Applying Deep Q-learning approach to Pacman problem

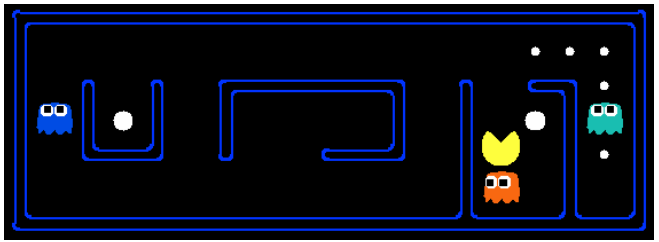
Rana Genedy, grana@vt.edu

Rhea Saxena, rhea2809@vt.edu

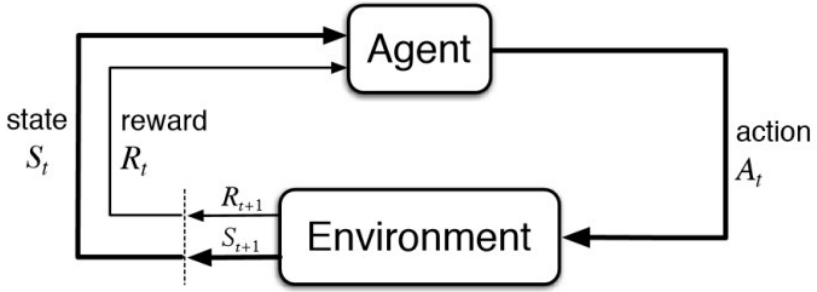
Pacman

Pac-man is a popular control problem in the field of AI, it's simple enough to understand but complex enough that it requires implementation of various strategies to find optimal solutions.

We use the original Berkeley code base to implement the base Pac-man game and implement simple Q-learning and Deep Q-Learning to compare results.



Reinforcement Learning



Reinforcement learning (RL) is an area of machine learning that is concerned with using intelligent agents (e.g., robots) that interact with their surrounding environment. These interactions generate rewards that influence the agents and steer their behavior till they take the optimum actions that maximize the cumulative rewards.

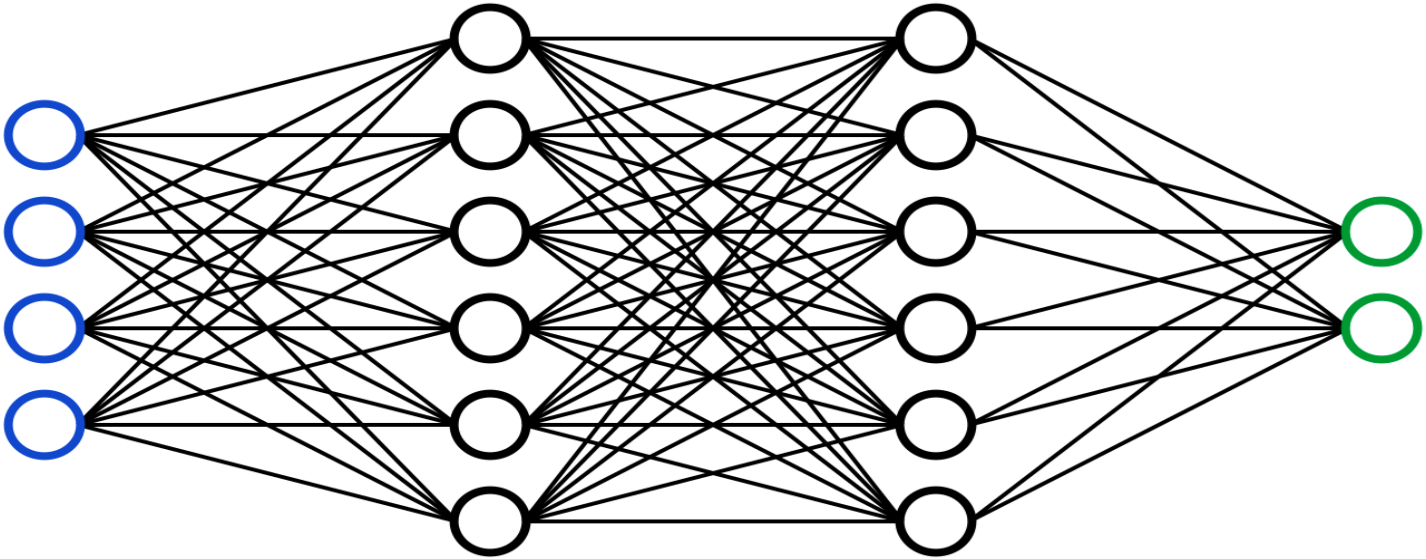
Q-Learning

$$Q(s_t, a_t) \leftarrow (1-\alpha) \cdot \underbrace{Q(s_t, a_t)}_{\text{old value}} + \underbrace{\alpha}_{\text{learning rate}} \cdot \overbrace{\left(\underbrace{r_t}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}} \right)}^{\text{learned value}}$$

Q-Learning is a off-policy learning algorithm that learns the value of an action in a particular state.

There is a finite state and action space to maintain a table lookup that maintains the current estimate of the Q-value. Howerwer, with fairly large or infinite state space, it becomes impossible to use a table.

Deep Q-Learning



Using deep Q-learning instead of deep supervised learning

Supervised deep learning

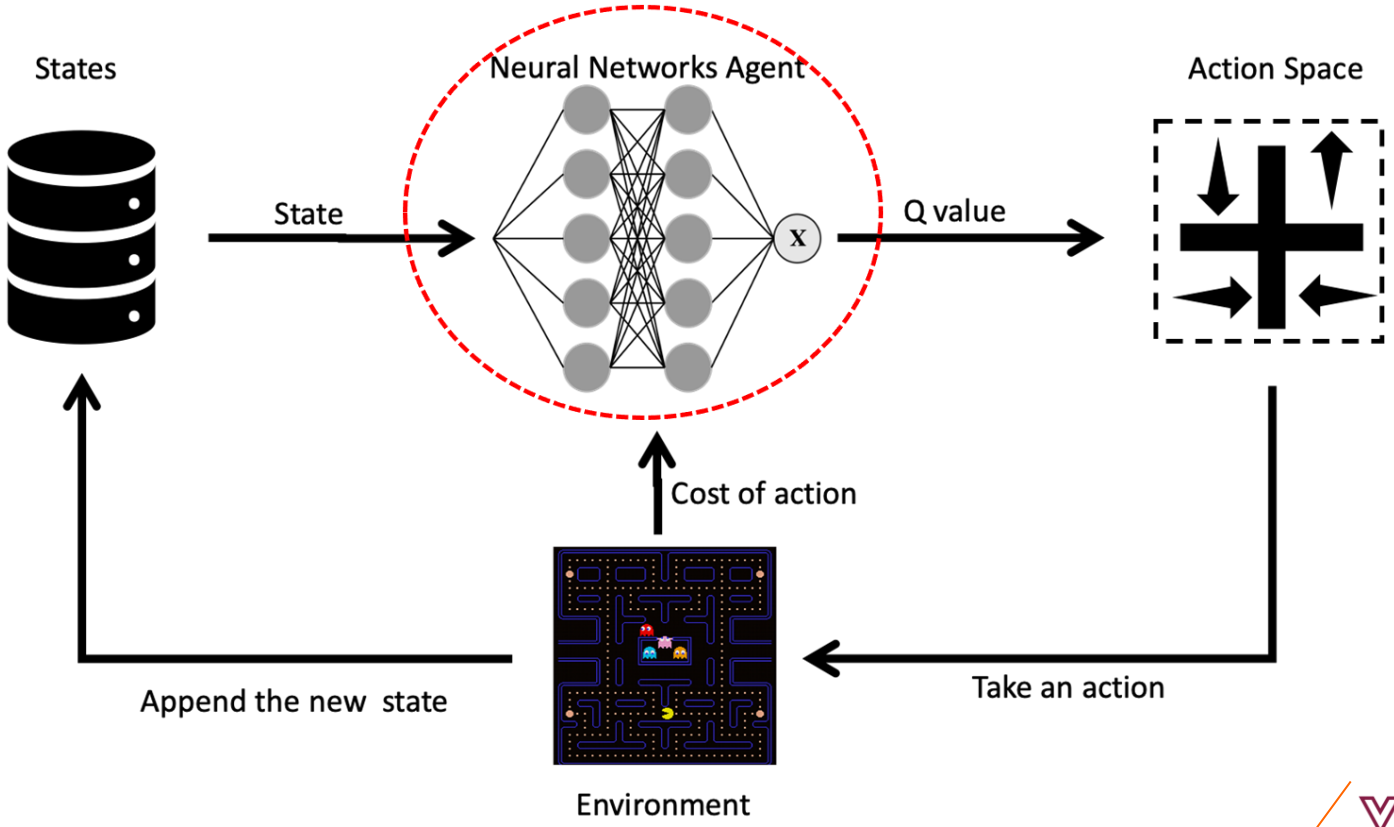
- Requires large amounts of labelled training data
- Assumes the data samples are independent
- Assumes a fixed underlying distribution
- Learns by exploitation

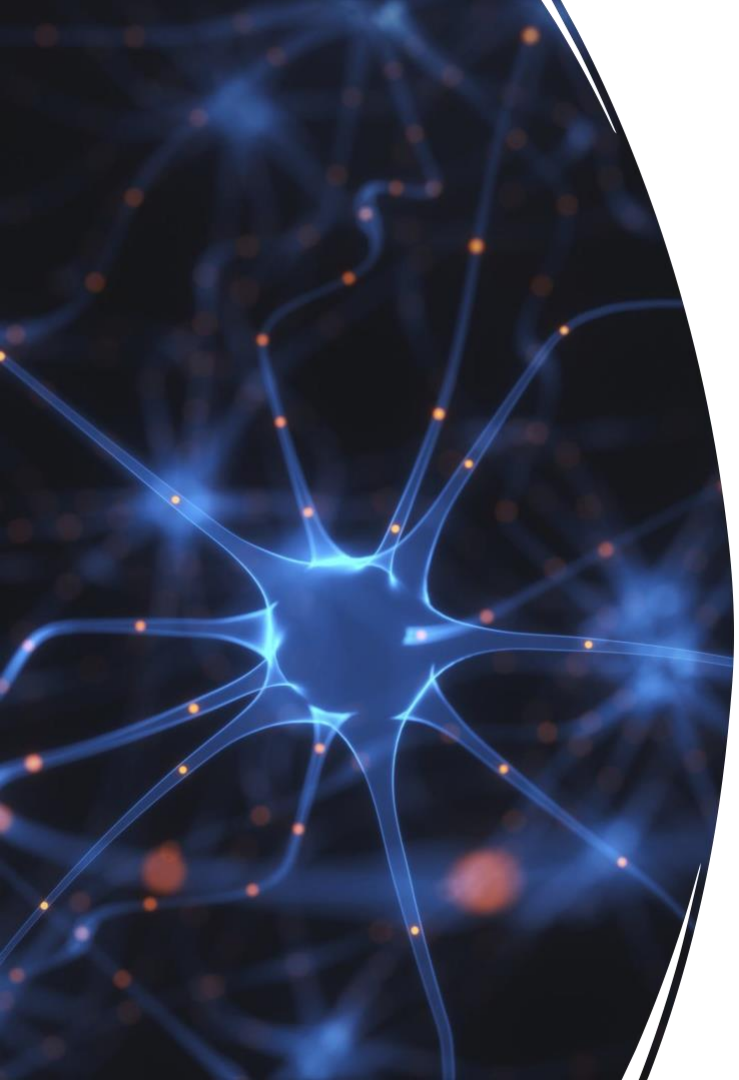
Reinforcement learning

- Learns from a scalar reward signal that is frequently sparse, noisy and delayed
- Encounters sequences of highly correlated state
- the data distribution changes as the algorithm learns new behaviors
- Slow learning

Deep Q-learning can overcome the challenges of the supervised deep learning and reinforcement learning to learn successful control policies from raw video data in complex RL environments

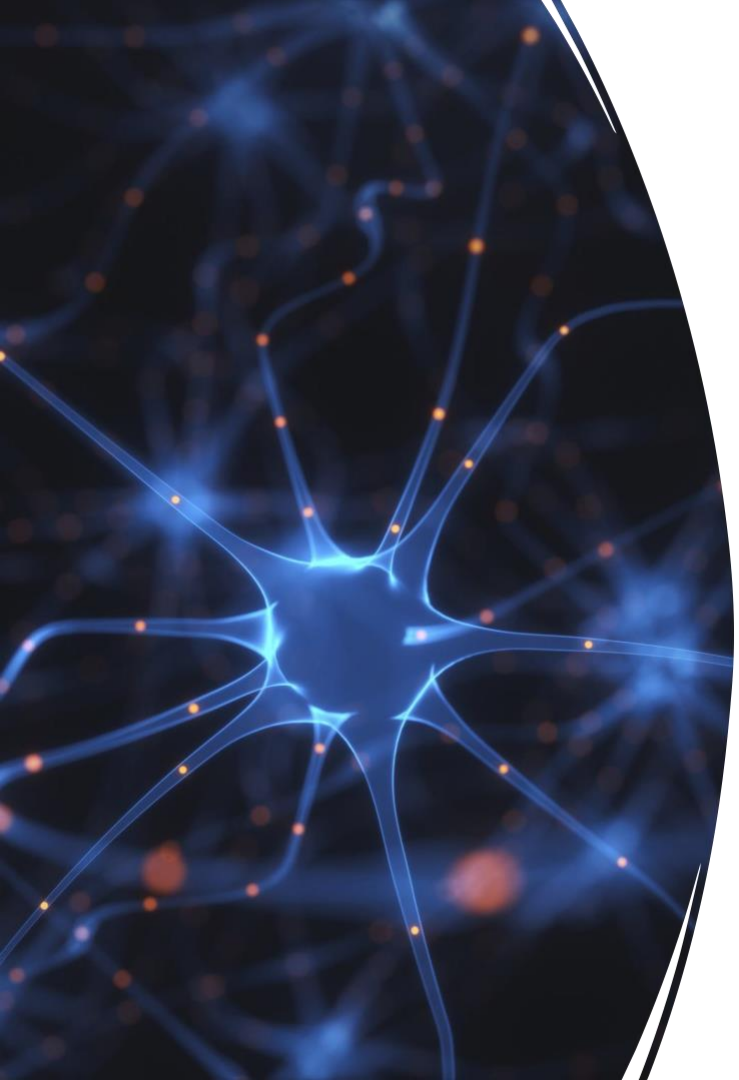
Applying it to PacMan problem





Neural Network Agent

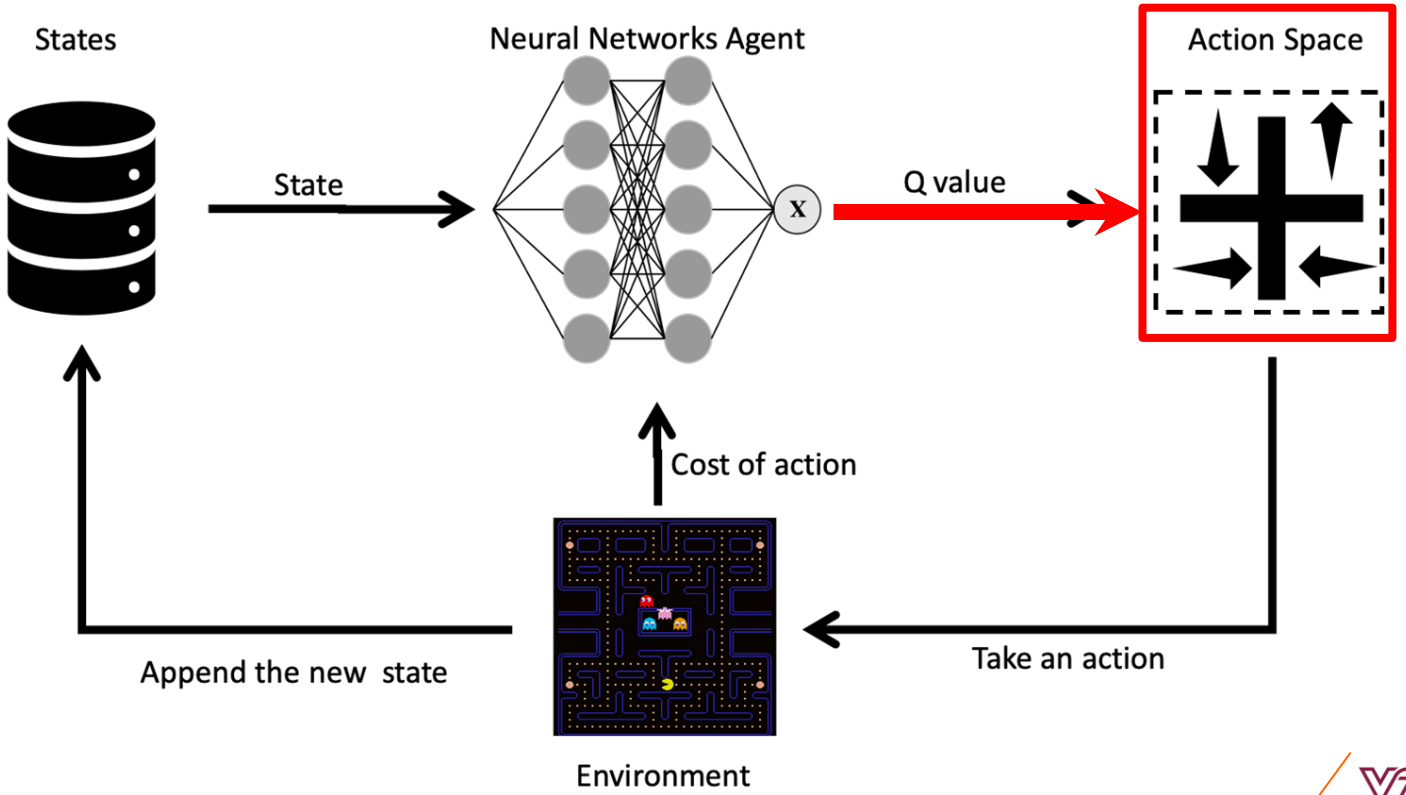
- Initialize the replay memory (training dataset)
- For each time step in the episode (episode = full game), select either a random action (explore), or the currently known, best action (exploit).
- Execute the chosen action and store the (processed) observed transition in the replay memory
- Experience replay: Sample a random minibatch of transitions from replay memory and perform gradient descent step on Q



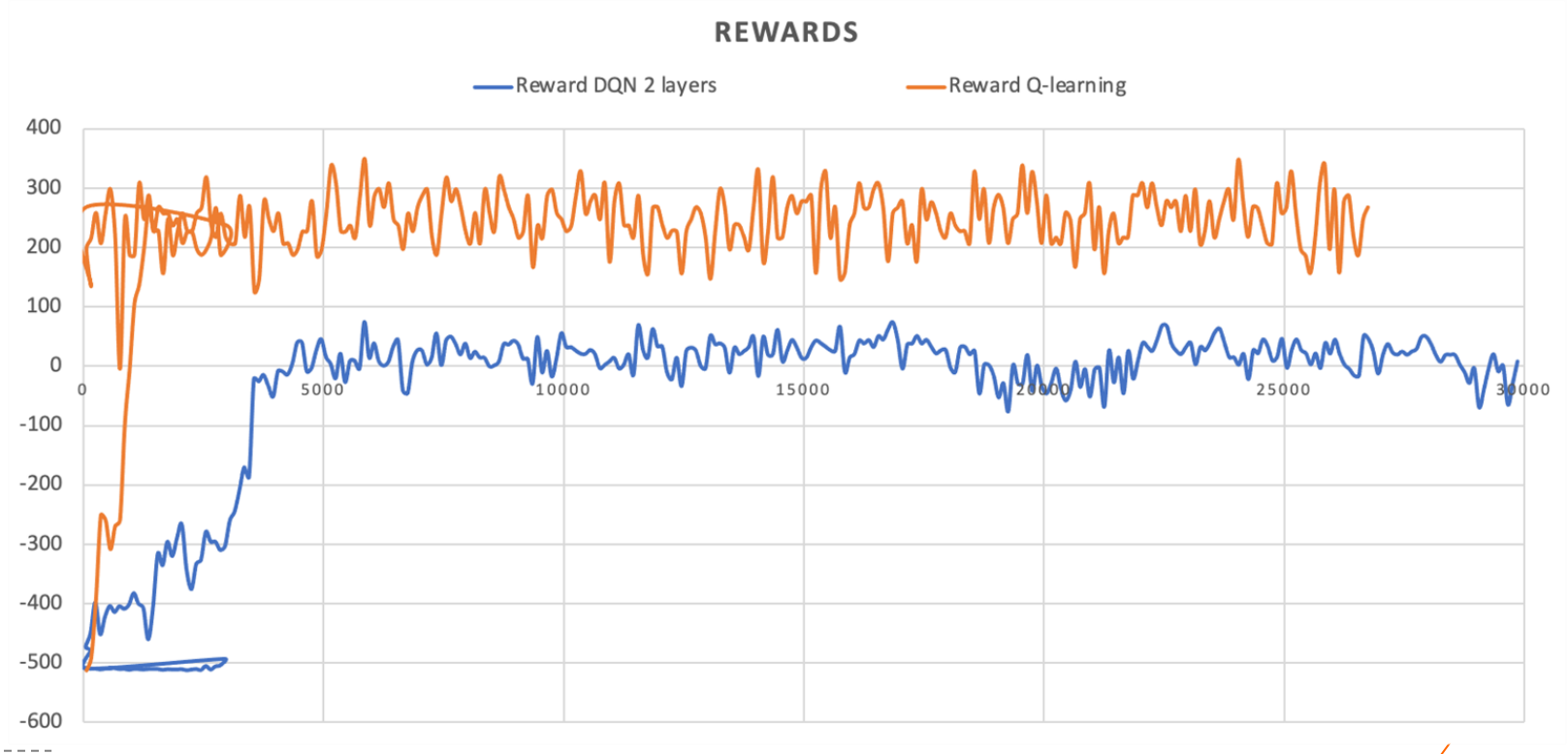
Neural Network Agent

- We have two identical Q approximators (DNN), Main Q and target $\hat{Q}$ model.
- Once every several steps set the target function, $\hat{Q}$ to equal Q (The target model have the same weights as the main model. It predicts with the target model every multiple steps (not every step). This gonna make the predictions more consistent and not all over the place.

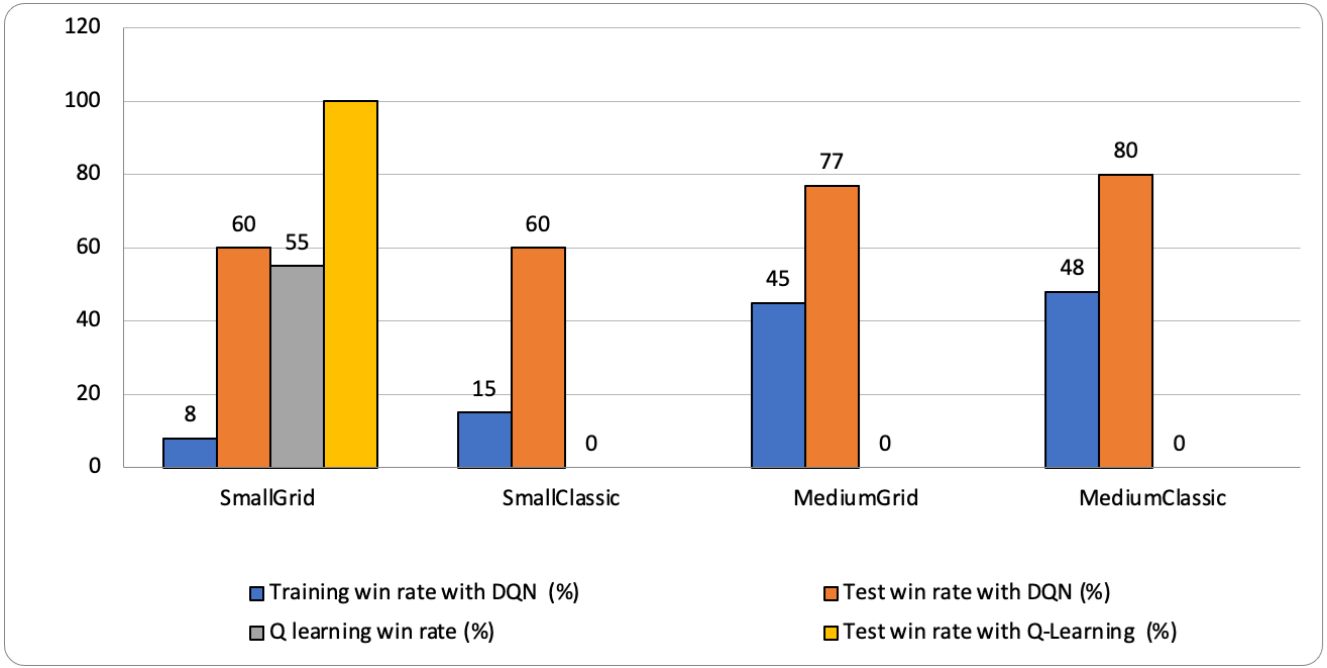
Applying it to PacMan problem



Results


REWARDS
Reward DQN 2 layers
Reward Q-learning
400
300
200
100
0
-100
-200
-300
-400
-500
-600
0
5000
10000
15000
20000
25000
30000

Results



Future Work

- Fine tune existing DQN networks for better results
- Compare results with other q-learning variants like Double Q-learning, Approximate Q-learning, Distributional Q-learning, Greedy GQ etc., to analyse the trade off between the benefits and disadvantages of the various alternatives

Questions ?

UNIVERSITY LIBRARIES
VIRGINIA TECH