policy toward maximizing the expected cumulative reward.

3.5.2. SAC with Peak and Average Constraints. We approach the problem of finding feasible policy π under peak and average constraints with Lagrangian relaxation, which turns the constrained optimization problem into the unconstrained form. Soft Actor-Critic (SAC) is designed with an actor policy and multiple critics for reward and average constraints. Recently, [**81**] proposed SAC optimization with a Lagrangian update form for only average constraints. The constrained problem can be expressed in the Lagrangian form with both peak and average constraints, which is equivalent to a saddle point problem given by

$$\min_{\lambda \geq 0} \max_{\pi} L(\pi, \lambda) = G_R^\pi - \lambda \left(G_C^\pi - d\right), \tag{3.13}$$

$$G_R^\pi = \mathbb{E}\left[\sum_t \gamma^t R(s, a) + \kappa \mathcal{H}(\pi(.|s))\right] \tag{3.14}$$

$$G_C^\pi = \mathbb{E}\left[\sum_t \gamma_c^t C(s, a)\right] - d \tag{3.15}$$

where G_R^π and G_C^π are the reward and cost functions, the reward with peak penalty is $R(s, a) = r(s, a) + c_p(s, a)$ and average constraint with peak penalty is $C(s, a) = c_a(s, a) + c_p(s, a)$. The saddle point problem can be solved with a primal-dual gradient descent approach in alternating optimization between policy π and λ. For policy optimization in constrained SAC, we train two separate critics: Q value function Q_θ and a cost Q function Q_θ^c with the soft Q update rule. Q_θ trains the reward and entropy while Q_θ^c trains the cost critic network for average constraints. T^π

refers to the Bellman backup operator and is applied to the critic networks repeatedly:

$$T^{\pi} Q_{\theta}(s,a) := R(s,a) + \gamma \cdot \mathbb{E}_{\boldsymbol{\pi}(s')}\left[Q_{\theta}(s',a')\right], \tag{3.16}$$

$$T^{\pi} Q^c_{\theta}(s,a) := C(s,a) + \gamma_c \cdot \mathbb{E}_{\boldsymbol{\pi}(s')}\left[Q^c_{\theta}(s',a')\right]. \tag{3.17}$$

This paper represents the losses with J and in total, there are five losses: policy J_{π}, reward critic J_Q, cost critic J_{Q^c}, entropy coefficient J_{κ}, and Lagrange coefficient J_{λ}. We have the same update rule as described in [**82**] for reward critic Q_{θ} and entropy coefficient κ. Due to the Lagrangian update, the policy loss with constraints is similar to [**81**]. However, authors in [**81**] only deal with a single average constraint for policy search, while our work deals with both peak and average constraints in discrete action settings.

The original update rule for SAC policy is designed for the continuous action domain and requires a reparameterization trick to pass gradients through the expectation. However, since we use a discrete action space in our RL model and policy outputs action distribution, we can do back-propagation directly. In this case, the update for policy becomes:

$$J_{\pi}(\phi) = E_{s \sim D, a \sim \pi_{\phi}}[\pi_{\phi}(s)^T(\kappa \log(\pi_{\phi}(s)) - Q_{\theta}(s,a) + \lambda_{\eta}(Q^c_{\theta}(s,a) - d))], \tag{3.18}$$

In constrained SAC formulation, we have two parameter space to be estimated for reaching an optimal policy:

- κ Entropy term for maximum entropy which can be updated as in the original SAC work.

$$J(\kappa) = \pi_{\phi}(s)^T\left[-\kappa(\log(\pi_{\phi}(s)))\right] \tag{3.19}$$

- λ for Lagrangian constrained problem with dual update.

$$J(\lambda) = E_{s \sim D, a \sim \pi_{\phi}}\left[\lambda_{\eta}(Q^c_{\theta}(s,a) - d)\right] \tag{3.20}$$

Setting different update frequencies for different coefficients leads to stable performance for achieving desired policies. Thus, in this work, we set different update frequencies for entropy weight κ and Lagrange multiplier λ. Furthermore, Lagrange multiplier λ can also be estimated with a separate neural network model. However, to keep policy search more straightforward, we use a similar search strategy with κ as proposed in [**82**]. The complete constrained SAC algorithm with peak and average constraints is given by Algorithm 1.

Algorithm 1 Soft Actor-Critic with Peak and Average Constraints

```
Initialize weights $\phi, \theta, \theta_c, \kappa, \lambda$
Initialize target network weights $\overline{\theta}, \overline{\theta_c}$
Initialize learning rate $\alpha$ for $\{Q_\theta, Q_\theta^c, \pi_\phi, \kappa, \lambda\}$
Initialize a replay memory $D$
for each iteration do
  for each environment step $t$ do
    $a \sim \pi_\phi$, $s_{t+1} \sim p(s_{t+1}|s, a)$
    Collect $r$, $c_p$ and $c_a$
    $D \leftarrow D \bigcup \{s, a, r, c_p, c_a, s_{t+1}\}$
  end for
  for each gradient step do
    Sample batch experience from $D$
    Update gradients for $\{\theta, \theta_c, \phi, \kappa, \lambda\}$
    $\theta \leftarrow \theta - \alpha_Q \nabla J_Q(\theta)$
    $\theta_c \leftarrow \theta_c - \alpha_{Q^c} \nabla J_{Q^c}(\theta_c)$
    if gradient step $\mod k = 0$ then
      $\phi \leftarrow \phi - \alpha_\pi \nabla J_\pi(\phi)$
      $\kappa \leftarrow \kappa - \alpha_\kappa \nabla J_\kappa(\kappa)$
    end if
    if gradient step $\mod n = 0$ then
      $\lambda \leftarrow \lambda - \alpha_\lambda \nabla J_\lambda(\lambda)$
    end if
    Update target network weights
    $\overline{Q_\theta} \leftarrow \tau Q_\theta + (1 - \tau)\overline{Q_\theta}$
    $\overline{Q_\theta^c} \leftarrow \tau Q_\theta^c + (1 - \tau)\overline{Q_\theta^c}$
  end for
end for
Output Optimized parameters $\phi, \theta, \theta_c, \kappa, \lambda$
```

3.5.3. RL model for TSCs. This work implements multi-objective constrained RL formulation to the SAC maximum entropy RL model. Here, the state of the RL agent forms traffic conditions at intersections with a lane-level queue length of incoming approaches and the current green phase. Our experiments show inevitable performance differences with varying state forms, including vehicle waiting time and/or traffic flow speed.

In our RL-TSC model, the action space is structured in a discrete format, enabling the RL agent to make decisions by selecting a specific green phase from a predefined set of available green phases. This discretization of the action space allows for more manageable decision-making and simplifies the learning process.

When the RL agent chooses an action that necessitates a phase change, we incorporate a yellow transition phase to facilitate a smooth transition between different green phases. The inclusion of the yellow transition phase helps prevent sudden changes in traffic flow and allows vehicles to safely adjust their movement before the next phase begins. By incorporating this transitional phase, we ensure a seamless and efficient TSC strategy that minimizes disruptions and enhances overall traffic flow.

We acquired a reward function from the literature that is designed for penalizing vehicle stops to reduce fuel consumption [**181**]. Lower fuel consumption leads to lower gas emissions. This reward function, known as *performance index* PI in the emission research community, has been widely used by the TSC optimization [**5**]. The reward function PI is defined as follows:

$$PI = D + K * S \tag{3.21}$$

where D represents the delay experienced by vehicles, K is a linear coefficient, and S denotes the stop penalty measured in terms of the number of stops at incoming roads. The PI index penalizes both delay and stops in seconds, providing a comprehensive metric for evaluating TSC performance.

3.6. Experiments

In this section, we present the experimental evaluation of our proposed *FAirLight*, which utilizes multi-objective constrained RL formulation to optimize fairness and air quality. The proposed *FAirLight* employs cooperation between agents through graph-neural networks [**202**], which learns the traffic representation from neighboring intersections through graph-attention networks similar to [**212**].

We performed all the tests on NVIDIA Titan Xp with 32 GB RAM and Intel i9-9900k CPU using Ubuntu 20.04 device. The code for all experiments is publicly available[1].

3.6.1. Experimental Setup. We conducted our experiments using a simulated urban traffic environment with a real traffic dataset on a multi-intersection environment with varying traffic

[1] https://github.com/ammarhydr/FAirLight

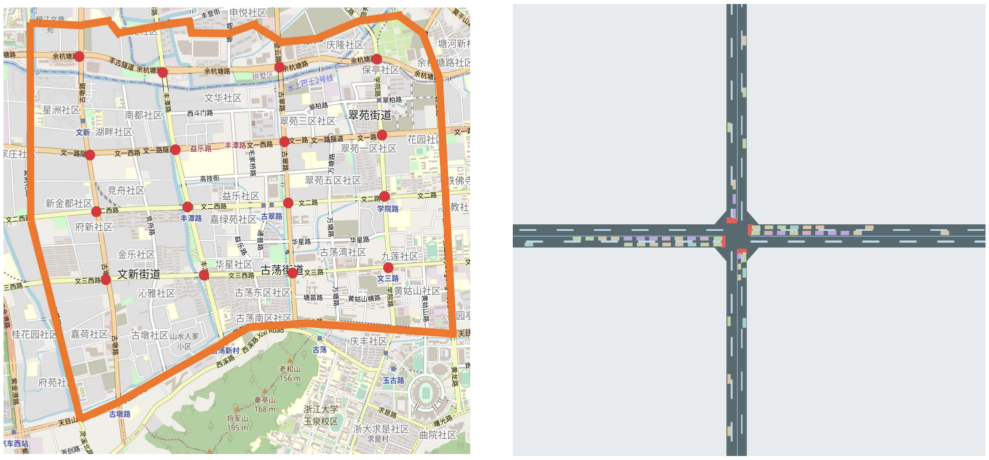


Figure 3.2. Representation of Experimented road networks, "Gudang sub-district, Hangzhou, China" and a single intersection.

density levels. The simulation was implemented using the Cityflow [**226**], which is widely used for traffic simulation and evaluation of traffic control systems.

Traffic Network Dataset

In our study, we conducted experiments on two distinct road network setups: a synthetic single-intersection road network and a city-level road network based on real data obtained from the city of Hangzhou, China (see Figure 3.2). The real traffic flow data contains information about the vehicles coming through the intersections that are collected with surveillance cameras. The dataset captured traffic flow for a duration of one hour and included details about vehicles using three different types of lanes at each intersection: left turn, through traffic, and right turn lanes.

The RL-TSC agent is responsible for selecting the appropriate green phase from a set of four available options: North-South Green, East-West Green, North-South Advance Left Green, and East-West Advance Left Green. Each chosen green phase was subsequently executed following a transition period involving a yellow phase. Our experiment design allowed us to evaluate the performance and effectiveness of the RL-TSC agent in both controlled synthetic road networks and real-world city-level road networks. By utilizing real traffic flow data, we aimed to create a more realistic and representative environment for assessing the agent's decision-making and traffic signal control capabilities. These experiments contribute to the understanding of RL-based traffic signal control methods and their applicability to real-world scenarios.

CO_2 **Emission Model**

The original Cityflow implementation does not support emission modeling. In this work, we utilized an emission model based on Handbook Emission Factors for Road Transport (HBEFA). While HBEFA includes an emission estimation model for varying vehicle models, we integrated CO_2 emission estimation model to Cityflow for only passenger vehicles using HBEFA-v3.1 with third-order polynomial and fixed coefficients [**118**]. As a result, the simulator collects instantaneous CO_2 statistics per vehicle in milliliters.

Constraint Settings

CO_2 emission rate is upper-bounded with the regression model. Given the characteristics in Figure 3.1, we choose a linear regression model with $L = 1$ in Eq. 3.4 for experiments. The goal of the RL agent is to reduce the CO_2 emissions at a signalized intersection by penalizing the green phases when the actual value exceeds the predicted value given the same number of vehicles. We design the CO_2 emission rate as an average constraint in the experiments. The threshold k in equation 3.5 is chosen as zero. While the actual constraint limit is the difference between the predicted and actual emission rates as in equation 3.5. However, the conservative constraint limit is applied on the predicted value as 10% lower than the ψ_t.

Considering the fairness, we formulate maximum green time as a peak constraint. Given the medium traffic demand on both road networks, the maximum green threshold for one direction is 45 seconds. The conservative version of the green threshold is selected as 35 seconds, known as the soft threshold constraint limit, to let the agent learn not to reach the hard constraint limit of 45 seconds.

3.6.2. Compared Methods. We compared the performance of our constrained RL-based TSC to four commonly used TSC methods:

- **Fixed-time**: The traffic signals switch between red and green phased at fixed intervals regardless of the traffic conditions.
- **Max-pressure** [**200**]: Max-pressure control is a TSC algorithm that aims to optimize traffic flow by adjusting traffic signal timings based on vehicles' pressure (or density) on the road using incoming and outgoing traffic. It is an adaptive and responsive control

method for changing traffic conditions, allowing it to adjust to different traffic patterns throughout the day.

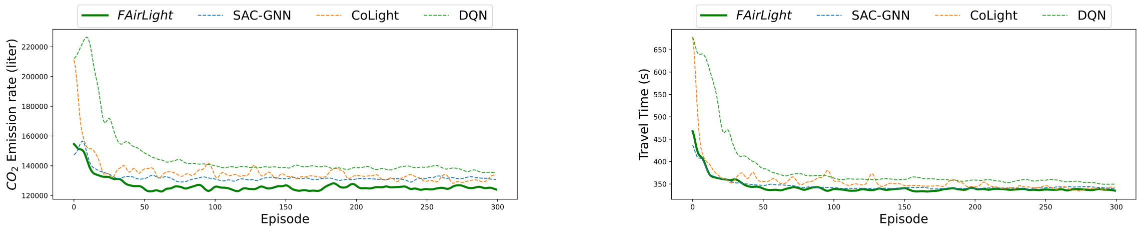
- **DQN** [**91**]: The standard value-based RL model minimizes the queue length or delay without fairness and emission constraints.
- **CoLight** [**212**]: An RL-based TSC model provides cooperation between neighboring intersections using attention models [**202**] for a network of intersections.
- **SAC-GNN** [**82**]: A standard Soft-Actor-critic RL model with graph-attention networks for TSC has been tested. Without cooperation between multiple intersections, SAC has a hard time converging. This approach becomes a standard soft-actor critic for single intersection network.

3.6.3. Ablation Study. Our study also aimed to examine the effects of two specific constraint formulations, namely "Only Peak" and "Only Average," on our constrained RL model for TSCs. The "Only Peak" constraint formulation involved applying a weighted sum of the green time and emission constraints to the RL model's reward function. In contrast, the "Only Average" constraint formulation utilized a separate critic q network that incorporated the sum of two constraints: maximum green time and emission.

By implementing and evaluating the constrained RL model with both constraint formulations, we could assess their respective impacts on the performance of *FAirLight*. The findings from this study provided insights into the effectiveness and limitations of the "Only Peak" and "Only Average" constraint formulations in optimizing TSC with a constrained RL model. This information can guide future research and development efforts in refining the constraint formulations and improving the overall performance of TSC systems.

3.6.4. Metrics. We evaluated the performance of the different traffic signal control methods using the following metrics:

- Average travel time: The average time refers to the travel time of vehicles through the simulated urban environment in terms of seconds.
- Queue length: The queue length is the number of stopping vehicles at intersections on average in terms of the number of vehicles.



(a) Average CO_2 Emission Rate for one hour

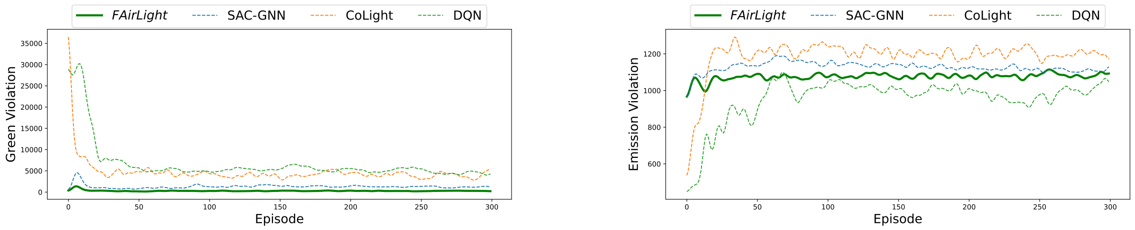
(b) Average Travel Time for one hour

FIGURE 3.3. The training process for the proposed *FAirLight* model compared with baseline methods in terms of CO_2 emission rate and average travel time. The lower the value is better the performance. The x-axis indicates the episode for one-hour traffic flow on the experimented road network.

- Average speed: The average speed is calculated by averaging the speed of all vehicles for one episode of traffic simulation. The constrained RL model minimizes the emission, and one leading factor for emission is the average speed. Speed is a frequently used metric for evaluating the emission of vehicles at signalized intersections.
- Total Emission: Total vehicle emission refers to the total amount of pollutants, CO_2, that are emitted by vehicles within a simulated environment in terms of liter. This is an accumulated statistic that considers the number of vehicles, their emission rates, and the amount of time they spend in the simulated environment. This metric is used to evaluate the environmental efficiency of different TSC strategies.
- Maximum green violation: Maximum green violation is a measure of the fairness of the traffic signal control system, calculated as the number of times a green phase assigns traffic flow more than maximum-green time.
- Emission violation: Emission violation rate considering the proposed constrained RL model refers to the rate at which RL-TSC allows vehicles to exceed the emission limits set for the simulated environment. We set the emission violation limit based on the pre-trained RL-TSC model, where the linear emission rate coefficients are estimated.

3.6.5. Numerical Results. Training Performance

In this work, we show the performance of *FAirLight* in terms of various aspects: traffic flow, emission, and constraint violations.



(a) Number of maximum green time violation (b) Number of CO_2 Emission Rate violation

FIGURE 3.4. The training process for the proposed *FAirLight* model compared with baseline methods in terms of constrained violations. The lower the value is better the performance. The x-axis indicates the episode for one-hour traffic flow on the experimented road network.

First, we assess the training performance of *FAirLight* on the Hangzhou TSC network. Figure 3.3 illustrates the results obtained from 300 episodes of one-hour traffic simulation, focusing on two key metrics: average CO_2 emission rate and average travel time. These metrics provide insights into the efficiency and environmental impact of *FAirLight*'s traffic signal control strategy during the training process. The results demonstrate that the *FAirLight* achieves the lowest CO_2 emission rate as compared to other baselines (see Figure 3(a)) by 8%. Furthermore, it is worth mentioning that previous studies, such as Stevanovic et al. [**192**] and De et al. [**42**], have also reported a limited improvement margin in terms of the emission rate.

The implementation of the proposed constrained RL model not only optimizes traffic flow but also effectively reduces air pollution at signalized traffic intersections. The model also improves travel time performance by prioritizing the reduction of CO_2 emission rate as one of its objectives. This multi-objective optimization approach reduces air pollution, ultimately positioning the proposed constrained RL model as the best controller in terms of both travel time and environmental impact (as depicted in Figure 3(b)).

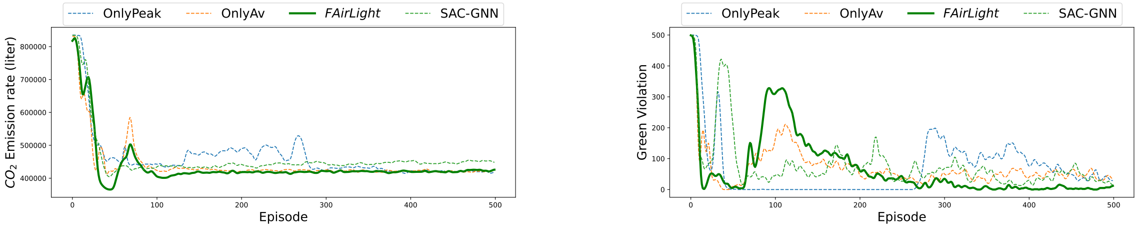
Regarding the baseline models, the *SAC-GNN* model, which employs the actor-critic policy gradient with maximum entropy reinforcement learning, demonstrates stable performance throughout the training process. It exhibits slight improvements over the *CoLight* model in terms of both travel time and emission reduction. In contrast, the *DQN* TSC model struggles to comprehend the environment's complexity and performs poorly compared to all other models. It yields the highest travel time and CO_2 emission rate, highlighting its limitations in optimizing the TSCs.

Next, we look at the constraint violation rates for different RL models. As shown in Figure 4(a), our model achieves the lowest violation rates through constrained optimization for peak constraints, maximum green time. On the other hand, the *DQN* TSC model achieves the lowest emission violations compared to all other methods (see Figure 4(b)). However, this does not lead to better emissions as *DQN* is the worst RL-TSC model. *FAirLight* lowers the constraint violations in both peak and average constraint formulations, leading to a fair traffic scheduling and CO_2 emission rate.

Table 3.1. Performance of different TSC models on Single intersection and Hangzhou 4x4 network intersections with respect to different metrics. Bold results show the best performance. While all the metric targets the lowest value, a higher speed is desired. The results are the average of the last 100 episodes.

Model Model	Travel T. (sec)	Emission (l)	Queue (#Veh.)	Speed (m/s)	Green V.	Emission V.
			Single Intersection			
FixedTime	355.6	631.1	160	**3.80**	-	49
MaxPressure	392.4	744.7	177	2.48	**0**	**48**
DQN	161.0	433.3	86.3	2.86	24	63.4
SAC-GNN	166.1	448.6	90.4	2.76	41	76.5
FAirLight	**156.0**	**421.3**	**83.1**	2.98	5	74.3
			Hangzhou 4x4 Network			
FixedTime	549.0	195.83	35.7	3.81	-	**572**
MaxPressure	407.0	159.25	20.0	4.51	11572	662
DQN	355.9	138.02	13.1	4.92	4892	979.6
CoLight	342.0	131.35	12.2	**4.96**	4031	1204.5
SAC-GNN	341.7	131.41	11.5	4.88	1266	1115.7
FAirLight	**337.7**	**125.21**	**11.1**	4.84	**277**	1083.5

To clarify the results, we show the performance metrics in Table 3.1, where the values average 100 episodes. The proposed *FAirLight* achieves the goals with the lowest travel time, emission rate, maximum green time violation rate, and the lowest queue length for both single intersection and Hangzhou networks. For single intersection *FAirLight* provides fair traffic scheduling by almost 79% better performance with average 5 maximum green time violations compared to the second-best TSC model *DQN* with average 24 violations. For the Hangzhou network, the proposed *FAirLight* has a



(a) Average CO_2 Emission Rate for one hour

(b) Number of maximum green time violation

FIGURE 3.5. The training process for the proposed *FAirLight* model compared with different variants in terms of emission rate and constrained violations. The lower the value is better the performance. The x-axis indicates the episode for one-hour traffic flow on the experimented road network.

similar fairness performance of average 79% green violation decrease compared to $SAC - GNN$. The other TSC models have unfair traffic scheduling performance, resulting in longer travel times for the light traffic direction by prioritizing the higher traffic demand incoming directions.

FAirLight slightly decreases the emission violation rates compared to unconstrained RL version *SAC-GNN*, which does not lead to the lowest CO_2 emission rate for both networks. *FixedTime* and *MaxPressure* controllers have the lowest emission violation rates, but their performance is not promising in lowering the CO_2 emission. The lower speed trend directs to a higher average travel time at TSCs as *FAirLight* is the best control method for reducing the average travel time and emission rate. Still, it fails to achieve that results in the average speed of running vehicles. Table 3.1 demonstrates that different TSCs have varying results on different metrics. At the same time, *FAirLight* provides consistent performance in terms of lower travel time, lower CO_2 emission rate, and fair traffic scheduling.

Ablation Performance

Lastly, we performed an ablation study to represent which part of the model performs better for giving good results on a single intersection network. With the ablation studies, we differentiate the *FAirLight* from *SAC-GNN* on different variants of constrained formulations.

In Figure 5(a) of the initial 100 episodes, we observed some fluctuations in the performance of our constrained RL model, particularly in terms of CO_2 emission rate. While the results appeared more favorable, it was primarily due to stuck traffic in one particular incoming traffic direction. In such scenarios, the emissions were significantly lower when vehicles were in idle mode, thereby misleading improved statistics. However, it is essential to note that this improvement was not representative of the overall performance, as the specific traffic conditions influenced it during those episodes.

Comparing the emission consumption with 'Only Peak', 'Only Average' and *SAC-GNN*, we found that our proposed TSC model, *FAirLight*, and ablation variants achieved a similar level of CO_2 emission rate of 421 Litre/Hour, demonstrating their effectiveness in reducing overall emissions. In contrast, the *SAC-GNN* model consumed slightly higher emissions, totaling 450 Litre/Hour. This difference highlights the advantage of incorporating the constrained RL framework to optimize TSC systems for emission reduction.

When considering the violation of green time constraints, *FAirLight* outperformed other variants and *SAC-GNN*. It achieved the lowest green violation rate, indicating its ability to manage green time allocations within the specified constraints. In comparison, *SAC-GNN* and the "Only Peak" and "Only Average" constraint formulations exhibited higher maximum green time violation rates. This suggests that *FAirLight* successfully balances optimizing traffic flow and adhering to the specified green time constraints, resulting in improved performance in fairness and lower emission with efficient traffic signal control.

3.7. Conclusion

This chapter introduces a novel approach to tackle the challenges of fair and environmentally friendly traffic scheduling through a multi-objective constrained RL model. By formulating the problem as a constrained multi-objective optimization task and integrating different objectives with a maximum entropy off-policy RL model, we aim to address both fairness and air quality concerns at the intersection and network levels. To evaluate the effectiveness of our model, we conducted experiments using both a synthetic dataset in a single intersection environment and a real traffic dataset from Hangzhou city, China, featuring a 4x4 intersection road network. The results demonstrate the effectiveness of the model in achieving lower travel times, reducing the CO_2 emission rate, and promoting fair traffic scheduling. Comparing our proposed *FAirLight* model with state-of-the-art RL-based and rule-based TSCs, we observed improvements in fairness and reductions in average CO_2 emission rates at signalized intersections. Additionally, our RL-TSC models showcased strong performance across various traffic flow objectives. By leveraging constrained RL and integrating peak and average constraints, we have demonstrated the potential to optimize traffic systems in a way that prioritizes fairness and sustainability.

In the future, we plan to generalize the performance of the *FAirLight* model to different road network structures and datasets. For this purpose, applying this model to larger city-level TSC models such as San Francisco, CA, or Manhattan, NY road network structure would be desirable.

CHAPTER 4

Adversarial Attacks and Defense in DRL Traffic Signal Controllers

4.1. Introduction

In recent years, data-driven approaches are often used to drive the design and performance evaluation of different control algorithms in Intelligent Transportation System (ITS). With the proliferation of such data-driven models and communication technologies, Information and Communication Technology (ICT) have revolutionized ITS by connecting different components: vehicles, road-side units and sensors, cameras, loop detectors and control modules such as ramp meters, traffic signal controllers via vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communications. In addition, some in-vehicle and road-side units are also connected to wide-area Internet via 4G/5G cellular technologies.

Learning-based control mechanisms in ITS, such as traffic flow control systems, travel demand prediction, and autonomous vehicles, take action based on real-time data from the environment. Traffic signal controller (TSC), which schedules the green/yellow/red phases at road intersections, plays a critical role in ITS, especially in busy urban settings. Control loops like TSCs often use real-time traffic information (e.g., captured by local cameras/sensors or broadcast messages from vehicles) to perform intelligent control decisions. This opens up the attack surface. Cybersecurity attacks such as falsified data may lead to erroneous control decisions, jeopardizing the safety and efficient operation of the transportation corridor. Mitigating risks due to those issues remains an open and active research area.

Machine learning (ML)-based learning models are classified into supervised learning, semi-supervised learning, unsupervised learning, and reinforcement learning (RL). The first three approaches use labeled or unlabeled training datasets to identify patterns and create models to discriminate between different output classes. On the other hand, RL learns by interacting with the environment and the actions are rewarded or penalized. The environment is typically stated in the form of a Markov decision process (MDP). RL agents exploit the knowledge to make cognitive choices, such as decision making and scheduling [**194**]. Today, popular learning-based controller approaches combine deep neural networks (DNN) with RL, referred as DRL, in which policy estimation is performed by neural networks. One good example application of such methods in ITS is estimating the optimal schedules of TSCs. In general, learning-based TSCs perform better than standard dynamic TSCs in terms of delay and throughput for isolated single-intersection and multi-intersection settings [**91**].

Learning based intelligent TSC agent collects messages from environment and schedules the traffic according to inputs. Recently, many DRL-based data driven solution methods are proposed in the literature for controlling TSCs in a network of intersections and a successful cyber-attack targeting such TSCs can cause chaos in cities. Regardless of the underlying technology (WAVE or 5G) for V2V or V2I communications, the defense mechanisms of learning based TSCs needs thorough investigation.

Learning-based TSCs may make wrong decisions or take wrong actions in the presence of adversarial attacks. In more advanced attack models known as insider attacks, attacker falsifies the data input by considering the target DNN structure of the learning model. There are two distinct clever adversarial attack settings on learning agents: white-box attack where attackers have access to the training model of learning agent and interacts with target model for generating adversarial inputs, and black-box attack where malicious inputs are generated from an estimated training model which is close to the true target model of learning agent [**37**]. In this study, we thoroughly investigate security vulnerabilities of DRL based TSCs under two adversarial attack models namely Fast Gradient Sign Method (FGSM) [**76**] and Jacobian-based Silency Map Attack (JSMA) [**163**] with white-box and black-box settings. We, then, proposed an online anomaly detection algorithm for detecting such adversarial attacks.

4.1.1. Adversarial Attacks on DRL-TSCs. The falsified data attacks generally designed with optimization techniques to identify which feature to perturb [**101**]. Similar to this analogy, the attack strategy in DRL targets DNN structures where policy of learning agent is calculated to find the minimum perturbation amount. There are two possible threat models for DRL-based TSCs; attack may be carried out in the cyber domain by directly accessing the input pipeline of

DRL agent or attack may be launched over the communication network by releasing falsified data from actual devices or Sybil devices to mislead the learning agent. Since FGSM attack perturbs all the input features only a slight amount, this attack can be launched purely in the cyber-domain without considering physical traffic conditions by accessing the input gate of the DRL agent. On the other hand, JSMA adversarial attack selects specific feature dimensions to perturb based on the constructed saliency map. JSMA can achieve this by using compromised vehicles or creating Sybil vehicles to send falsified data to TSCs.

In order to assess the impact of these adversarial attacks on different DRL-based TSCs, we consider both value-based, namely Deep Q Network (DQN), and policy-gradient with actor-critic-based, advantage actor-critic (A2C), DRL algorithms. We simulate the following: (i) single-intersection TSC scenario trained with DQN and A2C approaches, and (ii) multi-agent grid like 4-intersection TSC scenario trained with A2C approach. Since the black-box attack assumes attacker does not have access to the actual target DNN model, we trained a separate DRL agent with different traffic demands and DNN settings for black-box attack. All the experiments are performed using a realistic SUMO traffic simulator. Detailed analysis shows that DRL-based TSCs are vulnerable to cyber-attack with or without knowledge of the trained DNN models.

4.1.2. Defense Mechanisms Against Adversarial attacks on DRL-TSCs. Adversarial attack surface for targeting DRL agents is very broad. Therefore protecting DRL agents against adversarial attacks is a challenging task. There are two general protection mechanism for DRL agents: (i) the agent builds a defense mechanism within the agent model that increases the robustness of DRL agent against the attacks, (ii) the agent is equipped with an external detection mechanism that detects the anomalies and raises an alarm. One possible mitigation strategy for external anomaly detectors is changing the controller model from learning-based one to another model such as max-pressure TSC or actuated TSC. Since gradient-based adversarial attacks such as FGSM and JSMA generally have a minimal perturbation on the data, it is also hard to differentiate adversarial samples from real samples with standard anomaly detectors.

Given the adversarial attacks FGSM and JSMA for single intersection and multi-intersection scenarios discussed in the previous subsection, we studied the performance of statistical anomaly detectors to detect even infinitesimally small anomalies. An ensemble anomaly detector that combines two sequential anomaly detection models and an autoencoder-based anomaly detection model with CUSUM-like detection model is evaluated on the gradient-based adversarial attacks. The experiments show that proposed ensemble sequential anomaly detection model achieves the best detection rate with different DRL agents and TSC scenarios.

4.1.3. Contributions. In this work, we characterize the impact of two state-of-the-art adversarial attack models on DRL-TSCs and evaluate multiple statistical anomaly-based detection techniques. Our ensemble detection mechanism outperforms the other statistical anomaly detection models. The contributions of this work can be summarized as follows.

- We demonstrate experimentally that both FGSM and JSMA adversarial attacks degrade the performance of DRL-based TSC agents as long as attack continues. White-box and black-box FGSM attacks have similar effects on TSC. However, black-box JSMA attack is less effective compared to white-box JSMA attacks.
- We developed and applied a sequential anomaly detection mechanism to the FGSM and JSMA adversarial attack on DRL-TSC scenarios with single intersection and multiple intersection models. The method combines multiple detection models in a computationally efficient method.
- The ensemble anomaly detection method is agnostic to both the model of the neural network policy and the type of adversary. Hence, the detection algorithm protects the DRL-TSC agents against different adversarial attack models.
- While different sequential anomaly detection models achieve the best performance on different attacks and DRL settings, our proposed ensemble model achieves the best detection performance on all the scenarios.

The rest of the chapter is organized as follows. Section 4.2 discusses related work while Section 4.3 provides background for DRL learning agents and TSC settings. We present our adversarial attack models in Section 4.4 and statistical anomaly detection model in 4.5. We discuss our adversarial attack and defense results in Section 4.6 and Section 4.7, respectively. Finally, Section 4.8 concludes the chapter.

4.2. Related Work

Adversarial machine learning is an active research field for data scientists. Many attack models and defense mechanisms have been studied by researchers for different ML models including DNNs [**209**]. DRL agents are vulnerable to different kind of adversarial attacks and detecting such adversarial attacks is a challenging task. In this section, we review the existing works on security of TSCs, DRL adversarial attacks and potential detection models.

4.2.1. Security of TSCs.

Initial studies on adaptive TSC methods are rule-based or threshold-based control methods where predefined values of different traffic parameters such as queue or delay can trigger adaptive rules [**160**]. Lately, many machine learning-based TSC control mechanisms have been proposed. One such approach leverages DNN in a RL agent referred to as DRL and applies it to a network of traffic intersections [**91**]. The performance of learning based TSCs are generally better than standard TSC controllers.

There are many security analysis papers in literature for different type of TSCs. In [**128**], the authors identified some of the underlying threats against TSCs and proposed a game-theoretic risk minimization model without specifying the type of TSC. The study assumes that attacker has access to the control center and manipulates the traffic lights directly. Security of single intersection and multiple intersection back-pressure based TSCs is studied in [**223**]. The same group later extended their study with multiple attack strategies with several protection algorithms [**222**]. With the advanced vehicular and communication technologies, vehicles expected to be communicate with the TSCs through Vehicular Ad Hoc Network (VANET). The security vulnerabilities of such VANET-based TSCs are investigated without considering a signal control mechanism in [**101**] where adversary uses decision three ML model to find the optimum perturbation. Although machine learning-based, especially DRL TSCs, offer promising performance gain, their security vulnerabilities need to be studied carefully. Apart from TSCs, there are various other studies on assessing the vulnerability of different ML-based ITS control mechanisms. Autonomous vehicles need to have a perfect perception while driving. Hence, deep learning has been exploited to process high-dimensional data. Since securing autonomous vehicles against malicious activities is an important and challenging task [**171**], the effects of adversarial attacks on DNN structures are studied in [**25**] where LIDARs of autonomous vehicles are under attack.

4.2.2. Adversarial attacks on DRL. There have been numerous studies on the adversarial attack models on the DNN policies of DRL agents. Adversarial attacks targeting DNNs are generally applicable to DRL agents. However, most of the DRL attack models are not applicable to DRL-TSC settings because it requires access to multiple parts of learning agent such as state, action and rewards and directly accessing the DRL-TSC components are challenging.

One of the earlier generative adversarial attack [**195**] targets the DNN classifier by perturbing the input data. The attack model is designed with constrained minimization approach using L_2 norm. Another constraint optimization adversarial attack for image classification task is proposed in [**27**]. Gradient-based adversarial attack models have promising results on DNN classifiers. Two well know gradient based adversarial attacks are FGSM [**76**] and JSMA [**163**] which deteriorate the performance of DNNs by crafting data input geared towards confusing the neural networks. These discussed adversarial attacks are know as the state of the art sequential adversarial attacks mainly proposed for DNNs.

Authors, in [**116**], presented a strategic attack reducing the number of attack times for DRL agents using random noise and FGSM attack strategies. With the transferability of neural networks, similar attack concepts can be extended to black-box attacks [**162**] and can target directly the DRL agents [**16**]. Since DRL agents estimate state values or policy values using DNNs, they are also vulnerable to adversarial attacks with white-box attack settings [**102**] and black-box attack settings [**16**]. A sequential adversarial attack for DRL agents is proposed in [**198**] in which adversarial samples are generated using adversarial transformer networks [**10**] on white-box attack strategy. Another strategic timing and target specific adversarial attack model for DRL agents is presented in [**132**]. The authors perturbed the input states selectively to reduce the visibility of attacker while achieving higher attack performance. Similar to our black-box attack settings, the authors in [**15**] injects perturbations from imitatively learned black-box model. There are also other adversarial attack models which are specific to application areas such as multi-agent robot interactions and path findings [**38, 70**].

4.2.3. Defense models for DRL. There are multiple defense options for the DRL agents including adversarial training, defensive distillation and adversarial detection. Adversarial training idea trains the learning model with adversarial samples that makes the learning model more robust. Several adversarial training-based defense mechanisms are exist in literature for DRL agents [**84**, **116**, **166**]. However, adversarial training is attack dependent and it is easy to fool the model with a different attack strategy. Another defense model is called defensive distillation that trains the DRL policy with a different DNN model and transfers pre-trained soft-max layer from the other trained model to increase the robustness of DRL agent [**164**]. However it is already proven that bypassing the defensive distillation method is easy with various techniques [**27**]. The other security model, which is more aligned with our proposed detection model, is adversarial detection that distinguishes the adversarial samples from the clean samples without modifying the DRL model. One of the earlier adversarial attack detection mechanism for DRL agents is proposed in [**133**] where defense mechanism detects the adversarial samples and suggests alternative actions for the DRL agent instead of the wrong action. A DNN-based adversarial sample detection model for DNNs is presented in [**146**]. The adversarial samples are classified and rejected by DNN models using the autoencoder reconstruction error similar to the robust autoencoder model [**234**].

Statistical properties of input data susceptible to divergence after the perturbation. The study in [**79**] analyzes two statistical distance measures maximum mean discrepancy and energy distance for detecting adversarial samples against several adversarial attacks including FGSM and JSMA. There are several adversarial detection models for DNN classifiers applicable to DRL agents [**18**, **59**]. Sophisticated adversarial detection models for DRL agents are also proposed in literature [**63**, **88**].

4.2.4. Summary. To date, there remains a limited understanding of the security vulnerabilities of learning-based ITS controllers and their impact on various operational performance metrics. In our project, we experimented another research direction of ITS security where we characterize the security vulnerabilities of TSCs when implemented with DRL model and proposed a novel statistical detection model. Main stream adversarial attack models continuously injects adversarial samples to the learning models and expects to fool the model quickly. To protect the DRL-TSC learning model we propose to use statistical sequential detection models with a novel ensemble detection algorithm that achieves to the best detection performance in all cases.

4.3. Overview of DRL-based Traffic Signal Controllers

4.3.1. Deep Reinforcement Learning. Reinforcement learning (RL) is a trial-and-error based learning algorithm where agent interacts with the environment and takes action to maximize cumulative reward. Mathematical formulation of RL is based on Markov Decision Process (MDP). In general RL agent interacts with environment and receives a numerical positive reward (penalty if it is negative). Continuously observing the state of the environment defined by s_t, taking action a_t, and receiving reward (or penalty) from the environment r_t, RL agent learns an action policy which defines how to behave by computing action value function $Q(s_t, a_t)$ after each iteration. In high dimensional environments, RL agent cannot estimate this action value functions easily. Through non-linear approximation, deep learning can estimate this function easily. Controlling RL agents with deep neural network based function approximations is called DRL. In this section, we explain two popular DRL algorithms, DQN and A2C.

4.3.1.1. *Deep Q-Network.* Deep learning extracts the features from data with multi-layered neural networks. Tabular Q-learning method stores every state-action pair in a q-table, however, controlling agents in high dimensional systems with tabular methods is not tractable. The pioneering algorithm called Deep Q-Network (DQN) approximates state-action value function $Q(s_t, a_t)$ using non-linear DNN models, which maps N dimensional state inputs to M dimensional actions (output). RL agent selects the best action from the output of DNNs [**149**] using Q-learning concept. Using DNNs for function approximation sometimes result in unstable learning performance. To ease this problem, temporal difference and batch learning techniques are used. DRL agent is controlled with target network every k steps by updating the main network with respect to target network. The agent may get stuck in a local optimal point due to recent trajectories and by randomly sampling stored experiments, DRL agent learns how to behave from a broad range of experiences.

4.3.1.2. *Advantage Actor-Critic.* Another main approach estimates policy function with gradient methods instead of estimating value function. However, policy gradient algorithms are not effective in large scaled applications due to high variance of the policy estimation. A general solution to this problem is to combine policy and value functions with an advantage function using two individual estimators, where the agent's behaviour is controlled with policy and the actions are balanced with value functions. These models are referred to as actor-critic RL. Synchronously updating both actor and critic estimators is known as advantage actor-critic (A2C) RL.

4.3.2. Deep Reinforcement Learning for TSC. In this section, we will discuss relevant DRL settings for single-agent and multi-agent settings. First, we will explain state, action and reward definitions, and then we will explain our collaboration technique for multi-agent RL model.

In this application, the state of the environment is described a vector of values for each incoming lane of the intersection. For one intersection, we created two valued vectors for each lane: one is average speed and the other is total number of vehicles. Position and speed of each vehicle can be collected from individual vehicles for calculating average speed and number of vehicles using V2I communication. Based on the information received from vehicles, the DRL agent in TSC selects a green phase from among possible green phases. The TSC at a single intersection (such as Fig. 4.1) has four possible green phases: North-South Green (NSG), East-West Green (EWG), North-South Advance Left Green (NSLG), and East-West Advance Left Green (EWLG). Each selected green phase is executed after a yellow phase transition. With the objective of maximizing cumulative reward, a scalar reward is computed after each action (phase selection in this case). There are several reward definitions for TSC settings such as vehicle waiting time, cumulative delay, and queue length. In our DRL-based TSC, we used the change of the vehicle waiting time at an intersection for one cycle as a reward function.

As mentioned earlier, applying deep learning techniques to RL can help compute the action value functions more efficiently. For DRL models, designing a neural network structure for better performance is another critical step. Multi-layer perceptron (MP), i.e., the standard fully connected neural network model, is a useful tool for classic data classification. In this project, we used MP with 4 layers in DQN and 5 layers in A2C with relu and softmax activation functions for policy estimations of learning agents.

To test more general cases in DRL-based TSCs, we also studied a multiple intersection scenario with multi-agent RL settings where interaction among agents is necessary to reach a global optimum performance. In multi-agent settings, each agent updates its policy by including the current state and reward functions of neighbor TSCs as well to decrease the overall delay in traffic. For this purpose, global state is found with concatenation of the local states of neighboring intersections and reward is generated by summing the local rewards of neighboring intersections.

4.4. Adversarial attacks on DRL

In data-driven learning algorithms, function estimator tunes the parameters precisely and carefully with respect to the training set. An adversary can manipulate the training set by injecting falsified data into the system. A smart way of attacking the learning agent is to inject carefully-crafted fake data that has very similar patterns with actual data. In white box attack model, the adversary has knowledge of the exact learning model and the corresponding output classes, and will manipulate the input to mislead the model. In black box attack model, the exact learning model is not known but the adversary can estimate a similar learning model to help generate input perturbation that can affect the target learning model.

In DRL controller, DNN function estimator, which estimates the action with respect to given state, is the most probable adversarial target. The objective of the adversary is to craft the data input in order to lead DNN to a wrong action. When the DNN of DRL is under attack, it may select an incorrect action. For targeting DRL-based controllers, adversarial attacks can be launched sequentially at every time step to mislead the system as quickly as possible or strategically at specific time steps to hide itself from the controller center. In this study, we simulated sequential FGSM and JSMA attack strategies on DRL-based TSCs, which plays a critical role in traffic management systems. The threat model of adversarial attacks on DRL-TSCs is shown in Fig 4.1.

4.4.1. Fast Gradient Sign Method. A clever attack model, fast gradient sign method (FGSM) introduced in [**76**], calculates the gradient of the cost function with respect to DNNs to maximize the perturbation using the L_∞ distance. Adversarial input is generated by adding generated adversarial data to the input state as follow:

$$\eta = \epsilon * \text{sign}(\nabla_x J(\theta, \boldsymbol{x}, a)) \tag{4.1}$$

where ϵ is the attack magnitude, J is the cost function of DNN, and θ is the model parameters. ∇_x refers to the gradient of the cost function related to model input state $\boldsymbol{x}$, and true action a.

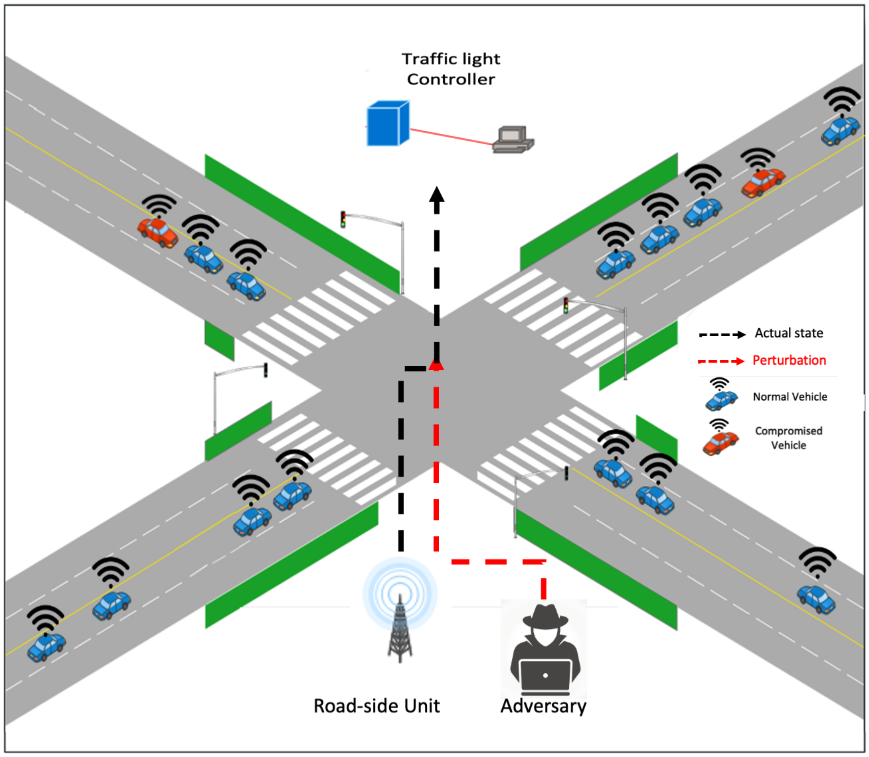


FIGURE 4.1. TSC is controlled with a DRL agent and an adversary that can attack the agent with falsified data which perturbs the input state. While adversary can input directly for FGSM attack, it can use compromised vehicles for JSMA attack.

The FGSM attack designed to be fast and effective by generating infinitesimal perturbation that is close to the true input with perturbation parameter e.g., $\epsilon = 0.007$. FGSM attack model is an untargeted where attacker do not specifies the target action when FGSM is launched. The optimal perturbation η satisfies $||\eta||_\infty < \epsilon$.

The perturbation amount η is added to the input data x:

$$\boldsymbol{x}_{adv} = \boldsymbol{x} + \eta. \tag{4.2}$$

In DRL-TSC, FGSM attack perturbs all the input features with very low values, therefore, launching this attack from the communication network requires to modify all the state dimensions that corresponds to each traffic lanes. The attack model assumes that the attacker has access to the input gate of DRL agent. By using this gate, attacker perturbs the input state $\boldsymbol{x}$ right before it goes into the DNN where Q values for each action is estimated. Launching FGSM with the black box settings is also possible. In this case, the attacker will not be able to access to the DRL agent directly, it only has access to the data pipeline of the DRL agent.

4.4.2. Jacobian-based Saliency Map Attack. Another attack model utilizes forward derivative, jacobian based saliency map attack (JSMA), presented in [**163**]. The intuition of JSMA attack is to find the influence of each state feature $\boldsymbol{x}_i$ to a specified output action a and then perturb only those specified feature dimensions. This influence relies on the jacobian matrix of outputs with respect to each action taken by the DRL agent using the forward gradient of the DNNs to construct adversarial saliency maps.

The adversary can control which input feature to perturb with respect to constructed saliency maps to achieve desired goal. In this attack model, attacker selects a target action for the DRL agent where the output of the DNN is Q values for each action. With greedy mechanism, action is selected from the DNN during the test phase with respect to given state $\boldsymbol{x}$ as:

$$a_t = \underset{a}{\operatorname{argmax}}\, Q(\boldsymbol{x}, a) \tag{4.3}$$

where a_t refers to the selected action by thr DRL agent at time t.

In our case, adversary tries to mislead DRL agent to select wrong action and for this purpose, the output Q value for the desired action should be increased. The Q values are the probabilities of corresponding actions. The adversary can increase the desired Q values estimated through DNNs by using the saliency map:

$$S^{+}(x_{(i)}, a) = \begin{cases} 0 \text{ if } \frac{\partial f(\boldsymbol{x})_{(a)}}{\partial x_{(i)}} < 0 \text{ or } \sum\limits_{a' \neq a} \frac{\partial f(\boldsymbol{x})_{(a')}}{\partial x_{(i)}} > 0 \\ \left(\frac{\partial f(\boldsymbol{x})_{(a)}}{\partial x_{(i)}}\right) \left| \sum\limits_{a' \neq a} \frac{\partial f(\boldsymbol{x})_{(a')}}{\partial x_{(i)}} \right| & \text{otherwise} \end{cases} \tag{4.4}$$

where i is the input feature of state $\boldsymbol{x}$, a is the action corresponding to the input, and a' is the other actions of DRL agent. In Equation 4.4, the first line of the expression rejects the negative target derivative with respect to action a and positive derivatives with respect to other actions a' of input state $\boldsymbol{x}$ feature i. The second line of Equation 4.4 extracts the positive forward derivative of state $\boldsymbol{x}$ of feature i given the action a. Based on the constructed silency map, adversary selects which input feature to perturb in order to mislead the agent for selecting the wrong action. Higher $S^{+}(x_{(i)}, a)$ values mean the attacker can more easily determine if increasing this feature either increase the Q value of the target action a or decrease the Q values of other actions. In the JSMA model, the attacker first selects which action to perturb randomly then based on that selected action it creates the saliency map. Using the saliency map attacker finds the best features to perturb.

The threat model for JSMA attack is different from the FGSM attack. Since JSMA perturbs specific features based on the saliency map, it is possible to launch this attack by compromising the communication between vehicles and TSC unit. In this attack model, attacker can use compromised vehicles and/or Sybil vehicles to broadcast falsified information in order to increase or decrease the corresponding feature dimension values.

4.5. Sequential anomaly detection for DRL-TSCs

The attackers can exploit wide range of vulnerabilities in DRL-TSCs, and attack patterns are generally unpredictable. Therefore, it is hard to model a defence mechanism for a broad range of anomalies. Besides, defining a parametric model, which tries to fit a probability distribution to the data, is not practical. Due to life threatening effect of misbehaved DRL-TSCs, it is critical to detect and mitigate adversarial attacks in a timely manner. Considering the major challenges in DRL-TSC, non-parametric sequential anomaly detectors are suitable for detecting streaming anomalies in online settings. There are three main reasons why we employed a non-parametric sequential statistical anomaly detectors for adversarial attacks on DRL-TSCs: (i) consecutive adversarial samples are more harmful for DRL controllers and need to be detected quickly, (ii) standard outlier detectors are susceptible to false alarms due to not considering temporal correlations in data, (iii) non-parametric sequential detectors have less miss-match error that results in lower detection error.

Statistical anomaly detectors operate by comparing the summary statistics extracted from the training set in offline phase and summary statistic of data in online phase for detecting potential anomalies. Since no single statistical property captures all anomaly types, we present a sequential anomaly detection model that extracts multiple summary statistics and leverages an ensemble model for online test phase. In this section, we first explain three summary statistic extraction models that are distance-based, PCA-based and Robust Autoencoder-based and present online sequential detection algorithm.

Let us first explain the data representation that is used for the rest of the chapter. The monitoring system observes d dimensional each data instance $\{\boldsymbol{x}_i^1, \ldots, \boldsymbol{x}_i^d\}$ that forms a set of nominal streaming data $\mathcal{X} = \{\boldsymbol{x} : j = 1, 2, \ldots, N\}$. Depending on the TSC setting and DRL model the size of d can change. In our experiments, DRL collects the summary statistics from each lane and forms d dimensional state information $\boldsymbol{x}_t$ at time t.

4.5.1. GEM-based Summary Statistic. Geometric Entropy Minimization (GEM) method defines an acceptance region for the offline training set based on the nearest neighbor statistics with respect to significance level α [**98**]. GEM-based computationally efficient summary statistic extraction method using bipartite kNN graph is presented in [**191**]. In the training phase summary statistic extracted as described in the following.

We begin with randomly partitioning the anomaly free dataset $\mathcal{X}_N$ into two subsets $\mathcal{S}_1$ and $\mathcal{S}_2$ with sizes N_1 and N_2 where $N = N_1 + N_2$. Then, for each data point $\boldsymbol{x}_j \in \mathcal{S}_1$, we find the kNN euclidean distance e_j from $\mathcal{S}_2$. Sum of the distances of $\boldsymbol{x}_j$ to its nth nearest neighbor in $\mathcal{S}_2$ can be denoted as:

$$d_j = \sum_{i=1}^{k} e_j(i). \tag{4.5}$$

Once $\{d_j : \boldsymbol{x}_j \in \mathcal{S}_2\}$ is computed and sorted in ascending order, we refer to this baseline set as $\boldsymbol{D}_{GEM}$.

4.5.2. PCA-based Summary Statistic. High dimensional observation may exhibit sparse data structure so underlying independent data dimension can be lower than the actual data dimension. When we represent data $\boldsymbol{x}_j$ in lower dimension as $\boldsymbol{y}_j$, the remaining parts $\boldsymbol{r}_j$ is the residuals. Adversarial noise injected to the actual data is mainly represented in residuals $\boldsymbol{r}_j$, hence the magnitude of the residuals $\|\boldsymbol{r}_j\|_2$ expected to be higher than normal data. Recently a PCA-based online anomaly detection model is proposed in [**119**]. Based on this intuition, and the same partitioning strategy, we follow the PCA-based training steps for set $\mathcal{S}_1$.

(1) Compute the sample mean $\bar{\boldsymbol{x}}$ and sample covariance matrix $\mathcal{Q}$

(2) Then, compute the eigenvalue $\{\lambda_j : j = 1, 2, ..., p\}$ and the eigenvectors $\{\boldsymbol{v}_j : j = 1, 2, ..., p\}$ of $\mathcal{Q}$

(3) Determine the dimension of $\boldsymbol{y}_t$, r, with respect to the desired level of data variance γ,

(4) Form the eigenmatrix corresponding the largest r eigenvalues $\lambda_{1,}\lambda_2, ..., \lambda_r$: $\boldsymbol{V} \triangleq [\boldsymbol{v}_1, \boldsymbol{v}_2, ..., \boldsymbol{v}_r]$

(5) Compute the residual term $\boldsymbol{r}_{j-PCA}$ for every sample $\boldsymbol{x}_j$ in set $\mathcal{S}_2$ as follows:

$$\begin{aligned} \boldsymbol{y}_j &= \bar{\boldsymbol{x}} + \boldsymbol{V}\boldsymbol{V}^T(\boldsymbol{x}_j - \bar{\boldsymbol{x}}) \\ \boldsymbol{r}_{j-PCA} &= \boldsymbol{x}_j - \boldsymbol{y}_j \\ &= (\boldsymbol{I}_p - \boldsymbol{V}\boldsymbol{V}^T)(\boldsymbol{x}_j - \bar{\boldsymbol{x}}) \end{aligned} \tag{4.6}$$

(6) Finally form the residual term vector $\boldsymbol{D}_{PCA}$ with $\{\|\boldsymbol{r}_{j-PCA}\|_2$: $\boldsymbol{x}_j \in \mathcal{S}_2\}$ in ascending order.

4.5.3. Robust Deep Autoencoder Summary Statistic. A deep autoencoder-based noise and outlier extraction technique is proposed in [**166**] as an unsupervised Robust Deep Autoencoder (RDA) anomaly detection algorithm. The proposed RDA learns the normal data behaviours with a regularization penalty term using different norms. The idea of the RDA combines the powerful nature of the Robust PCA model [**23**] with autoencoders that recovers low dimensional $\boldsymbol{y}_t$ iteratively by removing the residuals $\boldsymbol{r}_t$ from the data $\boldsymbol{x}_t$.

Training procedure of the RDA-based summary extraction model starts with pre-training the model with the sample set $\mathcal{S}_1$. After pre-training the model with certain number of episodes, which is 10 in our experiments, RDA is trained with sample set $\mathcal{S}_2$ and summary statistic $\boldsymbol{D}_{RDA}$ is formed from $\|\boldsymbol{r}_{j-RDA}\|_2$ as a baseline.

4.5.4. Sequential Anomaly Detector. In the test phase, summary statistics d_{t-GEM}, $\|\boldsymbol{r}_{t-PCA}\|_2$ and $\|\boldsymbol{r}_{t-PCA}\|_2$ of each anomaly detection model is found for the new data point $\boldsymbol{x}_t$ independently. The anomaly score expected to be higher in the case of adversarial attack. Since the procedure is the same for all three models, we explain the remaining anomaly statistic extraction algorithm for the GEM model as an example. For a new data point $\boldsymbol{x}_t$, once d_{t-GEM} summary score is computed using (4.5), tail probability of p_t would be computed with respect to baseline set $\boldsymbol{D}_{GEM}$ as follow:

$$p_t = \frac{1}{N_2} \sum_{\boldsymbol{x}_j = \mathcal{S}_2} \mathbb{1}\{d_j > d_{t-GEM}\} \tag{4.7}$$

which shows the fraction of the baseline summary statistics $\boldsymbol{D}_{GEM}$ greater than d_t. Given the significance level α, we can get a real valued statistical score in log scale with

$$s_{GEM} = \log(\frac{\alpha}{p_t}), \tag{4.8}$$

if the tail probability $p_t < \alpha$, we can consider $\boldsymbol{x}_t$ as an outlier. We follow the same approach in equations (4.7) and (4.8) to calculate s_{PCA} and s_{RDA} scores. Since the three scores are independent from each other, they can be calculated in parallel. For extracting the final anomaly score, we sanitized the three anomaly scores using a simple averaging as follows:

$$s_t = \frac{1}{3} \sum (s_{GEM}, s_{PCA}, s_{RDA}) \tag{4.9}$$

Note that the anomaly scores s_t can be positive or negative values with respect to the existence of anomalies. Instead of sample-by-sample anomaly declaration we propose to use model-free CUSUM-like anomaly detection approach [**14**]:

$$\begin{aligned} g_t &\leftarrow \max\{0, g_{t-1} + s_t\},\ g_0 = 0 \\ \mathcal{T} &= \inf\{t : \max\{0, g_t \geq h\} \end{aligned} \tag{4.10}$$

where g_t refers to the decision statistic. The anomaly is declared if enough sequential anomaly evidence is accumulated. The detection threshold h is chosen to strike a balance between minimum detection delay and lower false alarm rate. While lower detection threshold h results in lower detection delay, it enables higher false alarm rate. The summary of the proposed anomaly detection technique is shown in Algorithm 2. The proposed sequential anomaly detector is also robust against system misbehaviour due to the nature of cumulative anomaly detection model.

```
Algorithm 2 Proposed Nonparametric Anomaly Detection
                    Offline Phase
Partition the training set $\mathcal{X}_N$ into two subsets $\mathcal{S}_1$ and $\mathcal{S}_2$ with sizes $N_1$ and $N_2$.
Compute GEM baseline set $\boldsymbol{D}_{GEM} = \{d_j : \boldsymbol{x}_j \in \mathcal{S}_1\}$
Compute PCA baseline set $\boldsymbol{D}_{PCA} = \{\|\boldsymbol{r}_{j-PCA}\|_2 : \boldsymbol{x}_j \in \mathcal{S}_2\}$
Compute RDA baseline set $\boldsymbol{D}_{RDA} = \{\|\boldsymbol{r}_{j-RDA}\|_2 : \boldsymbol{x}_j \in \mathcal{S}_2\}$
                Online Detection Phase
Initialization: $t \leftarrow 0$, $g_0 \leftarrow 0$.
while $g_t < h$ do
    $t \leftarrow t + 1$.
    Obtain the new data point $\boldsymbol{x}_t$.
    Compute statistic $s_{GEM}$, $s_{PCA}$ and $s_{RDA}$
    Form ensemble statistic $s_t$ with averaging as in (4.9)
    $g_t \leftarrow \max\{0, g_{t-1} + \hat{s}_t\}$.
end while
Declare an anomaly and stop the procedure.
```

4.6. Adversarial Attack Performance

In this section, we evaluated the impact of adversarial attacks on DRL-based TSCs using SUMO [**138**] real-time vehicular traffic simulator, with Tensorflow Python API[1] for DRL-based controller and CleverHans Python API for adversarial input generation built upon Tensorflow [**161**].

[1] Allows to create and train ML models without loss of speed or performance.

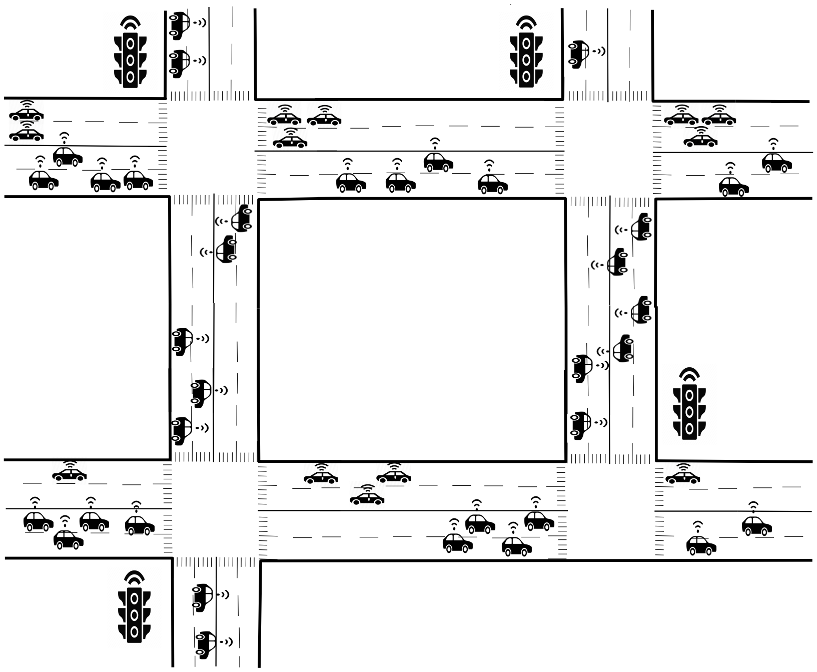


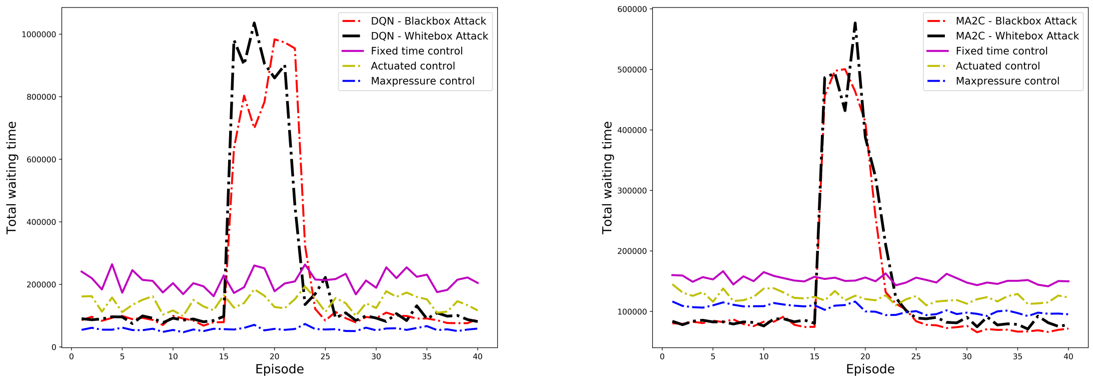
FIGURE 4.2. Traffic scenario for multi-agent multi-intersection TSCs.

We simulated both a single-intersection and multi-intersection environments with DQN and A2C DRL-based TSCs. For single-intersection scenario, we observed similar results for DQN and A2C DRL-based TSCs. For the rest of the chapter, we only present results for DQN for the single-intersection case. Value-based DQN approaches do not perform well for large environments. Therefore, we only simulated multi-agent A2C model based RL controllers for multiple intersections traffic scenario organized in 2x2 grid topology. The structure of each of the 4 intersections is shown in Fig 4.2. The DRL agent selects among four possible green phases as described in Section 4.3.2. The traffic is generated with the arrival rate of one vehicle per second spanning 1-hour simulation time. For each arrival, travel route is assigned with random origin and destinations selection.

We implemented FGSM and JSMA adversarial attacks for both white-box and black-box attacks. One technical challenge we faced is the lack of computational resources to launch these adversarial attacks continuously (for more than 5 episodes), as it requires high memory footprints due to the batch gradient of the NNs[2]. All our experiments compare the performance of DRL TSCs with three baselines. One of the baselines is standard fixed time TSC where traffic lights are allocated to different phases with pre-defined durations. We also compared our method with two adaptive controller methods: queue-based actuated TSC, and max-pressure-based TSC [**214**]. Maximum phase duration for both actuated controller and max-pressure controller is set to be 45 seconds. All the attacks experimented in this study starts after 15 episodes and the attack continues for 5 episodes, where every episode spans one hour of traffic simulation. After the attack terminates, we observed the performance of the learning agent for an additional 20 episodes. In the absence of attack, DQN achieves the second lowest total waiting time (only slightly inferior to Maxpressure) for the single-intersection case while multi-agent A2C model achieves the lowest total waiting time for multiple-intersection scenario.

4.6.1. White-box Insider Attack. Regardless of the DNN structure, learning models are vulnerable to white-box adversarial attacks, even with a very slight perturbation on input data. White-box adversarial attacks assume that the attacker has access to the target model of learning policy.

[2] We employed transfer learning while simulating adversarial attacks on both single-intersection DQN and multi-intersection A2C scenarios. We saved the NN model weights after training the agents, and launched the attack using the latest NN weights. We repeated this for each attack episode.



(a) FGSM attack for single-intersection DQN model. (b) FGSM attack for multiple intersections Multi-agent A2C model

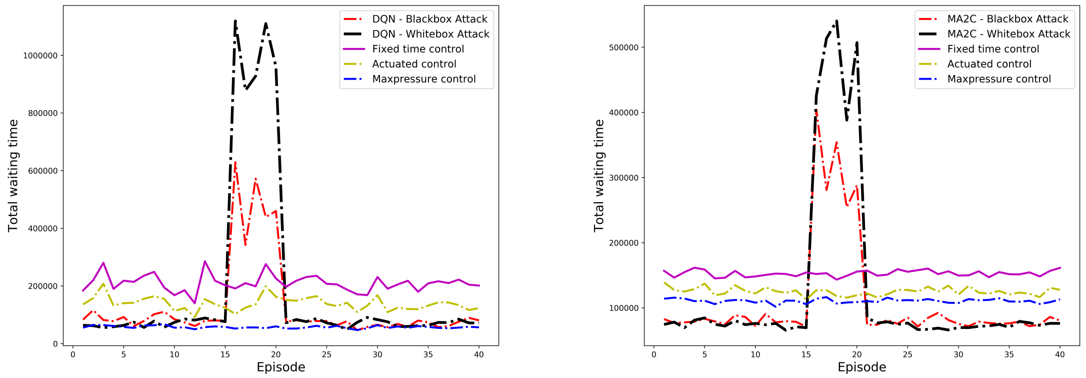
Figure 4.3. FGSM White-box and black-box attack results for DQN and multi-agent A2C with 0.007 attack magnitude using FGSM attack model. Attack continues 5 episodes from 15 to 20. Both white-box and black-box attacks continuously effects the performance of DRL agent while attack continues.

4.6.1.1. *Attack Model.* Using the target model, launching an adversarial attack with FGSM and JSMA models on the DNN of RL agents is possible.The adversary launches the attacks on DRL-based TSCs by injecting anomaly to the original input state. Since DNN is the policy of a learning agent, selecting correct action of the DRL agent will be affected by the white-box attack.

For FGSM attack, an attacker will perturb the input state with very small changes that are invisible by the controller. As pointed out in the original FGSM paper [**76**], minimal perturbation leads to the DNN to classify output to a wrong class. We used the same attack magnitude $\epsilon = 0.007$ as in the original FGSM attack [**76**] for DQN and A2C TSC simulations.

For JSMA attack, the attacker constructs the saliency map of given input state with respect to randomly selected action using the forward gradient of the DNN. In this attack model, we found that the attacker needs to perturb at least 40% of the feature dimensions to mislead the DRL agent, hence, we selected $\gamma = 0.4$ as an input parameters for our experiments.

4.6.1.2. *Results.* Fig. 4.3 and Fig. 4.4 show the results from FGSM and JSMA, respectively. After the attack is launched, both DQN and A2C TSCs perform poorly during the attack duration with FGSM and JSMA attacks. Although DRL settings are different, single-intersection TSC (Fig. 3(a) and 4(a)) and multi-agent multi-intersection TSC (Fig. 3(b) and 4(b)) are both affected, and the



(a) JSMA attack for single-intersection DQN model

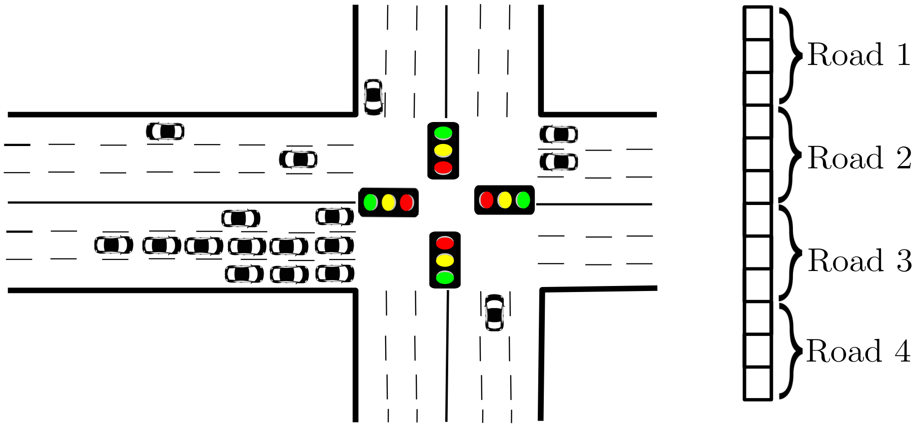
(b) JSMA attack for multiple intersection multi-agent A2C model

FIGURE 4.4. JSMA attack continues with 10% of data perturbation for single agent DQN model and with 40% of data perturbation for multi-agent A2C model. The attack injects falsified data by selecting specific lanes of the intersection. The attack starts at episode 15 and lasts for 5 episode and ends in episode 20.

total waiting time in the traffic exceeds even the fixed-time controller. While the total waiting time increases almost 10x for single-intersection, it increases almost 6x for multi-intersection immediately after the white-box FGSM and JSMA attacks are launched. DRL agents cannot respond to these attack models and the attack continuously effects the learning agents as long as the DRL agent is targeted because the DQN and A2C agents do not recognize the attacks. For FGSM attack, the total waiting time decreases to pre-attack levels in 5 episodes after the attack ends in both the single-intersection DQN and the multi-intersection A2C cases. On the other hand, for JSMA attack, the total waiting time decreases to the pre-attack levels immediately right after the attack ends.

4.6.2. Black-box External Attack. In black-box attack scenario, the attacker does not have a precise knowledge about the model. Here, we investigate the vulnerability of the DNN policies for DRL-based TSCs when the attacker does not have access to the actual target model.

4.6.2.1. *Attack Model.* The transferability of trained DNNs allows attacker to train a separate learning model and use it to generate adversarial perturbation. Both FGSM and JSMA adversarial attacks require knowledge of DNNs for calculating gradients regarding to the DNN policy. Practically it is not hard to train a separate policy for TSCs using real traffic maps on traffic simulators, and an



Traffic intersection for a 4 way intersection

Feature-based state vector for one intersection

FIGURE 4.5. Feature-based vectorized state representation.

attacker can do this training at a very low cost. In this work, we are proposing a practical black-box attack strategy where the attacker uses the same number of layers for training a different DNN policy as the original learning agent. Also, for training a separate DNN, the attacker considers linear activation functions instead of the ReLU and Random Uniform DNN initialization technique instead of Glorot initialization [**71**]. We assumed that attacker is not able to predict true travel demand on the simulator. Therefore, we trained our adversarial policy with slightly different traffic demands. Since we simulate the same adversarial attacks with black-box attack settings, to have a precise comparison, we kept the same attack magnitudes as $\epsilon = 0.007$ for FGSM attack and $\gamma = 0.4$ for JSMA attack similar to white-box attacks.

4.6.2.2. *Results.* The results of black-box adversarial attacks on DRL TSCs have similar patterns with the white-box attacks for FGSM attack model. However, the impact of the JSMA attack decreases to the half compared to white-box JSMA attacks in terms of the total waiting time. The results for the three baseline TSCs are almost identical across two adversarial attack models. Red dashed lines in Fig. 4.3 and Fig. 4.4 shows the adversarial attack results for DQN and multi-agent A2C under black-box attacks. Similar to the white-box settings, DRL agent is severely impacted by the attack resulting in average 9x and 6x increase in total waiting time in single and multi-intersection scenarios respectively during the FGSM attack. The black-box JSMA attack increases the total waiting time 5x and 3x for single intersection and multi-intersection scenarios likewise

FGSM attack. The impact of the attack continues throughout 5 attack episodes by performing worse than the other three control methods in both attack cases. Similar to the white-box attack case, while the recovery period of DRL agent under FGSM attack is about 4 episodes after the episode 24th, DRL agent recovers itself immediately after the attack terminates for JSMA attack.

4.7. Adversarial Detection Performance

After showing the vulnerability of DRL-TSCs against adversarial attacks, we evaluate the proposed statistical anomaly detection model on DRL-TSCs. The nature of adversarial attack for white-box and black-box attack settings are almost the same in terms of the data perturbation. Therefore, in this section, we only evaluated the detection performance on white-box FGSM and JSMA adversarial attacks on single intersection DQN-TSC and multi-intersection MA2C-TSC (see Fig. 4.2). We also use the same attack magnitudes as described in Section 4.6 for evaluating the performance of statistical detectors.

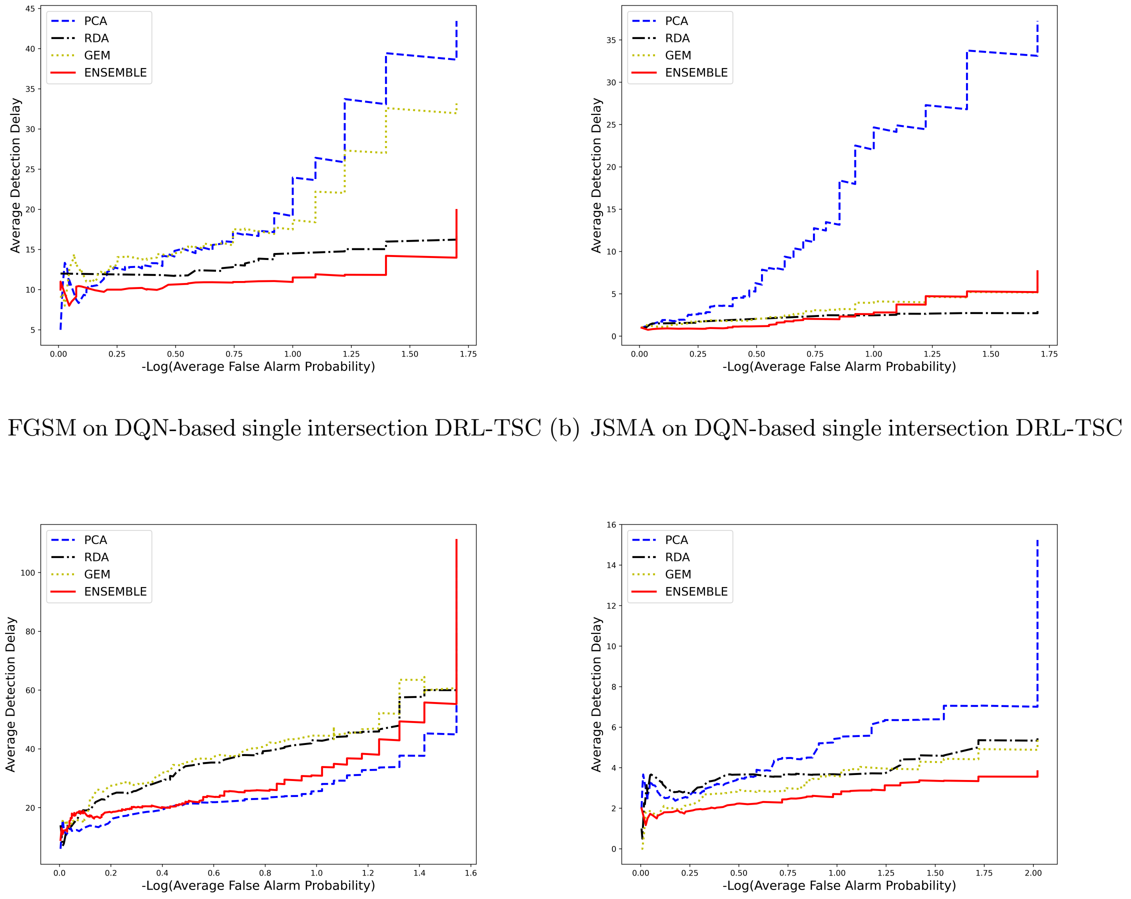
For evaluating the ensemble statistical detection performance, we compare the proposed algorithm with individual adversarial detectors PCA, RDA and GEM models. We use the same CUSUM-like detection structure on each model. Note that each anomaly detection algorithm is most effective in recognizing different anomaly types. While noise injections on all input vectors such as FGSM attacks can be detected by PCA anomaly detection model easily, selective perturbation-based anomalies such as JSMA can be detected with RDA and GEM models effectively.

We quantified the detection performance in terms of three metrics. Quick and accurate detection performance is presented with average detection delay vs false alarm rates, which is our first result representation. Later, we present the performance of sequential detectors on ROC (Receiver Operating Characteristics) curve and AUC (Area Under The Curve) scores which are the two leading performance metrics for classification tasks. While ROC is the probability curve for true positive rate vs false positive rate, AUC score quantifies how much the model is capable of distinguishing between classes.

4.7.1. Sequential Detector Setup. To generate training and test sets for sequential detectors, we collect anomaly-free training states and test sets that include anomalies from the DRL-TSCs. For single intersection TSC model, the DRL setup has relatively low dimensional state format since each lane corresponds to two dimensions in state vector which are the number of vehicle and average speed. This form of state known as feature-based vectorized state representation (Se Fig. 4.5). The number of vehicle and average speed per lane states are concatenated to form final state representation. For example, the state definition for our single intersection DQN-TSC , which has 4 incoming roads with a 4 lane single intersection, is 32 units column vector.

Regarding the single-intersection DQN, sequential detectors are trained with 1 episode of anomaly-free traffic flow. Then, the detectors are trained on FGSM and JSMA adversarial attacks using 50 test episodes where adversarial attack starts after 200 state samples. Regarding the multi-intersection MA2C, we followed similar data collection procedure with slight changes. In our MA2C model, every intersection has a different number of approaching lanes, therefore, the state dimensions varies in MA2C model. We have 3 group of state representation for 4 intersections as 82, 86, 92. After collecting neighborhood information, two of four intersections have the same size of state dimensions. Due to having different state dimensions, each agent of MA2C model is trained and tested separately, then, test results are concatenated. Adversarial attack for 1 episode is highly time consuming. Hence, the number of test samples are relatively low which is 35 MA2C episode. In total, we have 105 test trials for MA2C-TSC model.

4.7.2. Results. Fig. 4.6 shows average detection delay vs false alarm probability results for the proposed ensemble model compared with the other statistical anomaly detectors. We observe that the proposed ensemble model has the lowest detection delay vs lower false alarm probability on FGSM attack to the single intersection DQN model (Fig 6(a)) and JSMA attack to multi-intersection MA2C model (Fig 6(d)). The ensemble model also performs closer to the other statistical detectors for JSMA to single intersection DQN (Fig 6(b)) and FGSM to multi-intersection MA2C (Fig. 6(c)). Due to invisible nature of FGSM attack, all detectors have higher detection delays. The proposed ensemble model is the second best detector among all. Except for FGSM attacks on MA2C, the ensemble model detects the adversarial samples within less than 10 samples. This means that the ensemble detector informs the DRL agent within 10 adversarial samples, which is small enough for taking an action against adversarial attack. The ensemble model is able to handle multiple adversarial attack types on different controller settings. The results can be extended to a broader range of adversarial attacks that may target the DRL-TSCs. One proposed mitigation strategy on



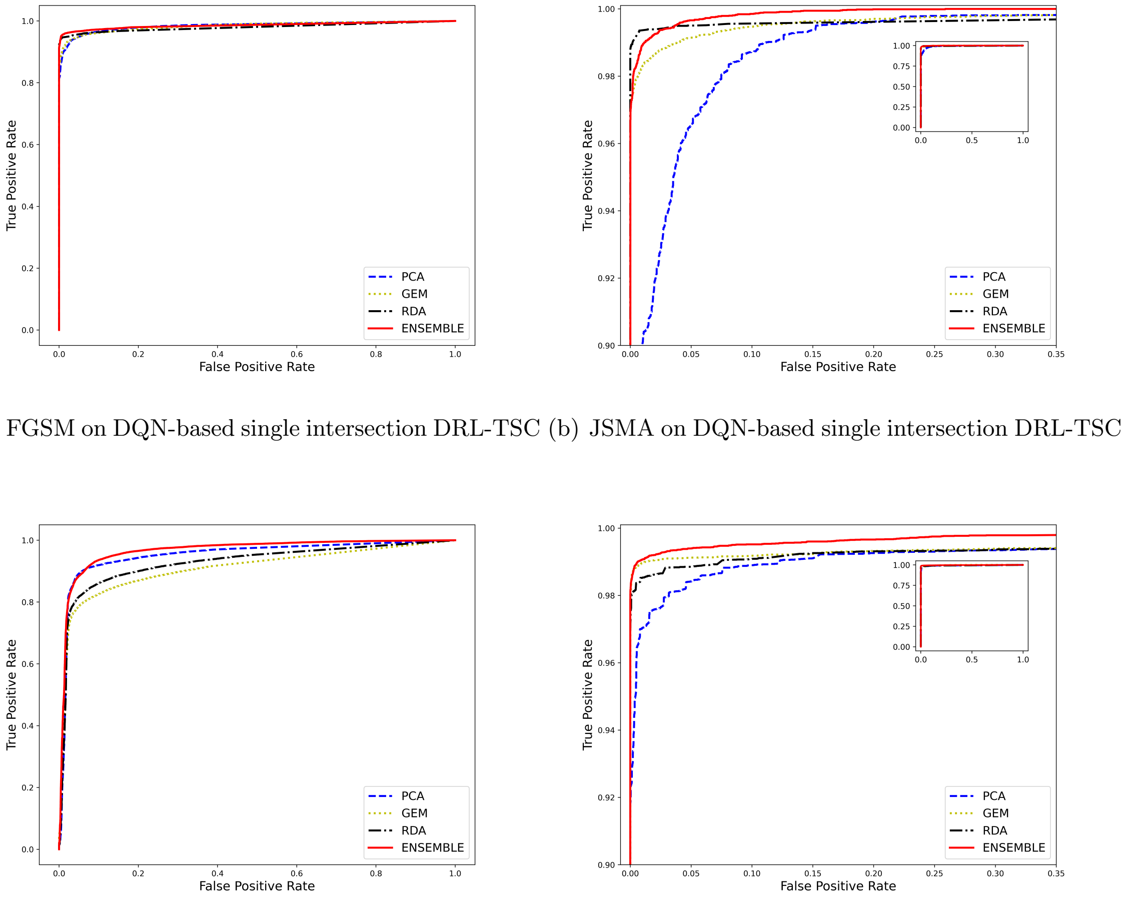
(a) FGSM on DQN-based single intersection DRL-TSC (b) JSMA on DQN-based single intersection DRL-TSC

(c) FGSM on MA2C-based multiple intersection DRL-TSC (d) JSMA on MA2C-based multiple intersection DRL-TSC

FIGURE 4.6. Comparison of sequential detection performances in terms of average detection delay vs false alarm period.

top of detecting the anomalies is switching to another TSC model such as max-pressure TSC after attacks are detected.

Next, we analyzed the overall detection performance with ROC curve and AUC scores. Since anomaly detectors are simple binary classifiers, evaluating the accuracy of anomaly detectors with ROC classifier curve is important where the curve do not assumes any distribution on data for producing classification performance. As depicted on Fig 4.7 and supported by the AUC scores in Table 4.1, the proposed ensemble model outperforms the all other statistical detectors. While the bold statistics shows the best detection performance, the green statistics tells the second best



(a) FGSM on DQN-based single intersection DRL-TSC (b) JSMA on DQN-based single intersection DRL-TSC

(c) FGSM on MA2C-based multiple intersection DRL-TSC (d) JSMA on MA2C-based multiple intersection DRL-TSC

Figure 4.7. ROC curves for different attacks and TSC settings with the proposed anomaly detection model.

detection performance on Table 4.1. It is clear from the green statistics that different statistical anomaly detectors performs differently on different threat models, however, proposed ensemble model has clear advantage over the other detectors with almost perfect detection performance.

4.8. Conclusions

We have demonstrated the impact of adversarial attacks on DRL-based TSCs for a single-intersection and multiple intersection cases using different threat models. First, we evaluated the adversary impact of two adversarial attack models: FGSM and JSMA using white-box, and practical

TABLE 4.1. AUC scores for different baselines for all configurations

TSC-Attack	PCA	RDA	GEM	Ensemble
DQN-FGSM	0.9844	0.9749	0.9839	**0.9895**
DQN-Jacob	0.9950	0.9980	0.9979	**0.9994**
MA2C-FGSM	0.9549	0.9258	0.9028	**0.9637**
MA2C-Jacob	0.9942	0.9951	0.9954	**0.9978**

black-box settings. The results show that the performance of a DRL agent decreases sharply after the attack starts in all attack models and total waiting time increases becoming worse than the standard TSC methods. While, white-box FGSM and JSMA attacks effects the learning performance with similar impact. black-box FGSM attacks has severe impact compared to black-box JSMA attacks. Second, we presented a non-parametric online anomaly detection model which detects different anomalies sequentially by combining three existing anomaly detection models with CUSUM-like algorithm. Through realistic SUMO traffic simulators, we evaluate the online detection performance of various anomaly detection approaches in the presence of adversarial attacks. The results show that the proposed ensemble model achieves superior performance in detecting anomalies in all threat models compared to other existing anomaly detectors.

The proposed study provides a security mechanism for known attack models. However, there are still some limitations that need to be addressed with further studies. While there are many different attack models, vulnerability of DRL-TSCs should be evaluated with more threat models. This work provides a novel anomaly detection model for DRL-TSCs but practical mitigation strategies and internal system robustness mechanisms should also be investigated. For future work, we plan to investigate on other types of adversarial attacks and provide an integration mechanism with the proposed anomaly detection model and internal robustness mechanisms.

CHAPTER 5

Differential Privacy in Aggregated Mobility Networks: Balancing Privacy and Utility

5.1. Introduction

With the proliferation of smartphones, GPS devices, and connected vehicles, geospatial-temporal datasets that capture the movements of these devices have emerged. The trajectories collected from these devices are excellent sources of mobility information for city planners and researchers. However, embedded in these datasets are details of the device owner's mobility patterns. Privacy concerns associated with lifestyle patterns, such as locations of home, work, and user specific points of interest, have inhibited the release of these datasets by the organizations collecting the data. As such, the datasets have remained siloed and their use has been limited to a few organizations.

Typical approaches for dealing with privacy issues, such as k-anonymity or other privacy methods, often result in a very high utility loss for the type of transportation studies that cities need. A naive approach of applying noise injection to the individual GPS points can perturb the trajectory too much, particularly in urban areas where moving GPS point a few blocks away can dramatically change the path (e.g., due to one-way streets or locations of highway exits).

Another approach to dealing with privacy issues is to aggregate the data before releasing the dataset. However, [**220**] has shown that even aggregated outputs may still have privacy issues. The authors were able to re-identify user trajectories from the aggregated trajectories. Additional studies found that examining trajectories that were often repeated indicated unique mobility patterns that could be associated to an individual. Other efforts to privatize mobility data [**170**, **231**], resulted in a high utility loss.

Differential privacy (DP) is a statistical privacy-preserving technique [**52**] that is designed to minimize leakage of information about individuals, while still preserving the characteristic patterns in the data. Differential privacy controls the degree of information that can potentially be exposed.

The DP control parameters can be tuned empirically for specific datasets, and specific application use-cases [**1**, **77**, **153**]. The goal of DP is to ensure that an adversary with background knowledge about the dataset cannot extract private information from the dataset.

The goal of our work is to design a DP-based approach that can achieve location privacy and still maintain the relevant information content for deriving transportation metrics of interest. Our approach focuses on designing a network-aware noise injection algorithm that uses the geospatial constraints to apply noise and privatize the sensitive information. In this paper, we present *a differentially private adaptive noise injection (DP-ANI) model* that generates an aggregated mobility network from raw GPS trajectory data.

The contributions of this paper include:

- We propose a differential privacy-based noise injection (DP-ANI) model that perturbs the origin-destination GPS points in an adaptive manner based on the road network's density.
- We apply the Sparce Vector Technique to select the adaptive range parameters privately.
- We evaluate the impact of the degree of perturbation of the noise injection model by comparing the geospatial statistics derived from the released mobility network after applying our DP-ANI model compared to the raw mobility data.

The remainder of the paper is organized as follows. We present related work in Section 5.2, an overview of the differential privacy in Section 5.3, and our metrics and models in Section 5.4. We then present our differentially private adaptive noise injection (DP-ANI) model in Section 5.5. After evaluating the experimental results of our privacy model in Section 5.6, we discuss the limitations of this work in Section 5.7. Finally, Section 5.8 concludes the paper.

5.2. Background and Related Work

5.2.1. The Importance of Privacy in Geospatial Datasets. Geospatial mobility datasets describe movement of vehicles, bikes, scooters, or pedestrians, and are often collected from users with their permission. However, when these datasets are shared with third parties, significant privacy issues may arise. Removing individual identifiers is not enough to achieve strong privacy because it is well known that linkage attacks, using multiple datasets, can allow attackers to identify the users even if directly identifiers, such as name, social security number, etc... are excluded from the dataset [**156**].

User patterns such as daily routines, extracurricular activities, and business activities are a part of behavioral privacy [**61**]. Mobility datasets capture these movements and have the potential to reveal information about behaviors of individuals and groups. Ideally, a privatized mobility dataset should maintain the overall movement patterns of the general population, while preventing the identification of small group or individual behaviors using inference attacks.

5.2.2. Related Work on Location Privacy. Protecting individual user location with DP has been studied extensively [**55**]. We can divide the location privacy models into two categories: privacy of online locations that shares the location data instantly and offline locations that uses for location dataset for extracting information. Privacy research on online location deals with the instant location information collected by user mobility applications such as navigation devices in real-time [**207**]. Offline location privacy research focuses on movement data (time ordered locations) and aggregated mobility datasets.

Privacy of Online Locations: There are several DP-based location models in the literature for preserving the privacy of online locations where location information is protected before it reaches to the data-center. For example, location-based social networks provide privacy for every location sample [**207**].

Generalizing and perturbing the actual location are the two popular approaches for location privacy. One of the early and well known location privacy techniques using DP models is called geo-indistinguishability [**7**] where the authors injected planar Laplace noise to the GPS points for hiding individual locations. A related study for real-time location sharing applications is proposed by [**54**] using circular noise functions. There are several privacy applications in the transportation domain inspired by the geo-indistinguishability [**189, 235**] for location privacy.

Another approach for location privacy is snapping individual points to a grid with lower resolution than the source data. The shared data is not the precise user location but is in a similar area. The authors presented a differentially private grid partitioning model for hiding user locations [**170**]. Differential privacy for grid partitioning of spatial crowdsourcing applications is studied with an adaptive multi-level grid decomposition technique in [**216**].

Anonymity-based privacy approaches seek to provide indistinguishability between users. Several k-anonymity privacy schemes in the case of real-time location sharing are presented in [**67**], and [**58**]. In [**154**], authors presented a location privacy approach for online locations using cloaking area [**108**] and differential privacy models.

While there are many privacy models for online location sharing platforms, real-time trajectory sharing requires more sophisticated and advanced privacy models due to spatial and temporal correlations of location traces. There are several privacy protection attempts for the online trajectories [**32**, **219**]. The authors, in [**218**], presented a DP-based planar isotropic location perturbation mechanism using a geometric sensitivity model.

Privacy of Offline Locations: Prefix-tree based aggregation is popular for achieving location privacy. One of the earlier methods creates a prefix tree and adds noise to the output node counts to achieve generalized trajectories [**35**]. Another prefix-tree and DP-based trajectory privacy method is proposed in [**231**] where authors used minimum description length method for clustering the trajectory segments on prefix-tree and injected a controlled noise to the count of clusters. Since generating differentially private trajectories requires a dense trajectory population to generalize or group trajectories, it is difficult to privatize individual trajectories. The methods presented in [**104**, **187**] aim to preserve the privacy of a trajectory using differential privacy.

There are different research directions regarding the type of aggregated geospatial data that is released: trajectories, networks, and statistics. Recently, a trajectory aggregation mechanism with DP was presented in [**69**], where the aggregation range was found privately using the Sparse Vector Technique (SVT). Inspired by this paper, we apply the SVT to our private range mechanism.An aggregate mobility data publication approach using a count-min-sketch method is presented in [**227**] for mobility distributions. This work evaluates the proposed aggregation mechanism against trajectory recovery attack model [**220**]. The framework studied in [**57**] applies filtering and adaptive sampling methods with differential privacy for sharing aggregate time-series statistics. The amount of aggregation is calculated with the proposed approach adaptively. Another aggregate geospatial statistic publishing approach with DP is presented in [**111**] where a sliding-window methodology captures event consequences on data stream. A DP-based data aggregation model is presented in [**148**] using a dataset from user call records. The traces are classified into several pre-defined groups, and noise is injected to the count of those groups. An online data aggregation method with location privacy is studied in [**206**] where authors presented a framework that dynamically groups and perturbs the statistics of aggregated locations using Laplace noise for achieving privacy protection. Markov decision modeling has recently been applied to aggregate mobility data privacy considering practical adversarial attacks to data privacy [**229**].

Recently, several private companies made location privacy tools available on their platforms. Gratel Labs [**121**] offers a synthetic location sampling model using generative models. Here Technologies [**197**] offers various location privacy techniques for protecting individual privacy, however, provenance of the data and the solutions are unknown.

Several re-identification attacks were able identify individuals from publicly available datasets, such as NYC [**50**] and London bike sharing [**2**]. The authors in [**199**] experimented with an attack strategy that recovers trajectories from aggregated location datasets. The key assumption in the paper is that the aggregated data is not anonymized.

Other location privacy methods include hidden Markov models and k-anonymity. One of the offline trajectory privatization methods proposed in [**157**] used a hidden Markov model for protecting trajectory datasets against multi-user correlation attacks. An example of the k-anonymity trajectory privatization method is presented in [**150**] where authors generalized movements to groups using a prefix-tree where leaves of the trees are removed if they have a value less than k. Transport network sharing for fleet vehicles is presented with k-anonymity and information theoretic approach using simulated ride-share datasets in [**95**] where the goal is to preserve the privacy of fleet trajectories by hiding the pickup and drop-off locations.

5.2.3. Aggregation of Mobility Data. Modeling mobility in urban regions often involves two main concepts: travel demand and infrastructure loading. Travel demand describes the mobility needs of a user population over a period of time. Infrastructure loading refers to the loading that the road network experiences as a function of the travel demand. The goal of transportation planning and operations is to ensure travel demand is served in the most efficient and safe manner.

Aggregation is a common approach for managing privacy in transportation datasets [**168**]. Individuals are clustered into an aggregated group of users that reduce the size and complexity of data. An example of publicly-available, aggregated mobility data is the Uber Movement website [**186**] where aggregated travel patterns for specified cities are released. The aggregation is done at both the temporal (one hour bins) and geospatial level (traffic analysis zone - TAZ).

5.2.4. Differentially Private Movement Datasets. Our mechanism preserves user-level privacy by hiding individual user participation in the aggregated dataset and prevents trajectory re-identification attacks by perturbing the origin and destination points. The amount of information loss associated with the applied privacy model is directly related to the granularity of the transportation problem of interest. In this paper, we focus on two transportation problems that require coarse or medium-level granularity and can be solved with our aggregated, differentially-private mobility network.

Congestion Analysis: An estimate of expected traffic congestion and the associated congestion mitigation plans are two key concerns for city planners. The transportation network is composed of links that define the road network. A zone refers to a certain area of the city and a collection of links. The capacity of links and their temporal changes are used to estimate traffic congestion using link-level aggregated statistics. Our privacy model generates differentially private aggregated link-level road metrics.

Major route identification: Differentially private traffic network generated from our model is a directed graph which shows the origins and destinations of traffic flows at the link level. It is possible to identify major traffic routes that may result in congestion in selected links. This output also allows predicting future traffic behaviors using some data-driven prediction models. One could train a machine learning agent with the privatized query response to predict travel behavior for the next day. Authors in [**173**] studied major route and busy traffic with an aggregated bicycle dataset. Our mechanism perturbs the origin/destination of trajectories before aggregation using the density and attributes of the localized links and preserves the main traffic routes.

5.3. Differential Privacy: Overview

Location privacy is the notion of privacy for aggregated mobility datasets. We require that the output of a query statistically guarantees privacy of individual user locations independent of the background knowledge. Differential privacy (DP) [**53**] guarantees that modifying the single input value has a negligible effect on the output statistical query. In this section, we summarize the general definitions and metrics of DP that are applicable to our problem.

We introduce the privacy concerning data $\mathbf{X} \in \mathcal{X}$ as vehicular mobility information in query $q \in \mathcal{Q}$. The data holder wants a mechanism that hides the sensitive information and reports the privacy preserved version of sensitive information using a randomized algorithm $\mathcal{A} : \mathbf{X} \times \mathcal{Q} \to \mathcal{D}$ where $\mathcal{Q}$ is the query space and $\mathcal{D}$ is the output space. DP promises that the algorithm $\mathcal{A}$ is differentially private such that participation or removal of a record results in minimal changes to the output of a query.

Let us first define the neighboring datasets:

Definition 1 (Neighboring Dataset). *Considering two databases* $\mathbf{X}$ *and* $\mathbf{X}'$*, if they differ by only one element* $\mathbf{x_i} \to \mathbf{x_i}'$ *corresponding to a link on the network, they are neighboring datasets.*

The above definition formalizes the adjacent or neighboring dataset that plays a crucial role in differential privacy.

Definition 2 (ϵ-Differential Privacy). *Given for every neighboring sets* $d \subset \mathcal{D}$*, a randomized algorithm* $\mathcal{A}$ *is* ϵ*-differentially private if*

$$Pr(\mathcal{A}(X) \in d) \leq e^{\epsilon} Pr(\mathcal{A}(X') \in d) \tag{5.1}$$

where ϵ *is a positive real number and probability comes from the randomness of the algorithm.* $\frac{Pr(\mathcal{A}(X) \in d)}{Pr(\mathcal{A}(X') \in d)}$ *is the privacy leakage risk for the randomized algorithm* $\mathcal{A}$.

ϵ-differential privacy is known as randomized response where adding or removing a single element from the database results in a similar probability. The smaller value of ϵ represents higher privacy guarantee and provides in-distinguishability.

In differential privacy, the appropriate epsilon is typically determined based on the sensitivity of the underlying data. The definition of sensitivity is given in [**52**] as follows:

Definition 3 (Sensitivity). *For any query function* f*:* $D \to R^n$*, the sensitivity of* f *is*

$$\Delta f = \max_{\mathbf{X},\mathbf{X}'} \left\| f(\mathbf{X}) - f(\mathbf{X}') \right\|_1 \tag{5.2}$$

for all datasets $\mathbf{X}$ *and* $\mathbf{X}'$*.*

Input perturbation and output perturbation are the two ways to implement differential privacy. When we want to release an aggregated mobility network, one way to protect privacy is through input perturbation. Laplace mechanism adds a random noise sampled from Laplace distribution:

Definition 4. *[Laplace Mechanism] For any function* $f\colon D \rightarrow R^n$*, the mechanism* $\mathcal{A}$ *gives* ϵ*-DP as follows:*

$$\mathcal{A}(D) = f(D) + Laplace(\epsilon, R) \tag{5.3}$$

Noise injection to the input can be done with different noise functions depending on the application requirements. Section 5.5.1 describes our additive noise method in detail. For a given sequence of queries and a threshold T, a mechanism called sparse vector technique (SVT) outputs a vector indicating whether each query answer is above or below T. The goal is to find the first index from query output that is above the threshold.

Definition 5. *[Sparse Vector Technique] Suppose* $f_1, f_2, .., f_k$*:*$\mathbf{X} \rightarrow R$ *be set of functions and* T *be a threshold for a database* $\mathbf{X}$*, the algorithm outputs binary outcome for each query answer* a_i *if it is above the noisy threshold or not:* $a_i \in \{\top, \bot\}^k$*.*

Definition 6. *[AboveThreshold] Given* $f_1, f_2, .., f_k$*:*$\mathbf{X} \rightarrow R$ *set functions with at most* L *sensitivity, AboveThreshold algorithm is* ϵ*-DP for every* $\epsilon > 0$ *[**141**].*

Definition 7. *[Composition] Let a set of randomized algorithms* $\mathcal{A}_1, ..., \mathcal{A}_k$ *that each* $\mathcal{A}_i$ *satisfies* ϵ_i*-DP.*

- *Sequential Composition: Let* $\mathcal{A}$ *be another randomized mechanism that executes* $\mathcal{A}_1, ..., \mathcal{A}_k$ *with independent randomness for each* $\mathcal{A}_i$*, then* $\mathcal{A}$ *satisfies* $(\sum_i \epsilon_i)$*-DP.*
- *Parallel Composition: Let dataset* $\mathbf{X}$ *is partitioned depterministically to different subsets* $\mathbf{X}_1, ..., \mathbf{X}_k$ *and executing each* $\mathcal{A}_i$ *with a different disjoint set* $\mathbf{X}_i$ *satisfies* $\max_i(\epsilon_i)$*-DP.*
- *Post-processing a randomized algorithm* $\mathcal{A}$ *that satisfies* ϵ*-DP does not break or consume any privacy budget.*

Given the composition properties and total ϵ privacy budget, DP-ANI builds different blocks according to composition properties to achieve a DP-satisfied randomized algorithm $\mathcal{A}$.

5.4. Our Metrics and Models

Protecting personally identifiable information is a crucial step before publishing the output of the queries. This section defines the existing structure and the privacy models we consider for our mobility dataset.

There are several things to be considered before applying our privacy model on mobility datasets. The privacy-protected aggregated mobility dataset includes both fleet and consumer trajectories. Fleet trajectories may reveal business-related information, such as customer pick up and drop off locations. Similarly, consumer trajectories can reveal daily behaviors of individuals. We solve the problem with two steps: (i) select a perturbation rate adaptively with respect to road network density for each OD of trajectories and (ii) perturb all OD GPS points and match them with new link. With our approach, the output aggregated mobility network is free from privacy issues of individual trajectories. We transform the point-wise GPS trajectories to an ordered series of road network links and enforce privacy on the aggregation of such trajectories.

Let $D(V, E)$ represent the road network as a weighted digraph, where the set of nodes V correspond to road intersection, set of edges E to roads, and weights that represent link metrics, such as length of the link or traffic volume. A link $\phi \in E$ connects intersections u and v where specific link attributes, such as number of lanes, speed limit, are stored in the link description. We have two set of trajectories: GPS trajectories and link trajectories. Let us define the GPS trajectories and then link trajectories:

1) GPS Trajectories: A sequence of GPS coordinates with l number of samples $\mathbf{x} \in \mathcal{G} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_l\}$ forms a GPS trajectory that reflects the continuous motion of the object. The set of all GPS trajectories are Ψ where $\mathcal{G} \in \Psi$.

2) Link Trajectories: Given the m number of vehicles on the road network, each vehicle travels between origins and destinations using an ordered link path generating a user travel path known as a *micro-graph* $\Phi \in D$. Every link trajectory has n number of links $\phi \in \Phi = \{\phi_1, \phi_2, ..., \phi_n\}$ and $\Phi \subset E$. The set of trajectories is the corpus of all link trajectories with m users $\Phi \in \Lambda = \{\Phi_1, \Phi_1, ..., \Phi_m\}$.

A link $\phi_{i,j}$ refers to the ith link of user j. Every link $\phi_{i,j} \in \Phi$ has a set of values: length of the link, travel time, speed, and link counts. The traffic density of the network is represented by aggregating trajectories and is referred to the *aggregated mobility network*. The raw mobility network is ϑ and the privacy preserved mobility network is Σ. The goal of our research is to release the privacy preserved aggregated mobility network from the raw mobility network $\vartheta \rightarrow \Sigma$.

Link count β refers to the number of times a link ϕ occurs on the mobility network ϑ. For example, if a link $\phi_{i,j}$ of selected trajectory Φ_j has occurred in the graph ϑ only once, then 1 is assigned for the link count value for corresponding link. The represented link model is built upon the road characteristics. Each link ϕ is classified into one of five classes in terms of the capacity and functional role of the road, called a functional class. Arterial roads have lower functional classes, rural streets have higher functional classes. Next, we introduce link matching.

3) Link Matching: GPS coordinates are an estimate of a device's location from satellite broadcast information and are generally enhanced with localized terrestrial information. These locations can be perturbed by localized environments, such as tree cover or urban canyons. As a consequence, GPS locations may not match to a link on the road network. Link matching generates ordered set of road network links describing the user's trajectory considering the road network $D(V, E)$ and GPS points [**172**]. In this paper, we used a link-matching algorithm designed by University of California, Berkeley Smart Cities and Sustainable Mobility Research Group [**143**].

5.5. Differentially Private Adaptive Noise Injection

Differential privacy is a probabilistic approach that provides privacy primarily by using injected noise. This noise injection aims to hide individual contributions to the overall statistic while preserving the statistical properties of data in the aggregate level. To make origins and destinations of trajectories differentially private at aggregated mobility network, our DP algorithm uses the first approach by injecting planar Laplace noise to the GPS points before generating an aggregated mobility network. The noise level changes the underlying characteristics of the output. The higher noise level leads to stronger privacy but less accurate results. For large and dense datasets, this inaccuracy will be less significant in the overall statistics because the required level of noise to achieve differential privacy is lower [**45**]. Now we explain our noise injection approach.

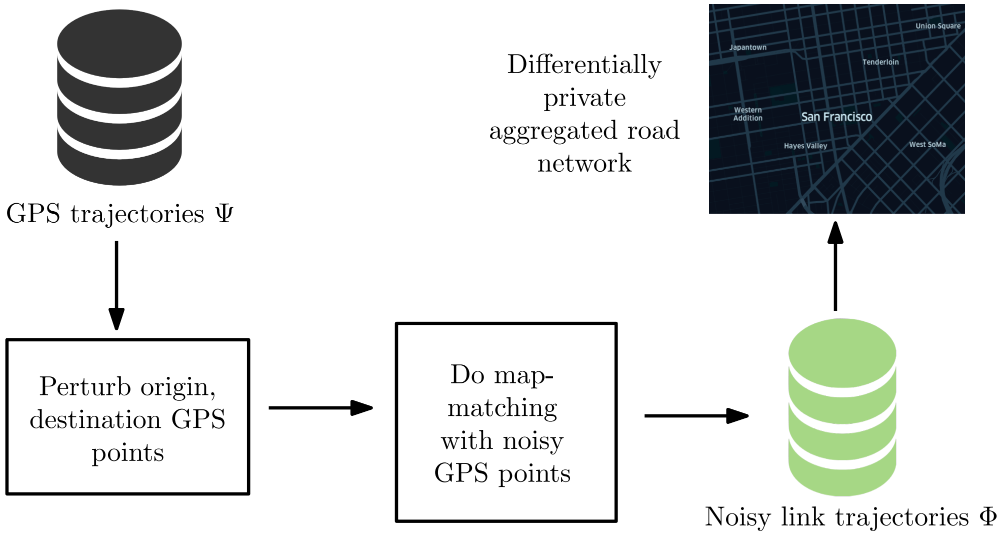

GPS trajectories Ψ
Perturb origin, destination GPS points
Do map-matching with noisy GPS points
Noisy link trajectories Φ
Differentially private aggregated road network
Japantown
Union Square
Tenderloin
Western Addition
San Francisco
Hayes Valley
West SoMa

FIGURE 5.1. Differentially private aggregated mobility network is released using our noise injection model and link matching algorithm.

Figure 5.1 shows the flowchart of our differentially-private, aggregated mobility network concept. A user starts from a geographical location called the *origin* and stops at another geographical location called the *destination*. These origin and destination locations are considered sensitive information that must be protected. A *trajectory* consists of the sampled locations between origin and destination. The link-matching algorithm infers the ordered set of links using GPS location data and the previously matched link path from the $D(V, E)$ network for each user's GPS points. The network $D(V, E)$ often constrains the link selection process. Since we release the aggregated mobility network, the privacy issue is reduced to the link set that has a low density of links and are connected to origins and destination nodes.

To provide privacy to the aggregate mobility datasets, we use a differentially-private, adaptive noise injection (DP-ANI) model, using planar Laplace noise. The origin and destination GPS points are obfuscated based on the network density, and noisy GPS points are matched with a new link using the link-matching algorithm. The two key parameters used for noise sampling are ϵ and R. While ϵ is responsible for the noise level, R is the distance parameter for moving the center of the noise in the geospatial domain. Output of the noise function is a new randomized GPS location in the same space. The noise function is a bounded probability distribution on polar coordinate systems.

5.5.1. Noise Sampling. One state of the art location-hiding approach to inject noise to the GPS dataset is called geo-indistinguishability [**7**]. The mechanism adds planar Laplace noise to the GPS point x_0 within an area of r. The output of geo-indistinguishability is a perturbed GPS location whose privacy level is ϵr. The radius r is the desired distance to provide privacy protection.

Geo-indistinguishability provides a guarantee that the probability of the exact GPS point presence decreases exponentially with the radius r for given location $\mathbf{x_0}$. This noise function is a linear distribution for problems in 1-D space. It is a 2-D surface for a geospatial problem. The probability density function (PDF) of such noise function for any point $\mathbf{x} \in \mathcal{R}^2$ is:

$$D_\epsilon(\mathbf{x_0})(\mathbf{x}) = \frac{\epsilon^2}{2\pi} e^{-\epsilon d(\mathbf{x_0},\mathbf{x})} \tag{5.4}$$

where $\frac{\epsilon^2}{2\pi}$ is a normalization factor. This PDF function is a planar Laplacian distribution that is sampled in polar coordinates instead of Cartesian coordinates $D_\epsilon(r,\theta) = \frac{\epsilon^2}{2\pi} r e^{-\epsilon r}$. A point in polar coordinates (r,θ), where r is the distance of $\mathbf{x}$ from $\mathbf{x_0}$ and θ is the angle, is randomly drawn. Since r generally adds a small perturbation to the GPS point, in this work, the r value is scaled with given R radius value from the input in order to provide adaptive noise structure. By scaling up the obfuscation parameter r we have a larger noise level that moves the GPS point further away in euclidean distance. The noise sampling approach involves the following steps:

- Draw θ uniformly in $[0, 2\pi)$,
- Draw p uniformly in $[0, 1)$,
- Find $\Gamma = C_\epsilon^{-1}(p)$,
- Set $r = \Gamma * R$ for larger noise with given input radius R
- Finally, calculate $\mathbf{z} = \mathbf{x} + [r\cos(\theta), r\sin(\theta)]$,

where $C_\epsilon^{-1}(p) = -\frac{1}{\epsilon}(W_{-1}(\frac{p-1}{\epsilon}) + 1)$ is the inverse cumulative distribution function of r, and W_{-1} is Lambert W function (the -1 branch).

The original implementation of geo-indistinguishability achieves privacy through fixed bounding range R. However, our approach provides geo-indistinguishability by adding controlled noise $L(\epsilon, R)$ to the origin and destination GPS points x_i within a certain range R in order to mask the actual locations using density-based private noise range selection method R. Selecting the same threshold

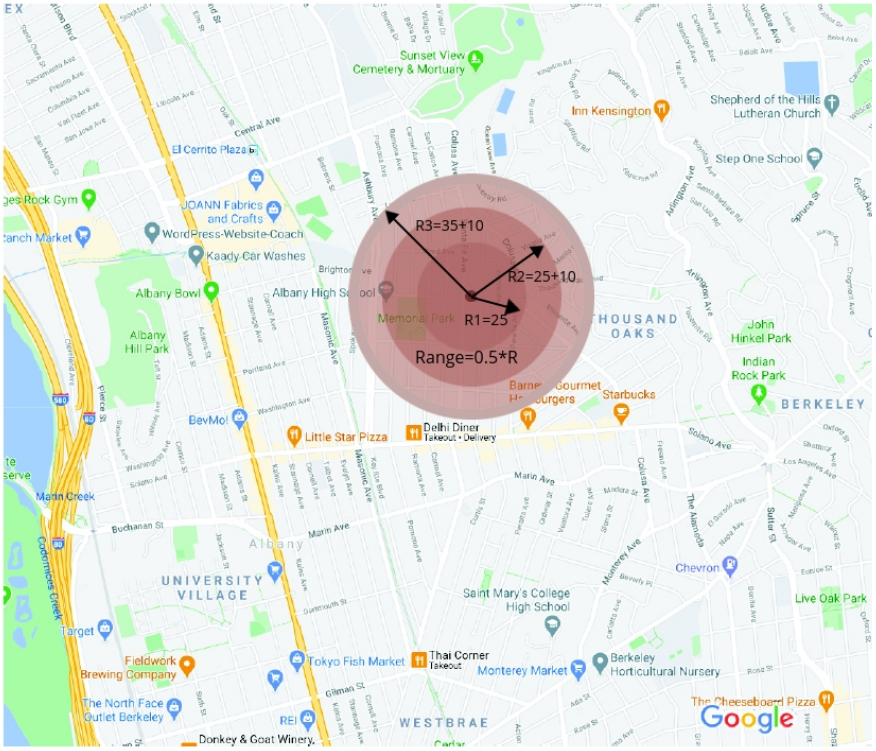


FIGURE 5.2. Buffer range for determining link density

from all noise function would allow adversaries to access private information through reverse engineering. Our work also privatizes the range value R with a different DP mechanism.

5.5.2. Private Bounding Range Selection. This section explains determining the bounding range and selecting the noisy link for the link-matching approach using planar Laplace noise introduced in the previous section. Randomly injecting noise without considering the network's density would not achieve the desired privacy level consistently. Some GPS points with few links nearby would match the noisy GPS point with the same link after the noise injection. To overcome this problem, the noise level is selected adaptively with respect to the link density of network $D(V, E)$.

The adaptive bounding range selection may also reveal some information about the user's location. Therefore, we employed a private bounding range selection algorithm using the sparse vector technique [**141**] to find the bounding range R privately. We were inspired to use the private parameter selection method for geospatial domains by [**69**].

Algorithm 3 Private Bounding Range Selection

```
***Input*** privacy budget $\epsilon_{radius}$, threshold $\tau$, initial bounding circle $z_{init}$, maximum bounding circle $R_{max}$, network $D(V, E)$, and number of iterations $k = R_{max}/Z$, link functional class $fc$.
$z \leftarrow z_{init}$
**for** $\ell = 1, ..., k$ **do**
    $N_\ell \leftarrow$ Number of links within $z$ at network $D(V, E)$
    $z$+=10 meters
**end for**
$\ell^* \leftarrow AboveThreshold(N_1, ..., N_k, \tau, \epsilon_{radius})$
$R \leftarrow z_{init} * \ell^*$
$BoindingSetFC \leftarrow$ links within $R$ with functional class $fc$
**return** $R$ and $BoindingSetFC$
```

Our approach generates noisy link-based trajectories while keeping the noisy trajectories close enough to the original trajectories to have similar traffic characteristics at aggregated mobility network. As we mentioned above, every link has functional class information, and the density-based noise function should move the GPS point to a place that matches with the same functional class.

Definition 8 (Density Function). *Given the ϵ value, bounding range R of the noise function $L(\epsilon, R)$ is selected privately using SVT differential privacy mechanism using the function $f(\theta)$ where θ is the network density in terms of the number of links.*

Our noise model perturbs every origin and destination of trajectories and applies link-matching to the noisy trajectories. Then, the aggregated mobility network is obtained with a DP guarantee.

The straightforward method would select a bounding range R considering the worst-case scenario given the whole trajectory dataset. However, this would lead to a poor utility at aggregated mobility network because the larger bounding range R takes the center of the noise function to a far distance, resulting in higher perturbation on GPS location. Instead, we developed an algorithm to select the bounding range R privately in Algorithm 3.

The method first adjusts the noise level using the link network density around the GPS point (see Fig. 5.2). To do so, starting from an initial radius $Z = Z_{init}$, the method increases the radius k iterations until reaching the maximum bounding range R_{max}. The R_{max} would be the area that covers the whole geospatial region of the dataset. For each iteration, N_ℓ contains the number of links within the bounding range Z at network $D(V, E)$. In the second phase, our algorithm selects an index ℓ^* given the threshold τ using *AboveThreshold* algorithm, an SVT algorithm described in [**52**].

Algorithm 4 Noisy link matching algorithm

```
Input $\Lambda, \Psi$
Input $\epsilon_{laplace}$ for planar Laplace noise function
for $\mathcal{G} \in \Psi$ do
    for $x \in \mathcal{G}$ do
        if For OD GPS points then
            Select $R$ and $BufferSetFC$ using Algorithm 3
            Inject adaptive noise to the GPS point $x$ using $L(\epsilon, R)$
            Select closest link from $BufferSetFC$
        else
            Select closest nodes from $V$
        end if
    end for
    Find candidate paths between closest nodes list
    Connect the candidate paths
    Build the noisy link-matched trajectory with connected links
end for
return The noisy link trajectory network $\Sigma$
```

The private bounding range R is acquired with respect to ℓ^* and corresponding road links within this bounding range from the same functional class links are stored in $BoindingSetFC$. Please refer to Algorithm 3 for the full pseudo-code.

Once the bounding range R is selected, our algorithm applies noise injection for the corresponding GPS point using $L(\epsilon, R)$. After the noisy GPS point is returned, the closer link to the GPS point in the set of the same functional class links is selected. Once all the links are found from the GPS points, the candidate paths are selected, and the noisy link trajectory is built using the candidate paths (see Algorithm 4). Finally, the noisy link-based trajectories generated with our approach form a mobility network that ensures the privacy of link-level OD locations.

5.5.3. Differential Privacy Analysis. This paper provides origin-destination privacy to aggregated mobility networks. For each trajectory, there are two sensitive locations, origin and destination. While two DP mechanisms are used in this paper, bounding range selection and noise injection, the sensitivity Δ is the same for output data. Adding and removing a user can only change the aggregated network visitation rate at most two links for each origin and destination. Therefore, the sensitivity of our DP mechanisms is 2.

Lemma 5.5.1. *Our bounding range selection algorithm satisfies ϵ_{radius}-DP.*

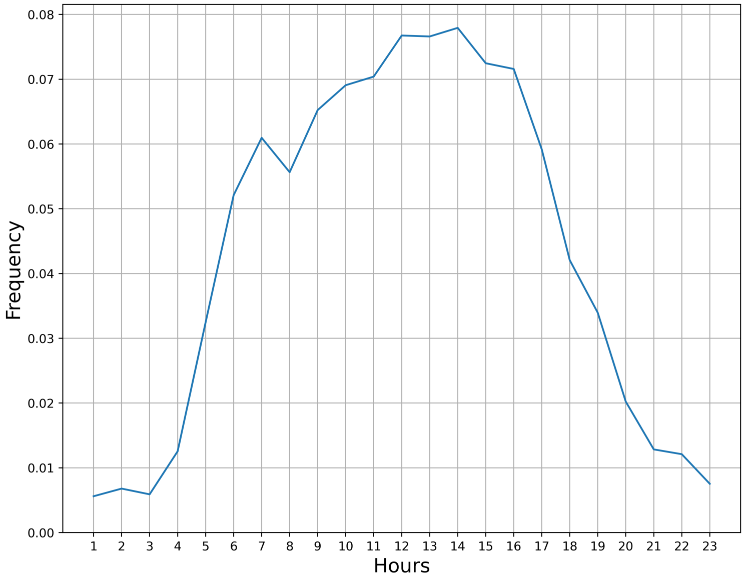


FIGURE 5.3. OD pair count for a typical week-day

PROOF. Given the sensitivity of two queries N_ℓ, where adding or removing a user location can change the link visitation at most two links by one, and the definition of SVT [**141**], the *AboveThreshold* algorithm is ϵ_{radius}-DP. Accordingly, the private bounding range selection algorithm is also ϵ_{radius}-DP. □

LEMMA 5.5.2. *Given the sequential and parallel composition properties of DP, the noisy link matching algorithm satisfies 2*($\epsilon_{laplace}$+ϵ_{radius}) DP for the output mobility network.*

PROOF. The aggregated mobility network is produced from the n user trajectories. Each trajectory has one origin and one destination GPS point. For each of these GPS points, noisy link matching algorithm selects a private bounding range R and adds planar Laplace noise using the private R. Given the sequential composition property, total privacy budget for each trajectory is $\epsilon_{user} = 2 * (\epsilon_{laplace} + \epsilon_{radius})$. Since each trajectory accessed only once and uses the same privacy budget ϵ_{user}, the overall privacy budget of output aggregated mobility network is ϵ_{user} given the parallel composition. In summary, our private aggregate mobility network satisfies ϵ_{user}-DP. □

5.6. Experimental Evaluation

5.6.1. Dataset Description. This project uses a real-world dataset collected in the San Francisco Bay area in California from January to February 2019. It contains two months of fleet and consumer GPS trajectories with varied sampling rates. To make the computation tractable, we extract a smaller region in the Berkeley area and time period for evaluating our privacy model. Most of the trajectories have sampling rates of less than 1-minute, which improved the ability of the link-matching algorithm to generate realistic link trajectories.

The dataset is created from a variety of location-sharing applications and GPS tracking devices. When the devices are active, location (lat, lon), speed, and heading are collected along with a unique device identifier. While we do not necessarily know that the device was active at the user's actual origin and destination points, we assume that the start and end points of GPS traces represent origins and destinations.

5.6.2. Temporal Correlation. Figure 5.3 shows the hourly distribution of the OD pairs in the San Francisco region. A temporal pattern is associated with the time of the day, with distinct rush hours where the number of OD pairs peaks. Hence, any aggregation methods should preserve this pattern if it is to be useful for traffic management applications. For example, aggregating mobility data from morning hours with afternoon non-busy hours would not reflect the real traffic patterns for the morning or afternoon. Therefore, we apply our privacy-preserved aggregation model for each 1 hour period. We show as an example the 1 pm to 2 pm time period.

5.6.3. Comparisons to Alternate Approaches. We compare the DP-ANI model to several techniques. A straightforward privacy method, similar to the k-anonymity approach in [**150**], removes successive links, either from the origin or to the destination, with less than k link counts from the aggregated network. We have chosen $k = 2$ for our experiments. We refer to this model as *OD successive remove.*

The privacy definition of DP-ANI is based on distance, the shortest distance between noisy and the original location. Therefore, we also included an ablation study to have a fair understanding for our DP-ANI model:

- $DP - ANI_{Fix}$**:** This version performs the same privacy mechanism for origin-destination without a private range selection. Here we assumed that the data curator sets a fixed threshold of a number of links for adaptive range selection, which is 25 road links for our network $D(V, E)$.
- $DP - ANI_{Max}$**:** Standard DP methods, mainly for tabular datasets, applies noise injection based on the sensitivity with respect to the worst case scenario. However, geospatial datasets are different, and their sensitivity definition differs from others. The other baseline model applies the worst-case maximum range to all trajectory ODs. Instead of setting the number of links and looking for a range value, we found the maximum range value with more than 25 road links in whole trajectory ODs.

The DP-ANI provides stronger privacy by trading off some utility of the dataset (i.e., answering a subset of finer-granularity queries). We show the performance of the proposed mechanism with respect to the original aggregated model without any privatization method in our experiments as *Original data*.

5.6.4. Utility Metrics. We have chosen practical utility metrics commonly used in transportation studies to quantify the efficiency of our privacy mechanism. In this section, we explain the importance of utility metrics. The goal is to have a higher similarity in the utility metrics between the original and privacy-preserved mobility networks, given the same level of privacy protection. In order to present results clearly, all the numerical results are normalized with respect to the original mobility network.

Spacial density analysis plays a crucial role in understanding human mobility [**86**]. Our first utility metric is the change in aggregated mobility network length. This is our primary utility metric since we aim to acquire a similar mobility network that retains the mobility characteristics. Minimizing the change in the length of aggregated mobility network makes output privatized data more useful.

Since our mechanism provides privacy for user ODs, the second metric we looked at is trajectory level utility: the rate of OD link that moves to another road link after noise injection. The goal is to displace as many OD links as possible.

The main characteristic of aggregated mobility networks is the link visitation rates or link counts. There are several distance metrics to compare the probability distributions. The Wasserstein distance metric is a good way of comparing the count query distributions quantifying the similarities of probability distributions given a metric space. Our count query metric is the number of link visitations named in figures as *Link Counts*.

The road network classifies the link in different classes, from local streets to highways, into five classes. It is essential to retain the road class distribution in the output network similar to the original aggregated network. We employed the Wasserstein distance on the road link functional class distribution for another aggregated utility metric.

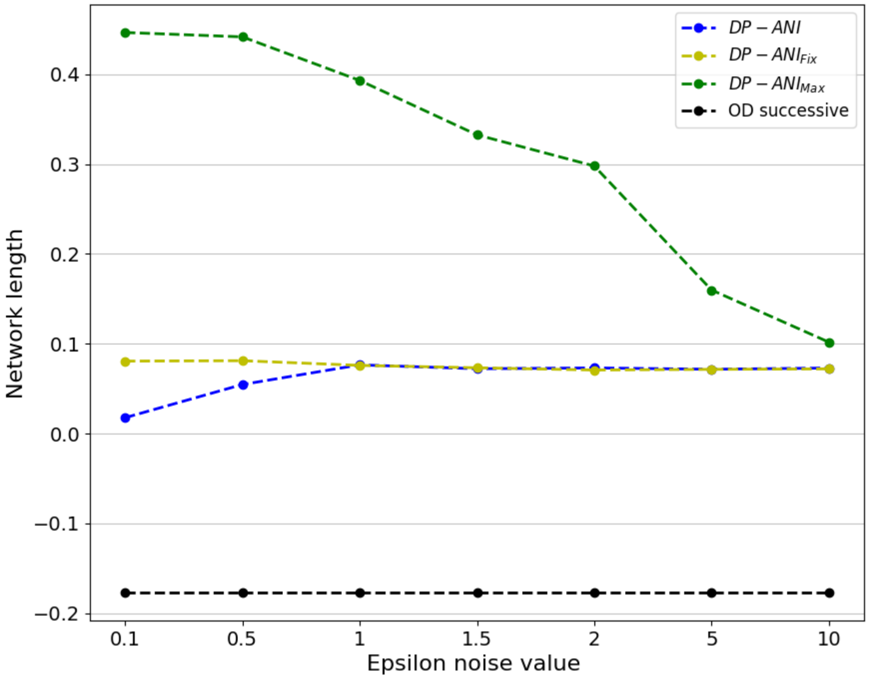


FIGURE 5.4. Network length for all trajectories with baseline comparisons.

5.6.5. Numerical Results. The DP-ANI method can extend or shrink the length of the trajectories. As such, fluctuations in the experimental results concerning the different levels of ϵ values are expected. For lower (or higher) ϵ values, we have higher (or lower) noise levels. A higher noise level will perturb more links. When we apply the DP-ANI method to the mobility dataset, the number of privatized single-link-count links vary depending on the privacy level ϵ. We use a

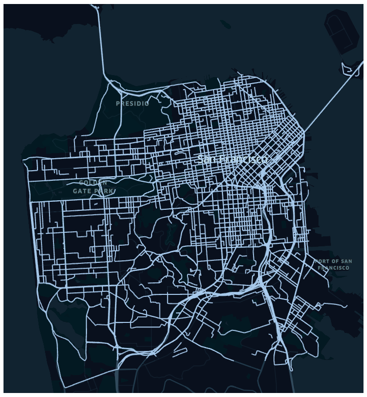


FIGURE 5.5. Only the original network

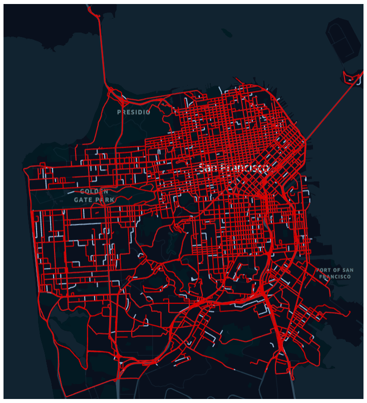


FIGURE 5.6. Original vs the DP-ANI model network

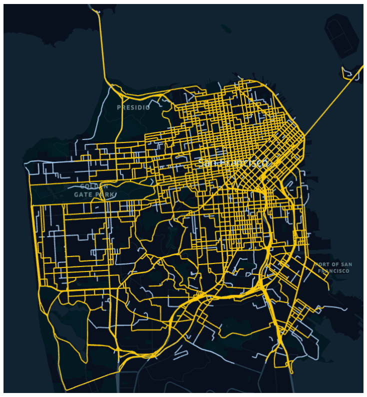


FIGURE 5.7. Original vs OD-successive link removal network

FIGURE 5.8. Aggregated trajectory networks for a 1 hour period are represented together on the map. The red link network represents the DP-ANI-AT model for $\epsilon = 1$, the yellow represents the OD successive link removing model, and the light blue represents the original model. Figure-(a) shows the original network only without any overlap. In Figure-(b) and Figure-(c), when the DP and OD-successive links overlap with the original network, the links appear as red and yellow, respectively. The network lengths are 195.2, 138.2, and 194.6 miles for networks with the DP-ANI-AT model, OD successive link removal model, original network respectively.

range of ϵ values between 0.1 and 10 to evaluate the performance of the DP-ANI. The chosen ϵ values are selected to reflect the lower and upper limits of the impact of the DP-ANI perturbation mechanism. We present our results with the same aggregation concept described in 5.6.2 and the same set of ϵ values for all the experiments.

The DP-ANI may affect the active network length as the noise may adjust the trajectory to a position that requires more maneuvers or positions a start location behind the original origin location or similarly ahead of the original destination. Figure 5.4 shows the network lengths of different privacy models with the set of ϵ values in terms of the divergence from the original aggregated network. The blue line shows the performance of our adaptive threshold DP-ANI model. 1.0 in the y-axis is the reference point for the original network. The DP-ANI model has the slightest change from the original network compared to other variants with less than 10%. While $DP - ANI_{Max}$ is the second-best-model, $DP - ANI_{Max}$ and *OD successive* yielded very high divergence from the original network. Note that since *OD successive* do not depend on the ϵ parameter, their resulting plots are flat across the horizontal axis.

Next, we visualize the aggregated networks to see the differences between the DP-ANI-AT model with $\epsilon = 0.1$ and other comparative models in Figure 5.8. *OD-successive*, a k-anonymity based approach, is another formal privacy protection model. For this reason, we compare our model with *OD-successive* and original network models. To preserve the utility of the privacy-preserving data, we would like the aggregated trajectory network to overlap with the original network except for origin/destination points/segments. In Figure 5.5, we only present the original aggregated network to reflect the actual coverage area. While the DP-ANI network shows a high similarity with the original network (Figure 5.6), there is a distinct difference between *OD-successive* and original networks shown in Figure 5.7, primarily in the outer areas of the San Francisco region.

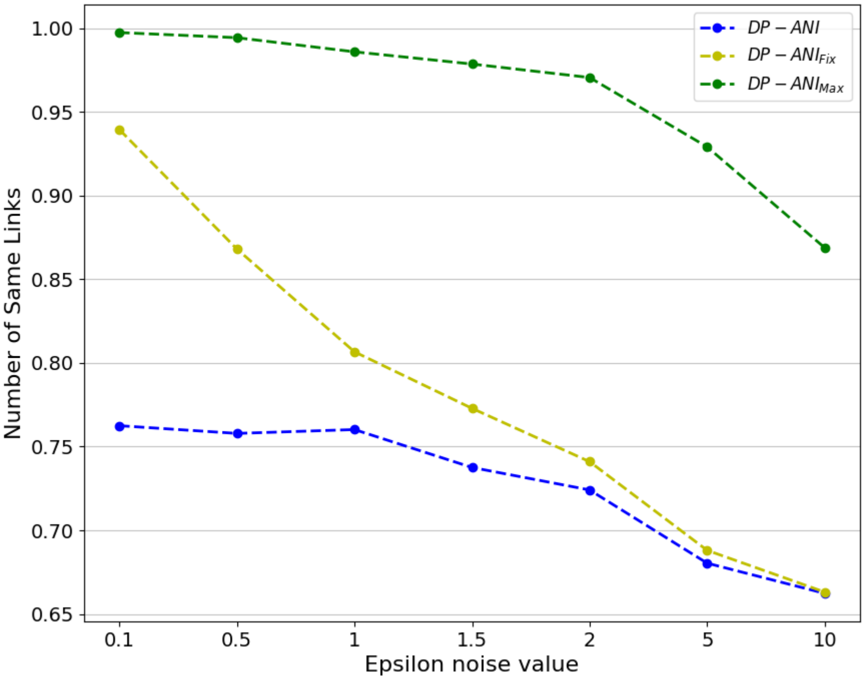


FIGURE 5.9. Comparison of different ϵ values and the change of percentage of ODs.

Next, we identify the ratio of unchanged OD links after the noise injection and aggregation steps. We only look at OD link displacements since we consider the link as not having any privacy concerns if it is not an origin or destination for any trajectory. Depending on the noise level and the density of the road network $D(V, E)$, the links can be matched with the same link after the noise injection. The value of ϵ defines the level of privacy. To quantify the privacy of our method, we inspect the privatized link ratio over the OD links with respect to different ϵ values. Fig. 5.9 shows the ratio of unchanged OD links for all ODs. The goal of the DP-ANI model is to move OD links. Therefore the output is expected to have higher ratios for the lower level of ϵ. The highest level of the privatized link ratio is observed with the lowest $\epsilon = 0.1$ with an average of 76%. $DP - ANI_{Max}$ and $DP - ANI_{Fix}$ achieve higher ratios compared to DP-ANI. However, they both have drawbacks: $DP - ANI_{Max}$ results in lower utility and $DP - ANI_{Fix}$ provides a lower privacy guarantee with a fixed range value that could leak more information due to less randomization. The lowest ϵ produces the fewest number of single-link-count OD-links.

Finally, we evaluate the performance of DP-ANI model with different variants and *OD-successive* model on the Wasserstein distributional similarity metric in Figure 5.10 and Figure 5.11. The lower value indicated higher similarity. For all the ϵ values, our method is more similar to the original aggregated mobility network at link count distribution and the functional class distribution. Increasing the ϵ value yields higher similarity in distributions for other variants DP-ANI-Max and DP-ANI-Fixed with a lower privacy guarantee.

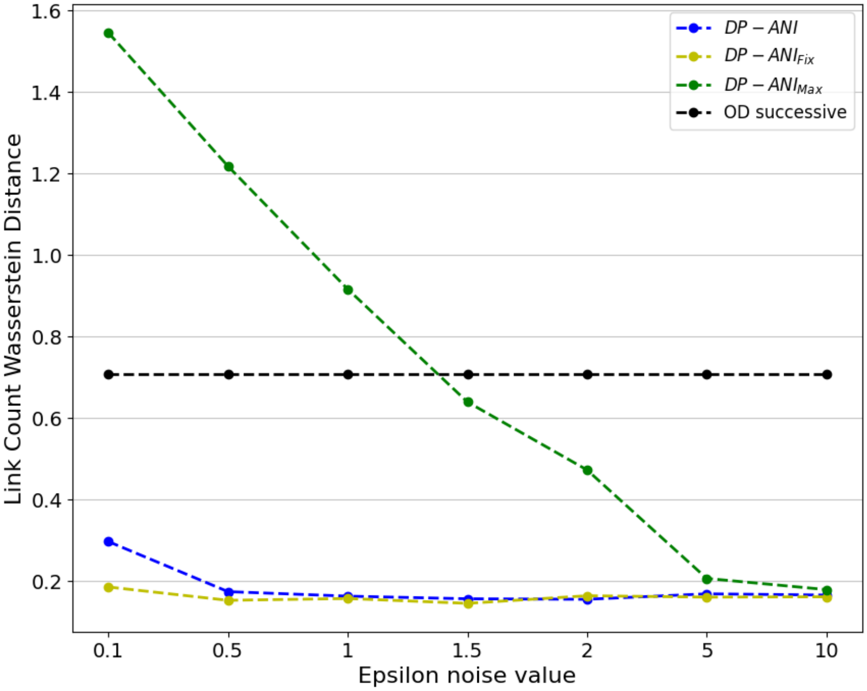


FIGURE 5.10. Wasserstein distance of link counts

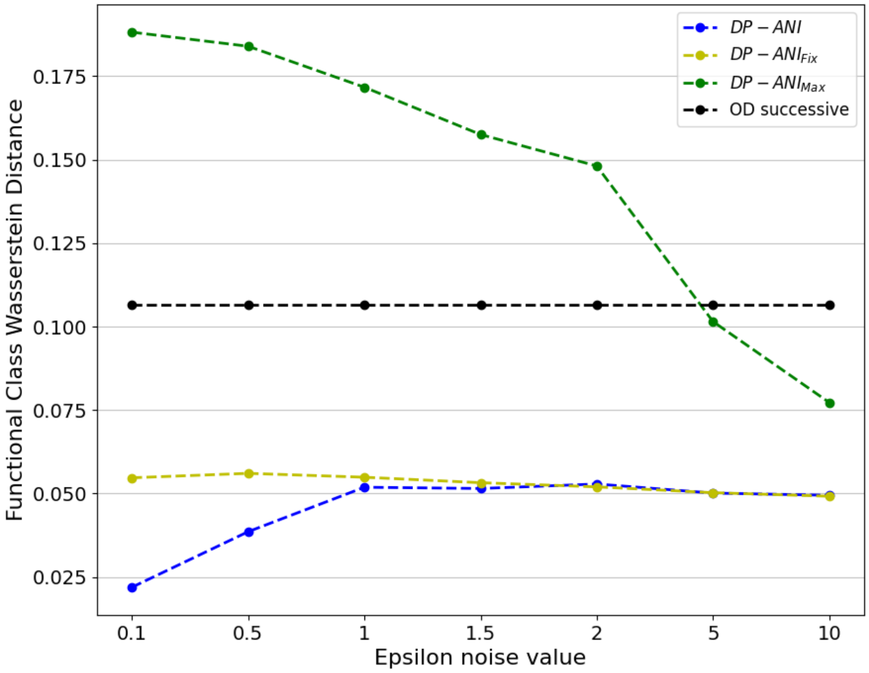


FIGURE 5.11. Wasserstein distance of link functional classes

5.7. Discussion

Privacy of aggregated mobility trajectories serves as a useful first step towards unlocking siloed trajectory data. Because geospatially constrained and temporally correlated mobility datasets have unique characteristics, it is difficult to preserve the privacy of individual trajectories. This paper describes a method generating an aggregated mobility representation of the original trajectory set using DP-based adaptive noise injection model. In this section, we summarize several limitations and discuss open research questions.

5.7.1. Limitations.

The first limitation of our study is aggregation scale. The generated aggregate mobility network aims to reflect the general patterns of human mobility, however, due to sparsity of most of the datasets that we used, the realistic travel patterns are not evident in the aggregated dataset as shown in the earlier Figure 5.3, where the approximately 1200 trajectories do not show the expected rush hour travel pattern. The sparsity of the dataset required high perturbation in order to privatize the dataset. In order to overcome this problem, aggregating three or more days of trajectories in one hour period can preserve the utility of the dataset, but this characteristic is highly dependent on the dataset.

Inherently perturbing origins and destinations can affect the meaning of the object. Although DP-ANI maintains the same functional class of the OD links, it can cause a catastrophic change in the individual trajectory that will affect the aggregated output. For example, when we perturb a link to the other side of the highway, or to a one way road, that moved link must connect with the rest of the trajectory. This may result in much longer routes and if the composition of the dataset involves many of these type of perturbations, this will significantly change the total travel distances of trajectories.

Finally, biases associated with fleet trajectories is a third consideration. Fleet trajectories should be privatized as patterns related to the fleet's business-model, however, this approach likely does not solve the fleet privacy problem, as a fleet's patterns may still be evident in the aggregated mobility model. Ideally, it is better to work with an evenly distributed fleet and consumer trajectories in the aggregated output to reflect the true travel patterns. Depending on the original dataset, removing fleet trajectories and working only with consumer trajectories may result in a very sparse dataset. This study used both fleet and consumer trajectories.

5.7.2. Open Research Directions. The DP-ANI model achieves an ideal level of privacy by preserving the privacy of more than 75% of the privacy concerned ODs on three aggregation models at around $\epsilon = 0.1$ for the experimented San Francisco area. Regarding to the privacy level, it could be interesting to test DP-ANI on a larger geographical region in order to define a proper ϵ range.

Future research will focus on generalizing the DP-ANI model to a broader range of aggregation concepts with different levels of granularities, such as zone level or OD levels, in order to assess confidentiality and applicability of the current approach. In addition, the computational cost of data processing on large datasets will likely require novel computational methods. A recent trend is to minimize centralized data collection and privatize the data at local or regional controller using real-time localized DP models [**208**]. Forming trajectories with perturbed or privatized local GPS locations may break the coherence of the trajectory and the usefulness of the trajectory. As such, investigation into how this localized privacy model affects the aggregation and privatization methods would be important.

5.8. Conclusion

This paper presents a differentially-private, adaptive noise injection model for aggregated trajectory networks that protect individual origin-destination locations. This method injects planar Laplace noise to the individual origin-destination GPS points by considering the density of the localized road networks. The actual perturbation distance for each GPS point is adjusted by considering the localized link density and this selection is performed privately with the Adaptive Thresholding method. After injecting noise into the GPS points, the location is matched to new network links, and a new origin-destination privatized trajectory is generated and integrated into the aggregated mobility road link network. We evaluated our differentially private mechanism for a variety of variants and a k-anonymity-based privacy model.

This project uses an aggregation concept that generates a privatized, aggregated mobility network for a specific regional dataset in San Francisco, CA. Future work will include extending this investigation to different types of mobility data aggregation models while also addressing the computational efficiency for large-scale datasets.

CHAPTER 6

Differential Private Map matching

6.1. Introduction

The pervasive use of location tracking devices and navigation tools generate a huge amount of spatio-temporal data associated with user mobility patterns. These collected user mobilities or trajectory data can be used for variety of purposes, such as advertising, transportation analysis, and personalized recommendation. However, mining such user movement information can reveal sensitive information, hence posing a legitimate privacy threat. Recent studies show that anonymized user trajectories are vulnerable to re-identification attacks even with just a few spatio-temporal points [**44**].

There have been several proposed privacy mechanisms for trajectory datasets based on two main concepts: indistinguishability and uninformativeness. The former approach via k-anonymity ensures that every trajectory is similar to one another. On the other hand, uninformativeness is achieved via differential privacy, where adversaries cannot retrieve extra information after accessing the dataset [**62**]. While indistinguishability privacy is achieved through suppression or generalization methods [**78, 211**], uninformativeness privacy is, in general, achieved by perturbation and noise injection [**7, 80, 96, 218**]. However, the existing privacy methods result in high utility loss when trajectory queries are performed on the protected mobility data due to several reasons, such as unreasonable location sequences or geospatial mismatches.

Most techniques in the literature protect the privacy of individual user trajectories with respect to other trajectory samples in database [**20, 96**].

However, this approach cannot guarantee user privacy in low-density datasets. This paper attempts to protect the privacy of every individual trajectory regardless of the rest of the data by masking origin and destinations (OD) with noise injection and protecting travel paths with randomized path selection. Another limitation of existing privacy-preserving methods is the higher mismatches of geospatial location sequences. Discretization of locations through grids or zones does not consider practical implications of the "private location".

We propose to incorporate the road segment densities, which intrinsically imply population densities, instead of grid or zone structures in designing our differential privacy mechanism.

Differential privacy (DP) provides statistical privacy protection by applying randomization techniques to the database and masking the personalized identifiers [**52**]. DP assures that an adversary with background knowledge about the dataset cannot extract private information from the dataset. The goal of this work is to design a DP-based privacy mechanism with deterministic constraints in order to have a lower bound for both location privacy and travel path privacy. The proposed scheme outputs a set of privacy-preserved trajectories at the road segment level.

Injecting a fixed level of noise to all geo-spatial positioning (GPS) samples cannot guarantee the privacy of locations. We have achieved promising results applying adaptive noise injection to origin destinations conditioned on the travel intensities of the associated road segments to protect the privacy of aggregated mobility networks [**93**].

In this study, we propose a two-stage differential privacy method for map-matching, called DPMM, to protect the privacy of individual trajectories. First we apply adaptive noise injection to OD locations. Second we match the GPS points to the road segments privately and select randomized paths between selected road segments to generate private user trajectories. Our contributions are listed below:

- We expanded our prior work [**93**] to protect user OD location privacy for individual trajectories by injecting Planar Laplace noise to the user OD GPS points.
- We employ the exponential DP mechanism to randomize travel path construction to protect individual user trajectories.
- Both the injected noise level and path selection are adapted based on link density of the location and the functional category of the localized links.
- Our experimental evaluations show that our DPMM scheme can protect user location and trajectory privacy while maintaining high utility by providing accurate query responses compared to raw data.

6.2. Background and Related Work

The privacy risk associated with trajectory datasets is at every level, including single location sample, whole trajectory level, and set of trajectories (community) level. There are two privacy concerns associated with user mobility data this paper addresses: location privacy and trajectory privacy. Location privacy refers to protecting individual user's true locations at any point in time. On the other hand, trajectory privacy protects the knowledge of specific path or route (a sequence of spatial-temporal samples) taken by a user [**29**, **43**, **99**]. Our goal is to apply DP to achieve both location and trajectory privacy without compromising the utility of the dataset (in terms of providing accurate response to a subset of fine-granularity queries).

Location Privacy: There are several DP-based location privacy studies in literature. One way of achieving location privacy is perturbation by injecting controlled noise to the location coordinates [**7**, **54**]. Laplace noise [**7**] and circular noise methods [**54**] are the two well-known perturbation-based location privacy models in DP community. Another approach for location privacy is forming location grids with lower resolution depending on the density, then sampling fake locations from the private grids [**170**, **216**]. Studies show that sampling fake locations cannot guarantee hiding the true locations due to statistical data correlations [**107**]. Hence, for OD location privacy, we employ adaptive noise injection methodology from [**93**] by considering the road segment density with the Laplace noise mechanism presented in [**7**]. Neither of these prior studies protect trajectory privacy.

We have previously introduced a location privacy mechanism for aggregated mobility datasets [**93**]. We propose selecting the magnitude of noise for ODs based on the road segment densities and the functional category of roads to form an aggregated mobility network. The noise injection is only applied to a subset of trajectory ODs if the road segment they belong to has less than a set density threshold. This work applies the idea of the adaptive noise injection approach to all trajectory ODs.

Trajectory Privacy: Privacy-preserving trajectory data publishing has been studied in literature extensively [**107**]. Compared to location privacy, trajectory privacy generally uses generative methods instead of location perturbation. Prefix-tree and human mobility model extraction approaches are the two main directions for trajectory privacy methods for DP. Researchers, in [**34**], apply DP with a prefix-tree data structure to user trajectory datasets by injecting noise to the count queries. A case study extension of this work with a real public transportation dataset is presented in [35]. More recently, several synthetic trajectory generation methods based on prefix-tree data structures with adaptive generalization approaches have been proposed [96, 231].

Another line of synthetic trajectory generation is based on modeling human movements [80,148]. This approach extracts features from user trajectories and injects controlled noise to the mobility distributions to make them private. However, human mobility characteristics are highly complicated, and the model-based methods do not capture the real mobility dynamics all the time [151]. Recently, synthetic data generation models with machine learning, especially deep learning, are attracting attention for either lack of available data or privacy concerns [60, 179]. Deep generative models-based privacy mechanisms have been proposed in literature to extract human mobility features with non-linear learners [75].

Instead of trajectory generation, several studies target different directions for the privacy of mobility trajectories. For example, dummy location injection [136], location swapping in the mixed zone [184], location generalization [125], and trajectory reconstruction [41] are some of the proposed approaches for trajectory privacy.

Since dealing with location sequences is challenging in the continuous domain, proposed schemes are generally in the discretized grid domain. However, having a grid-like discrete representation cannot prevent geospatial mismatching. For instance, when a location is randomly sampled from a grid where the road network is sparse, mostly generated sample points to a non-sense location. This restriction practically results in higher utility loss. So instead, DPMM discretizes the locations to road segments, resulting in more realistic trajectories.

6.3. Methodology Overview

Protecting personally identifiable information is crucial before publishing the user mobility data. Differential privacy is a probabilistic approach that provides privacy through noise injection and/or randomized selection. We propose a method for generating differentially private mobility trajectories with map-matching, called DPMM, to protect personal identifiers. This section summarizes the notations and definitions that are required for the proposed DPMM privacy model.

6.3.1. Notations and Metrics. Let $D(V, E)$ represent the road network as a weighted digraph, where the set of nodes V correspond to a road intersection, the set of edges E to roads, and weights that represent link metrics, such as length of the link or traffic volume. A link $\phi \in E$ connects intersections u and v where specific link attributes, such as the number of lanes, and speed limit, are stored in the link description. We have two sets of trajectories: GPS trajectories and link trajectories. Let us define the GPS trajectories and then link trajectories:

1) GPS Trajectories: A sequence of GPS coordinates with l number of samples $\mathbf{p} \in T = \{\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_l\}$ form a GPS trajectory that reflects the continuous motion of the object. The set of all GPS trajectories are Ψ where $T \in \Psi$.

2) Link Trajectories: Given m number of vehicles $\Phi \in \Lambda = \{\Phi_1, \Phi_1, ..., \Phi_m\}$ on the road network, each vehicle travels between ODs using an ordered link path generating a user travel path known as a *micro-graph* $\Phi_i \in D$. Every link trajectory has n number of links $\phi \in \Phi = \{\phi_1, \phi_2, ..., \phi_i, ..., \phi_n\}$ and $\Phi \subset E$. The raw link trajectory is Λ and the privacy preserved link trajectory is Σ. The goal of our research is to release the privacy preserved link trajectories using the raw trajectories $\Lambda \to \Sigma$. Every link in network $D(V, E)$ includes the road characteristics. Each link ϕ is classified into one of five classes in terms of the capacity and functional role of the road, called a functional class. Arterial roads have lower functional classes, rural streets have higher functional classes. Next, we introduce the general concept of map matching algorithms.

3) Map-Matching: GPS coordinates are an estimate of a device's location using satellite broadcast information. However, these locations do not always represent the exact travel path due to several intrinsic and environmental errors such as satellite geometry, signal blockage, tree cover, or urban canyons [**17**]. Consequently, GPS locations may not match a link on the road network. Map-matching generates an ordered set of road network links describing the user's trajectory considering the road network $D(V, E)$ and GPS points [**172**].

Map-matching algorithms play an essential role for transportation engineers as part of trajectory processing to minimize trajectory errors [**31**]. Since most GPS trajectories already require map-matching as a pre-processing before using them in transportation applications, DPMM eases the burden of map matching by generating privacy preserved link trajectories given raw GPS trajectories.

6.3.2. Differential Privacy. Location privacy and path privacy are the two main notion of privacy for trajectories in this work. We require that the output of a query statistically guarantees the privacy of individual user locations independent of the background knowledge. Differential privacy (DP) [**53**] guarantees that modifying the single input value has a negligible effect on the output statistical query. In this section, we summarize the general definitions and metrics of DP that are applicable to our problem.

We introduce the privacy concerning data $\mathbf{X} \in \mathcal{X}$ as vehicular mobility information in query $q \in \mathcal{Q}$. The data holder wants a mechanism that hides the sensitive information and reports the privacy preserved version of sensitive information using a randomized algorithm $\mathcal{A} : \mathbf{X} \times \mathcal{Q} \to \mathcal{D}$ where $\mathcal{Q}$ is the query space and $\mathcal{D}$ is the output space. DP promises that the algorithm $\mathcal{A}$ is differentially private such that participation or removal of a record results in minimal changes to the output of a query.

Let us first define the neighboring datasets:

Definition 9 (Neighboring Dataset). *Considering two databases* $\mathbf{X}$ *and* $\mathbf{X}'$, *if they differ by only one element* $\mathbf{x_i} \to \mathbf{x_i}'$ *corresponding to a link trajectory, they are neighboring datasets.*

The above definition formalizes the adjacent or neighboring dataset that plays a crucial role in differential privacy.

Definition 10 (ϵ-Differential Privacy). *Given for every neighboring sets* $d \subset \mathcal{D}$, *a randomized algorithm* $\mathcal{A}$ *is* ϵ*-differentially private if*

$$Pr(\mathcal{A}(X) \in d) \leq e^{\epsilon} Pr(\mathcal{A}(X') \in d) \tag{6.1}$$

where ϵ *is a positive real number and probability comes from the randomness of the algorithm.* $\frac{Pr(\mathcal{A}(X) \in d)}{Pr(\mathcal{A}(X') \in d)}$ *is the privacy leakage risk for the randomized algorithm* $\mathcal{A}$.

ϵ-differential privacy is known as randomized response where adding or removing a single element from the database results in a similar probability. The smaller value of ϵ represents higher privacy guarantee and provides in-distinguishability.

An appropriate epsilon, in DP, is typically determined based on the sensitivity of the underlying data. The definition of sensitivity is given in [**52**] as follows:

DEFINITION 11 (Sensitivity). *For any query function $f\colon D \rightarrow R^n$ that maps the dataset D to fixed sized real numbers, the sensitivity of f is defined as*

$$\Delta f = \max_{\mathbf{X},\mathbf{X}'} \left\| f(\mathbf{X}) - f(\mathbf{X}') \right\|_1 \tag{6.2}$$

for all neighboring datasets $\mathbf{X}$ and $\mathbf{X}'$.

DEFINITION 12 (Composition). *Let a set of randomized algorithms $\mathcal{A}_1, ..., \mathcal{A}_k$ that each $\mathcal{A}_i$ satisfies ϵ_i-DP.*

- *Sequential Composition: Let $\mathcal{A}$ be another randomized mechanism that executes $\mathcal{A}_1, ..., \mathcal{A}_k$ with independent randomness for each $\mathcal{A}_i$, then $\mathcal{A}$ satisfies $(\sum_i \epsilon_i)$-DP.*
- *Parallel Composition: Let dataset $\mathbf{X}$ is partitioned depterministically to different subsets $\mathbf{X}_1, ..., \mathbf{X}_k$ and executing each $\mathcal{A}_i$ with a different disjoint set $\mathbf{X}_i$ satisfies $\max_i(\epsilon_i)$-DP.*
- *Post-processing a randomized algorithm $\mathcal{A}$ that satisfies ϵ-DP does not break or consume any privacy budget.*

Given the composition properties and total ϵ privacy budget, the proposed DPMM builds different blocks carefully according to composition properties to achieve a DP satisfied randomized algorithm $\mathcal{A}$.

DP guarantees privacy for both numerical and non-numerical queries. While noise injection is a leading method for numerical queries, exponential mechanism is a mainly used mechanism for non-numerical queries [**52**, **144**].

Input perturbation and output perturbation are the two ways to implement DP. When we want to achieve OD location privacy on trajectories, one way to do is through input perturbation, where noise is injected into the GPS points. Using noise function $L(\epsilon, R)$, the GPS points are perturbed based on the below definition.

DEFINITION 13 (Laplace Mechanism). *For any function $f\colon D \rightarrow R^n$, the mechanism $\mathcal{A}$ gives ϵ-DP as follows:*

$$\mathcal{A}(D) = f(D) + Laplace(\epsilon, R) \tag{6.3}$$

Noise injection to the input can be done with different noise functions depending on the application requirements. Section 6.4.2 describes the additive noise method in detail. By injecting noise into the input GPS points, the method guarantees that the OD locations of trajectories are differentially private.

For privacy on non-numerical queries, exponential mechanism selects an output from input domain taking into consideration of a score function $q(\mathbf{X}, r)$ where r is the discrete output from the domain. Exponential mechanism assigns higher probabilities for the higher score to incentivize the higher utility outcomes.

Definition 14. *[Exponential Mechanism] Let $q : (\mathbf{X}, \mathbf{R}) \rightarrow R$ be a score function for a database $\mathbf{X}$ and domain specific discrete outputs R, the algorithm $\mathcal{A}$,*

$$\mathcal{A}(D, q) = \left\{ \textit{return } r \in R \textit{ with probability} \propto \exp \frac{\epsilon q(D, r)}{2\Delta q} \right\} \tag{6.4}$$

satisfies ϵ-DP.

6.4. Private Map Matching

This section describes the components of the proposed DP-based map-matching algorithm for trajectory privacy that generates synthetic link-level user trajectories. DPMM guarantees statistical privacy protection for link trajectories with noisy ODs and randomized travel paths. Figure 6.1 shows the flowchart of our DPMM mechanism. We transform the point-wise GPS trajectories into an ordered series of road network links and enforce privacy on trajectories with the road segment.

6.4.1. Trajectories with Waypoints. Trajectories are time-ordered sequential location samples, and the sampling rate varies depending on the device. Before private path construction, we represent the trajectories with fewer GPS waypoints that retain the movement characteristics. This waypoint approach only preserves the critical locations enough for movement representation by removing insignificant locations. For example, in a higher sampling rate trajectory, sequential path construction may result in redundant extra paths due to frequent path findings (see Section 6.4.4 for more details). Furthermore, the frequent path selection also consumes more privacy budget ϵ. In summary, the waypoint representation enhances the path quality and decreases the computational complexity by dealing with fewer location pairs.

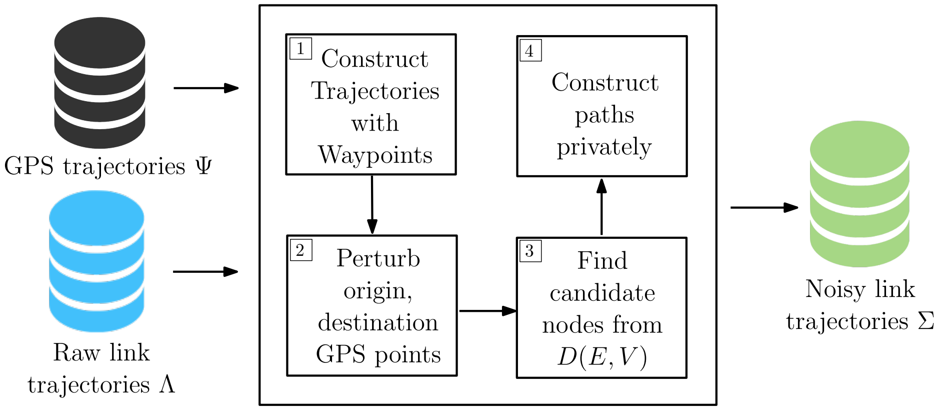

GPS trajectories Ψ
Raw link trajectories Λ
1 Construct Trajectories with Waypoints
2 Perturb origin, destination GPS points
3 Find candidate nodes from $D(E, V)$
4 Construct paths privately
Noisy link trajectories Σ

FIGURE 6.1. Differentially private link trajectory generation scheme.

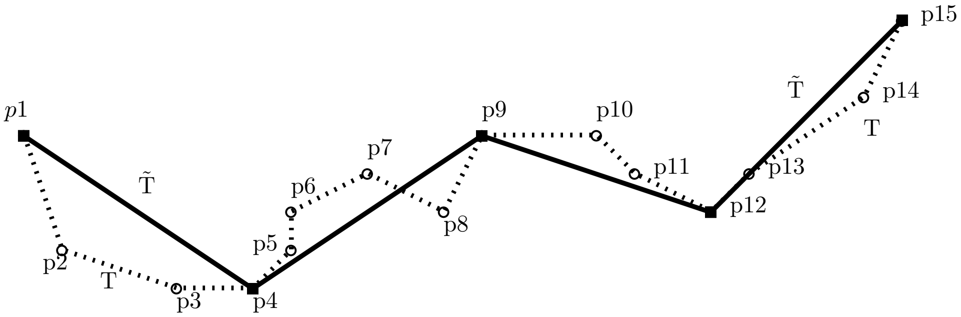


FIGURE 6.2. Trajectory Simplification

For a trajectory T, let n coordinates be $p_1, p_2, ..., p_n$ where every p_i is represented with (x_i, y_i) and $n-1$ line segments be $\overline{p_1p_2}, ..., \overline{p_{n-1}p_n}$. Figure 6.2 shows a toy trajectory simplification example from the original trajectory T to simplified trajectory $\tilde{T}$. Original trajectory has 15 coordinate points $p_1, p_2, ..., p_{15}$. Using trajectory simplification, we can represent trajectory T with waypoints $p_1, p_4, p_9, p_{12}, p_{15}$, which allow us to find approximate paths between distant points. The first step of the proposed DP mechanism is to represent trajectory with fewer waypoints.

In literature, there are several algorithms to decimate curves that are composed of line segments as we have in trajectories. We consider non-parametric Ramer–Douglas–Peucker (RDP) algorithm for representing higher sampling rate trajectories with sample waypoints [**49**]. RDP is a heuristic method that we attached to the DPMM to retain important GPS waypoints in the randomization process and help generate more practical travel paths.

RDP recursively approximates the whole trajectory to fewer points representation starting from $\overline{p_1p_n}$ line segment and an error bound σ, which also known as simplification error. RDP then calculates distance offset of each point coordinate from $p2$ to p_{n-1} with perpendicular distance. Let p_k be the point with maximum of perpendicular distances from $\overline{p_1p_n}$. If $\sigma_k > \sigma$, RDP splits the line segment into two sub-segments $\overline{p_1p_k}$ and $\overline{p_kp_n}$ where σ_k is the offset distance from p_k to $\overline{p_1p_n}$. The simplification continues recursively for $\overline{p_1p_k}$ and $\overline{p_kp_n}$. The RDP terminates if $\sigma_k \leq \sigma$ or $\overline{p_ip_j}$ is a consecutive segment with $j - i = 1$. It worth mentioning that RDP only removes the unnecessary middle points of trajectories by keeping the OD points in $\tilde{T}$. The time complexity of RDP is $\mathcal{O}(n^2)$.

6.4.2. Private Origin-Destinations. Traveling from one geographical location called the origin to another geographical location called the destination is sensitive information that must be protected. The map-matching algorithm infers the ordered set of road segments (links) using GPS locations from the $D(V, E)$ network and finds paths for each pair of user's GPS points.

We recently propose an adaptive noise injection model for location privacy on aggregated mobility networks in [**93**]. DPMM employs the previous noise injection methodology for trajectory privacy on ODs. This method injects adaptive Planar Laplace noise to the GPS points before matching them with an appropriate link to provide OD privacy on the map-matching algorithm.

The OD GPS points are obfuscated based on the network density with noise injection and they are matched with a new link. The two key parameters used for noise sampling are ϵ and R. While ϵ is responsible for the noise level, R is the distance parameter for moving the center of the noise in the geospatial domain. The output of the noise function is a new randomized GPS location in the same space.

Geo-indistinguishibility is one of the noise injection methods for hiding GPS locations [**7**]. The Laplace noise is sampled from a bounded probability density function on polar coordinate systems instead of Cartesian space as follows:

$$D_\epsilon(r, \theta) = \frac{\epsilon^2}{2\pi} r e^{-\epsilon r} \tag{6.5}$$

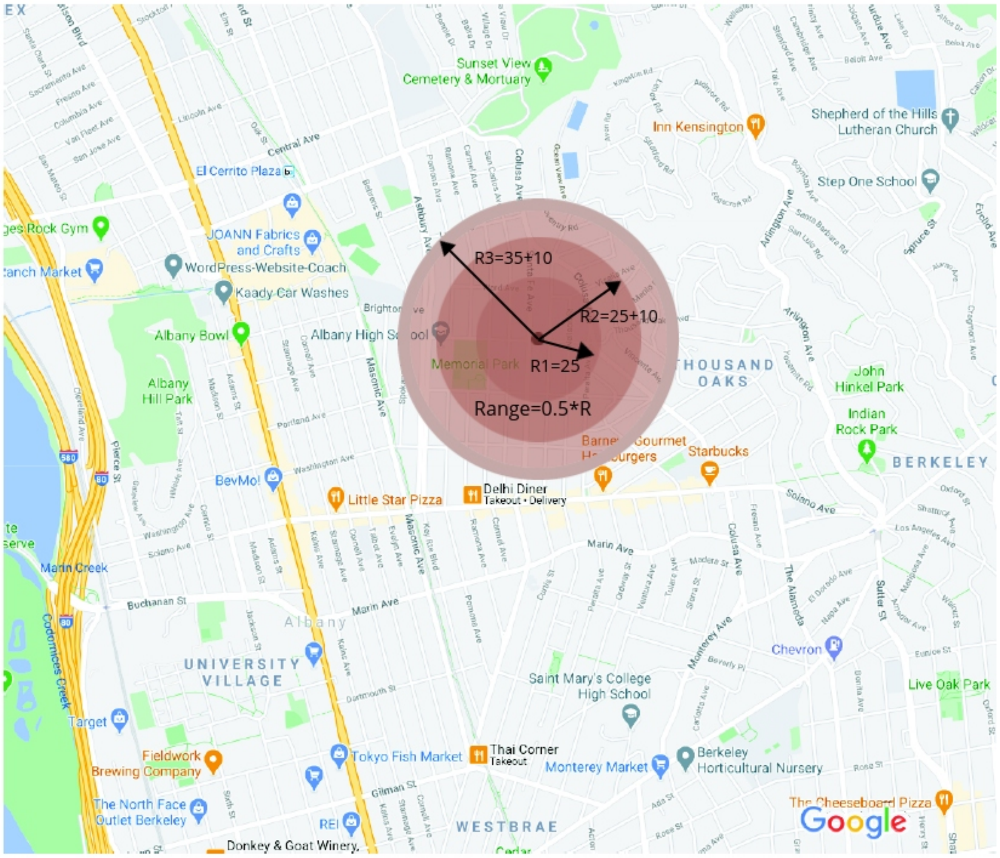


FIGURE 6.3. Buffer range for determining link density

where r is the distance of $\mathbf{x}$ from $\mathbf{x_0}$, θ is the angle and $\frac{\epsilon^2}{2\pi}$ is a normalization factor. While θ is random uniformly drawn from $[0, 2\pi)$, ϵ is direct user input and r is scaled with given radius R from the input. We refer readers to the original study for the details of noise sampling [**7**].

The experimented geo-indistinguishability provides location privacy by adding controlled noise $L(\epsilon, R)$ to the OD GPS points x_i within a certain range R in order to mask the actual locations using density based noise range selection method R.

Density-Aware Noise Injection: This section explains the density-aware structure of the noise injection approach using Planar Laplace noise proposed in [**7**]. Randomly injecting noise without considering the network's density would not achieve the desired privacy level all the time. The DPMM provides location privacy for trajectories even for outliers by selecting noise level adaptively with respect to the link density of network $D(V, E)$

Algorithm 5 Adaptive noise magnitude selection

```
***Input*** $h_1$ for the number of links in the buffer range
***Input*** $h_2$ for the number of links in the buffer range that belong to the same functional class
***Input*** $Z$ initial buffer range
$LinkSet \leftarrow$ empty set
$LinkSetFC \leftarrow$ empty set
**while** Size of $LinkSet \leq h_1$ or Size of $LinkSetFC \leq h_2$ **do**
    Find the $LinkSet$ links within buffer zone
    Find the $LinkSetFC$ links within buffer zone
    Z+=10 meters
**end while**
$R \leftarrow \frac{1}{2}Z$
**return** $R$ and $LinkSetFC$
```

The ODs of trajectories are the most vulnerable parts due to revealing users' start and end locations, such as home or office addresses. Therefore, providing privacy for ODs requires much attention. In this work, we consider the link density around the OD of trajectories to define the level of noise that needs to be injected. As we mentioned earlier, every link has functional class information, and DPMM moves the GPS point to a place that matches a new link with the same functional class of the original link.

Link density in the road network quantifies the populations in general. While central areas have more streets and intersections, which implies more population, the rural places have fewer road segment connections due to limited populations. Therefore, it is easier to provide privacy for the people who live in central areas. On the other hand, it is hard for the rural areas since location traces are unique in the outskirts of the communities. We define a density function for noise injection as follows:

Definition 15. *[Density Function] Given the ϵ value, radius R of the noise function $L(\epsilon, R)$ is selected with respect to $R = f(\theta)$ where θ is the network density in terms of the number of links (road segments).*

For each trajectory, OD GPS points are perturbed with the noise injection model. The DPMM adjusts the noise using the link density around the GPS point with respect to a cloaking region (see Fig. 6.3). To do so, starting from an initial radius Z, the proposed mechanism increases the radius Z until finding a certain number of links and the same functional class links. The number of all links, h_1, and the number of the same functional class links, h_2, are user-defined parameters based on the geographical region and density of the network $D(V, E)$. Sparse vs dense structured regions or shapely vs end-to-end intersection-based network would require different hyperparameters. For example, this project considers shapely road network, which divides the end-to-end intersection road link to the small links based on the road curves and doing map-matching with a different road network requires different h_1 and h_2 hyperparameters. Once the number of all links and the same functional class links reach the thresholds h_1 and h_2, the center of the final radius Z, which satisfies the two thresholds, is selected as the input for noise function $L(\epsilon, R)$ where $R = \frac{1}{2}Z$. As the Laplace noise is 2-dimensional, we select the half of distance value Z for this noise model and sample a GPS point with given parameters (see Algorithm 5). After the noisy GPS point is returned from $L(\epsilon, R)$, all the nodes belonging to the same function class links are selected as candidate nodes for path construction.

6.4.3. Candidate Nodes. To construct the path of a trajectory using the network $D(V, E)$, map-matching first needs to have candidate nodes for each GPS point. However, due to geospatial constraints, selecting a single candidate node given the GPS point does not guarantee to match with the correct node. For instance, if the GPS is close to a one-way road and a two-way street with a similar distance, the GPS point may belong to both. Selecting the best node depends on the direction and the next GPS point. To mitigate the geospatial constraints, we propose to choose a set of candidate nodes to find paths. Besides, selecting a travel path randomly using multiple candidate paths increases the privacy (see Section 6.3.2).

While we select candidate nodes for OD GPS points from the same functional class links using threshold h_2, for waypoints, we find candidate nodes from all the links using the threshold h_1. For every waypoint, we follow the same cloaking-region approach we followed for noise injection to find candidate nodes (see Fig. 6.3). However, we do not restrict candidate links to have the same functional class criteria for waypoints to increase the randomization in the path construction. The cloaking region method takes the following inputs for each waypoint from $\tilde{T}$: threshold h_1 for searching the number of links, initial radius Z, and road network $D(V, E)$. The output is h_1 number of links, and the nodes that belongs to those links are collected as a candidate node-set. Candidate nodes for each waypoint are stored in separate containers. For our experiments, we prefer to use the same threshold h_1 for ODs and waypoints. Still, the parameters can be adjusted depending on the geographical region and network $D(V, E)$ structure. Algorithm 6 summarises the candidate node selection.

Algorithm 6 Candidate node selection

```
Input $h_1$ for the number of links in the buffer range
Input $Z$ initial buffer range $LinkSet \leftarrow$ empty set
while Size of $LinkSet \leq h_1$ do
    Find the $LinkSet$ links within buffer zone Z
    Z+=10 meters
end while
return  Candidate Nodes from $LinkSet$
```

6.4.4. Private Paths. A user's travel path could allow an adversary to infer further information about the user's identity by linking other available information to the user path. For instance, the adversary may know several locations of a user, such as home or office location and specific automatic toll booths that the user passed through. If a path matches with known locations, the adversary may identify the user. Since locations are sensitive and easy to link with user identities, minimizing the actual travel paths of a user reduces the risk of re-identification by adversaries. Instead of revealing the true path of a trajectory, randomizing the paths in the same trajectory direction using waypoints limits the true travel path while providing similar travel within the same vicinity.

DPMM selects the travel routes for the sequence of waypoints randomly to construct privacy-preserving paths. First, for a simplified trajectory $\tilde{T}$, the proposed method finds candidate paths between waypoints using the candidate node-set, which is constructed using Algorithm 6. Then, since we do not intend to find the shortest path, we implement A^* path finding algorithm with the euclidean distance between nodes as heuristics [**85**]. A^* algorithm combines the Dijkstra shortest path algorithm with greedy search methods [**46**] and finds reasonable paths by using heuristics to guide the path finding direction.

For every pair of points p_i and p_j in $\tilde{T}$, candidate paths are stored with the corresponding travel distance. The proposed DPMM selects a travel path randomly with probability proportional to the travel distance. The shorter travel distance has a higher chance of being traveled by the user. Therefore, the selection mechanism assigns a higher probability to the shorter travel distance path. To achieve this, we inversely normalized the distances between 0 and 1. Then, we select the path privately using DP exponential mechanism. Note that our model's sensitivity Δp is 1 because maximum travel distance is always bounded to 1 due to normalization. DPMM follows this procedure sequentially, and the final private link trajectory protects the user travel paths along with OD privacy.

6.4.5. Travel Path Adjustment. The network used for map-matching is a directed graph. Depending on the road traffic direction, network $D(V, E)$ has separate links for incoming and outgoing links. Randomized path selection sometimes may result in unreasonable travel paths that go reverse and make a u-turn or o-turn reaching the same node visited before. We remove the travel loops after private path selection to prevent redundant paths taken by the private map-matching. Our experimental analysis shows that the loops on raw trajectories are less than 1% in our dataset; removing the loops after private path selection decreases utility loss. Note that the ϵ-DP privacy guarantee still holds with post-processing.

6.4.6. Complete Trajectory Construction. The proposed private map-matching algorithm combines noise injection and private selection DP methods, as we discussed in separate sections above. Algorithm 7 summarizes the privacy protection mechanism. First, the algorithm creates a waypoints trajectory $\tilde{T}$ by keeping the OD as it is. Next, it injects the proposed adaptive Laplace noise to the OD GPS points and forms candidate nodes from the same functional class links. The third step of the proposed algorithm finds candidate nodes for every waypoint in $\tilde{T}$. In the fourth step, the proposed mechanism finds candidate paths between every consecutive node-set using A^* routing algorithm. Then, it selects paths privately from the candidate paths using the exponential-DP method. Finally, it connects selected candidate paths and removes the travel loops. The algorithm terminates after generating all the private link trajectories from GPS trajectories.

6.5. System Analysis

6.5.1. Differential Privacy Analysis. The DPMM distributes the privacy budget ϵ evenly to the sub-processes while guaranteeing ϵ-DP. Representing the raw GPS trajectory with $s + 1$ waypoints including ODs results in s paths that needs to be private. Total ϵ budget divided to ϵ_i for OD noise injection and number of waypoints such that $\sum_{i=1}^{2+s} \epsilon_i$. While OD noise injection provides

Algorithm 7 Privacy Preserving Map-Matching

```
Input $\Lambda, \Psi$, $D(V, E)$,
$h_1$ for number of links in the buffer range,
$h_2$ for number of same functional class links in the buffer range,
$Z$ initial buffer range
for $T \in \Psi$ do
    Build waypoints trajectory $\tilde{T}$ from $T$ using RDP
    for $p \in \tilde{T}$ do
        if $p$ is Origin or Destination then
            Select $R$ and $BufferSetFC$ using Algorithm 5
            Inject adaptive noise to the GPS point $p$ using $L(\epsilon, R)$
            Form candidate nodes set from $BufferSetFC$ links
        else
            Form candidate nodes from Algorithm 6
        end if
    end for
    Find candidate paths with $A^*$ for candidate nodes
    Select private paths with exponential-DP mechanism
    Connect privately selected paths
    Remove the node loops as in Section 6.4.5
    Build the noisy link matched trajectory with connected links
end for
return Noisy link trajectories $\Sigma$
```

privacy with the property of parallel composition, private path construction provides privacy with sequential composition. Post-processing on map-matched trajectories, such as removing the travel loops, does not violate the ϵ-DP privacy.

The smaller value of ϵ represents higher privacy and indistinguishability, whereas higher ϵ gives more accuracy to the output trajectory. Due to the geospatial and temporal nature of user movements, it is also essential to preserve the accuracy of the generated trajectories while achieving a reasonable privacy guarantee. The data owner can adjust the privacy budget with respect to the sensitivity for both OD and path privacy. If the data owner wants to hide the ODs more, he/she can select a smaller ϵ value for noise injection to the ODs, which increases the perturbation. The same analogy can be applied to path privacy too. In summary, we left privacy budged distribution to the data owner, and this aspect is out of the scope of this work.

6.5.2. Attack Resilience.

Outlier Leakage. A trajectory may have OD points that are unique in a sense and reveal vulnerable information about user identity [**80**]. Threat on outlier trajectories mainly applies to rural areas, such as travel between a hospital and a farmhouse. Injecting the same noise magnitude to all GPS points cannot provide privacy for every GPS point. Moving GPS points slightly can provide privacy in central locations. However, repositioning locations in an outlier area at the same level as in central areas may not offer the same privacy. The proposed privacy mechanism deals with outlier trajectory ODs by perturbing them adaptively with respect to road segment density.

Partial Sniffing. An adversary may have access to a sub-trajectory of a user that participated in the trajectory dataset through physical tracking or social networking. Then, an adversary may try to infer the rest of the user travel that passes through the locations in the sub-trajectory. Let a user's sub-trajectory T_{sub} be known by the adversary; there is a high chance to reveal the user's rest of the travel if the adversary can find a matching T from trajectory database [**80**]. DPMM prevents adversaries from making such inferences with two concepts: OD privacy and path privacy. For example, a true trajectory may travel from a local street to a hospital. When an adversary gets access to a partial trajectory T_{sub} of user trajectory T, he/she may try to infer the home address and the purpose of the travel. However, since the proposed privacy mechanism does not disclose the true ODs and travel path, the adversary cannot correctly identify the user information from the privacy preserved trajectory $T_p \in \Sigma$.

6.6. Evaluation

6.6.1. Dataset Description. This project uses a real-world dataset collected in the San Francisco Bay area in California with fleet and consumer GPS trajectories. We process one day one hour of trajectories (between 1 pm and 2 pm) from the city of San Francisco. In total, the experiments apply DPMM to 833 user trajectories. The dataset is created from various location-sharing applications and GPS tracking devices. When the tracking device is active, location (lat, lon), speed, and heading are collected along with a unique device identifier. Trajectories have varying sampling rates due to being collected from different sources. However, most of the trajectories have sampling rates of less than 1-minute.

6.6.2. Comparisons to Alternate Approaches. We compared the proposed privacy mechanism with two well-known DP-based private trajectory generators: AdaTrace [**80**], and DPT [**96**]. While AdaTrace generates synthetic trajectories by learning the mobility patterns, DPT constructs

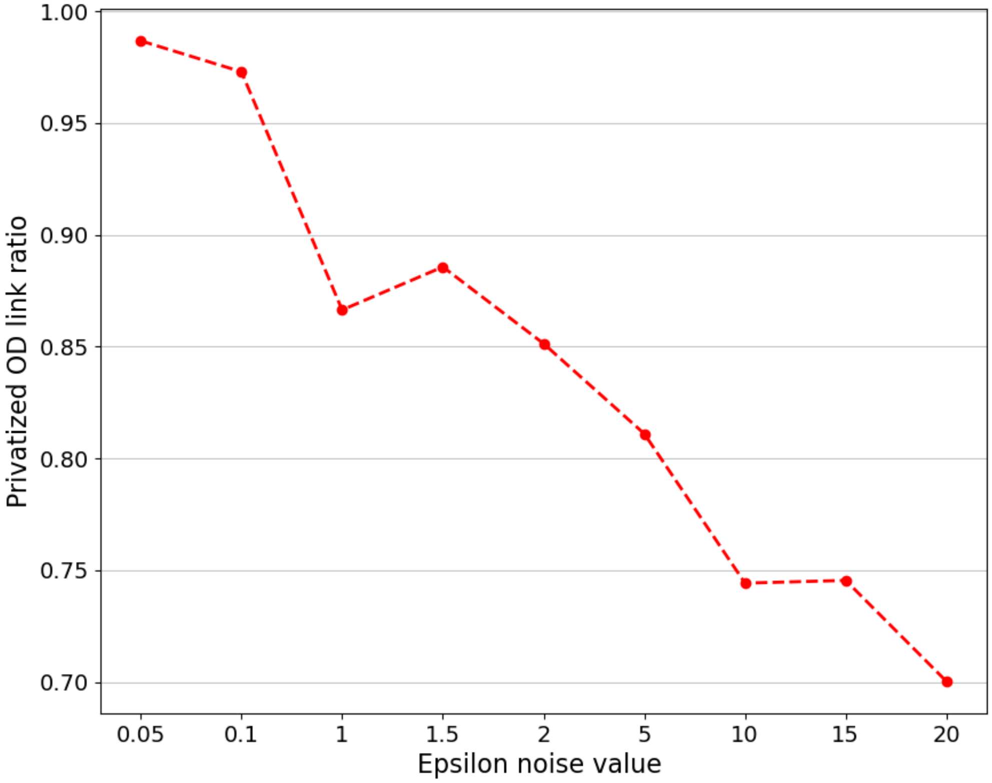
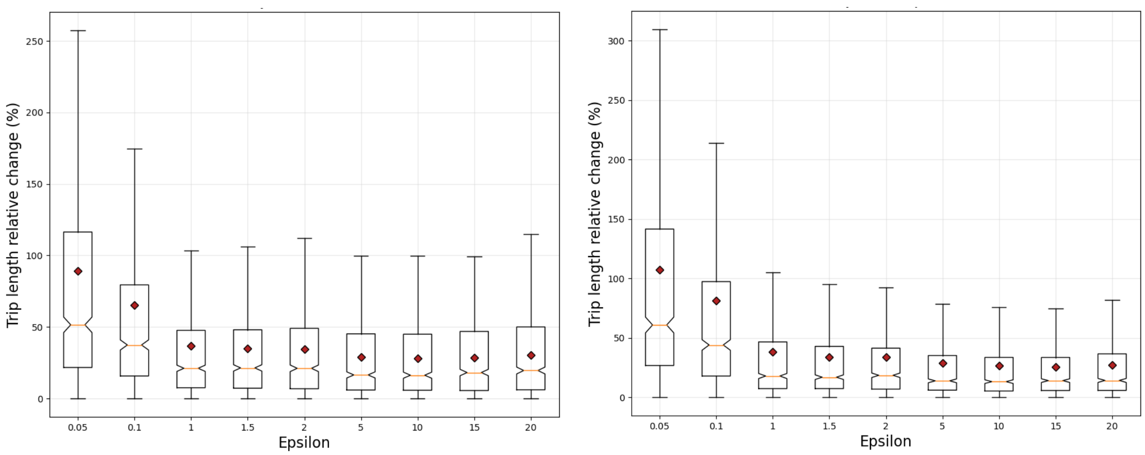


FIGURE 6.4. Comparison of different ϵ values and the change of OD-links for different for 1 hour period of trajectories between 1pm and 2pm.

prefix-tree to generate private user trajectories. We acquired the original implementations from respective authors. Both AdaTrace and DPT models generate synthetic GPS trajectories instead of link trajectories. For a fair comparison between the proposed DPMM and benchmark models, we applied map matching to AdaTrace's and DPT's GPS trajectories to generate equivalent link-level trajectories for our analysis. This is referred to as DP-free version of our map-matching algorithm.

The utility is closely related to the size of the database for benchmark AdaTrace [**80**] and DPT [**96**] models. However, the utility for DPMM is bounded by the density of the road network. Therefore, to achieve better utility for the benchmark models, we trained their respective implementations with a whole day of trajectories within the region. The total number of GPS trajectories for one day in San Francisco city in our database is 9249.

Along with other studies in the literature, we also compare DPMM method with different variants:



(a) DPMM vs original link trajectories

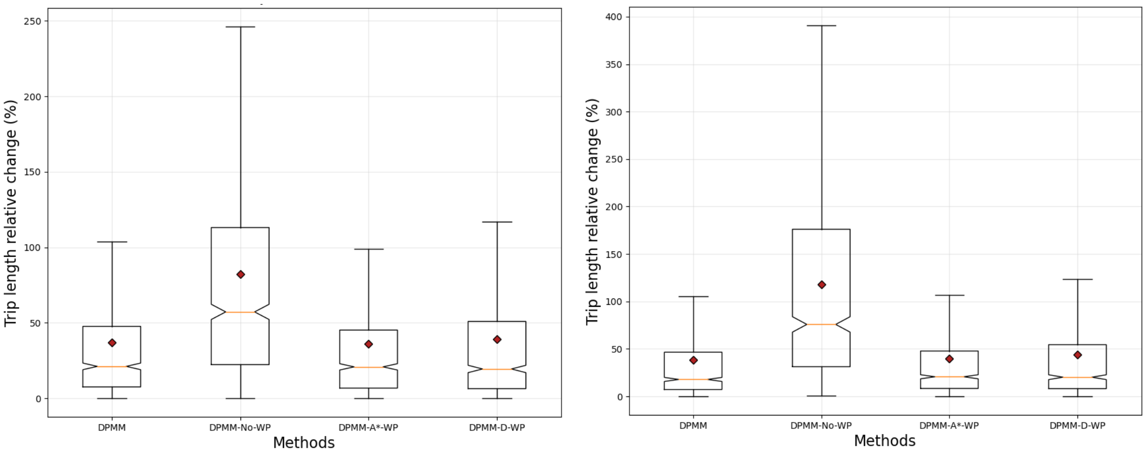
(b) DPMM vs GPS trajectories

FIGURE 6.5. Performance of DPMM is compared with the different ϵ values with respect to original link and GPS trajectories.

- **DPMM-No-WP:** This version performs the same privacy mechanism for OD while selecting paths from trajectory T without waypoint sampling and not experimenting trajectory post-processing (removing the loops).
- **DPMM-A*-WP:** Based on DPMM-No-WP, DPMM-A*-Way adds waypoint sampling to base model in order to provide more privacy and less utility loss. This version does not perform trajectory post-processing (removing the loops).
- **DPMM-D-WP:** Following the DPMM-A*-Way, this method uses Dijkstra path finding algorithm instead of A^*. Dijkstra guarantees to find the shortest path while A^* does not.

6.6.3. Utility Metrics. We have chosen utility metrics commonly used in transportation studies including individual trajectory level and aggregated level queries. In this section, we explain the importance of the utility metrics and present some use-case examples as we define the metrics. The goal is to have a higher similarity in the utility metrics between original and privacy-preserved trajectories given the same level of privacy protection.

Individual Utility Metrics: Mobility trajectories are complicated, and evaluating the quality of privacy-preserved trajectories with aggregated statistics alone is not sufficient. For example, the OD Similarity metric for AdaTrace when compared to original trajectories (Table 6.1) indicates high level of OD similarity between the two. However, their respective actual trajectories show distinct differences (as shown in Figure 8(a)). Since the proposed DPMM perturbs only the OD GPS points, its distortion on the trajectory and geographical mismatch is limited.



(a) DPMM vs original link trajectories

(b) DPMM vs GPS trajectories

FIGURE 6.6. Performance of DPMM is compared with different methods with respect to original link and GPS trajectories on $\epsilon = 1.0$.

We evaluate the utility of the proposed DPMM model at the individual trajectory level with different variants of the DPMM mechanisms. The relative trip length change of the link trajectories before and after applying DPMM is proposed as a utility metric in this study. Without DPMM, the base map-matching algorithm matches the GPS points with the nearest links and connects such links with the shortest path algorithms. Using the same relative change formulation, we compared the change of the privacy preserved trajectories with clean link trajectories and GPS trajectories. The trip length of the GPS trajectories is calculated using the euclidean distance between the sequence of the GPS points.

Aggregated Utility Metrics: Spacial density analysis plays a key role in understanding human mobility [**86**]. Our first aggregated utility metric, mainly used for graph data, is the query error that quantifies the error in the characteristics of most visited places. Minimizing the query error makes output privatized data more useful [**34**, **80**, **217**]. For this metric, 500 road links are sampled uniformly across all regions from the network $D(V, E)$. Then, the normalized absolute difference between the number of real and synthetic trajectories passing through each link is computed by the following:

$$\text{error}(Q(\Sigma)) = \frac{|Q(\Psi) - Q(\Sigma)|}{\max\{Q(\Psi), s\}}, \tag{6.6}$$

where $Q(\Psi)$ and $Q(\Sigma)$ are the number of trajectories that pass the certain links for the set of original trajectories vs privacy preserved trajectories, respectively, and s is sanity bound for mitigating the effect of the extremely small selective queries. We specified the sanity bound s as 1% of the users.

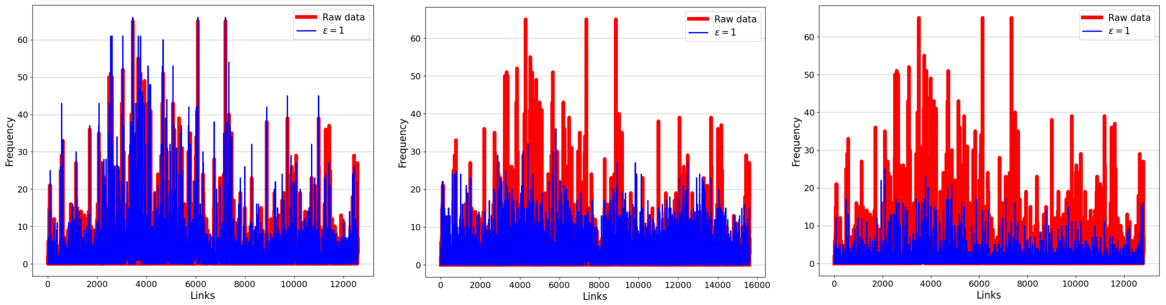
The travel characteristics of moving objects, such as personal vehicles and public transportation for spatio-temporal analysis can provide valuable insights for transportation analysts [**169**, **183**]. The second aggregated utility metrics measures the similarity of the OD distributions, called OD Similarity. This metric evaluates how much the overall characteristics are preserved in terms of OD links. Jensen-Shannon divergence (JSD) is a well-known similarity metric mainly used for measuring the similarity of two probability distributions [**131**]. We employ JSD for OD similarity.

The third metric measures the changes of the Vehicle Miles Traveled (VMT), which can be useful for different purposes, such as ride-sharing [**97**] and land use [**182**], for link trajectories, called VMT Change. Link count refers to the number of times a link occurs on the aggregated link trajectory network. The last utility metric compares link count distribution between original and privacy preserved link trajectories.

6.6.4. Numerical Results. We evaluate the performance of DPMM with benchmark studies and other DPMM variants from two different aspects: change in the privacy preserved trajectories at the individual level and aggregated level. When we apply the DPMM method to the trajectory database, depending on the privacy level ϵ, the utility varies in terms of the privatized OD link ratio and trip lengths. In addition, the experiments quantify the query similarity metrics at an aggregated level with respect to other trajectory privacy methods and compare the results with other studies in the literature. We use a range of ϵ values between 0.05 and 20 to evaluate the performance of the DPMM. The ϵ values are selected to reflect the lower and upper limits of the impact of the DPMM privacy mechanism.

Individual Trajectory-level Analysis: Regardless of the other user's movements, every OD link may have privacy concerns, and matching an OD with a different link hides the true end location of the user. We inspect the fraction of OD links that are different from the original raw trajectory after the proposed noise injection, which we refer to as the *privatized link ratio*. Depending on the noise level and the road network density $D(V, E)$, DPMM may still match the links with the same link after the noise injection. To quantify the privacy of our method, we inspect the privatized link ratio over the total number of OD links with respect to different ϵ values. For 833 trajectories, we have 1666 OD links. Fig. 6.4 shows the performance of DPMM in terms of OD privacy. The goal of the proposed mechanism is to move OD links to different links. Therefore the output is expected to have higher ratios for the lower level of ϵ. The highest level of the privatized link ratio is observed with the lowest $\epsilon = 0.05$ with an average of 98.7%.

Next, Figure 6.5 quantifies the absolute trip length change with different ϵ values at the trajectory level. Figure 5(a) and Figure 5(b) illustrate the comparison of DPMM link trajectories with original link trajectories and GPS trajectories, respectively. The goal is to retain higher utility with a lower ϵ value. The lowest value of $\epsilon = 0.05$ generates the highest dissimilarity between privacy-preserved DPMM link trajectories and original link and GPS trajectories. The distortion in DPMM link trajectories is sensitive to the geographical region, road link density, and the link functional class. The average trip length change varies between 89% and 30% for original link trajectories and between 107% and 27% for GPS trajectories on different ϵ values. For instance, for $\epsilon = 1$, the absolute average distortion on trip lengths is 36.8% and 38.1% for original link trajectories and GPS trajectories, respectively. Since the sequence of GPS trajectories do not reflect the actual trip length, having a higher trip length error regarding link trajectory is expected. Increasing the ϵ noise value decreases distortion and the level of privacy that DPMM guarantees.



(a) DPMM (b) AdaTrace (c) DPT

FIGURE 6.7. Link count densities on aggregated network level for DPMM with baseline comparisons using $\epsilon = 1$

The last individual-level utility metric compares the proposed DPMM with other variants in terms of the change in the trajectory trip length (see Figure 6.6). The ϵ is selected as 1.0 for this part of the experiments. Figure 6(a) and Figure 6(b) illustrates the change in the trip length of different DPMM variants where DPMM outperforms the others with the lowest mean difference. The results clearly show the impact of the trajectory sampling from T to $\tilde{T}$ and removing the loop in the trips regarding the utility with the same privacy level. Another interesting observation is that the Dijkstra algorithm has very close similarity with A^* routing algorithm. While A^* does not guarantee the shortest path, the Dijkstra algorithm guarantees to find the shortest path. Since the users do not take the shortest path all the time, selecting a candidate path using A^* routing algorithm is a more reasonable choice due to the unpredictability of user behaviors.

Aggregate-level Analysis: Table 6.1 presents the aggregated utility results of different metrics for $\epsilon = 1.0$. The proposed DPMM performs better for all metrics due to its ability to handle each trajectory separately. On the other hand, AdaTrace and DPT achieve varying performance on different metrics. While DPT outperforms the AdaTrace in terms of the most visited places (*Query Error*) and origin-destination densities (*OD Similarity*), AdaTrace produces more similar trajectories to proposed DPMM in terms of the trip length, as shown with the *VMT Change* statistics. In summary, the DPMM succeeds in keeping the trajectory patterns in the same region while hiding true OD locations and travel paths. Therefore, the results in Table 6.1 reflect the superiority of the proposed algorithm.

Next, we evaluate the proposed DPMM and benchmarks with original link trajectories in terms of link count distribution to understand how link counts changes as a function of privacy. Figure 6.7 shows the link count distribution with respect to the original trajectories for different privacy

TABLE 6.1. Comparison of the aggregated utility metrics with benchmark studies for $\epsilon = 1$. The lower value is the better for Querry Error and OD Similarity metrics. For VMT Error, value closer to zero is better. The bold and green results show the best performance and the second best performance, respectively.

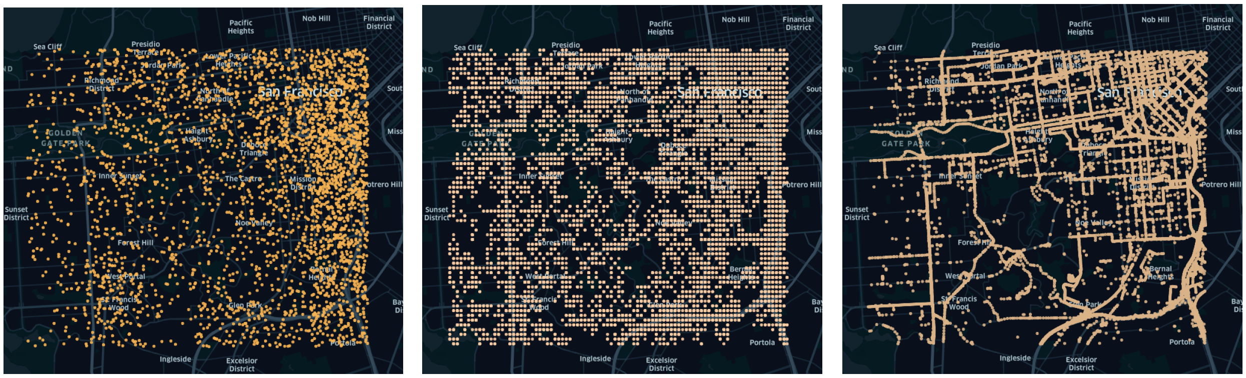
	DPMM	AdaTrace	DPT
Query Error	**0.146**	0.353	0.264
OD Similarity	**0.065**	0.081	0.068
VMT Change	−**0.072**	0.164	−0.641

mechanism with $\epsilon = 1$, the ideal privacy level. The results illustrate that DPMM preserves the link densities compared to baseline models AdaTrace and DPT.

Finally, we compare the spatial densities of the benchmark models with the original trajectories using the same number of samples. Figure 6.8 shows the visual representation of spatial densities for the raw GPS points, AdaTrace [**80**] and DPT [**96**]. The population densities and major routes are clearly observed in raw GPS distributions. However, AdaTrace and DPT has some sort of density awareness while missing the major routes. Note that since AdaTrace and DPT do not consider geospatial constraints, resulting trajectories are sampled in traffic-free areas such as city-parks and national-preserve areas. Compared to these baselines, the proposed DPMM model provides privacy-protected trajectories at the road network level that prevents to have such unrealistic trajectories.

6.7. Conclusion

In this paper, we present a differentially-private map-matching algorithm for the privacy of mobility trajectories. Proposed mechanism protects individual OD locations with adaptive noise injection model and travel paths with exponential DP method. The DPMM injects planar Laplace noise to the individual OD GPS points by considering the density of the localized road network and the functional class of the links. The actual perturbation level for each GPS point is adjusted by



(a) AdaTrace (b) DPT (c) Original Trajectories

Figure 6.8. Visual representation of the original trajectories vs privacy preserved trajectory densities for benchmark models. Proposed DPMM does not produce GPS trajectories, hence, it does not have visual comparison with benchmarks.

considering the localized link density. Next, proposed DPMM uses a waypoint sampling method for constructing travel paths privately. We evaluate our DPMM method for a variety of noise levels by comparing it with several comparative privacy models at individual trajectory and aggregated statistics.

The advantage over the literature of DPMM does not rely on population density with respect to other samples in the database, rather it considers link density in the road network. Due to map-matching, DPMM prevent geographical mismatches with the road structures which is a common problem for other baseline models. While this project provides OD location privacy with travel path privacy for individual user trajectories, DPMM does not guarantee the generation of the repeated trajectories due to the randomized nature of the mechanism. This resulting distortion is a form of the utility trade-off. Future work will include extending this investigation to different types of mobility datasets while also addressing the aforementioned limitations.

CHAPTER 7

MobilityGPT: Enhanced Human Mobility Modeling with a GPT model

7.1. Introduction

The widespread integration of location-based services and smart GPS devices, such as smartphones and watches, has made continuous monitoring of human mobility both desirable and feasible. These technologies capture diverse and detailed human movement information, with mobility trajectories representing the finest granularity of individual-level mobility characteristics. Such trajectories are crucial in various applications, including mobility modeling, commercial business analysis, and disease spread control [**30**, **190**].

Despite increasing demand for human mobility trajectory datasets, numerous challenges hinder their access and distribution [**115**]. First, these datasets are typically collected by private companies

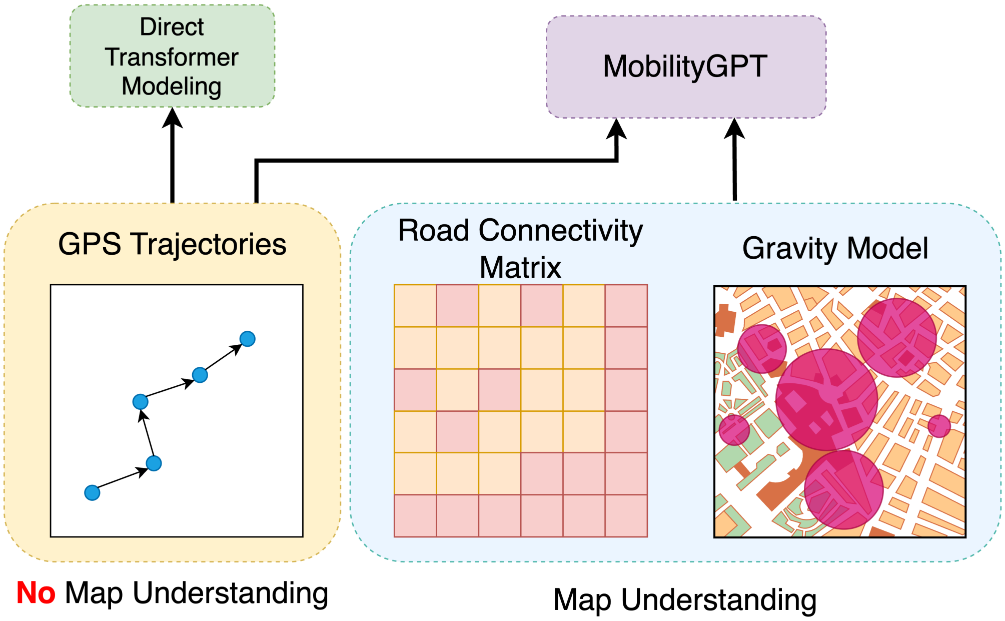


FIGURE 7.1. Generative models that do not incorporate mobility characteristics struggle to capture human mobility patterns accurately.

or government agencies, hence presenting privacy concerns due to the potential disclosure of an individual's sensitive lifestyle patterns (e.g., home or office addresses and points of interest). Legal implications, such as those outlined in the California Data Privacy Act and GDPR, highlight the importance of carefully handling spatiotemporal traces. Second, datasets owned by companies may expose proprietary business models and are often inaccessible for research purposes [**204**]. Lastly, publicly available datasets often lack diversity or quality, with gaps in data points and intrinsic noise that significantly reduce their utility. These limitations impede the progress of urban planning or transportation research [**139**]. Therefore, it is imperative to establish alternative trajectory data sources that are both high-quality and accessible for research purposes.

Current generative models designed for human mobility exhibit multiple drawbacks: they fail to capture the sequential mobility characteristics, often lack the continuity of generated trajectories, and do not consistently adhere to geospatial constraints. Approaches based on GANs [**74**] and VAEs [**114**] structure data in a tabular format, which, unfortunately, fails to preserve the inherent correlations among locations. Models like LSTMs, and sequential GANs [**225**] struggle to maintain the smooth continuity of locations and capture realistic human mobility characteristics. A further limitation of these methods lies in the elevated mismatches observed in geospatial location sequences. In short, a successful generative model for trajectory datasets should precisely grasp the intrinsic spatial-temporal behaviors.

Large language models (LLM), such as GPT (Generative Pre-trained Transformer), offer a robust foundation for sequence generation tasks and can be refined through fine-tuning to cater to distinct objectives. Trajectories and sentences share several similarities that make GPTs a promising approach for trajectory generation. Both consist of ordered sets of elements chosen from finite pools (road links and words, respectively). They exhibit semantic or spatio-temporal relationships, adhering to distinct rule systems such as language rules for sentences and geographical constraints for trajectories. Due to these parallels, the techniques developed for natural language processing can be adapted to model and generate realistic trajectories. [**152**] Besides, the gravity model holds significance in human mobility modeling as it provides a structured framework for estimating and understanding the flow of interactions between different locations [**238**]. Using gravity as part of human mobility modeling could bring further advantages to synthetic trajectory generation tasks.

In this study, we leverage the power of GPT for synthetic trajectory generation. However, directly training GPT on mobility data without geospatial insights would generate unrealistic sequences. To generate more realistic synthetic trajectories, we introduced several innovative methods, including (1) gravity-aware sampling, which incorporates gravity modeling of trajectory data into training updates, and (2) the use of road connectivity matrix (RCM) masking to eliminate disconnected location sequences from logits (see Figure 7.1). Pretraining GPT models on trajectory data captures the general sequence characteristics in an unstructured manner. We, furthermore, proposed an automated fine-tuning pipeline that improves the trajectory quality by leveraging transportation-specific metrics to evaluate and optimize the generated sequences without relying on human labeling. Our multi-objective framework, MobilityGPT, not only extends the application of GPT models to human mobility modeling but also collectively enhances the model's capacity to capture intricate patterns and adhere to realistic geospatial constraints (e.g., trajectories consist of road segments that are indeed connected).

Our contributions can be summarized as follows:

- We propose an LLM-inspired MobilityGPT human mobility modeling approach incorporating multiple unique features with a geospatially aware pretraining method and automated fine-tuning approach without human feedback.
- While gravity-aware sampling trains a generative model with respect to Origin-Destination (OD) pair-gravity values, the next sequence prediction is conditioned on the RCM that preserves the continuity of the generated trajectories.
- We propose a novel method for constructing a preference dataset to fine-tune MobilityGPT using reinforcement learning, enhancing the similarity of generated trajectories in terms of travel length.
- Through comprehensive experimental analysis using real datasets, we demonstrated that the proposed methods can generate trajectories with high fidelity while preserving essential statistical and semantic properties of human mobility.

7.2. Related Work

Generating synthetic mobility trajectories from real data is a promising approach for protecting sensitive information [**109**]. Traditional *model-based* methods, such as those parameterizing human mobility, have limitations due to the complexity of mobility characteristics. Early data-driven approaches, employing techniques like gravity models and decision trees [**105**, **165**], face challenges in capturing sequential transitions between locations. In contrast, *model-free* techniques, particularly generative machine learning models, have gained traction for their ability to learn from data without external input parameters [**12**]. Such generative models have been explored for human mobility modeling using various data representations, including grids in tabular format [**158**], image-like trajectory modeling [**24**], and sequential grid format [**60**]. However, these models exhibit limitations in generating trajectories that accurately capture geospatial complexity.

One promising generative model direction relies on GPS sampling. Due to mismatches, the LSTM-based trajectory generation model cannot provide continuous motion of human mobility in diverse geospatial areas [**179**]. A similar continuous GPS trajectory generation method with a U-Net neural network using Diffusion models is proposed in [**236**]. Another generative direction with ML for mobility trajectories is employing two-stage training methods [**106**, **137**, **210**]. A recent GAN-based model, relying on road links with the A^* path-finding algorithm between regions employing two-stage GANs, has shown promise in identifying diverse paths for specified origin-destination pairs [**106**]. However, it falls short in generating synthetic trajectories, as the generation process initiates with an origin-destination pair from testing trajectories. Finally, sequential GAN methods have also been employed for mobility modeling with different variants [**60**, **225**]. Unlike previous methods, MobilityGPT leverages the auto-regressive GPT model for trajectory generation, enabling a comprehensive exploration of spatial and temporal distributions with high-quality outputs by incorporating road links into sequence modeling.

The GPT models, developed by [**174**], transformed the field of Natural Language Processing (NLP) with transformer's capabilities. GPT, with its pre-training on diverse language data, demonstrated remarkable proficiency in understanding and generating coherent sequences to support various NLP tasks. The transformer-based GPT concept has been applied to diverse applications, from vision [**33**], music [**11**] to network data [**113**]. In addition, researchers have made great progress on various fine-tuning models for enhancing the GPT model capabilities [**21**, **122**, **230**]. However, these fine-tuning models mainly rely on human feedback that evaluates the quality of the generated sequences. This study proposes a geospatially-aware GPT model, MobilityGPT, with an innovative fine-tuning approach for generating synthetic mobility trajectories.

7.3. Preliminaries

7.3.1. Discretization with Map-Matching. Let $D(V, E)$ represent the road network as a weighted digraph, where the set of nodes V correspond to road intersection, set of edges E to roads, and weights representing link metrics, such as length of the link or traffic volume. There are two sets of trajectories: GPS trajectories and link trajectories.

1) GPS trajectories: A sequence of GPS coordinates with l number of samples $\mathbf{x} \in \mathcal{G} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_l\}$ forms a GPS trajectory that reflects the continuous motion of the object. The set of all GPS trajectories are Ψ where $\mathcal{G} \in \Psi$.

2) Link trajectories: Given the s number of vehicles on the road network, each vehicle travels between origins and destinations using an ordered link path generating a user travel path known as a *micro-graph* $\Phi \in D$. Every link trajectory has n number of links $\phi \in \Phi = \{\phi_1, \phi_2, ..., \phi_n\}$ and $\Phi \subset E$. The set of trajectories is the corpus of all link trajectories with s users $\Phi \in \Lambda = \{\Phi_1, \Phi_2, ..., \Phi_s\}$.

The map-matching process is crucial in refining GPS coordinates, which provide an approximate device location but may not consistently represent the exact travel path due to various errors. Map-matching generates an ordered set of road network links describing the user's trajectory after accounting for the underlying road network $D(V, E)$ and GPS points [**172**]. Addressing location sequences in the continuous domain is challenging in conventional human mobility modeling, often resorting to a discretized grid domain. However, this grid-like representation does not entirely eliminate geospatial mismatching risks. To overcome this, our MobilityGPT model discretizes locations into road segments with map-matching, resulting in more meaningful and realistic generated trajectories.

7.3.2. Problem Definition. In an urban setting, a generative model is tasked with learning the intrinsic mobility characteristics from a provided set of link trajectories, $\Phi \in \Lambda = \{\Phi_1, \Phi_2, ..., \Phi_s\}$. The objective is to generate synthetic trajectories that closely resemble the training trajectories in terms of various utility metrics, thereby capturing and reproducing the key features of mobility present in the input data.

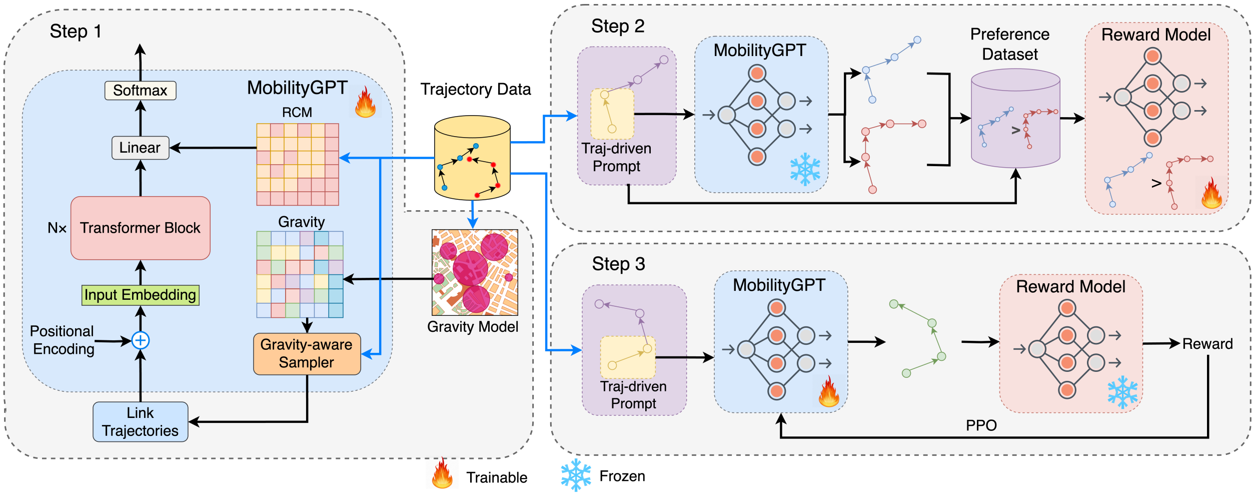


FIGURE 7.2. Overview. The proposed MobilityGPT framework employs Generative Pretrained Transformer (GPT) architectures and self-adaptive reinforcement learning (RL) fine-tuning methods within a human mobility-aware training pipeline for synthetic trajectory generation. The process involves three steps: 1) Pretraining the GPT model with a gravity model and road connectivity matrix, 2) Constructing a trajectory-driven reward model, and 3) Fine-tuning the MobilityGPT pre-trained model using RL policy optimization methods with the trained reward model providing valuable feedback.

7.4. Mobility Modeling with Generative Transformers

This section describes the components of the proposed MobilityGPT for generating synthetic high-fidelity trajectories. MobilityGPT achieves this through a multi-objective training pipeline. Figure 7.2 illustrates the framework of our MobilityGPT mechanism, which can be divided into two parts: pretraining for learning sequence characteristics and fine-tuning for enhancing the quality of generated trajectories. The pretraining stage (Step 1 in Figure 7.2) learns the mobility modeling on the road-link sequences using the gravity of training data and RCM. The fine-tuning stage (Steps 2 and 3 in Figure 7.2) aims to improve the similarity of synthetic trajectories to the training trajectory samples in terms of travel length.

7.4.1. Generative Transformers. Recent breakthroughs in NLP have proven that transformer architectures are remarkably effective in processing word sequences. We have found that trajectory sequences, representing the spatiotemporal movements of individuals or objects, share similarities with sentences across four key dimensions: sequential dependencies, spatial relationships, contextual embeddings, and variable-length sequences. First, trajectory sequences exhibit sequential dependencies similar to sentences, where the order of locations matters. Additionally, trajectories involve spatial relationships between locations, just as words in a sentence convey semantic relationships. Furthermore, trajectory sequences, like sentences, can have variable lengths. Finally, trajectories benefit from contextual embeddings that consider the entire sequence, similar to the contextual understanding of words in sentences.

Given these strong similarities with trajectories and word sequences, two key elements of transformers, self-attention mechanism, and autoregressive generation, empower MobilityGPT to handle sequential data efficiently. Self-attention mechanism captures the long-range dependencies and weighs the importance of each location in the context of the entire trajectory, enabling the modeling of sequential dependencies. Besides, autoregressive sequence generation involves predicting one element at a time based on the context of preceding elements, contributing to the model's ability to generate coherent and contextually relevant trajectories.

The self-attention mechanism computes a weighted sum of values V based on attention scores A assigned to each element in the input sequence:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right) \cdot V \tag{7.1}$$

where Q, K, and V represent the query, key, and value matrices, respectively. The softmax operation normalizes the attention scores, and d_k is the dimensionality of the key vectors. The weighted sum is then used as the output of the self-attention mechanism.

In an autoregressive nature of transformers, the probability of predicting the next element y_t in the sequence given the context $Y_{<t}$ is computed using the chain rule of probability:

$$P(y_t|Y_{<t}) = P(y_1) \cdot P(y_2|y_1) \cdot \ldots \cdot P(y_t|y_{t-1}) \tag{7.2}$$

By formulating human mobility trajectories as a sequence generation task with road links from $D(V, E)$, transformer-based models can be applied to generate realistic and contextually relevant synthetic trajectories given the real trajectories. The tokenization of road links, combined with the self-attention mechanism, enables the model to capture spatial dependencies and generate coherent and meaningful sequences of road links. Next, we explain our unique tokenizer modeling.

7.4.2. Tokenizer Modeling. Tokenization, the breakdown of sequences into smaller units or tokens, is a key step in preparing raw data for transformer models. This process enables transformers to efficiently process and understand the hierarchical structure, relationships, and semantic meaning within the data. The influence of tokenization extends to trajectory sequences, where tokens represent locations or spatiotemporal points. Transformers, utilizing attention mechanisms, assign weights to tokens, enabling the model to focus on relevant trajectory segments during predictions. In MobilityGPT, tokens symbolize locations in terms of road links, and the contextual embeddings derived from tokenization contribute to generating embeddings that consider the holistic context of the entire trajectory.

For MobilityGPT, we introduce a tokenization approach to enhance the model's capabilities. Specifically, we incorporate an "end of trajectory" token, denoted as `<EOT>`, which is appended to each trajectory during training. Given a trajectory sequence $\phi \in \Phi = \{\phi_1, \phi_2, ..., \phi_n\}$ in terms of road links, where ϕ_i denotes the i-th token in the trajectory. The tokenization process is defined as follows:

$$\mathbf{\Phi}_{\text{tokenized}} = \{\phi_1, \phi_2, ..., \phi_n, \texttt{<EOT>}\} \tag{7.3}$$

This inclusion of `<EOT>` serves a dual purpose. First, it acts as a sentinel token to signify the end of a trajectory. This is expressed as if $\phi_i = \texttt{<EOT>}$, then trajectory $i + 1$ begins. The model leverages this information to capture the distinct OD points of each trajectory effectively. Second, this tokenization allows for generating diverse and randomized trajectory sequences.

Utilizing this tokenization strategy is integral to our model's ability to capture spatial patterns, differentiate between trajectories, and generate realistic sequences with varied ODs. This formulation contributes to the robustness and versatility of our trajectory generation approach.

7.4.3. Map Understanding with Gravity Model and Road Connectivity Matrix. Transformer-based models excel in capturing sequential information within long data sequences. However, human mobility modeling presents distinct challenges, necessitating a comprehensive understanding of geographical attributes and constraints. For instance, the sequence of links does not include some hidden attributes of human mobility, such as regional mobility flows (traveling from eastbound of city to west), traffic rules (must right-turn, no U-turn), and trip length. We have developed two innovative methods to overcome these challenges: the Gravity Model and the Road Connectivity Matrix (RCM). The Gravity Model enhances the transformer's capability to understand the mobility flow patterns between regions, providing a context of spatial importance. Together, RCM allows the transformer to learn the complex structure of the road network $D(V, E)$. These strategies collectively train the transformer with an understanding of geospatial information without needing auxiliary data.

1) Gravity model. We first discretize the geospatial area into a grid of different regions, denoted as $r \in \mathcal{R}$. We calculate the weight of each region, $\text{RegionW}(r_x)$, as the number of trajectories that either start or end in that particular region r_x. Next, we calculate the gravity value to quantify the traffic flow between regions for each trajectory that travels between origin and destination regions denoted as r_x and r_y. The Gravity model is defined as:

$$\text{Gravity}(r_x, r_y) = \frac{\text{RegionW}(r_x) \times \text{RegionW}(r_y)}{d^2(r_x, r_y)} \tag{7.4}$$

Here, $d^2(r_x, r_y)$ represents the square of the euclidian distance between regions r_x and r_y. This formulation is based on the principle that the interaction between two regions is directly proportional to the product of their respective weights and inversely proportional to the square of the distance between them. During training, as shown in Figure 7.2, a gravity-aware sampler selects a trajectory sequence $\Phi = \{\phi_1, \phi_2, ..., \phi_n\}$ based on the weight $\text{Gravity}(r_x, r_y)$, where $\phi_1 \in r_x$ and $\phi_n \in r_y$.

2) Road connectivity matrix. The clarity of road connectivity is often diminished when using GPS trajectories compared to traditional 2D maps. This is because, in some instances, roads that appear connected on a map might not be so in reality due to physical barriers or design differences, such as one-way streets and pedestrian zones. To mitigate this issue, we propose a method to precisely extract road connectivities from real-world data and incorporate this information into the training process of MobilityGPT. To achieve this, we define the Road Connectivity Matrix

(RCM) based on the links from link trajectories, denoted as Φ:

$$RC_{\phi_x,\phi_y} = \begin{cases} 1 & \text{if } (\phi_x, \phi_y) \text{ is a consecutive pair in } \Phi, \\ & \quad \text{or} \\ & \text{if } \phi_x \text{ or } \phi_y \text{ is in the last row/column,} \\ 0 & \text{otherwise.} \end{cases} \tag{7.5}$$

RCM effectively represents the connectivity between different road links. If two links, ϕ_x and ϕ_y, are directly connected in the real world, their corresponding element in RCM is set to 1; otherwise, it is set to 0. Additionally, one row and one column, both filled with 1s, are added to the matrix to facilitate the generation of `<EOT>`. Specifically, the last linear layer of the transformer, denoted as F, is adjusted by the RCM prior to the softmax activation function:

$$F' = F \cdot RC \tag{7.6}$$

$$\text{Softmax}(F') = \frac{e^{F'}}{\sum e^{F'}} \tag{7.7}$$

this integration is visualized in Figure 7.2, showing how the RCM modifies the output of the transformer's last linear layer. By multiplying F with the RCM, we ensure that only the connections between actually connected roads are considered. This process filters out any inaccuracies in road connectivity that might have been introduced by GPS data, thus enabling the model to have a more accurate understanding of the actual road network.

7.4.4. Reinforcement learning from trajectory feedback (RLTF). Fine-tuning is essential for sequence generative models [**228**]. Pre-trained transformer architectures provide a foundation, but fine-tuning customizes models to specific objectives, such as similarity of trajectories, enabling them to learn complex patterns and nuances. This process is particularly vital in applications like human mobility, where capturing intricate movement patterns is crucial. Preference datasets are crucial for fine-tuning, allowing domain knowledge and critique to be integrated into training. Building on this, we propose a novel approach to construct a trajectory-aware preference dataset that combines established fine-tuning strategies with a new preference feedback dataset - all without

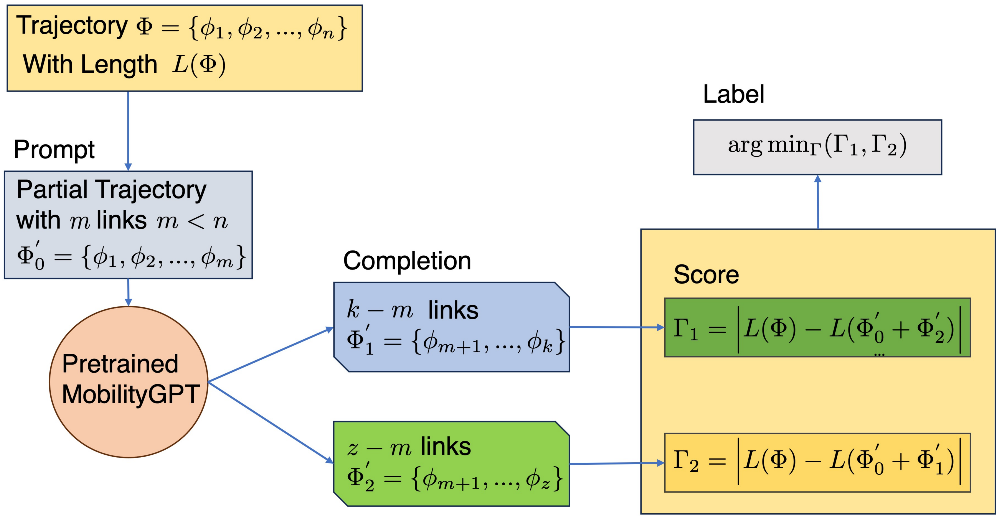

Trajectory $\Phi = \{\phi_1, \phi_2, ..., \phi_n\}$
With Length $L(\Phi)$
Prompt
Partial Trajectory
with m links $m < n$
$\Phi_0^{'} = \{\phi_1, \phi_2, ..., \phi_m\}$
Pretrained
MobilityGPT
Completion
$k - m$ links
$\Phi_1^{'} = \{\phi_{m+1}, ..., \phi_k\}$
$z - m$ links
$\Phi_2^{'} = \{\phi_{m+1}, ..., \phi_z\}$
Score
$\Gamma_1 = \left| L(\Phi) - L(\Phi_0^{'} + \Phi_2^{'}) \right|$
$\Gamma_2 = \left| L(\Phi) - L(\Phi_0^{'} + \Phi_1^{'}) \right|$
Label
$\arg\min_\Gamma(\Gamma_1, \Gamma_2)$

FIGURE 7.3. The scheme of fine-tuning dataset generation for Reinforcement Learning from Trajectory Feedback

requiring human labeling. This enables the model to self-improve based on its output feedback, minimizing human intervention and addressing scalability and bias challenges [**159**].

Reinforcement learning has proven to be a powerful approach for capturing the long-term dependencies and spatiotemporal characteristics inherent in human mobility modeling, owing to the complexity of this task [**180**]. The proposed approach, incorporating our novel preference dataset with reinforcement learning, termed Reinforcement Learning from Trajectory Feedback (RLTF), in an automated feedback process, involves several key steps. First, a reward dataset is created by pairing prompts (partial trajectories) with compilations (complete trajectories generated by the model) and assigning scores based on the similarity of the compilation lengths to the original trajectory lengths (see Figure 7.3). This reward dataset is then utilized to train a reward model **U** through supervised learning. Subsequently, the trained reward model **U** is employed to fine-tune the MobilityGPT model through a Proximal Policy Optimization (PPO) policy π, effectively leveraging reinforcement learning without requiring human feedback.

Preference dataset construction for RLTF. Instead of relying on human feedback in fine-tuning of MobilityGPT, we propose using quantitative metrics to construct the preference dataset. Given a partial trajectory prompt, we generate two complete trajectories and assess their similarity to the actual trajectory by assigning a trajectory similarity score based on the relative lengths of the generated and reference trajectories from the training set. This score measures how well the generated trajectories match the characteristics, particularly length, of the training data, serving as a proxy for human preference without requiring human labeling.

The process, as described in Figure 7.3, begins with a full trajectory $\Phi = \phi_1, \phi_2, \ldots, \phi_n$ of length $L(\Phi)$. The first m road-links $\Phi'_0 = \phi_1, \phi_2, \ldots, \phi_m$ serve as a prompt for MobilityGPT to predict two possible continuations: $\Phi'_1 = \phi_{m+1}, \ldots, \phi_k$ and $\Phi'_2 = \phi_{m+1}, \ldots, \phi_z$. These are concatenated with Φ'_0 to form two complete trajectories, evaluated using $\Gamma_1 = |L(\Phi) - L(\Phi'_0 + \Phi'_2)|$ and $\Gamma_2 = |L(\Phi) - L(\Phi'0 + \Phi'1)|$. The trajectory with lower Γ is labeled Φ_{chosen}, and the other Φ_{rejected}, forming a preference dataset for RLTF. By leveraging this proximity-based metric, we eliminate the need for human labeling, streamlining the dataset creation process and potentially mitigating biases introduced by human subjectivity.

Reward model training. A Reward (preference) Model $\mathbf{U}$ is trained using the proposed preference dataset to predict the probability that a given trajectory is chosen over a rejected alternative. The loss function is formulated as follows:

$$-\log(\sigma(\mathbf{U}(\Phi_{\text{chosen}}) - \mathbf{U}(\Phi_{\text{rejected}}))) \tag{7.8}$$

The goal of the loss function is to maximize the probability that the chosen trajectory has a higher score, thereby encouraging the model to assign a larger difference between Φ_{chosen} and Φ_{rejected}. As the training progresses, the model becomes more capable of recognizing the characteristics of trajectories that align with the preferred trip length.

Fine-tuning with reinforcement learning. Given the pre-trained MobilityGPT model and supervised trained reward model, we train an RL policy that optimizes the MobilityGPT model for generating higher-quality synthetic trajectories. Since policy learning relies on the reward that policy receives from the environment, the reward model provides feedback for RL policy optimization. We followed a general policy optimization method as in [**237**]. The RL agent learns a fine-tuning policy π, which is optimized with the well-known Proximal Policy Optimization (PPO) method [**185**]. The reward of policy π consists of two terms: reward model logits given the prompt Φ'_p and compilation Φ'_c pair received from the MobilityGPT as $\mathbf{U}(\Phi'_p, \Phi'_c)$ and policy shift constraint in terms of the KL divergence given the baseline policy π. The full reward can be represented as follows:

$$\mathrm{R}(\Phi'_p, \Phi'_c) = \mathbf{U}(\Phi'_p, \Phi'_c) - \beta \log\left(\frac{\pi^{\mathrm{PPO}}_{\theta'}(\Phi'_c|\Phi'_p)}{\pi^{\mathrm{base}}(\Phi'_c|\Phi'_p)}\right). \tag{7.9}$$

It is worth emphasizing that the neural network architecture of the reward model $\mathbf{U}$ can be simplified to enhance efficiency and interpretability. Moreover, there is the option to incorporate an online fine-tuning stage, where the policy π and the reward model $\mathbf{U}$ are periodically re-trained. However, in our experimental assessments, we did not delve into the performance of the retraining process. This aspect and the formal exploration of privacy-preserving ML models are left for future research endeavors.

Table 7.1. Comparison of the utility metrics with benchmark studies. Bold and underlined results show the best and second-best results, respectively. For all metrics, lower values indicate better performance. While query error relies on normalized error rates for most visited places, other metrics use the Jensen-Shannon distributional divergence statistic.

	Porto-Taxi					BJ-Taxi				
Methods	Query E.	OD	Trip L.	Radius	Gravity	Query E.	OD	Trip L.	Radius	Gravity
SeqGAN	**0.138**	0.116	0.220	0.191	0.326	**0.063**	0.296	0.162	0.275	0.120
LSTM-TrajGAN	0.234	0.254	0.435	0.425	0.277	0.078	0.182	0.398	0.365	0.219
TS-TrajGen	0.309	0.120	0.161	0.115	0.248	0.277	0.105	0.570	0.485	0.336
DiffTraj	0.267	0.177	0.126	0.123	0.252	0.074	0.146	0.158	0.123	0.114
MobilityGPT	0.144	**0.114**	**0.124**	**0.107**	**0.225**	0.066	**0.099**	**0.123**	**0.102**	**0.105**

7.5. Experiments

MobilityGPT's effectiveness is validated through extensive experiments on real-world datasets against recent benchmarks. We will release our codes and data upon acceptance.

7.5.1. Experimental setup. 1) Dataset description. We have used two datasets of GPS trajectories from taxi drivers in Porto, Portugal, and Beijing, China. The Porto dataset has a sampling period of 15 seconds, whereas the Beijing dataset has a sampling period of 1 minute. Both datasets have been preprocessed and map-matched by [**106**] to remove any errors. The Porto dataset contains $695,085$ unique trips on $11,095$ road links, and the Beijing dataset contains $956,070$ trajectories on $40,306$ road links.

7.5.2. Benchmark models. We performed experiments on original implementation hyperparameters for all benchmark models using their publicly available source codes. We experimented with similar structured generative models with our proposed MobilityGPT model, including discrete and continuous structure formats.

- **SeqGAN** [**225**]: SeqGAN is a state-of-the-art sequential generative model that trains trajectories in discrete sequences with RL policy gradient methods using Monte Carlo sampling.
- **LSTM-TrajGAN** [**179**]: This is a tabular LSTM model by learning the normalized GPS location divergences with respect to the center of the dataset.
- **TS-TrajGen** [**106**]: TS-TrajGen is a method based on GANs that incorporates a modified $A*$ path search algorithm. This adjustment ensures spatial continuity in the generated data. Additionally, the method includes topological constraints to align the generated trajectories with existing road networks, ensuring road matching. TS-TrajGen requires fixed OD pairs as inputs. We use OD pairs from the real training set as inputs during inference.
- **DiffTraj** [**236**]: is the most recent and promising generative model for mobility trajectories. DiffTraj employs a diffusion model in the U-Net structure for generating GPS trajectories.

7.5.3. Evalaution metrics. To evaluate the performance of the generative models, we need a proper comparison metric with a wide range of mobility characteristics that quantifies the quality of generated trajectories given the real trajectories. In this project, we employed four well-known comparison metrics from literature [**80, 86**] in addition to our proposed gravity and connectivity utility metrics. These metrics have been widely used for the utility of synthetic mobility data generation models [**90, 236**].

Query Error: The query error, mainly used for evaluating data synthesis algorithms, is a popular metric for synthetic data generation models. We use spatial counting queries in the form of "the number of trajectories passing through a certain road-link". Given road network $D(V, E)$, we uniformly sampled 500 road links from the road network. Then, the normalized absolute difference between the number of real and synthetic trajectories passing through each of these links is computed by the following:

$$\text{QE}(f(\Sigma)) = \frac{|f(\Psi) - f(\Sigma)|}{\max\{f(\Psi), s\}}, \tag{7.10}$$

where $f(\Psi)$ and $f(\Sigma)$ stand for the query outcome from the original and synthetic trajectories, respectively, and s is sanity bound for mitigating the effect of the extremely small selective queries. We specified the sanity bound s as 1% of the users.

Apart from the query error, the main utility metrics inspect the distributional similarities for synthetic trajectories with respect to real trajectories that train the generative model. Jensen-Shannon divergence (JSD) is a well-known similarity metric mainly used for measuring the similarity of two probability distributions [**131**]. We employ JSD for several human mobility characteristics.

We can look at overall distributions of the generated trajectories Λ_{gen} and real trajectories Λ_{real}, and JSD can provide a summary statistic for a pair of distributions. These are the main distributions we extract from the Λ_{gen} and Λ_{real}:

- **OD**: This metric evaluates the origin-destination similarity between two datasets for the overall characteristics preserved in terms of OD links.
- **Trip Length**: This metric considers the travel distances of trajectories by using the total length of link trajectories.
- **Radius**: In human mobility patterns, the spatial range of daily movements is an important metric. We examined the user's radius of gyration within the controlled road network area.
- **Gravity**: The gravity summary metric examines the impact of the gravity sampling model that was introduced for MobilityGPT at high-level mobility patterns with the same JSD method.

Connectivity: The connectivity metric quantifies the percentage of trajectories that are fully connected from origin to destination, owing to the RCM that samples the next road link conditioned on the connected road links, thereby capturing the underlying road network topology and connectivity patterns.

7.5.4. Model Details. MobilityGPT utilizes a minimal version of the standard decoder-only transformer architecture [**175**], built upon the minimal GPT implementation from [**110**]. Table 7.2

Table 7.2. Model and training parameters utilized in MobilityGPT

Parameter	Value
N. layers	6
N. heads	4
N embeddings	64
Batch size	64
Learning rate	1e-5
Training size	80%
Test size	20%
Training steps	3000
Block size (Porto)	300
Block size (Beijing)	60
Max. Traj. length (Porto)	278
Max. Traj. length (Beijing)	60

summarizes the important model and training parameters employed in this work. Since the Porto and Beijing datasets exhibit different trajectory lengths and characteristics, we utilize distinct block sizes for training and maximum trajectory lengths for synthetic trajectory generation. However, we employ the same set of hyperparameters across both datasets for all training stages, including the reward model and PPO training.

7.5.5. Comparison with benchmark generative methods. Table 7.1 summarizes the performance of MobilityGPT and benchmark models. We sampled 5000 random trajectories from the test set and generated same number of synthetic trajectories. The results show the significant advantage of MobilityGPT over existing state-of-the-art models for trajectory generation. Across both Porto-Taxi and BJ-Taxi, MobilityGPT consistently outperforms all baseline models, achieving the best or second-best performance on multiple utility metrics.

Notably, MobilityGPT demonstrates up to 24.07% improvement over the second-best model in modeling trip lengths and up to 17.07% improvement in capturing spatial radius on BJ-Taxi. Additionally, it exhibits up to 9.27% improvement in capturing gravity patterns on Porto-Taxi. MobilityGPT excels in accurately modeling intricate characteristics of individual trajectories, such as spatial and temporal patterns, by effectively capturing the relationships between locations and trajectories, addressing a significant limitation of previous approaches. Its remarkable performance in modeling OD distributions highlights its capability to capture underlying patterns governing traffic flow and human mobility. Overall, MobilityGPT's superior performance across multiple metrics highlights its effectiveness as a comprehensive and powerful trajectory generation model, positioning it as a valuable tool for applications in transportation, urban planning, and mobility analysis.

The primary objective of mobility generative models is to produce trajectories that closely resemble the original data by capturing the overall mobility characteristics, rather than generating exact replicas. Figure 7.4 depicts the spatial densities generated by baseline methods for Porto City. While all the maps demonstrate that the generated trajectories accurately capture the city's geographic profiles, MobilityGPT has a clear representation of geospatial density in the generated trajectories compared to the original test trajectories. Clearly, MobilityGPT produces trajectories that closely resemble the originals both in urban areas and the outskirts of the city, surpassing other methods significantly.

7.5.6. Ablation study. As MobilityGPT incorporates three distinct contributions to the standard GPT model for comprehensively capturing human mobility patterns – the gravity model, RCM, and RL fine-tuning (RLTF) – we performed experiments using different variants of these models, evaluated using three comparison metrics as shown in Table 7.3a. On average, with the additional components of MobilityGPT, we observe a 7.4% improvement in trajectory similarities in terms of ODs and a 19.5% improvement over trip length similarities. Furthermore, without the RCM, MobilityGPT fails to preserve connectivity in the generated trajectories. RCM ensures connectivity by conditioning the generation of the next link on connected links, effectively capturing the spatial constraints inherent in human mobility patterns.

We compare different fine-tuning methods with our proposed preference dataset construction approach: supervised fine-tuning (SFT) and direct preference optimization (DPO) [**177**] and summarized their performance in Table 7.3b. We found that RLTF is effective for capturing the complexity of mobility modeling and outperforms other fine-tuning strategies.

An additional ablation study on the Beijing dataset, shown in Table 7.4, investigated the impact of MobilityGPT's key components. Similar to the Porto dataset, the ablation study on the Beijing dataset exhibits the synergistic effects of MobilityGPT's novel components – RCM, Gravity Model, and RLTF fine-tuning – in accurately modeling diverse mobility characteristics, boosting

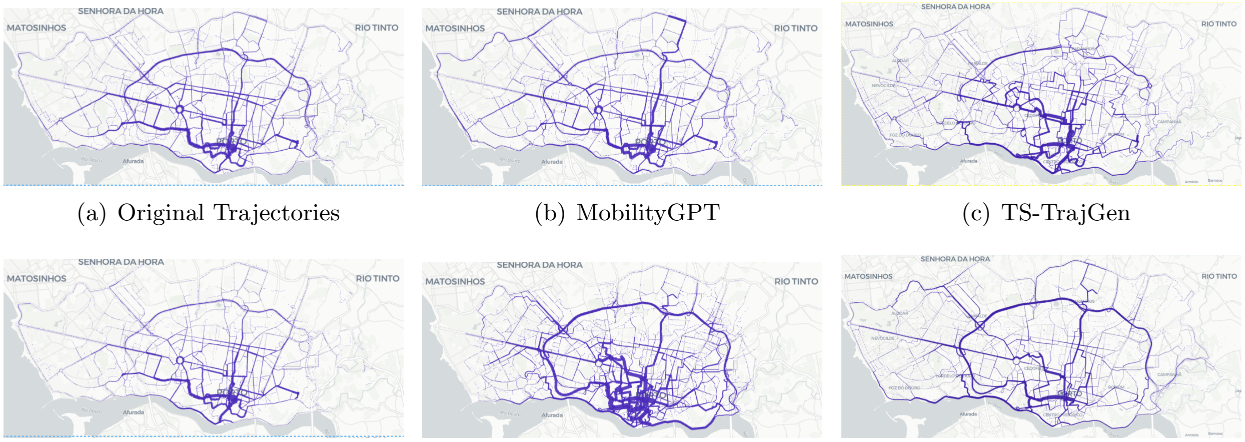
Table 7.3. Ablation study shows the influence of (a) removing each component on the performance of MobilityGPT and (b) on different methods for fine-tuning.

RCM	Gravity Model	RLTF	OD	Trip L.	Conn.
✗	✗	✗	0.123	0.154	0.87
✗	✓	✗	0.122	0.146	0.83
✓	✗	✗	0.120	0.147	1.0
✓	✓	✗	0.114	0.140	1.0
✓	✓	✓	**0.114**	**0.124**	**1.0**

(a)

SFT	DPO	RLTF	OD	Trip Length
✓			0.116	0.171
	✓		0.115	0.163
		✓	**0.114**	**0.124**

(b)



(a) Original Trajectories

(b) MobilityGPT

(c) TS-TrajGen

(d) SeqGAN

(e) LSTM-TrajGAN

(f) DiffTraj

Figure 7.4. Visual representation of the geospatial density of different methods. MobilityGPT exhibits the highest similarity to the original trajectories.

performance on key metrics like OD and Trip Length while the connectivity priors ensure accurate capture of network-level patterns.

Table 7.4. Ablation on Beijing dataset study for different components of MobilityGPT.

RCM	Gravity Model	RLTF	OD	Trip Length	Connectivity
✗	✗	✗	0.102	0.201	0.330
✗	✓	✗	0.103	0.151	0.367
✓	✗	✗	**0.093**	0.145	1.0
✓	✓	✗	0.099	0.131	1.0
✓	✓	✓	0.099	**0.123**	**1.0**

7.5.7. Effect of temperature on MobilityGPT inferencing. The temperature parameter in LLMs, including MobilityGPT, influences the randomness in generating outputs. At lower temperatures, the model behavior is more deterministic, often selecting the most likely next token. As the temperature increases, the model introduces greater randomness, leading to a diverse array of outputs.

Interestingly, as illustrated in Figure 7.5, MobilityGPT demonstrates a clear trend: its performance tends to increase with higher temperature settings. This observation suggests that mobility data generation can benefit from the randomness of GPT-like models.

7.5.8. Tokenizer Modeling. In the development of the MobilityGPT tokenizer model, we explore the use of specific tokens to mark the start (`<BOT>`) and end (`<EOT>`) of trajectories. Our aim is to assess how these tokens affect the model's performance in human mobility modeling, particularly for sequential trajectory data. Our findings reveal a key insight: using the `<EOT>` token alone significantly improves performance. This result indicates that emphasizing the end of a trajectory is more beneficial than marking its beginning with a separate token for our mobility modeling task. The effectiveness of the `<EOT>` token emphasizes the importance of nuanced data representation and the need for task-specific tokenization strategies in sequential data generation with transformer models. An ablation study, detailed in Table 7.5, compares single and dual-token approaches using various metrics. This study shows that using a single token type outperforms the dual-token method in the MobilityGPT workflow, highlighting the advantages of customized tokenization.

7.5.9. Computational Demand. Evaluating computational demand is essential for ensuring the practical feasibility, scalability, and environmental sustainability of generative models. To facilitate a fair comparison, we trained all benchmark models and MobilityGPT on the same computational resources, utilizing an NVIDIA TITAN RTX GPU with 24GB memory, and the

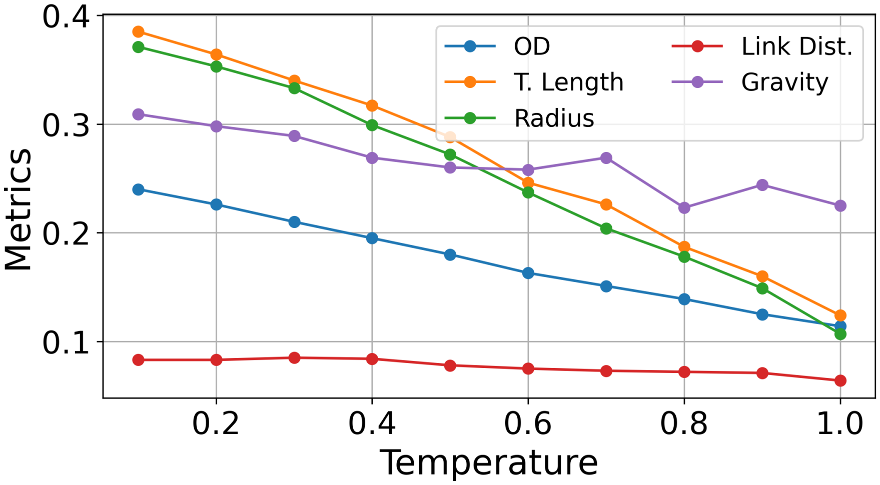


Figure 7.5. Performance of MobilityGPT under different temperature settings.

Table 7.5. Ablation study on the effectness of <EOT>.

<BOT>	<EOT>	OD	Trip Length	Radius	Gravity
✓	✓	**0.110**	0.151	0.128	0.235
✗	✓	0.114	**0.124**	**0.107**	**0.225**

results are depicted in Table 7.6. MobilityGPT balances model size and efficiency: 5.5M params, lower than TS-TrajGen (23.2M & 7.6M) but higher than LSTM-TrajGAN (100k). Its ∼5hr runtime outperforms TS-TrajGen ($> 200hr$) and DiffTraj ($48 - 60hr$) by an order of magnitude, despite LSTM-TrajGAN's $30min - 1hr$ efficiency at the cost of poor predictive performance.

Table 7.6. Computational performance of different models vs MobilityGPT

	N_param		Total running time	
Model	BJ	Porto	BJ	Porto
SeqGAN	2.6M	800	∼ 12hr	∼ 5hr
LSTM-TrajGAN	100k	100k	∼ 30mins	∼ 1hr
TS-TrajGen	23.2M	7.6M	> 200hr	> 200hr
DiffTraj	15.7M	15.7M	∼ 48hr	∼ 60hr
MobilityGPT	5.5M	5.5M	∼ 5hr	∼ 5hr

MobilityGPT achieves a compelling balance between model capacity and scalability through its moderate size and computational efficiency. Despite having fewer parameters than some benchmarks, MobilityGPT achieves state-of-the-art performance across datasets, requiring substantially less time and resources. Unlike LSTM-TrajGAN, which trades off predictive accuracy for efficiency, MobilityGPT demonstrates the feasibility of combining high performance and practical efficiency within a single model. This makes MobilityGPT a promising choice for practical deployments where both predictive power and computational feasibility are essential.

7.6. Conclusion

This paper presents a GPT-based generative method for modeling human mobility characteristics. The proposed mechanism considers a multi-objective learning method with pre-training and fine-tuning stages. MobilityGPT pre-training leverages the understanding of human mobility using the string self-attention structures by conditioning the trajectory sampling with gravity model and sequence sampling with road connectivity matrix. To further enhance the MobilityGPT capabilities for generating similar trajectories to real trajectories, we propose a fine-tuning strategy that does not require a human assessment. The novel fine-tuning method trains a reward using the trajectory trip length and employs the reward model as an import to the RL policy gradient method. The experiments validate the effectiveness of the proposed MobilityGPT approach for generating realistic mobility trajectories from real trajectory samples by comparing with four state-of-the-art models.

Limitation: MobilityGPT achieves state-of-the-art results while handling road network constraints, but lacks formal privacy guarantees for synthetic trajectories. Future directions involve improving generalizability and incorporating practical privacy assurances.

CHAPTER 8

Conclusion and Future Work

In this thesis, we study a complete solution for an AI-based ITS controller, TSC. For the the performance of RL-TSCs, we proposed a constrained optimization-based RL-TSC model constraining on maximum green time for fairness and maximum fuel consumption for air quality. Regarding security, adversarial attacks with a detection mechanism on RL-TSCs are presented as an example controller mechanism. In addition to that, the thesis introduces differential privacy-based data sensitization algorithms for individual trajectories and aggregated mobility datasets. This chapter presents our ongoing and future work in two sections. First, since we discussed the security of DRL-TSCs with literature review in Chapter 4, here, we only discuss the future research directions. Next, we introduce a synthetic trajectory generation model for mobility datasets using generative machine learning methods, including a detailed literature review and future research directions.

8.1. Multi-objective Constrained Reinforcement Learning for TSCs

TSCs have an essential role in managing the traffic flow in urban areas. However, rule-based existing TSCs cannot respond well to heavy and dynamic traffic conditions. There have been several attempts to optimize the TSCs with different methods [**214**]. We have shown in Chapter 2 and Chapter 3 that multi-objective RL with constraints has promising performance on city-level multi-intersection TSC scenarios in terms of lowering traffic congestion and fuel consumption. In Chapter 2, we investigate the effectiveness of learning-based TSCs in reducing fuel consumption and emissions compared to conventional TSCs while also being fair for different traffic directions. The findings highlight a correlation between emissions, fuel consumption, and waiting times. While learning-based TSCs perform well in some scenarios, they may be outperformed by other controllers, such as the max pressure traffic controller. We then, in Chapter 3, introduce a novel approach using a constrained multi-objective RL model, aiming to improve fairness and air quality in traffic scheduling. Experimental results show reduced travel times, lower CO2 emissions, and fairer traffic scheduling with the proposed model compared to state-of-the-art TSCs. While TSCs can improve traffic efficiency with RL, the performance of joint control of the learning-based TSCs and connected automated vehicle (CAV) is an open research direction.

8.1.1. Future Work.

The planning and control of CAVs have demonstrated promise in various scenarios [**103**], including CAV platooning, mixed traffic scenarios involving CAVs and human-driven vehicles (HDVs), and CAVs interacting with traffic infrastructures like on-ramps [**135**] and traffic intersections [**193**]. To manage these components—CAVs and TSCs—in extensive traffic settings, several approaches can be employed, including centralized, decentralized, or mixed-control strategies, each offering its unique advantages and limitations.

Since CAVs and TSCs are structured independently, it is challenging to control them jointly in a stochastic environment. Using only mathematical optimization methods leads to higher computational inefficiency, centralized control, or lower performance, decentralized control where the computational costs are minimized. Focusing on a mixed hierarchical control approach with predictive control strategy for CAVs and TSCs, utilizing a reinforcement learning-based policy for TSCs can be a promising research direction. To bridge the gap between CAV control and learning-based TSCs, we will study eco-driving strategies with joint control of CAVs and RL-TSCs. A model-based RL model with a predictive horizon can be integrated with classic optimization-based control methods that cooperatively control CAVs and TSCs. Different traffic scenratios including single intersection and multi-intersection traffic environments with multi-agent RL models, can be a promising research direction.

8.2. Adversarial Attacks and Detection models on RL-TSC

The DRL-TSC agent expects to receive data from the same distribution with it is trained. However, the input data to the DRL agent might be from an unknown distribution with malicious intention or without malicious intent, such as measurement noise. While unintentional noise can still harm the DRL agent, carefully crafted intentional out-of-distribution samples affect the performance of DRL vastly. The authors in [**140**] observe similar conclusions for a CartPole game controlled with DRL as we have proved in Chapter 3.

Adversarial samples generally target the DRL in the test phase from the DNNs representing the DRL agent's policy. Most of the adversarial attacks on DNNs apply to the DRL-TSCs, but the success rate of attacks differs by the attack model, and it requires experimental evaluation. There are several challenges of detecting adversarial examples in DNN-based learning models. First, since adversarial samples are generated by some optimization models instead of random perturbation, it is hard to define a proper boundary between adversarial samples and clean samples. Second, the anomalies depend on the data type; therefore, the success rate of detection models is highly correlated with the data type [**26**].

There are different defense techniques for DNNs when adversarial attacks target the DRL to mislead the DRL agent to improper actions. The detectors in the literature can be classified in different groups as supervised and unsupervised. The supervised detectors consider specific attack models. On the other hand, unsupervised, more generalized, defense mechanisms are not limited by specific attack strategies. In this research, we are working on more realistic defense models that can detect unknown attacks in the implementation (test) phase of DRL.

8.2.1. Future Work. We presented an overview of the defense models of DNNs in Chapter 4 with a proposed detection mechanism using a sequential statistical detector. As a result, we can further improve the performance of anomaly detectors. This section discusses several future research directions with several baseline models in the literature.

Defense models against adversarial attacks in the implementation (test) phase can be achieved in two approaches:

- Attach an additional detection module to the learning agent, thereby keep the learning model unchanged.
- Design a defense mechanism on the learning model by either estimating the parameter changes on the learning model itself or modifying it for robust classification.

8.2.1.1. *External Adversarial Input Detection.* External adversarial input detection models require an additional detection module that checks the input data before it is fed into the DNNs. Two leading works on this group of defense models are presented in [**146**], which identifies the adversarial samples using PCA, and [**79**], compares the distribution of samples with training set to identify the adversaries.

Our preliminary results indicate that external detectors such as sequential models are powerful tools for detecting adversarial attacks on DRL-TSCs by detecting an average of 98% adversaries with minimal detection delay.

DRL-TSC controller is a real-time system where a detector needs to detect adversaries timely to prevent the impact of such attacks on traffic. The state information collected from the environment is a time sequence traffic condition. Recurrent neural networks (RNN) capture the dynamics of time sequences. Hence, detecting adversarial state sequences with RNNs would give convincing results. We will explore RNN-based sequential models for detecting DRL adversarial attacks on TSCs.

8.2.1.2. *Internal Defense.* Building a defense model using the intrinsic properties of the learning model has two research directions: (i) detects and rejects the adversarial samples [**22**], (ii) provides a robust classifier [**164**, **205**]. While most mechanisms try to identify adversarial samples and mitigate the effect of adversarial samples, in the second approach, the defense mechanism does not explicitly identify the adversaries; instead, the learning model selects correct action even when the input is perturbed. In this research, our focus is on designing a detection module using the DRL features.

Multiple research papers detect the adversarial samples by observing the DNN weights or activation functions, especially on image datasets [**72**, **142**, **147**, **233**]. Recently a framework for detecting adversarial samples on DNN classifiers is presented in [**178**] where authors focus on test-time detection of anomalous inputs to a DNN classifier using multiple layer class conditional representations. Another recent study provides a defense mechanism that includes robustness to the DNN classifier by proposed training and adversarial sample detection with K+1 classification for K classes [**188**].

The data structure in DRL-TSC is different, and state samples are timely traffic information that allows DRL-TSC agents to select optimum actions to minimize the traffic delay. As the DRL-TSC agent receives time series state, statistical properties of DNN policy change over time. Although the policy of DRL agents is DNNs, there are still several differences between DRLs and DNNs. First, DRL models have more diverse inputs than DNN-based classifiers. Accordingly, it is hard to find correct boundaries for DRL when using highly dynamic environments such as TSCs. Second, DRL agents generally have a limited number of actions, which are the output of DNNs. However, DNN classifiers, in general, have more output classes. We plan to integrate the DRL properties with DNNs to design statistical anomaly detectors that provide a robust learner for DRL-TSCs. In addition, we will explore whether having a limited number of DNN outputs (actions) provide clear boundaries for inputs.

8.2.1.3. *Adversarial Attacks on Different TSC Configurations.* The performance of DRL models is highly dependent on an accurate and concrete state definition. Therefore, there are many different state representations used for DRL-TSCs. We can classify the state representations in three main groups: raw RGM images, discrete traffic state encoding (DTSE), and feature-based vectors. The previous experiments show that adversarial attacks are highly data-dependent [**26**]. In Chapter 3, we demonstrated 8 attack configurations: FGSM, JSMA attack models, single-DQN vs. MA2C DRL models, black-box vs. white-box attack settings. For in-depth adversarial attack analysis, we plan to launch adversarial attacks for many state representations in different configurations and design proper defense mechanisms for a broader area of adversarial attacks. We also would like to explore such adversarial attacks on larger-scale real road networks like the San Francisco downtown network. Using larger and realistic TSC networks, we will understand if adversarial attacks will cause much more severe network-level impact, e.g., massive congestion.

8.3. Differential Privacy for Mobility Dataset

This project comprises two parts that focus on enhancing privacy in mobility data. Chapter 5 introduces a differentially-private adaptive noise injection model for aggregated trajectory networks, protecting individual origin-destination (OD) locations. It perturbs GPS points using planar Laplace noise, with perturbation distances adjusted based on localized road network density using sparse vector technique. This selection is performed in a private manner using the Adaptive Thresholding method. In Chapter 6, we present a differentially-private map-matching (DPMM) algorithm that protects OD locations and travel paths. Similar to the first part, it employs planar Laplace noise injection, but it also considers the density of localized road networks and the functional class of the road links. The level of perturbation for each GPS point is adjusted based on the localized link density. The DPMM algorithm utilizes a waypoint sampling method for privately constructing travel paths. Both parts offer enhanced privacy for mobility data and outperform other models, particularly in preventing geographical mismatches with road structures.

8.3.1. Future Work. The widespread adoption of location-based services and smart GPS devices (smartphones/watches) make continuous monitoring of human mobility both desirable and feasible. Such mobility data can enable different smart urban planning and CAV driver safety applications. Several mobility data exchange platforms exist, e.g., Open Mobility Foundation[1] or Mobility Dataspace[2]. However, such spatiotemporal traces can reveal private lifestyle patterns (e.g., home/office addresses, points of interest). Removing personal identifiers from the dataset does not adequately provide privacy because attackers can still re-identify users [**50**]. These privacy concerns inhibit free sharing of mobility or CAVs data across multiple entities. In this project, we plan to incorporate differential privacy and generative machine learning (such as GPT) techniques to sanitize raw datasets and generate synthetic mobility data that retain key movement characteristics or driving behavior from raw data. We have previously applied differential privacy methods to preserve the privacy of mobility datasets [**90**, **93**] and proposed synthetic data generation tools with GPTs. We plan to build on our past success to generate synthetic mobility data with formal DP guarantees. Our target mobility data includes individual locations/trajectories and aggregated mobility datasets. The expected outcome is a foundational building block for a data exchange platform to enable privacy-preserving sharing of mobility datasets.

Our proposed project focuses on synthetic mobility data generation, which addresses two challenges: privacy concerns and a lack of publicly available mobility/CAVs data. One way to deal with data privacy is to generate synthetic data that exhibit similar characteristics as real data but is deprived of personally identifiable information. While several privacy-preserving techniques exist for aggregated mobility datasets, producing synthetic mobility datasets at the individual level is still a challenge. As of today, we still do not have well-proven methods for generating realistic mobility trajectories with proven privacy guarantees. Differential privacy (DP) is a statistical privacy-preserving technique [**52**] that is designed to minimize the leakage of information about individuals, while still preserving the characteristic patterns in the data. Differentially private machine learning provides formal privacy guarantee to the ML applications. However, applying existing models to synthetic trajectory generation is not practical due to the unique constraints of mobility datasets. At the individual level, each synthetic trajectory should maintain realistic

[1] https://www.openmobilityfoundation.org/

[2] https://mobility-dataspace.eu/

mobility patterns and follow real-world conditions such as traffic rules (i.e., respect common sense knowledge).

CAV research (especially in automating perception tasks) can face limited progress due to insufficient training data. Since existing publicly available datasets [**36**, **123**, **167**] are not rich enough for large-scale evaluations, researchers resort to simulating realistic human trajectories with generative models. AI/ML-based generative model is a promising approach for generating synthetic datasets, which has been explored for text, image, and video generations. Although there are several attempts at mobility data generation, their capabilities are limited to specific data types.

Bibliography

[1] *Diffprivlib: The IBM Differential Privacy Library.* `https://github.com/IBM/differential-privacy-library`.

[2] *Siddle, j. i know where you were last summer: London's public bike data is telling everyone where you've been.* `https://vartree.blogspot.com/2014/04/i-know-where-you-were-last-summer.html`, 2021 (accessed April 5th, 2021).

[3] U. E. P. Agency, *Fast facts on transportation greenhouse gas emissions*, Fast Facts on Transportation Greenhouse Gas Emissions, (2020).

[4] K. Ahn, H. Rakha, A. Trani, and M. Van Aerde, *Estimating vehicle fuel consumption and emissions based on instantaneous speed and acceleration levels*, Journal of transportation engineering, 128 (2002), pp. 182–190.

[5] S. Alshayeb, A. Stevanovic, and N. Dobrota, *Impact of various operating conditions on simulated emissions-based stop penalty at signalized intersections*, Sustainability, 13 (2021), p. 10037.

[6] S. Alshayeb, A. Stevanovic, N. Mitrovic, and E. Espino, *Traffic signal optimization to improve sustainability: A literature review*, Energies, 15 (2022), p. 8452.

[7] M. E. Andrés, N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi, *Geo-indistinguishability: Differential privacy for location-based systems*, in Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security, 2013, pp. 901–914.

[8] Q. Bai, A. S. Bedi, M. Agarwal, A. Koppel, and V. Aggarwal, *Achieving zero constraint violation for constrained reinforcement learning via primal-dual approach*, in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 36, 2022, pp. 3682–3689.

[9] Q. Bai, A. S. Bedi, and V. Aggarwal, *Achieving zero constraint violation for constrained reinforcement learning via conservative natural policy gradient primal-dual algorithm*, in Proceedings of the AAAI Conference on Artificial Intelligence, 2023.

[10] S. Baluja and I. Fischer, *Learning to attack: Adversarial transformation networks*, in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32, 2018.

[11] B. Banar and S. Colton, *A systematic evaluation of gpt-2-based music generation*, in International Conference on Computational Intelligence in Music, Sound, Art and Design (Part of EvoStar), Springer, 2022, pp. 19–35.

[12] M. Baratchi, N. Meratnia, P. J. Havinga, A. K. Skidmore, and B. A. Toxopeus, *A hierarchical hidden semi-markov model for modegeling mobility data*, in Proceedings of the 2014 ACM international joint conference on pervasive and ubiquitous computing, 2014, pp. 401–412.

[13] H. Barbosa, M. Barthelemy, G. Ghoshal, C. R. James, M. Lenormand, T. Louail, R. Menezes, J. J. Ramasco, F. Simini, and M. Tomasini, *Human mobility: Models and applications*, Physics Reports, 734 (2018), pp. 1–74.

[14] M. Basseville, I. V. Nikiforov, et al., *Detection of abrupt changes: theory and application*, vol. 104, Prentice hall Englewood Cliffs, 1993.

[15] V. Behzadan and W. Hsu, *Adversarial exploitation of policy imitation*, arXiv preprint arXiv:1906.01121, (2019).

[16] V. Behzadan and A. Munir, *Vulnerability of deep reinforcement learning to policy induction attacks*, in International Conference on Machine Learning and Data Mining in Pattern Recognition, Springer, 2017, pp. 262–275.

[17] M. Berber, A. Ustun, and M. Yetkin, *Comparison of accuracy of gps techniques*, Measurement, 45 (2012), pp. 1742–1746.

[18] A. N. Bhagoji, D. Cullina, C. Sitawarin, and P. Mittal, *Enhancing robustness of machine learning systems via data transformations*, in 2018 52nd Annual Conference on Information Sciences and Systems (CISS), IEEE, 2018, pp. 1–5.

[19] A. Y. Bigazzi and M. Rouleau, *Can traffic management strategies improve urban air quality? a review of the evidence*, Journal of Transport & Health, 7 (2017), pp. 111–124.

[20] V. Bindschaedler and R. Shokri, *Synthesizing plausible privacy-preserving location traces*, in 2016 IEEE Symposium on Security and Privacy (SP), IEEE, 2016, pp. 546–563.

[21] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al., *Language models are few-shot learners*, Advances in neural information processing systems, 33 (2020), pp. 1877–1901.

[22] S. Bulusu, B. Kailkhura, B. Li, P. Varshney, and D. Song, *Anomalous instance detection in deep learning: A survey*, tech. rep., Lawrence Livermore National Lab.(LLNL), Livermore, CA (United States), 2020.

[23] E. J. Candès, X. Li, Y. Ma, and J. Wright, *Robust principal component analysis?*, Journal of the ACM (JACM), 58 (2011), pp. 1–37.

[24] C. Cao and M. Li, *Generating mobility trajectories with retained data utility*, in Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, 2021, pp. 2610–2620.

[25] Y. Cao, C. Xiao, B. Cyr, Y. Zhou, W. Park, S. Rampazzi, Q. A. Chen, K. Fu, and Z. M. Mao, *Adversarial sensor attack on lidar-based perception in autonomous driving*, in Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, 2019, pp. 2267–2281.

[26] N. Carlini and D. Wagner, *Adversarial examples are not easily detected: Bypassing ten detection methods*, in Proceedings of the 10th ACM workshop on artificial intelligence and security, 2017, pp. 3–14.

[27] ———, *Towards evaluating the robustness of neural networks*, in 2017 ieee symposium on security and privacy (sp), IEEE, 2017, pp. 39–57.

[28] N. Casas, *Deep deterministic policy gradient for urban traffic light control*, arXiv preprint arXiv:1703.09035, (2017).

[29] S. Chang, C. Li, H. Zhu, T. Lu, and Q. Li, *Revealing privacy vulnerabilities of anonymous trajectories*, IEEE Transactions on Vehicular Technology, 67 (2018), pp. 12061–12071.

[30] S. Changruenngam, D. J. Bicout, and C. Modchang, *How the individual human mobility spatio-temporally shapes the disease transmission dynamics*, Scientific Reports, 10 (2020), p. 11325.

[31] P. Chao, Y. Xu, W. Hua, and X. Zhou, *A survey on map-matching algorithms*, in Australasian Database Conference, Springer, 2020, pp. 121–133.

[32] K. Chatzikokolakis, C. Palamidessi, and M. Stronati, *A predictive differentially-private mechanism for mobility traces*, in International Symposium on Privacy Enhancing Technologies Symposium, Springer, 2014, pp. 21–41.

[33] M. Chen, A. Radford, R. Child, J. Wu, H. Jun, D. Luan, and I. Sutskever, *Generative pretraining from pixels*, in International conference on machine learning, PMLR, 2020, pp. 1691–1703.

[34] R. Chen, B. Fung, and B. C. Desai, *Differentially private trajectory data publication*, arXiv preprint arXiv:1112.2020, (2011).

[35] R. Chen, B. C. Fung, B. C. Desai, and N. M. Sossou, *Differentially private transit data publication: A case study on the montreal transportation system*, in Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2012, pp. 213–221.

[36] T. Chen, M. Kaafar, and R. Boreli, *The where and when of finding new friends: Analysis of a location-based social discovery network*, in Proceedings of the International AAAI Conference on Web and Social Media, vol. 7, 2013.

[37] T. Chen, J. Liu, Y. Xiang, W. Niu, E. Tong, and Z. Han, *Adversarial attack and defense in reinforcement learning-from ai security view*, Cybersecurity, 2 (2019), p. 11.

[38] T. Chen, W. Niu, Y. Xiang, X. Bai, J. Liu, Z. Han, and G. Li, *Gradient band-based adversarial training for generalized attack immunity of a3c path finding*, arXiv preprint arXiv:1807.06752, (2018).

[39] T. Chu, J. Wang, L. Codecà, and Z. Li, *Multi-agent deep reinforcement learning for large-scale traffic signal control*, IEEE Transactions on Intelligent Transportation Systems, (2019).

[40] S.-B. Cools, C. Gershenson, and B. D'Hooghe, *Self-organizing traffic lights: A realistic simulation*, in Advances in applied self-organizing systems, Springer, 2013, pp. 45–55.

[41] Y. Dai, J. Shao, C. Wei, D. Zhang, and H. T. Shen, *Personalized semantic trajectory privacy preservation through trajectory reconstruction*, World Wide Web, 21 (2018), pp. 875–914.

[42] B. De Coensel, A. Can, B. Degraeuwe, I. De Vlieger, and D. Botteldooren, *Effects of traffic signal coordination on noise and air pollutant emissions*, Environmental Modelling & Software, 35 (2012), pp. 74–83.

[43] E. P. de Mattos, A. C. Domingues, and A. A. Loureiro, *Give me two points and i'll tell you who you are*, in 2019 IEEE Intelligent Vehicles Symposium (IV), IEEE, 2019, pp. 1081–1087.

[44] Y.-A. De Montjoye, C. A. Hidalgo, M. Verleysen, and V. D. Blondel, *Unique in the crowd: The privacy bounds of human mobility*, Scientific reports, 3 (2013), pp. 1–5.

[45] D. Desfontaines and B. Pejó, *Sok: Differential privacies*, Proceedings on Privacy Enhancing Technologies, 2020 (2020), pp. 288–313.

[46] E. W. Dijkstra et al., *A note on two problems in connexion with graphs*, Numerische mathematik, 1 (1959), pp. 269–271.

[47] D. Ding, X. Wei, Z. Yang, Z. Wang, and M. Jovanovic, *Provably efficient safe exploration via primal-dual policy optimization*, in International Conference on Artificial Intelligence and Statistics, PMLR, 2021, pp. 3304–3312.

[48] U. DOT, *Intelligent transportation systems-safety solutions: Preventing crashes and saving lives*, Office of The Assistant Secretary for Research And Technology, (2018).

[49] D. H. Douglas and T. K. Peucker, *Algorithms for the reduction of the number of points required to represent a digitized line or its caricature*, Cartographica: the international journal for geographic information and geovisualization, 10 (1973), pp. 112–122.

[50] M. Douriez, H. Doraiswamy, J. Freire, and C. T. Silva, *Anonymizing nyc taxi data: Does it matter?*, in 2016 IEEE international conference on data science and advanced analytics (DSAA), IEEE, 2016, pp. 140–148.

[51] W. Du, J. Ye, J. Gu, J. Li, H. Wei, and G. Wang, *Safelight: A reinforcement learning method toward collision-free traffic signal control*, arXiv preprint arXiv:2211.10871, (2022).

[52] C. Dwork, *Differential Privacy*, in Proceedings of the 33rd International Colloquium on Automata, Languages and Programming, part II (ICALP), vol. 4052 of Lecture Notes in Computer Science, July 2006, pp. 1–12.

[53] ———, *Differential privacy: A survey of results*, in International conference on theory and applications of models of computation, Springer, 2008, pp. 1–19.

[54] E. ElSalamouny and S. Gambs, *Differential privacy models for location-based services*, Transactions on Data Privacy, 9 (2016), pp. 15–48.

[55] F. Z. Errounda and Y. Liu, *An analysis of differential privacy research in location and trajectory data*, Research Square, (2020).

[56] B. Eysenbach and S. Levine, *Maximum entropy rl (provably) solves some robust rl problems*, in International Conference on Learning Representations, 2021.

[57] L. Fan and L. Xiong, *An adaptive approach to real-time aggregate monitoring with differential privacy*, IEEE Transactions on knowledge and data engineering, 26 (2013), pp. 2094–2106.

[58] F. Fei, S. Li, H. Dai, C. Hu, W. Dou, and Q. Ni, *A k-anonymity based schema for location privacy preservation*, IEEE Transactions on Sustainable Computing, 4 (2017), pp. 156–167.

[59] R. Feinman, R. R. Curtin, S. Shintre, and A. B. Gardner, *Detecting adversarial samples from artifacts*, arXiv preprint arXiv:1703.00410, (2017).

[60] J. Feng, Z. Yang, F. Xu, H. Yu, M. Wang, and Y. Li, *Learning to simulate human mobility*, in Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2020, pp. 3426–3433.

[61] R. L. Finn, D. Wright, and M. Friedewald, *Seven types of privacy*, in European Data Protection: Coming of Age, Springer, 2013, pp. 3–32.

[62] M. Fiore, P. Katsikouli, E. Zavou, M. Cunche, F. Fessant, D. Le Hello, U. Aivodji, B. Olivier, T. Quertier, and R. Stanica, *Privacy in trajectory micro-data publishing: a survey*, Transactions on Data Privacy, 13 (2020), pp. 91–149.

[63] V. Gallego, R. Naveiro, and D. R. Insua, *Reinforcement learning under threats*, in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, 2019, pp. 9939–9940.

[64] M. Gastaldi, C. Meneguzzer, R. Rossi, L. Della Lucia, and G. Gecchele, *Evaluation of air pollution impacts of a signal control to roundabout conversion using microsimulation*, Transportation research procedia, 3 (2014), pp. 1031–1040.

[65] A. Gattami, *Reinforcement learning of markov decision processes with peak constraints*, arXiv preprint arXiv:1901.07839, (2019).

[66] A. Gattami, Q. Bai, and V. Aggarwal, *Reinforcement learning for constrained markov decision processes*, in International Conference on Artificial Intelligence and Statistics, PMLR, 2021, pp. 2656–2664.

[67] B. Gedik and L. Liu, *Protecting location privacy with personalized k-anonymity: Architecture and algorithms*, IEEE Transactions on Mobile Computing, 7 (2007), pp. 1–18.

[68] W. Genders and S. Razavi, *Asynchronous n-step q-learning adaptive traffic signal control*, Journal of Intelligent Transportation Systems, 23 (2019), pp. 319–331.

[69] B. Ghazi, N. Kamal, R. Kumar, P. Manurangsi, and A. Zhang, *Private aggregation of trajectories*, Proceedings on Privacy Enhancing Technologies, 4 (2022), pp. 626–644.

[70] A. Gleave, M. Dennis, C. Wild, N. Kant, S. Levine, and S. Russell, *Adversarial policies: Attacking deep reinforcement learning*, arXiv preprint arXiv:1905.10615, (2019).

[71] X. Glorot and Y. Bengio, *Understanding the difficulty of training deep feedforward neural networks*, in Proceedings of the thirteenth international conference on artificial intelligence and statistics, 2010, pp. 249–256.

[72] I. Golan and R. El-Yaniv, *Deep anomaly detection using geometric transformations*, arXiv preprint arXiv:1805.10917, (2018).

[73] Y. Gong, M. Abdel-Aty, J. Yuan, and Q. Cai, *Multi-objective reinforcement learning approach for improving safety at intersections with adaptive traffic signal control*, Accident Analysis & Prevention, 144 (2020), p. 105655.

[74] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, *Generative adversarial networks*, Communications of the ACM, 63 (2020), pp. 139–144.

[75] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, *Generative adversarial networks*, arXiv preprint arXiv:1406.2661, (2014).

[76] I. J. Goodfellow, J. Shlens, and C. Szegedy, *Explaining and harnessing adversarial examples*, arXiv preprint arXiv:1412.6572, (2014).

[77] Google, LLC, *Differential Privacy.* `https://github.com/google/differential-privacy`, Sept. 5, 2019.

[78] M. Gramaglia and M. Fiore, *Hiding mobile traffic fingerprints with glove*, in Proceedings of the 11th ACM Conference on Emerging Networking Experiments and Technologies, 2015, pp. 1–13.

[79] K. Grosse, P. Manoharan, N. Papernot, M. Backes, and P. McDaniel, *On the (statistical) detection of adversarial examples*, arXiv preprint arXiv:1702.06280, (2017).

[80] M. E. Gursoy, L. Liu, S. Truex, L. Yu, and W. Wei, *Utility-aware synthesis of differentially private and attack-resilient location traces*, in Proceedings of the 2018 ACM SIGSAC conference on computer and communications security, 2018, pp. 196–211.

[81] S. Ha, P. Xu, Z. Tan, S. Levine, and J. Tan, *Learning to walk in the real world with minimal human effort*, in Conference on Robot Learning, PMLR, 2021, pp. 1110–1120.

[82] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, *Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor*, in International conference on machine learning, PMLR, 2018, pp. 1861–1870.

[83] K. Han, H. Liu, V. V. Gayah, T. L. Friesz, and T. Yao, *A robust optimization approach for dynamic traffic signal control with emission considerations*, Transportation Research Part C: Emerging Technologies, 70 (2016), pp. 3–26.

[84] Y. Han, B. I. Rubinstein, T. Abraham, T. Alpcan, O. De Vel, S. Erfani, D. Hubczenko, C. Leckie, and P. Montague, *Reinforcement learning for autonomous defence in software-defined networking*, in International Conference on Decision and Game Theory for Security, Springer, 2018, pp. 145–165.

[85] P. E. Hart, N. J. Nilsson, and B. Raphael, *A formal basis for the heuristic determination of minimum cost paths*, IEEE transactions on Systems Science and Cybernetics, 4 (1968), pp. 100–107.

[86] S. Hasan, C. M. Schneider, S. V. Ukkusuri, and M. C. González, *Spatiotemporal patterns of urban human mobility*, Journal of Statistical Physics, 151 (2013), pp. 304–318.

[87] S. Hausberger, D. Engler, M. Ivanisin, and M. Rexeis, *Update of the emission functions for heavy duty vehicles in the handbook emission factors for road traffic*, Federal Environment Agency, Vienna/Austria 2003, (2003).

[88] A. J. Havens, Z. Jiang, and S. Sarkar, *Online robust policy learning in the presence of unknown adversaries*, arXiv preprint arXiv:1807.06064, (2018).

[89] A. Haydari, D. Chen, Z. Lai, and C.-N. Chuah, *Mobilitygpt: Enhanced human mobility modeling with a gpt model*, arXiv preprint arXiv:2402.03264, (2024).

[90] A. Haydari, C.-N. Chuah, M. Zhang, J. Macfarlane, and S. Peisert, *Differentially private map matching for mobility trajectories*, in Proceedings of the 38th Annual Computer Security Applications Conference, 2022, pp. 293–303.

[91] A. Haydari and Y. Yilmaz, *Deep reinforcement learning for intelligent transportation systems: A survey*, IEEE Transactions on Intelligent Transportation Systems, (2020).

[92] A. Haydari, M. Zhang, C.-N. Chuah, and D. Ghosal, *Impact of deep rl-based traffic signal control on air quality*, in 2021 IEEE 93rd Vehicular Technology Conference (VTC2021-Spring), IEEE, 2021, pp. 1–6.

[93] A. Haydari, M. Zhang, C.-N. Chuah, J. Macfarlane, and S. Peisert, *Adaptive differential privacy mechanism for aggregated mobility dataset*, arXiv preprint arXiv:2112.08487, (2021).

[94] HBEFA, *Handbuch für emissionsfaktoren des strassenverkehrs (hbefa) (handbook of emission factors for road traffic)*, Umweltbundesamt Berlin, Bundesamt für Umwelt, Wald und Landschaft Bern, (August 2019), pp. Infras AG, Bern.

[95] B. Y. He and J. Y. Chow, *Optimal privacy control for transport network data sharing*, Transportation Research Part C: Emerging Technologies, (2019).

[96] X. He, G. Cormode, A. Machanavajjhala, C. M. Procopiuc, and D. Srivastava, *Dpt: differentially private trajectory synthesis using hierarchical reference systems*, Proceedings of the VLDB Endowment, 8 (2015), pp. 1154–1165.

[97] A. Henao and W. E. Marshall, *The impact of ride-hailing on vehicle miles traveled*, Transportation, 46 (2019), pp. 2173–2194.

[98] A. O. Hero, *Geometric entropy minimization (gem) for anomaly detection and localization*, in Advances in Neural Information Processing Systems, 2007, pp. 585–592.

[99] B. Hoh and M. Gruteser, *Protecting location privacy through path confusion*, in First International Conference on Security and Privacy for Emerging Areas in Communications Networks (SECURECOMM'05), IEEE, 2005, pp. 194–205.

[100] H.-C. Hu and S. F. Smith, *Learning model parameters for decentralized schedule-driven traffic control*, in Proceedings of the International Conference on Automated Planning and Scheduling, vol. 30, 2020, pp. 531–539.

[101] S. E. Huang, W. Wong, Y. Feng, Q. A. Chen, Z. M. Mao, and H. X. Liu, *Impact evaluation of falsified data attacks on connected vehicle based traffic signal control*, arXiv preprint arXiv:2010.04753, (2020).

[102] S. H. Huang, N. Papernot, I. J. Goodfellow, Y. Duan, and P. Abbeel, *Adversarial attacks on neural network policies*, in 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings, 2017.

[103] R. Hult, M. Zanon, S. Gros, and P. Falcone, *Optimal coordination of automated vehicles at intersections: Theory and experiments*, IEEE Transactions on Control Systems Technology, 27 (2018), pp. 2510–2525.

[104] K. Jiang, D. Shao, S. Bressan, T. Kister, and K.-L. Tan, *Publishing trajectories with differential privacy guarantees*, in Proceedings of the 25th International Conference on Scientific and Statistical Database Management, 2013, pp. 1–12.

[105] S. Jiang, Y. Yang, S. Gupta, D. Veneziano, S. Athavale, and M. C. González, *The timegeo modeling framework for urban mobility without travel surveys*, Proceedings of the National Academy of Sciences, 113 (2016), pp. E5370–E5378.

[106] W. Jiang, W. X. Zhao, J. Wang, and J. Jiang, *Continuous trajectory generation based on two-stage gan*, arXiv preprint arXiv:2301.07103, (2023).

[107] F. Jin, W. Hua, M. Francia, P. Chao, M. Orlowska, and X. Zhou, *A survey and experimental study on privacy-preserving trajectory data publishing*, IEEE Transactions on Knowledge and Data Engineering, (2022).

[108] P. Kalnis, G. Ghinita, K. Mouratidis, and D. Papadias, *Preventing location-based identity inference in anonymous spatial queries*, IEEE transactions on knowledge and data engineering, 19 (2007), pp. 1719–1733.

[109] A. Kapp, J. Hansmeyer, and H. Mihaljević, *Generative models for synthetic urban mobility data: A systematic literature review*, ACM Computing Surveys, 56 (2023), pp. 1–37.

[110] A. Karpathy, *mingpt*. `https://github.com/karpathy/minGPT`. Accessed: May 17th 2024.

[111] G. Kellaris, S. Papadopoulos, X. Xiao, and D. Papadias, *Differentially private event sequences over infinite streams*, Proceedings of the VLDB Endowment, 7 (2014), pp. 1155–1166.

[112] M. A. Khamis and W. Gomaa, *Adaptive multi-objective reinforcement learning with hybrid exploration for traffic signal control based on cooperative multi-agent framework*, Engineering Applications of Artificial Intelligence, 29 (2014), pp. 134–151.

[113] D. K. Kholgh and P. Kostakos, *Pac-gpt: A novel approach to generating synthetic network traffic with gpt-3*, IEEE Access, (2023).

[114] D. P. Kingma and M. Welling, *Auto-encoding variational bayes*, arXiv preprint arXiv:1312.6114, (2013).

[115] X. Kong, Q. Chen, M. Hou, H. Wang, and F. Xia, *Mobility trajectory generation: a survey*, Artificial Intelligence Review, 56 (2023), pp. 3057–3098.

[116] J. Kos and D. Song, *Delving into adversarial attacks on deep policies*, arXiv preprint arXiv:1705.06452, (2017).

[117] B. Kővári, B. Pelenczei, S. Aradi, and T. Bécsi, *Reward design for intelligent intersection control to reduce emission*, IEEE Access, 10 (2022), pp. 39691–39699.

[118] D. Krajzewicz, M. Behrisch, P. Wagner, R. Luz, and M. Krumnow, *Second generation of pollutant emission models for sumo*, in Modeling mobility with open data, Springer, 2015, pp. 203–221.

[119] M. N. Kurt, Y. Yilmaz, and X. Wang, *Real-time nonparametric anomaly detection in high-dimensional settings*, IEEE transactions on pattern analysis and machine intelligence, (2020).

[120] J. Kwak, B. Park, and J. Lee, *Evaluating the impacts of urban corridor traffic signal optimization on vehicle emissions and fuel consumption*, Transportation Planning and Technology, 35 (2012), pp. 145–160.

[121] G. Labs, *Using generative, differentially-private models to build privacy-enhancing, synthetic datasets from real data*, 2020 (accessed April 5th, 2021).

[122] N. Lambert, L. Castricato, L. von Werra, and A. Havrilla, *Illustrating reinforcement learning from human feedback (rlhf)*, Hugging Face Blog, (2022). https://huggingface.co/blog/rlhf.

[123] L. Leal-Taixé, M. Fenzi, A. Kuznetsova, B. Rosenhahn, and S. Savarese, *Learning an image-based motion context for multiple people tracking*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 3542–3549.

[124] Y. LeCun, Y. Bengio, and G. Hinton, *Deep learning*, nature, 521 (2015), pp. 436–444.

[125] M. Li, L. Zhu, Z. Zhang, and R. Xu, *Achieving differential privacy of trajectory data publishing in participatory sensing*, Information Sciences, 400 (2017), pp. 1–13.

[126] X. Li, G. Li, S.-S. Pang, X. Yang, and J. Tian, *Signal timing of intersections using integrated optimization of traffic quality, emissions and fuel consumption: a note*, Transportation Research Part D: Transport and Environment, 9 (2004), pp. 401–407.

[127] Y. Li, X. Xu, J. Xiao, S. Li, and H. T. Shen, *Adaptive square attack: Fooling autonomous cars with adversarial traffic signs*, IEEE Internet of Things Journal, 8 (2020), pp. 6337–6347.

[128] Z. Li, D. Jin, C. Hannon, M. Shahidehpour, and J. Wang, *Assessing and mitigating cybersecurity risks of traffic light systems in smart cities*, IET Cyber-Physical Systems: Theory & Applications, 1 (2016), pp. 60–69.

[129] X. Liang, X. Du, G. Wang, and Z. Han, *A deep reinforcement learning network for traffic light cycle control*, IEEE Transactions on Vehicular Technology, 68 (2019), pp. 1243–1253.

[130] T. Lillicrap, J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, *Continuous control with deep reinforcement learning*, CoRR, abs/1509.02971 (2016).

[131] J. Lin, *Divergence measures based on the shannon entropy*, IEEE Transactions on Information theory, 37 (1991), pp. 145–151.

[132] Y.-C. Lin, Z.-W. Hong, Y.-H. Liao, M.-L. Shih, M.-Y. Liu, and M. Sun, *Tactics of adversarial attack on deep reinforcement learning agents*, arXiv preprint arXiv:1703.06748, (2017).

[133] Y.-C. Lin, M.-Y. Liu, M. Sun, and J.-B. Huang, *Detecting adversarial attacks on neural network policies with visual foresight*, arXiv preprint arXiv:1710.00814, (2017).

[134] C. Liu, X. Xu, and D. Hu, *Multiobjective reinforcement learning: A comprehensive overview*, IEEE Transactions on Systems, Man, and Cybernetics: Systems, 45 (2014), pp. 385–398.

[135] H. Liu, W. Zhuang, G. Yin, R. Li, C. Liu, and S. Zhou, *Decentralized on-ramp merging control of connected and automated vehicles in the mixed traffic using control barrier functions*, in 2021 IEEE International Intelligent Transportation Systems Conference (ITSC), IEEE, 2021, pp. 1125–1131.

[136] X. Liu, J. Chen, X. Xia, C. Zong, R. Zhu, and J. Li, *Dummy-based trajectory privacy protection against exposure location attacks*, in International Conference on Web Information Systems and Applications, Springer, 2019, pp. 368–381.

[137] Q. Long, H. Wang, T. Li, L. Huang, K. Wang, Q. Wu, G. Li, Y. Liang, L. Yu, and Y. Li, *Practical synthetic human trajectories generation based on variational point processes*, in Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2023, pp. 4561–4571.

[138] P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, and E. Wießner, *Microscopic traffic simulation using sumo*, in 2018 21st International Conference on Intelligent Transportation Systems (ITSC), IEEE, 2018, pp. 2575–2582.

[139] M. Luca, G. Barlacchi, B. Lepri, and L. Pappalardo, *A survey on deep learning for human mobility*, ACM Computing Surveys (CSUR), 55 (2021), pp. 1–44.

[140] B. Lütjens, M. Everett, and J. P. How, *Certified adversarial robustness for deep reinforcement learning*, in Conference on Robot Learning, PMLR, 2020, pp. 1328–1337.

[141] M. Lyu, D. Su, and N. Li, *Understanding the sparse vector technique for differential privacy*, arXiv preprint arXiv:1603.01699, (2016).

[142] S. Ma and Y. Liu, *Nic: Detecting adversarial samples with neural network invariant checking*, in Proceedings of the 26th Network and Distributed System Security Symposium (NDSS 2019), 2019.

[143] J. Macfarlane, A. Patire, K. Deodhar, and C. Laurence, *Mobile device data analytics for next-generation traffic management*, UC Office of the President: University of California Institute of Transportation Studies, (2021).

[144] F. McSherry and K. Talwar, *Mechanism design via differential privacy*, in 48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07), IEEE, 2007, pp. 94–103.

[145] H. Mei, X. Lei, L. Da, B. Shi, and H. Wei, *Libsignal: An open library for traffic signal control*, arXiv preprint arXiv:2211.10649, (2022).

[146] D. Meng and H. Chen, *Magnet: a two-pronged defense against adversarial examples*, in Proceedings of the 2017 ACM SIGSAC conference on computer and communications security, 2017, pp. 135–147.

[147] D. J. Miller, Y. Wang, and G. Kesidis, *Anomaly detection of attacks (ada) on dnn classifiers at test time*, in 2018 IEEE 28th International Workshop on Machine Learning for Signal Processing (MLSP), IEEE, 2018, pp. 1–6.

[148] D. J. Mir, S. Isaacman, R. Cáceres, M. Martonosi, and R. N. Wright, *Dp-where: Differentially private modeling of human mobility*, in 2013 IEEE International Conference on Big Data, IEEE, 2013, pp. 580–588.

[149] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al., *Human-level control through deep reinforcement learning*, nature, 518 (2015), pp. 529–533.

[150] A. Monreale, G. L. Andrienko, N. V. Andrienko, F. Giannotti, D. Pedreschi, S. Rinzivillo, and S. Wrobel, *Movement data anonymity through generalization*, Transactions on Data Privacy, 3 (2010), pp. 91–121.

[151] J. D. Morgan, *Geoaware-a simulation-based framework for synthetic trajectory generation from mobility patterns*, Master's thesis, Wright State University, 2020.

[152] M. Musleh, M. F. Mokbel, and S. Abbar, *Let's speak trajectories*, in Proceedings of the 30th International Conference on Advances in Geographic Information Systems, 2022, pp. 1–4.

[153] J. Near, *Differential Privacy at Scale: Uber and Berkeley Collaboration*, in Engima, USENIX, January 16, 2018.

[154] H. Ngo and J. Kim, *Location privacy via differential private perturbation of cloaking area*, in 2015 IEEE 28th Computer Security Foundations Symposium, IEEE, 2015, pp. 63–74.

[155] T. Nishi, K. Otaki, K. Hayakawa, and T. Yoshimura, *Traffic signal control based on reinforcement learning with graph convolutional neural nets*, in 2018 21st International Conference on Intelligent Transportation Systems (ITSC), IEEE, 2018, pp. 877–883.

[156] K. Nissim, T. Steinke, A. Wood, M. Altman, A. Bembenek, M. Bun, M. Gaboardi, D. R. O'Brien, and S. Vadhan, *Differential privacy: A primer for a non-technical audience*, in Privacy Law Scholars Conference, vol. 3, 2017.

[157] L. Ou, Z. Qin, Y. Liu, H. Yin, Y. Hu, and H. Chen, *Multi-user location correlation protection with differential privacy*, in 2016 IEEE 22nd International Conference on Parallel and Distributed Systems (ICPADS), IEEE, 2016, pp. 422–429.

[158] K. Ouyang, R. Shokri, D. S. Rosenblum, and W. Yang, *A non-parametric generative model for human trajectories.*, in IJCAI, vol. 18, 2018, pp. 3812–3817.

[159] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, et al., *Training language models to follow instructions with human feedback*, Advances in Neural Information Processing Systems, 35 (2022), pp. 27730–27744.

[160] K. Pandit, D. Ghosal, H. M. Zhang, and C.-N. Chuah, *Adaptive traffic signal control with vehicular ad hoc networks*, IEEE Transactions on Vehicular Technology, 62 (2013), pp. 1459–1471.

[161] N. Papernot, F. Faghri, N. Carlini, I. Goodfellow, R. Feinman, A. Kurakin, C. Xie, Y. Sharma, T. Brown, A. Roy, A. Matyasko, V. Behzadan, K. Hambardzumyan, Z. Zhang, Y.-L. Juang, Z. Li, R. Sheatsley, A. Garg, J. Uesato, W. Gierke, Y. Dong, D. Berthelot, P. Hendricks, J. Rauber, and R. Long, *Technical report on the cleverhans v2.1.0 adversarial examples library*, arXiv preprint arXiv:1610.00768, (2018).

[162] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, *Practical black-box attacks against machine learning*, in Proceedings of the 2017 ACM on Asia conference on computer and communications security, 2017, pp. 506–519.

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

[164] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, *Distillation as a defense to adversarial perturbations against deep neural networks*, in 2016 IEEE symposium on security and privacy (SP), IEEE, 2016, pp. 582–597.

[165] L. Pappalardo, S. Rinzivillo, and F. Simini, *Human mobility modelling: exploration and preferential return meet the gravity model*, Procedia Computer Science, 83 (2016), pp. 934–939.

[166] A. Pattanaik, Z. Tang, S. Liu, G. Bommannan, and G. Chowdhary, *Robust deep reinforcement learning with adversarial attacks*, arXiv preprint arXiv:1712.03632, (2017).

[167] S. Pellegrini, A. Ess, and L. Van Gool, *Improving data association by joint modeling of pedestrian trajectories and groupings*, in European conference on computer vision, Springer, 2010, pp. 452–465.

[168] R. A. Popa, A. J. Blumberg, H. Balakrishnan, and F. H. Li, *Privacy and accountability for location-based aggregate statistics*, in Proceedings of the 18th ACM Conference on Computer and Communications Security, 2011, pp. 653–666.

[169] J. W. Powell, Y. Huang, F. Bastani, and M. Ji, *Towards reducing taxicab cruising time using spatio-temporal profitability maps*, in International Symposium on spatial and temporal Databases, Springer, 2011, pp. 242–260.

[170] W. Qardaji, W. Yang, and N. Li, *Differentially private grids for geospatial data*, in 2013 IEEE 29th International Conference on Data Engineering (ICDE), IEEE, 2013, pp. 757–768.

[171] A. Qayyum, M. Usama, J. Qadir, and A. I. Al-Fuqaha, *Securing connected & autonomous vehicles: Challenges posed by adversarial machine learning and the way forward*, IEEE Commun. Surv. Tutorials, 22 (2020), pp. 998–1026.

[172] M. A. Quddus, W. Y. Ochieng, and R. B. Noland, *Current map-matching algorithms for transport applications: State-of-the art and future research directions*, Transportation research part c: Emerging technologies, 15 (2007), pp. 312–328.

[173] M. P. Raadsen, M. C. Bliemer, and M. G. Bell, *Aggregation, disaggregation and decomposition methods in traffic assignment: Historical perspectives and new trends*, Transportation Research Part B: Methodological, 139 (2020), pp. 199–223.

[174] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, et al., *Improving language understanding by generative pre-training*, OpenAI, (2018).

[175] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, *Language models are unsupervised multitask learners*, 2019.

[176] M. Raeis and A. Leon-Garcia, *A deep reinforcement learning approach for fair traffic signal control*, in 2021 IEEE International Intelligent Transportation Systems Conference (ITSC), IEEE, 2021, pp. 2512–2518.

[177] R. Rafailov, A. Sharma, E. Mitchell, C. D. Manning, S. Ermon, and C. Finn, *Direct preference optimization: Your language model is secretly a reward model*, Advances in Neural Information Processing Systems, 36 (2024).

[178] J. Raghuram, V. Chandrasekaran, S. Jha, and S. Banerjee, *A general framework for detecting anomalous inputs to dnn classifiers*, in International Conference on Machine Learning, PMLR, 2021, pp. 8764–8775.

[179] J. Rao, S. Gao, Y. Kang, and Q. Huang, *Lstm-trajgan: A deep learning approach to trajectory privacy protection*, arXiv preprint arXiv:2006.10521, (2020).

[180] S. Reed, K. Zolna, E. Parisotto, S. G. Colmenarejo, A. Novikov, G. Barth-Maron, M. Gimenez, Y. Sulsky, J. Kay, J. T. Springenberg, et al., *A generalist agent*, arXiv preprint arXiv:2205.06175, (2022).

[181] D. I. Robertson, *Transyt: a traffic network study tool*, TRANSYT, (1969).

[182] C. Rodier, *Review of international modeling literature: Transit, land use, and auto pricing strategies to reduce vehicle miles traveled and greenhouse gas emissions*, Transportation Research Record, 2132 (2009), pp. 1–12.

[183] M. Saberi, H. S. Mahmassani, D. Brockmann, and A. Hosseini, *A complex network perspective for characterizing urban travel demand patterns: graph theoretical analysis of large-scale origin–destination demand networks*, Transportation, 44 (2017), pp. 1383–1402.

[184] J. Salas, D. Megías, and V. Torra, *Swapmob: Swapping trajectories for mobility anonymization*, in International Conference on Privacy in Statistical Databases, Springer, 2018, pp. 331–346.

[185] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, *Proximal policy optimization algorithms*, arXiv preprint arXiv:1707.06347, (2017).

[186] M. Senn, *Uber Movement*, 2020 (accessed October 7, 2020).

[187] D. Shao, K. Jiang, T. Kister, S. Bressan, and K.-L. Tan, *Publishing trajectory with differential privacy: A priori vs. a posteriori sampling mechanisms*, in International Conference on Database and Expert Systems Applications, Springer, 2013, pp. 357–365.

[188] F. Sheikholeslami, A. Lotfi, and J. Z. Kolter, *Provably robust classification of adversarial examples with detection*, in International Conference on Learning Representations, 2020.

[189] D. Shi, J. Ding, S. M. Errapotu, H. Yue, W. Xu, X. Zhou, and M. Pan, *Deep q-network-based route scheduling for tnc vehicles with passengers' location differential privacy*, IEEE Internet of Things Journal, 6 (2019), pp. 7681–7692.

[190] K. Siła-Nowicka, J. Vandrol, T. Oshan, J. A. Long, U. Demšar, and A. S. Fotheringham, *Analysis of human mobility patterns from gps trajectories and contextual information*, International Journal of Geographical Information Science, 30 (2016), pp. 881–906.

[191] K. Sricharan and A. Hero, *Efficient anomaly detection using bipartite k-nn graphs*, Advances in Neural Information Processing Systems, 24 (2011), pp. 478–486.

[192] A. Stevanovic, J. Stevanovic, and C. Kergaye, *Optimization of traffic signal timings based on surrogate measures of safety*, Transportation research part C: emerging technologies, 32 (2013), pp. 159–178.

[193] N. Suriyarachchi, R. Quirynen, J. S. Baras, and S. Di Cairano, *Optimization-based coordination and control of traffic lights and mixed traffic in multi-intersection environments*, in 2023 American Control Conference (ACC), IEEE, 2023, pp. 3162–3168.

[194] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*, MIT press, 2018.

[195] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, *Intriguing properties of neural networks*, arXiv preprint arXiv:1312.6199, (2013).

[196] T. Tan, F. Bao, Y. Deng, A. Jin, Q. Dai, and J. Wang, *Cooperative deep reinforcement learning for large-scale traffic grid signal control*, IEEE transactions on cybernetics, (2019).

[197] H. Technologies, *Here offers privacy tools for enterprises to anonymize personal data and manage user consent globally*, 2021 (accessed April 5th, 2021).

[198] E. Tretschk, S. J. Oh, and M. Fritz, *Sequential attacks on agents for long-term adversarial goals*, arXiv preprint arXiv:1805.12487, (2018).

[199] Z. Tu, F. Xu, Y. Li, P. Zhang, and D. Jin, *A new privacy breach: User trajectory recovery from aggregated mobility data*, IEEE/ACM Transactions on Networking, 26 (2018), pp. 1446–1459.

[200] P. Varaiya, *Max pressure control of a network of signalized intersections*, Transportation Research Part C: Emerging Technologies, 36 (2013), pp. 177–195.

[201] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, *Attention is all you need*, Advances in neural information processing systems, 30 (2017).

[202] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, *Graph Attention Networks*, International Conference on Learning Representations, (2018).

[203] M. Wan, M. Han, L. Li, Z. Li, and S. He, *Effects of and defenses against adversarial attacks on a traffic light classification cnn*, in Proceedings of the 2020 ACM Southeast Conference, 2020, pp. 94–99.

[204] J. Wang, X. Kong, F. Xia, and L. Sun, *Urban human mobility: Data-driven modeling and prediction*, ACM SIGKDD explorations newsletter, 21 (2019), pp. 1–19.

[205] Q. Wang, W. Guo, K. Zhang, A. G. Ororbia, X. Xing, X. Liu, and C. L. Giles, *Adversary resistant deep neural networks with an application to malware detection*, in Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2017, pp. 1145–1153.

[206] Q. Wang, Y. Zhang, X. Lu, Z. Wang, Z. Qin, and K. Ren, *Rescuedp: Real-time spatio-temporal crowd-sourced data publishing with differential privacy*, in IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications, IEEE, 2016, pp. 1–9.

[207] S. Wang, Q. Hu, Y. Sun, and J. Huang, *Privacy preservation in location-based services*, IEEE Communications Magazine, 56 (2018), pp. 134–140.

[208] T. Wang, X. Zhang, J. Feng, and X. Yang, *A comprehensive survey on local differential privacy toward data statistics and analysis*, Sensors, 20 (2020), p. 7030.

[209] X. Wang, J. Li, X. Kuang, Y.-a. Tan, and J. Li, *The security of machine learning in an adversarial setting: A survey*, Journal of Parallel and Distributed Computing, 130 (2019), pp. 12–23.

[210] X. Wang, X. Liu, Z. Lu, and H. Yang, *Large scale gps trajectory generation using map based on two stage gan*, Journal of Data Science, 19 (2021), pp. 126–141.

[211] K. Ward, D. Lin, and S. Madria, *A parallel algorithm for anonymizing large-scale trajectory data*, ACM Transactions on Data Science, 1 (2020), pp. 1–26.

[212] H. Wei, N. Xu, H. Zhang, G. Zheng, X. Zang, C. Chen, W. Zhang, Y. Zhu, K. Xu, and Z. Li, *Colight: Learning network-level cooperation for traffic signal control*, in Proceedings of the 28th ACM International Conference on Information and Knowledge Management, 2019, pp. 1913–1922.

[213] H. Wei, G. Zheng, V. Gayah, and Z. Li, *Recent advances in reinforcement learning for traffic signal control: A survey of models and evaluation*, ACM SIGKDD Explorations Newsletter, 22 (2021), pp. 12–18.

[214] H. Wei, G. Zheng, V. V. Gayah, and Z. Li, *A survey on traffic signal control methods*, CoRR, abs/1904.08117 (2019).

[215] H. Wei, G. Zheng, H. Yao, and Z. Li, *Intellilight: A reinforcement learning approach for intelligent traffic light control*, in Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2018, pp. 2496–2505.

[216] J. Wei, Y. Lin, X. Yao, and J. Zhang, *Differential privacy-based location protection in spatial crowdsourcing*, IEEE Transactions on Services Computing, (2019).

[217] X. Xiao, G. Bender, M. Hay, and J. Gehrke, *ireduct: Differential privacy with reduced relative errors*, in Proceedings of the 2011 ACM SIGMOD International Conference on Management of data, 2011, pp. 229–240.

[218] Y. Xiao and L. Xiong, *Protecting locations with differential privacy under temporal correlations*, in Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, 2015, pp. 1298–1309.

[219] Y. Xiao, L. Xiong, S. Zhang, and Y. Cao, *Loclok: Location cloaking with differential privacy via hidden markov model*, Proceedings of the VLDB Endowment, 10 (2017), pp. 1901–1904.

[220] F. Xu, Z. Tu, Y. Li, P. Zhang, X. Fu, and D. Jin, *Trajectory recovery from ash: User privacy is not preserved in aggregated mobility data*, in Proceedings of the 26th International Conference on World Wide Web, 2017, pp. 1241–1250.

[221] C.-C. Yen, D. Ghosal, M. Zhang, and C.-N. Chuah, *A deep on-policy learning agent for traffic signal control of multiple intersections*, in 2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC), IEEE, 2020, pp. 1–6.

[222] ———, *Security vulnerabilities and protection algorithms for backpressure-based traffic signal control at an isolated intersection*, IEEE Transactions on Intelligent Transportation Systems, (2021).

[223] C.-C. Yen, D. Ghosal, M. Zhang, C.-N. Chuah, and H. Chen, *Falsified data attack on backpressure-based traffic signal control algorithms*, in 2018 IEEE Vehicular Networking Conference (VNC), IEEE, 2018, pp. 1–8.

[224] J. Young Park, R. B. Noland, and J. W. Polak, *Microscopic model of air pollutant concentrations: Comparison of simulated results with measured and macroscopic estimates*, Transportation research record, 1750 (2001), pp. 64–73.

[225] L. Yu, W. Zhang, J. Wang, and Y. Yu, *Seqgan: Sequence generative adversarial nets with policy gradient*, in Proceedings of the AAAI conference on artificial intelligence, vol. 31, 2017.

[226] H. Zhang, S. Feng, C. Liu, Y. Ding, Y. Zhu, Z. Zhou, W. Zhang, Y. Yu, H. Jin, and Z. Li, *Cityflow: A multi-agent reinforcement learning environment for large scale city traffic scenario*, in The world wide web conference, 2019, pp. 3620–3624.

[227] J. Zhang, Q. Yang, Y. Shen, Y. Wang, X. Yang, and B. Wei, *A differential privacy based probabilistic mechanism for mobility datasets releasing*, Journal of Ambient Intelligence and Humanized Computing, 12 (2021), pp. 201–212.

[228] S. Zhang, L. Dong, X. Li, S. Zhang, X. Sun, S. Wang, J. Li, R. Hu, T. Zhang, F. Wu, et al., *Instruction tuning for large language models: A survey*, arXiv preprint arXiv:2308.10792, (2023).

[229] W. Zhang, B. Jiang, M. Li, and X. Lin, *Privacy-preserving aggregate mobility data release: An information-theoretic deep reinforcement learning approach*, IEEE Transactions on Information Forensics and Security, 17 (2022), pp. 849–864.

[230] Z. Zhang, X. Han, Z. Liu, X. Jiang, M. Sun, and Q. Liu, *Ernie: Enhanced language representation with informative entities*, arXiv preprint arXiv:1905.07129, (2019).

[231] X. Zhao, D. Pi, and J. Chen, *Novel trajectory privacy-preserving method based on prefix tree using differential privacy*, Knowledge-Based Systems, (2020), p. 105940.

[232] G. Zheng, X. Zang, N. Xu, H. Wei, Z. Yu, V. Gayah, K. Xu, and Z. Li, *Diagnosing reinforcement learning for traffic signal control*, arXiv preprint arXiv:1905.04716, (2019).

[233] Z. Zheng and P. Hong, *Robust detection of adversarial attacks by modeling the intrinsic properties of deep neural networks*, in Proceedings of the 32nd international conference on neural information processing systems, 2018, pp. 7924–7933.

[234] C. Zhou and R. C. Paffenroth, *Anomaly detection with robust deep autoencoders*, in Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining, 2017, pp. 665–674.

[235] L. Zhou, L. Yu, S. Du, H. Zhu, and C. Chen, *Achieving differentially private location privacy in edge-assistant connected vehicles*, IEEE Internet of Things Journal, 6 (2018), pp. 4472–4481.

[236] Y. Zhu, Y. Ye, S. Zhang, X. Zhao, and J. Yu, *Difftraj: Generating gps trajectory with diffusion probabilistic model*, Advances in Neural Information Processing Systems, 36 (2024).

[237] D. M. Ziegler, N. Stiennon, J. Wu, T. B. Brown, A. Radford, D. Amodei, P. Christiano, and G. Irving, *Fine-tuning language models from human preferences*, arXiv preprint arXiv:1909.08593, (2019).

[238] G. K. Zipf, *The p 1 p 2/d hypothesis: on the intercity movement of persons*, American sociological review, 11 (1946), pp. 677–686.