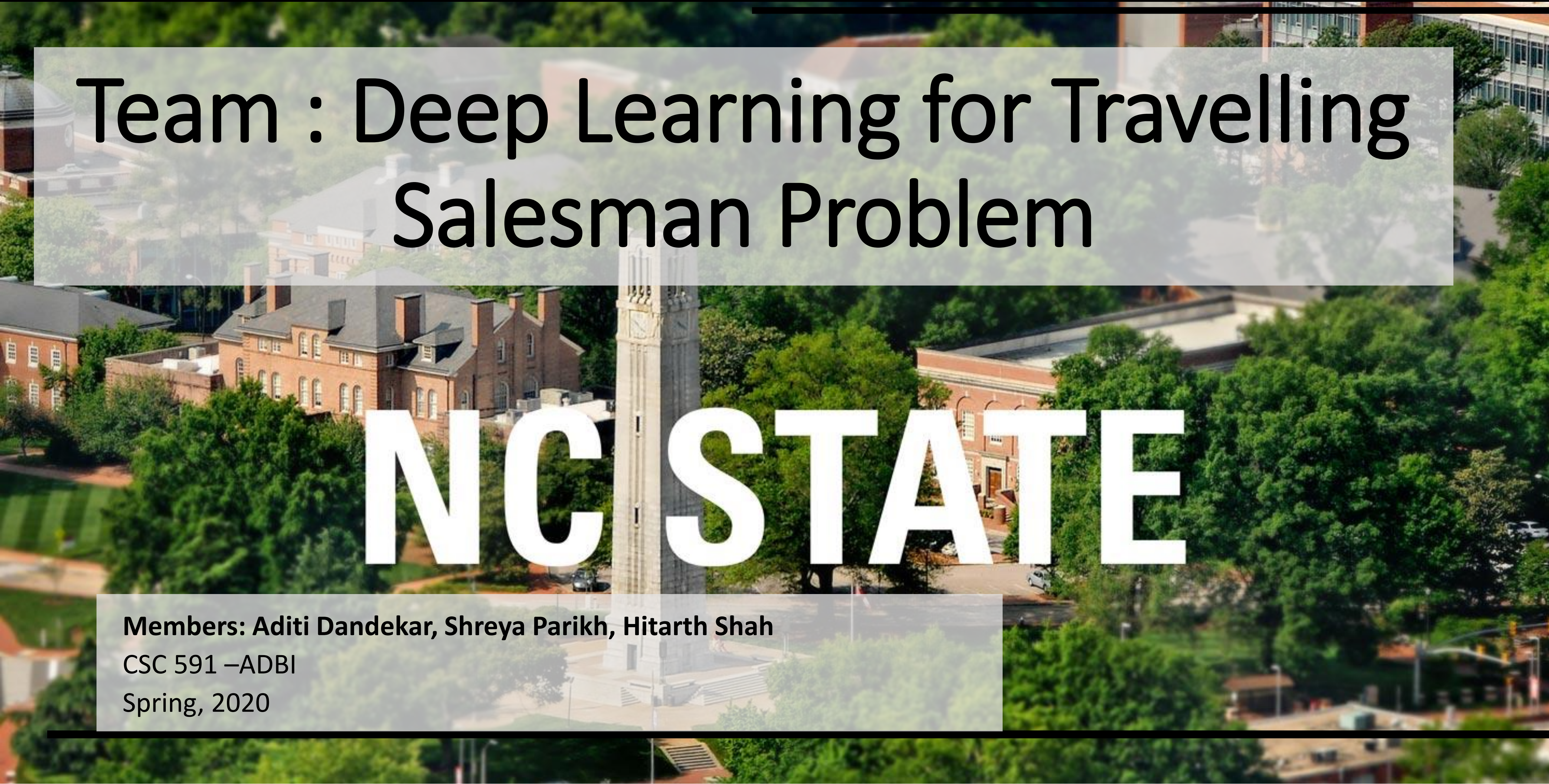


Team : Deep Learning for Travelling Salesman Problem

Members: Aditi Dandekar, Shreya Parikh, Hitarth Shah

CSC 591 –ADBI

Spring, 2020

NC STATE UNIVERSITY

PRESENTATION AGENDA

- Problem Statement
- Dataset
- Description of Algorithm
- Experiment
- Results
- Comparative Analysis
- Conclusion

PROBLEM STATEMENT

- The Travelling Salesman Problem is a well-known problem in operational research which consists in finding the shortest possible tour connecting a list of cities, given the matrix of distances between these cities.
- The travelling purchaser problem and the vehicle routing problem are both generalizations of TSP.
- The objective is to find a set of delivery routes satisfying some requirements or constraints and giving minimal total cost.
- In our proposed work, we will develop a framework with the capability of solving TSP using Deep Learning.

DATASET

The algorithm will generate random graphs with number of nodes 10, 20, 30, etc.

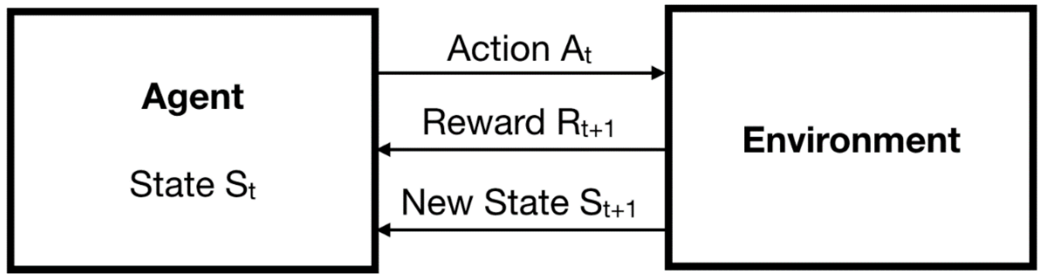
The graphs generated are fully connected.

The nodes are placed uniformly at random on the [0,10]x[0,10] square.

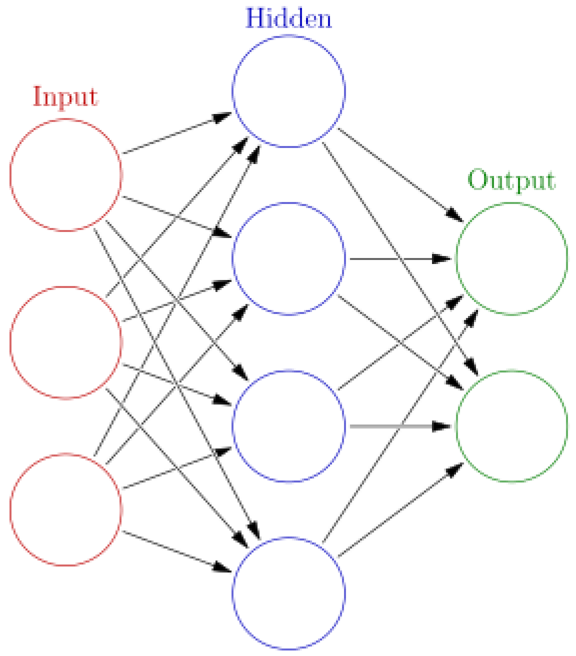
The edge weights are simply all the pairwise Euclidean distances between the nodes.

METHODOLOGY

REINFORCEMENT LEARNING



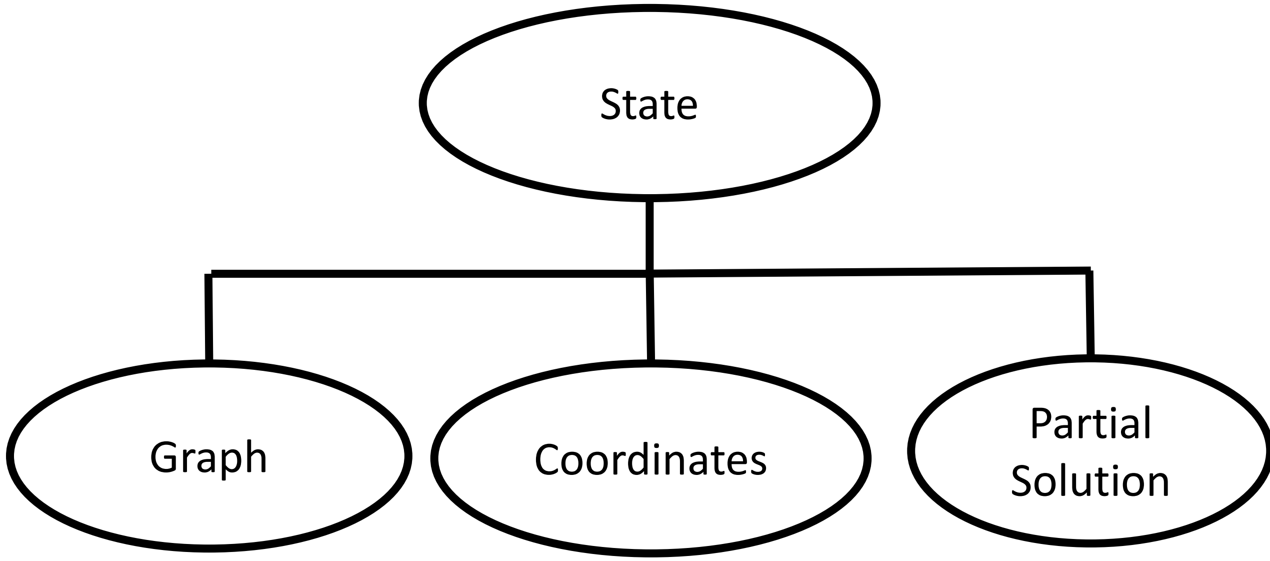
NEURAL NETWORKS



Toolbox: Python and PyTorch

REINFORCEMENT LEARNING

- Agent is interested in finding short tours.
- Environment is fully observed and trivial.
- Agent makes a deterministic move along the graph edge and observes the travelled distance.



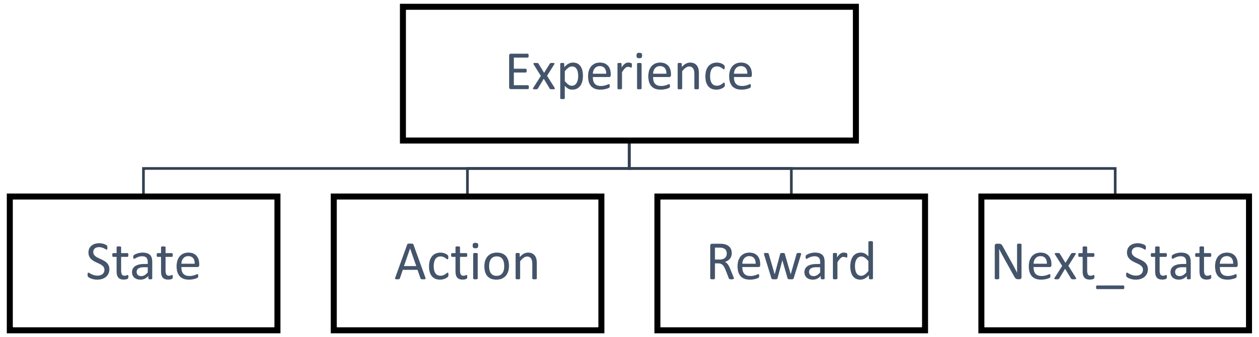
DESCRIPTION OF ALGORITHM

Q-Learning : seeks to learn a policy that maximizes the total reward

- Is an off-policy reinforcement learning algorithm that seeks to find the best action to take given the current state.
- Learns an *action-value function* $Q(s, a)$: for a given state *s* and a given action *a*, returns the expected cumulative reward that the agent will obtain until the end of one instance of the TSP.
- When we have all the possible actions, we let the agent pick the actions *a* that maximizes the estimated expected return $Q(s, a)$ from any state *s*.

Q-FUNCTION NEURAL NETWORK

- While training the agent, we store it’s experience in the memory.
- A random batch of experiences is picked from the memory and targets are defined for each of these experiences.
- These targets are used by the Network to learn the best action from a given state.



EPSILON AND EPISODES

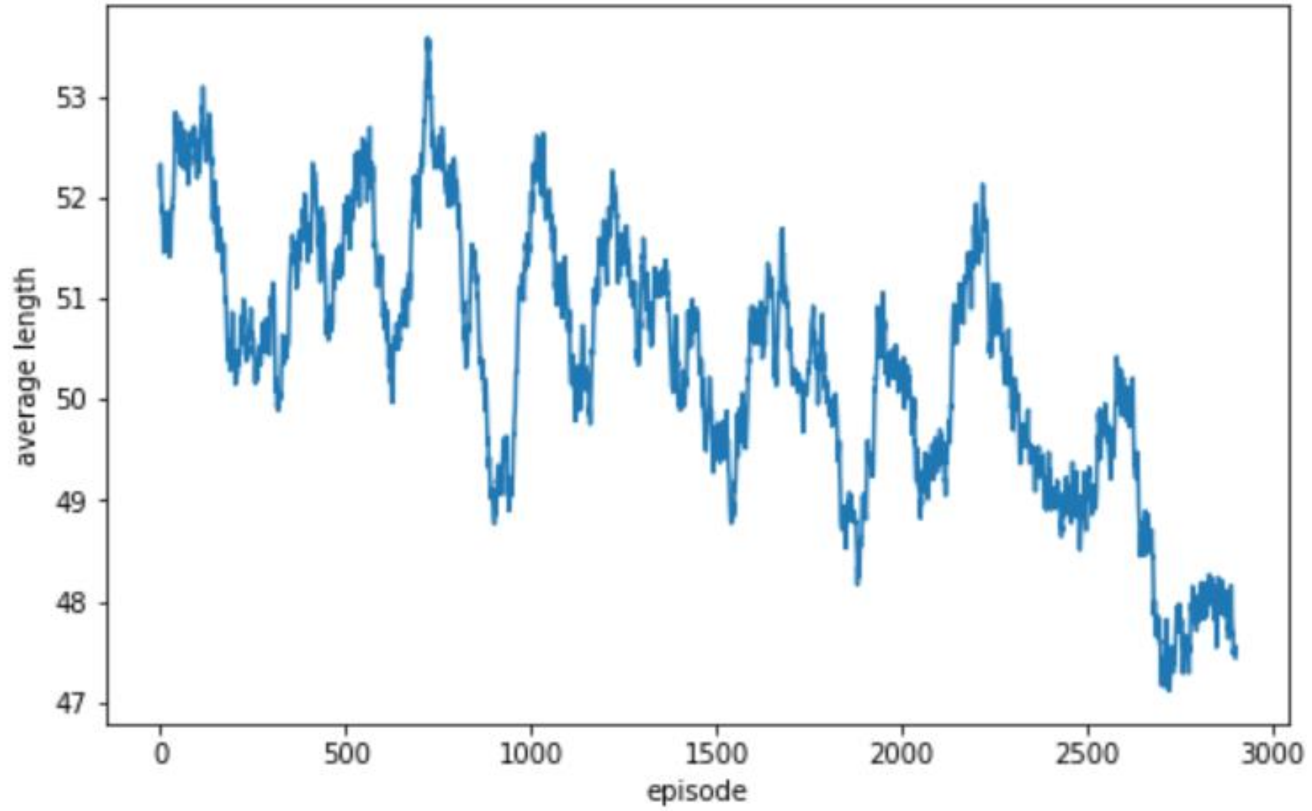
- Epsilon is set to an initial value close to 1 in the beginning with an appropriate decay to limit the randomness over episodes
- In order to store the experiences(state – action – reward) from Q-Learning, the memory capacity was set to 10000
- For convergence purposes, the number of episodes was set to 3000

RESULTS

At the beginning of the training,

- the function Q() is randomly initialized
- decisions are uniformly random
- probability E is close to 1

```
Text(0.5, 0, 'episode')
```



Moving Average of the Total Distance over the Course of Training for 10 Nodes

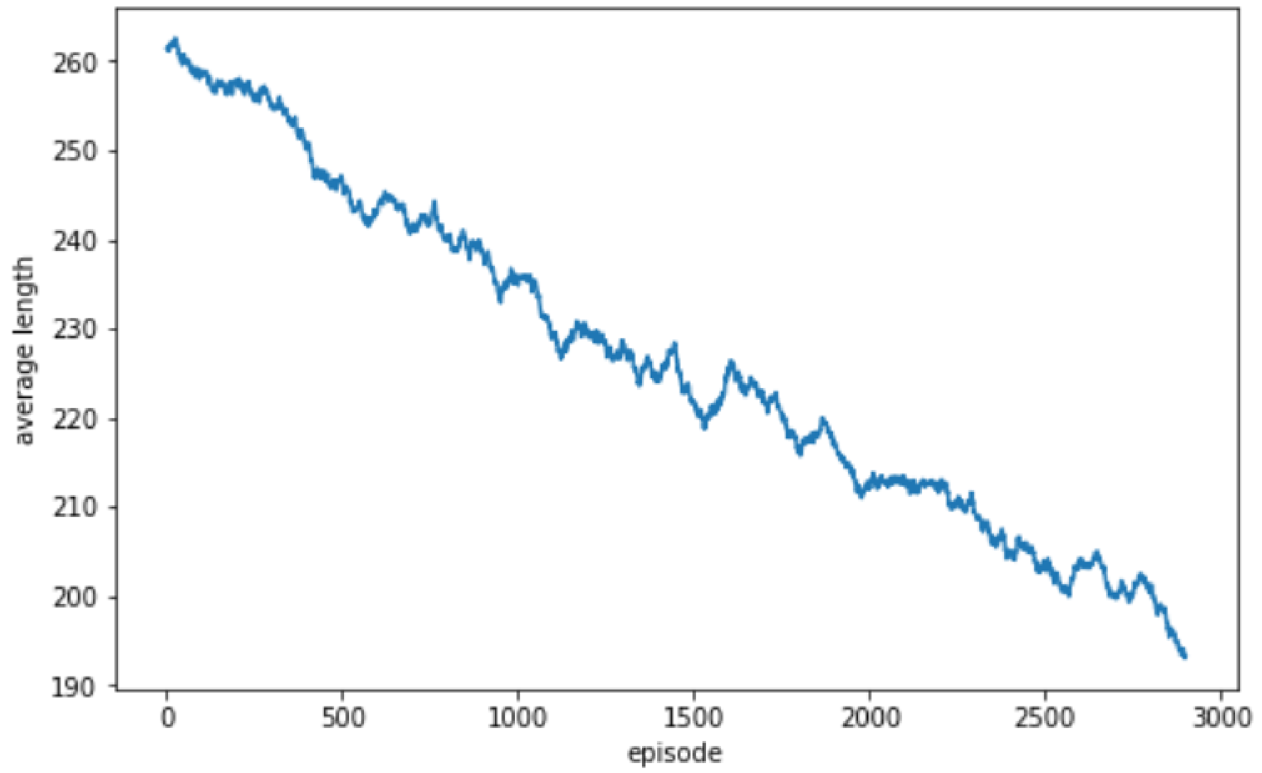
NC STATE UNIVERSITY

RESULTS

Towards the end of the training,

- Q() is better
- decisions are increasingly better
- probability E is smaller

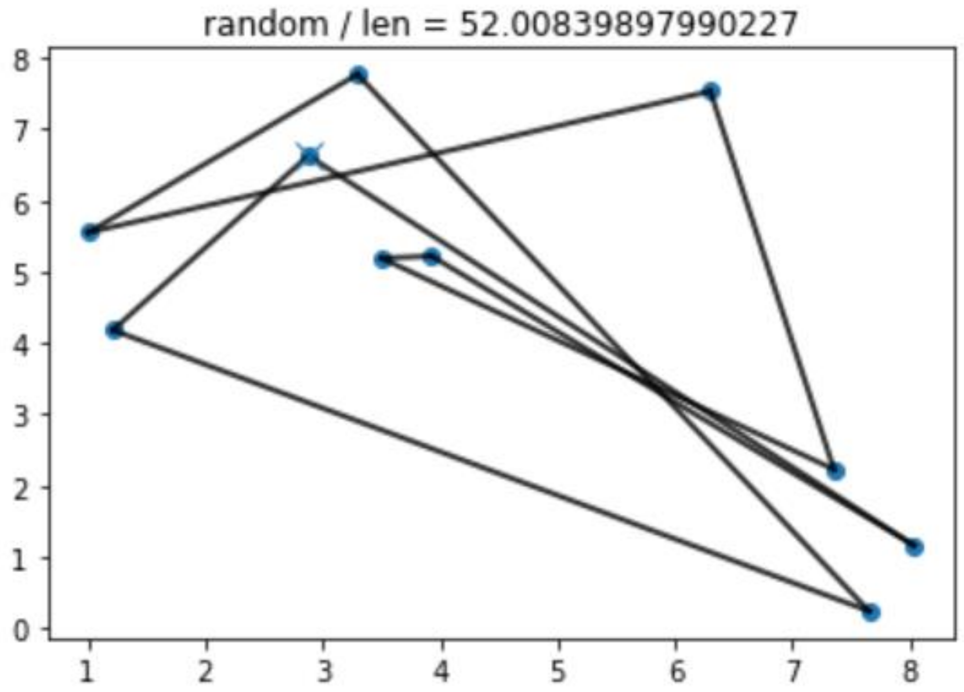
```
Text(0.5, 0, 'episode')
```

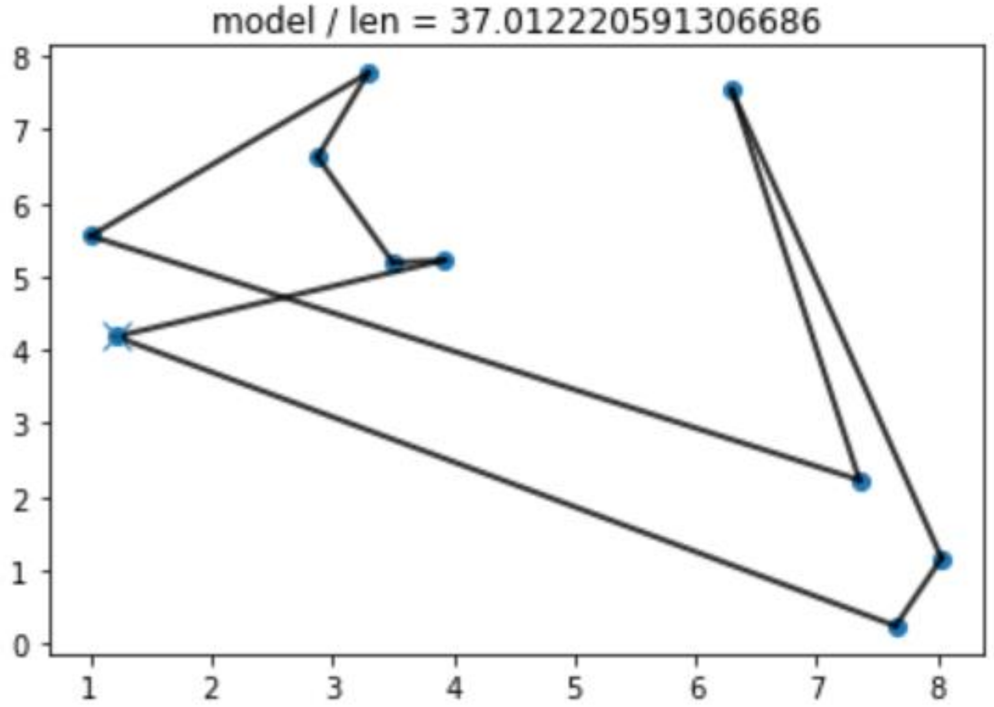


Moving Average of the Total Distance over the Course of Training for 50 Nodes

RANDOM

LEARNING AGENT





There are some examples in which the agent is failing and performing like random strategy. But we can see in the above example, that the learning agent does not always fail. Also, we can observe a pattern where agent is preferring a path which means the agent is learning.

COMPARATIVE ANALYSIS

NUMBER OF NODES	Q-LEARNING	NEAREST NEIGHBORS	GOOGLE ORTOOLS
10	37.01	25.5	19
20	73.15	40.11	27
50	208.85	58.24	23

CONCLUSION

- Although we have never encoded any rule about how to find shortest total distance, the agent is able to make a better decisions by just observing the travelled distance.

- As we can see in the comparative analysis, we are no where near to the performance of TSP solvers(Google ORTOOLS and Nearest Neighbor Algorithm), but we can see that our learning algorithm has given pretty good results.

THANK YOU