신호 최적화 최종발표

ETRI연구연수생 강민수

Contents

- Current Model
  1. 3x3Grid Single-Agent Model without Constraints(Depreciated)
  2. 3x3Grid Decentralized Model without Constraints(Depreciated)
  3. 3x3Grid/Dunsan Decentralized Model with Practical Constraints
  4. Dunsan Decentralized CNN/CNN-reduced Model with Practical Constraints

- Upcoming Model
  1. Continuous Action Model with Practical Constraints

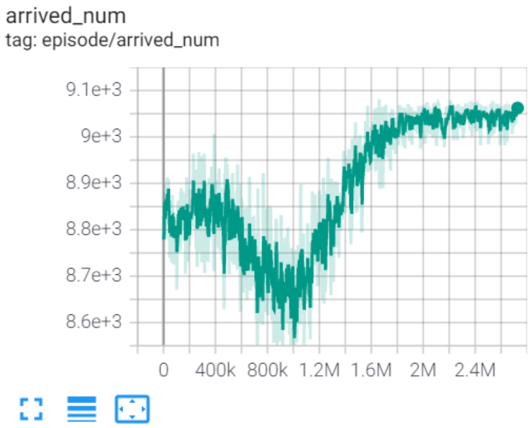
3x3Grid Single-Agent/Decentralized Model without Constraints(Depreciated)

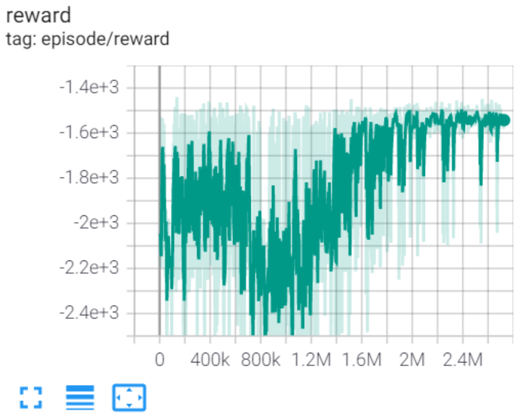
- Condition (seed fixed)
  1. Generating probability 0.133 (left, right), 0.388 (up, down)
  2. Restricted direction of vehicles(via)/Not restricted direction(no via)
  3. Central agent(n_1_1), 9 Agents(decentralized)
- State
  1. The number of inflow vehicles from each inflow edge(left, straight)
- Action
  1. Deciding next phase(without an order), every 20s (with all yellow 3 seconds)
- Reward
  1. Penalty on Pressure (= inflow-outflow)

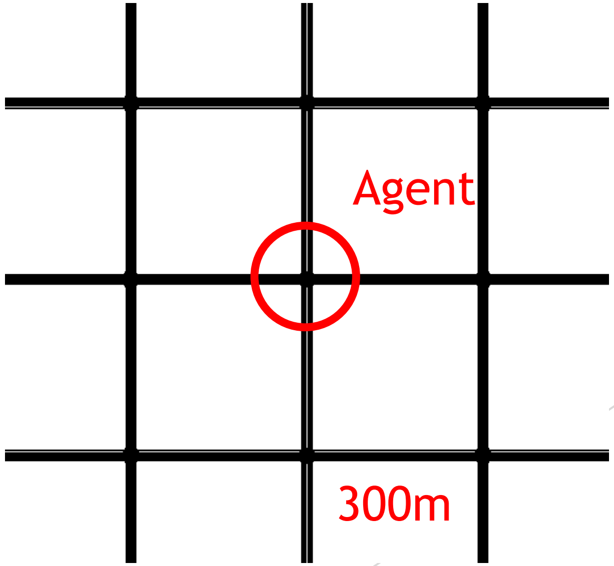
Comparison between Simulation & Single Agent

- Simulation 0.133 (left, right), 0.388 (up, down), phase 42(G), 3(Y)
  1. The number of arrived number: 8295 (no via), 8794 (via)
- Experiment(Single-Agent Model)
  1. The number of average arrived number: 9060 (no via), 9281 (via)

5~8% Increase





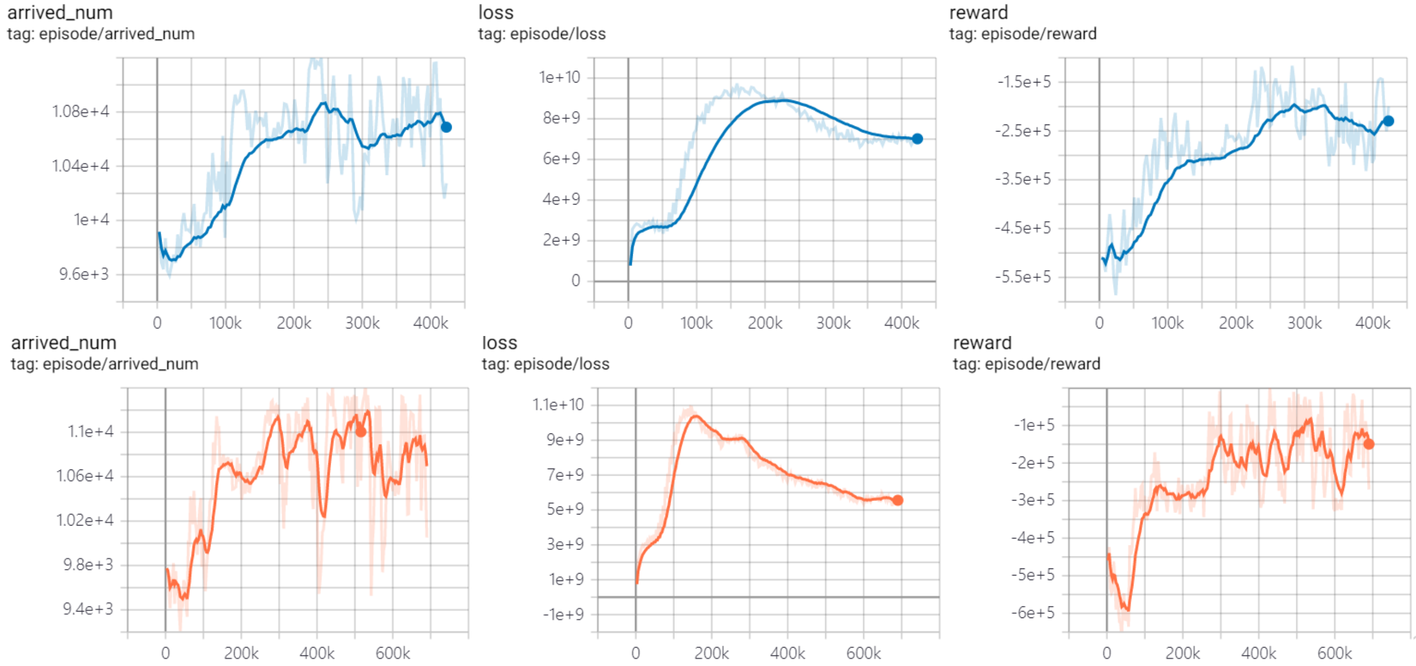


Comparison between Simulation & Decentralized Agents

- Simulation 0.133 (left, right), 0.388 (up, down), phase 20(G), 3(y)
    1. The number of arrived number: 9474 (no via), 9765(via)
- Experiment(Decentralized Agents Model)

10~11% Increase

    1. The number of avg arrived number: 10600(no via, blue), 10800 (via, orange)



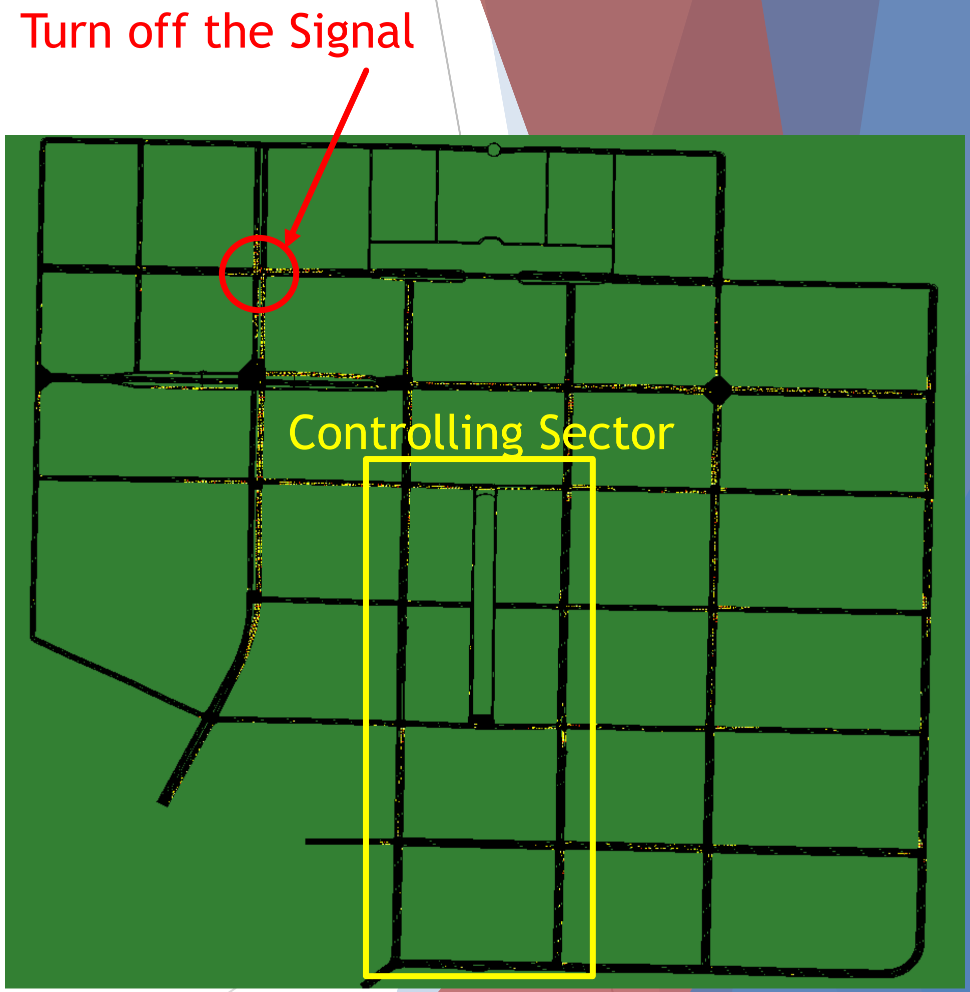
3x3Grid/Dunsan Decentralized Model with Practical Constraints

- Condition(seed fixed)
  1. Random routing vehicle generation (period 0.8), scale 1.1 (by randomTrips.py)
  2. 9 Decentralized Agents(No offsets, Update Asynchronously)
  3. 4 Phases(Vertical/Horizontal straight and left) 37s each, followed by all yellow 3s
  4. Model applied min/max duration(28s,49s) and phase period(160s)
- State
  1. The number of inflow vehicles from each inflow edge(left, straight)
  2. Update phase demand(# of vehicles inflow) from end of phase period

3x3Grid/Dunsan Decentralized Model with Practical Constraints

- Condition(seed fixed)
  1. **Real Demand(0-3[scaled 2times],7-10am[scaled 0.7 times])**
  2. 9 Decentralized Agents(Update asynchronously by offsets)
  3. **Using Each designated phase rule in xml file**
  4. **Model learned from only first 3600 steps**

- State
  1. The number of inflow vehicles from each edge(left, straight)
  2. Update demand(# of vehicles inflow) from end of phase period



3x3Grid/Dunsan Decentralized Model with Practical Constraints

- Agent
  1. Decentralized DQN, Soft Target Update(Tau: 0.001)
  2. Common phase-based ratio discrete distribution
  3. 2 Actions/agent = time action, rate action
  4. Action Space
     1) Time action space: $\max(\min_i(phaseMax_i - common_i, common_i - phaseMin_i))$
     2) Rate action space: 17, distributing time to each phase ex) [0,1,0,-1],[0,0,0,0],[1,1,-1,-1]

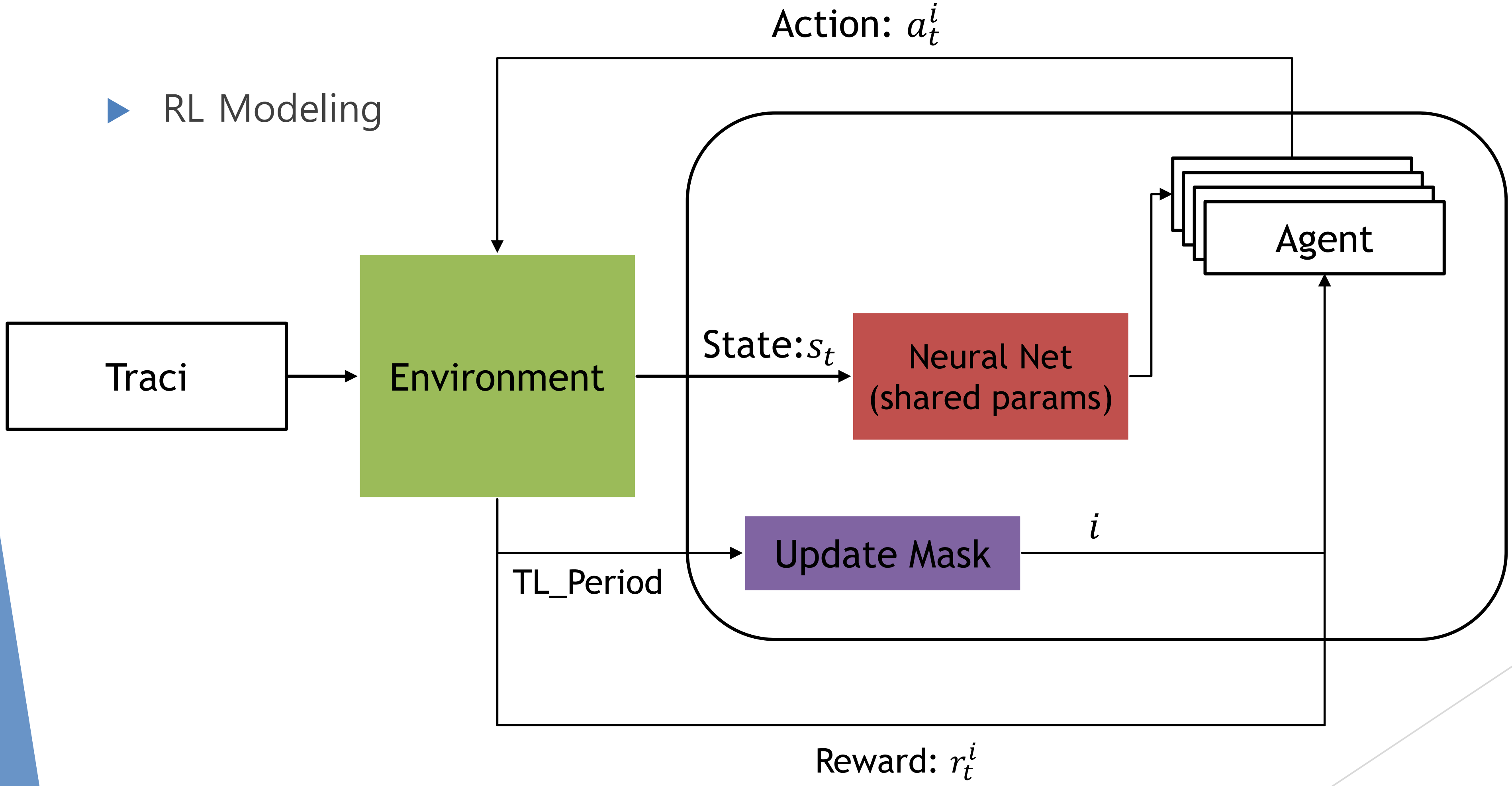
Dunsan Rate Action

# of phase	Action space
2	2
3	3C2*2+1
4	4C2*2+4C4*4+1
5	5C2*2+5C4*4+1

- Reward
  1. Reward= -Pressure = (inflow HaltingNumber – outflow VehicleNumber)
  2. If action is beyond the threshold(min/max duration), give penalties
  3. Update end of each phase

Decentralized Model with Practical Constraints

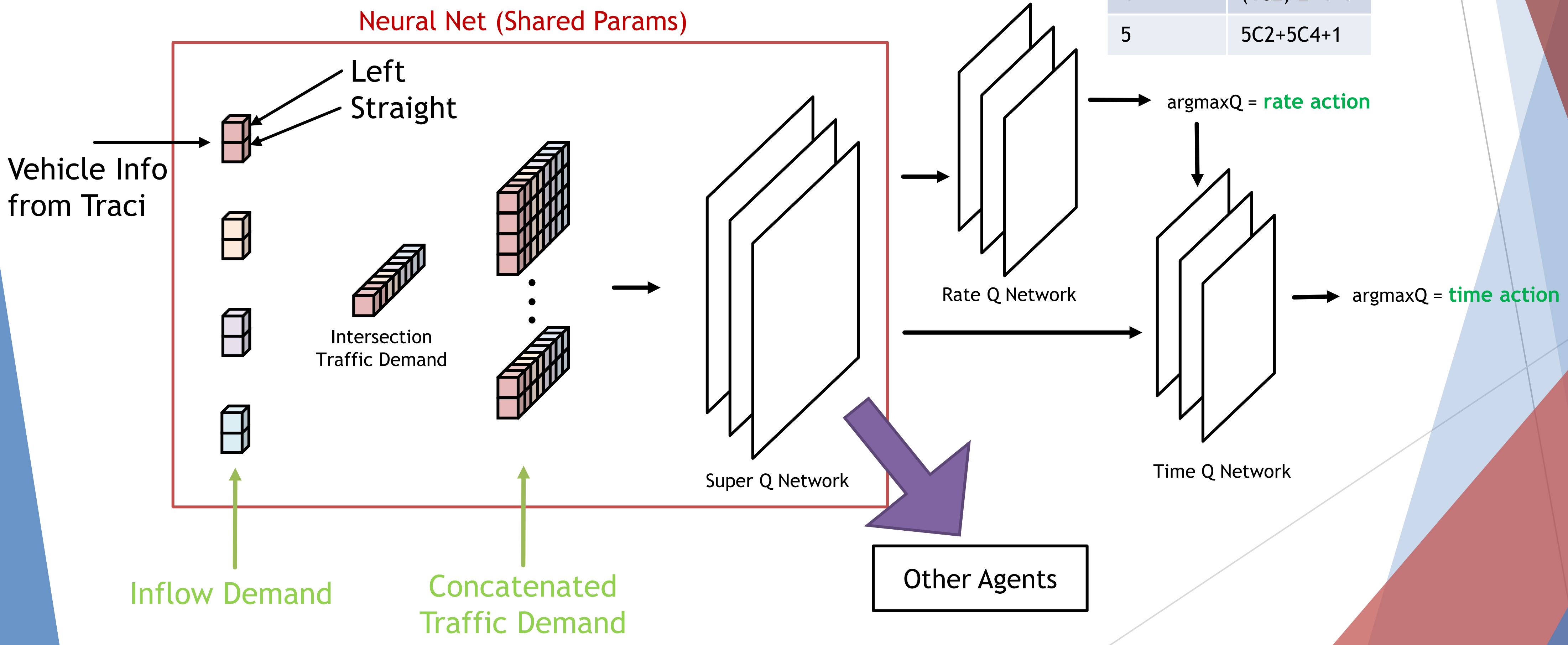
- RL Modeling



Decentralized Model with Practical Constraints

# of phase	Action space
2	2
3	7
4	(4C2)*2+4+1
5	5C2+5C4+1

- DNN Modeling



Decentralized Model with Practical Constraints

- 3x3 grid
  1. 9 Agents
  2. Random Routing Generation
     → 1 period, 1.5 scaled

Avg Value	Simulation	Decentralized
Travel Time (per edge)	61.0s	Not converged
Velocity	3.43m/s	-
Waiting Time (whole trip)	40.2s/veh	-

- Dunsan
  1. 11 Agents
  2. Real Demand based training
     → 0-3am, 2.0 scaled

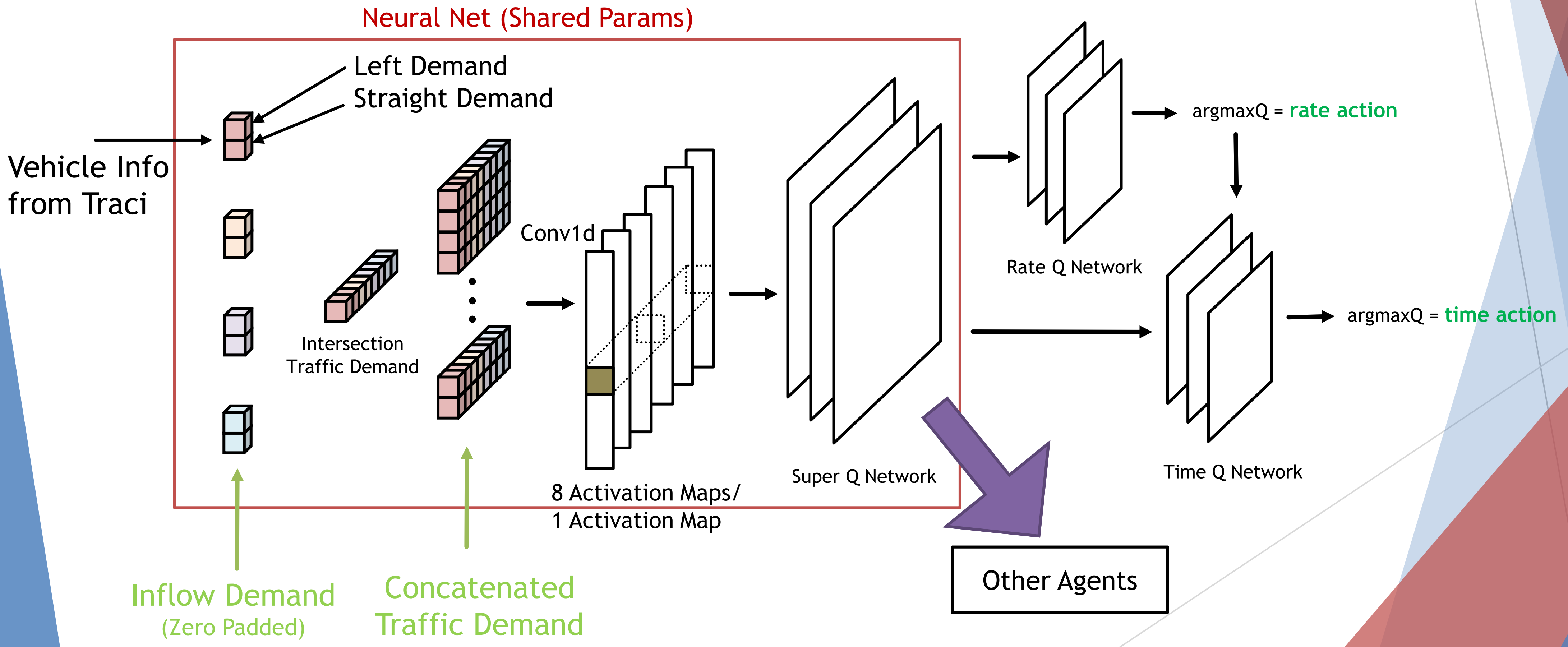
Avg Value	Simulation	Decentralized
Travel Time (per edge)	13.82s	14.52
Velocity	5.93m/s	5.769m/s
Waiting Time (whole trip)	495.5s/veh	534.1s/veh
Arrived Num	10951	10863

Dunsan Decentralized CNN/CNN-reduced Model with Practical Constraints

- CNN based Decentralized Model
  1. CNN: Convolution Neural Network
     1) Feature extraction in selected region → Apply to intersection information
     2) Advantage: less parameters than Fully Connected Network → Lightweighting
  2. CNN based Model (Feature Extraction) 1000epochs
     1) Current Channel: 8 (zero-padded demand of vehicles from inflow edge)
     2) After CNN, the number of channel: 16
     3) Learning Time: 8h 11m(0-3am), 30h(7-10am)
  3. CNN based Lightweight Model
     1) By using Conv1d, dimensionality reduced to 1 channel with feature
     2) Learning Time: 6h 10m, 26h(7-10am)

Dunsan Decentralized CNN/CNN-reduced Model with Practical Constraints

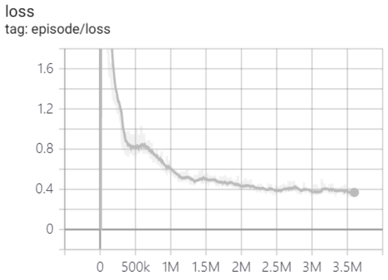
- CNN based model(Single Agent based Model)

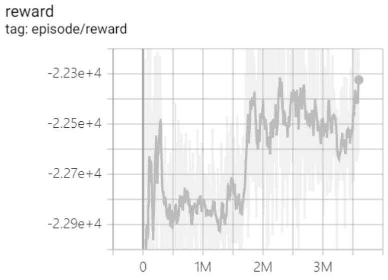


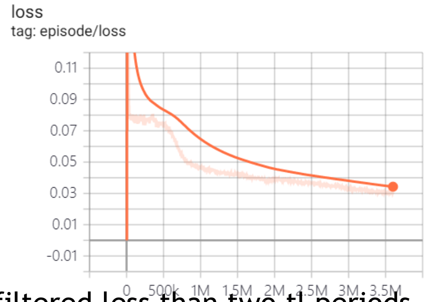
Analysis of Decentralized CNN-reduced Model

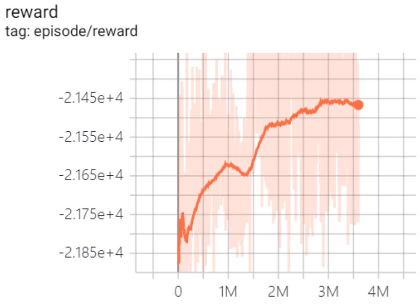
- CNN Model(0to3am)
  1. 8 feature → Checking demand change
- CNN-reduced Model(0to3am)
  1. 1 feature → Current traffic info

Avg Value(interest edges)	Simulation	Decentralized-CNN	Decentralized-CNN-reduced
Travel Time (per edge)	33.8s	47.5s	42.4s
Avg Velocity(edge veh)	5.99m/s	5.52m/s	5.82m/s
Waiting Time (during whole trip)	165.1s/edg	210s/edg	175.5s/edg
Arrived Num (all edge)	11167	11022 (02-19_09-29-17)	11114 (02-23_09-36-26)









Travel Time: mean value of getTraveltime for all steps that are filtered less than two tl periods
Avg Velocity: average velocity from edge that defined to control by model

Analysis of Decentralized CNN-reduced Model

- Avoiding Overfitting(Validation)
  1. Add Dropout Layer between fully connected layer in super Q Network
  2. Apply peak demand(7-10am) to the model learned from free demand(0-3am)

Avg Value	Simulation	Before adding dropout	After adding dropout
Travel Time (per edge)	43.546s	54.716s	44.5s
Velocity(edge veh)	5.413m/s	5.187m/s	5.46m/s
Waiting Time (count all veh in edg)	208.95s/edg	287.42s/edg	219.54s/edg
Arrived Num	12111	11873	12175(02-24_15-20-23)

- Conclusion: For wide coverage, strengthen performance by dropout layer

Analysis of Decentralized CNN-reduced Model

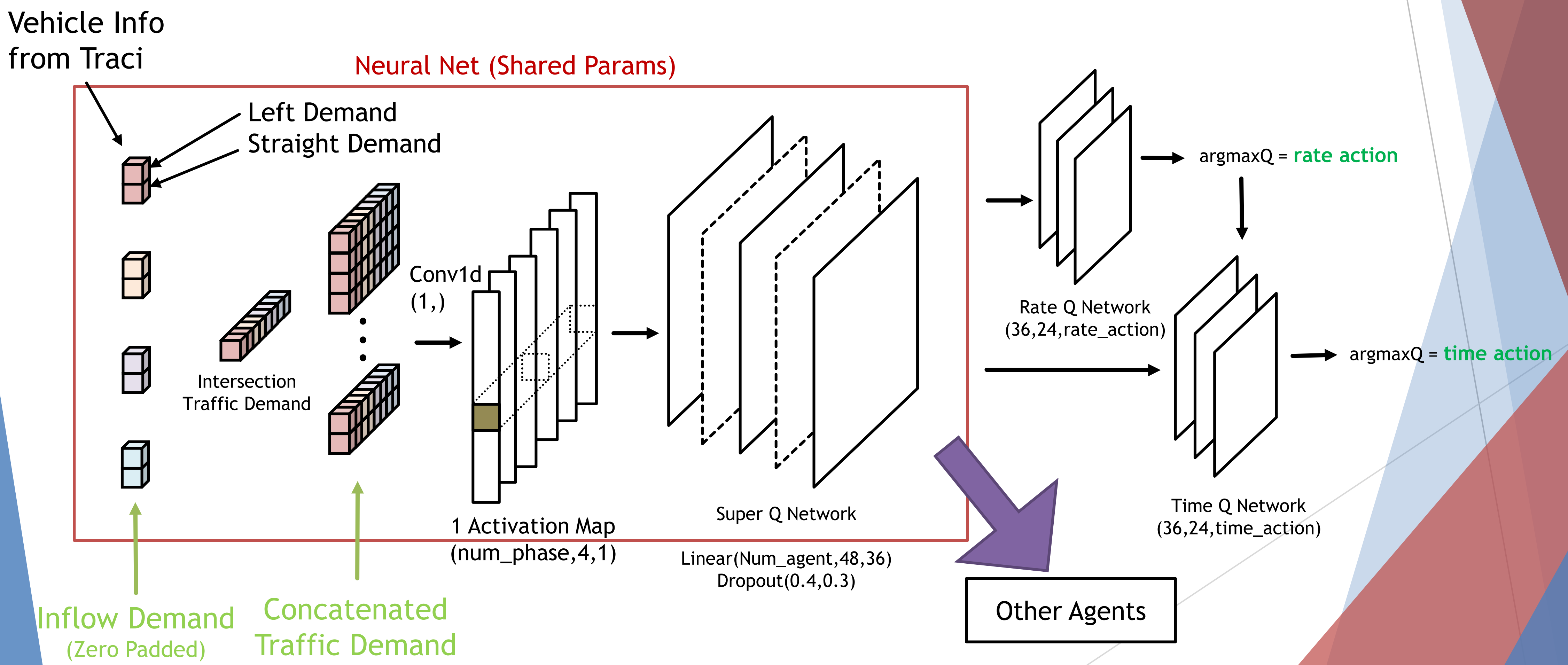
- Practical Constraints without minimum of duration condition
  1. Removing penalty in action that is less/more than min/maximum length of duration
  2. Purpose: Verifying whether regulations lead traffic jam or not
  3. Learning process and Reward comparison

	Learning with Constraint	Learning without Constraint
Learning Time	6h 14m	5h 56m
Avg converge reward	9.2e-3(02-25_08-50-45,not converge yet)	-8.8e-3(02-24_15-20-23)
Avg velocity(sim in 7-10)	5.40m/s	5.46m/s

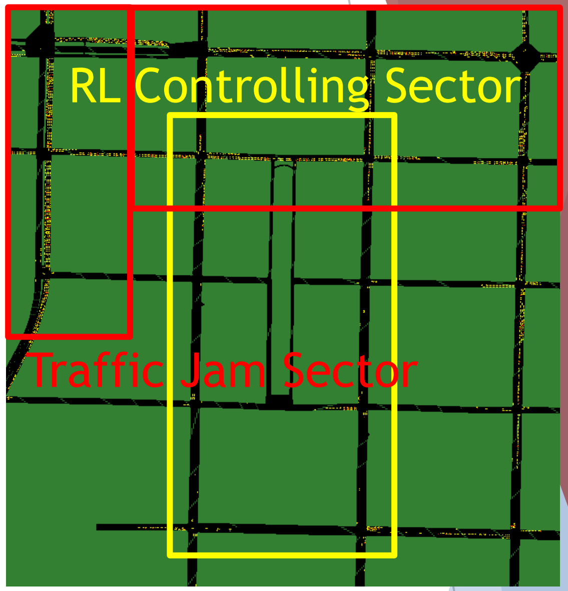
- Conclusion: 1) Current signal is incomplete (scope to improve more)
  2) Hard to learn model with constraint
  3) Arbitrary Penalty -> Not converge

Final Model

- Decentralized CNN-reduced Deep Reinforcement Learning model
    1. Hyperparameters: tau 0.001, gamma 0.99, lr 1e-4, eps 0.8, both decay_rate 0.99, batch 64



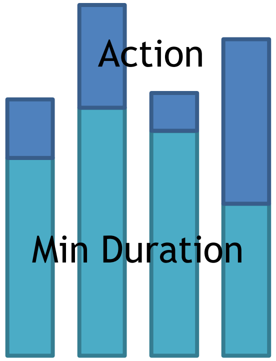
Reason for failing experiment

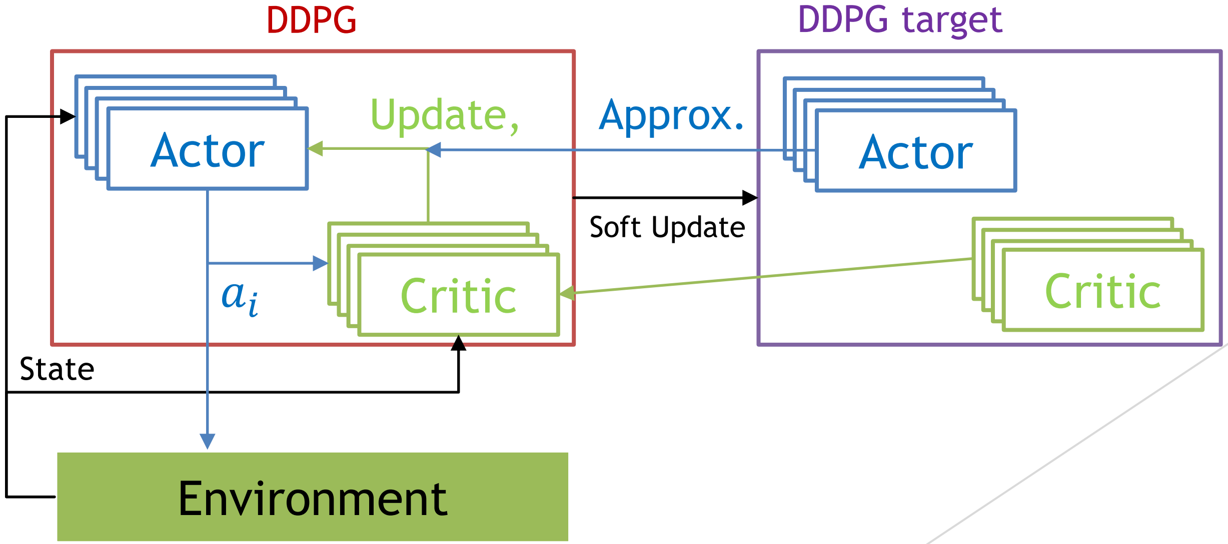


- Generated Actual data biased for fixed time signal control
- Congested road conditions not being reflected by scaled data
    1. Using 7-10am demand instead of 0-3am
    2. Trade off between learning time(x3 in 7-10) and performance
- Hard to improve the performance due to fixed offset value
- Due to lack of other traffic light control, lead RL traffic lights lower performance
- No convergence due to arbitrary penalty in case of exceeding threshold(min/max duration)
- In SUMO, road deletions due to absence of pocket(left turn) lead unintended traffic jam

Continuous Action Model with Practical Constraints

- DDPG based Model
  1. Current action space: Time, Rate → Phase length (for each phase)
  2. Splitting rest of time (Total TL period – sum of min durations for each phase)
  3. Phase distribution by the results of softmax(a_i)
     1) Sum of results by softmax function = 1 → Easy to distribute length of phase





Thank you