Traffic Control Systems

Using Reinforcement Learning

GROUP 12
Rohan Mistry
Harry Leung
Derek Xu

AGENDA

01 PROJECT OVERVIEW

02 RESEARCH AND DEVELOPMENT

03 RESULTS AND ANALYSIS

04 INNOVATION AND IMPACT

05 REFLECTION AND LEARNING

06 CONCLUSION

PROJECT OVERVIEW

- Traffic congestion is a major issue affecting urban areas worldwide, leading to increased travel times, air pollution, and commuter frustration
- Occurs often when road demand exceeds supply, particularly during peak hours

SIGNIFICANCE

- Far-reaching effects on economic efficiency, public health, and environmental sustainability
- Congested traffic systems contribute to wasted fuel and time, resulting in billions of dollars in economic losses

REAL-WORLD IMPACT

According to Petroski, traffic congestion and associated delays costs U.S. economy over $120 billion annually

U.S. traffic congestion resulted in average driver spending 51 hours annually in traffic, equating to nearly $1,000 in lost time and increased pollution

INCREASED VEHICLE EMISSIONS

AIR QUALITY DEGRADATION

CLIMATE CHANGE

PROJECT OVERVIEW (CONT.)

Objectives

- Use multi-agent proximal policy optimization (MAPPO) to train a reinforcement learning (RL)-based model to control traffic lights for optimizing traffic flow
- Each traffic signal is independent agent, optimizing flow at each intersection
- Find out how using the RL model has improved various metrics such as wait times, queue lengths at red lights, and overall throughput

Expected Outcomes

- The reward of our model after training will be positive, which reflects that we have successfully trained a model that manages traffic flow well
- Throughout our training, major metrics we use to evaluate the quality of traffic flow, such as wait times and queue lengths, improve steadily
- Metrics evaluating the quality of our model stabilize at the end of training

RESEARCH AND DEVELOPMENT

Technologies Used

- Ray
  - Machine learning (ML) framework that provides capabilities for various types of training algorithms, including proximal policy optimization (PPO)
  - Used to train our model with MAPPO
- CityFlow
  - RL environment that simulates urban traffic
  - Used to create a custom environment that is compatible with Ray's implementation of PPO
- Gymnasium
  - Used Gymnasium in conjunction with CityFlow for defining observation and action spaces in environment

Process

- Generated custom traffic grid networks and vehicle flows with varying sizes
- Implemented a multi-agent RL environment class with CityFlow that simulates real-world traffic
- Utilized Ray's implementation of MAPPO to train the model with centralized training and decentralized execution
- Observation space: local states per intersection, global state combines intersection states
- Reward function minimizes waiting, penalizes long queues, maximizes throughput
- Used automated hyperparameter tuning
- Still working on scaling problem to larger grid scenarios and creating more robust reward function

RESEARCH AND DEVELOPMENT (CONT.)

Adaptations/changes

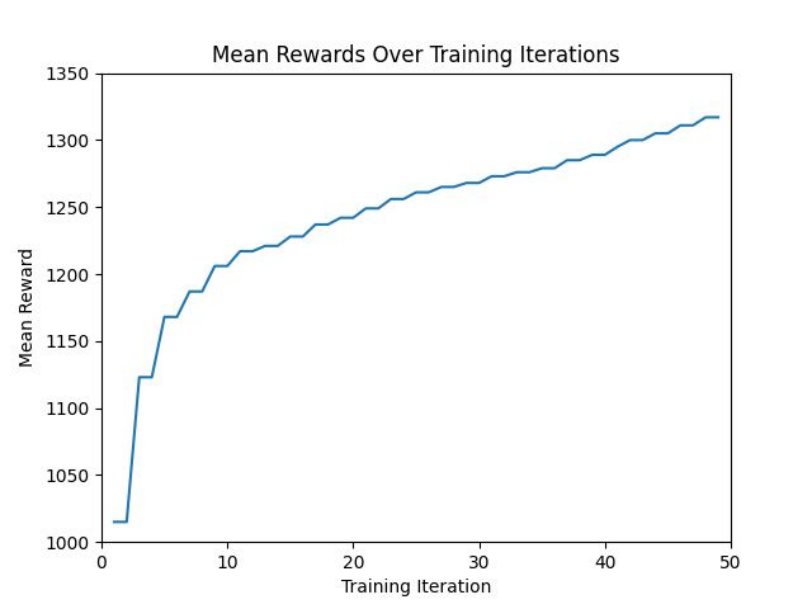
- Not using a dataset in training
  - The dataset we planned to utilize did not have many features that represent the quality of traffic flow, so we rather simulated scenarios with CityFlow, which provides more data necessary for training
- Not using PPO implementation from Hugging Face
  - We believed that it is better to have our own source code, which we can more easily understand
- Moving from stable_baselines3 to Ray's RLlib
  - stable_baselines3 does not handle MAPPO well, while it is better supported on RLlib

Challenges

- Creating the environment class to call PPO implementation on
  - Understanding how to use CityFlow with it
  - Debugging CityFlow API
- Implementing and optimizing PPO algorithm with tuning to monitor training behavior on many iterations
  - Takes lots of time and compute power
- Keeping track of features for each intersection is very complex

RESULTS AND ANALYSIS

- Current results:
  - Mean reward for fixed action traffic signals: 992.33
  - Mean reward for current MAPPO: 1317.10
- PPO trained on 1000 max steps over 50 iterations
- Current hyperparameters: gamma=0.99, train_batch_size=2048, minibatch_size=64, num_epochs=10, grad_clip=0.5, clip_param=0.2
- Takes into account policy for each individual intersection which includes throughput, queue count, and waiting count
- Still working on developing more elaborate reward function with more advanced tracking features and being able to scale to larger grids but preliminary results are positive, showing that even early stages of PPO implementation are far superior to fixed signals



INNOVATION AND IMPACT

Innovation

- Our implementation uses a multi-agent approach to traffic control, while other implementations tend to be single-agent
  - Each agent controls traffic lights for a single intersection
  - Increases flexibility when training
  - Even approaches that use many lanes and intersections typically use single-agent approaches

Impact

- Being able to use RL to find optimal traffic light signal timings can inform what timings should be best used in the real world
- An environment and model similar to what we have used can be used on real-world intersections to determine the best timings, thereby improving traffic flow in the real world
- Helps reduce travel time, congestion, and environmental impact resulting from cars being on the road longer
- Economic benefits from shorter transportation times on roads

REFLECTION AND LEARNING

Reflection

- Having to create our own environment was very time-consuming
- Debugging integration between Ray, CityFlow, and Gym was complicated
- Project helped us understand the complexities of scaling RL solutions and the difficulties of having to implement multi-agent solutions
- Realized importance of tool selection because not every framework fits every use case and we had to adapt multiple tools before settling on right ones

Learning

- We learned how to use existing RL environments and new frameworks to facilitate model, and CityFlow API helped immensely
- Learned how to fine-tune and iterate over RL models with highly-complex traffic system
- Balancing centralized training and decentralized execution was tricky at first, where each agent has its own policy and executes based on local observations but training is centralized
- Running tuning algorithms that we implemented requires lots of compute power and many hours so we are working on simplifying that so we can use visualization tools at our disposal to chart best hyperparameter configurations

CONCLUSION

- Proposed traffic control system uses multi-agent PPO for adaptive traffic management
- Enables each intersection to act as coordinated agents and dynamically adjust signal timing based on real-time traffic flow
- MAPPO algorithm: addresses limitations of static and single-agent systems; more scalable for complex urban networks
- Expected benefits: reduced travel time and improved response to traffic variability
- Difficulties with tuning reward function due to compute power, debugging environment issues, and scaling to larger grids
- Potential to revolutionize traffic management in high population areas, enhancing overall economic efficiency and environmental sustainability
- Will continue to refine model and try to improve upon existing solution for more optimized rewards