

Deep Sarsa Replay model for real Time Traffic Control

Ogbeide Oluwatosin Lara¹, Olabode Olatubosun², Agbonifo Oluwatoyin³ and Boyinbode Olutayo⁴

1. Department of Computer science, Adekunle Ajasin University, Akungba Akoko, Ondo state

2. Department of Information systems, Federal University of Technology, Akure, Ondo state

3. Department of Information systems, Federal University of Technology, Akure, Ondo state

4. Department of Information Technology, Federal University of Technology, Akure, Ondo state

Date of Submission: 10-11-2024

Date of Acceptance: 20-11-2024

ABSTRACT

Traffic congestion is a major factor currently affecting the global economy thereby costing billions of dollars in terms of the time lost in traffic, productivity and fuel wastage. Although, various researchers have made concerted effort to proffer solution to the problems of traffic congestion, in addition, recent reinforcement learning research has also presented multiple possible solutions. However, based on the literature, most of the reinforcement learning traffic control agents do not possess the real-time traffic control capabilities to efficiently deal with the complex and ever-changing nature of traffic on an average urban city road. This paper presents a Real Time deep reinforcement learning model for Traffic Control for optimization of vehicle flow (minimizing waiting time and traffic congestion). The Deep SARSA (DSARSA) learning algorithm with experience replay applied in the training process. The DSARSA replay model is built to extract traffic information from all lanes by applying multilayer perceptron with two hidden layers. Then, at the output layer an approximated q-value is produced and updated with SARSA algorithm. For training, a random batch of experiences is sampled from the replay buffer. This helps break the correlation between consecutive experiences, stabilizes training and enables the model to be able deal appropriately with any traffic condition. Experiments on 1000 episode of low, medium and heavy traffic flow was conducted on SUMO traffic simulator. TraCL allows interface with the SUMO simulation using python showed that the proposed DSARSA replay model achieved a state-of-the-art performance compared to other baseline DRL models applied in traffic control in terms of learning stability and reward maximization.

Keywords: approximated, congestion, SARSA, optimization, python, replay, simulation, stability, traffic.

I. INTRODUCTION

Traffic control systems has been quite poor in many developing countries, despite the growth in transport demand and supply [27]. The rapid growth in the use of vehicles has overloaded the urban roads thereby causing traffic congestion. The resultant traffic congestion has become impeding to human lives. Traffic congestion on road networks generally occurs as a result of excessive use of road infrastructure beyond capacity, and it is characterized by slower speeds, longer trip hours and increased vehicular queuing [25]. Traffic congestion is an increasing problem in cities. Sub urban spend more of their time commuting to work, school, shopping, and social event as well as dealing with traffic congestion and accidents.

This creates an urgent need to operate our transportation systems with maximum efficiency. Man, nations, regions and the world would be severely affected in terms of all forms of development without an efficient traffic control system, which is a key factor for physical and economic growth[24]. Traffic congestion has become a major menace on urban roads as a result of inefficient traffic control systems.

Traffic congestion has numerous negative impacts on the economy of a city and a country as a whole. For example, modern businesses, industries, trades and general activities depend on transport and transport infrastructure, this is because movement of goods and services from place to place are becoming vital and inseparable aspects of global and urban economic. Traffic congestion on an intersection may be caused by various factors

like indiscipline drivers who show little respect for other road users. Also, traffic congestion maybe caused by the drivers' ignorance of the prevailing traffic conditions. The driver may not have knowledge of a particular routes traffic condition before entering into it and he may then decide to use another route in order to avoid being delayed. Another factor that may lead to traffic congestion is drivers neglecting traffic signal. There are three main traffic control signals, Red, yellow and Green.

Traffic Control Signals determines traffic phase in order to manage the flow of traffic and reduce congestion. Several scholars have propounded theories and applied different methodologies in the development of traffic signal control systems that can operate efficiently at real time [29]; see also[28],[29],developed traffic control system using RFID. Although the work of these scholars have contributed to the body knowledge, however the amount of time vehicles were delayed was still a major challenge considering the dynamically changing nature of traffic of urban roads in major cities around the world. Recently, due to the incapability of dealing with dynamic multi-direction traffic in previous methods researchers have intensified their focus on the application of Deep Neural Networks and Reinforcement Learning (RL) algorithm in control systems.

One of the challenges observed by the researcher in (Lucas, 2000) is on how to integrate a mechanism for utilizing past experiences of traffic conditions using the normal traffic pattern of that vicinity and still control the road traffic in that environment at real-time has prompted the development of an on- policy Deep reinforcement learning algorithm with experience replay that can control traffic at real time.

The rest of this paper is organized as follows. Section 2 discusses the related literature on deep reinforcement learning traffic control models. Section 3 gives a full description of the architecture of the proposed DSARSA replay model, Section 4 describes the traffic simulation, performance evaluation metrics, and experimental results, and Section 5 concludes.

II. RELATED WORKS

Scholars around the world have propounded theories on different methodologies on traffic control systems. [29] developed a dynamic traffic control system using RFID. Each individual vehicle is equipped with the special radio frequency identification (RFID) tag to track the vehicle and RFID reader was used to read the RFID tags

attached to the vehicle's windshield.[28] developed an RFID controlled traffic signal model. The RFID reader mounted on the required traffic signals is used to record the important details like the license number and vehicle type of each vehicle that passes the particular signal from the RFID tag fixed in the vehicles' license plates and store them in a database. [25] explored a prioritized traffic light controller that effectively deals with traffic disorder and public noise during emergency and reduce it to the minima in the sense that the control is done automatically without creating unnecessary attention. The system installed at the junction will respond to an interrupt by an emergency vehicle by detecting it while approaching an intersection lane through the RFID attached to it.

2.1 DEEP REINFORCEMENT LEARNING TRAFFIC CONTROL SYSTEMS

Over the years now, the work of the several scholars have shown that the efficacy of Deep reinforcement learning agents in traffic control. [35] developed an on-line Sarsa(λ)-based traffic signal light optimization model for Real-Time Traffic Light Control. The Sarsa(λ)-based model gains considers delay time, the number of waiting vehicles, and the integrated saturation from its experiences to learn and determine the optimal actions. However, SARSA(λ) is sensitive to different models of reward and initialization[12]. The results obtained from the work of these authors show that the performance of SARSA(λ) can be influenced by the reward type and significantly influenced by the initialization of the Q-table when an undirected exploration is used. This work was improved in the work of [21], the authors applied SARSA Learning approach in control traffic lights. The authors applied SARSA algorithm in minimizing the total time traffic travel, individual vehicle travel time to four lanes and waiting time of vehicles The system consists of two layers which are environment's perception and traffic management. [11] developed a traffic signal control system which takes advantage of this new, high-quality data, with minimal abstraction. The focuses on the application of deep reinforcement learning methods to build a truly adaptive traffic signal control agent in the traffic micro-simulator SUMO. They proposed a new state space, the discrete traffic state encoding, which is information dense. Also (Mousavi and Schukat, 2017) in their work utilized two kinds of reinforcement learning algorithms which are deep policy-gradient and value-function based agents that can predict the best possible traffic signal for a traffic intersection. The policy-gradient

based agent maps its observation directly to the control signal, however the value-function based agent first estimates values for all legal control signals. The agent then selects the optimal control action with the highest value. The result of the work was shown using SUMO traffic simulator. One shortcoming of this work is stated in (Mousavi and Schukat, 2017)[5] that it cannot function perfectly in a larger-sized network with multiple intersections. Processing of raw traffic data under huge state space is proposed in [30]. This is achieved by integrating the deep Q network (DQN), Q-learning algorithm and deep CNNs to find an optimal control policy of traffic signals. This deep reinforcement learning-based traffic signal control method was implemented using high-resolution event-based data. The system was able to achieve cost-effective and efficient adaptive traffic signal control.

[1] explored data-guided traffic planning and control using Reinforcement Learning (RL). Several recent techniques were embedded in the Deep RL algorithm that improve the original Deep Q-Networks (DQN) for discrete control and the traffic-related interpretations that followed were discussed. The authors developed a novel DQN-based algorithm for Traffic Control (called TC-DQN+) as a tool for fast and more reliable traffic decision-making. Kanis et al., (2021a) developed a traffic light timing optimization method based on double dueling deep Q-network, Max Pressure, and Self-organizing traffic lights (SOTL), which controls traffic flows by dynamically adjusting the duration of traffic lights in a cycle, whether the phase is switched based on the rules set in advance and the pressure of the lane. The SUMO is used to simulate two traffic scenarios. The selected two types of evaluation indicators and compared four methods to verify the effectiveness of the agent. However in the work of [4] the authors noted the fact that in a road traffic junction scenario, the vehicle typically receives partial observations from the transportation environment, In this study, the safety performance of three baseline DRL models (DQN, A2C, and PPO) was evaluated and proposed a self-awareness module from an attention mechanism for DRL to improve the safety evaluation for an anomalous vehicle in a complex road traffic junction environment, such as intersection and roundabout scenarios, based on four metrics: collision rate, success rate, freezing rate, and total reward. [20] developed a novel approach that allows for better formulation of state and reward definitions in order to boost the performance of the traffic signal controller agent.

The double deep Q-Network (DDQN) along with prioritized experience replay (PER) was utilized for the agent architecture. To evaluate the performance of our approach, series of simulations using the Simulation of Urban Mobility (SUMO) environment. Two multilayer perceptron neural networks are employed, the first one which is the main network, is used to predict Q-values of actions. While the second network, namely, the target network, is involved in updating the first network. Most recently [33] developed a reliable deep reinforcement learning based controller for a highly dynamic environment and investigated the resilience of these controllers to a variety of environmental disruptions. The agent is trained using deep Q-learning and experience replay. The model is evaluated in the traffic micro-simulator SUMO. The simulation results demonstrate that the developed method is effective at shortening queues when there is disruption. The works of these authors have contributed greatly to the body of knowledge especially in the area of traffic optimization in order to reduce traffic congestion. However, [35] only explored SARSA learning in traffic control without integrating it with Deep learning hereby having no mechanism for acquiring the state of the current environment properly. Also, [14],[30] and [32] applied DDPG and DQN algorithm with experience replay which are off policy deep reinforcement learning algorithms. Off policy methods are independent of the agent's actions. It evaluates or improve a policy different from that used to generate the data this makes it to have learn properly from the most current state of the environment. One of the challenges observed in the work of (Lucas, 2000) is on how to integrate a mechanism for utilizing past experiences of traffic conditions using the normal traffic pattern of that vicinity and still control the road traffic in that environment at real-time. This will require a deep learning model that will sense the current state of the traffic environment at real time. Also, for off-policy frameworks, evaluation becomes challenging as there is too much exploration while an on-policy method estimates the value of a policy while still using it. This study intends to approaches this challenge by the use deep SARSA reinforcement learning replay model for the control of road traffic at real-time. The purpose of this research work is to develop a traffic control agent that can control traffic at real-time reduce traffic congestion.

III. METHODOLOGY

To address the challenges of the existing architectures, we propose a novel Deep SARSA replay traffic control model to extract current traffic information from the incoming lanes by applying multilayers perceptron, updating the Q-value using SARSA algorithm and exploring the richness of utilizing past experience of that particular traffic vicinity in the training process. This section presents feature extraction of the traffic state, action selection process, q-value update, the experience replay buffer and the system architecture of the proposed DSARSA replay traffic control model.

3.1 FEATURE EXTRACTION OF THE TRAFFIC STATE

Multilayer perceptron (MLP) will be used for state feature extraction to effectively capture the complex relationships and patterns in the input state data. The traffic status is fed into the neural network. In a Deep SARSA traffic light control model, the state representation typically involves combining various features of the traffic environment into a single state vector that is fed into the MLP. Specifically, the number of vehicles and the current traffic light state are two key components that will be concatenated to form this input vector.

- Suppose there are N incoming lanes at the specific intersection.
- The number of vehicles in each lane can be represented as a vector v

$$v = [v_1, v_2, \dots, v_N] \quad (1)$$

where v_i represents the number of vehicles on lane i at time t .

The current traffic light phase is encoded as a one-hot vector p

$$p = [p_1, p_2, \dots, p_M] \quad (2)$$

where $p_j = 1$ if the current phase is j , and $p_j = 0$ otherwise. The state is represented as a vector, which serves as the input to the MLP.

The state vector s that represents the entire traffic situation is obtained by concatenating the vehicle count vector c and the traffic light state vector p .

Mathematically, this can be expressed as:

$$s = [v_1, v_2, \dots, v_N, p_1, p_2, \dots, p_M] \quad (3)$$

where s is the input vector of size $N + M$ which include the current input information at time step t in the traffic simulation.

The MLP contains two hidden layers, where each neuron applies a linear transformation to the inputs, followed by a non-linear activation function (e.g., ReLU). The output of this hidden layer, h_1 , becomes the input to the next layer.

Mathematically, the input to neuron j is computed as a weighted sum of the outputs.

$$z_j = \sum_{i=1}^n w_{ij} h_i + b_j \quad (4)$$

where

- w_{ij} is the weight associated with the connection between the i -th neuron in the previous layer and the j -th neuron in the current hidden layer.
- h_i is the output from the i -th neuron in the previous layer/
- b_j is the bias term for neuron j

After computing the weighted sum z_j , ReLU activation function $f(z)$ is applied to introduce non-linearity. This step allows the MLP to learn complex patterns in the data.

$$f(z) = \max(0, z) \quad (5)$$

The hidden layer transforms the input data into a new space by combining features non-linearly. This transformation is crucial because it allows the network to capture complex patterns in the input data about certain traffic conditions and light phases these together influence future traffic states. The output j of each neuron in the hidden layer is then passed as input to the neurons in the next layer (which could be another hidden layer or the output layer). This process continues through all the hidden layers in the network, allowing the MLP to learn increasingly abstract features at each layer.

Output Layer: The final layer of the MLP produces the Q-values. The hidden layers help in learning the complex relationships between the current traffic state (including vehicle counts and traffic light phases) and the future rewards associated with different actions. The final output layer uses these learned features to predict Q-values, which indicate the expected future reward for each possible action in the given state. These inputs are the activations h_j from the neurons in the final hidden layer i.e.

The output of neuron j in the hidden layer is

$$h_j = f(z_j) \quad (6)$$

where h_j is the activated output of neuron j from the last hidden layer. If the final hidden layer has m neurons, the input to the output layer can be represented as a vector h .

$$h = [h_1, h_2, \dots, h_m] \quad (7)$$

Each neuron in the output layer computes a weighted sum of the inputs from the last hidden layer. The output layer has k neurons, where each neuron corresponds to a different action (e.g., different traffic light phases).

For the l -th neuron in the output layer, the weighted sum is computed as:

$$z_j = \sum_{j=1}^m w_{jl} h_j + b_l \quad (8)$$

where: w_{jl} is the weight associated with the connection between the j -th neuron in the last hidden layer and the l -th neuron in the output layer. b_l is the bias term for the l -th neuron in the output layer.

The output layer is responsible for producing Q-values, which are estimates of the expected future rewards for each action given the current state.

Unlike hidden layers, the output layer typically does not apply a non-linear activation function. Instead, the raw weighted sum z_l is directly used as the Q-value for action a_1 ,

$$Q(s, a_1) = z_l \quad (9)$$

where $Q(s, a_1)$ is the estimated Q-value for taking action a_1 in state s .

The output vector can be expressed as:

$$Q(s) = [(Q(s, a_1), Q(s, a_2), \dots, Q(s, a_k))] \quad (10)$$

where k is the number of possible actions (traffic light phases). The action with the highest Q-value is often chosen as the optimal action in a given state. Linear activation is used to predict Q-values. Initializing the Q-value function, $Q(s,a)$ using the deep neural network.

A linear stack of layers are involved as shown in Figure 1. The MLP consists of an input layer, two fully connected (dense) layers, where each neuron in a layer is connected to every neuron in the previous and next layers and then it has an output layer. The input acquired from SUMO through its built-in functionalities, first hidden layer with 24 neurons and ReLU activation. The input dimension is `state_size`. The next is the second hidden layer with 24 neurons and ReLU activation and the Output layer with a number of neurons equal to the action space size, the action space size is 3.

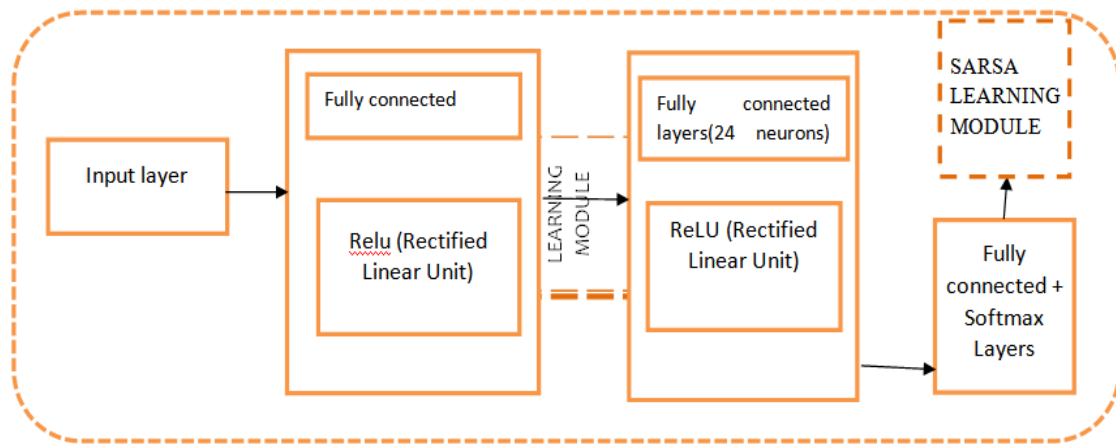


Figure 1: The Deep learning Q function approximation process

3.2 ACTION SELECTION PROCESS

The agent selects an action (changing the traffic light phase) based on the current state. All lanes are set to green at the initial stage and then an action is chosen randomly. This is done using epsilon greedy policy. After taking an action, the agent observes the reward and then get the next state. This is known as the exploration process.

DSARSA typically uses an epsilon-greedy policy for selecting actions. The agent balances exploration (trying new actions) and exploitation

(choosing the best-known action based on current knowledge).

- Exploration: With a probability of epsilon (ϵ), the agent chooses a random action. This helps the model explore new strategies and avoid getting stuck in local optima.
- Exploitation: With a probability of $1 - \epsilon$, the agent chooses the action with the highest predicted Q-value, leveraging what it has learned so far.

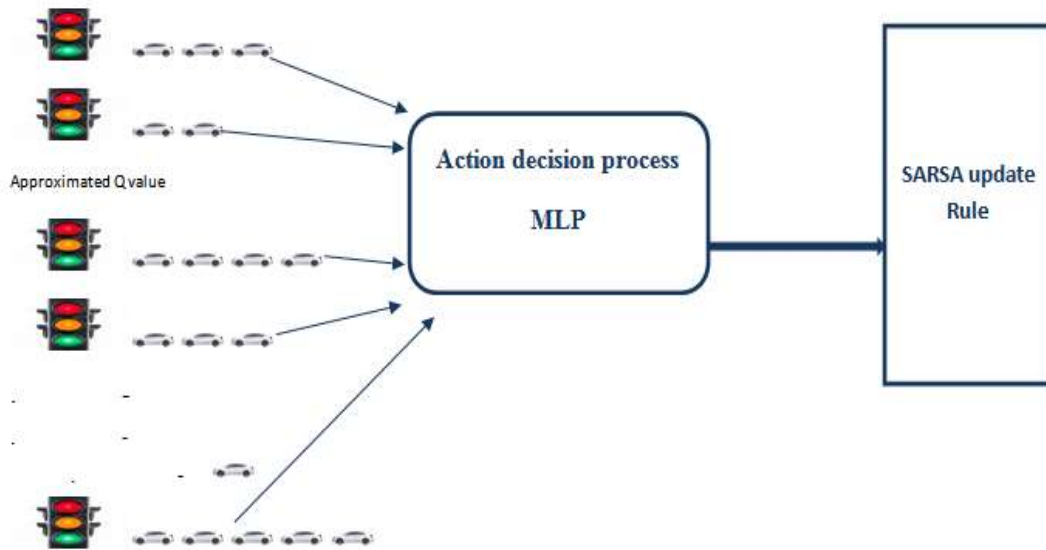


Figure 2: Action decision process

The Q-function $Q(s_t, a_t; \theta)$ approximates the expected return (future cumulative reward) starting from state s_t and taking action a_t , where θ represents the parameters (weights and biases) of the MLP. The MLP takes the state-action pair (s_t, a_t) as input and outputs the Q-value.

The Deep SARSA agent then updates the Q-value based on the observed reward and the estimated future rewards using the following update rule:

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha [r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)] \quad (11)$$

where:

- s_t is the current number of vehicle on lane i at time t .
- a_t is the action (traffic light signal at lane i taken at time t)
- r_{t+1} is the reward received after taking action a_t
- s_{t+1} is the next state after taking action a_t
- a_{t+1} is the next action chosen at state s_{t+1}
- α is the learning rate.
- γ is the discount factor.

3.3 EXPERIENCE REPLAY BUFFER

Experience Replay is a technique widely used in training deep reinforcement learning models, including Deep SARSA, to improve learning efficiency and stability. Experience Replay helps in breaking the correlation between consecutive experiences during training. It stores the agent's experiences in a buffer and samples random mini-batches from this buffer to train the model. Experiences are stored at each time step in the simulation into the buffer. Periodically, sample

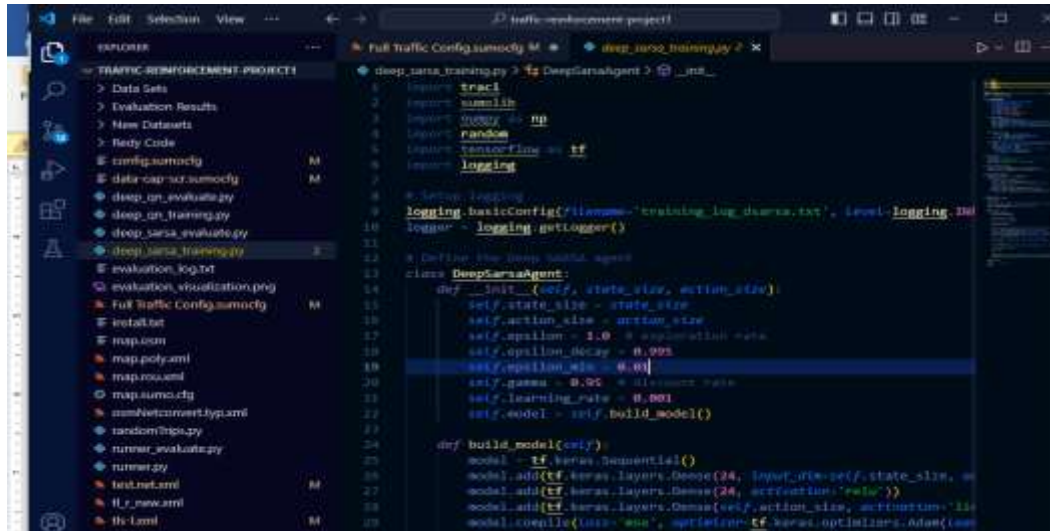
a mini-batch of experiences from the buffer. These experiences are used to perform updates on the neural network that estimates the Q-values. During each training step, the TD error is calculated for the sampled mini-batch from the replay buffer. This TD error is used to compute the gradient of the loss function with respect to the neural network's weights. The Adam optimizer uses these gradients to update the weights of the neural network. The optimizer adjusts the learning rate dynamically for each weight, depending on the gradients' history, leading to more stable and faster convergence. The environment is initialized at the beginning of each episode. Epsilon greedy policy will be used for action selection selecting a random action otherwise select the action with the highest Q-value (exploitation).

IV. IMPLEMENTATION AND TRAINING

This section presents the details of our implementation, and the standard performance metrics considered in the evaluation of the proposed model against benchmarking architectures. We then presented a detailed comparison of the performance of the proposed model based on quantitative experiments and generalization experiments. For a fair comparison, the existing benchmarking architectures and the proposed DSARSA replay were implemented using the python and trained on SUMO. To train our models, we have selected an intersection along Oba Adesida road Akure, Ondo State, Nigeria. The Deep SARSA model is

initialized. The neural network weights network weights are initialized. Hyperparameters like the learning rate α , discount factor γ , exploration rate,

epsilon ϵ , and the decay rate of epsilon ϵ are set as shown in the code snippet in figure 1.1



```

import tracl
import sumolib
import random
import tensorflow as tf
import logging

logging.basicConfig(filename='training_log_data.txt', level=logging.DEBUG)
logger = logging.getLogger()

# Define the DeepSARSA agent
class DeepSARSAAgent:
    def __init__(self, state_size, action_size):
        self.state_size = state_size
        self.action_size = action_size
        self.epsilon = 1.0 # exploration rate
        self.epsilon_decay = 0.995
        self.epsilon_min = 0.01
        self.gamma = 0.95 # discount rate
        self.learning_rate = 0.001
        self.model = self.build_model()

    def build_model(self):
        model = tf.keras.Sequential()
        model.add(tf.keras.layers.Dense(24, input_dim=self.state_size, activation='relu'))
        model.add(tf.keras.layers.Dense(24, activation='relu'))
        model.add(tf.keras.layers.Dense(self.action_size, activation='linear'))
        model.compile(loss='mse', optimizer=tf.keras.optimizers.Adam(learning_rate=0.001))
  
```

Figure 3: Initialization of the Deep SARSA replay model

SUMO is responsible for generating the traffic simulation while TraCL allows interface with the SUMO simulation using python. TraCL is used to start and control the simulation as shown in these lines of code.

Experience replay is initialized and past experiences stored (state, action, reward, state, action) and sampling is done randomly during training to break correlation between consecutive experiences. The SUMO simulation is reset to the initial state, starting a new episode. The agent observes the initial traffic state from SUMO, which number of vehicles on each lane at the intersection.

The agent selects an action (traffic light phase) for the current state using an ϵ -greedy policy. With probability ϵ , the agent selects a random action to explore the action space. With probability $(1-\epsilon)$, the agent selects the action that maximizes the Q-value based on the neural network's predictions. The selected action (e.g., switching the traffic light phase) is applied in the SUMO environment. The SUMO simulation runs for a few seconds (or timesteps), updating vehicle positions, speeds, and traffic conditions based on the selected action.

4.3 PERFORMANCE METRICS

Comparison was done between the other benchmarking architectures and the DSARSA in the training process for low, moderate and busy traffic flow. The fixed-timing due to its nature and metrics for its evaluation, was evaluated based on the average speed of each vehicle and the average waiting time on each vehicle on each lane the result is as shown Figure 4

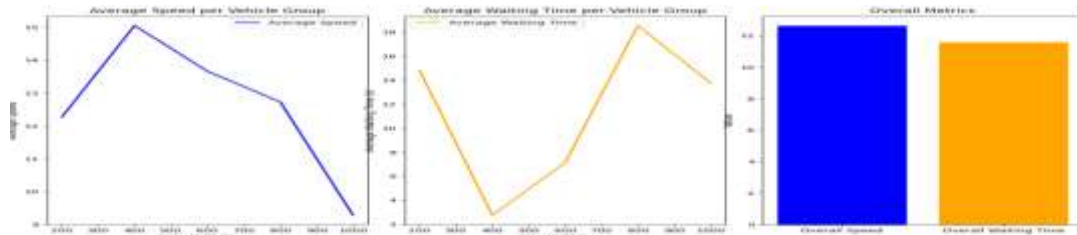


Figure 4: Fixed timing results based on Average speed and average waiting time per episode.

The inverse relationship between average speed and waiting time suggests that as traffic increases or congestion builds up, the average speed tends to decrease, leading to longer waiting times.

Other standard metrics used for DRL models in traffic control like Total reward, average

waiting time and epsilon decay will be used in the evaluating and comparing the performance of the DQN, Deep SARSA and Deep SARSA Replay model given the same level of traffic congestion and the same number of episodes. The results are shown in Figure 5-7:

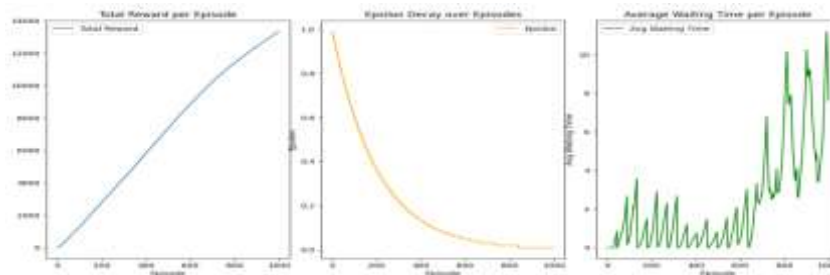


Figure 5: Deep SARSA results reward, epsilon decay and average waiting time per episode

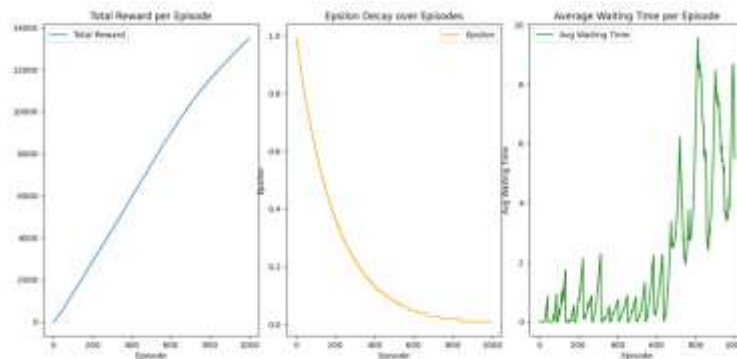


Figure 6: Deep SARSA results reward, epsilon decay and average waiting time per episode

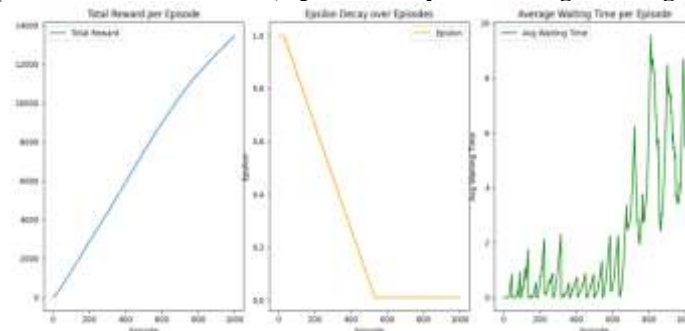


Figure 7: Deep SARSA results reward, epsilon decay and average waiting time per episode

MODEL	TOTAL REWARD	AVERAGE WAITING TIME	EPSILON DECAY
DQN	As the agent gains more experience through episodes, it achieves higher rewards,	The average waiting time per episode fluctuates but exhibits an overall increasing trend with spikes.	The epsilon value decreases gradually as the number of episodes increases.
DSARSA	The total reward continues to increase linearly as the number of episodes	There is continuous decline in epsilon this reflects a strategic transition from exploration to more deterministic behavior, relying on the accumulated experience to make optimal decisions	There was rising and fluctuating of average waiting time which suggests that while the agent is becoming more proficient at achieving rewards, it is struggling to manage waiting time effectively as training progresses.
DSARSA REPLAY	The total reward increases steadily over episodes, indicating that the agent is learning to optimize the traffic light control.	Epsilon starts at 1.0 and decays linearly to near-zero around episode 600, then remains constant.	With annealing learning shows a reduction in average waiting time towards the ending episodes as training progresses.

DQN, Deep SARSA control methods were observed to demonstrate improvements under different traffic flows for 1000 episodes. The same set of hyperparameters to train all the models; we have used the Adam optimizer with annealing learning rate starting with initial learning rate 0.001, discount factor 0.95 for 1000 episodes and the replay buffer size was 2000. The Deep SARSA replay outperforms both the DQN and the Deep SARSA in terms of long-term total rewards, waiting time, and quicker convergence of epsilon decay. Therefore, the Deep SARSA replay model appears to have better overall performance with annealing learning. Deep SARSA with Replay allows the model to learn from past experiences by replaying them from a buffer, which can help stabilize learning, particularly in environments with noisy or complex dynamics. This might explain the steady improvement in total reward. Although early in training, the agent makes larger updates to its policy, but as training progresses with the annealing learning rate, these updates become smaller. This allows the agent to fine-tune its behavior.

V. DISCUSSION

The DSARSA replay traffic control model with an annealing learning rate has shown notable improvements in traffic management performance compared to baseline models like DQN and standard DSARSA. One key advantage of the annealing learning rate is its adaptive reduction in

step size as training progresses, allowing the model to refine its policy more precisely over time. This flexibility enhances the model's ability to reduce waiting times effectively, particularly under dynamic traffic conditions where frequent, abrupt adjustments are needed.

Compared to the DQN model, which generally relies on a static learning rate, the DSARSA replay model with annealing can better adapt to varying traffic flows, leading to more efficient phase transitions in the traffic light cycles. This adaptability is critical in traffic control, where over-adjustment can lead to inefficiencies, while under-adjustment can increase congestion. Furthermore, the DSARSA replay model inherently benefits from the integration of replay memory, which helps smooth out learning by revisiting past experiences and allows the model to generalize better across different traffic scenarios.

Empirical results indicate a significant reduction in the average waiting time for vehicles, a direct measure of improved traffic flow. This improvement highlights the value of the annealing learning rate in enhancing the model's ability to balance exploration and exploitation, especially in complex urban traffic networks. Consequently, the DSARSA replay with annealing emerges as a robust and effective solution, providing a balanced and responsive approach to traffic light scheduling that outperforms both DQN and traditional DSARSA models.

REFERENCES

- [1]. S. Alemzadeh, R. Moslemi, R. Sharma, and M. Mesbahi, (2020). Adaptive Traffic Control with Deep Reinforcement Learning: Towards State-of-the-art and Beyond. July. <http://arxiv.org/abs/2007.10960>
- [2]. F. Almahamid, (2021). Reinforcement Learning Algorithms: An Overview and Classification. IEEE, 1.
- [3]. K. Bálint, T. Tamás and B. Tamás, (2022). Deep Reinforcement Learning based approach for Traffic Signal Control. Transportation Research Procedia, 62(March), 278–285. <https://doi.org/10.1016/j.trpro.2022.02.035>
- [4]. Z. Cao & J. Yun, (2022). Self-Awareness Safety of Deep Reinforcement Learning in Road Traffic Junction Driving. 1–10. <http://arxiv.org/abs/2201.08116>
- [5]. N. Casas (2017). Deep Deterministic Policy Gradient for Urban Traffic Light Control. 1–38. <http://arxiv.org/abs/1703.09035>
- [6]. A. Cheng, M. Pang, and P. Pavlou. (2020). Mitigating Traffic Congestion: The Role of Intelligent Transportation Systems. The London School of Economics and Political Science, 1–56.
- [7]. W. Cheng, S. Wang, and X. Cheng. (2014). Virtual track: Applications and challenges of the RFID system on roads. IEEE Network, 28(1), 42–47. <https://doi.org/10.1109/MNET.2014.6724105>
- [8]. T. Chu, J. Wang, L., Codeca and Z. Li, (2020). Multi-agent deep reinforcement learning for large-scale traffic signal control. IEEE Transactions on Intelligent Transportation Systems, 21(3), 1086–1095. <https://doi.org/10.1109/TITS.2019.2901791>
- [9]. O. D., Ese and O. E. Ighodalo, (2017). An Intelligent System for Traffic Control in Smart Cities: A Case Study. American Journal of Artificial Intelligence, 1(1), 36–43. <https://doi.org/10.11648/j.ajai.20170101.15>
- [10]. P. Geetha, S., Kumar, R. Pradeep, and S. Pradeep (2020). Smart Traffic Control system using RFID. International Research Journal of Engineering and Technology (IRJET), 7(3), 2416–2421.
- [11]. W. Genders and S. Razavi, (2016). Using a Deep Reinforcement Learning Agent for Traffic Signal Control. February. <http://arxiv.org/abs/1611.01142>
- [12]. M. Grzes (2008). Robustness Analysis of SARSA (λ): Different Models of Reward and Initialisation. Robustness Analysis of SARSA (λ): Different. September, 144–156. <https://doi.org/10.1007/978-3-540-85776-1>
- [13]. J. Gu, Y. Fang, Z., Sheng and P. Wen, (2020). Double deep Q-network with a dual-agent for traffic signal control. Applied Sciences (Switzerland), 10(5). <https://doi.org/10.3390/app10051622>
- [14]. M., Guo, P. Wang, C., Chan, and S. Askary, (2019). A Reinforcement Learning Approach for Intelligent Traffic Signal Control at Urban Intersections. IEEE Intelligent Transportation Systems Conference (ITSC), 4242–4247. <https://doi.org/doi:10.1109/ITSC.2019.8917268>
- [15]. S. Kanis, L. Samson, D. Bloembergen, and Bakker, T. (2021a). Back to Basics: Deep Reinforcement Learning in Traffic Signal Control. In Proceedings of The 10th International Workshop on Urban Computing (UrbComp 2021) (Vol. 1, Issue 1). Association for Computing Machinery. <http://arxiv.org/abs/2109.07180>
- [16]. S. Kanis, L. Samson, D. Bloembergen, and Bakker, T. (2021b). Deep Reinforcement Learning in Traffic Signal Control. In Proceedings of The 10th International Workshop on Urban Computing (UrbComp 2021) (Vol. 1, Issue 1). Association for Computing Machinery.
- [17]. A. Kekuda, R. Anirudh and M. Krishnan, (2021). Reinforcement Learning based Intelligent Traffic Signal Control using n-step SARSA. Proceedings - International Conference on Artificial Intelligence and Smart Systems, ICAIS 2021, 379–384. <https://doi.org/10.1109/ICAIS50930.2021.9395942>
- [18]. Z., Li, C. Xu, and G. Zhang, (2020). A Deep Reinforcement Learning Approach for Traffic Signal Control Optimization.
- [19]. X. Liu, M. Zhu, and S. Borst, (2023a). Deep Reinforcement Learning for Traffic Light Control in Intelligent Transportation Systems. IEEE Transactions on Network Science and Engineering, 1(Xx), 1–17.
- [20]. X. Liu, M. Zhu, and S. Borst, (2023b). Deep Reinforcement Learning for Traffic Light

- Control in Intelligent Transportation Systems. *IEEE Transactions on Network Science and Engineering*, 1, 1–17.
- [21]. M. Rezzai, W. Dachry, F. Moutaouakkil, and H. Medromi. (2015). Designing an Intelligent System for Traffic Management. *Journal of Communication and Computer*, 12(3), 123–127. <https://doi.org/10.17265/1548-7709/2015.03.004>
- [22]. S. S. Mousavi and M. Schukat, (2017). Traffic Light Control Using Deep Policy-Gradient and Value-Function Based Reinforcement Learning. 2.
- [23]. M. Muresan, G. Pan and L. Fu, (2019). Adaptive Traffic Signal Control with Deep Reinforcement Learning An Exploratory Investigation, 97th Annual Meeting of the Transportation Research Board, doi:10.48550/arXiv.1901.00960
- [24]. O. Ogbeide and A. Akingbesote. (2018). Designing a Front-Back End Solution for the Issuance of Drivers' License by FRSC in Nigeria. *Global Journal of Computer Science and Technology: E Network, Web & Security*, 18(4), 28–41.
- [25]. C. Okoh, A. Oluwole, and O. Akinsanya (2022). Design and Implementation of Intelligent Traffic Control System using Programmable Logic Controller, *Journal of Engineering and Technology*, 7(3), <https://doi.org/10.46792/fuoyejet.v7i3.858>
- [26]. C. Onyeneke (2018). Causes and Effects of Traffic Congestions in Nigeria. *Transportation Journal of ASCE*, 18(5).
- [27]. B. O Oyebola, (2018). Prioritized Traffic Light Controller: A Technical Approach to Avoid Traffic Disorder and Public Noise During Emergency in Nigeria. *International Journal of Higher Education and Research*, 7(1), 11–20.
- [28]. S. Sayani, and P. Nanvani, (2019). Traffic Analysis and Estimation using Deep Learning Techniques. *International Journal of Engineering Research & Technology (IJERT)*, 8(9), 803–807.
- [29]. P. Starkey and J. Hine, (2014). Poverty and sustainable transport How transport affects poor people with policy implications for poverty reduction A literature review (Issue October).
- [30]. P. Vijayaraman and J. Jayarin, (2019). Intelligent Traffic Management using RFID Technology. *International Journal of Recent Technology and Engineering (IJRTE)*, 8(4), 1743–1745. <https://doi.org/10.35940/ijrte.c5833.118419>
- [31]. P. Waghare, P. Nalawade, N. Vanare, and J. Jadhav (2017). Dynamic traffic control system using RFID technology. *International Journal of Advance Research in Science and Engineering*, 6(4), 994–999.
- [32]. S. Wang, X. Xie, K. Huang, J. Zeng, and Z. Cai (2019). Deep reinforcement learning-based traffic signal control using high-resolution event-based data. *Entropy*, 21(8), 1–16. <https://doi.org/10.3390/e21080744>
- [33]. H. Wei, (2018). IntelliLight: A Reinforcement Learning Approach for Intelligent Traffic Light Control. *International Conference on Knowledge Discovery & Data Mining*. <https://doi.org/10.1145/3219819.3220096>
- [34]. C. Yen, D. Ghosal, M. Zhang, and N. Chuah, (2020). A Deep On-Policy Learning Agent for Traffic Signal Control of Multiple Intersections. 2020 IEEE 23rd International Conference on Intelligent Transportation Systems, ITSC 2020. <https://doi.org/10.1109/ITSC45102.2020.9294471>
- [35]. Z. Zeinaly, M. Sojoodi, and S. Bolouki, (2023). A Resilient Intelligent Traffic Signal Control Scheme for Accident Scenario at Intersections via Deep Reinforcement Learning. 1–27.
- [36]. X. Zhou, (2014). A Sarsa (λ)-Based Control Model for Real-Time Traffic Light Coordination. *The Scientific World Journal*, 7. <https://doi.org/10.1155/2014/759097>
- [37]. X., Zhou, F. Zhu, Q. Liu, Y. Fu, and W. Huang, (2014). A Sarsa(λ)-based control model for real-time traffic light coordination. *The Scientific World Journal*, 2014. <https://doi.org/10.1155/2014/759097>