

To the Graduate Council:

I am submitting herewith a thesis written by Zachary Sublett entitled "GAT-DQN for Unsignalized Mixed Traffic Control." I have examined the final paper copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Computer Science.

Dr. Wezi Li, Major Professor

We have read this thesis
and recommend its acceptance:

Dr. Wezi Li

Dr. Schuman

Dr. Mockus

Accepted for the Council:

Dixie L. Thompson

Vice Provost and Dean of the Graduate School

To the Graduate Council:

I am submitting herewith a thesis written by Zachary Sublett entitled “GAT-DQN for Unsignalized Mixed Traffic Control.” I have examined the final electronic copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Computer Science.

Dr. Wezi Li, Major Professor

We have read this thesis
and recommend its acceptance:

Dr. Wezi Li

Dr. Schuman

Dr. Mockus

Accepted for the Council:

Dixie L. Thompson

Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)

GAT-DQN for Unsignalized Mixed Traffic Control

A Thesis Presented for the

Master of Science

Degree

The University of Tennessee, Knoxville

Zachary Sublett

August 2025

© by Zachary Sublett, 2025
All Rights Reserved.

*I would like to dedicate this thesis to my parents, for always believing in me and
being there to support me.*

Abstract

We look at the problem of unsignalized mixed traffic control in large urban areas and propose a method for topology independence that should allow for a more generalizable approach to this problem. We will utilize a Graph Attention Network (GAT) for feature extraction and a Dueling Deep Q-Network (DQN) for autonomous vehicle control. to optimize real world traffic flow data from a benchmark dataset. SUMO (Simulation of Urban Mobility) was used to conduct the experiments.

Table of Contents

1	Introduction	1
1.1	Chapter Overview	1
1.2	Introduction to the Traffic Control Problem	1
1.3	Background and Motivation	3
1.3.1	Reinforcement Learning	3
1.3.2	Graph Neural Networks	4
1.4	Purpose of This Thesis	5
1.5	Thesis Outline	5
1.6	Chapter Conclusion	6
2	Literature Review	7
2.1	Chapter Overview	7
2.2	Reinforcement Learning Based Approaches to Traffic Control	8
2.2.1	Unsignalized Traffic Control	11
2.3	Graph Neural Networks for Traffic Prediction and Control	12
2.3.1	Graph Convolutional Networks (GCNs)	13
2.3.2	Graph Attention Networks (GATs)	13
3	Methodology	15
3.1	Chapter Overview	15
3.2	Graph Modeling	15
3.3	Graph Attention Network	16

3.4	Dueling DQN	17
3.5	Prioritized Experience Replay (PER)	18
3.6	Reinforcement Learning Components	19
3.6.1	Action Space	19
3.6.2	Epsilon Greedy Policy	20
3.6.3	Reward Function	20
3.7	Chapter Conclusion	21
4	Experimental Setup and Results	22
4.1	Chapter Overview	22
4.2	Experimental Setup	22
4.2.1	Simulation Environment	22
4.2.2	Dataset Description	23
4.2.3	Scenario Selection	23
4.3	Model Training	24
4.3.1	Training Setup and Configuration	24
4.3.2	Training Performance	26
4.4	Results and Analysis	28
4.4.1	Performance on Training Roundabout	28
4.4.2	Performance on Untrained Maps	29
4.4.3	Quantitative Comparison	31
4.5	Ablation Study	32
4.6	Improving Model With Ablation Results	33
4.7	Chapter Conclusion	34
5	Conclusion	36
5.1	Summary and Contributions	36
	Bibliography	38

List of Tables

4.1	Benchmark Dataset Summary	24
4.2	GAT-DQN Training Hyperparameters and Configuration	26
4.3	Average Wait Time and Relative Improvement for GAT-DQN	32
4.4	Average Wait Time Across All Maps (Ablated Models)	33
4.5	Average Wait Time Across All Maps (Improved Models)	34

List of Figures

3.1	Representation of the traffic graph.	16
4.1	Example of dataset SUMO map.	25
4.2	Training performance measured by reward, average speed, wait time, and throughput over 100 episodes.	27
4.3	Average wait time of GAT-DQN across four different roundabout maps.	28
4.4	Average wait time on an intersection map across all difficulty levels. .	29
4.5	Average wait time on a roundabout map across all difficulty levels. .	30

Chapter 1

Introduction

1.1 Chapter Overview

This chapter introduces the motivation and background for the problem of intelligent traffic control, specifically in the context of unsignalized mixed traffic scenarios. We will first take a look at the challenges modern urban areas face with traffic congestion and the limitations of rule based control strategies. Followed by the prominence of reinforcement learning based methods in unsignalized mixed traffic control, as well as how Graph Neural Networks (GNNs) can be useful to traffic environments.

1.2 Introduction to the Traffic Control Problem

Traffic congestion is a pressing issue in urban areas [1], especially in large and growing cities. As population and vehicle usage rise, so does the volume of traffic, leading to congested roads, long delays, and increased risk of accidents. However, the field of intelligent transportation systems aims to address these problems through strategic traffic control methods. Traditionally, this has meant optimizing traffic signal timing using rule based methods [2]. Rule based methods rely on fixed timing plans or control logic, often based on traffic flow data. The main drawback to these types of systems is the lack of adaptability; this can make handling unexpected traffic patterns more

challenging. As a way to address these issues, recent research into intelligent traffic control has focused on using machine learning methods.

Traffic control methods can be grouped into two main traffic control strategies, which are signalized and unsignalized traffic control. Signalized traffic control refers to only using traffic signals as a way to control the flow of traffic, whether it be a rule based or ML approach. An example of this could be to optimize traffic light patterns to reduce traffic flow.

Unsignalized traffic control typically refers to the control of traffic flow without traffic lights. Instead of using or optimizing traffic signals, traffic is coordinated through the actions of autonomous vehicles, also called robot vehicles (RVs), which interact with human driven vehicles (HVs). These RVs rely on machine learning methods to reduce congestion and enhance safety [3, 4, 5]. Mixed traffic is the term used to describe an environment that is comprised of both HVs and RVs. Optimizing flow in mixed traffic environments has become the focus of many studies. While using RVs to control traffic flow in a mixed environment has not yet been applied to the real world and has primarily only been tested in simulations or experimental settings, it could one day become a reality as autonomous driving technology continues to advance [3].

To address this need, researchers have explored reinforcement learning (RL), a subset of ML where autonomous vehicles learn optimal driving behaviors through trial and error [6, 4]. RL has become very useful in robotics and control tasks [7, 8]. When combined with graph neural networks (GNNs), RL enables agents to capture the spatial and relational dynamics of traffic [9, 10, 11]. This type of data is useful for traffic control and prediction applications.

In this work, we propose a framework, GAT-Dueling DQN, for unsignalized traffic control in mixed traffic scenarios. This approach uses a graph structure to model the spatial positions and interactions of vehicles to capture traffic behaviors. Our method will also make use of Graph Attention Networks (GATs) for spatial feature

extraction. The extracted features will then be passed into a Dueling Deep Q-Network (Dueling DQN).

Unlike standard DQN models, the Dueling DQN architecture separates state values from the advantages of given actions. This allows the agent to better differentiate between important and redundant actions, which can be useful in traffic scenarios where many actions could yield the same results.

Each robot vehicle (RV) selects actions independently using its own GAT-Dueling DQN module, which enables decentralized decision making. This differs from centralized control setups used in prior works [12, 13]. Our system is implemented in SUMO simulations, where it is tested on road topologies that consist of roundabouts and intersections. Intersections are commonly studied in this field, but roundabouts are less so.

1.3 Background and Motivation

Now, we will take a look at some common approaches and uses for reinforcement learning as well as Graph Convolutional Networks (GNNs).

1.3.1 Reinforcement Learning

Reinforcement Learning (RL) is a paradigm in which agents learn to take actions that maximize cumulative rewards through interaction with their environment. In traffic scenarios, this means learning to take actions that optimize metrics such as reducing waiting times and increasing safety.

One widely used reinforcement learning (RL) approach is Q-learning. Q-learning learns a state action value function, called a Q-function, the Q-function is learned by updating Q-values based on rewards by taking actions for given states. The traditional tabular Q-learning becomes challenging in high dimensional state or action spaces due to the curse of dimensionality, where the number of possible state action pairs grows

exponentially [6]. This is particularly problematic in traffic environments that can have large state action spaces. To address this limitation, Deep Q Networks (DQNs) were introduced, which use neural networks to approximate the Q function.

The Dueling DQN architecture [7] further extends this by splitting the Q-function into two separate estimators: one for the state value and one for the advantage of each action. By learning these components independently, Dueling DQNs can improve the agent’s ability to distinguish valuable actions, even when many choices seem equally good.

1.3.2 Graph Neural Networks

Graph Neural Networks (GNNs) expand neural networks to graph structured data, where inputs are not constrained to fixed size vectors but consist of nodes and their interconnected edges. The graph structure is suitable for modeling interactions in traffic systems, where vehicles naturally form relational features with each other. A GNN can aggregate information from its neighbor nodes in order to learn context aware representations.

A common variant of GNNs is the Graph Convolutional Network (GCN), which uses convolutions to extract graph features. In GCNs, node embeddings are updated through weighted averages of neighboring features. This enables the model to learn from both local attributes and structural patterns [14]. In traffic scenarios, GCNs have been applied to predict flow or aid in decision making across connected agents.

Building on this, Graph Attention Networks (GATs) introduce an attention mechanism to learn the importance of different neighbors rather than treating all connections equally [15]. GATs can enable each vehicle to selectively prioritize information from more relevant neighbors, such as those directly ahead or potentially about to collide. These attention weighted embeddings capture relational dependencies which could be useful for safe and efficient behavior in dense traffic.

1.4 Purpose of This Thesis

The purpose of this thesis is to explore and develop an intelligent traffic control method for unsignalized mixed traffic environments. This method is to be topology independent, meaning that it is not constrained to a specific road topology. Specifically, we focus on enabling decentralized decision making for robot vehicles (RVs) operating alongside human driven vehicles (HVs) in complex traffic scenarios such as intersections and roundabouts.

We plan to achieve this by combining Graph Attention Networks (GATs) with Dueling Deep Q Networks (Dueling DQNs), where the GAT is responsible for the feature extraction of the graph based traffic representation, and the Dueling DQN is responsible for control and decision making.

1.5 Thesis Outline

This thesis is organized into the following chapters:

- **Chapter 1: Introduction**

Discusses the traffic control problem. Introduces common reinforcement learning techniques as well as Graph Neural Networks.

- **Chapter 2: Literature Review**

Reviews previous research in traffic control. Looks at both traditional RL methods as well as deep RL methods. Also looks at the use of GNNs for traffic control.

- **Chapter 3: Methodology**

Introduces and discusses our proposed methodology for unsignalized mixed traffic control.

- **Chapter 4: Experimental Setup and Results**

Discusses the setup of training and experiments as well as results.

- **Chapter 5: Conclusion**

Concludes the thesis based on analysis of results.

1.6 Chapter Conclusion

This chapter establishes the need for more adaptive and intelligent traffic control systems. We examined the limitations of traditional rule based systems and introduced reinforcement learning and GNNs for traffic control scenarios. The rest of the thesis will build off of this as we explore related work, present our own methodology, and evaluate its effectiveness.

Chapter 2

Literature Review

2.1 Chapter Overview

This chapter provides a look at related works in the domain of traffic control, focusing primarily on reinforcement learning approaches and graph neural network (GNN) architectures. Here is a breakdown of the main research areas:

- Reinforcement learning based approaches to traffic control
 - Unsignalized Traffic Control
- Graph Neural Networks for Traffic Prediction and Control
 - Graph Convolutional Networks (GCNs)
 - Graph Attention Networks (GATs)

The section on reinforcement learning (RL) approaches explores traditional and deep reinforcement learning (DRL) frameworks used in traffic control. This includes value based and policy based methods, along with the comparison of model based vs. model free learning. This section also takes a look at advancements in DRL such as dueling architectures and prioritized experience replay. Next, we look at unsignalized traffic control, highlighting several approaches that use autonomous

vehicles to influence and coordinate traffic. Finally, we will take a look at the use of Graph Neural Networks (GNNs) in traffic prediction and control, considering both traditional Graph Convolutional Networks (GCNs) as well as Graph Attention Networks (GATs).

2.2 Reinforcement Learning Based Approaches to Traffic Control

To give an idea of the common approaches for the traffic control problem, let's look at a comprehensive study of traffic control strategies performed by Wei et al.[\[2\]](#). This paper identifies the ongoing challenges of urban traffic congestion and highlights the costs of inefficient traffic management on the environment, economy, and society. It details several traffic control methods ranging from more traditional rule based methods to more modern machine and reinforcement learning based methods. For the purposes of this thesis we will primarily look at the reinforcement learning based methods.

Reinforcement learning (RL) can be a very efficient way to optimize traffic control. Often, more efficient than traditional traffic signal control methods. This paper discusses several types of RL methods that could be used for traffic control.

The first is model based RL approaches, which explicitly build a transition model of the environment, allowing for more efficient exploration of potential traffic scenarios. By simulating different traffic conditions, it can plan future actions more effectively, leading to faster convergence in learning. However, model based approaches require the environment to be accurately modeled, which can be a challenge given how variable real world traffic can be.

Model free RL can learn optimal traffic control policies directly from experience without requiring an explicit model of the environment. It adaptively adjusts signals based on observed states with a given action and reward. This approach requires

a large amount of training data but is highly flexible and proven to deliver optimal results for a variety of traffic conditions. Two common examples of model free RL approaches are value based and policy based approaches. Value based approaches estimate the expected reward for different actions, and policy based approaches directly optimize the selection of actions.

A popular value based RL approach is Q-learning, which focuses on approximating the state action value function, allowing the system to determine which traffic signal adjustments are most rewarding. Q-learning can be an effective method but tends to struggle in higher dimensional scenarios.

Policy based approaches learn a direct mapping from states to actions, which can make them more adaptable in terms of decision making. An example of a policy based reinforcement learning method used for traffic control is the actor critic method. In this approach, the "actor" selects actions based on current policy estimates, while the "critic" evaluates the effectiveness of these actions and then improves upon them. This can enhance the stability and efficiency of learning, which makes it really useful for traffic control scenarios.

The study [6] provides a comprehensive overview of using deep reinforcement learning (DRL) in traffic signal control. It shows that traditional rule based and adaptive methods struggle to manage persistent urban congestion, prompting the integration of reinforcement learning (RL) techniques. The study highlights the limitations of traditional RL approaches, such as their susceptibility to the curse of dimensionality and their inefficacy in handling complex, dynamic traffic environments. To overcome these challenges, one common technique is for RL frameworks to integrate neural networks to approximate value functions and optimize traffic signal control strategies.

This same study also explores various DRL methodologies, emphasizing multi agent DRL frameworks for large scale traffic networks. These frameworks incorporate deep Q networks (DQNs), actor critic models, and synchronous advantage actor critic methods, which use coordinated decision making across multiple intersections. The

study also explores different traffic state representation techniques, including discrete time traffic state encoding (DTTSE) and convolutional neural network (CNN) based feature extraction, which enables more effective processing of raw traffic data.

The study also discusses open traffic data and its role in enhancing DRL applications. The study concludes that standardized, accessible traffic datasets are crucial for developing scalable and efficient DRL models. The integration of open traffic data with DRL techniques allows for more robust model training and evaluation, leading to better performing models and the possibility of improved real world applications.

The DRL frameworks detailed in [6] give an overview of hyper parameter tuning, model architecture design, and reward function optimization. It also shows the benefits of hierarchical and decentralized control architectures toward computational efficiency. This paper gives a great baseline for optimizing traffic control.

Other RL related techniques that help improve traffic control include the dueling network architecture and prioritized experience replay buffer. The dueling architecture, introduced in [7], breaks its estimation of the Q-function into an estimation of a state-value function and an estimation of an action-advantage function. This is supposed to enable more stable and efficient learning. This approach has been shown to yield superior performance in decision making tasks like traffic control, where differentiating between valuable and less valuable states is necessary.

Prioritized Experience Replay (PER), proposed in [8], attempts to improve learning efficiency by sampling experiences based on their importance to the agent instead of random sampling. PER uses temporal difference (TD) error to sample experiences in the buffer. TD shows how far off the agent’s prediction of the reward was from the actual outcome. This allows the model to focus on more informative experiences in order to aid in its training. This technique can be useful when used in multi agent traffic environments.

2.2.1 Unsignalized Traffic Control

Wang et al. [3] address the complexities of unsignalized mixed traffic control. This consists of traffic control scenarios where the flow of traffic is not controlled by traffic lights. Instead, in this case, robot controlled vehicles (RVs) are used to control the flow of human controlled vehicles (HVs). Their proposed method utilizes local observations and a shared reinforcement policy to enable RVs to make decisions such as stop or go. The desired goal is to reduce congestion and improve efficiency. They test their method on real world traffic data from Colorado Springs, CO, USA. The results show that even with smaller RV penetration rates (percentage of RVs), their method can reduce congestion under high traffic demands, which significantly outperforms traditional traffic light control. Their approach is also highly adaptable as it was able to perform well on five unseen intersections.

Xiao et al. [4] propose a model free reinforcement learning policy for mixed traffic control optimization. Their approach is the first to work on topology independent road structures, meaning the traffic control policy does not have to be designed around the constraints of a specific road topology. This method uses an ego RV (robot vehicle) to observe the relative position and velocity of both human driven and robot vehicles in front of and behind it. The focus is on optimizing longitudinal speed, as lateral movement is more challenging to optimize because it adds additional complexity. The soft actor critic algorithm is used to optimize the policy. The results show that their policy efficiently optimizes traffic flow at both intersections and roundabouts.

Another unsignalized approach to mixed traffic control is shown in [5]. It uses cooperative decision making, which focuses on the interaction between connected automated vehicles (CAVs) and human controlled vehicles (HVs). They propose a decentralized multi agent reinforcement learning approach, in the form of a multi agent proximal policy optimization (MAPPO) algorithm, which is enhanced with an attention mechanism (Attn MAPPO). Their goal is to improve decision making efficiency by allowing vehicles to extract and prioritize relevant information from

neighboring agents to optimize traffic behavior. Each vehicle makes decentralized decisions based on shared information. The attention module is incorporated to focus on the most relevant information from surrounding vehicles. The algorithm learns a joint policy that enables vehicles to anticipate potential interactions and cooperatively make decisions. It uses a weighted reward assignment scheme, which assigns greater rewards to vehicles approaching the intersection, ensuring that decision making prioritizes safety and efficiency. They evaluate their approach using comparative experiments against heuristic rule based models and traditional MARL approaches. The results indicate that their approach outperforms baseline methods in complex traffic scenarios. The attention based extraction method enhances the ability of CAVs to make informed decisions.

2.3 Graph Neural Networks for Traffic Prediction and Control

Graph Neural Networks (GNNs), which include Graph Convolutional Networks (GCNs) and Graph Attention Networks (GATs), are well suited for capturing spatial and temporal dependencies in traffic systems. These models represent intersections and road segments as nodes and edges in a graph, enabling structured learning from complex traffic networks.

Graph based reinforcement learning models have the ability to model spatial dependencies between intersections. For example, Devailly et al. [10] propose an inductive graph reinforcement learning method that uses node embeddings and graph convolutional layers to handle new traffic network configurations without retraining. Similarly, GPlight [9] integrates graph neural networks (GNNs) with deep reinforcement learning to capture topological relationships among intersections and dynamically adapt signal control strategies.

2.3.1 Graph Convolutional Networks (GCNs)

Graph Convolutional Networks perform feature aggregation by computing a weighted average of neighboring node features. These models assume a fixed adjacency matrix and treat all neighbors equally, making them suitable for capturing spatial dependencies when the road network topology is known.

[16] proposes a deep reinforcement learning framework for multi intersection traffic control using an edge weighted GCN encoder and a unified structural decoder. This EGU-RL model, integrates spatial relationships between intersections and leverages a reinforcement learning policy to optimize traffic signal timing. Their edge weighted GCN captures varying degrees of spatial influence based on road distances and connections, improving both accuracy and efficiency.

Another approach, Adaptive Graph Convolutional Recurrent Network (AGCRN) [14], enhances traditional GCNs by integrating Node Adaptive Parameter Learning (NAPL) and Data Adaptive Graph Generation (DAGG). NAPL assigns unique parameters to each node, allowing the model to capture intersection specific traffic patterns, while DAGG learns the adjacency structure dynamically using data driven node embeddings. Combined with Gated Recurrent Units (GRUs), AGCRN models both spatial and temporal patterns effectively, improving traffic flow forecasting.

This 2023 study [17] shows that using GCNs within a deep reinforcement learning framework allows agents to optimize traffic signal policies by utilizing both real time traffic conditions and historical data patterns.

2.3.2 Graph Attention Networks (GATs)

Graph Attention Networks (GATs) [15] use an attention mechanism into GNNs, allowing nodes to assign different importance levels to their neighbors during feature aggregation. They differ from GCNs in that they do not rely on fixed aggregation weights. Instead, GATs compute attention coefficients using self attention over graph structured data. These coefficients are dynamically learned using shared parameters

that account for both the target and source node features. This enables adaptive weighting and better captures relationships in a graph.

GATs are also particularly useful in traffic systems where node influence is not uniform and may change over time. For example, the Spatio Temporal Causal Graph Attention Network (STCGAT) [18] constructs spatial adjacency subgraphs at each time step without requiring a predefined road topology. This allows the model to adapt to evolving spatial relationships and better learn traffic flow dynamics.

Similarly, the Dynamic Spatial Temporal Graph Attention Network (DynST-GAT) [19] integrates temporal convolutional networks with graph attention layers to model spatial and temporal dependencies. This enables dynamic signal control in traffic environments with fluctuating traffic patterns.

In multi agent reinforcement learning settings, attention mechanisms have also shown benefits. The GAT-DDPG model [11] uses GAT layers within a decentralized deep deterministic policy gradient framework, enhancing cooperation among agents at multiple intersections through attention weighted communication.

Chapter 3

Methodology

3.1 Chapter Overview

In this approach to unsignalized mixed traffic control, we combine a Graph Attention Network (GAT) with a Dueling Deep Q-Network (DQN) architecture. This chapter will outline the methodology of our approach, foreseeing on the main components of our model. The three main components include the Graph Attention Network (GAT), Dueling Deep Q-Network (DQN), and prioritized experience replay (PER). This section will also explore our method for modeling traffic as a graph, training exploration, and our models reward function.

3.2 Graph Modeling

At each step of the simulation, a dynamic graph is created to represent the traffic environment as shown in figure 3.1 Each node represents a vehicle with a 5 dimensional feature vector: the x position, the y position, speed, orientation (angle), and waiting time.

The graph is defined as $G = (X, A)$, where $X \in \mathbb{R}^{N \times d}$ is the node feature matrix which represents the individual vehicle attributes. The adjacency matrix is represented by $A \in \mathbb{R}^{N \times N}$, where N is the number of vehicles. The edges between

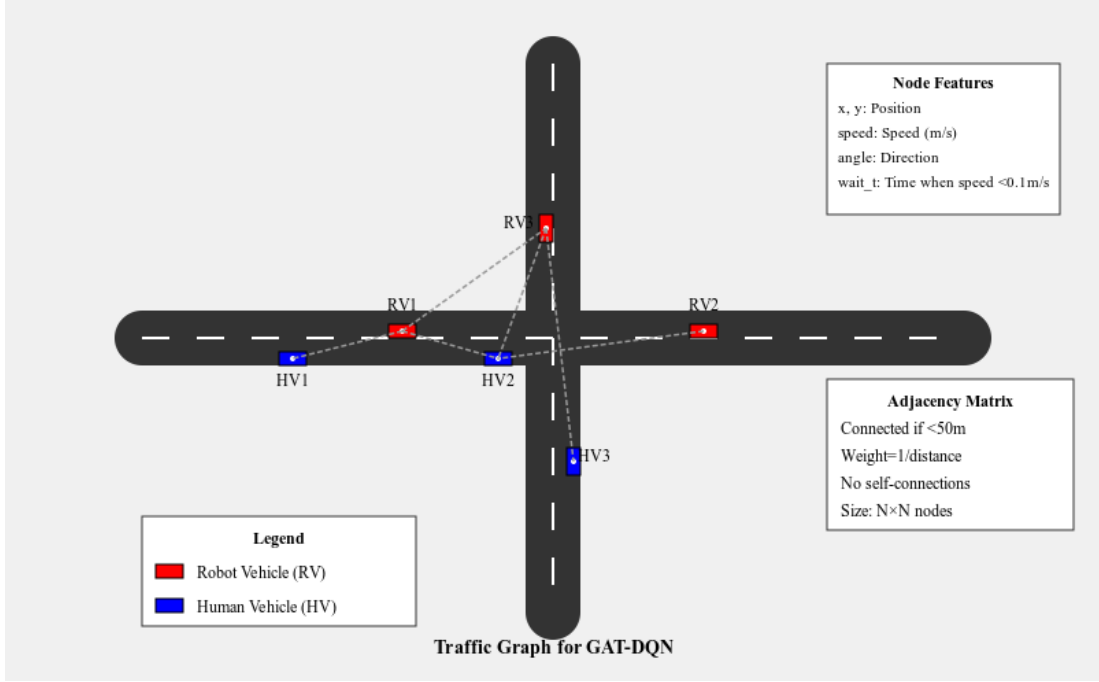


Figure 3.1: Representation of the traffic graph.

nodes represent the distance between two vehicles. An edge exists if the distance between two vehicles is below a fixed threshold (less than 50 m). Edges are weighted in an inversely proportional relationship to the distance, which encourages the model to emphasize information from closer vehicles.

The graph is assembled at every simulation step to reflect the evolving traffic flow. It is then passed to the GAT-Dueling DQN model for action selection. Our approach to building the graph is inspired by prior works [10, 9].

3.3 Graph Attention Network

Once we have traffic modeled as a graph, the graph’s node feature and adjacency matrices are passed to the Graph Attention Network (GAT). The GAT [15] is used to enable attention based aggregation of nodes in the traffic graph. For each node i and its neighbor j , a shared attention mechanism calculates a scalar attention coefficient e_{ij} , which shows how important a given node j ’s features is to a given node i :

$$e_{ij} = \text{LeakyReLU}(a^\top [Wh_i \parallel Wh_j]) \quad (1)$$

Here, h_i and h_j are the feature vectors of nodes i and j , respectively; W is a learnable weight matrix, and a is a learnable attention vector. The notation $[\cdot \parallel \cdot]$ denotes concatenation of transformed features. This mechanism is based on the formulation originally introduced by Veličković et al. [15] and was used in traffic signal control [6].

The coefficients e_{ij} are normalized across all neighbors of a given node i using the softmax. The updated node embedding is obtained by applying these attention weights to the transformed features of neighboring nodes:

$$h'_i = \sigma \left(\sum_{j \in \mathcal{N}_i} \alpha_{ij} Wh_j \right) \quad (2)$$

All together this process is as follows: each attention layer performs a linear transformation, computes attention scores using Equation (1), masks the scores using the adjacency matrix, and applies softmax. The result is used to compute weighted feature aggregation per Equation (2).

The model stacks two such GAT layers, with 8 attention heads in the first layer with each producing 64 dimensional outputs. A single head is used in the second layer for consolidating the outputs. Layer normalization is used, as commonly adopted in traffic control tasks involving GNNs [5, 4].

The purpose of the GAT structure is to allow the network to focus selectively on the most influential neighboring vehicles, which should ideally improve spatial context awareness in the mixed traffic environment.

3.4 Dueling DQN

The feature embeddings extracted by the GAT attention layers are passed into a Dueling Deep Q-Network (Dueling DQN). Instead of a standard DQN, the dueling

architecture is split into two separate fully connected streams after the shared feature representation. The first stream is responsible for estimating the state-value function $V(s)$, which computes how desirable a specific state is, regardless of the action taken. The second stream estimates the advantage function $A(s, a)$, which determines how beneficial it is for a given action to be taken in a given state compared to other actions. This separation should allow the network to more effectively learn value estimates, especially in scenarios where many actions result in similar outcomes, such as a traffic environment. The final Q-values are computed by combining the value and advantage outputs as follows [7]:

$$Q(s, a) = V(s) + \left(A(s, a) - \frac{1}{|\mathcal{A}|} \sum_{a'} A(s, a') \right) \quad (3)$$

The advantage stream is defined as a sequence of linear layers with a ReLU activation, projecting the final graph embedding to an $A(s, a)$. Similarly, the value stream projects to a scalar $V(s)$. These outputs are broadcast and combined as in Equation (3) to produce Q-values per each robot vehicle (RV).

This approach allows each RV to receive individualized Q-value estimates from a shared model, allowing them to act independently while maintaining shared knowledge across the traffic network. The design promotes decentralized yet coordinated behavior in mixed traffic settings, which can be useful in unsignalized environments with high interaction [4, 5].

3.5 Prioritized Experience Replay (PER)

Our approach also uses a prioritized experience replay (PER) buffer [8]. Which is a refinement of the standard experience replay used in reinforcement learning. A typical experience replay buffer stores past interactions of the agent such as the state it was in, the action it took, the reward it received, as well as the resulting next state. The agent samples randomly from this buffer during training. This random sampling

ensures a diverse and uncorrelated training set where all experiences are considered equally important.

The PER improves upon this by prioritizing experiences that the agent is currently less experienced in handling. Temporal difference (TD) error is used to measure this. TD shows how far off the agent’s prediction of the reward was from the actual outcome. Transitions that have high TD errors are sampled more frequently, as they suggest the model still has more to learn. To achieve this, each transition is assigned a priority score computed as:

$$\text{priority}_i = (|\delta_i| + \varepsilon)^\alpha$$

where δ_i is the TD error, ε is a small constant, and α determines the level of prioritization. This formulation is based on the method proposed by Schaul et al. [8].

A problem that arises when prioritizing certain samples over others is that a bias is introduced due to changing the distribution of experiences the model learns from. To correct for this, importance sampling (IS) weights are used during training. These weights help re-balance the learning updates in order to reduce the bias.

3.6 Reinforcement Learning Components

3.6.1 Action Space

The action space used in this approach is defined as a discrete set with three longitudinal control actions: Accelerate, Decelerate, and Maintain Speed. These actions directly influence the vehicle’s speed using fixed increments. Lane changes and lateral control decisions are left to the SUMO simulator’s default driving behavior, which follows car following models and traffic rules. This helps simplify the control task and aligns with prior work in deep reinforcement learning for traffic [6, 3]. Focusing the agent’s learning on longitudinal movements only, reduces the dimensional complexity of the action space.

3.6.2 Epsilon Greedy Policy

A ϵ -greedy policy is used to balance exploration and exploitation. In reinforcement learning exploration refers to trying new actions to see what effects they have, while exploitation refers to using known actions that will yield good rewards. At each decision step, the agent selects a random action, and the action with the highest Q-value with probability. Based on a given exploration rate the agent will either choose the random action (exploration) or the action with the highest Q-value (exploitation). The higher the exploration rate the more likely the agent is to try random actions. The exploration rate decays over the training episodes, enabling exploration during early learning while favoring exploitation later one.

3.6.3 Reward Function

The reward function used in our approach encourages smooth and efficient driving behavior while penalizing traffic for waiting and unsafe decisions. At each timestep t , the reward for a robot vehicle is calculated as:

$$r_t = w_{\text{speed}} \cdot \frac{v_t}{v_{\text{max}}} - w_{\text{wait}} \cdot \text{wait}_t - w_{\text{collide}} \cdot \text{collision}_t \quad (3)$$

Here, v_t is the vehicle's current speed and v_{max} is the maximum speed, forming a speed ratio that rewards faster movement. The wait_t term captures waiting time, discouraging congestion. The collision_t term is a binary penalty applied if the agent is involved in a collision. This reward function aligns with formulations seen in recent literature on reinforcement learning in traffic control [4, 3, 11].

The weights w_{speed} , w_{wait} , w_{collide} are used to prioritize which parts of the reward should affect the model the most. Speed and waiting times were prioritized the most in our implementation.

3.7 Chapter Conclusion

This chapter presented our methodology behind our approach for unsignalized mixed traffic control using a GAT-Dueling DQN architecture. We discuss each of the major components of the model. Which includes the GAT, Dueling DQN and PER. We also discuss are method form formulating traffic in a graph structure and how it is represented to our model. Then, finally discussing our reinforcement learning approaches such as the epsilon greedy policy and our reward function.

Chapter 4

Experimental Setup and Results

4.1 Chapter Overview

This chapter presents the experimental setup and the results of the evaluation of our model. The purpose of these experiments is to test the model’s ability to improve traffic efficiency and reduce vehicle wait times under realistic traffic conditions. This chapter will outline the simulation environment, dataset, and training setup, used to train our model. Followed by an analysis of our models performance. Both seen (training) and unseen (test) maps are included in the evaluation. We also conduct an ablation study to analyze the impact of the model’s individual components. Such as the GAT, Dueling DQN, and prioritized replay buffer. Based on these information gathered from the ablation study we then to to implement models that incorporate the best performing parts of the ablation study, and evaluate their performance.

4.2 Experimental Setup

4.2.1 Simulation Environment

Training and testing of the model were conducted using the SUMO (Simulation of Urban Mobility) traffic simulator. SUMO is an open source platform for simulation

and modeling of urban traffic dynamics. This will help us train and evaluate our model in real-world scenarios.

4.2.2 Dataset Description

In order to represent realistic traffic environments, maps were selected from the benchmark dataset that was proposed by Xiao et al. [4]. This benchmark dataset contains 444 traffic scenarios derived from real-world topologies, including both intersections and roundabouts. Each topology is categorized into the three difficulty levels, easy, medium, and hard. The difficulty is based on increasing traffic demand (vehicles per hour).

Table 4.1 provides an in depth breakdown of the benchmark dataset used for training and evaluation. As shown in the table, the easy level is generally characterized by simple topologies (3-4 legs, 1-2 lanes) and low traffic demand (400-1500 veh/hr), the purpose of this is to test basic coordination in the conditions of little traffic congestion. The medium level features more complex topologies (4-6 legs, 2-3 lanes) with a larger traffic flow (1500-3000 veh/hr), this challenges how well the model can adapt to tougher levels of congestion. The hard level includes the most challenging environments with even more complex topologies than the previous difficulty (7+ legs, multiple lanes) and dense traffic (3000-5000 veh/hr), this difficulty is can show how well the model can avoid deadlock. Deadlock is when the traffic comes to a complete standstill and nothing is moving. Deadlock is much more likely in higher congested environments. It is possible to encounter deadlock at the easier level, but is more frequent at the medium and hard levels.

4.2.3 Scenario Selection

For the purposes of training, a subset of the benchmark dataset was selected. The subset included five intersection and five roundabout maps. For each of the selected maps, all three difficulty versions of the map were used. These resulted in a total

Table 4.1: Benchmark Dataset Summary

Difficulty	Topology	Traffic Demand	Key Challenges
Easy low congestion	3–4 legs, 1–2 lanes	400–1500 veh/hr	Basic coordination
Medium moderate congestion	4–6 legs, 2–3 lanes	1500–3000 veh/hr	Adaptive control
Hard dense congestion	7+ legs, high lane counts	3000–5000 veh/hr	Deadlock avoidance

of $5 \times 3 + 5 \times 3 = 30$ training scenarios. This ensures that the model will learn a variety of topologies on varying degrees of traffic demands. Several roundabouts and intersections of varying difficulties were used to evaluate the effectiveness of the model. Figure 4.1 shows an example roundabout scenario.

4.3 Model Training

4.3.1 Training Setup and Configuration

Our GAT-DQN model was trained on the subset benchmark dataset. During training, the robot vehicle (RV) rate was fixed at 0.70, meaning that 70% of vehicles were under the control of the GAT-DQN agent, while the remaining HVs followed the human-like Intelligent Driving Model (IDM) which is enabled by SUMO. This RV rate was chosen because it allows enough RV vehicles to have influence over the traffic while not completely removing the effect of the HVs.

Vehicles were spawned using a probabilistic demand based approach to simulate realistic traffic flow. Each entry lane was assigned a predefined vehicle arrival rate (vehicles per hour), which varied depending on the scenario difficulty. Vehicles were

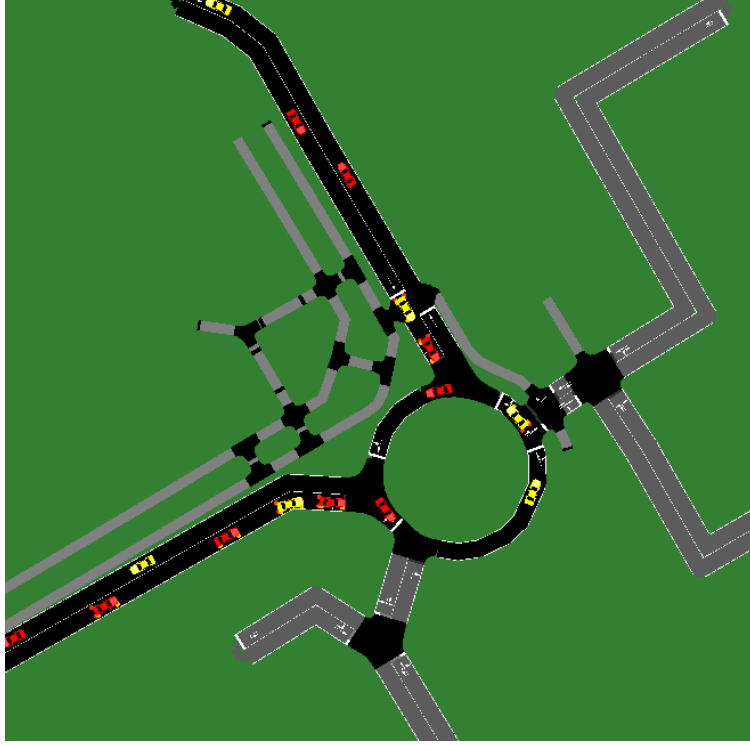


Figure 4.1: Example of dataset SUMO map.

introduced into the network at randomized intervals using and randomly assigned to be an RV or HV based in the fixed RV rate. Similar approaches are used in [3, 4].

Table 4.2 shows some of the hyperparameter used to train the model. It was trained for 100 episodes, with each episode lasting for a specified number of simulation steps. In this case the step size is 3000. The hidden dimensions which are the dimension of the hidden layers in the network were set at 128, 64, and 32 respectively. The discount factor determines how much the agent weighs current rewards versus future rewards. This was set at 0.70 to favor long term rewards. The exploration rate, which determines the trade off between trying new actions, or sticking to already known actions. The exploration rate is set to 0.8 to favor exploring new actions.

Table 4.2: GAT-DQN Training Hyperparameters and Configuration

Parameter	Value / Description
Episodes	100
RV Rate	0.70
Hidden Dimensions	[128, 64, 32]
Batch Size	32
Discount Factor (γ)	0.70
Exploration Rate (ϵ)	decaying from 0.8 to 0.0001
Step Size	3000 steps

4.3.2 Training Performance

Figure 4.2 gives a representation of the GAT-DQN model’s training progress. This particular model was trained on a single medium roundabout map for 100 episodes with an RV rate of 0.70. It was evaluated on four different training metrics: reward, average vehicle speed, average waiting time, and vehicle throughput.

The reward plot shows the reward the model receives for each episode. In the earlier episodes the model shows severe drops reaching as low as -175,000. These huge negative spikes likely correspond to episodes that faced severe gridlock or congestion. However, the model is able to recover from this as spikes in the later episodes are not as low, and it even gets rewards above zero. This suggests that it is learning from these failures.

The vehicle speed fluctuates heavily throughout training. The variable pattern suggests the model experiences both successful episodes where it is able to move traffic along somewhat fast and challenging episodes where traffic slows significantly—likely due to congestion or deadlock. There is no clear trend in the vehicle speed, which may indicate that the model is prioritizing minimizing waiting time over optimizing vehicle velocity. It could also be that the model has not yet discovered an optimal cruising speed under mixed traffic conditions.

The average wait time plot also shows heavy spikes reaching as high as 120 seconds in early training but seems to become more controlled as the episodes progress. Later

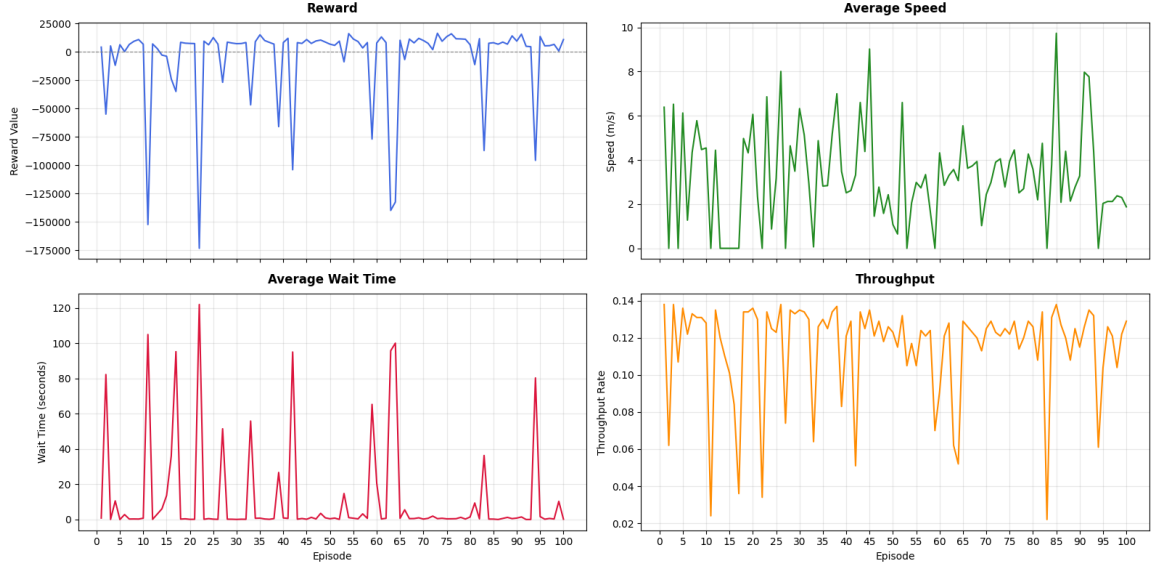


Figure 4.2: Training performance measured by reward, average speed, wait time, and throughput over 100 episodes.

episodes show more stable wait times staying below 10–20 seconds. Though spikes still occur, they are less frequent. This trend mirrors the reward metric, indicating that the model is learning to reduce congestion and wait time over time.

Throughput remains relatively stable around 0.12–0.14, with periodic dips to as low as 0.02. This consistency suggests that the model maintains a reasonable flow of vehicles even when other metrics fluctuate. However, the occasional steep drops indicate moments when traffic flow is severely restricted, likely due to deadlock or poor action selection.

These combined training metrics reveal a somewhat progressive improvement in the training process. The model appears to learn effective strategies over time but still occasionally encounters episodes with significant performance degradation. This highlights the stochastic nature of the environment and the importance of robust policy learning.

4.4 Results and Analysis

4.4.1 Performance on Training Roundabout

We first evaluated the model on a roundabout map that the model was trained on. This is to see how the model performs on a map it has been introduced to before. Figure 4.3 shows the model’s performance on said roundabout with all three difficulty levels.

For easy scenarios, there are very low wait times with a median around 0.7 and a small inner quartile range (IQR) of around 0.5 to 1.8. This is great performance but, isn’t anything unexpected as the model only has to deal with low congestion.

The medium difficulty has increased wait times with a median of around 3.0 and a wider variance with an IQR of around 2.0 to 4.9. This indicates that the model struggled more with the moderate traffic conditions as opposed to the easier ones which is to be expected.

The hard difficulty actually shows better performance than medium difficulty, with a lower median wait time of around 2.0 and smaller IQR of around 1.7 to 2.8.

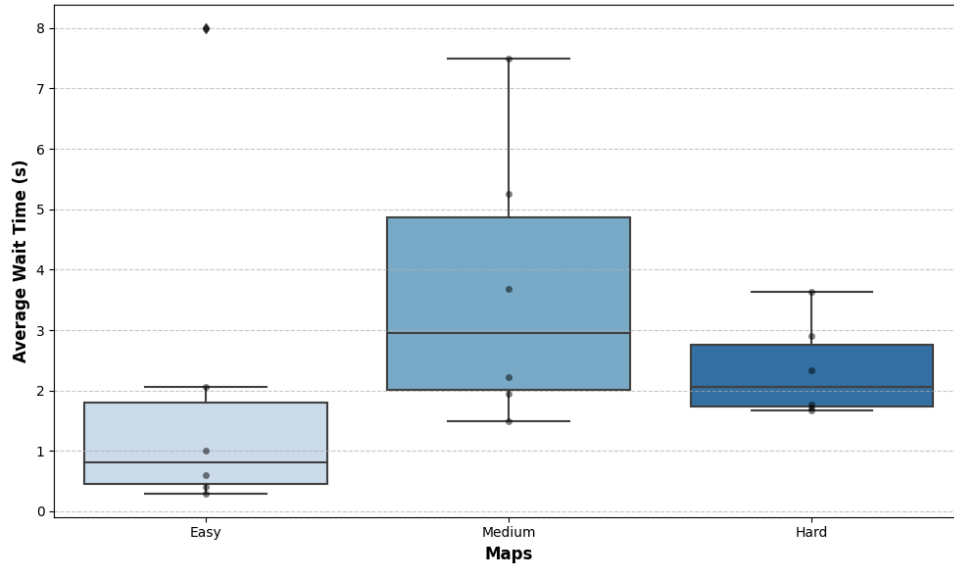


Figure 4.3: Average wait time of GAT-DQN across four different roundabout maps.

This unexpected result shows that the model may have learned better on the hard version of this map rather than the medium one. This could be because the hard difficulty came later in its training, after the model would have already learned the fundamentals on the easy and medium difficulties.

The consistently low wait times across all difficulties demonstrate that the model had learned efficiently and can preform well on a familiar environment.

4.4.2 Performance on Untrained Maps

After evaluating the performance of the model on a map it was trained on, we will now evaluate it against maps it has not seen before. Figure 4.4 shows the model's performance on an intersection map that was not in the training set. Again, it is being evaluated across all three difficulty levels.

For Easy, the model shows good performance with a median wait time of 2.5 and consistent results with a small IQR of around 2.0 to 3.0.

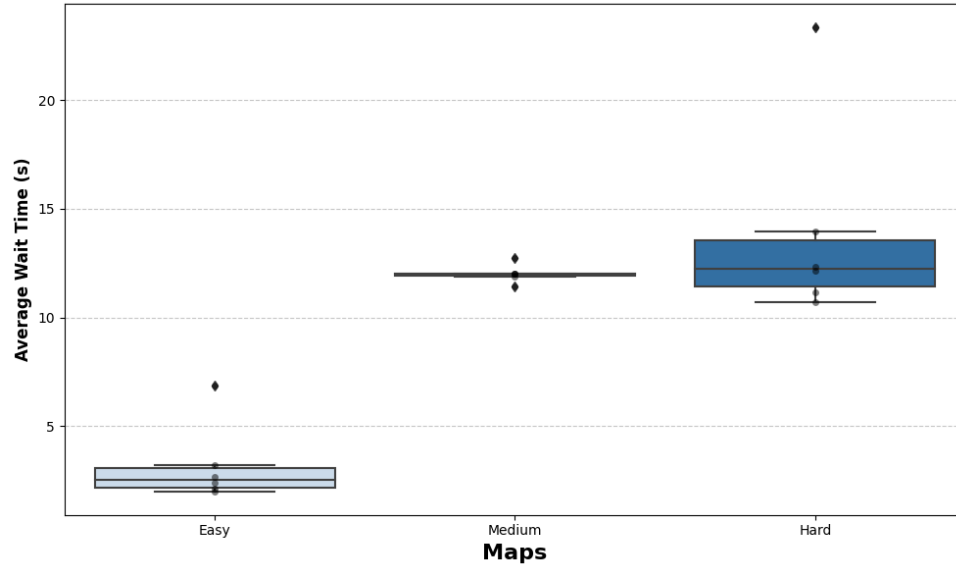


Figure 4.4: Average wait time on an intersection map across all difficulty levels.

With medium difficulty, the model shows increased but still reasonable wait times with a median of around 12.0. The results are very tightly clustered together which indicates good performance under moderate levels of congestion.

At the Hard difficulty level, the model exhibits the highest wait times with a median of around 12.5, but these are still well performing wait times. The increase in wait times for each difficulty is expected, but it differs from the results of the map from the training set, shown in figure 4.3.

The increasing wait times across difficulties show the challenge of managing traffic under higher density. The presence of outliers in both Easy and Hard scenarios suggests some edge cases where the model struggles to maintain optimal flow, this could be due to vehicle arrival patterns that create temporary difficulties in coordination.

Like the previous evaluation, Figure 4.5 shows the models performance for all difficulties on an unseen roundabout map.

For the easy difficulty, the model shows exceptional performance with near zero wait times across all episodes.

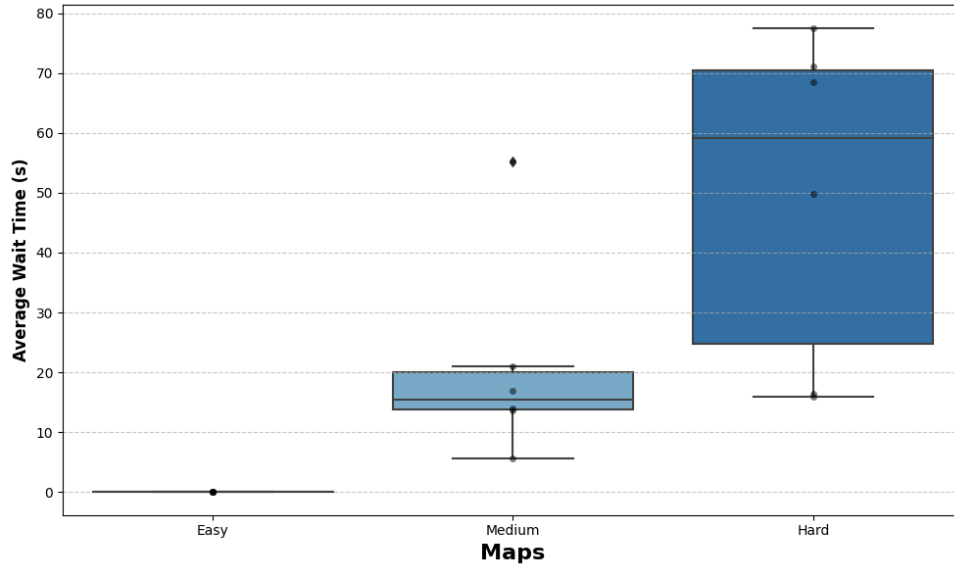


Figure 4.5: Average wait time on a roundabout map across all difficulty levels.

Medium shows that there are moderate wait times with a median of around 15.0. It generally performs worse than the medium difficulty on the intersection map in figure 4.4. This could be an indication that the model is not as effective on roundabouts.

With hard, there are dramatically increased with a median around 60.0. This is the models worse performance so far. It had a difficult time managing the high density roundabout traffic. The wide spread seems to suggest that the model performed inconsistently the entire time.

The stark contrast between Easy and Hard difficulty levels that the model tends to struggle more with roundabouts. Specifically in very dense traffic. The extreme variance in Hard difficulty indicates that the model’s performance becomes highly sensitive to specific traffic patterns in complex roundabout scenarios, suggesting potential limitation to the current approach.

4.4.3 Quantitative Comparison

Table 4.3 provides a quantitative comparison against baseline performance (i.e., no GAT-DQN control). The GAT-DQN model demonstrates significant improvements in average wait times for both intersection scenarios: a 76.1% reduction at medium difficulty and a 24.5% reduction at hard difficulty. On the roundabout map, the model also achieves a strong improvement at medium difficulty (40.7%), suggesting that it can effectively learn coordination strategies in moderately congested circular topologies.

However, the model under performs in the roundabout hard scenario, increasing average wait time by 70.3% relative to the baseline. This indicates potential challenges in handling highly congested roundabouts, which may involve complex multi-agent interactions and deadlock risks that the current GAT-DQN architecture fails to resolve effectively. This case highlights an area for future work, such as incorporating explicit conflict resolution mechanisms or hybrid policy architectures.

Table 4.3: Average Wait Time and Relative Improvement for GAT-DQN

Map Type	Difficulty	Baseline (s)	GAT-DQN (s)	Improvement (%)
Intersection	Medium	50.2	12.0	76.1%
Intersection	Hard	14.7	11.1	24.5%
Roundabout	Medium	35.6	21.1	40.7%
Roundabout	Hard	29.3	49.9	-70.3%

Overall, the results validate the effectiveness of GAT-DQN in most tested scenarios, while also emphasizing the importance of robustness under extreme traffic conditions.

4.5 Ablation Study

After a general analysis of our model’s performance, we now conduct an ablation study to better understand which components contribute most to the effectiveness of the proposed architecture, and which might be improved or reconsidered. To do this, we trained and evaluated three variants of our full model: a GCN-Dueling DQN with no attention mechanism (No GAT), a GAT-DQN without the dueling architecture (No Dueling), and a GAT-Dueling DQN with a standard uniform replay buffer in place of prioritized experience replay (No PER). Each model was tested across a diverse set of medium and hard maps from our dataset, featuring a range of intersection and roundabout topologies. Easy maps were excluded from evaluation due to negligible variation in performance across models.

The results of this ablation study are shown in Table 4.4, which shows the average vehicle wait time of all models across all tested maps. All three ablated models actually outperformed the full model. The GAT-Dueling DQN without prioritized experience replay (No PER) performed the best, with an average wait time of 152.92 seconds, compared to 186.76 seconds for the full model.

This suggests that prioritized sampling may have actually been a hindrance to the model. This hindrance could be in the form of a bias towards experiences with high

Table 4.4: Average Wait Time Across All Maps (Ablated Models)

Model	Average Wait Time (s)
Full	186.76
No GAT	169.50
No Dueling	161.36
No PER	152.92

temporal difference (TD) error that ended up not being representative or informative in the long term. A standard replay buffer that randomly samples experiences may be preferred for this approach.

The model without the dueling component (No Dueling) also showed strong performance, with an average wait time of 161.36 seconds. This indicates that the separation of value and advantage functions may not be necessary for this approach.

Removing the attention mechanism (No GAT) resulted in a slightly higher average wait time of 169.50 seconds, which was still lower than the full model. This shows that the GAT enhances the ability to reason over topological dependencies, the model without it still captures enough information. Which could be due to the inherent graph structure of the environment and node level features to make reasonably effective decisions.

Overall, these findings indicate that a better approach to this problem could be a GAT-Dueling DQN with a standard replay buffer or simply a standard GCN-DQN.

4.6 Improving Model With Ablation Results

Now, we are going to create models based on the results of the ablation study to see how well they performed. We are going to focus on two. Those being a standard GCN-DQN with no attention mechanism, no dueling architecture, and a standard esperance replay. These models are based on the best performing aspects of the ablation study.

Table 4.5 shows the results of the new models. It appears that that no PER significantly improves performance of the GAT-Dueling DQN as it performs much better compared to the full model in table 4.4. The no PER model got a total average wait time of 171.89 across all maps, while the full model got an average wait time of 186.76. This shows that random experience sampling is better for mixed traffic environments, than prioritized sampling likely due to a bias that can arise with experiences that have high TD error.

Still, the GAT-Dueling DQN underperformed. The standard GCN-DQN model performed slightly better, achieving the lowest average wait time across all maps. While the difference is not as drastic as observed in the ablation study, it still suggests that the added complexity of the attention mechanism and dueling architecture may not provide significant advantages in this particular setup. One possible explanation is that the simpler GCN-DQN model generalizes better in mixed traffic environments by avoiding the complexity of additional components. The GAT may also struggle to consistently assign meaningful attention weights in these traffic scenarios. These results suggest that in certain traffic control scenarios, simplified architectures may sometimes yield more reliable performance.

4.7 Chapter Conclusion

The results from this chapter provide a comprehensive evaluation of the GAT-Dueling DQN architecture across a range of traffic scenarios. Overall, the model performs well in low to moderate traffic environments, particularly on maps it has encountered during training.

Table 4.5: Average Wait Time Across All Maps (Improved Models)

Model	Average Wait Time (s)
GAT-Dueling DQN	171.89
GCN-DQN	164.50

However, performance varies on unseen maps, especially in roundabout scenarios with higher density traffic (medium or hard difficulties), where wait times increased significantly. These findings suggest that the model is learning effective control strategies but, it still struggles under more extreme conditions.

The ablation study revealed that some architectural components may actually hinder performance. The main one being the prioritized experience replay buffer (PER), which was likely due to bias caused by sampling based on TD error. The performance gains offered by the attention mechanism and dueling architecture were marginal or inconsistent. To further explore this the we developed simplified models, such as the standard GCN-DQN with stand replay and the GAT-DQN with standard replay. Both outperformed the original full model.

Chapter 5

Conclusion

5.1 Summary and Contributions

This thesis presented a graph based deep reinforcement learning approach for mixed traffic control in unsignalized environments. Consisting of topologies that include both intersections and roundabouts. The proposed method utilized a GAT-DQN architecture, where a Graph Attention Network (GAT) module encoded the traffic environment as a graph and a Dueling Deep Q-Network (DQN) component selected control actions for robot vehicles (RVs). Vehicles were modeled as nodes in a graph, with edges representing relevant spatial relationships, allowing the model to learn in a topology independent way.

The system was trained and evaluated in the SUMO traffic simulator using maps derived from realistic intersection and roundabout layouts. From the dataset proposed in [4]. Through experimentation across multiple difficulty levels the model demonstrated the ability to improve traffic efficiency, in certain scenarios by reducing vehicle wait times. Notably, the model’s performance was more consistent in simple and moderately complex scenarios. The model still needs to be improved to adapt to more complex scenarios.

An ablation study was conducted to evaluate the contribution of key components: the GAT, the Dueling DQN, and prioritized experience replay. The ablation study revealed that simpler configurations such as no GAT, no Dueling DQN, and no prioritized experience replay (PER) tended to perform better. This suggests that complexity in model design does not always translate to better performance. The ablation study also showed that the component that contributed to the model’s worst performance was the PER. Likely due to sampling bias, the standard experience replay with random sampling seemed to be a better fit for our model.

The experimental results demonstrated that while the GAT-DQN approach does achieve traffic congestion improvements for certain scenarios and conditions, its performance typically decreased in higher density. Nevertheless, the model was able to reasonably generalize across unseen maps creating a topology independent solution for intersections and roundabouts, supporting the use of graph based representations in reinforcement learning for traffic control.

This work demonstrates the combination of graph neural networks with deep reinforcement learning for topology independent traffic control. While there are challenges that remain, especially in complex and congested scenarios, and improvements to be made to the model, the methods shown in this thesis show potential for adaptive traffic control.

Bibliography

- [1] D. Schrank, B. Eisele, T. Lomax, and J. Bak, “2019 urban mobility report.” Texas A&M Transportation Institute, 2019. [1](#)
- [2] H. Wei, G. Zheng, V. Gayah, and Z. Li, “A survey on traffic signal control methods,” 2020. [1](#), [8](#)
- [3] D. Wang, W. Li, L. Zhu, and J. Pan, “Learning to control and coordinate mixed traffic through robot vehicles at complex and unsignalized intersections,” *The International Journal of Robotics Research*, Oct. 2024. [2](#), [11](#), [19](#), [20](#), [25](#)
- [4] C. Xiao, D. Wang, X. Tang, J. Pan, and Y. Ma, “Optimizing efficiency of mixed traffic through reinforcement learning: A topology-independent approach and benchmark,” 2025. [2](#), [11](#), [17](#), [18](#), [20](#), [23](#), [25](#), [36](#)
- [5] J. Liu, P. Hang, X. Na, C. Huang, and J. Sun, “Cooperative decision-making for cavs at unsignalized intersections: A marl approach with attention and hierarchical game priors,” 2024. [2](#), [11](#), [17](#), [18](#)
- [6] M. Gregurić, M. Vujić, C. Alexopoulos, and M. Miletić, “Application of deep reinforcement learning in traffic signal control: An overview and impact of open traffic data,” *Applied Sciences*, vol. 10, no. 11, 2020. [2](#), [4](#), [9](#), [10](#), [17](#), [19](#)
- [7] Z. Wang, T. Schaul, M. Hessel, H. Van Hasselt, M. Lanctot, and N. De Freitas, “Dueling network architectures for deep reinforcement learning,” *International Conference on Machine Learning (ICML)*, 2016. [2](#), [4](#), [10](#), [18](#)

- [8] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, “Prioritized experience replay,” *International Conference on Learning Representations (ICLR)*, 2016. [2](#), [10](#), [18](#), [19](#)
- [9] Z. Hu, S. Xu, L. Li, and S. Jin, “Gplight: Integrating graph neural networks with deep reinforcement learning for traffic signal control,” 2020. [2](#), [12](#), [16](#)
- [10] J. Devailly, A. de La Fortelle, and F. Moutarde, “Inductive graph reinforcement learning for traffic control,” 2020. [2](#), [12](#), [16](#)
- [11] X. Yang and Others, “Gat-ddpg: Multi-agent traffic signal control using graph attention networks,” *Journal of Transportation Research Board*, 2025. [2](#), [14](#), [20](#)
- [12] T. Zhou and L. Sun, “Reasoning graph reinforcement learning for mixed autonomy at unsignalized intersections,” *IEEE Transactions on Intelligent Transportation Systems*, 2023. [3](#)
- [13] J. Klimke *et al.*, “Learning to negotiate intersections in mixed-autonomy traffic using graph neural networks,” *IEEE Transactions on Intelligent Vehicles*, 2023. [3](#)
- [14] L. Bai, L. Yao, C. Li, X. Wang, and C. Wang, “Adaptive graph convolutional recurrent network for traffic forecasting,” 2020. [4](#), [13](#)
- [15] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, “Graph attention networks,” *International Conference on Learning Representations (ICLR)*, 2018. [4](#), [13](#), [16](#), [17](#)
- [16] H. Wang, H. Chen, Q. Wu, C. Ma, and Y. Li, “Multi-intersection traffic optimisation: A benchmark dataset and a strong baseline,” *IEEE Open Journal of Intelligent Transportation Systems*, vol. 3, pp. 126–136, 2022. [13](#)
- [17] A. C. Hang Xing and X. Zhang, “Traffic flow prediction with gcn and reinforcement learning,” *ScienceDirect*, 2023. [13](#)

- [18] Y. Zhang, J. Guo, J. Wang, *et al.*, “Stegat: Spatio-temporal causal graph attention network for traffic flow prediction,” *arXiv preprint arXiv:2203.10749*, 2022. [14](#)
- [19] Y. Lin, X. Wang, J. Tang, and W. Wang, “Dynstgat: Dynamic spatial-temporal graph attention network for traffic signal control,” *arXiv preprint arXiv:2109.05491*, 2021. [14](#)

Vita

Zachary Sublett was born and raised in Kingsport, Tennessee. He choose to study Computer Science at Tennessee Technological University where he received his bachelor's degree. Upon graduation from Tennessee Tech he decided to continue his eduction by pursuing a master's degree in Computer Science at the University of Tennessee Knoxville, focusing his studies on the applications of machine learning and AI.