Transportation Research Record
2020, Vol. XX(X) 2–16
©National Academy of Sciences:
Transportation Research Board 2020
Article reuse guidelines:
sagepub.com/journals-permissions
DOI: 10.1177/ToBeAssigned
journals.sagepub.com/home/trr
SAGE

Traffic Management of Autonomous Vehicles using Policy Based Deep Reinforcement Learning and Intelligent Routing

Anum Mushtaq[1], Irfan ul Haq[1], Muhammad Azeem Sarwar[1], Asifullah Khan[1] and Omair Shafiq[2]

Abstract

Deep Reinforcement Learning (DRL) uses diverse, unstructured data and makes RL capable of learning complex policies in high dimensional environments. Intelligent Transportation System (ITS) based on Autonomous Vehicles (AVs) offers an excellent playground for policy-based DRL. Deep learning architectures solve computational challenges of traditional algorithms while helping in real-world adoption and deployment of AVs. One of the main challenges in AVs implementation is that it can worsen traffic congestion on roads if not reliably and efficiently managed. Considering each vehicle's holistic effect and using efficient and reliable techniques could genuinely help optimise traffic flow management and congestion reduction. For this purpose, we proposed a intelligent traffic control system that deals with complex traffic congestion scenarios at intersections and behind the intersections. We proposed a DRL-based signal control system that dynamically adjusts traffic signals according to the current congestion situation on intersections. To deal with the congestion on roads behind the intersection, we used re-routing technique to load balance the vehicles on road networks. To achieve the actual benefits of the proposed approach, we break down the data silos and use all the data coming from sensors, detectors, vehicles and roads in combination to achieve sustainable results. We used SUMO micro-simulator for our simulations. The significance of our proposed approach is manifested from the results.

Introduction

Continuous progress in the automotive industry has made fully automated systems a reality that can handle the whole driving process independently without requiring any human interruption. Recent advancements in cutting edge technology promise to help in the prevention of accidents by avoiding unsafe lane changes, drifting into adjacent lanes, generating warnings to vehicles while backing up, and automatic braking systems when a vehicle suddenly stops or slows down (*1*). A combination of advanced software and hardware technologies are used in these vehicles to achieve the desired output. Fully autonomous vehicles advance from level 0 (fully controlled by driver, no automation) to level 5 (no driver presents, full automation) (*2*) (*3*). The level 5 vehicles require a complex set of algorithms for different functionalities such as path planning, scene recognition (*4*), object recognition and tracking etc.(*5*), (*6*).

Today there are many severe challenges faced by the transportation industry, one of them is the high density of vehicles in critical areas that leads to traffic congestion which is directly associated with an increase in carbon dioxide emissions, unwanted delays, noise, parking issues (*7*), accidents and unnecessary fuel consumption (*8*). These situations become further complicated when there are vehicles with no human drivers present inside. To address these concerns, there is a dire need for efficient solutions for traffic management and control. These solutions should be capable of generating benefits such as reducing delays, travel times, congestion management, and environmental pollution (*9*). In the context of AVs, it is imperative to carefully analyze traffic demand and predict future traffic conditions so that more accurate and effective route optimization is done. These methods could be very effective for mitigating the adverse effects of traffic congestion and enabling city management entities by improving traffic flow.

There have been many AI techniques which have been used in the last few years for traffic flow and control

[1] Pakistan Institute of Engineering and Applied Sciences, Islambad, Pakistan
[2] School of Information Technology Carleton University, Ottawa, ON,CANADA

Corresponding author:
Irfan ul Haq, irfanulhaq@pieas.edu.pk

(*10*), (*11*), (*12*). However, Reinforcement Learning (RL) has caught the particular interest of the research community working on autonomous vehicles. The most important quality of RL is beguiling– a method of dealing agents by reward and punishment schemes without going into details of how results will be achieved (*13*). DRL is the extension of the RL that is poised to revolutionize the artificial intelligence industry. DRL plays a significant role in building autonomous systems and previously intractable scale problems. The algorithms of DRL provide a higher-level understanding of autonomous systems allowing control policies to be directly learned by the inputs of the real-world (*14*). DRL algorithms are beneficial to solve the traffic flow problems such as congestion, delays and unnecessary emissions by increasing the efficiency of existing infrastructure systems. One of the essential systems to improve is the Traffic Signal Controllers (TSC) because one of the most critical areas of congestion is the traffic signals, where a situation could worsen if not properly managed, especially in AVs.

In this paper, we focused on the development of intelligent TSC on the intersections by taking advantage of recent AI advancements using DRL techniques for reducing congestion, minimizing delays, and reducing queue lengths. As the road intersections are meant to disperse traffic flow, they can easily lead to obstacles in traffic flow and cause accidents. Statistics reveal that more than one-third of total traffic delays are due to delays on the intersections and thus lead to more than 50 per cent of the total accidents (*15*). Analyzing intersections, exploring efficient methodologies for traffic management on intersections and developing intelligent traffic signal controls have immense significance. Taking these measures can improve traffic conditions by increasing traffic efficiency while ensuring traffic safety. To address these concerns, an intelligent approach is proposed in this paper for improving traffic flow on intersections and balancing traffic while minimizing delays. Following are the contributions of this paper:

- We are proposing a DRL-based framework for traffic signal management in complex traffic environments. A smart traffic signal controller takes inputs from various sensors and is trained using deep reinforcement learning. The intelligent signal optimally selects the appropriate signal sequence that should be enacted according to the current state of intersection in the dynamically changing complex simulation environment.

- We also propose a technique for load balancing the traffic by rerouting the vehicles on the road network to alternate paths. On the one hand, congestion at the intersection is controlled by intelligent signal whereas on the other hand traffic coming behind the intersections is rerouted to other paths between their Origin-Destinations to reduce the congestion at the intersection and behind the intersection, thus smoothing traffic flow.

We used SUMO (a tool for traffic simulations) (*16*) for simulating the road networks, traffic generation and the infrastructure. The organization of next sections is as follows: In Section II we showed the related work in the current area. In Section III, we give a detailed explanation of our proposed approach. In Section IV experimental setup is presented while Section V presents the results and detailed discussions on them. Finally Section VI concludes the paper.

Literature Review

This study constitutes different fields of study including reinforcement learning, autonomous vehicles, routing and traffic flow. Earlier studies on traffic control using RL were limited due to the lack of computational power and simple simulations (*17*), (*18*), (*19*), (*20*). Continuous improvements have been made in this area in early 21st century and variety of realistic and complex simulations have been developed. Recently many AI techniques have been used in traffic signal controls such as Deep Q-learning [(*21*), (*22*)], Q-learning (*23*), (*24*), and fuzzy (*25*). Traffic researchers use micro-simulators which are the most popular tools for producing real world behaviour of traffic and model distinct entities for individual vehicles. The type of reinforcement learning, reward definition, state/action space definition, traffic simulator, vehicle generation model and traffic network geometry used is different in different studies. In previous studies state space is defined as some traffic's attribute and the most popular attributes are traffic flow (*26*), (*27*) and number of queued vehicles (*20*), (*18*), (*28*), (*29*). All available signal phases define the action space (*27*), (*30*) or action space could be restricted to only green phase (*26*), (*29*), (*28*). Change in queued vehicles and change in delay are most common definitions used for rewards (*26*), (*29*), (*28*). A comprehensive review of traffic signals controlled by reinforcement learning is given in (*31*) and (*32*).

Technological advances in this area ensure efficient transportation systems, and on an unprecedented scale, a large amount of varied data could be collected. This high-quality data could be used with minimal abstraction as in (*33*) authors proposed a control system for traffic lights that make use of this type of data using deep reinforcement learning. They proposed a discrete state encoding, which input the deep convolutional neural net and new state space. Q-learning algorithm is used with experience replay. The cumulative delay, average travel time and average queue length decreased effectively. They used traffic simulator SUMO for their simulations. In (*34*) authors investigate the optimized learning policies for traffic signals. They control traffic light problems by combining the coordinated algorithm with the Q-learning algorithm and giving a new reward function. Their approach reduces the travel time effectively and minimizes the possible causes of instability in reinforcement learning. In (*25*), fuzzy logic is used to calculate the traffic light's optimum extension time. Two sensors are used to identify traffic flow and extend the time membership function used by the fuzzy light controller. In the fuzzy model a dynamic environment is analyzed, and model regularly change itself to adapt this environment. Changing the model repeatedly and applying it to adjust with the environment consumes huge amount of processing power and eventually system's performance degrades.

In (*29*), traffic congestion is reduced by minimizing queue length, and fixed signals are used for turning signal green. The duration of green time is extended based on the queue lengths. Its performance proves better than fully fixed signals. In (*35*) different reward functions are used for comparing the performance of agents. The authors used real-world constraints to simulate a junction such as realistic controllers, sensor inputs, green times, and calibrated demand and stage sequencing. Their reward time was based on queue length, time spent, junction throughput, average speed, lost time etc., and their performance measure depended on average waiting time. Their results showed that, across all demand levels, speed maximization causes the lowest average waiting time. In (*36*) authors proposed a traffic control strategy using a deep reinforcement learning algorithm. Their algorithm tunes parameters and relaxes the need for fixed traffic demand assumptions.

Managing traffic flow is an essential part of the Intelligent Transportation System (ITS), but many challenges need to be addressed. Because of the variable and complex traffic dynamics, model-based control methods do not perform well. Model-free data-driven techniques such as RL are more suitable methods for such situations. The DRL method is good for signal management, but for the efficient traffic flow and congestion reduction, only traffic signal controllers are insufficient. All the studies mentioned above only focus on traffic signal controls; however, there is a need to use some hybrid techniques to maintain the overall traffic flow. Using a combination of techniques, the congestion on the intersection and the whole network could be managed efficiently, and a better traffic management system could be developed. To achieve this goal, in our previous study (*37*), we proposed a technique using value-based reinforcement learning and smart rerouting, which showed very effective results. In the current study, we used policy-based DRL methodology to balance the traffic on the road infrastructure.

Proposed Approach

AVs are a long-standing goal of AI, and navigation of these vehicles requires highly efficient models and infrastructure. Although there have been many developments in this area for the last two decades, there is still a considerable need for intelligent systems to replace experienced human drivers. Successful navigation of autonomous vehicles from one point to another is not sufficient enough for the AVs; instead, we also would require intelligent algorithms, smart infrastructure and proper congestion management techniques in future. We present an intelligent policy iteration based DRL algorithm for traffic signal control. This intelligent algorithm dynamically control signals based on the current situation at the intersection and resolves congestion. It reduces the long wait times, long queues and delays at the intersections, thus making traffic lights efficient and resolving congestion. We then applied the rerouting technique to the traffic behind the intersection. The vehicles coming behind that have yet not reached the intersection do not have to stuck and delayed due to the congested intersection. By checking the traffic congestion on the intersection, vehicles behind the intersections are rerouted to other routes based on their origin-destination matrix if there are vehicles more than the threshold limit. The graphical representation of our proposed approach is shown in Fig 1.

States

According to our approach, the traffic signal phase, vehicles speed and position represent states. The current state of the vehicles could be obtained by the V2I communication using sensors deployed on the road network . We divided the road into small segments, and each segment has some sensors embedded on it, which returns true in the presence of the vehicle and false in its absence.

Actions

Actions are the set of functions performed by the traffic lights. Traffic light becomes green for those vehicles whose waiting time and queue length are greater than others. Actions are performed by looking at the current state of the intersection dynamically.

Reward

One of the significant and most important part of the RL is reward as it shows the output of the specific action in a specific state. The subsequent actions should be selected very carefully as they are dependent on the output of the reward function. Reward contain normative content, stipulating the goal of the agent and could be negative or positive. In our approach, we used vehicle's waiting time as a reward in a specific lane. Waiting time of each newly entered vehicle is recorded and total time of all the vehicles in a specific lane is added to calculate commutative waiting time. The vehicle that leaves the lane, its waiting time is not added to the total delay time. Our global objective is the overall improvement and management of traffic flow and congestion reduction, and to achieve this, we use a local objective function. The reward is calculated as old waiting time - new waiting time. Therefore, the reward will be negative if the current waiting time is more than the previous and the reward will be positive

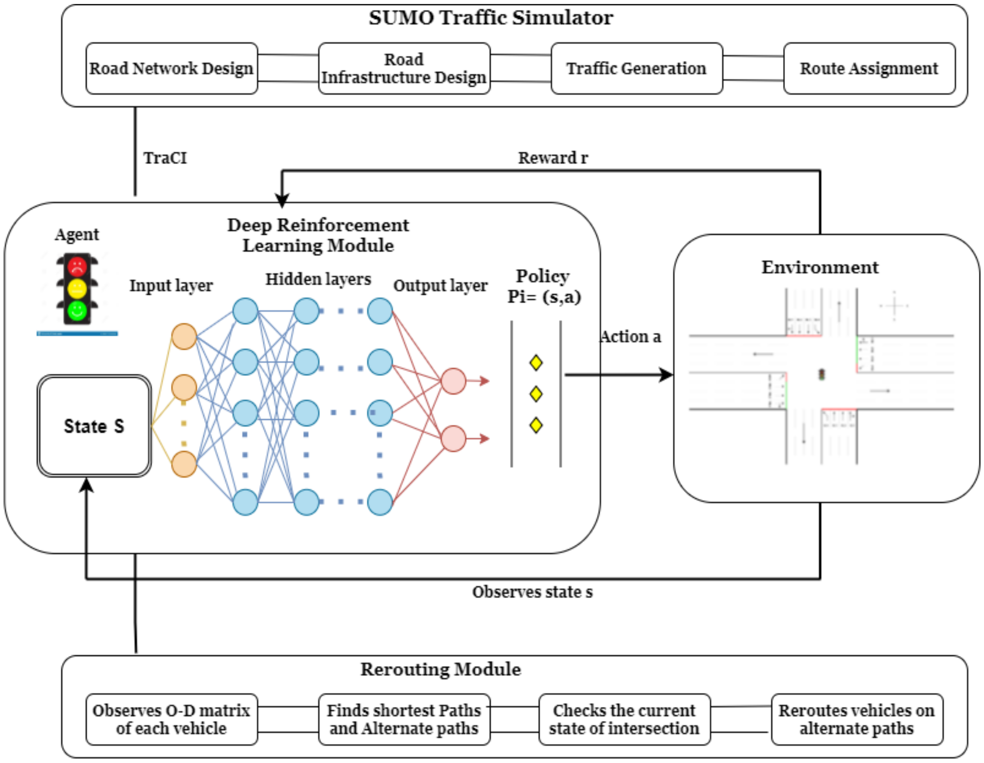

SUMO Traffic Simulator
Road Network Design
Road Infrastructure Design
Traffic Generation
Route Assignment
TraCI
Reward r
Agent
Deep Reinforcement Learning Module
Input layer
Hidden layers
Output layer
Policy Pi= (s,a)
Action a
Environment
State S
Observes state s
Rerouting Module
Observes O-D matrix of each vehicle
Finds shortest Paths and Alternate paths
Checks the current state of intersection
Reroutes vehicles on alternate paths

Figure 1. Intelligent Traffic Signal Management and rerouting [Proposed Approach]

if the current waiting time is less than the previous waiting time.

In our proposed approach, we modelled our problem as MDP, where agents interact with the environment, learns from it, and based on its learning it take decisions. The objective is to maximize the performance by learning optimal policies that are best suitable at that instance of time. We could represent an MDP as a set S, A, T, γ, I, R here S_t are the states, A_t represents the actions, T represents the transition function that shows the probability of transition between states,γ is discount factor, I is the initial state's distribution and R_t represents the reward. if the reward and probability are unknown than MDP can be treated as RL's problem. In RL environment, an agent in state $s_{t_i} \in S_t$, at each time t_i, takes an action $a_{t_i} \in A_t$ depending on observation and then transition function T leads it to new state $s_{t_i+1} \in S_t$. The reward $r_{t_1} \in R_t$ could be maximized by learning an optimal policy $\pi : S_t \times A_t \rightarrow [0, 1]$. The probability of selecting an action in any state represents its optimal policy. The expected cumulative discounted future reward can be calculated as

$$R_t = E\left[\sum_{k=0}^{\infty} \gamma^k \, r_{t+k}\right] \qquad (1)$$

Here γ is used as trade-off between exploration vs exploitation.

If x denotes the result on the output layer than we can say that

$$f(x) = \begin{cases} x, & \text{if } x \geq 1 \\ 0, & \text{otherwise} \end{cases}$$

In RL learning, the optimal policy is beneficial for continuous and high dimensional spaces, which require very high computational power and memory. Defining policies contain a manageable set of parameters that are less consuming and efficient. In our case, the problem is to optimize traffic signals intelligently to determine what action to take at state (s) to maximize reward. This objective could be obtained by fine-tuning its vectors of parameters θ_i and selecting the best action for policy π.

$$\pi(a_i|s_i, \theta_i) = P_r\{A_t = a_i|S_t = s_i,\ \theta_t = \theta_i\} \qquad (2)$$

It can be defined as the probability of an action a_i at state s_i is the policy π with θ_i parameter.

System training and objective function

There could be several objective functions used while managing traffic optimization problems like balancing queue lengths, increasing throughput, minimizing the number of vehicles at intersections etc. Minimizing the delay is the main objective of our agent, that is directly proportional to maximizing traffic throughput and minimizing queue length

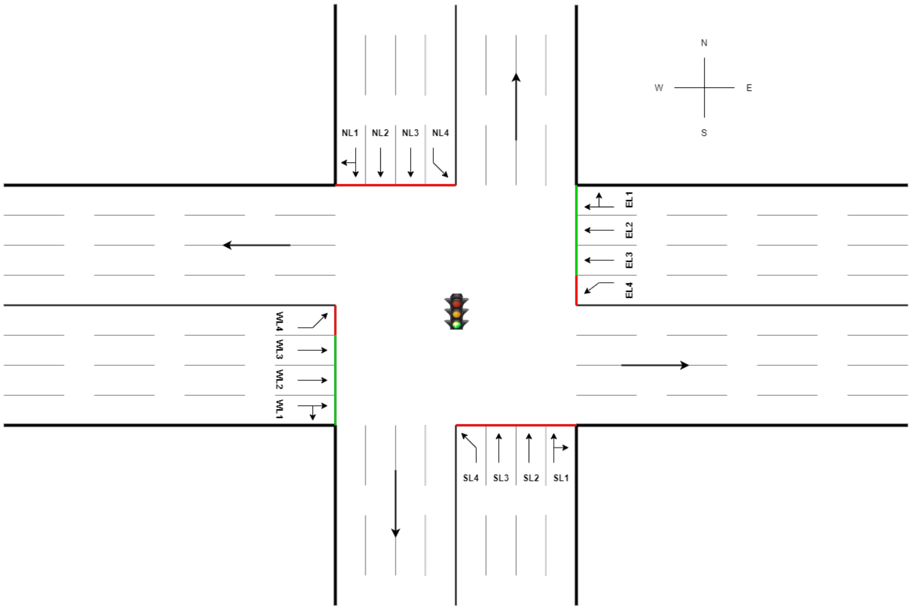


Figure 2. Traffic light Intersection used for simulation

in this research. To maximize reward based on θ_i, we will define our objective function as:

$$J(\theta_i) \doteq V_{\pi\theta i}(s_0) \tag{3}$$

Where $V_{\pi\theta i}$ is value function, $\pi_{\theta i}$ is policy and s_0 is initial state. It indicates that maximizing $J(\theta_i)$ means maximizing $V_{\pi\theta i}(s_i)$

$$\nabla J(\theta_i) = \nabla V_{\pi\theta i}(s_0) \tag{4}$$

According to the policy gradient theorem (*38*)

$$\nabla J(\theta_i) \propto \sum_{s_i} d(s_i) \sum_{a_i} q_\pi(s_i, a_i) \nabla\pi(a_i|s_i, \theta_i) \tag{5}$$

Here $d(s_i)$ is the distribution under π which means the probability of being at state s_i while following policy π. Where $q(s_i, a_i)$ is the action value function under π, and $\nabla\pi(a_i|s_i, \theta_i)$ is gradient of π given state and θ_i.

As the objective is to maximize the reward, so for every episode T, starting from 1 to the maximum number of episodes, we define $J(\theta_i)$ as the expected sum of rewards $R_t(s_i, a_i)$, following the policy $\pi(\theta_i)$. So if we take episode as trajectory τ from 1 to T, then the expected reward is the summation of all the trajectories of the probability as per θ_i, τ times the return of this trajectory $R_t(\tau)$.

$$J(\theta_i) = E\left[\sum_{t=0}^{T} R_t\,(s_i\,, a_i)\,; \pi_{\theta_i}\right] = \sum \tau\; P(\tau\,;\,\theta_i)\; R_t(\tau) \tag{6}$$

Algorithm 1 Training of Neural Network based on policy gradient method

```
Initialize parameters i.e. $\pi(s_i|\theta_i^\pi)$ randomly
Initialize step counter 0, $t \leftarrow 0$
Initialize Memory's Reply Buffer RB
Initialize set of sample trajectory policy $S_0, A_0, R_0, S_1, A_1, R_1, ..., S_T$
while Epochs= 1 ; Epochs ¡ Total Epochs ; Epochs++ do
    Start the Simulation counter with 1st Step t
    Initialize observation state $s_0$
    Set $t_{start} = t$
    foreach $t \leftarrow 0$ to $t$ do
        Perform action $a_i = \pi(s_i|_i a^\pi)$
        Analyze reward $r_t$ and next state $s_{i+1}$
        Update transition values $(s_0, a_0, r, s_n)$ in RB
    Sample small batches from RB
    Define Objective function $J(\theta_i) \doteq V_{\pi\theta i}(s_0)$
    Calculate probability for actions at specific state by $\nabla J(\theta_i) \propto \sum_s d(s_i) \sum_a q_\pi(s_i, a_i) \nabla\pi(a_i|s_i, \theta_i)$
    Define expected sum of rewards as $J(\theta_i) = E\left[\sum_{t=0}^{T} R_t\,(s_i\,, a_i)\,; \pi_{\theta_i}\right] = \sum \tau\; P(\tau\,;\,\theta_i)\; R_t(\tau)$
    Maximize reward according to $\underset{\theta_i}{m}ax J(\theta_i) = \underset{\theta_i}{max} \sum \tau\; P(\tau\,;\,\theta_i)\; R_t(\tau)$
    Do optimization as $\nabla_{\theta_i} J(\theta_i) = \frac{1}{m} \sum_{i=1}^{m}\, \sum_{t=0}^{T}\, \nabla_{\theta_i}\, \log \pi_{\theta_i}(a_i\,|\,s_i)\,(Q(s_i\,, a_i) - \underline{V}_{\varnothing}(s_i)$
```

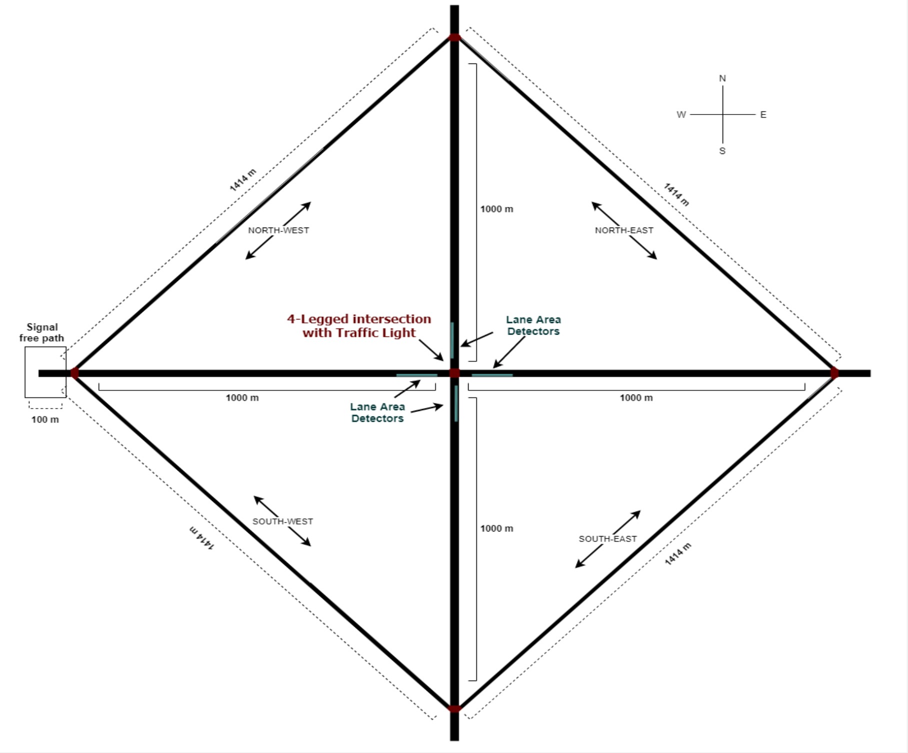


Figure 3. Directions of road network

Here, our aim is to find such parameter set θ_i that maximizes our reward $J(\theta_i)$.

$$\underset{\theta_i}{max} J(\theta_i) = \underset{\theta_i}{max} \sum \tau \; P(\tau\,;\,\theta_i)\; R_t(\tau) \qquad (7)$$

Taking derivative and doing mathematical calculations from policy based gradient, we got the following objective function for our traffic light optimization problem:

$$\nabla_{\theta_i} J(\theta_i) = \frac{1}{m} \sum_{i=1}^{m} \sum_{t=0}^{T} \nabla_{\theta_i} \; \log \pi_{\theta_i}(a_i \,|\, s_i)\; R_t(\tau^i) \qquad (8)$$

In the above equation, $J(\theta_i)$ is the objective function, say minimizing the cumulative wait time of vehicles, m is the total episodes in the simulation, π is the policy dependent on the θ_i such that if we change the θ_i it will affect the policy as well. When vehicles are at the intersection waiting for the light phase to turn green, then the policy indicates the probability of which light phase will be activated in the current state. τ^i is the ith episode, $R_t(\tau^i)$ is the return (total reward) of τ^i, and T is the number of step in τ^i. We can summarize the above equation as the average of all m trajectories is $J(\theta_i)$, and each trajectory is the sum of episodes. At each episode, we take the derivative of the log of the policy and multiply it with reward $R(\tau^i)$. Here, an issue that occurs with $R(\tau^i)$ is that when our traffic light agent knows nothing about the environment at the start of the simulation, it takes random actions that may cause an increment in the negative reward. We want to minimize negative rewards during the learning process, and after many episodes, when our reward becomes 0, the $J(\theta_i)$ will be 0. In this situation, our neural network model will not learn anything new. To avoid this problem, we use discounted rewards.

$$R_t = \sum_{t'}^{T} = t\,\gamma^{t'}\; rt' \qquad (9)$$

Now the equation 8 becomes

$$\nabla_{\theta_i} J(\theta_i) = \frac{1}{m} \sum_{i=1}^{m} \sum_{t=0}^{T} \nabla_{\theta_i} \; \log \pi_{\theta_i}(a_i \,|\, s_i)\; R_t \qquad (10)$$

The R_t in the above equation will return the rewards when we take certain action at step t, but to know which reward value is optimal or which value is not good, we need some baseline or reference point with which we can compare our results. For this purpose, we will take an average of the results of all actions and use it as a reference point. we will use $Q(s_i, a_i)$ that is the value of the specific a_i at s_i, and $V(s_i)$ is the average of all rewards at s_i. Using these values, we can rewrite the above equation as

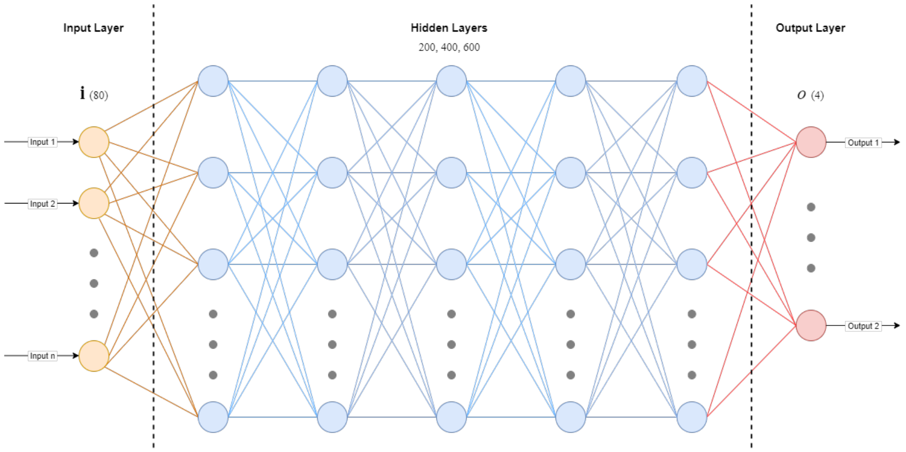


Figure 4. Architecture of Neural Network Model Used

$$\nabla_{\theta_i} J(\theta_i) = \frac{1}{m} \sum_{i=1}^{m} \sum_{t=0}^{T} \nabla_{\theta_i} \log \pi_{\theta_i}(a_i \,|\, s_i) \left(Q(s_i\,, a_i) - V_{\varnothing}(s_i)\right) \tag{11}$$

In the above equation, $\pi(a_i|s_i)$ tells us which light phase to activate and $Q(s_i, a_i) - V(s_i)$ tells us how good our action was.

Rerouting of vehicles

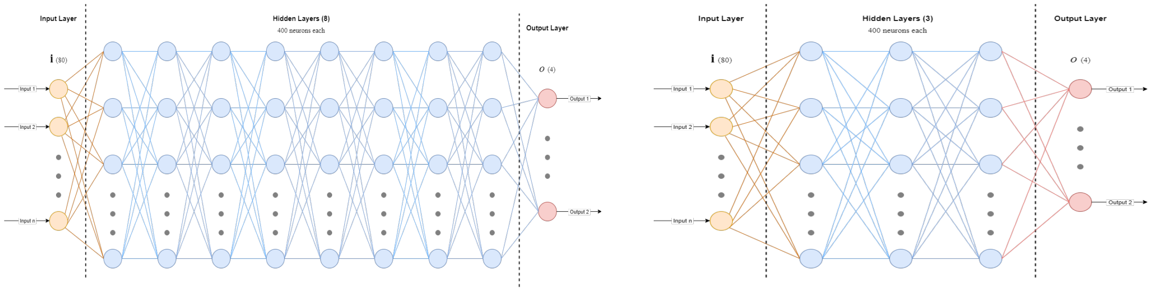
Traffic on the intersection is managed by RL based intelligent traffic lights that reduce the waiting times of vehicles at the intersection thus reducing long queue lengths. The main purpose of this paper is to optimize the flow of traffic while minimizing the delay. To optimize traffic flow, Vehicles that are stuck in the congestion in long queues behind the intersection are rerouted to less congested paths to load-balance the traffic on the intersection during peak hours. These alternate paths are computed from the perspective of each vehicle's Origin-Destination (OD). Lane area detectors are used to detect the traffic information on a specific road segment. The traffic data from these detectors is maintained in a separate log file and analyzed every thirty(s) and a critical density limit is set. When the density of vehicles exceeds that threshold limit, the vehicles on the road segment behind the intersection are rerouted to the alternate path leading to the vehicle's destination. The detectors give information about the direction, number of vehicles and their speed on the road segment. In this way, traffic is managed well at the intersection and behind the intersection, providing a solid metaphor for resolve congestion and optimize routes.

Algorithm 2 Vehicle's Rerouting

```
Initialize the Memory parameter M
Set vehicles with their IDs such as
$V_i = [v_1, v_2, v_3, ...., v_n]$
Calculate O-D against each vehicle ID
Find all Routes [Ri] against each vehicle O-D i.e
$R_i = [r_1, r_2, r_3, ...., r_n]$
Calculate shortest Routes using Dijkstra algorithm
Sort the routes in ascending order
Get the vehicle's density at intersection
Compute each route's Travel time $(T_{time})$
Store $T_{time}$ in route file
foreach $V_i at Intersection I$ do
    Observe the current state of I
    Calculate Total-Wait-Time $(T_{wt})$ at I
    Add the $T_{wt}$ to the $T_{time}$ of shortest path
    Calculate Updated-Total-Wait-Time $(U_{twt})$
    if $U_{twt} > T_{time}$ of alternate routes then
    Reroute Vehicles to the alternate routes ;
    else if $(U_{twt} \leq alternate\, routes)$ then
        Stay on the same route
```

Simulation Setup

The detail of simulation experiments is discussed in this section.



((a)) 8 Layered Neural Network Model

((b)) 3 Layered Neural Network Model

Figure 5. Comparisons of Neural Networks: Deep vs Shallow

Road Network Used

The length of the road network is shown in Fig.3, that is 1100 meters that we used in our simulations. In the road network, multiple paths are used to reach from one point to other in order to simulate the rerouting of the vehicles in case of congestion. In Fig.2, we can see the intersection used for traffic signal control. We used one route containing traffic intersection with traffic lights while all other routes are signal-free. The length of the road network from left, right, north and south towards the intersection are 1100 (1000+ 100) meters, while the road on diagonals is 1414 meters on each side. These diagonals are used as an alternate path which is mostly signal free. The roads connected to the intersection contain four lanes (4 incoming, 4 outgoings); two middle lanes for vehicles that move towards the left and right lane for vehicles that go towards left and right.

Route Assignment in Network

We used the following two methods for assigning routes to the vehicles.

1. Static route assignment: Static assignment is done before starting the simulation. These are initial fixed routes in which 60% of vehicles will use straight paths through the intersection, and the remaining 40% will use signal-free routes based on directions that are normally lengthier than signalled routes.

2. Dynamic route assignment: Routes are assigned dynamically using TraCI in SUMO, analysing the current state of the traffic. If vehicles' density crosses the threshold limit on certain areas, then dynamically, vehicles are rerouted towards alternate routes.

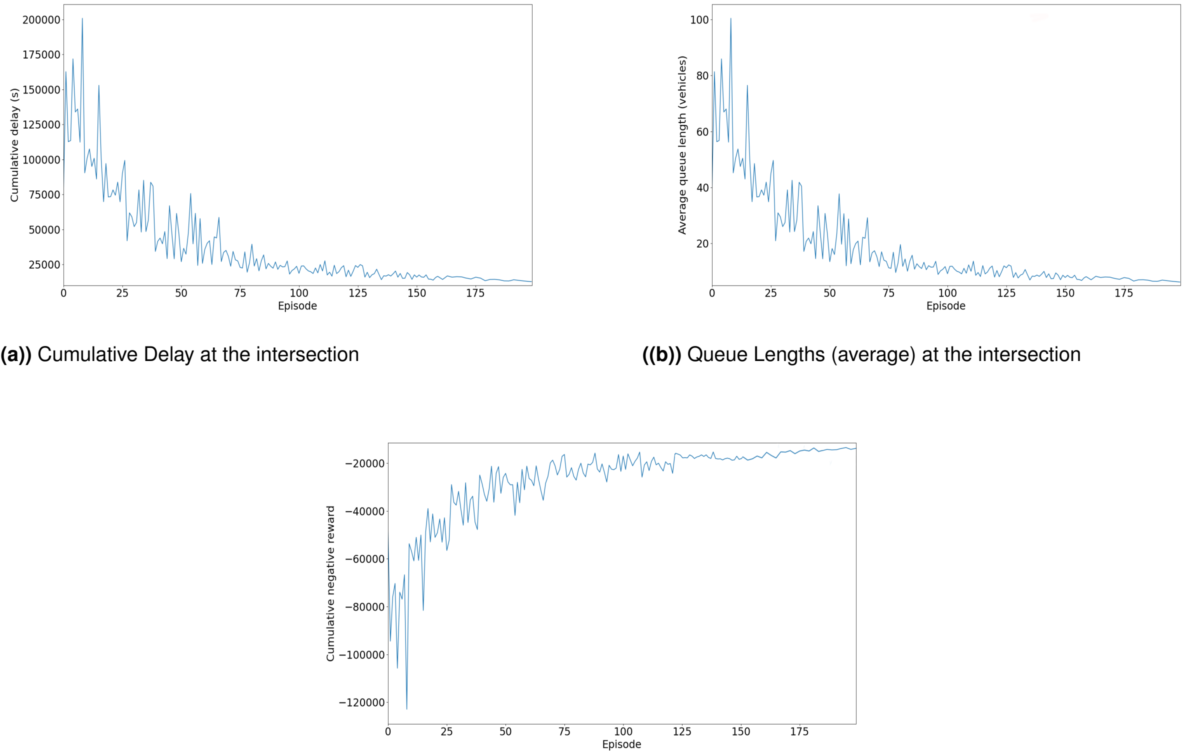
Detectors used

We used four detectors on each lane's side to get the current information of vehicles on the road. The total number of detectors used are 16 for our experiments, and after every 30 seconds, the data of detectors are analyzed. Detectors are placed on south, east, west, and north, and the data of these detectors is stored separately in a log file of each side. When vehicles on the road exceed certain threshold limit, these detectors send a warning message to vehicles coming from the roads, behind the intersection, to wait or change their route according to the rerouting strategy. The direction and speed, along with vehicle's number determined by these detectors. We used this information to shift traffic to other routes where traffic is low in volume.

Simulation Parameters	
Parameter	Value
Transmission Range	1100 (m)
Simulation Time	9000 (s)
Number of Vehicles per Episode	4000
Simulation Time for each vehicle	39 (min)
Simulation Map	4 leg intersection
Number of traffic lights	01
Simulator	SUMO

Traffic generation

We used almost 4000 different types of vehicles for our simulations, such as cars, trailers, buses and ambulances having varying speeds. Each vehicle has its O-D matrix, based on which route is assigned to vehicles. At the start, all vehicles are assigned the shortest route from their origin to destination. However, if the shortest route is congested, these vehicles could be rerouted to alternate paths from their origin to destination, which could be longer but faster. Our simulation used our intersection route as the shortest path for



((a)) Cumulative Delay at the intersection

((b)) Queue Lengths (average) at the intersection

((c)) Cumulative Negative Reward

Figure 6. Results of average delay, queue lengths and rewards for NN1

60% of the vehicles and every vehicle has to take the route containing intersection while other 40% vehicles depend on the intersection's situation. If the intersection is congested, then these vehicles are shifted towards an alternate route. In our proposed approach, 60% traffic is controlled using intelligent traffic signals based on reinforcement learning. In contrast, the remaining 40% of vehicles are managed by rerouting to alternate paths. These alternate paths may be longer than the shortest route but could be the fastest routes depending on the situation.

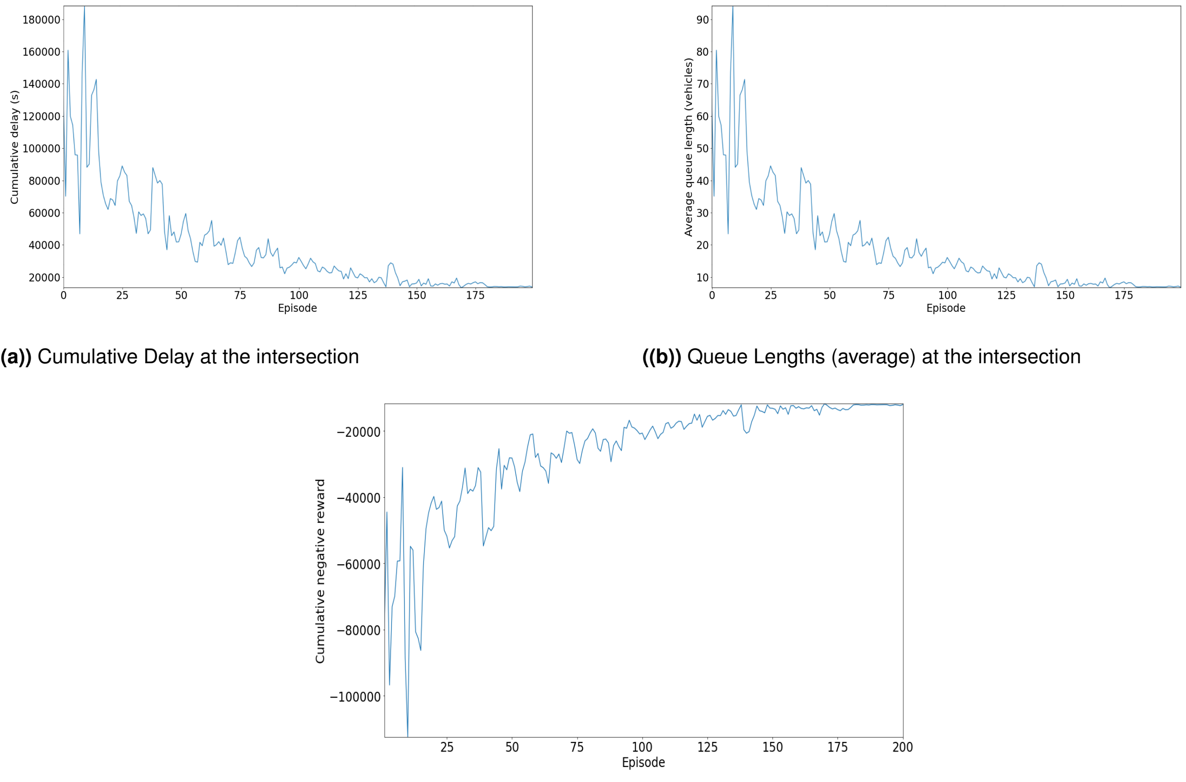
Sensors used in network

Different types of sensors are used for getting the current traffic situation on the roads. We used 80 sensors embedded on different points on lanes for incoming traffic. Each side of the four-legged intersection has 20 sensors, among which 10 sensors are used on three lanes, i.e. one right lane and two straight lanes, because these lanes use the same signal. The left lane uses the remaining 10 sensors. The sensors are deployed at appropriate distance from each other, however the sensors' density is higher near the traffic signal. Each sensor has a unique id and provides information about the position of vehicles according to the lanes. A sensor transmits information when vehicles come in its vicinity. The information from these sensors is used to track the vehicles that are heading towards the intersection.

Training Parameters	
Parameter	Value
Total Episodes	200
Batch Size	200
Memory Size	4500
Number of Actions	4
GUI	True
Green phase time	4(s)
Yellow phase time	2(s)
Gamma Value	0.50
Number of states	80
Maximum Step	2500

Performance Parameters

The parameters used for performance evaluation are as follows:



((a)) Cumulative Delay at the intersection

((b)) Queue Lengths (average) at the intersection

((c)) Cumulative Negative Reward

Figure 7. Results of average delay, queue lengths and rewards for NN2

Reward The first and the most important parameter used for the performance evaluation is the reward function. Two types of rewards could be uses, i.e. positive and negative. The agent aims to maximize the positive reward or minimize the negative reward. In the presented study, we are minimizing the negative reward and evaluating results based on that. The negative reward tending towards zero means that our agent is learning gradually.

Delay During peak hours, when the intersection is crowded with heavy traffic and vehicles get stuck in jams, delay increases. After applying our approach, the cumulative delay starts to decrease, which shows improved performance.

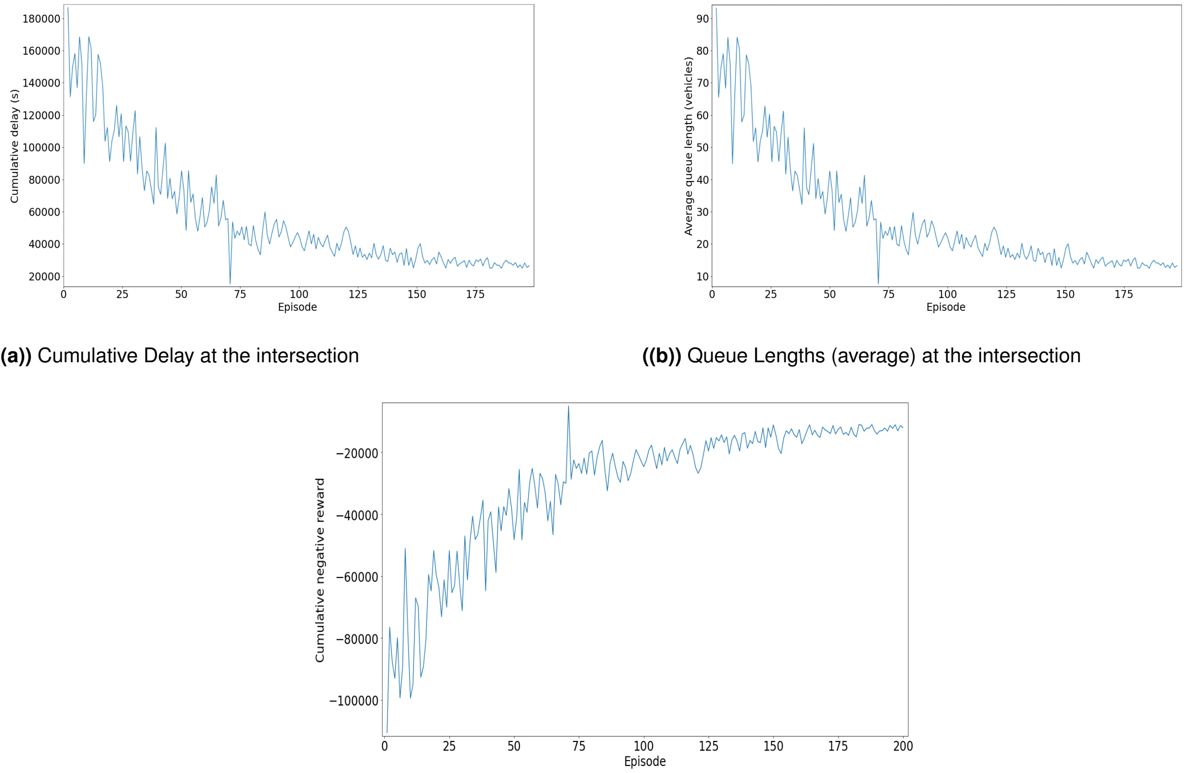
Queue Length The queue length is another critical parameter that shows congestion during high traffic peaks. The long vehicle's queues shows the critical situation at the intersection. The performance parameter used is named as ”average queue length” and performance is measured as, when queue lengths start decreasing, it means performance is increasing.

Simulation Time Another critical parameter is the simulation time for all episodes. The simulation stops as the last car leaves the road segment. The maximum time for simulation is 9000(s), and this time will decrease with the improvement in the performance.

Neural Network's Structure	
Parameter	Value
Neural network layers	200, 400, 600
Activation Function	Relu
Loss Function	Mean Squared Error
Optimization Function	Sigmoid Activation Function

Results and Discussion

Our results are based on three phases. In first phase, we ran our simulation with the configuration using the conventional setting (pre-timed fixed) of traffic lights and recorded the outputs. The second phase shows the implementation of intelligent traffic lights using DRL for controlling traffic and its outputs. In the third phase, we used rerouting with intelligent traffic lights and recorded the results. Comparisons



((a)) Cumulative Delay at the intersection

((b)) Queue Lengths (average) at the intersection

((c)) Cumulative Negative Reward

Figure 8. Results of average delay, queue lengths and rewards for NN3

are shown at the end of this section. Now we will further explore the results of all these phases in detail.

Using pre-timed fixed signals:

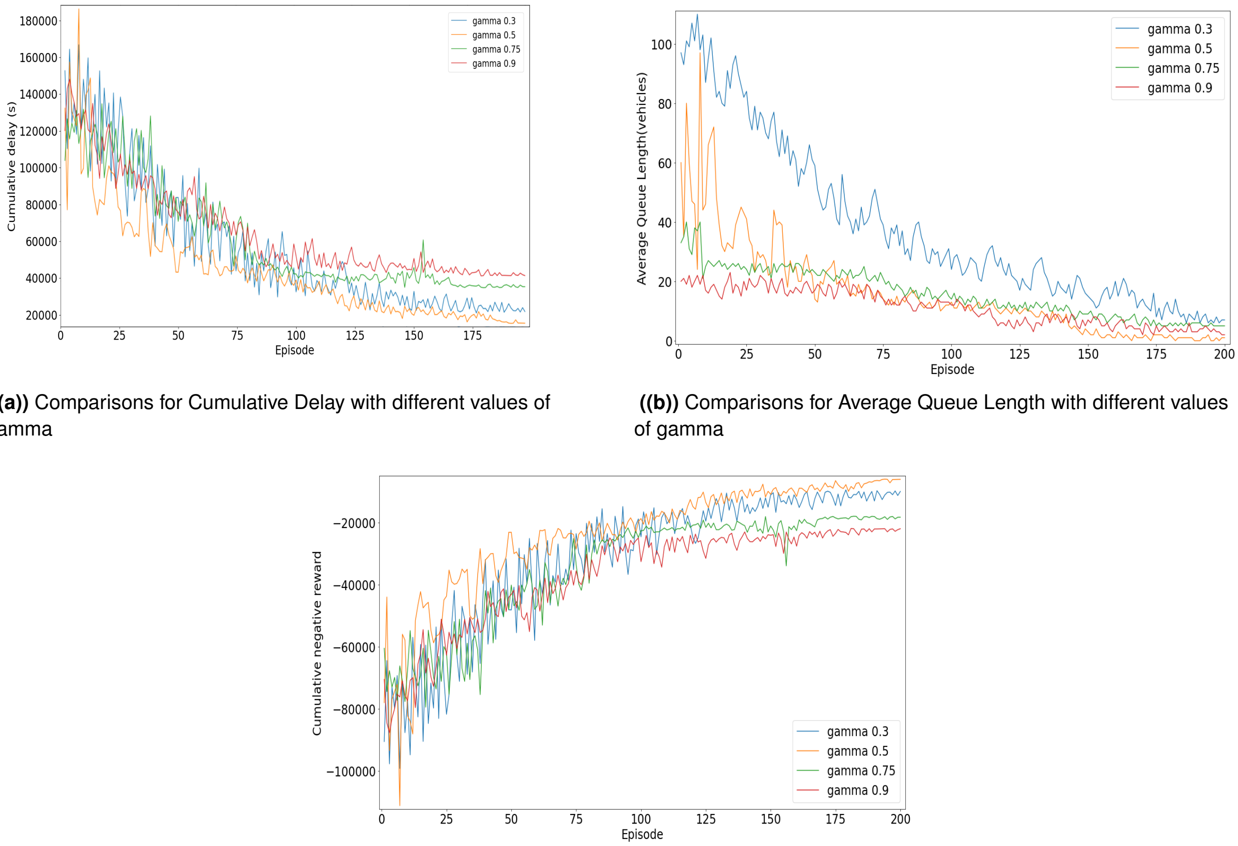
In this phase, we used pre-timed fixed signals for our experiments. Without implementing any intelligent traffic lights, we ran 4000 vehicles in the simulation and observed the traffic conditions. After few seconds, it was observed that vehicles at the intersection and behind the intersection got stuck due to the massive congestion on the intersection. The waiting time of the vehicles and delay increased with longer queue lengths. The system configuration and settings used for all experiments are the same. Results of experiments of 1st phase are shown in the 1st bar of Fig. 10. The green bar shows that 4000 vehicles take almost 7392 (s) to reach the destination due to congestion on the roads and the intersection.

Using RL-based controlled traffic lights

In this phase, we used a policy-based DRL approach for traffic light control. Here, our agent consider the present condition of road's intersection from the data coming from detectors, sensors and data from other infrastructural elements. Then it computes the vehicle's wait time at intersections, queue lengths and takes appropriate action according to the situation. The signal that should be turned green depends on the computed wait time. Our agent measures the average wait time at the intersection and turns the signal on for that lane whose waiting time is more. The main purpose of our agent is to reduce the queue length, negative reward and cumulative delay of the vehicles.

The Fig.4 presents the architecture of Neural Network (NN) we used for our experiments. We used five-layered neural network models with different neurons on hidden layers, i.e. 200, 400 and 600 neurons. The input layer has 80 states, while the output layer has 4 states. The hidden layer of our neural network contains 200 neurons named NN1, NN2 with 400 neurons, and NN3 with 600 neurons.

In all these three models, we can see a general trend that as the agent starts to learn about the environment with the passage of epochs, the commutative delay starts to decrease as well as the queue length. In contrast, the reward starts to increase, i.e. negative reward tends towards zero. In Fig.6(a), Fig.7(a), Fig.8(a) we can observe that cumulative wait time (delay) starts decreasing as number of episodes increases. The traffic rate is the same on all lanes, and the average waiting time is measured at the end of each episode.



((a)) Comparisons for Cumulative Delay with different values of gamma

((b)) Comparisons for Average Queue Length with different values of gamma

((c)) Comparisons for Cumulative Negative Reward with different values of gamma

Figure 9. Results of gamma value comparisons

Similarly in Fig.6(b), Fig.7(b), Fig.8(b) we can see that the average queue length of vehicles also starts decreasing which shows the congestion reduction on the intersection. As the agents learns the environment, it goes on managing traffic signals more properly and efficiently, because of which traffic flow is improved, and queues are reduced. In Fig.6(c), Fig.7(c), Fig.8(c), we can see that total negative reward starts decreasing as agent learns more about the environment. After performing any action, the agent gets a reward (positive or negative), and our policy is updated based on that reward. In our case, we are using the negative reward. The maximum reward obtained shows that this is our optimal policy. In Fig.6, Fig.7, Fig.8, we can see that after 175th episode there is no notable change in reward so we can say that after 175th episode we got our optimal policy.

The red bar in Fig.10 shows almost 20% improvement in results with neural network model 1 when reinforcement learning-based traffic lights are applied on road networks. Similarly, there is 16% and 10% improvements in results while using neural network model 2 and 3 respectively. Here we can notice that when we increased neurons in hidden layers, the performance is degrading, so we can say the neural network model 1 is performing best for our case. There are no hard and fast rules for choosing the hidden layer's neurons; it depends on systematic experimentation.

Using Reinforcement based traffic lights with Rerouting

In this phase of experiments, we used a combination of techniques for traffic congestion management. On traffic intersection, we used deep reinforcement learning-based traffic lights optimization. For the vehicles that are still not reached at intersection, and stuck in the congestion due to the long queue lengths at the Intersection, we rerouted them to alternate paths. This technique helped in two ways: first, the vehicles rerouted to alternate paths could get their destination without waiting for long time at the congested intersection and taking paths that are a bit lengthier but mostly less congested. Secondly, congestion was reduced on the intersection and road behind the intersection due to the rerouting of vehicles as now most vehicles were rerouted towards other paths, and vehicles were not continuously

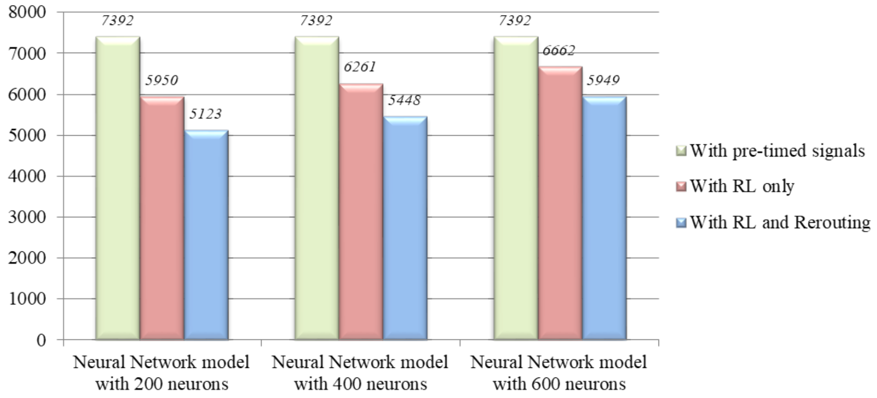


Figure 10. Performance of Proposed Approach

coming towards the intersection. In other words, traffic is load-balanced and divided into many paths that would improve the traffic flow and reduce congestion. In Fig.10, the blue bar shows the improvement of results by 14% using 1st NN, 13% using 2nd NN and 11% using 3rd NN. Thus the total reduction in time by using reinforcement learning and rerouting on NN1 is 34%, NN2 is 29% and NN3 is 21%. The best results are 34% using NN1.

Changing Gamma Value

We want to make some comparisons changing the gamma values on our neural networks and see the effects at this point of experiments. We did experiments on 4 values of gamma i.e. 0.3, 0.5, 0.7 and 0.9. Fig.9 shows the effect of changing these values. From the figure, we can notice that the gamma value 0.5 is better than other values.

Comparisons with Deeper vs Shallower Networks

We made some comparisons in the last set of experiments using some deeper and shallower neural network models. A more deep neural network model has more hidden layers, while in a shallow neural network model, fewer hidden layers are used. In Fig.5(a) we used a deep NN-model with varying hidden layers between 5 to 8, and in Fig.5(b) we used a shallow network with the number of hidden layers reduced from 5 to 3. The reason for these types of experiments is to show how a five-layered NN model is optimized. The red bar in the graph in Fig.11 shows that using a deep model with 8 hidden layers causes only a 3% reduction in waiting time using RL and 13% reduction using rerouting with the same scenario. Similarly, a shallow network using only 3 hidden layers reduces the waiting time by 6% when reinforcement learning is applied on traffic lights, and smart rerouting further reduces it to 13 %. Thus the total simulation time is increased by 16% as we increased the number of hidden layers compared to five layered neural network model that performs best for our system. Similarly, decreasing the number of hidden layers reduces the simulation time by 19% and degrades our performance. These experiments suggest that for our approach, a five-layered neural network model performs the best. As traffic data is dynamic and non-linear, we have to find the best system configurations for our models so we perform different types of experimentation.

Conclusion

In this article, we proposed our approach for congestion management at road intersection and load balancing the vehicles in dynamically changing complex traffic environments. We applied the DRL approach to optimize the intelligent traffic lights capable to take run-time decisions after analyzing the present state of the intersection. For this purpose, We used a deep neural network models with different number of neurons and hidden layers. As a result, we figured out the optimum configuration for our neural network architecture, a five-layered NN-model with 200 neurons performs best for our system. We found that using DRL-based signal controllers improve traffic flow performance on intersections by 20%. We then managed the traffic that is not yet reached at the intersection by rerouting it to alternate paths to avoid the congested intersections. It further improved our performance by 14%. The total improvement in the performance is 34% which increased the efficiency, throughput and reduced the congestion, waiting time, delay and queue lengths of vehicles. The results shows the effectiveness of our proposed strategy to solve the traffic congestion problem while improving the flow management of traffic, especially when all vehicles are on their own without any driver present in them. The approach is equally helpful for regular vehicles and in mixed autonomy. The rerouting approach could be beneficial for emergency vehicles or special vehicles such as VIP vehicles or ambulances.

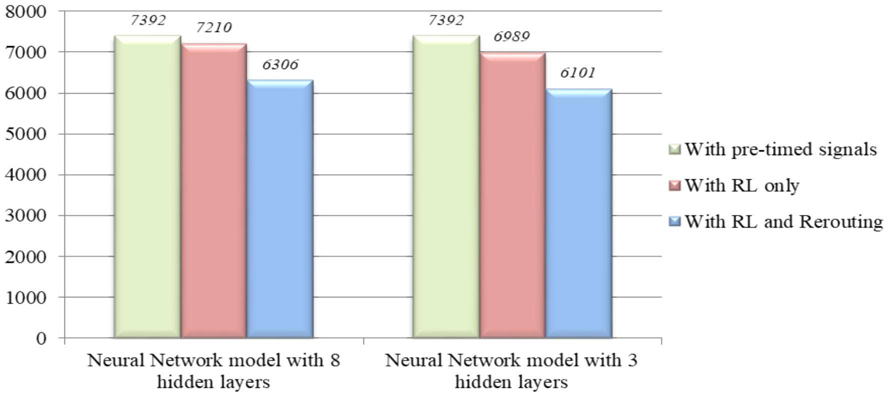


Figure 11. Performance using deep and shallow models

At present, we are also working on extending this approach for more complex traffic scenarios using multi-intersection techniques and real-world road networks to help support the concept of futuristic intelligent cities.

Acknowledgements

This Research is supported by IT Endowment Fund Pakistan Institute of Engineering and Applied Sciences.
Website: `http://www.pieas.edu.pk/`

References

1. Piotr Czech, Katarzyna Turoń, and Jacek Barcik. Autonomous vehicles: basic issues. *Zeszyty Naukowe. Transport/Politechnika Ślaska*, 2018.
2. Marialena Vagia, Aksel A Transeth, and Sigurd A Fjerdingen. A literature review on the levels of automation during the years. what are the different taxonomies that have been proposed? *Applied ergonomics*, 53:190–202, 2016.
3. Peter A Hancock, Illah Nourbakhsh, and Jack Stewart. On the future of transportation in an era of automated and autonomous vehicles. *Proceedings of the National Academy of Sciences*, 116(16):7684–7691, 2019.
4. Shashi D Buluswar and Bruce A Draper. Color machine vision for autonomous vehicles. *Engineering Applications of Artificial Intelligence*, 11(2):245–256, 1998.
5. Shinpei Kato, Eijiro Takeuchi, Yoshio Ishiguro, Yoshiki Ninomiya, Kazuya Takeda, and Tsuyoshi Hamada. An open approach to autonomous vehicles. *IEEE Micro*, 35(6):60–68, 2015.
6. Hironobu Fujiyoshi, Tsubasa Hirakawa, and Takayoshi Yamashita. Deep learning-based image recognition for autonomous driving. *IATSS research*, 43(4):244–252, 2019.
7. F Gómez-Bravo, F Cuesta, and A Ollero. Parallel and diagonal parking in nonholonomic autonomous vehicles. *Engineering applications of artificial intelligence*, 14(4):419–434, 2001.
8. Jorge Luis Zambrano-Martinez, Carlos T Calafate, David Soler, Lenin-Guillermo Lemus-Zúñiga, Juan-Carlos Cano, Pietro Manzoni, and Thierry Gayraud. A centralized route-management solution for autonomous vehicles in urban areas. *Electronics*, 8(7):722, 2019.
9. Jeremy A Carp. Autonomous vehicles: problems and principles for future regulation. *U. Pa. JL & Pub. Aff.*, 4:81, 2018.
10. Elnaz Namazi, Jingyue Li, and Chaoru Lu. Intelligent intersection management systems considering autonomous vehicles: A systematic literature review. *IEEE Access*, 7:91946–91965, 2019.
11. Anum Mushtaq, Asifullah Khan, Omair Shafiq, et al. Traffic flow management of autonomous vehicles using platooning and collision avoidance strategies. *Electronics*, 10(10):1221, 2021.
12. Hamid Khayyam, Bahman Javadi, Mahdi Jalili, and Reza N Jazar. Artificial intelligence and internet of things for autonomous vehicles. In *Nonlinear approaches in engineering applications*, pages 39–68. Springer, 2020.
13. Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4:237–285, 1996.
14. Kai Arulkumaran, Marc Peter Deisenroth, Miles Brundage, and Anil Anthony Bharath. Deep reinforcement learning: A brief survey. *IEEE Signal Processing Magazine*, 34(6):26–38, 2017.
15. Jingfei Yu, Li Wang, and Xiuling Gong. Study on the status evaluation of urban road intersections traffic congestion base on ahp-topsis modal. *Procedia-Social and Behavioral Sciences*, 96:609–616, 2013.
16. Daniel Krajzewicz, Jakob Erdmann, Michael Behrisch, and Laura Bieker. Recent development and applications of sumo-simulation of urban mobility. *International journal on advances in systems and measurements*, 5(3&4), 2012.
17. Thomas L. Thorpe and Charles W. Anderson. Traffic light control using sarsa with three state representations. Technical report, IBM Corporation, 1996.
18. Marco Wiering. Multi-agent reinforcement learning for traffic light control, 2000.

19. Elmar Brockfeld, Robert Barlovic, Andreas Schadschneider, and Michael Schreckenberg. Optimizing traffic lights in a cellular automaton model for city traffic. *Physical review E*, 64(5):056132, 2001.
20. Baher Abdulhai, Rob Pringle, and Grigoris J Karakoulas. Reinforcement learning for true adaptive traffic signal control. *Journal of Transportation Engineering*, 129(3):278–285, 2003.
21. Li Li, Yisheng Lv, and Fei-Yue Wang. Traffic signal timing via deep reinforcement learning. *IEEE/CAA Journal of Automatica Sinica*, 3(3):247–254, 2016.
22. Seyed Sajad Mousavi, Michael Schukat, and Enda Howley. Traffic light control using deep policy-gradient and value-function-based reinforcement learning. *IET Intelligent Transport Systems*, 11(7):417–423, 2017.
23. Yongquan Liao and Xiangjun Cheng. Study on traffic signal control based on q-learning. In *2009 Sixth International Conference on Fuzzy Systems and Knowledge Discovery*, volume 3, pages 581–585. IEEE, 2009.
24. Weirong Liu, Gaorong Qin, Yun He, and Fei Jiang. Distributed cooperative reinforcement learning-based traffic signal control that integrates v2x networks' dynamic clustering. *IEEE transactions on vehicular technology*, 66(10):8667–8681, 2017.
25. IN Askerzade and Mustafa Mahmood. Control the extension time of traffic light in single junction by using fuzzy logic. *International Journal of Electrical & Computer Sciences IJECS–IJENS*, 10(2):48–55, 2010.
26. PG Balaji, X German, and Dipti Srinivasan. Urban traffic signal control using reinforcement learning agents. *IET Intelligent Transport Systems*, 4(3):177–188, 2010.
27. Itamar Arel, Cong Liu, Tom Urbanik, and Airton G Kohls. Reinforcement learning-based multi-agent system for network traffic signal control. *IET Intelligent Transport Systems*, 4(2):128–135, 2010.
28. Monireh Abdoos, Nasser Mozayani, and Ana LC Bazzan. Holonic multi-agent system for traffic signals control. *Engineering Applications of Artificial Intelligence*, 26(5-6):1575–1587, 2013.
29. Yit Kwong Chin, Nurmin Bolong, Aroland Kiring, Soo Siang Yang, and Kenneth Tze Kin Teo. Q-learning based traffic optimization in management of signal timing plan. *International Journal of Simulation, Systems, Science & Technology*, 12(3):29–35, 2011.
30. Samah El-Tantawy, Baher Abdulhai, and Hossam Abdelgawad. Multiagent reinforcement learning for integrated network of adaptive traffic signal controllers (marlin-atsc): Methodology and large-scale application on downtown toronto. *IEEE Transactions on Intelligent Transportation Systems*, 14(3):1140–1150, 2013.
31. Samah El-Tantawy, Baher Abdulhai, and Hossam Abdelgawad. Design of reinforcement learning parameters for seamless application of adaptive traffic signal control. *Journal of Intelligent Transportation Systems*, 18(3):227–245, 2014.
32. Patrick Mannion, Jim Duggan, and Enda Howley. An experimental review of reinforcement learning algorithms for adaptive traffic signal control. *Autonomic road transport support systems*, pages 47–66, 2016.
33. Wade Genders and Saiedeh Razavi. Using a deep reinforcement learning agent for traffic signal control. *arXiv preprint arXiv:1611.01142*, 2016.
34. Elise Van der Pol and Frans A Oliehoek. Coordinated deep reinforcement learners for traffic light control. *Proceedings of Learning, Inference and Control of Multi-Agent Systems (at NIPS 2016)*, 2016.
35. Alvaro Cabrejas-Egea, Shaun Howell, Maksis Knutins, and Colm Connaughton. Assessment of reward functions for reinforcement learning traffic signal control under real-world limitations. *arXiv preprint arXiv:2008.11634*, 2020.
36. Yilun Lin, Xingyuan Dai, Li Li, and Fei-Yue Wang. An efficient deep reinforcement learning model for urban traffic control. *arXiv preprint arXiv:1808.01876*, 2018.
37. Anum Mushtaq, Irfan Ul Haq, Muhammad Usman Imtiaz, Asifullah Khan, and Omair Shafiq. Traffic flow management of autonomous vehicles using deep reinforcement learning and smart rerouting. *IEEE Access*, 9:51005–51019, 2021.
38. Richard S Sutton, David A McAllester, Satinder P Singh, Yishay Mansour, et al. Policy gradient methods for reinforcement learning with function approximation. In *NIPs*, volume 99, pages 1057–1063. Citeseer, 1999.