**Dynamic Traffic Light Control with Deep Q-Learning Algorithm**

Erenay Arsal

Manisa Celal Bayar University, TR-45030 Manisa, Turkey

**Abstract**

The project's objective is to harness and implement machine learning methods for better traffic flow control at crossroads that depend on traffic light systems. Considering that experimentation with real roads and traffic lights involves legal and practical inoperability, a reasonable simulation of actual traffic scenarios was created for this project. This environment would be interfaced with live camera feeds and advance image processing for real-time data collection and interaction with the AI model. The model follows reinforcement learning by considering the density of the traffic and modifies the light sequence accordingly. The main purpose is to minimize waiting times and optimize the flow of vehicles at every juncture.

The first approach was in developing the AI model with real-time data and its observation in optimizing traffic management. But there were a specific development challenge, with the reward function, that sometimes corrupted the decision-making process of the AI. Several iterations were made towards improving the model and the reward function in order to have an accurate traffic control system. With the solution of such problems, the system will be able to realize the prospect that AI-powered traffic management embeds for urban mobility and reduced environmental impacts including fuel and emission wastes. This project fosters efficient, sustainable, and scalable traffic solutions. Finally, the reward function performs effectively within the model and fulfils its purpose. The model operates efficiently and has been calibrated based on past experiences, yielding promising results.

Keywords- DQN, Traffic Lights management with DQN

# **Introduction**

Traffic congestion is a growing concern with significant environmental and economic impacts. Increased fuel consumption, carbon emissions, and commuting costs are just some of the consequences. Traditional traffic control systems, unable to adapt to dynamic conditions, exacerbate these issues, emphasizing the need for modern, adaptive solutions.

The following project uses machine learning, more specifically reinforcement learning, to optimize the flow of traffic light systems using real-time data. It incorporates live feeds from cameras and image processing to dynamically alter the sequences of lights with the aim of minimizing wait times and subsequent emissions. A simulated environment is used for training and testing the model due to many legal and logistical barriers.

Key challenges included designing a robust reward function to ensure effective decision-making and training the model to adapt to unpredictable traffic scenarios. The goal is to provide a scalable, sustainable traffic management solution aligned with global sustainability goals, particularly SDG 11 and SDG 13. Future work will focus on real-world implementation and refinement for broader applications in smart city projects.

# **Problem Statement**

**1. Intersection Layout and Lane Assignment:**

The intersection is designed with four incoming and outgoing lanes, divided into opposing pairs. Lanes 1 and 3 handle North-South traffic, while Lanes 2 and 4 handle East-West traffic. Each lane allows certain vehicle movements left-turn, straight-through, and right-turn. This setup ensures full coverage of all directions, and the traffic is managed accordingly in real time.

## **2. Signal Phases and Alternation:**

The traffic lights function in major two phases: Phase 1 allows the traffic flow in Lanes 1 and 3 (North-South), while Lanes 2 and 4 are closed. At the same time, in Phase 2, Lanes 2 and 4 (East-West) go active, while Lanes 1 and 3 closes up. Each phase can take anywhere from 15 to 30 seconds to let the vehicles pass through without any problem. This alternation avoids conflicts and maintains a smooth flow of traffic.

## **3. Real-Time Data Collection:**

The live traffic data is collected using cameras integrated with YOLO object detection technology. The system identifies vehicles, their movements. Further processing of this real-time data feeds it into a simulation environment, which allows dynamic adjustment of the traffic lights based on the observed conditions.

## **4. Reinforcement Learning Framework:**

The traffic light system is controlled based on a reinforcement learning approach and uses Deep Q-Learning. The model dynamically optimizes the signal timings by 'learning' from real data. It uses Boltzmann Exploration to balance new strategy exploration and successful strategy exploitation. This ensures the efficient adaptation of the system to dynamic traffic patterns.

## **5. Problem Formulation:**

The objective of this study is to minimize vehicle waiting times and optimize traffic flow at a four-way intersection using a machine learning-based approach. The problem is modelled as a **Markov Decision Process (MDP)** with the following components:

* **State:** The real-time traffic data, including vehicle counts and lane-specific densities, as observed through YOLO object detection.
* **Action:** The decision to activate either the 1-3 lane pair (North-South) or the 2-4 lane pair (East-West) for a dynamically determined duration (15-30 seconds).
* **Reward:** A function designed to minimize vehicle waiting times, reduce congestion, and improve traffic throughput.

The Deep Q-Network (DQN) agent operates in a partially observable environment where real-time traffic conditions are provided as inputs. By continuously adjusting signal timings based on observed data, the model aims to learn optimal strategies for managing traffic flow. The Boltzmann Exploration technique is applied to ensure a balance between exploring new actions and exploiting successful ones. The primary goal is to achieve a scalable, efficient, and adaptable traffic control system.

# **Methodology**

## **Deep Q-Network (DQN) Implementation**

Deep Q-Network is a reinforcement learning algorithm that combines Q-learning with deep neural networks to approximate state-action values in complex environments. Unlike the traditional Q-learning, DQN predicts Q-values using a neural network, which is very effective for large state spaces. Main techniques such as experience replay and target networks stabilize training and improve performance. The dynamically learned optimal policies maximize cumulative rewards, and applications to which DQN is particularly fit include traffic management, given that real-time decision-making is essential.

The Deep Q-Network (DQN) is implemented to optimize traffic light phases dynamically by learning from real-time traffic conditions. The implementation models the problem as a Markov Decision Process (MDP), with the following components:

* 1. **State Representation**

The state vector captures real-time traffic information for each lane, including:

* **Vehicle Count (n):** The number of vehicles in each lane.
* **Waiting Time (w):** The average waiting time for vehicles in each lane.
* **Other Features:** Lane-specific features such as the density and vehicle flow status.

The state is structured as a numerical vector with features for all lanes, enabling the agent to make informed decisions.

* 1. **Action Space**

The action space consists of two components:

1. **Lane Pair Selection:​**
   * The agent selects one of two lane pairs to open:
2. **Duration Selection:**
   * The duration of the green phase is chosen from a discrete range:

.

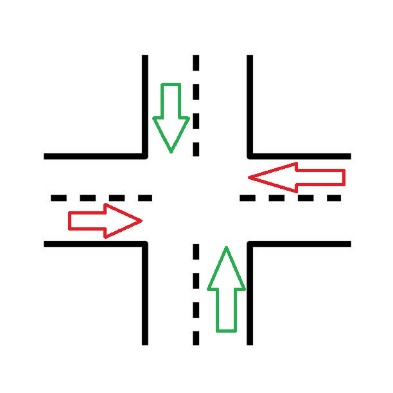
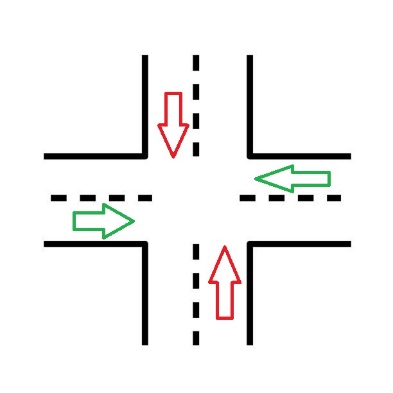
The agent predicts both lane pair and duration for the current phase based on the state.

Figure 2(Action 2)

Figure 1(Action 1)

* 1. **Neural Network Architecture**

The DQN model is implemented as a multi-head neural network with the following components:

* **Input Layer:** Accepts the state vector (s).
* **Hidden Layers:** Two fully connected layers with 128 neurons each, activated by ReLU functions.
* **Output Heads:**
  + **Lane Head:** Outputs logits for two lane pair options (a(lane)=1,2).
  + **Duration Head:** Outputs logits for 16 possible green phase durations (a(duration) ∈ {15,…,30}).

The network's forward pass is defined as:

*Lane Output=Softmax(fc\_lane(h2​)),*

*Duration Output=Softmax(fc\_duration(h2​)),*

*where h2=ReLU(fc2(ReLU(fc1(s))))*

* 1. **Reward Function**

The reward function incentivizes efficient traffic management by:

**Reducing Lane Density:**  
Reward for reducing the number of vehicles in opened lanes:

**Penalizing Congestion in Unopened Lanes:**

Penalty for high vehicle accumulation in unopened lanes:

where d is the density in unopened lanes.

**Encouraging Clearance of Opened Lanes:**

Additional reward if opened lanes are cleared completely:

The total reward is:

* 1. **Exploration Strategy**

The agent employs Boltzmann Exploration for action selection, where:

* Probabilities for lane selection P(lane) and duration selection P(duration)are calculated using the softmax function:
  1. **Training Process**

The DQN agent is trained using the following steps:

**Experience Replay:**  
Transitions (s, a, r, s′) are stored in a replay buffer and sampled in batches for training.

**Loss Function:**  
The loss is computed as the Mean Squared Error (MSE) between the target Q-value and the predicted Q-value:

where γ is the discount factor.

**Optimization:**

The model parameters are updated using the Adam optimizer, minimizing the combined loss for lane and duration predictions.

## **Simulation** **Environment**

Data from YOLO is formatted and sent into a queue for asynchronous transfer to the simulation environment. The environment processes this data to compute a state vector comprising vehicle counts, light statuses, and average waiting times for each lane. This state vector is passed into the DQN model, which predicts optimal traffic light actions. The actions dynamically manage lane phases and durations, ensuring efficient traffic flow while minimizing vehicle waiting times. Feedback from the simulation updates the DQN model through rewards and new state transitions.

**Formatted Data Transfer**  
The YOLO object detection system generates formatted outputs containing key traffic information, including the lane ID, object type, tracking ID, and timestamp for each detected vehicle. These dictionaries are sent into a queue in real-time to facilitate asynchronous data handling. The queue acts as a buffer, ensuring smooth data transfer between YOLO and the simulation environment.

**State Construction for DQN**

The simulation environment retrieves data from the queue and processes it into a structured **state vector** that encapsulates the real-time traffic conditions. The construction of the state vector is outlined as follows:

**Vehicle Count (​):**  
For each lane (**​i**), the number of detected vehicles is calculated:

**Traffic Light Status (​):​**  
The current light status for each lane is appended to the state vector:

* **1:** Green light.
* **0:** Red light.

**Average Waiting Time (​):​**  
The simulation calculates the average waiting time for vehicles in each lane:

* If the lane contains vehicles, their waiting times are averaged:
* If no vehicles are present, the waiting time is set to 0.

**Full State Vector:**  
The complete state vector for the DQN model is constructed as:

This captures vehicle counts, light statuses, and waiting times for all lanes.

## **Reward Function Design**

The reward function is part of the DQN model that will guide the learning by encouraging certain desired behaviours and penalizing undesired outcomes. Considering traffic signal control, the underlying reward function should be designed to optimize the flow of vehicles, minimize the waiting time of the vehicles, and reduce congestion. Herein, an elaborated construction and implementation of the reward function is presented.

## **Training Process**

The training process of the Deep Q-Network (DQN) is designed to optimize traffic signal control by learning policies through interactions with the simulation environment. At each timestep, the DQN agent observes the current state of the intersection, which includes the number of vehicles in each lane, the current signal status, and average waiting times. Based on this state, the agent selects an action using a probabilistic decision-making approach, such as the Boltzmann exploration strategy, to balance exploration and exploitation.

Each action involves choosing a pair of lanes to open and determining the duration for which the green signal remains active. The action is applied to the simulation environment, and the agent receives a reward based on the effectiveness of the decision. Rewards are designed to minimize waiting times and congestion while ensuring balanced traffic flow across all lanes.

The experiences—comprising the state, action, reward, and next state—are stored in a replay buffer. A mini-batch of these experiences is sampled periodically for training. The network is updated by minimizing the temporal difference (TD) error between the predicted Q-values and target Q-values, calculated using the Bellman equation. To stabilize training, target network updates occur at regular intervals.

Training is conducted over multiple episodes, with each episode simulating a fixed duration of traffic flow under varying conditions. The agent iteratively improves its policies by refining its ability to manage traffic efficiently, balancing the needs of all lanes, and reducing overall vehicle waiting times. Once the model exhibits robust performance in the simulated environment, it is evaluated to verify its effectiveness across diverse traffic scenarios.

# Results

The evaluation of the dynamic traffic control system managed by the DQN agent under varying traffic conditions reveals its adaptability and efficiency in maintaining a smooth traffic flow. Initially, during periods of moderate traffic, the system demonstrated clear advantages over the fixed-timing system by significantly reducing the average waiting times across all lanes. However, during high traffic density periods, the average waiting times under the DQN-based dynamic system exhibited a slight increase. Despite this, the system prevented congestion and ensured continuous traffic flow, highlighting its superiority in managing complex traffic patterns.

The fixed-timing system, with its static schedule, often fails to address real-time traffic conditions, leading to disproportionate utilization of lanes. This results in severe bottlenecks in high-traffic lanes while underutilizing low-traffic lanes, creating inefficiencies and, at times, complete gridlock during peak hours.

By contrast, the DQN agent dynamically prioritizes lanes with higher traffic densities. As the traffic load increases, the agent adapts by allocating longer green light durations to the congested lanes, sometimes providing consecutive green light cycles to mitigate the buildup. This real-time responsiveness ensures that heavy traffic does not escalate into gridlock and that lighter traffic lanes are not unnecessarily prioritized. The result is an even distribution of flow, where no lane becomes critically overloaded, maintaining an overall fluid and efficient traffic system.

Moreover, this dynamic prioritization means the system actively addresses evolving traffic patterns, preventing accumulation at critical junctions and ensuring that vehicles in high-density lanes are processed more effectively. This adaptive behaviour is particularly critical during peak hours when the traditional fixed-timing systems fail to keep up with the fluctuating traffic demands.

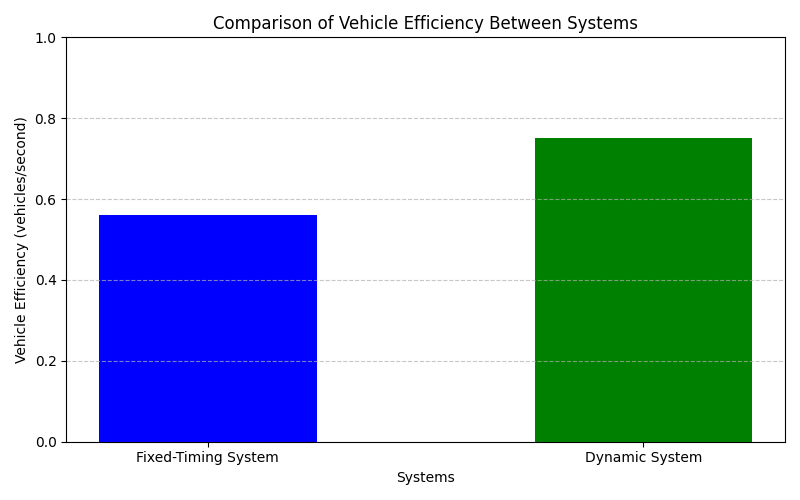
In summary, while the dynamic system's average waiting times increase marginally during peak hours, the ability to prioritize high-traffic lanes and maintain an uninterrupted flow sets it apart. This demonstrates the potential of reinforcement learning-based systems like the DQN agent to revolutionize traffic management, optimizing performance under both normal and high-demand conditions, and significantly reducing the risk of gridlock in urban traffic systems. These findings underscore the need for adaptive, AI-driven solutions to replace static, outdated traffic management systems. 

Figure 3(Low Frequency)

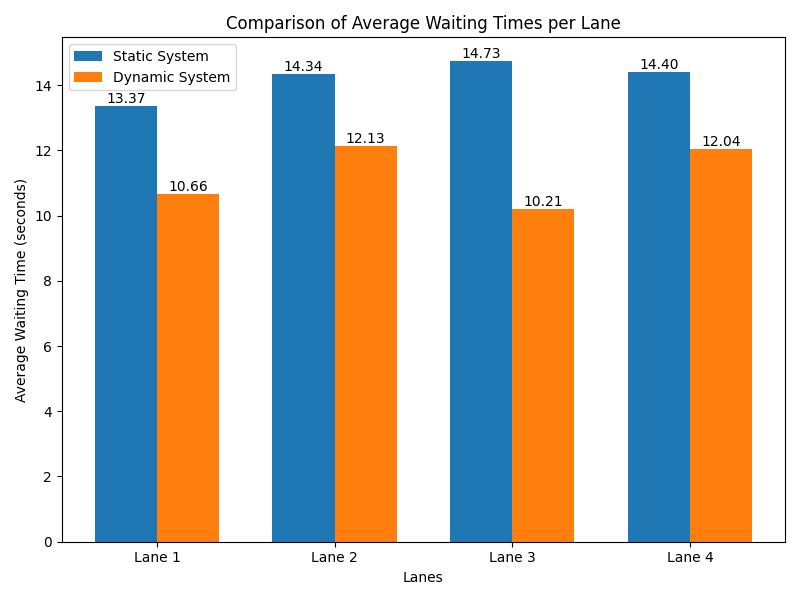


Figure 4(Low Frequency)

The results of this study demonstrate the significant potential of reinforcement learning, specifically Deep Q-Networks (DQN), in optimizing urban traffic management systems. The dynamic traffic control system developed in this research successfully adapts to real-time traffic conditions, prioritizing heavily congested lanes and ensuring smoother overall traffic flow. By dynamically adjusting green light durations and giving precedence to high-traffic areas, the DQN agent mitigates the risk of bottlenecks and gridlock, even during peak traffic periods.

The comparison between the DQN-based system and traditional fixed-timing systems highlights the advantages of adaptive traffic management. While the fixed-timing system operates on predetermined cycles, it fails to accommodate varying traffic densities, often leading to underutilized lanes and heavily congested ones. The DQN system, however, dynamically responds to traffic changes, reducing average waiting times during moderate traffic and preventing critical congestion during peak hours.

Despite a slight increase in average waiting times for some lanes during high-density periods, the DQN system outperforms fixed-timing systems in maintaining continuous flow and avoiding traffic collapse. This behaviour is critical for modern urban settings where fluctuating traffic demands require flexible and efficient management solutions.

This research underscores the transformative potential of AI-driven solutions in traffic management, paving the way for smarter and more responsive urban infrastructure. Future work could explore the integration of additional data sources, such as weather conditions or real-time accident reports, to further enhance the adaptability and robustness of the system. Additionally, testing the system on a larger scale or in real-world scenarios could validate its scalability and effectiveness in diverse traffic environments.

In conclusion, the implementation of DQN agents for dynamic traffic management represents a significant step forward in addressing the challenges of modern traffic systems. By reducing congestion and optimizing traffic flow, such systems not only enhance commuter experience but also contribute to environmental sustainability by minimizing idle times and fuel consumption. This study sets the foundation for future innovations in adaptive traffic control, promising a more efficient and intelligent approach to urban traffic management.

# References

* 1. Arel, I., Liu, C., Urbanik, T., & Kohls, A. G. (2010). Reinforcement learning-based multi-agent system for network traffic signal control. *Artificial Intelligence Review*, 29(2), 107-118.
  2. Li, L., Lv, Y., & Wang, F.-Y. (2016). Traffic signal timing via deep reinforcement learning. *IEEE/CAA Journal of Automatica Sinica*, 3(3), 247-254.
  3. Ma, X., Tao, Z., Wang, Y., Yu, H., & Wang, Y. (2017). Long short-term memory neural network for traffic speed prediction using remote microwave sensor data. *Transportation Research Part C: Emerging Technologies*, 54, 187-197.
  4. Mnih, V., Kavukcuoglu, K., Silver, D., et al. (2015). Human-level control through deep reinforcement learning. *Nature*, 518(7540), 529-533.
  5. Sutton, R. S., & Barto, A. G. (2018). *Reinforcement Learning: An Introduction*. MIT Press.
  6. van der Pol, E., & Oliehoek, F. A. (2016). Coordinated deep reinforcement learners for traffic light control. *Proceedings of the 30th Conference on Neural Information Processing Systems (NeurIPS)*.
  7. Wei, H., Zheng, G., Yao, H., & Li, Z. (2018). IntelliLight: A reinforcement learning approach for intelligent traffic light control. *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2496-2505.
  8. Behrisch, M., Bieker, L., Erdmann, J., & Rudzewicz, D. (2011). SUMO–simulation of urban mobility: an overview. *Proceedings of the Third International Conference on Advances in System Simulation*, 63-68.
  9. OpenAI's ChatGPT language model was utilized to generate insights and assist in drafting sections of this work.
  10. <https://medium.com/@Mert.A/how-to-use-yolov8-and-yolo-nas-for-object-detection-8c5893939480>