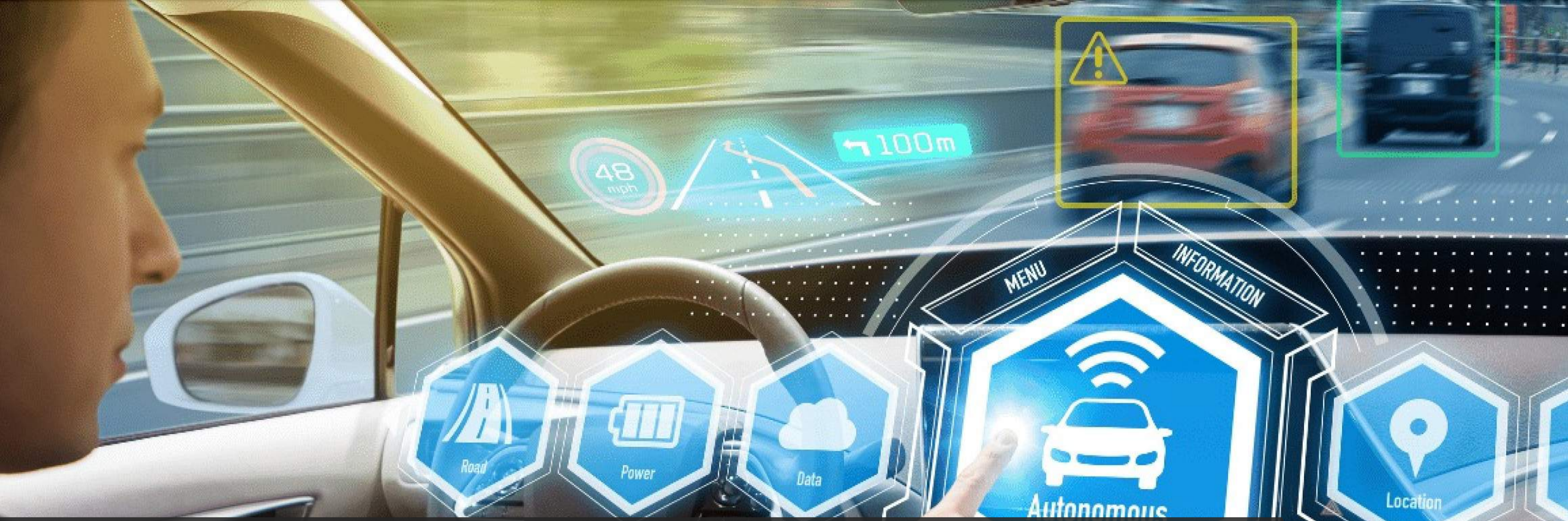


Self-Driving Car for Highway Intersection in Mixed Traffic

By,
Raghava Devaraje Urs
Shiv Kumar Ganesh
Kumuda B G Murthy

CMPE-260 Sec 49 – Reinforcement Learning

Significance To The Real World

A self-driving car is an autonomous vehicle that has the ability to sense its surroundings and move safely without any human intervention. This technology has gained high recognition in the 21st century.

In our project, we plan to implement Reinforcement learning based simulated self-driving car using DQN algorithm for intersection. Our feature scope expansion is fine tuning the model and park car automatically when no objects are seen in a parking spot.

Why Reinforcement Learning for Self Driving

Because the cost of human driving data collection at large scale can be prohibitive, another promising approach is training a policy in simulation using reinforcement learning.

In Reinforcement Learning, the agent or decision-maker learns what to do—how to map situations to actions—so as to maximize a numerical reward signal.

We evaluate our model on a challenging intersection-crossing task involving up to 15 vehicles perceived simultaneously. We show that our proposed method provides significant quantitative improvements, and that it enables to capture interaction patterns in a way that is visually interpretable.

Goal of the Project

We study the design of learning architectures for behavioral planning in a dense traffic setting. Such architectures should deal with a varying number of nearby vehicles, be invariant to the ordering chosen to describe them, while staying accurate and compact.

We observe that the two most popular representations in the literature do not fit these criteria and perform badly on a complex negotiation task.

We propose an attention-based architecture that satisfies all these properties and explicitly accounts for the existing interactions between the traffic participants. We show that this architecture leads to significant performance gains and can capture interactions patterns that can be visualized and qualitatively interpreted.

Reinforcement Learning Terminologies

Agent: Car

Environment : Highway environment – Intersection

State: Position, Heading, Velocity

Action: Slower , Faster, No Operation

Rewards: Drive with a speed, Avoid Collison with neighboring vehicle

Deep Q-Network

1) DQN algorithm combines the Q-Learning algorithm with deep neural networks (DNNs).

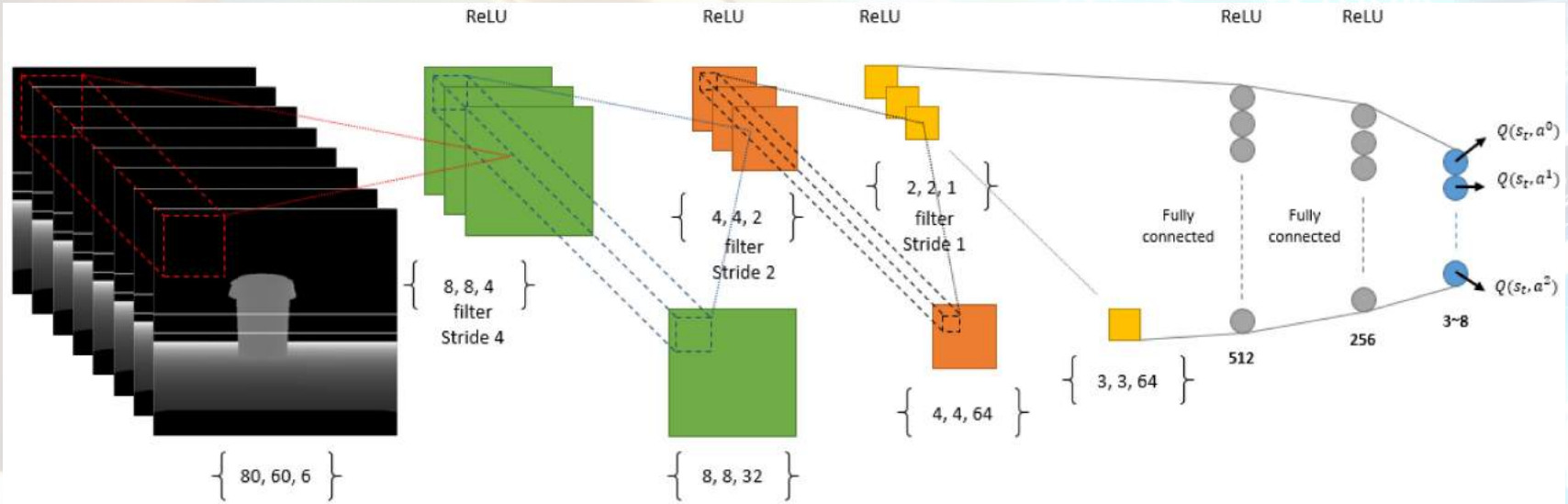
2) As it is well known in the field of AI, DNNs are great non-linear function approximators. Thus, DNNs are used to approximate the Q-function, replacing the need for a table to store the Q-values

3) The first one is called the main neural network, represented by the weight vector θ, and it is used to estimate the Q-values for the current state s and action a: Q(s, a; θ).

4) The second one is the target neural network, parametrized by the weight vector θ´, and it will have the exact same architecture as the main network, but it will be used to estimate the Q-values of the next state s´ and action a´.

All the learning takes place in the main network. The target network is frozen (its parameters are left unchanged) for a few iterations (usually around 10000) and then the weights of the main network are copied into the target network, thus transferring the learned knowledge from one to the other. This makes the estimations produced by the target network more accurate after the copying has occurred.

Working of DQN



On a higher level, Deep Q learning works as such:

1) Gather and store samples in a replay buffer with current policy

2) Random sample batches of experiences from the replay buffer (known as Experience Replay)

3) Use the sampled experiences to update the Q network

4) Repeat 1-3

Bellman Optimality Equation

The optimal action-value function $Q^* = \max_\pi Q^\pi(s)$ satisfies the Bellman Optimality Equation:

$$Q^*(s,a) = (\mathcal{T}Q^*)(s,a) \stackrel{\text{def}}{=} \mathop{\mathbb{E}}_{s' \sim P(s'|s,a)} \max_{a' \in A} [R(s,a) + \gamma Q^*(s',a')]$$

The Deep Q-Network (DQN) algorithm implements this idea by using a neural network model to represent the action-value function Q.

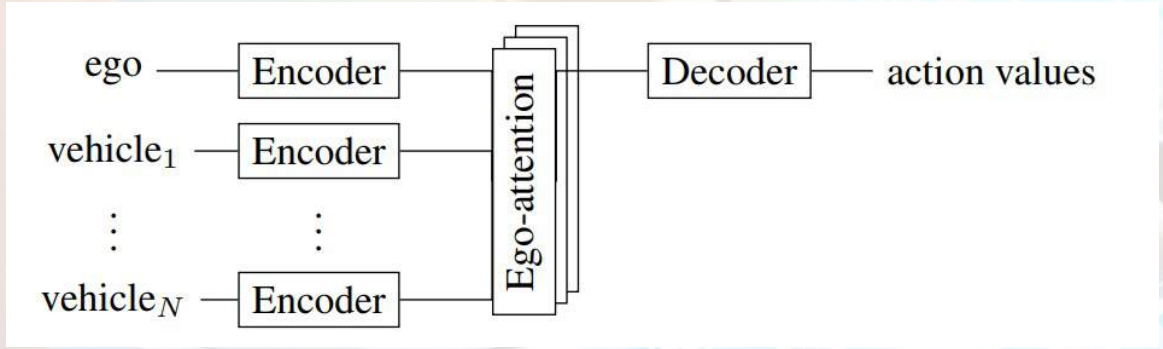
Attention Mechanism

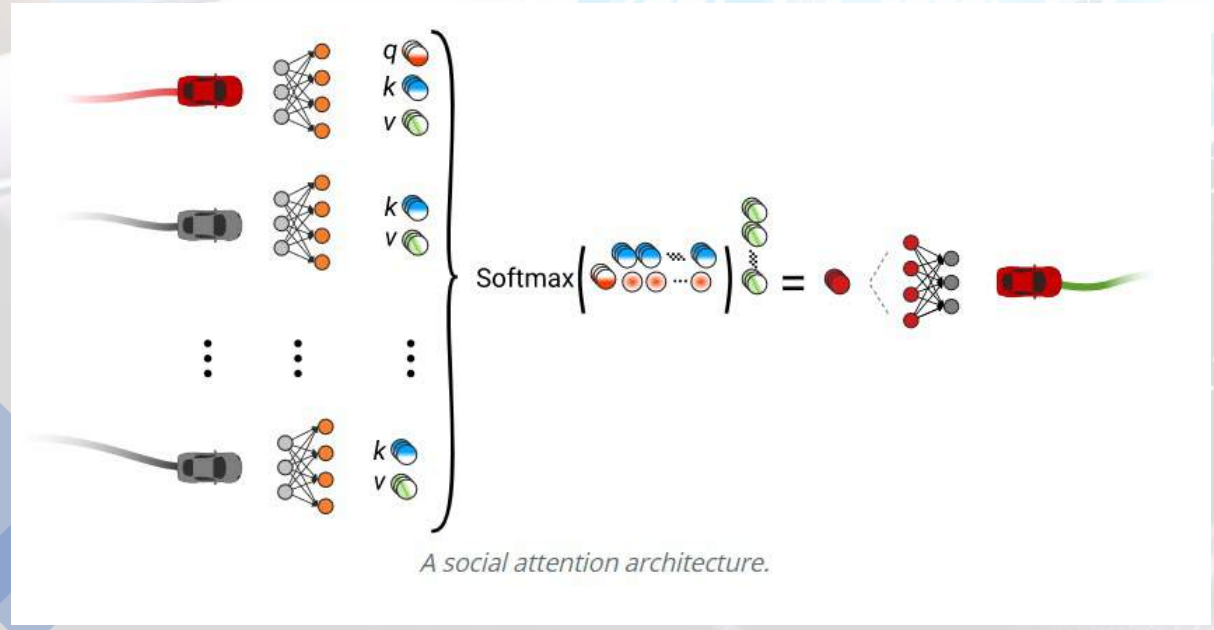
In order to apply a reinforcement learning algorithm such as DQN to an autonomous driving problem, a state space S must first be chosen, that is, a representation of the scene. When social interactions are relevant to the decision, the state should at least contain a description of every nearby vehicle. A vehicle driving on a road can be described in the most general way by its continuous position, heading and velocity. Then, the joint state of a road traffic with one ego-vehicle

Attention mechanisms: The attention architecture was introduced to enable neural networks to discover inter-dependencies within a variable number of inputs. It has been used for pedestrian trajectory forecasting with spatiotemporal graphs and with spatial and social attention using a generative neural network.

In the present work, we use a multi-head social attention mechanism to capture vehicle-to-ego dependencies and build varying input size and permutation invariance into the policy model.

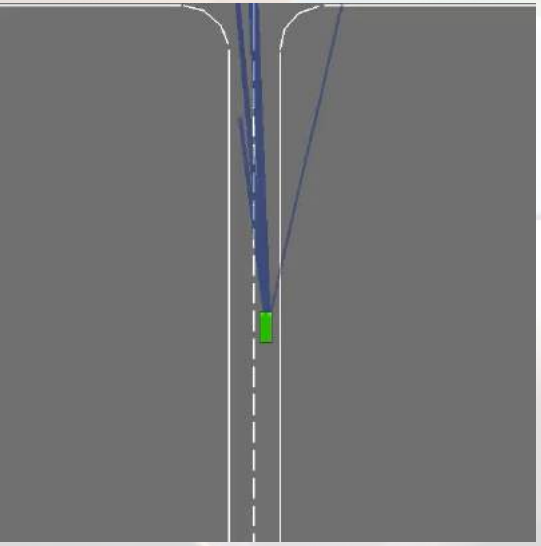
Model Architecture



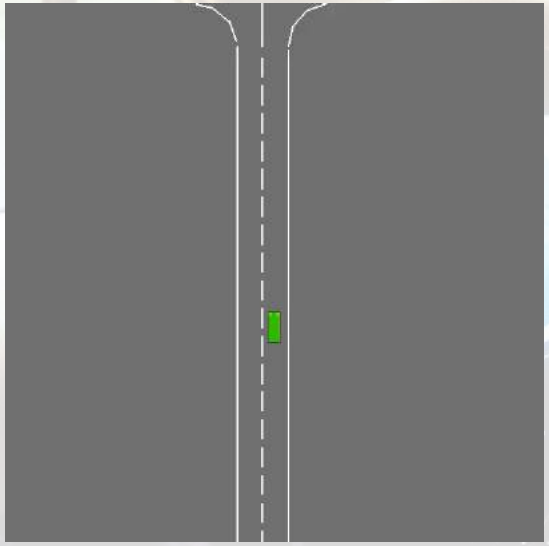


A social attention architecture.

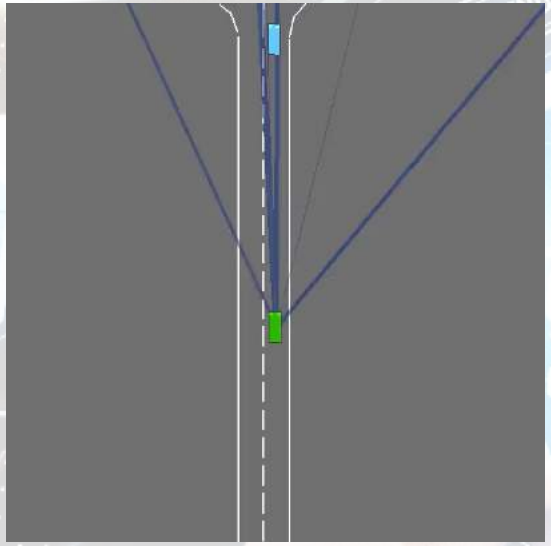
Hyperparameter Tuning



Gamma = 0.95



Gamma = 0.98



Gamma = 0.99

Demo

https://github.com/vrmusketeers/RL_Self_Driving_Car_Intersection/blob/main/RL_Self_Driving_Car_Intersection_DQN.ipynb

THANK YOU