Identifying Vulnerabilities in Road Networks with Deep Reinforcement Learning

By

JINYOUNG PUNG
THESIS

Submitted in partial satisfaction of the requirements for the degree of

MASTER OF SCIENCE

in

Computer Science

in the

OFFICE OF GRADUATE STUDIES

of the

UNIVERSITY OF CALIFORNIA

DAVIS

Approved:

Dipak Ghosal, Chair

Raissa M. D'Souza

Michael Zhang

Committee in Charge

2022

TABLE OF CONTENTS

ABSTRACT

Multiple disruptions in road networks have the potential for cascading effects that can cause significant degradations of network performance resulting in large increases in travel time. However, it is challenging to identify vulnerable combinations of links in road networks due to the complex interdependency of road segments and a prohibitively large number of possible combinations of links. In this paper, we present a deep reinforcement learning (DRL) framework to identify vulnerable combinations of links in road networks. We let a DRL agent select links to disrupt, and its policy is parameterized with deep neural networks (DNNs). The policy is directly learned from the consequences of disruptions of links (i.e., congestion incurred by the disruptions) in traffic simulations where multiple links are disrupted in a sequence. As a case study, we analyzed vulnerable combinations of links in the road network in the city of Davis in California, and compared the criticality of the disruptions of links selected by the proposed DRL-based method and heuristic-based methods that use betweenness centrality or traffic counts of links. In the results, we observed that disruptions by the DRL agent induced significantly larger congestion and increase in travel time of vehicles than heuristic-based methods. Furthermore, the links selected by the DRL agent reveal that the disrupted links are selected considering both the static properties such as the topology of the road network and the capacity of the links as well as dynamic properties imposed by the traffic demand.

ACKNOWLEDGEMENTS

I am grateful for the generous support of the Republic of Korea Navy which allowed me the opportunity to pursue a master's degree in computer science. The experience of my graduate education at the University of California, Davis expanded my academic background and professional techniques in computer science which is essential for my position as an information and communications officer in ROK Navy. During the past 2 years of my academic journey, there have definitely been periods of struggle but overwhelmingly this has been an incredibly rewarding journey made possible by some amazing and kind people who I will attempt to thank here.

To Professor **Dipak Ghosal**, my committee chair, mentor, and primary advisor. I was very lucky to meet you and take your classes that inspired me to research these interesting topics. Thank you for your invaluable guidance, patience, and encouragement that helped me to grow as a scientist.

To Professor **Raissa D'Souza** and Professor **Michael Zhang**, my committee members, and secondary advisors. Thank you for your insight, feedback, and advice as it was influential and essential in this thesis in that it needed views from various fields of study. I was always excited to meet you all and discuss through zoom every week.

Lastly, my study would not be possible without my loving wife, my life, **Dasom Lee**. Thank you for trust and believing in me and especially for your dedication to taking care of two little babies. I love you to the end of time.

I love you too, my little daughters **Seah Pung** and **Seyoung Pung**. Be kind when you grow up.

1 INTRODUCTION

Intelligent Transportation Systems (ITS) integrate sensing, control, analysis, and communication technologies into transportation systems to resolve current challenges in traditional transportation systems, such as congestion, accident risks, carbon emissions, and air pollution [1] [2]. Although the deployment of ITS is being accelerated thanks to the recent advances in Information and Communication Technology (ICT), the new technologies and connectivity in ITS bring diverse threats by cyber-physical attacks. Attacks on ITS have implications within the physical world and may cause sudden disruptions of critical links in road networks resulting in severe degradation of the road transport systems [2]. In this context, identifying vulnerabilities in road networks is becoming more essential for ITS, so that traffic system managers can effectively allocate resources for prevention and build contingency plans in case of disruptions.

Traditionally, road network vulnerability analysis mostly considered a single disruptive event where a road segment or a group of roads in the same region is disrupted in a single disruptive event [3] [4] [5] [6] [7] [8] . Each link (or a group of links) in a road network is disrupted iteratively and the corresponding consequence of the disruption (e.g., increase in travel time) is measured and ranked to identify vulnerable components. These brute-force approaches with a single disruptive event may be reasonable for threats by disasters or accidents as they occur independently in arbitrary locations. However, malicious attackers have a clear objective to degrade quality of service provided by the road network. As a result, they are likely to disrupt multiple, carefully selected links to maximize the impact of the attack. In these cases, multiple target links in different parts of the network will be selected taking into account both the local and non-local effects that disruption can cause. However, brute-force approaches are not computationally feasible to identify vulnerable combinations of links as the number of possible combinations of links grows exponentially with the number of road segments.

As the traffic flow in the road network introduces interdependence among different sections of the road network, the combined effect of disruptions on a small vulnerable set of links may cause cascading degradations. However, individual vehicles interact in a complex way and the dynamic nature of road networks makes it challenging to identify critical disruption scenarios on a combination of links. Furthermore, the set of most vulnerable links in a multiple-link failure scenario is not simply the combinations of the most vulnerable links with a single-link failure scenario. Finally, even the vulnerable links are not necessarily connected or located in the vicinity of each other [9] [10]. Thus, identifying the criticality of multiple links requires a different approach from the vulnerability analysis for a single disruptive event.

Recently, the application of deep reinforcement learning (DRL) has demonstrated significant successes in diverse control tasks [11] [12] [13] [14] [15] [16] [17]. DRL combines deep neural networks (DNNs) with a framework of reinforcement learning, which enables agents to progressively learn better policies that map complex system states from high-dimensional input data to actions that yield higher rewards. Recently, there has been an increasing interest in DRL-based approaches in the applications of transportation systems such as traffic management systems and autonomous driving [18]. Nevertheless, the application of DRL in road network vulnerability analysis, or in particular, for identifying vulnerable combinations of links, has not been investigated.

In this paper, we propose a DRL framework that aims at identifying vulnerable combinations of links in case of a sequence of disruptions in road networks. We let a DRL agent repeatedly observe the statistical data of a road network, take action to select the target roads to disrupt, and learn the criticality of links from the impact of the disruption. As a case study, we analyzed vulnerable combinations of links in the road network of the city of Davis in California, and compared the criticality of the disruptions of links selected by the proposed DRL-based approach and heuristic-based methods that use betweenness centrality or traffic count of links. We observed the DRL agent identified critical combinations of links causing significantly higher congestion and increase in travel time of vehicles than links selected by heuristics. The contribution of this study can be summarized as follows.

1) We propose, for the first time, a DRL framework to identify critical combinations of links in a road network.
2) We show the robustness of the proposed method with the results with different settings, e.g., traffic demand and the number of disruptions
3) Numerical results demonstrate that the proposed method significantly outperforms heuristic-based methods that use betweenness centrality and traffic counts.

The remainder of the paper is organized as follows. Section 2 introduces the literature review in this field of study, Section 3 describes the adversarial model that determines the problem formulation and disruption scenario, Section 4 presents our custom traffic simulator model used in this study, Section 5 describes the proposed DRL-based approach, Section 6 includes results from a case study in the road network in the city of Davis in California, and Section 7 concludes the paper.

2 RELATED WORKS

While several authors argue that the concept of vulnerability or criticality may have different definitions depending on the context [19] [20] [21] [22], from the road network point of view, it is generally related to the consequence of disruptive events on road segments. For example, in [20] vulnerability in transportation system is defined as "a susceptibility to incidents that can result in considerable reductions in road network serviceability”. Similarly, in [3] road network vulnerability analysis is defined as “the study of potential degradations of the road transport system.”

The dominant approach to assess the consequence of disruptive events is simulation-based method. In most analyses reported in the literature, candidate links to disrupt in the network are removed iteratively, and the impact of each disruption is measured in terms of the increase in travel cost (e.g., travel time) of the vehicles in the network. The vulnerability of links is ranked in accordance with the amount of degradation of network performance [3] [4] [5] [6] [7] [8]. Candidate links can be selected from all the links, or from a subset of important links to make large-scale analysis computationally feasible. A common weakness in such simulation-based methods is that computational cost significantly increases as the number of candidate links increases. At least a single simulation is required for each possible disruptive event, and the number of simulations should be increased to achieve statistically meaningful results considering the stochasticity of traffic simulations [23].

Especially for multiple disruptive events, the number of possible combinations of links can be prohibitively large even for a moderately sized set of candidate links, thus it may not be computationally feasible to simulate the outcomes of all possible combinations. Thus, prior studies for multiple disruptive events focused on optimization approaches that mathematically model the problem, instead of simulation-based methods. The study in [9] formulated the problem as bi-level programming where the upper-level problem is to decide which edge to disrupt, and the lower-level problem is to determine the traffic flows in user equilibrium. They defined an objective function that represents the summation of increased travel time of travelers, and the objective function is linearized to be convex and to have a globally optimal solution. To identify the most vulnerable combination of links, the objective function is optimized to have maximum value. The study was extended to a larger network by adding a preliminary analysis that identifies a set of potentially more vulnerable links and takes only those as candidate links to disrupt [10]. There are several weaknesses in these methods. They either are applicable to only very small network due to high computational cost [9] or take a small subset as candidate links where the results can be highly biased depending on the selection criteria for the candidate links [10]. Also, the user equilibrium traffic assignment ignores the fact that individual vehicles may make en-route routing decisions on their ways to avoid the disrupted links.

In this thesis, we propose a DRL framework for identifying vulnerable combinations of links. We model a sequence of disruptive events as a Markov decision process (MDP), and train the DRL agent to select a link to disrupt at each timestep throughout dynamic traffic simulations. The main difference between our proposed DRL-based method and the above-mentioned simulation-based methods is that the DRL agent can extract knowledge through experiences from traffic simulations to improve its policy. The DRL agent begins with a random policy, but progressively learns better policies to degrade the performance of the road network. Thus, as training proceeds, critical links are progressively more disrupted by the DRL agent, and links whose disruptions have little impact on the network are not selected. This enables the DRL agent to successfully identify vulnerability in the network without brute force simulations by focusing on the critical links.

3 ADVERSARIAL MODEL

In this section, we describe the adversarial model in our study that determines the problem formulation and the disruption scenario. First, we describe the system for analysis and the threat by an adversary against the network. Next, we determine the number of disruptive events. Finally, we present an adversarial model that models the problem as a Markov decision process to apply a DRL-based method.

3.1 SYSTEM AND THREAT

We consider a transportation system for our study as a road network in ITS with the following assumptions:

1. The system has a central controller that has perfect sensing of traffic volume on each road segment. The controller estimates the travel time of road segments and advertises it to vehicles in the network.
2. Vehicles in the network are Connected and Autonomous Vehicles (CAVs) that driving is fully automated (SAE Level 5 [24]). Vehicles route to the shortest path calculated by Dijkstra's algorithm [25] based on the information given by the controller.

We consider a threat by an adversary against the system with the following assumptions:

1. The adversary conducts data-integrity attacks [26] against the traffic delay model of the controller, and the expected travel time of attacked road segments is advertised to have an unacceptably long delay. The attacked roads are logically

disrupted as vehicles avoid routing over them and re-route to alternative paths, increasing traffic congestion in the rest of the network and reducing traffic efficiency.

2. The disrupted links remain disrupted until the end of the simulation as we simulate a relatively short period of time (up to several hours).

Although we consider a cyber-physical threat in ITS, traditional road networks also have a threat of disinformation attacks that cause a similar effect. The study in [27] showed that a malicious adversary can use disinformation (e.g., false notification of road works) to influence travelers to avoid targeted links and create bottlenecks.

3.2 NUMBER OF DISRUPTIONS

We assume that the adversary has resource constraints and can only target a relatively small number of road segments to achieve the objective. For physical attacks (e.g., physically destroying or blocking road segments), it is clear that attackers have finite capability due to the resource constraints on physical assets. Even for logical attacks, that attack must be launched on a small number of segments to avoid being detected by system managers so that the impact of the attack can last for a long time. In this case, the resource constraints are determined by the detectability of the target system. Thus, determining the number of disruptions is a case-specific problem that is closely related to the resource constraints of attackers. Our study focuses on vulnerability analysis given a fixed number of disruptions. In this paper we implemented independent environments with different numbers of disruptions from 2 to 5, and trained a DRL agent for each environment to show that our proposed framework is robust and can be applied to various settings.

3.3 ADVERSARIAL MODEL FOR DRL FRAMEWORK

We model a disruption scenario as a decision-making problem in which an adversary determines multiple disruptive events on the road network. The target links to disrupt are selected by an agent. The adversarial model may have different formulations depending on the length of interval I_d between disruptive events.

If I_d is equal to 0, the adversary disrupts all the targeted links simultaneously as a single disruptive event. Since this scenario has a single decision point, it can be modeled as a bandits problem, such as *multi-armed bandits* (MAB). In bandits setting, an agent has a set of possible actions, and each action is associated with a fixed but unknown reward probability distribution where a reward measures the success and failure of the action. In the bandits-based adversarial model, an action represents selecting a full combination of links to disrupt and a reward represents the degree of service degradation of the road network. However, this formulation has two major challenges. First, each action is independent of each other regardless of whether they share the same selection of links or not. Thus, every action should be sampled at least once, making the problem-solving process similar to the traditional simulation-based method. Second, the size of the action space is prohibitively large. The possible number of combinations of links is $\binom{N_l}{N_d}$ where N_l is the number of links and N_d is the number of disruptions. Getting experiences of all the possible actions is not computationally feasible since even a moderately-sized road network consists of hundreds or thousands of links.

If I_d is greater than 0, disruptions occur in a sequence, and there are multiple decision points that decide which link to disrupt at each disruptive event. Thus, we modeled the problem as a Markov decision process (MDP). A MDP is defined by a 5-tuple $\langle S, A, P, R, \gamma \rangle$ where each element is defined as follows:

1. S is a set of possible states where a state is a concrete and unambiguous representation of the operating environment. In our scenario, a state includes the statistical information of the current road network such as traffic distribution, travel time of links, and which links are disrupted.
2. A is a set of possible actions the agent can apply (i.e., select and disrupt a link).
3. P is a state transition probability matrix that defines how the environment, the road network including travelers in our case, evolves after applying an action.
4. R is the reward function that returns the reward for the action of an agent at each timestep. A reward is the feedback by which we measure the success or failure of an action taken by an agent given a specific state.
5. The discount factor γ is a value between 0 and 1 that defines the importance of the rewards obtained by future decisions. If γ is 0, only the immediate reward matters to the agent, and if γ is 1, the future rewards are equally important to the immediate reward. In our adversarial model, γ should have a value near 1 since the

effect of disruption may not be observed unless the full combination of links is disrupted.

Then, the objective of the adversary is to find a policy that takes a state from S as input and maps the state to an action in A that maximizes the *discounted cumulative rewards* $G = \sum_{t=0}^{T} \gamma^t R(s_t,\ a_t)$. Compared to the combinatorial action space in the bandits setting, the MDP formulation of the adversarial model has a significantly smaller action space, which is the number of candidate links to disrupt regardless of the number of disruptions. It allows the agent in the MDP adversarial model to avoid the *curse of dimensionality* for sampling the experience of each action. Thus, we considered the MDP adversarial model in our study, and considered fixed length I_d for simplicity.

Table 1. *Notation glossary used in this thesis*

Symbol	Description
I_d	Length of interval between disruptive events.
N_l	Number of links
N_d	Number of disruptions
S	Set of possible states
s_t	State of the environment for timestep t
A	Set of possible actions
a_t	Action of the agent for timestep t
P	State transition probability matrix
R	Reward function
r_t	Reward for a state-action pair (s_t, a_t) for timestep t
γ	Discount factor
G	Discounted cumulative rewards
λ_g	Average vehicle generation rate
sat_e	Saturation rate of an edge e

N_{veh}^{e}	Number of vehicles in the queue of an edge e
N_{lane}^{e}	Number of lanes of an edge e
L^{e}	Length of an edge e in meters
L_{veh}	Average length of a vehicle
VDF	Volume-delay function
B	Free-flow travel time
L	Signal light delay
C	Congestion delay
$\beta_1, \beta_2, \beta_3$	Slopes of piece-wise linear function for C
α_1, α_2	Break points of piece-wise linear function for C
k	Maximum number of disruptions in a simulation
N_{veh}	Number of vehicles in the road network
C^{B}	Edge betweenness centrality

One of the classical methods to solve the MDP is to evaluate all the possible state-action pairs. However, it is computationally very expensive with large state and action spaces. Instead of this brute-forcing approach, we take advantage of the powerful generalization techniques of DRL. DRL extracts knowledge from visited states that can be used for unexplored states using deep neural networks. This enables a DRL agent to make a successful decision in states not experienced in advance. Figure 1 illustrates the DRL framework for our road network disruption scenario. At the beginning of the scenario, the timestep t is initialized as 1. The agent observes the initial state $s_{t=1}$ of the environment, and given the state, takes an action that selects and disrupts a link. The environment elapses I_d time and returns the next state s_{t+1} and reward r_t for the current state and action pair (s_t, a_t). This process is iterated until the terminate timestep that the number of disruptions reaches to the maximum number of disruptions. The policy of the agent is initialized with a random policy, but the agent progressively learns better policy that yields higher cumulative rewards by exploring different disruption strategies.

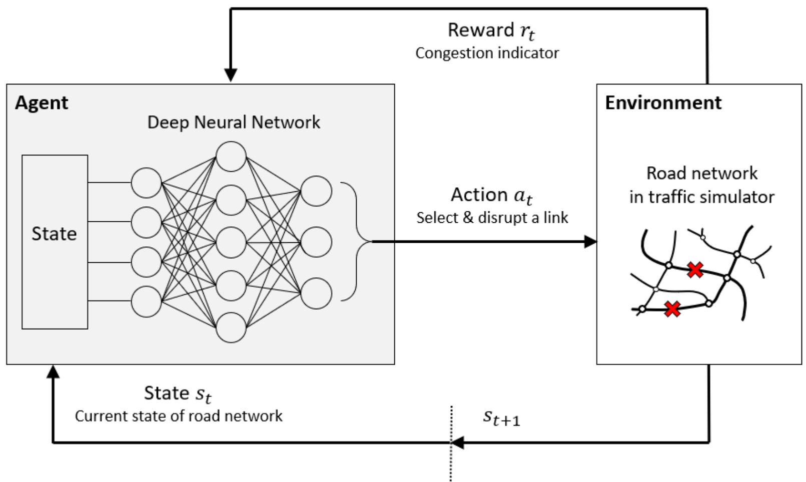

Figure 1. *DRL framework for road network disruption scenario.*

4 SIMULATION MODEL

In our experiments, we used a custom-built traffic simulation model to simulate the disruption scenario in a real-world road network. This section describes the architecture of the simulation model and the simulation scenario.

4.1 MODEL ARCHITECTURE

We built a process-based discrete-event simulator that models the operation of a road network system as a sequence of events. Each event occurs on a specific process, and the state of the system changes as a result of the event. The initial state of the system is an empty road network that is represented as a weighted directed graph with the nodes being the traffic intersections, the edges being the road segments, and the weights being some attribute of road segments (e.g., travel time or length). Each edge has a queue where vehicles that enter the edge are required to wait as much as the travel time of the edge. When the simulation begins, traffic is dynamically assigned in the network, and each individual vehicle proceeds over the queueing links following the shortest path route to its destination.

The simulation model has three logically separated processes: traffic generation process, vehicle movement process, and disruption process.

1. **Traffic generation process** generates new vehicles to travel within the road network and injects them into the system following a Poisson process with rate λ_g vehicles per second. Thus, the interval between vehicle generations is a random

variable that has the exponential distribution $Exp(\lambda_g)$. Origin and destination (OD) nodes are sampled for each new vehicle, where the probability of sampling a node for origin or destination is weighted by the traffic demands. The shortest path between an OD pair is calculated by Dijkstra's algorithm [25], and the new vehicle is added to the tail of the queue of the start edge of the shortest path.

2. **Vehicle movement process** lets the vehicles in the queue of each edge move forward to the next edge in their path if the next edge is not full and the vehicles waited in the current queue at least the travel time of the current edge. Thus, congestion in the network backpropagate as vehicles gets delayed in congested edges. An edge is considered full if its saturation rate is equal to or greater than 1. The saturation rate of an edge e is defined by

$$sat_e = \frac{N_{veh}^{e} \times L_{veh}}{N_{lane}^{e} \times L^{e}} \tag{1}$$

where N_{veh}^{e} is the number of vehicles in the queue of the edge, L_{veh} is the average length of a vehicle, N_{lane}^{e} is the number of lanes of the edge, and L^{e} is the length of the edge in meters. L_{veh} is set as 4.5m in our study. In simulations, an event for this process occurs in every second, which iterates a loop over all the edges in the network to check the head of the queue and move vehicles that satisfy the conditions.

3. **Disruption process** disrupts edges in the road network. The first disruption occurs at the beginning of this process, and a new edge is disrupted every I_d seconds until the number of disruptions reaches the designated maximum number of disruptions

for the simulation. On disruption, vehicles in the network re-route to alternative paths to avoid the disrupted edge.

In real-world road networks, traffic delay is affected by a myriad of factors, thus the travel time of a road segment is not always the same as the free-flow speed, and changes over time. Our custom simulator uses a volume-delay function (VDF) to reflect this dynamic nature of traffic delay, and the travel time of each edge is re-calculated whenever the traffic volume on the edge changes. The travel time of an edge e computed by VDF is defined by

$$VDF(e) = B(1 + C) + L \tag{2}$$

$$C = \begin{cases} \beta_1 \times sat_e, & 0 \le sat_e < \alpha_1 \\ \alpha_1\beta_1 + (sat_e - \alpha_1) \times \beta_2\,, & \alpha_1 \le sat_e < \alpha_2 \\ \alpha_1\beta_1 + \alpha_2\beta_2 + (sat_e - \alpha_2) \times \beta_3, & \alpha_2 \le sat_e \le 1 \end{cases} \tag{3}$$

where B is the free-flow travel time that is computed by dividing the length of the edge by the speed limit, L is the signal light delay where a constant delay is assigned depending on road types, and C is the congestion delay that is a function of sat_e where the value of sat_e is clipped to 1. The definition of VDF is similar to traditional VDFs [28] where the delays increase slowly for small traffic volumes and increase very steeply for large traffic volumes. For simplicity, we used a piecewise-linear function with three different slopes ($\beta_1 = 1,\ \beta_2 = 3,$ and $\beta_3 = 20$) and two break points ($\alpha_1 = 0.5$ and $\alpha_2 = 0.8$) for congestion delay C.

4.2 SIMULATION SCENARIO

The scenario of a simulation can be decomposed into three different periods: warming-up period, disruption period, and cooling-down period. First, the scenario starts with warming-up period because the initial state of the network is an empty network. At the beginning of simulation, the vehicles injected into the system are like driving at midnight with small traffic volumes and drive near free-flow speed without congestion delay. Thus, the data obtained near the start of a simulation have a strong bias toward smaller travel time. The warming-up period enable the network to fill with vehicles before we observe and analyze the behavior of vehicles. Next, disruption period takes place with disruption process that disrupts an edge in the network in every disruption interval I_d. This period produces the perturbations that degrade the throughput of the network. Finally, in cooling-down period, new traffic is not generated, and no more link is disrupted. The network discharges remaining vehicles in the system by letting them continue to finish their travel. Cooling-down period allows the effect of congested queues to be accounted for the measurement of travel time in results. Figure 2 shows operating processes in each period throughout a simulation run.

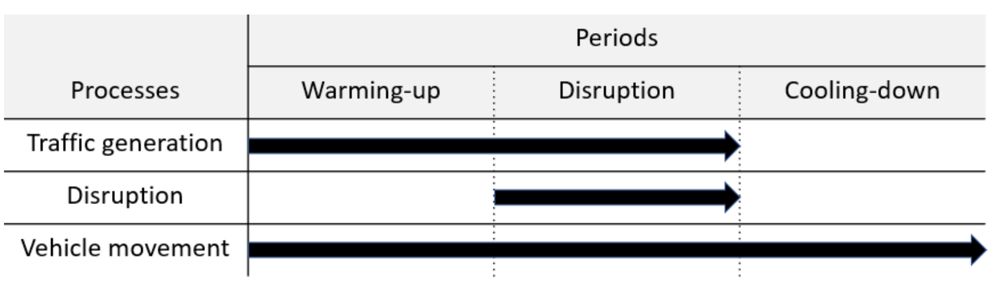

Figure 2. *Processes operating during different phases of the simulation run.*

5 DESCRIPTION OF THE DRL FRAMEWORK

In this section, we describe our proposed DRL framework. We defined state/action space representations, and the reward function to perform vulnerability analysis in road networks. These representations determine how the DRL agent recognizes the state of the environment, applicable actions for the DRL agent, and the aim that the DRL agent optimizes its policy.

We modeled a simulation run as a k-step episode where k is the maximum number of disruptions in the simulation. As mentioned in *Section 3.2*, we built four independent environments with different values of k from 2 to 5, and trained an independent DRL agent for each environment. DRL agents iteratively observe the state of the environment and take an action at each timestep. If timestep t reaches k, the agent takes the last action, the rest disruption period proceeds without any disruption, and the environment returns the terminate state $s_{t=k+1}$. Figure 3 shows the k-step episode for a simulation run.

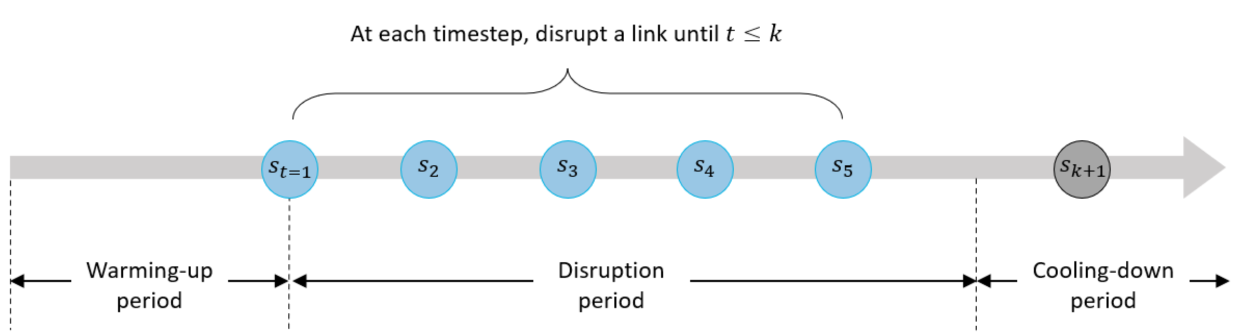

Figure 3. *A simulation run modeled as a k-step episode. The figure illustrates an episode with k=5. The episode starts with the initial state s_1. At the last timestep t=k, the agent takes the last action for the episode and the environment returns the terminate state s_{k+1}.*

5.1 STATE SPACE REPRESENTATION

For an unambiguous state space representation, the state should include not only information about the road segments but also information about the traffic volume and traffic flow. To obtain this observation, the statistical information of both a link and the vehicles on the link is represented as a vector for each link, and the vectors for all the edges are concatenated to a single vector as a state, which is the input to the agent. The vector for each edge has seven elements as follows:

1. *Saturated queue length*: the number of vehicles that causes the saturation rate of the edge to become greater than or equal to 1. If this value is small, the edge has a low capacity for vehicles, and thus it has more chance to be congested. It can be also used as a congestion indicator when combined with *current queue length*.
2. *Current queue length*: the number of vehicles that are currently on the edge. In the observation in which the vectors for all the edges are concatenated, this element shows the distribution of traffic volume.
3. *Traffic count*: the number of vehicles that visited the edge between the current timestep t and last timestep $t-1$. If $t=1$, the traffic count in the warming-up period is measured since the initial observation is at the beginning of the disruption period. This is an important element in that a high traffic count implies that the link is frequently selected for the shortest paths for the vehicles, contributing to the functionality of the transportation network.

4. *Flow count*: the number of vehicles that have the edge in their routes beyond the current edge. This element allows the agent to recognize the directions that the vehicles are heading.
5. *Disrupted or not*: binary flag whether the edge is disrupted or not.
6. *Speed limit*: speed limit on the edge in km/h.
7. *Length*: length of the edge in meters.

5.2 ACTION SPACE REPRESENTATION

We defined a discrete action space where each action corresponds to disrupting a specific link from a set of candidate links. Thus, the size of the action space is the same as the number of candidate links. Candidate links may either include all the links in the network or be a set of pre-selected links determined by specific link criteria. However, if the study network is large, limiting the candidate links to pre-selected links allows the agent to have a concise action space, so that it can easily explore the sample space of trajectories.

5.3 REWARD FUNCTION

With respect to the reward function, we use the increase in the number of vehicles in the network, caused by the disruptions. To determine the increase in the number of vehicles due to disruptions, we need a baseline on the number of vehicles over time in a fully operational network without any disruption. We ran a simulation with a baseline scenario in which the disruption period is replaced with the same length of additional warming-up period. This implies that the simulation has the same runtime but does not have any disruption. Considering the stochasticity of the traffic simulator, we repeated the simulations 20 times and determined the average number of vehicles at each timestep. For the state and action pair (s_t, a_t) at the timestep t, we reward the agent as the difference in the number of vehicles at the next timestep $t + 1$ between the network with disruptions and the baseline (i.e., the fully operational network). For the immediate reward at the last timestep $t = k$, the difference is measured at the terminate state $s_{t=k+1}$.

The simulation time at the terminate state $s_{t=k+1}$ is a time point in the middle of the cooling-down period that the vehicles without congestion finished their travels but the vehicles in congested areas are still not fully discharged, which should be calibrated for a specific road network. We consider the rewards obtained at $t < k$ as a short-term effect of disruptions since the effect of disruptions is measured in a relatively short disruption interval I_d. On the other hand, the reward obtained at $t = k$ has a long difference in simulation time between t where the disruption occurred by the action a_t and the termination timestep $t + 1$ where the reward for a_t is measured. Thus, the reward represents a long-term effect of disruptions. Also, the disruption of the full combination of k links is complete at the last timestep $t = k$, and we can finally observe the entire combined effect of disruptions only at the last reward. Thus, we put more weight on the last reward by dividing the reward by a constant $m = 3$ if $t < k$. The reward function R for a state and action pair (s_t, a_t) at timestep t is defined by

$$R(s_t, a_t) = \begin{cases} \dfrac{N_{veh}(s_{t+1}) - b_{t+1}}{m} & t < k \\ N_{veh}(s_{t+1}) - b_{t+1} & t = k \\ 0 & \text{if } a_t \text{ is not unique} \end{cases} \quad (4)$$

where $N_{veh}(s_t)$ is the number of vehicles in the system at the timestep t, and b_t is the number of vehicles in the fully operational network at the same simulation time as timestep t. Since the action space does not change, the agent may select an already disrupted edge. To train the agent to avoid this duplicate action, we do not reward for the actions that are already taken in the previous timesteps.

There are two reasons that we used the number of vehicles to reward the agent instead of other metrics such as travel time of vehicles. First, the vehicles are entering the system at an average rate λ_g vehicles per second. Under this condition, the increase in the number of vehicles represents that the transportation functionality of the network is degraded as the discharge rate of the vehicles is decreased. Second, the agent does not have to continue the cooling-down period until the network fully discharges all the vehicles. If we use the travel time of vehicles for rewards, all the vehicles must finish traveling to measure the travel time. This will make sampling episodes for the agent to become computationally more expensive. However, the number of vehicles can be measured at any time point throughout a simulation. In the cooling-down period, vehicles in the areas without congestion finish their travel very soon, and there is a rapid drop in the number of vehicles at the beginning of the period. Then, the number of vehicles decreases very slowly with a long tail since the vehicles in the congested areas have long delays. We can measure the difference in the number of vehicles between the disrupted network and the baseline after the rapid drop at the beginning of the period, instead of waiting for the full discharge. We measured the travel time of vehicles for evaluation, not training, and found that there is a strong correlation between the number of vehicles over time in the system and the travel time of vehicles.

5.4 DRL ALGORITHM

Besides the design of state/action space representations and reward function, the selection of a DRL algorithm that is appropriate for the nature of the problem is also important. Our problem has a large action space where an action is to select and disrupt a link from hundreds or thousands of candidate links. Thus, algorithms based on Q-learning are likely to be infeasible due to the curse of dimensionality [29], and algorithms that can deal with the large action space should be selected. We choose to use Proximal Policy Optimization (PPO) [30] which is one of the state-of-the-art DRL algorithms. PPO is an actor-critic method that directly optimizes its policy with a clipped surrogate objective and entropy regularization, which enables stable training and improved exploration. Also, the algorithm operates with multiple workers so that training episodes can be sampled fast. It is important to note that, however, this specific choice of DRL algorithm is not fundamental to our proposed approach.

6 A CASE STUDY

In this section, we perform a case study that adapts our proposed DRL-based solution for city-level road network vulnerability analysis of the city of Davis in California. We evaluated the effect of disruptions by DRL agents in different settings (e.g., traffic demand, number of disruptions), and compared the criticality of the disruptions of links selected by the proposed DRL-based approach with heuristic-based methods that use betweenness centrality or traffic counts of links.

6.1 ROAD NETWORK DATA

OSMnx [31] is a Python package that downloads road networks from OpenStreetMap [32] and constructs them into weighted directed graphs with the nodes being the traffic intersections, the edges being the road segments, and the weights being some attribute of road segments including the length and type of roads. Using OSMnx, we represented the road network of the city of Davis as a weighted directed graph.

The complete road network has a hierarchical structure. It includes high-level roads (*motorway, primary, secondary,* and *tertiary roads*) which transport a large number of vehicles at fast speeds and low-level *residential roads* which have low speed limits and are used to provide access between high-level roads and local residential areas. To simplify the network, we omitted all the residential roads, and iteratively removed network elements that do not contribute to overall transportation functionality, such as self-loops, dead-ends, and interstitial nodes that lie on the same road line. After simplification, we got the output network that has 376 nodes and 753 edges. Excluding the links that consist of small lamps at the intersections, we take 625 edges as a set of candidate links to disrupt for the DRL agent. Figure 4 shows the complete and simplified road networks and the road type of edges.

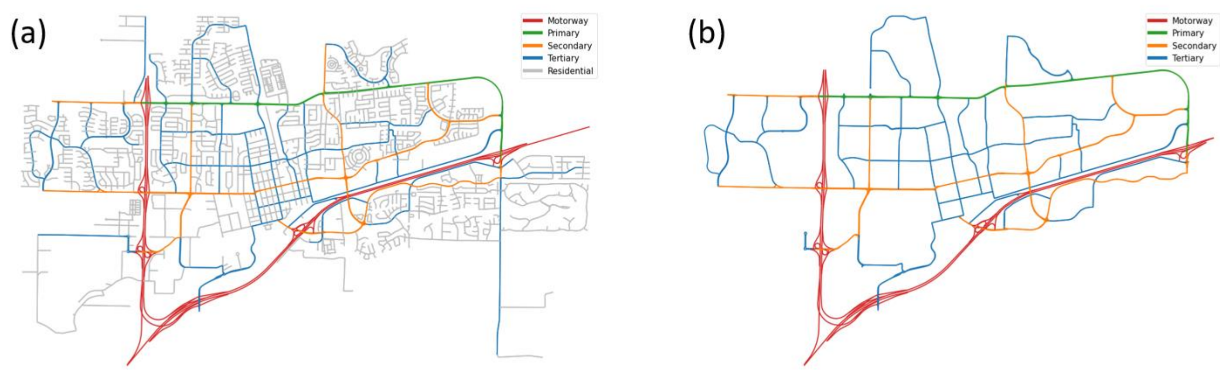

Figure 4. *The road network in the city of Davis. The color of edges represents the road type. (a) is the complete road network and (b) is the simplified road network.*

As described in Section 4.1, in the simulation model, the travel time of an edge is determined by the free-flow travel time, congestion delay, and signal light delay. To obtain the free-flow travel time of the road segments, we assigned speed limits depending on the road types. Specifically, we assign 110 km/h for motorways, 60 km/h for primary roads, 50 km/h for secondary roads, and 40 km/h for tertiary roads. To obtain the free-flow travel time, we divided the length of each edge by its speed limit. The signal light delay is set 0, 10, 10, and 6 seconds for motorways, primary, secondary, and tertiary roads, respectively.

6.2 TRAFFIC DEMAND

Davis is a small college town with a consistent traffic pattern in which most traffic is concentrated at the University of California, Davis (UC Davis). Commuters to the university who are non-residents in Davis enter the city mainly from I-80, the freeway that runs through the south of Davis. We use this empirical knowledge about the city to build traffic demands where the probability of sampling a node for origin or destination is weighted by the traffic demands. We consider two traffic demands one for morning rush hour and the other for the evening rush hour.

In the morning traffic demand, the eastbound and westbound nodes of the I-80 have 10% and 15% of probability respectively to be sampled as an origin node, and all other nodes have 75% probability of being sampled as an origin. For the probability to sample destinations, 5 nodes surrounding UC Davis, 7 nodes for popular groceries, and 7 nodes for the downtown area are assigned probability of 50%, 10%, and 10%, respectively, and all other nodes have the uniform random probability of 30%. Figure 5 shows the locations of the nodes with weights in the morning traffic demand. In the evening traffic demand, the weights for sampling origin and destination nodes are switched from the morning traffic demand, so that the traffic demand has a reversed direction.

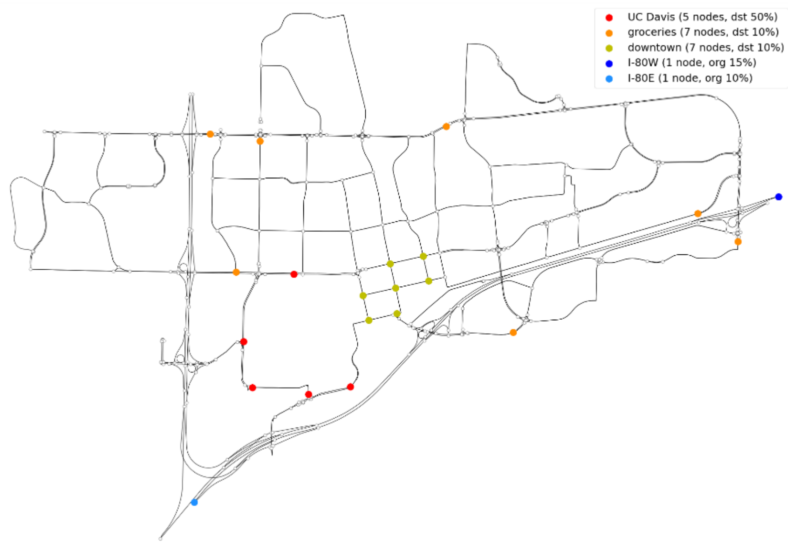

Figure 5. *Traffic demand in the morning. Popular areas are weighted for sampling OD nodes.*

6.3 CALIBRATION

Calibration is the adjustment of parameters in a traffic simulation model to improve the model's ability to reproduce local driver behavior and traffic performance characteristics [33]. In our study, we calibrated several essential parameters in the simulation model: the length of the warming-up, disruption, and cooling-down periods, and the traffic generation rate.

The warming-up period is generally included in traffic simulation studies, and as a rule of thumb, the longest travel time of the system or double the travel time is recommended for the length of the warming-up period [23] [33]. The travel time between possible OD pairs in the road network in Davis has the distribution in Figure 6, where the longest travel time is 970 seconds. We set the length of the warming-up period to 1000 seconds as it satisfies the guidelines for calibrating simulation models. Furthermore, in our preliminary experiments with a wide range of traffic generation rates, we found the steep increase in the number of vehicles at the beginning of simulations turns to a steady slope after 1000 seconds in common.

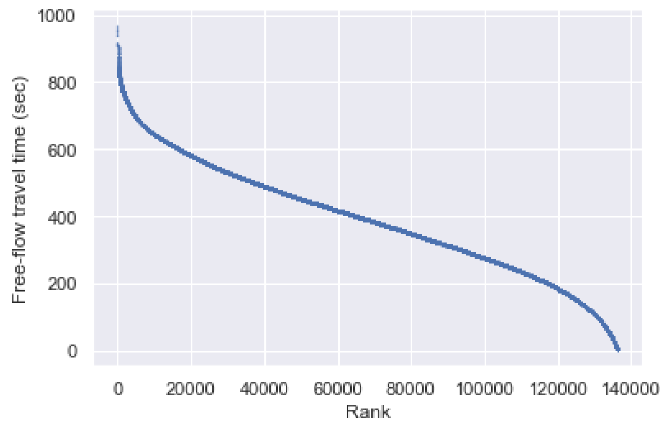

Figure 6. *Free-flow travel time between OD pairs in the road network in Davis.*

For the evaluation scenario, the cooling-down period continues until all the vehicles in the network are fully discharged so that the travel of vehicles can be measured. However, for the training scenario, we stop the cooling-down period before vehicles are fully discharged for faster training, and the agent gets rewards based on the number of remaining vehicles that suffer from congestion. Thus, the length of the cooling-down period for the training scenario should be enough to discharge traffic in areas without congestion. We set 1500 seconds to be the length of the cooling-down period, which is one and a half times the warming-up period since the travel time of vehicles is likely to be larger than the beginning of the simulation as congestion delay is added.

The length of interval I_d between disruptions is an important parameter for both training the agent and the criticality of disruptions. If I_d is too short, disruptions occur almost simultaneously, and it is difficult to observe the short-term effects of the individual disruptions which are used for the rewards for the state and action pairs at non-terminal timesteps. On the other hand, if I_d is too long, disruptions become sparse, allowing the network to have more time to digest the perturbations, and the cascading effect of disruptions may decrease. We set 200 seconds to I_d, which is about half the mean travel time of the possible OD pairs in the system, considering the tradeoff between short and long intervals. In our study, we consider maximum 5 disruptions in a simulation, thus the total length of the disruption period is set to 1000 seconds. In order to have the same traffic volume, the length of the disruption period is the same 1000 seconds regardless of the number of disruptions in a simulation. If the number of disrupted links reach the designated number of disruptions in a simulation, the rest disruption period proceeds without additional disruptions.

The traffic generation rate λ_g determines the number of vehicles that are injected into the system per second. The value of λ_g should be reasonably high since we consider the traffic demands in rush hours and the adversary against the network is likely to attack when the traffic volume is high to maximize the impact of attacks. However, a too high value of λ_g is unrealistic and may cause grid-lock in simulations where vehicles block each other and the network is locked [23]. To find the proper value for λ_g, we performed a linear search by running simulations with different values of λ_g from 0.5 to 5. In the simulations, the runtime for each period was the same as the training scenario for the DRL agent, and disruptions did not occur. The results in Figure 7 show that vehicles are almost fully discharged after the cooling-down periods for 0.5 and 1 vehicles per second (v/s), but beyond 1.5 v/s the number of remaining vehicles at the end of simulations rapidly increases. Thus, we selected 1.5 v/s for the value of λ_g. At this rate the traffic volume is large enough to remain after the cooling-down period, but congestion is not unacceptably heavy or causes grid-locks.

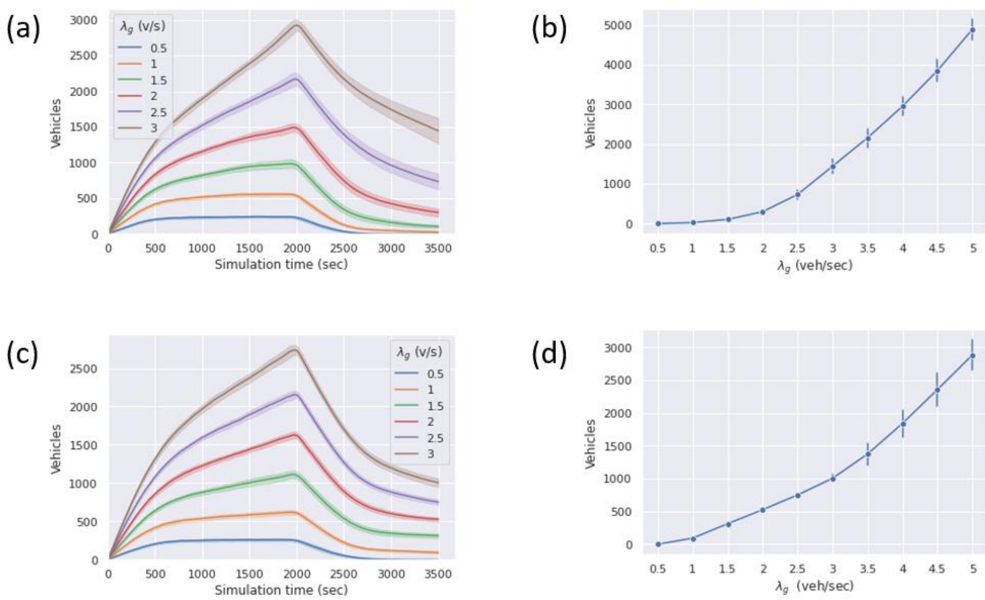

***Figure 7**. The number of vehicles over time (left column) and the number of remaining vehicles at the end of simulations (right column). (a) and (b) are for the morning traffic demand, (c) and (d) are for the evening traffic demand.*

6.4 EXPERIMENTS AND RESULTS

In our experiments, we evaluated the DRL-based framework described in Section 5 with the adversarial model and simulation model described in Section 3 and Section 4, respectively. First, we check the robustness of our proposed method by training independent DRL agents in different environments where each environment has a different number of disruptions from 2 to 5 and has morning or evening traffic demand. This experiment shows the applicability of the method in various settings. Next, we study the rationale behind the behavior of the DRL agents by analyzing the disrupted links by the agents. Finally, we compare the performance of our DRL agent with respect to other heuristic-based methods that use betweenness centrality or traffic counts to select links to disrupt.

Regardless of the number of disruptions or traffic demands in the environment, DRL agents can improve their policies to obtain higher cumulative rewards throughout training. In Figure 8 (a) and (c), DRL agents in all the environments obtain a small mean episode reward at the beginning of training, which means that the performance of disrupted networks is almost the same as the fully operational network without any disruption. However, the DRL agents progressively improve their policies and obtain higher mean episode reward through training. This implies that they can identify vulnerable combinations of links and the disruptions have great impacts on the networks. After 250k training timesteps, the policies of DRL agents converged and obtained steady mean episode rewards.

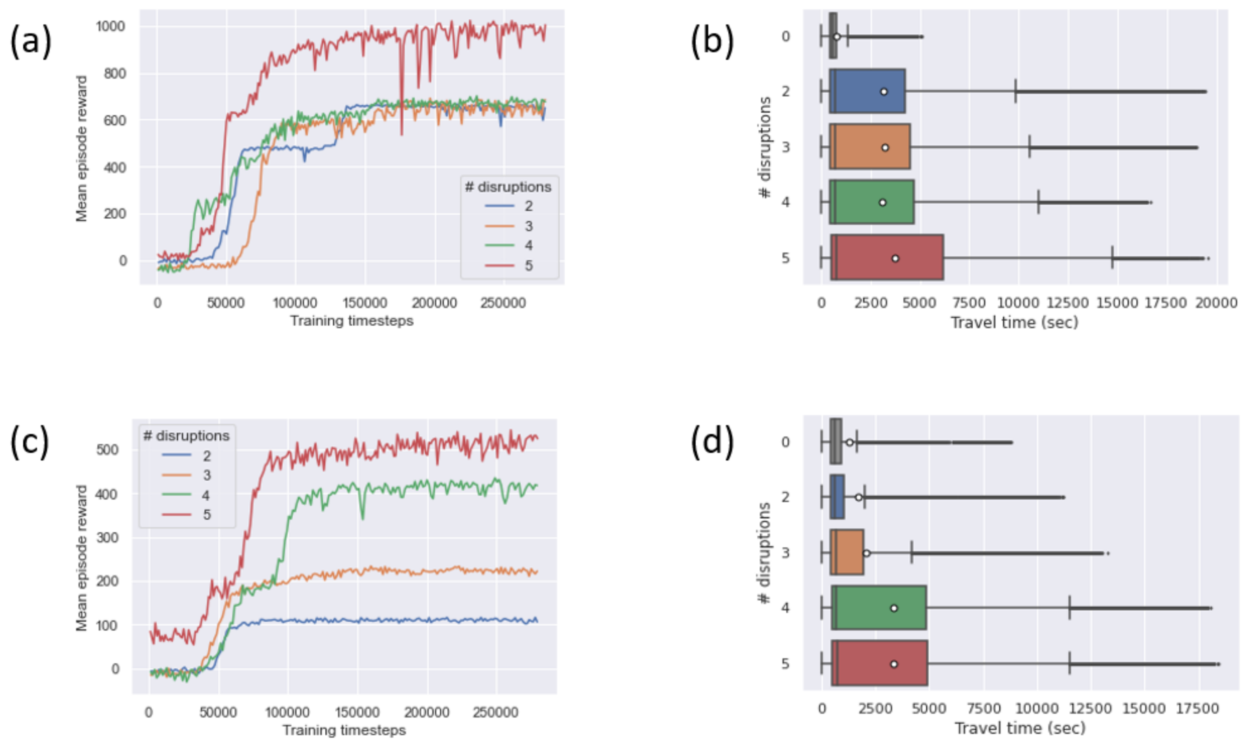

Figure 8. *Mean episode reward over training (left column) and the travel time of vehicles in environments (right column) with different number of disruptions. White marks in the box plots represent the mean travel time. (a) and (b) are for the morning traffic demand, (c) and (b) are for the evening traffic demand.*

We evaluated the DRL agents with an evaluation scenario in which the cooling-down period continues until all the vehicles in the system finish their travel, so that the measured travel time includes the effect of congestion. The travel time of the vehicles whose travel is finished before the end of the warming-up period is not measured as they are not affected by disruptions. Considering the stochasticity of traffic simulation, we ran the evaluation scenario 20 times for each agent. We included the results in a fully operational network without disruption for comparison as a benchmark. Figure 8 (b) and (d) show the evaluation results. We can observe that although the medians do not have a big difference, the number of vehicles that suffered from large delays greatly increases with the disruptions by the DRL agent, thus the average travel time also increases. This results clearly show that the DRL agents can identify the vulnerabilities in the network and cause congestion to degrade the performance of the network.

To better understand the behavior of the DRL agents, we investigated the selections of the DRL agents. Figure 9 shows the most frequently selected combinations of links by the DRL agents in the evaluations with 5 disruptions. We can observe that the agents first disrupted the bottleneck links for the access between UC Davis and the rest of the network, and then disrupted the freeways that have a high capacity. Since there are many commuters to UC Davis, disruption of the bottleneck links to the university forces the traffic to be concentrated on the remaining bottlenecks causing heavy congestion. Also, the disruption of the high-capacity roads compels the vehicles to go through low-capacity roads that have larger delays and easily get congested. This shows that the DRL agent captures the singularities of the network topology, traffic demands, and the capacity of the roads, and exploit the knowledge to select critical combinations of links.

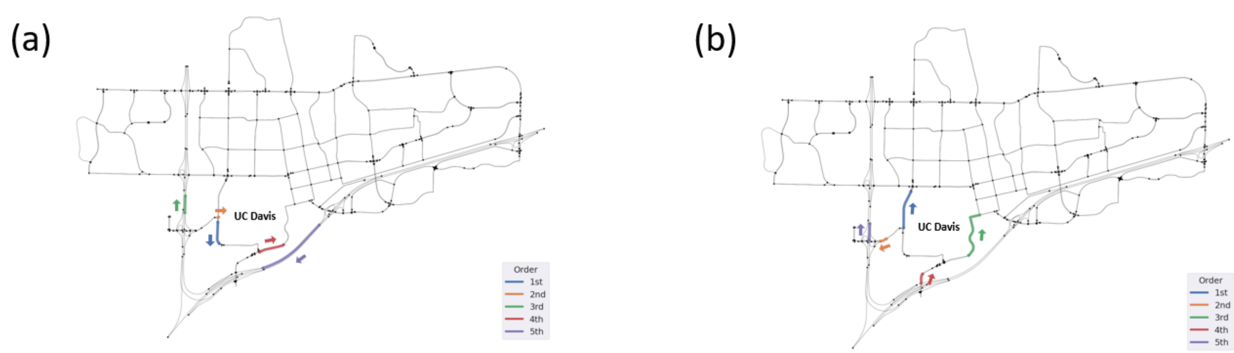

Figure 9. *The most frequently disrupted links by the DRL agents in the environments with 5 disruptions. (a) morning traffic demand (b) evening traffic demand.*

We evaluated the proposed DRL-based method in the environment with 5 disruptions against two different heuristic-based methods that use betweenness centrality and traffic counts respectively. Many prior studies [34] [35] [36] [37] in road network analysis used betweenness centrality to identify important road segments. Betweenness centrality [38] shows how frequently an edge lies on the shortest paths connecting a pair of nodes in a graph. In a road network, the higher betweenness of an edge, the more it provides the shortest routes between OD pairs in the network and is likely to contribute to the transportation in a city. The betweenness centrality C^B of an edge e is defined by

$$C^B(e) = \frac{1}{N(N-1)} \sum_{s,t \in V} \frac{\sigma(s,t|e)}{\sigma(s,t)} \tag{5}$$

where N is the number of nodes in a graph, V is the set of nodes, $\sigma(s,t)$ is the number of shortest paths between an origin and destination pair (s,t), and $\sigma(s,t|e)$ is the number of those paths that passing through edge e. Considering that vehicles reroute to avoid disruptions, we repeated to compute C^B in the network, disrupt the edge with the highest C^B, and then re-compute C^B and disrupt the next edge with the highest C^B until 5 edges are disrupted. Figure 10 shows the 5 links selected by this approach. In the evaluation scenario, one of the pre-selected edges is disrupted at each timestep in the order the edges are selected by the heuristic.

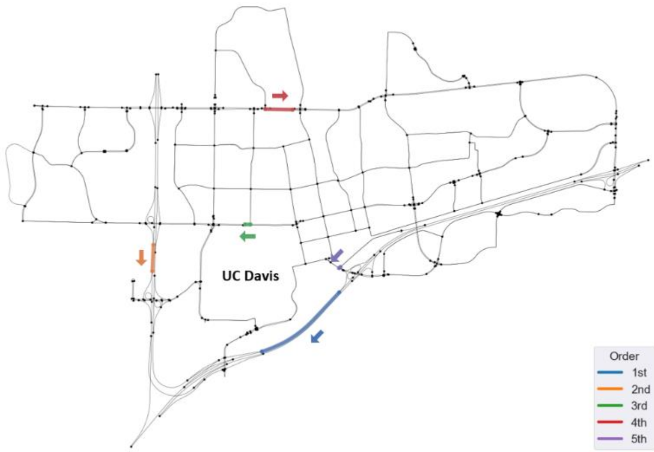

Figure 10. *5 selected links by the heuristic that uses betweenness centrality.*

The other heuristic-based method is to use traffic counts to select links to disrupt. Traffic counts of a road segment show the traffic volume that went through the road for a specific period in the past, which are often used to identify critical flows and improve traffic systems in the real world, and are collected by many governmental traffic departments [23]. In the evaluation scenario, traffic counts are measured between the current timestep and the last timestep by the number of vehicles that entered each edge. For the initial timestep, the traffic counts in the warming-up period are measured. At each timestep, the edge with the highest traffic counts is disrupted.

Figure 11 shows the travel time of vehicles in the environments in which 5 links to disrupt are determined by heuristic-based methods (C^B and traffic counts) and our proposed DRL-based method, respectively. The results in the fully operational network are included for comparison as a benchmark. We can observe that the DRL-based method significantly outperforms the heuristic-based methods with both morning and evening traffic demands, causing a huge increase in the travel time of vehicles. These results imply that the heuristic-based methods are limited in identifying vulnerabilities compared to the proposed DRL-based method. Since the heuristics focus on specific aspects of the network, they may ignore some of the complex and dynamic nature of road networks. For example, although betweenness centrality is a useful indicator to identify topological singularities in networks, it overlooks the traffic demands in road networks and the time-varying delays of road segments. On the other hand, the DRL-based method does not have any pre-defined heuristic. Instead, the DRL agent directly optimizes its policy to cause large degradation of the performance of a road network. Unlike heuristic-based methods that specific possibly useful metrics are explicitly used to determine vulnerable links, the DRL-based approach is an effect-based method that the reward for the DRL agent is obtained only from the consequence of disruptions (i.e., traffic volume in the road network). However, after proper training, the DRL agents can make decisions that various conditions are comprehensively reflected such as traffic demands, topological singularities, and congestion-induced delays, as shown in Figure 9.

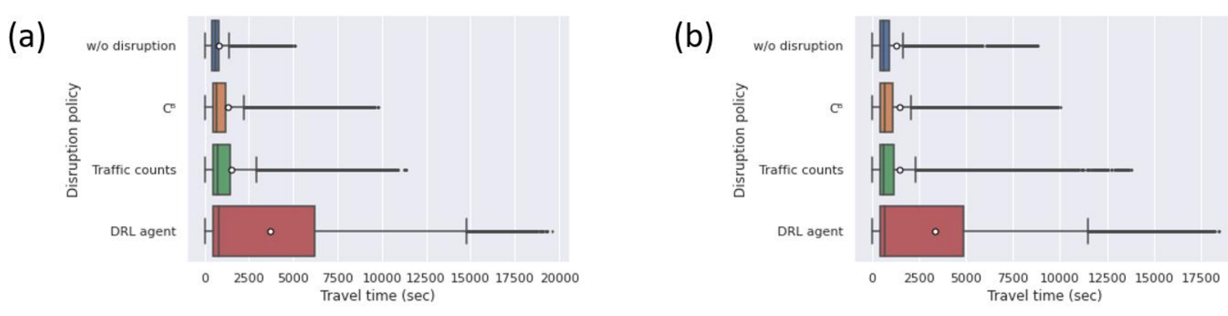

Figure 11. *The travel time of vehicles in the environments with different disruption policies. White marks in the boxplot represent the mean travel time. (a) morning traffic demand (b) evening traffic demand.*

7 CONCLUSION AND FUTURE WORKS

In this thesis, we propose a DRL framework to identify vulnerable combinations of links in road networks. Our proposed method parameterized the policy of the DRL agent with DNNs and train the agent with the experiences from traffic simulations. Contrary to the dominant approaches in this field of study that use brute force simulations with disruptions on all the possible combinations of links, the DRL-based approach takes advantage of the powerful generalization techniques of DRL to extract knowledge from experienced disruptions that can be used for unexplored disruptions using DNNs. This enables the DRL agent to identify disruptions on links that cause significant degradation in the service quality of the road network. The results in our case study demonstrate that the DRL agents can make successful decisions to disrupt critical links considering various conditions in road networks such as traffic demands, topological singularities, and congestion-induced delays.

An interesting future research topic would be training DRL agents in environments with flexible scenarios such as the interval between disruptions is not fixed, the network has elastic traffic demand, or disrupted links can be recovered. For simplicity, we controlled these elements stationary in the environments and do not include them in the observation space. However, we believe DRL agents can potentially learn in more complex and flexible environments in which these elements are variable and included in the observation space so that a general DRL agent can be used for various settings rather than training an individual DRL agent for each different environment.

REFERENCES

[1] Y. Lin, P. Wang and M. Ma, "Intelligent transportation system (ITS): Concept, challenge and opportunity," in *2017 ieee 3rd international conference on big data security on cloud (bigdatasecurity), ieee international conference on high performance and smart computing (hpsc), and ieee international conference on intelligent data and security (ids)*, 2017.

[2] D. Hahn, A. Munir and V. Behzadan, "Security and privacy issues in intelligent transportation systems: Classification and challenges," *IEEE Intelligent Transportation Systems Magazine* , vol. 13.1, pp. 181-196, 2019.

[3] E. Jenelius and L.-G. Mattsson, "Road network vulnerability analysis: Conceptualization, implementation and application," *Computers, environment and urban systems*, vol. 49, pp. 136-147, 2015.

[4] E. Jenelius, T. Petersen and L.-G. Mattsson, "Importance and exposure in road network vulnerability analysis," *Transportation Research Part A: Policy and Practice*, vol. 40.7, pp. 537-560, 2006.

[5] Q. Qiang and A. Nagurney, "A unified network performance measure with importance identification and the ranking of network components," *Optimization Letters*, vol. 2.1, pp. 127-142, 2008.

[6] E. L. de Oliveira, L. da Silva Portugal and W. P. Junior, "Determining critical links in a road network: vulnerability and congestion indicators," *Procedia-Social and Behavioral Sciences*, vol. 162, pp. 158-167, 2014.

[7] V. L. Knoop, M. Snelder, H. J. van Zuylen and S. P. Hoogendoorn, "Link-level vulnerability indicators for real-world networks," *Transportation Research Part A:*

Policy and Practice, vol. 16.5, pp. 843-854, 2012.

[8] V. Shekar, L. Fiondella, S. Chatterjee and M. Halappanavar, "Quantitative assessment of transportation network vulnerability with dynamic traffic simulation methods," in *2017 IEEE International Symposium on Technologies for Homeland Security (HST)*, 2017.

[9] D. Z. Wang, H. Liu, W. Szeto and A. H. Chow, "Identification of critical combination of vulnerable links in transportation networks–a global optimisation approach," *Transportmetrica A Transport Science,* vol. 12.4, pp. 346-365, 2016.

[10] S. A. Bagloee and others, "Identifying critical disruption scenarios and a global robustness index tailored to real life road networks," *Transportation research part E: logistics and transportation review,* vol. 98, pp. 60-81, 2017.

[11] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602,* 2013.

[12] OpenAI, *OpenAI Five*, https://blog.openai.com/openai-five/, 2018.

[13] O. Vinyals and others, *AlphaStar: Mastering the Real-Time Strategy Game StarCraft II,* https://deepmind.com/blog/alphastar-mastering-real-time-strategy-game-starcraft-ii/: DeepMind, 2019.

[14] OpenAI, M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray and others, "Learning dexterous in-hand manipulation," *The International Journal of Robotics Research,* vol. 39, no. SAGE Publications Sage UK: London, England, pp. 3--20, 2020.

[15] A. R. Mahmood, D. Korenkevych, G. Vasan, W. Ma and J. Bergstra, "Benchmarking reinforcement learning algorithms on real-world robots," in

Conference on robot learning, 2018.

[16] Y. Zhu, R. Mottaghi, E. Kolve, J. J. Lim, A. Gupta, L. Fei-Fei and A. Farhadi, "Target-driven visual navigation in indoor scenes using deep reinforcement learning," in *2017 IEEE international conference on robotics and automation (ICRA)*, 2017.

[17] S. Levine, C. Finn, T. Darrell and P. Abbeel, "End-to-end training of deep visuomotor policies," *The Journal of Machine Learning Research,* vol. 17, pp. 1334-1373, 2016.

[18] A. Haydari and Y. Yilmaz, "Deep Reinforcement Learning for Intelligent Transportation Systems: A Survey," *IEEE Transactions on Intelligent Transportation Systems,* Vols. vol. 23, no. 1, pp. 11-32, 2022.

[19] S. Einarsson and M. Rausand, "An approach to vulnerability analysis of complex industrial systems," *Risk Analysis,* vol. 18 (5), p. 535–546, 1998.

[20] K. Berdica, "An introduction to road vulnerability: what has been done, is done and should be done," *Transport Policy,* vol. 9, p. 117–127, 2002a.

[21] Å. Holmgren, "Vulnerability analysis of electric power delivery networks," LIcentiate thesis TRITA-LWR LIC 2020, Department of Land and Water Resources Engineering, KTH, Stockholm, 2004.

[22] M. A. Taylor, "Critical infrastructure and transport network vulnerability: developing a method for diagnosis and assessment. In: Nicholson, A., Dantas, A. (Eds.)," in *Proceedings of the Second International Symposium on Transportation Network Reliability (INSTR)*, Christchurch, New Zealand, 2004.

[23] C. Antoniou, J. Barcel, M. Brackstone, H. Celikoglu, B. Ciuffo, V. Punzo, P. Sykes, T. Toledo, P. Vortisch and P. Wagner, "Traffic simulation: case for guidelines,"

2014.

[24] "SAE J3016 (Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles)," SAE International, 2021.

[25] E. W. Dijkstra and others, "A note on two problems in connexion with graphs," *Numerische mathematik*, pp. 269--271, 1959.

[26] J. Lin, W. Yu, N. Zhang, X. Yang and L. Ge, "Data integrity attacks against dynamic route guidance in transportation-based cyber-physical systems: Modeling, analysis, and defense," *IEEE Transactions on Vehicular Technology,* vol. 67.9, pp. 8738-8753, 2018.

[27] M. Waniek, G. Raman, B. AlShebli, J. C.-H. Peng and T. Rahwan, "Traffic networks are vulnerable to disinformation attacks," *Scientific reports,* vol. 11.1, pp. 1-11, 2021.

[28] A. Saric, S. Albinovic, S. Dzebo and M. Pozder, "Volume-delay functions: A review," in *International Symposium on Innovative and Interdisciplinary Applications of Advanced Technologies,* 2018.

[29] Y. Duan, X. Chen, R. Houthooft, J. Schulman and P. Abbeel, "Benchmarking deep reinforcement learning for continuous control," in *International conference on machine learning,* 2016.

[30] J. Schulman, F. Wolski, P. Dhariwal, A. Radford and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint,* vol. arXiv:1707.06347, 2017.

[31] G. Boeing, "OSMnx: New methods for acquiring, constructing, analyzing, and visualizing complex street networks," *Computers, Environment and Urban Systems,* vol. 65, pp. 126-139, 2017.

[32] M. Haklay and P. Weber, "Openstreetmap: User-generated street maps," *IEEE Pervasive computing,* vol. 7.4, pp. 12-18, 2008.

[33] R. Dowling, A. Skabardonis, V. Alexiadis and others, "Traffic analysis toolbox, volume III: Guidelines for applying traffic microsimulation modeling software," United States. Federal Highway Administration. Office of Operations, 2004.

[34] Y. Zhang, X. Wang, P. Zeng and X. Chen, "Centrality characteristics of road network patterns of traffic analysis zones," *Transportation research record,* vol. 2256.1, pp. 16-24, 2011.

[35] K. Park and A. Yilmaz, "A social network analysis approach to analyze road networks," in *ASPRS Annual Conference,* San Diego, CA, 2010.

[36] S. Porta, P. Crucitti and V. Latora, "The network analysis of urban streets: a primal approach," *Environment and Planning B: planning and design,* vol. 33.5, pp. 705-725, 2006.

[37] P. Crucitti, V. Latora and S. Porta, "Centrality measures in spatial networks of urban streets," *Physical Review E,* vol. 77.3, 2006.

[38] M. Girvan and M. E. Newman, "Community structure in social and biological networks," *Proceedings of the national academy of sciences,* vol. 99.12, pp. 7821-7826, 2002.