Deep Reinforcement Learning for Solving AGVs Routing Problem

Authors: Chengxuan Lu, Jinjun Long, Zichao Xing, Weimin Wu, Yong Gu, Jiliang Luo, and Yi-Sheng Huang

Reporter: Chengxuan Lu

CONTENTS

1 Introduction

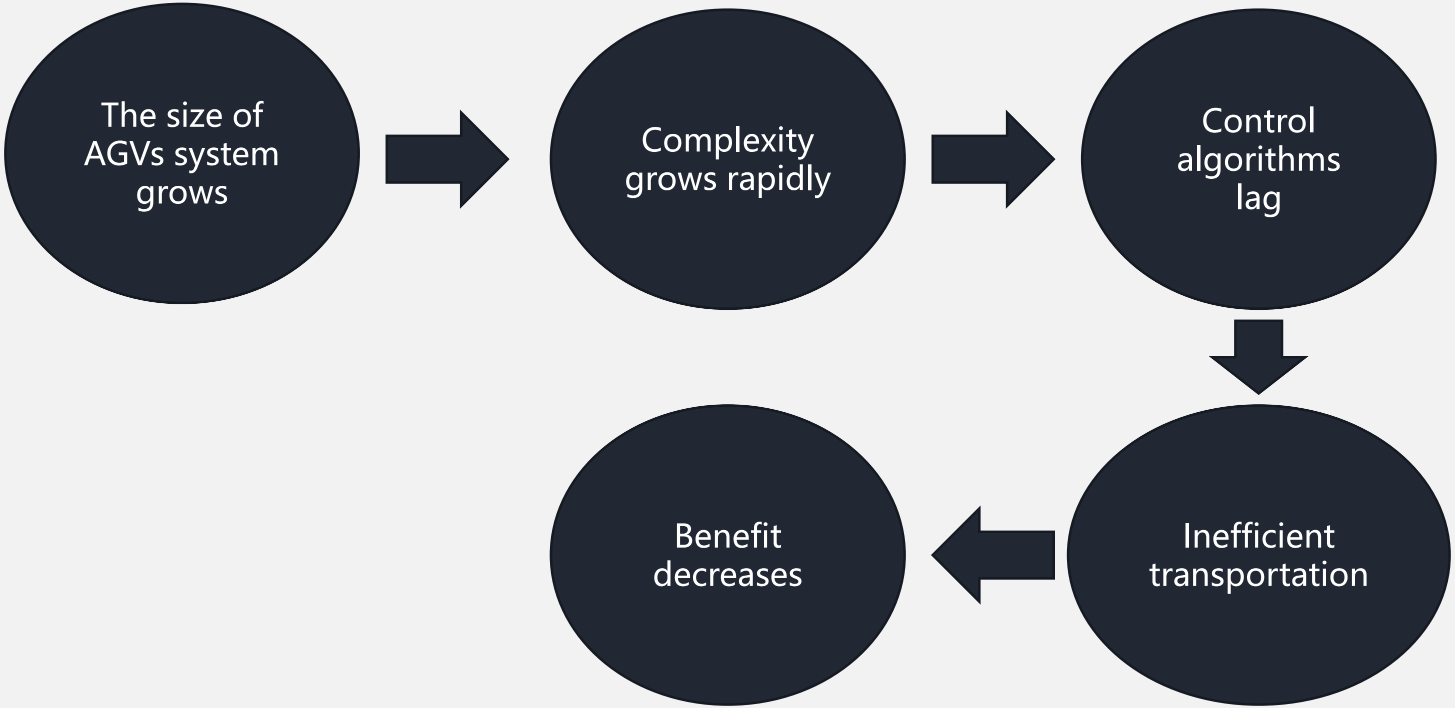
Background

AGV（automated guided vehicle）

- Flexible
- Efficient
- High capacity
- Unmanned

AGVs working in a warehouse

Background


The size of AGVs system grows
Complexity grows rapidly
Control algorithms lag
Inefficient transportation
Benefit decreases

Background

	Advantage	Disadvantage
Exact approaches	Optimal solution	Extreme high time complexity
Heuristics	Good solution	High time complexity
Meta-heuristics	Good solution	Cannot response in real-time
Regulations	Response in real-time	Relative suboptimal solution

Development of Deep Reinforcement Learning (DRL)

Atari
Raw pixels

AlphaGo
Defeat human champion

AlphaGo Zero
Without human knowledge

StarCraft, DOTA
Huge state space
Defeat human expert

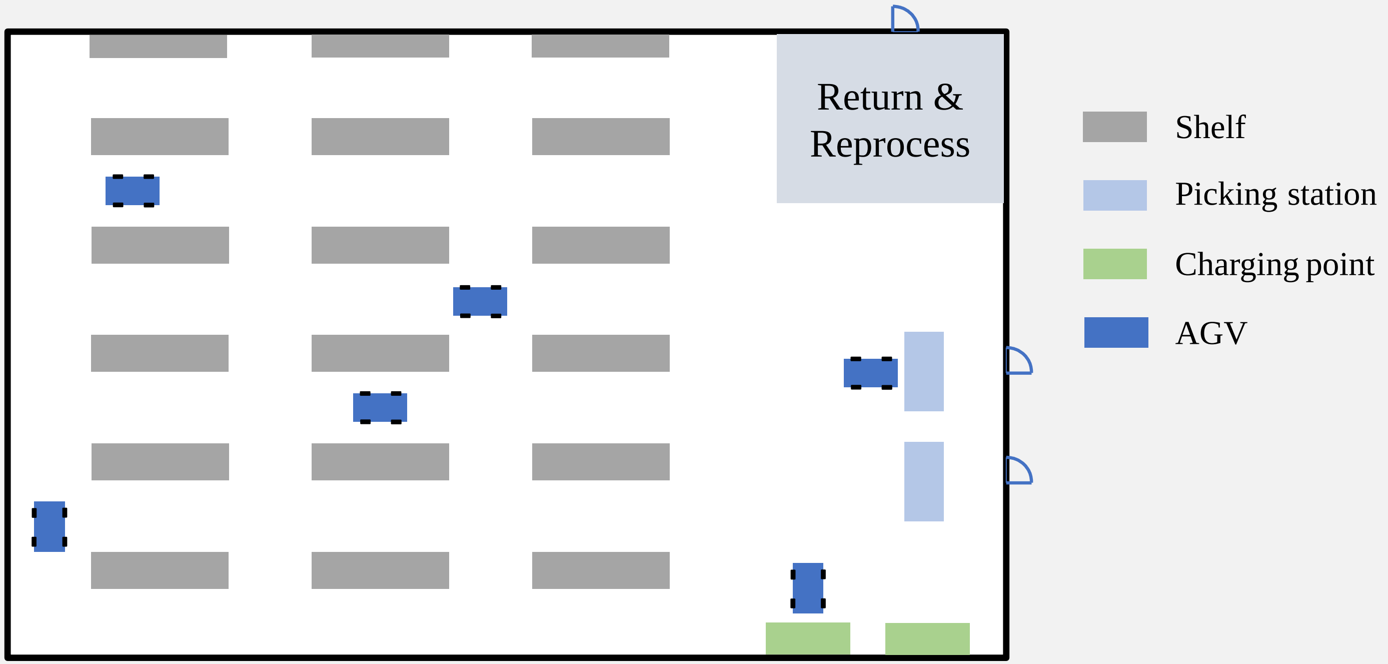
Similar Researches

- **Tabular reinforcement learning. Curse-of-dimensionality.**
- **Supervised data will influence the final performance.**
- **AGVs number is small.**

[Xue, T. et al., 2018], [Kamoshida, R. et al., 2017], [Zhao, M. et al., 2019]

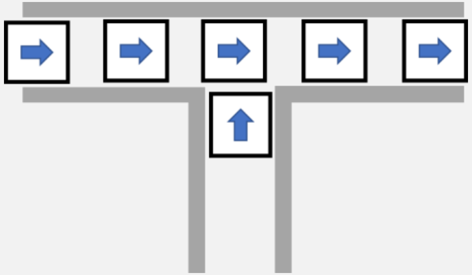
2 AGVs Routing Problem

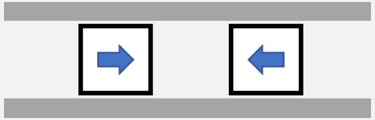
AGVs system



A simplified AGVs working environment

AGVs Routing

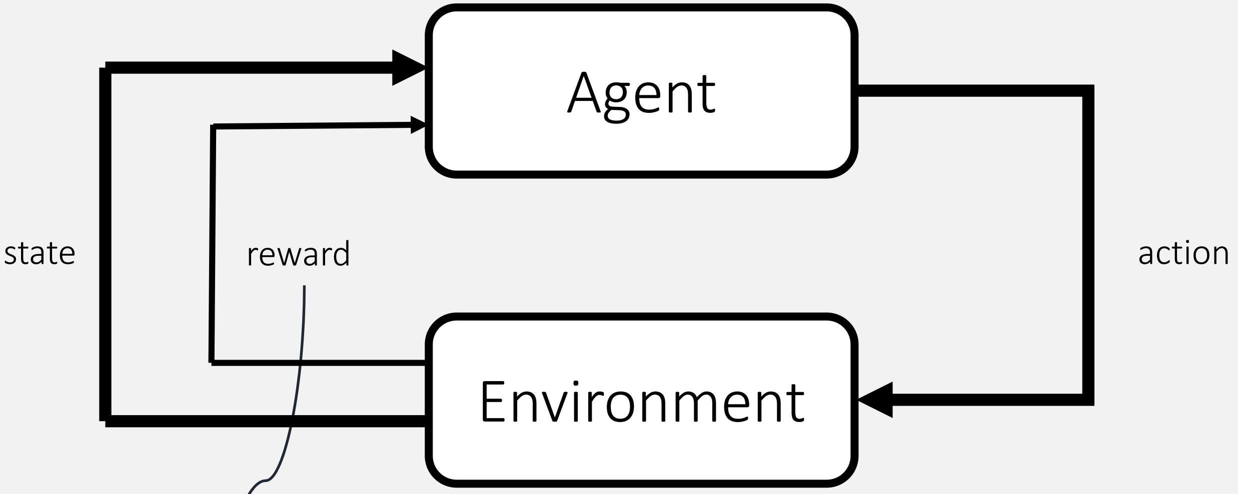




Problems in AGVs routing

3 DRL Framework

Markov decision processes (MDPs)



An MDP

$$G_t \doteq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots$$

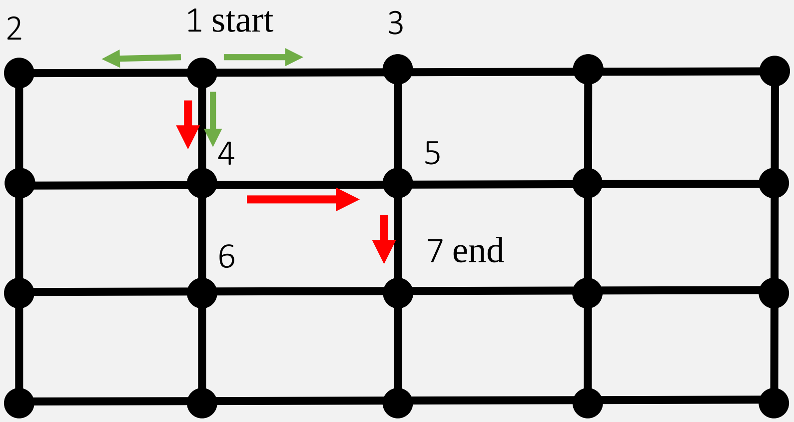
The goal to maximize

An MDP is a natural real-time responding model

Modeling Routing Problem based on MDPs

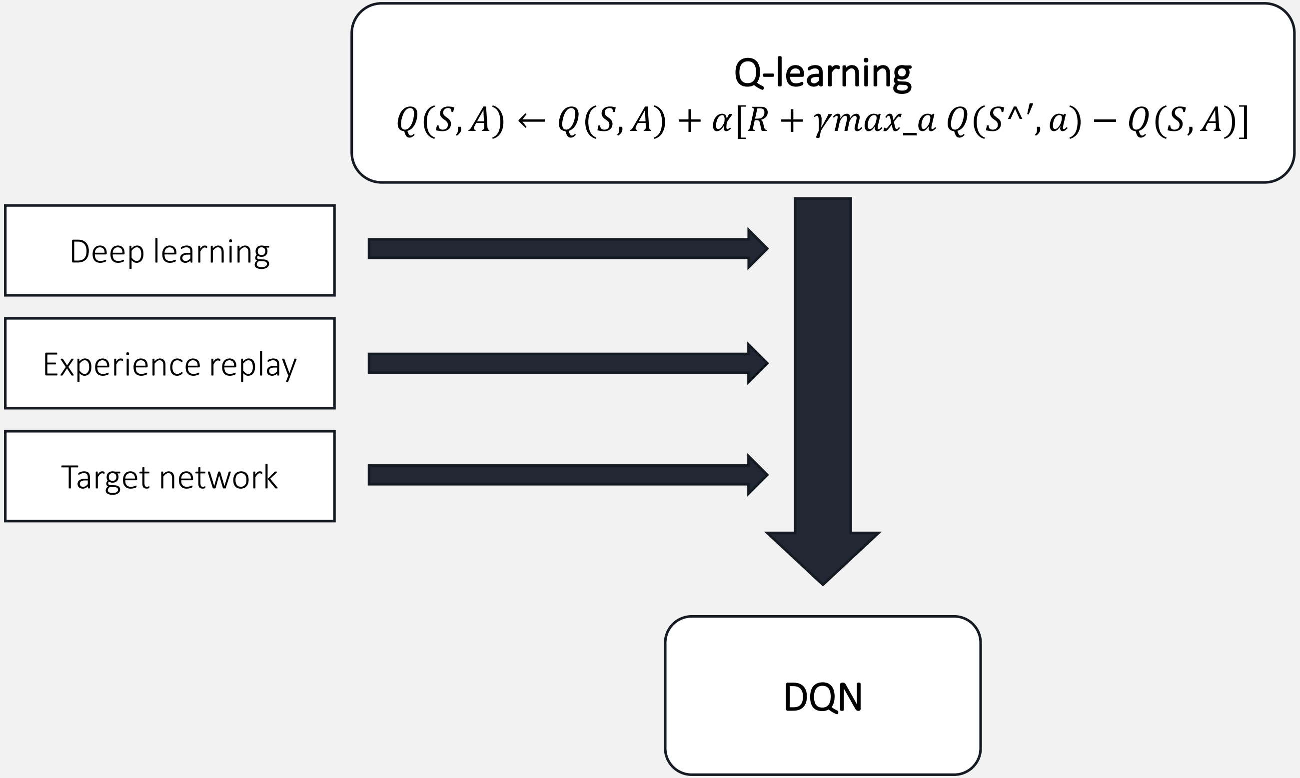
Conventional routing mode
Plan the total or a part of the route before depart

MDPs mode
A series of decisions in time sequence (decide step by step based on the real-time information)
$S_0, A_0, R_1, S_1, A_1, R_2, S_2, A_2, R_3, \ldots$



After the agent reaches point 4 from point 1, it will see the latest state which may be different from the state in point 1

DQN



Advantage of DQN:

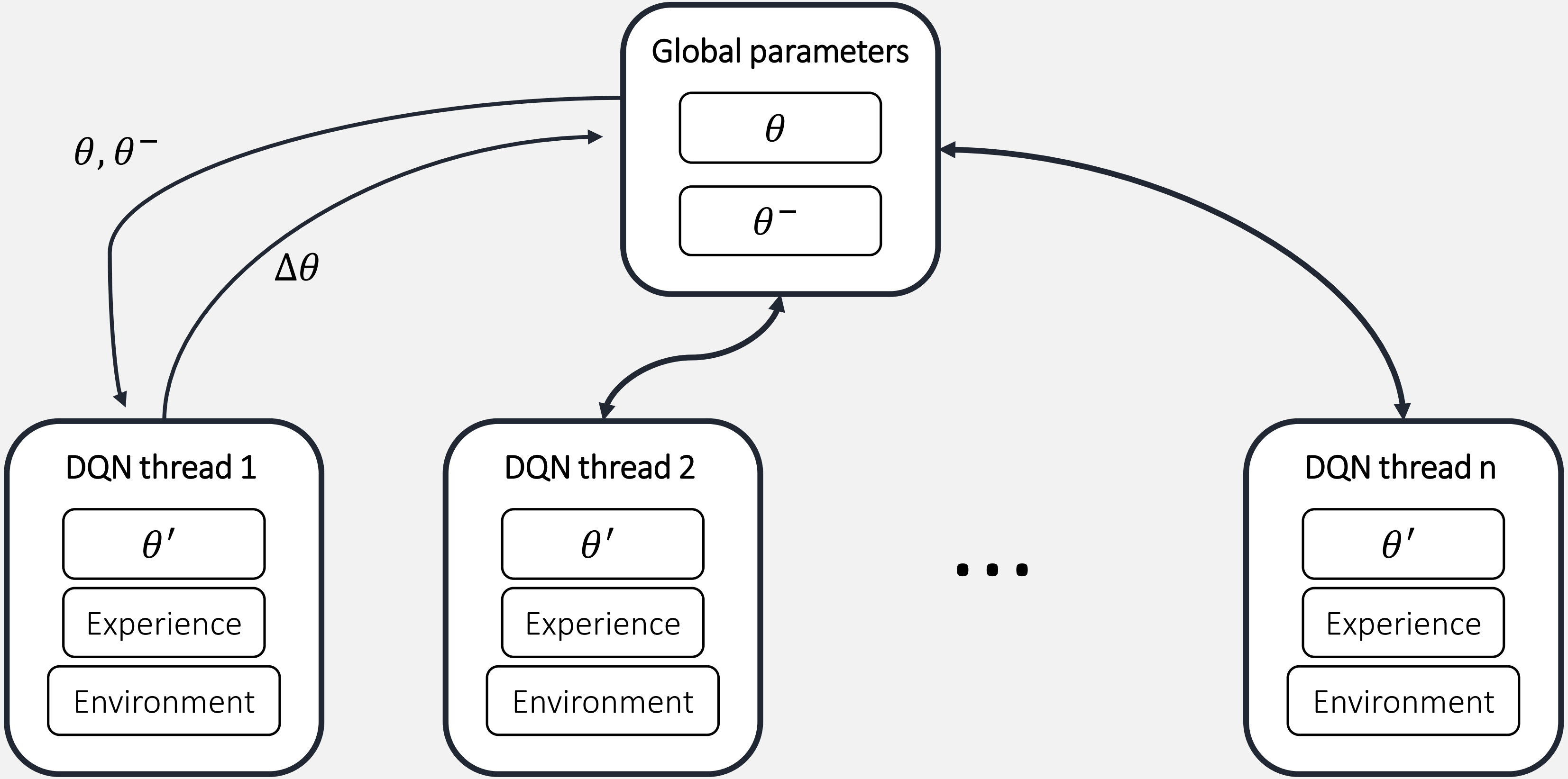
- Increase stability
- Sample efficient

Disadvantage of DQN :

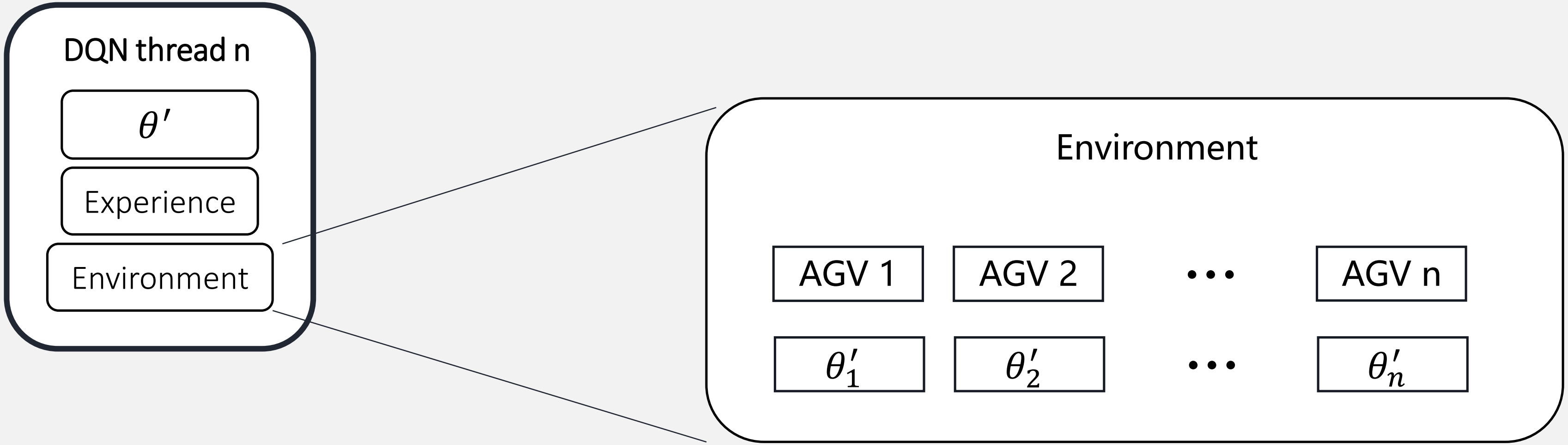
- Low training speed
- Cannot use multiple CPUs

[Mnih, V., et al., 2015]

Asynchronous DQN



Parameter sharing

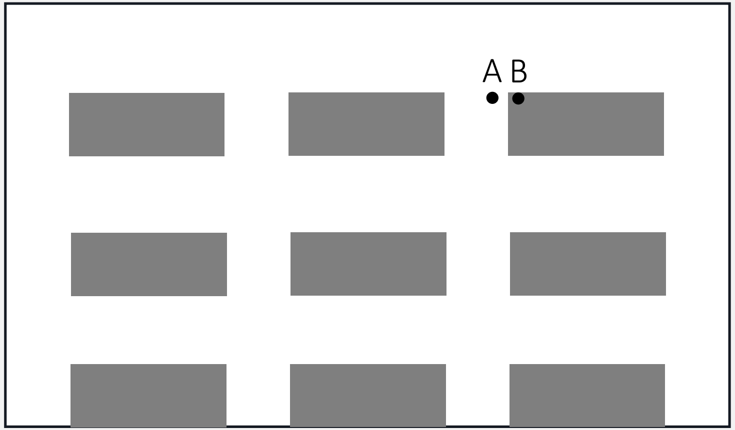


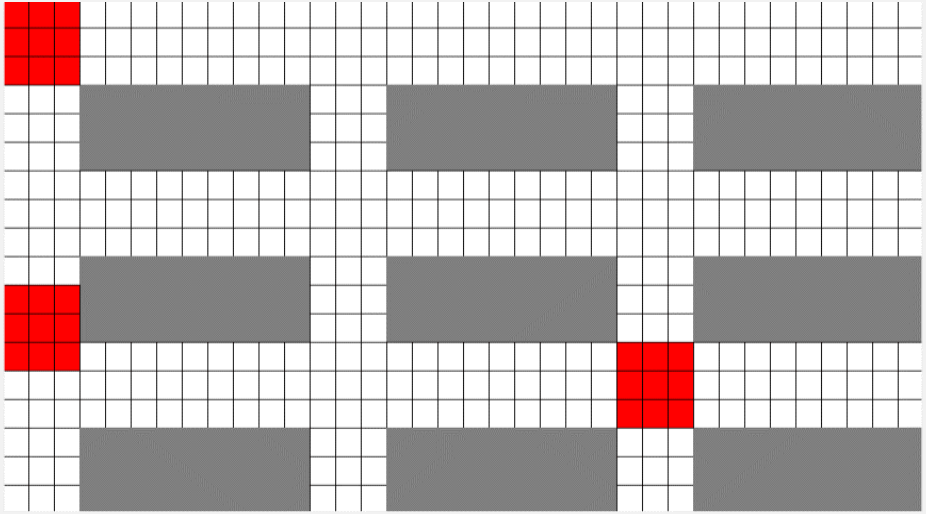
The joint action of a multi-agent problem suffers curse-of-dimensionality

Make $\theta' = \theta'_1 = \theta'_2 = \cdots = \theta'_n$

4 Feature Processing

Discretization of Continuous Features



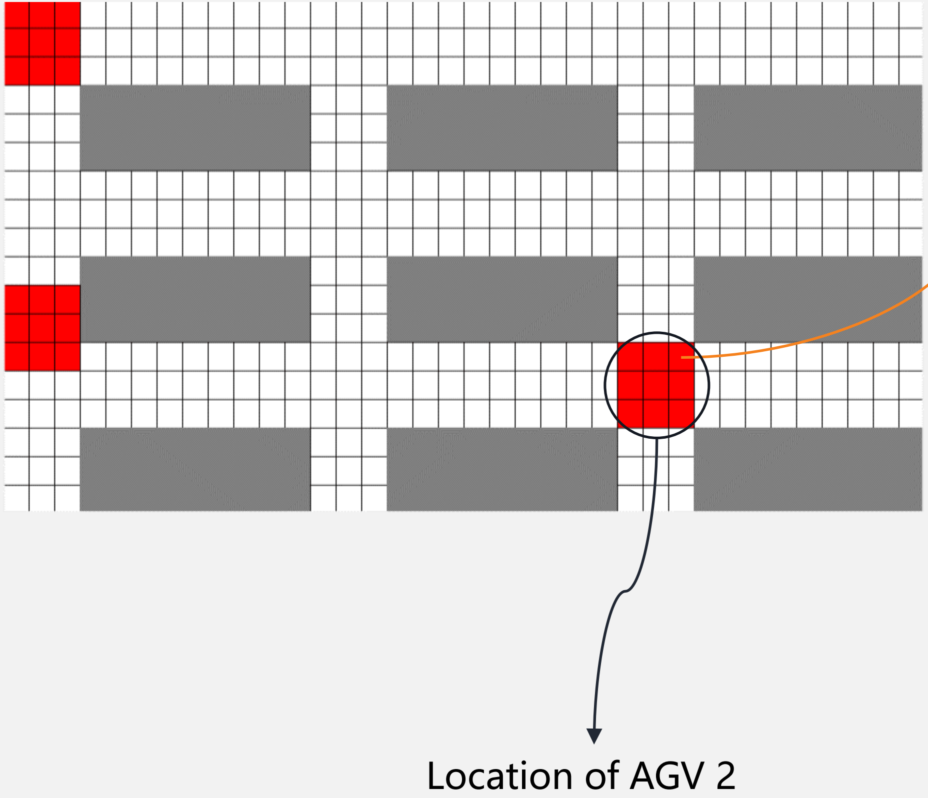


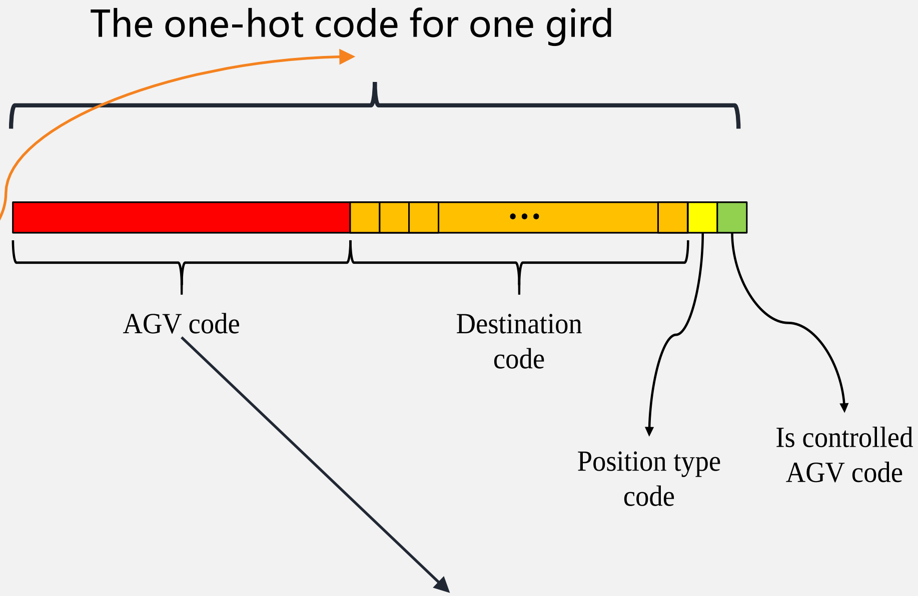
point A: (1.1, 3.2) on a track,
point B: (1.05, 3.2) on an obstacle.

Jump characteristic

Discretize the map into grids

One-hot code





Free	AGV 1	AGV 2	AGV 3
0	0	1	0

AGV code

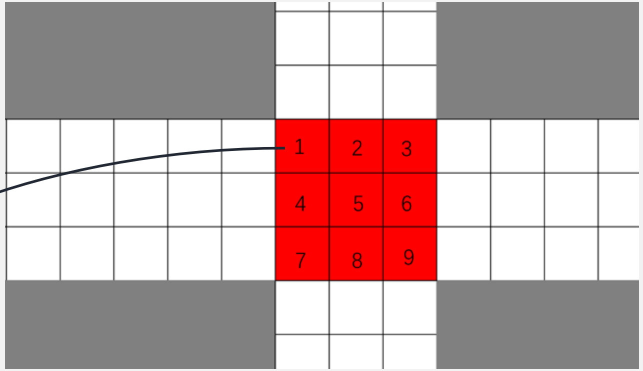
Word Embedding

cat: [-0.065, -0.035, 0.019, -0.026, 0.085,...]

dog: [-0.019, -0.076, 0.044, 0.021,0.095,...]

table: [0.027, 0.013, 0.006, -0.023, 0.014, ...]

Use an embedding vector to represent a word



1: [-0.032, -0.095, 0.039, 0.036, 0.038,...]

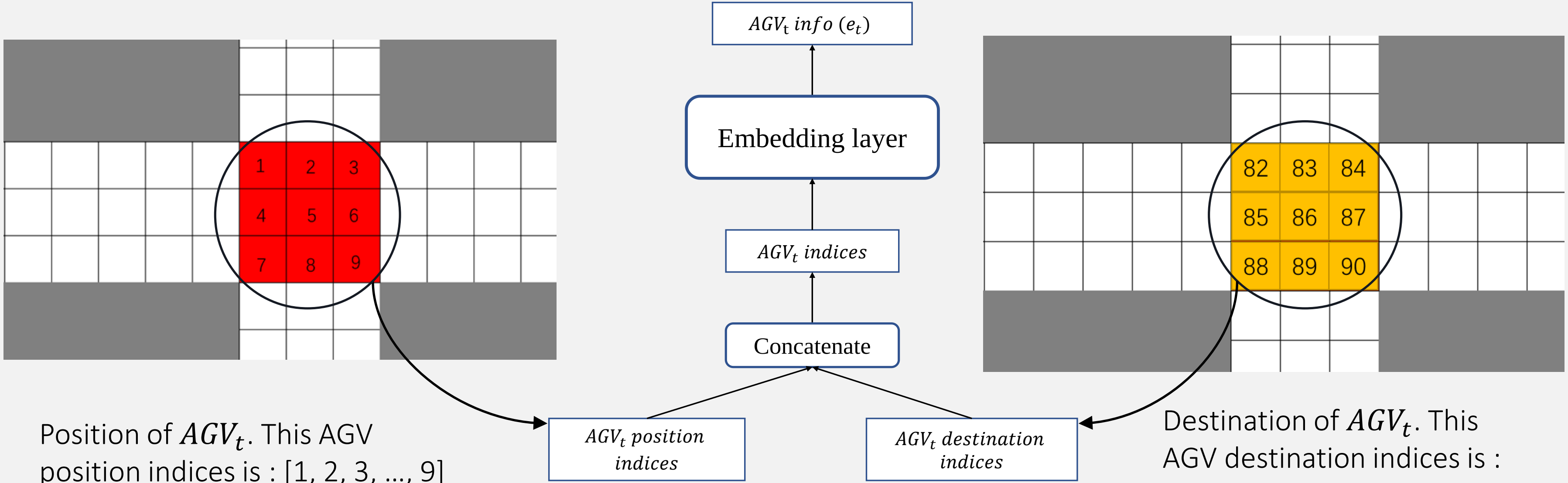
2: [-0.018, -0.076, 0.042, 0.021,0.055,...]

...

96: [0.123, 0.098, 0.066, -0.028, -0.076, ...]

Use an embedding vector to represent a grid

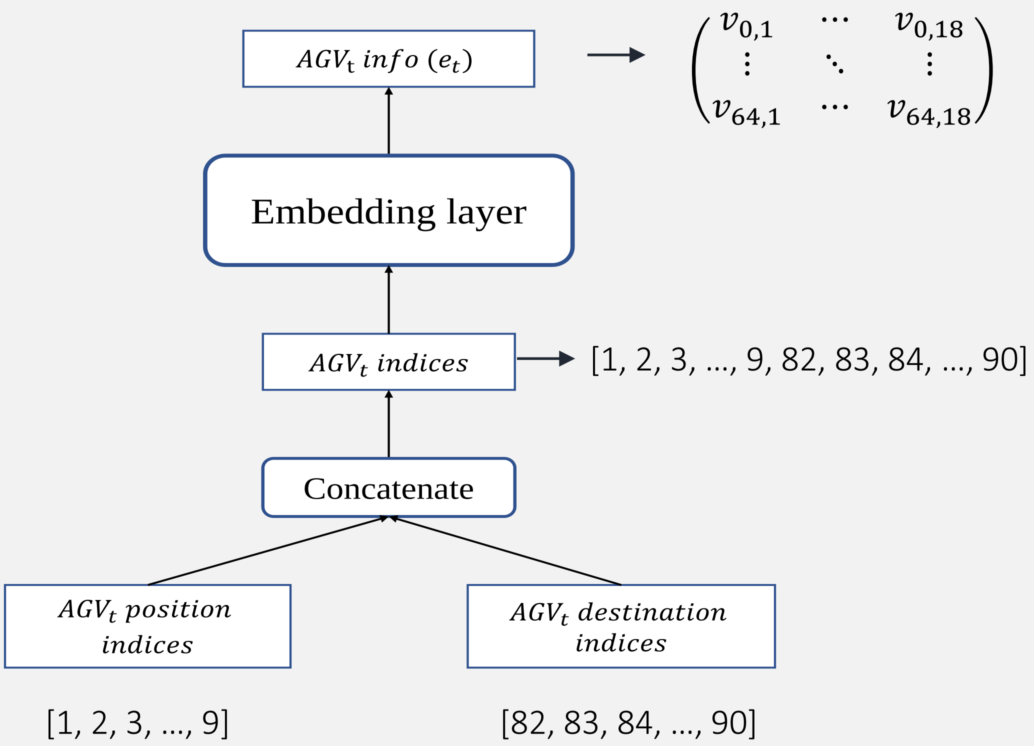
Embedding Code



Position of AGV_t. This AGV position indices is : [1, 2, 3, …, 9]

Destination of AGV_t. This AGV destination indices is : [82, 83, 84, …, 90]

Embedding Code



Suppose embedding dimension is 64

After concatenating, the length of indices is 18

Comparison of Input Size of One-hot and Embedding

Suppose:

Map discretization	Number of girds of one AGV occupies	AGVs number	Embedding dimension
n*n	m*m	k	dim

	One-hot	Embedding
Input size	$n^2(2k+4)$	$k \times dim \times m^2 \times 2$

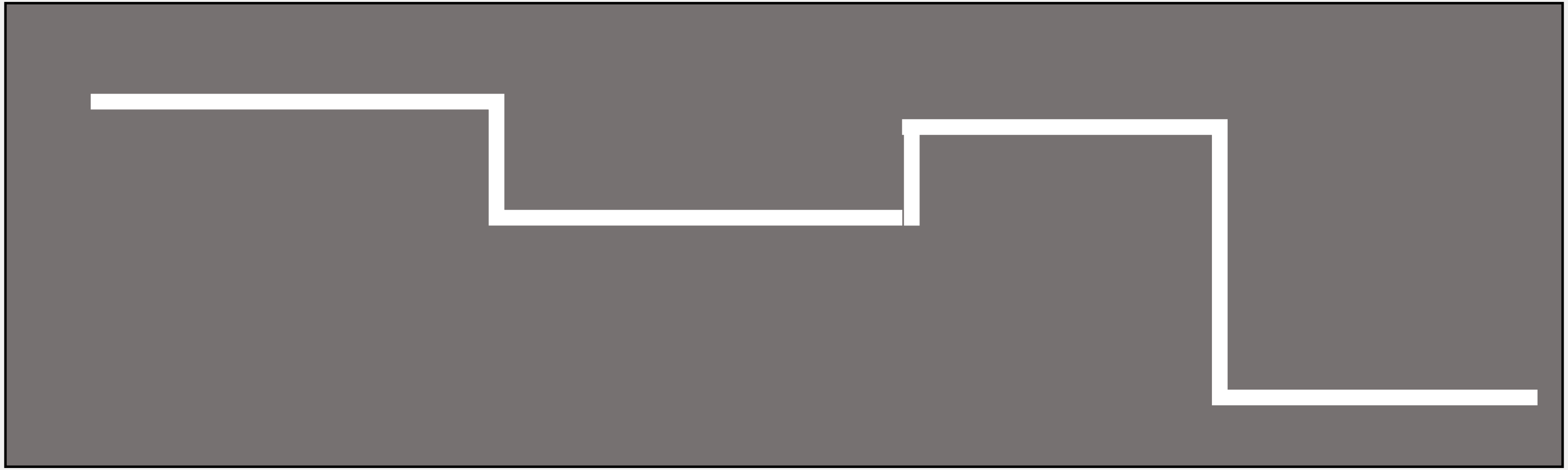
Make n=100, m=3,
a=10, dim=32

	One-hot	Embedding
Input Size	240000	5760

The embedding input is about 41.7 times smaller than the one-hot input

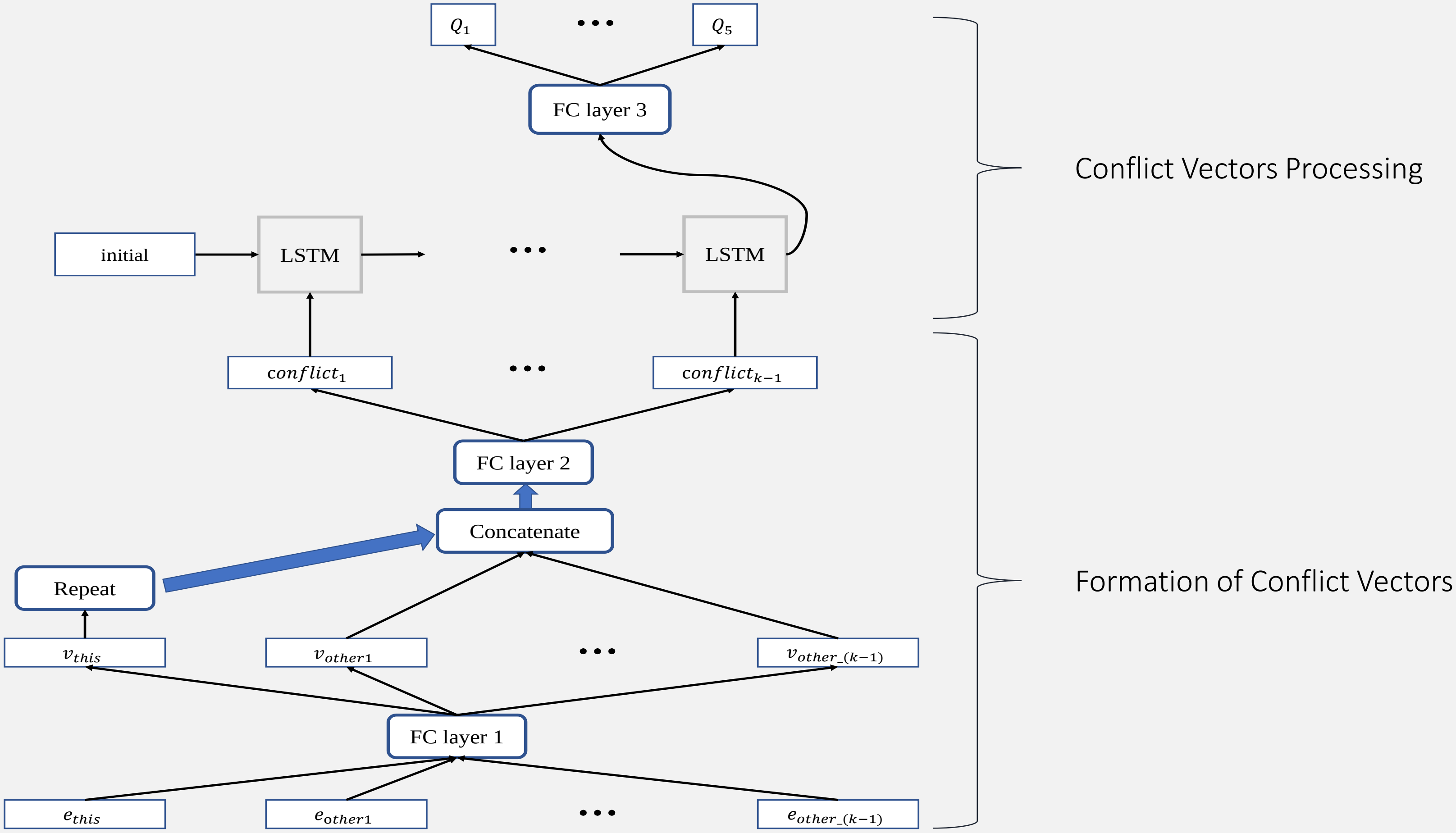
Comparison of Input Size of One-hot and Embedding

Advantage of Embedding：
- Need less data
- Suitable for complex terrains

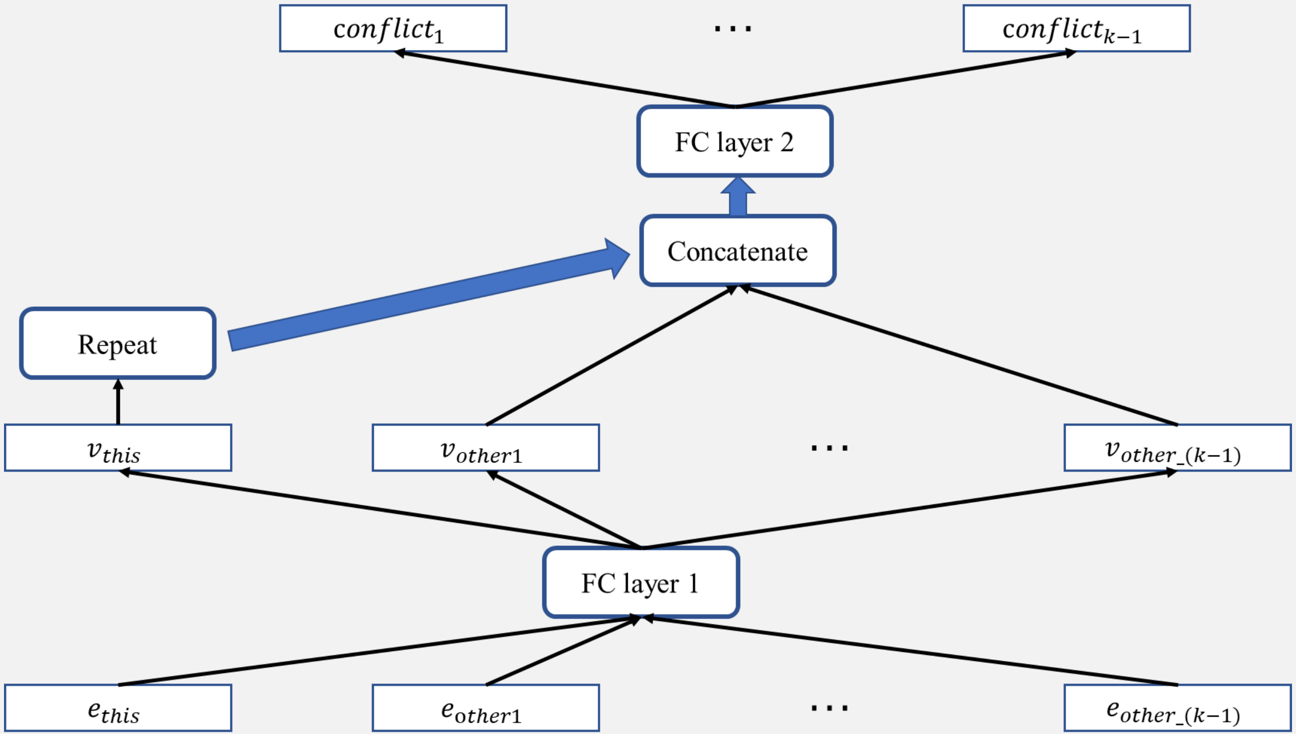


An example of sparse road scene

5 Neural Network Architecture

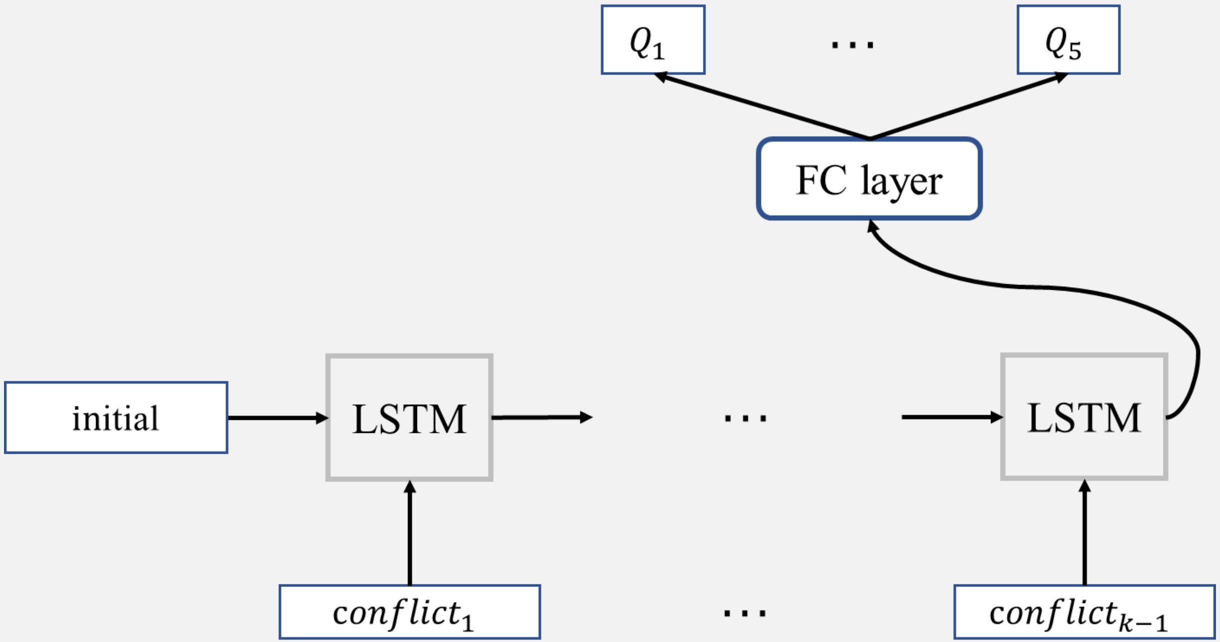

Q_1
Q_5
FC layer 3
initial
LSTM
LSTM
$conflict_1$
$conflict_{k-1}$
FC layer 2
Concatenate
Repeat
v_{this}
v_{other1}
$v_{other\_(k-1)}$
FC layer 1
e_{this}
e_{other1}
$e_{other\_(k-1)}$
Conflict Vectors Processing
Formation of Conflict Vectors

Formation of Conflict Vectors



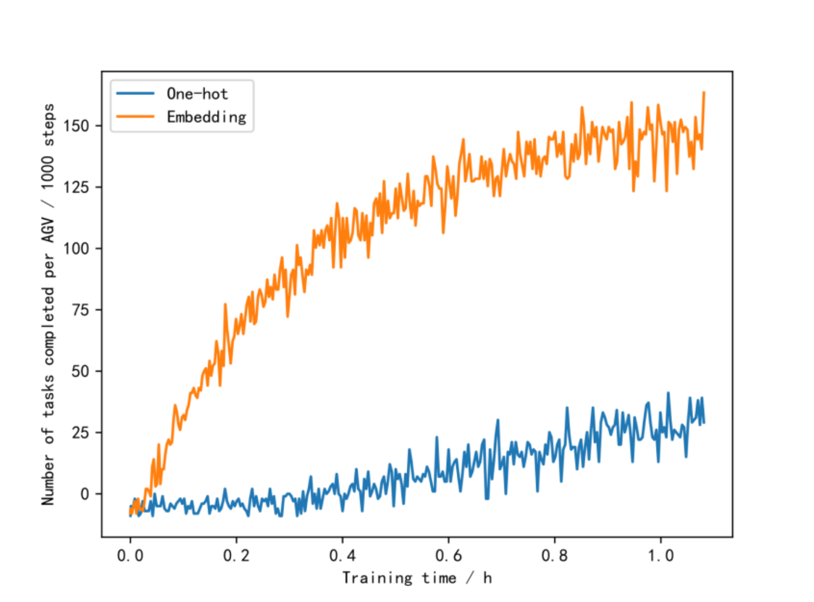
Parameter sharing. The input data should be formed from the perspective of the ego of this agent

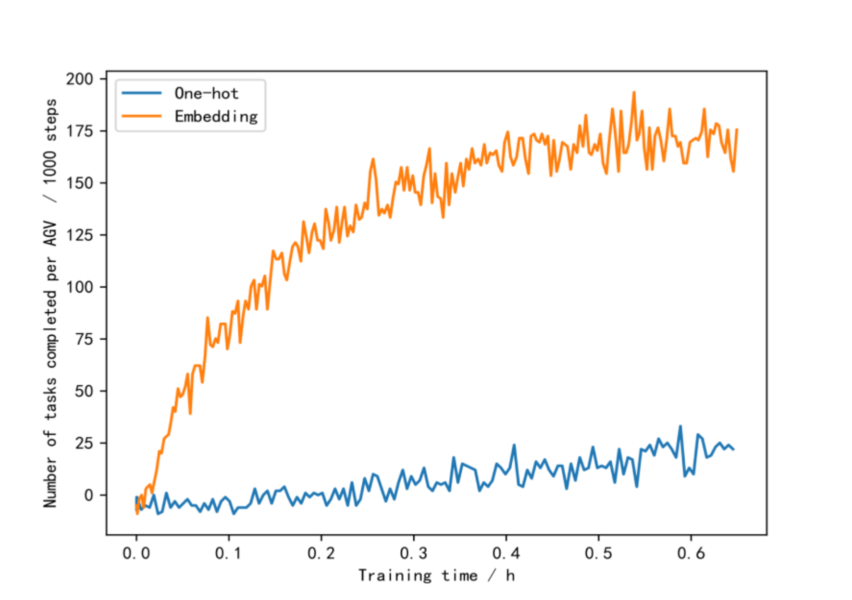
Conflict Vectors Processing



6 Experiments

Comparison of One-hot and Embedding



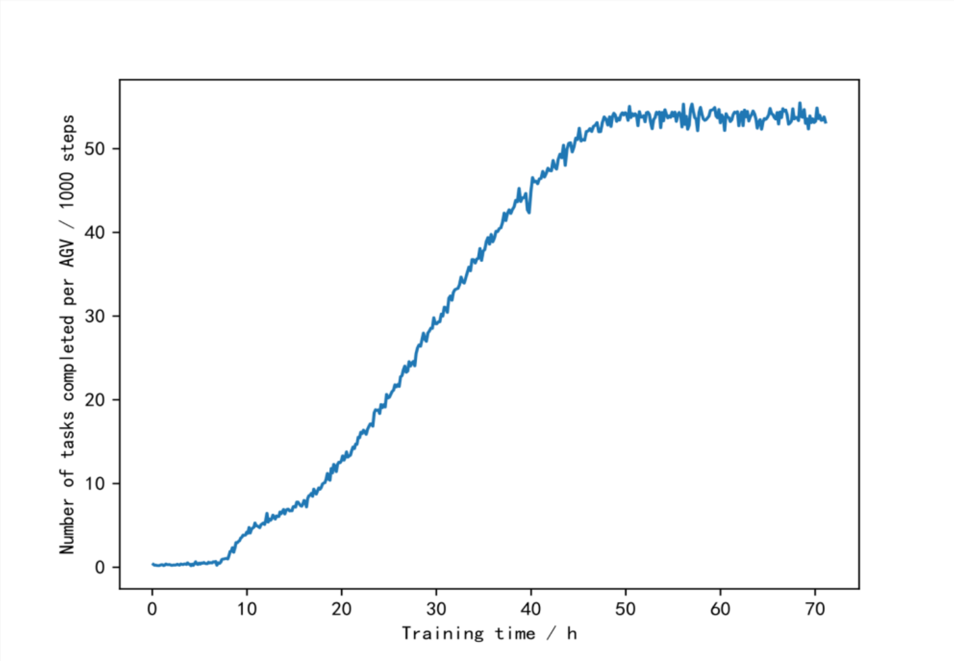


Track proportion: 62%

Track proportion: 43%

The network for one-hot: 2 conv blocks + 2 FC layers.

Results



Training process of a scene with 22 AGVs

Config:
- Grids: 28 * 14
- AGVs number: 22
- Embedding dimension: 64
- CPU: 56 cores

Results

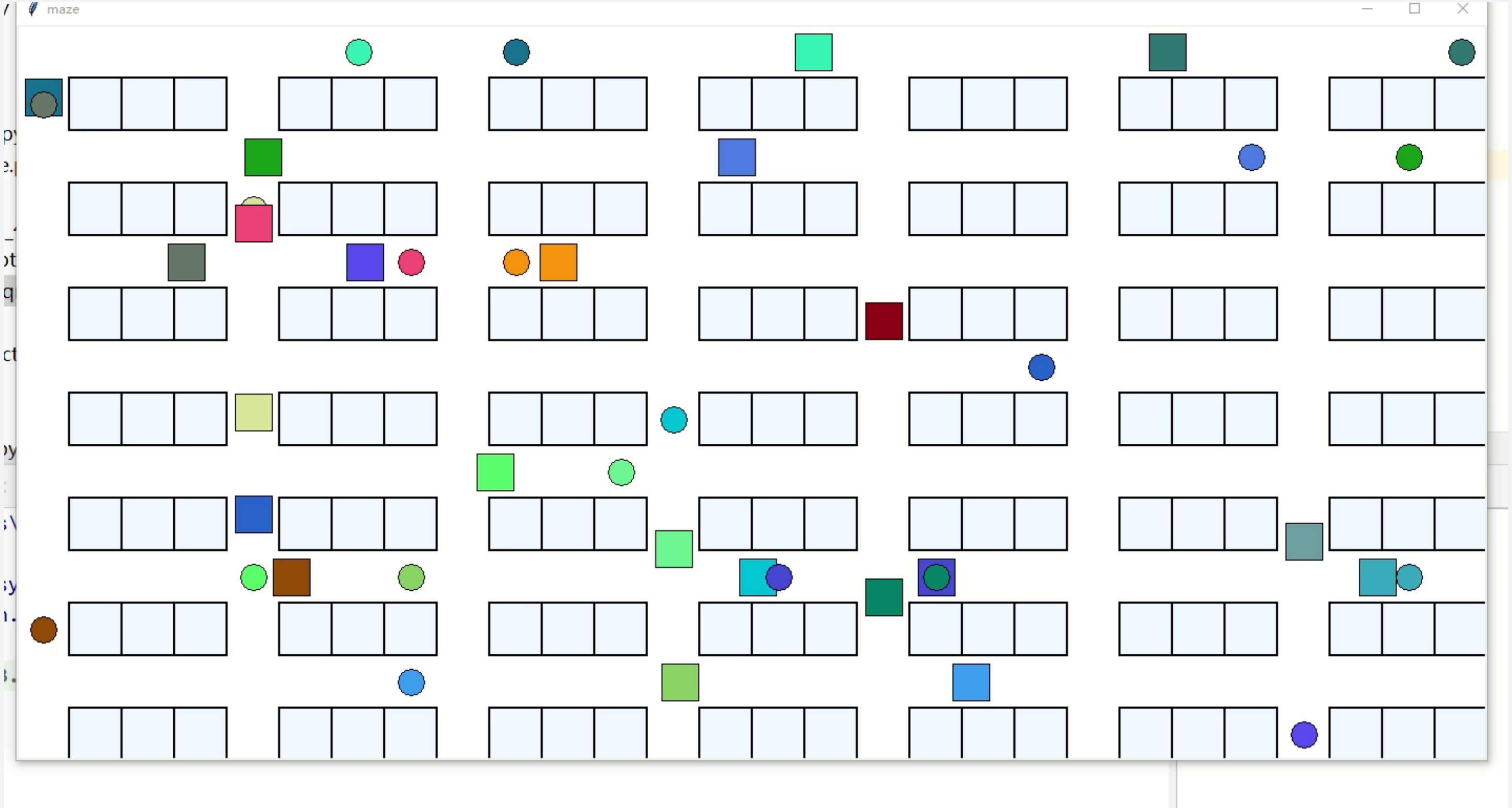
	Average Performance
Random	0.355
Regulation	46.1
Asynchronous DQN	54.2

Performance comparison between random, regulation, and asynchronous DQN

Asynchronous DQN is about 153 times better than the regulation and 18% better than the regulation method in performance.

Performance: average number of tasks completed by one AGV every 1000 steps

Simulation



22 AGVs, 56 CPUs, 72 hours

7 Conclusion

Conclusion

Method:

- Model the AGVs routing problem into an MDP.
- Improve CPU utilization by the asynchronous technique.
- Use the embedding technique to represent grids.
- LSTM is exploited to process features.

Result:

- Our model has advantages over conventional methods both in responding speed and getting more optimal solutions.

Thank you

Authors: Chengxuan Lu, Jinjun Long, Zichao Xing, Weimin Wu,
Yong Gu, Jiliang Luo, and Yi-Sheng Huang

Reporter: Chengxuan Lu