

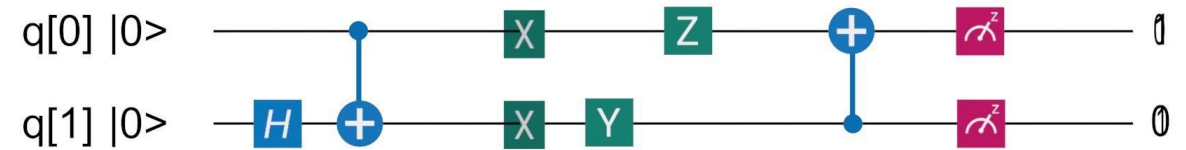
Quantum Deep Advantage

Coach: Asier Arranz
Eden Chen
Sam Tonetto
Makoto Nakai
Yinjie Zhou
Sitong Liu

#31

Motivation

- Automate building optimized circuits to prepare quantum states on real devices.
- We automated the quantum circuit-building process with reinforcement learning and validated our method by performing simple numerical experiments on a Bell State.



(Credit: Daniel Serrano, IBM)

Environment rewards the agent based on **gate-count** and **accuracy**.

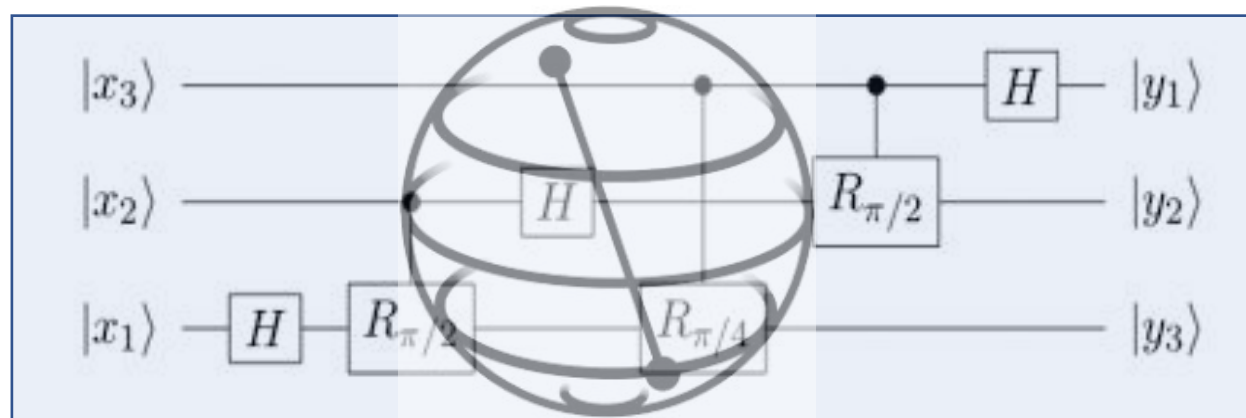
Agent



An agent places gates on a circuit every timestep.

Reward
/Punishment

Action



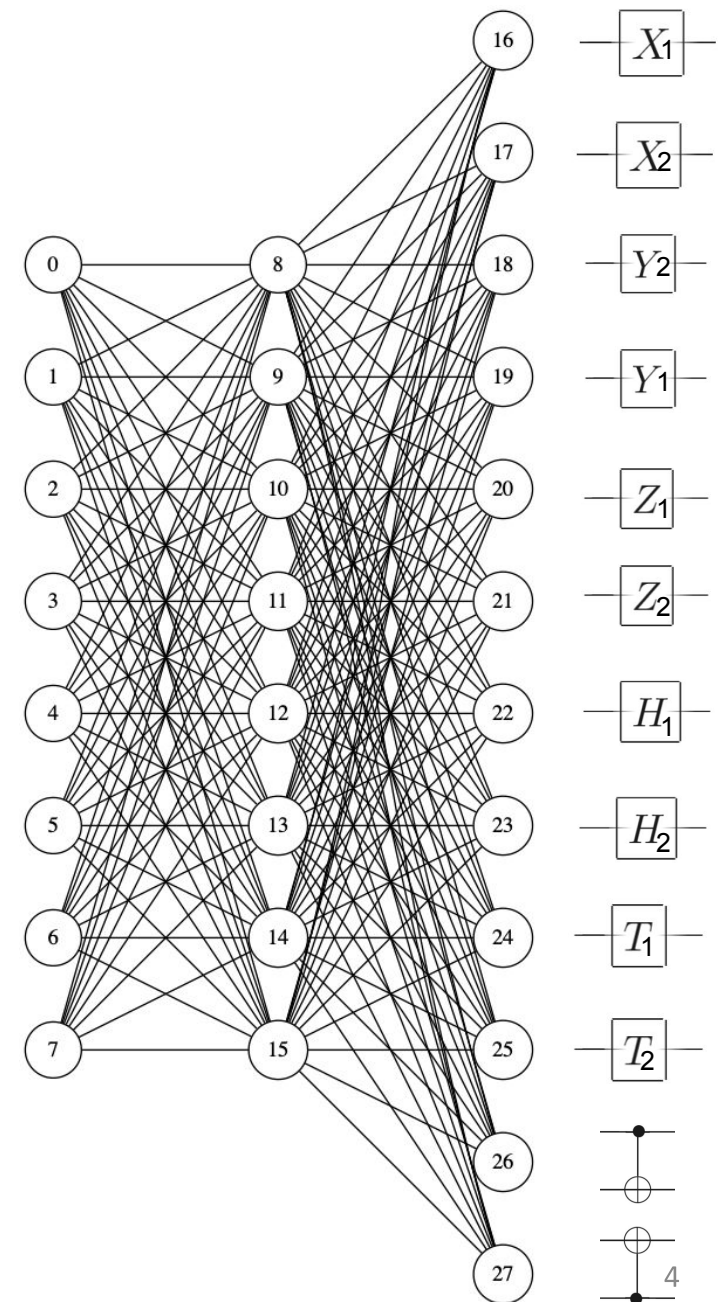
Environment: Qiskit Aer

Neural Network Implementation

- Tensorflow
 - Vanilla Neural Network
 - LSTM Network
 - **Could explore many more episodes with a GPU!**

To deal with the *Internal Covariate Shift* while training, we employ **Layer Normalization** (Ba, Kiros, and Hinton, 2016).

$|\psi_t\rangle$





Setting up Agent and Environment

State Space

All 2 qubits states

Action Space

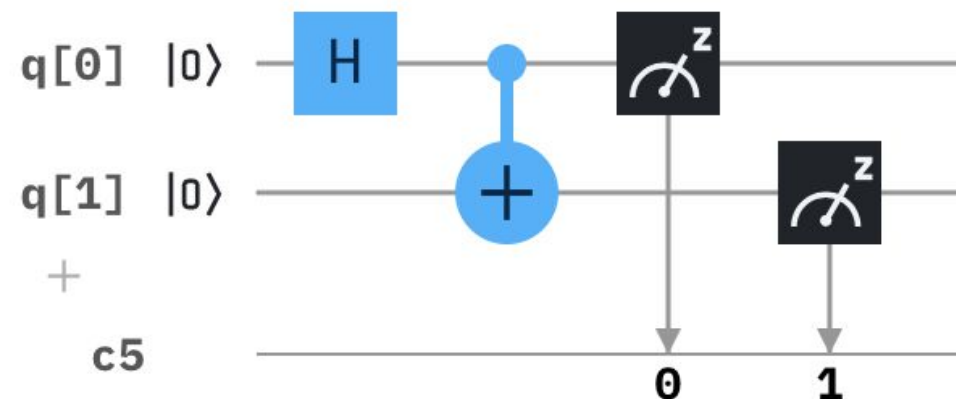
X_1 Y_1 Z_1 H_1 T_1  
 X_2 Y_2 Z_2 H_2 T_2

Rewards

$$\text{Reward} = \begin{cases} \frac{100}{\text{\#gates}} & \text{correct state} \quad \checkmark \\ -1 & \text{incorrect state} \quad \times \end{cases}$$

Target State: Bell State

$$|\psi_{\text{target}}\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}}$$



Qiskit as a simulator for quantum circuits

```
# apply step on qubit
qc = QuantumCircuit(2)
qc.initialize(self.state, [0, 1])
# apply X gate on qubit action[1]
if action[0] == "X":
    qc.x(action[1])
elif action[0] == "Y":
    qc.y(action[1])
```

...

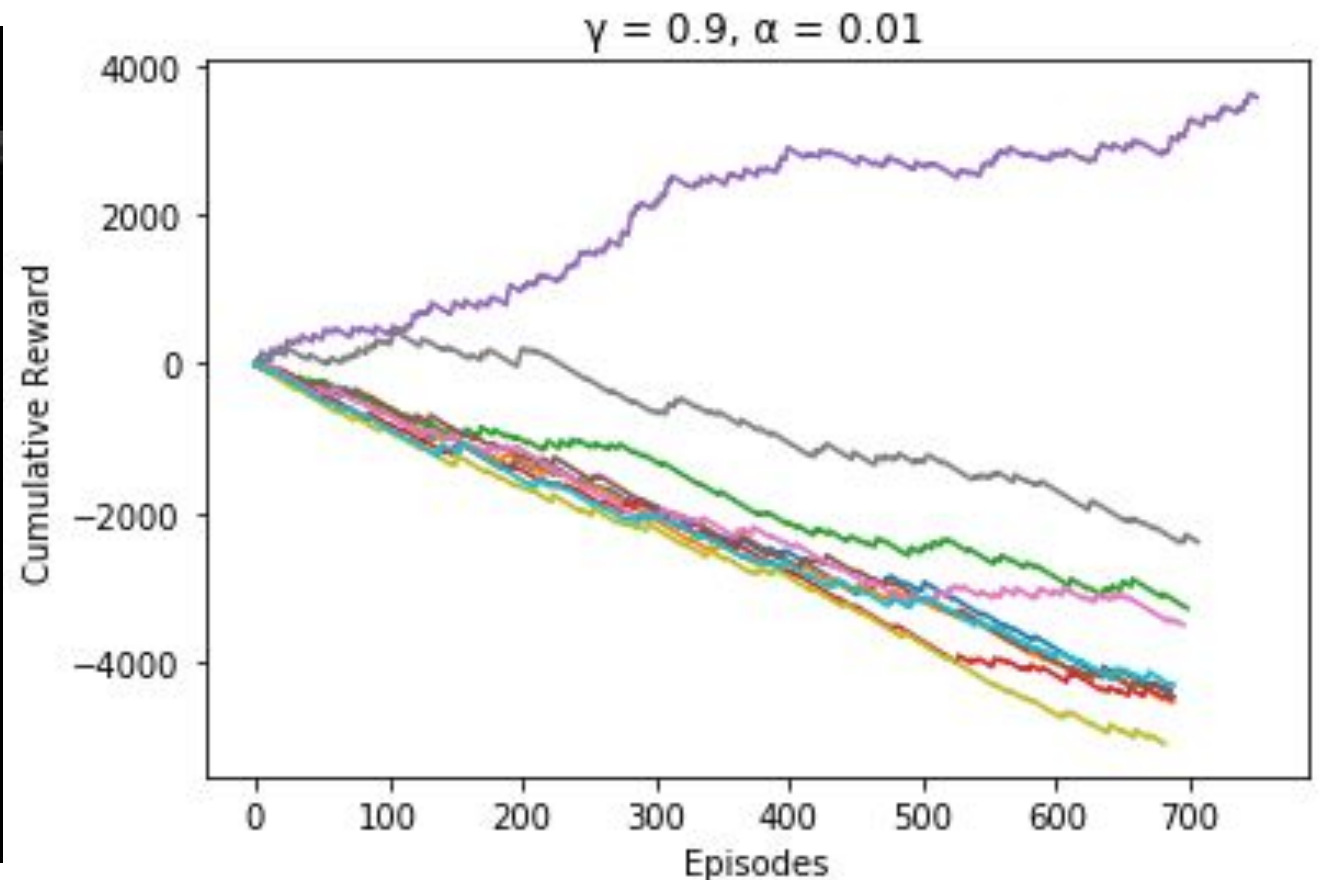


Qiskit Aer is a high-performance simulator framework for quantum computing algorithms. It allows us to see the full state-vector after applying each gate without destroy the entanglement.

Bell State Attempt 1: 4x4x12 – Vanilla NN

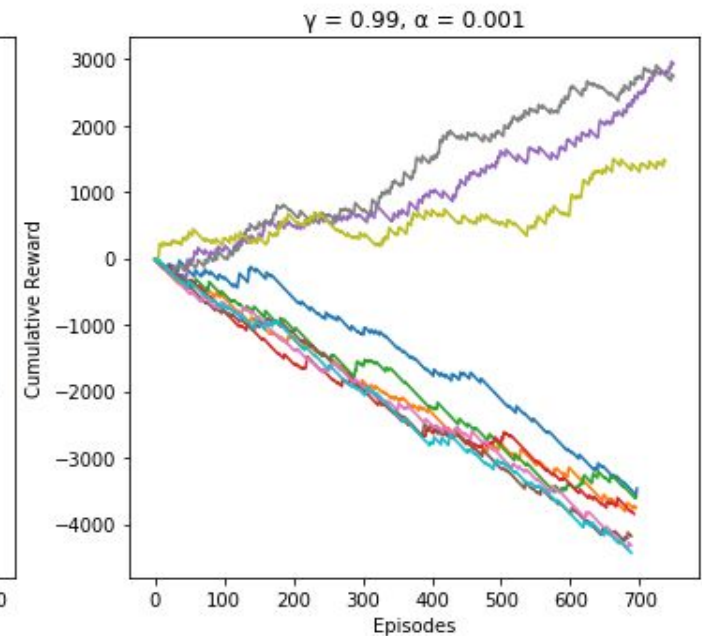
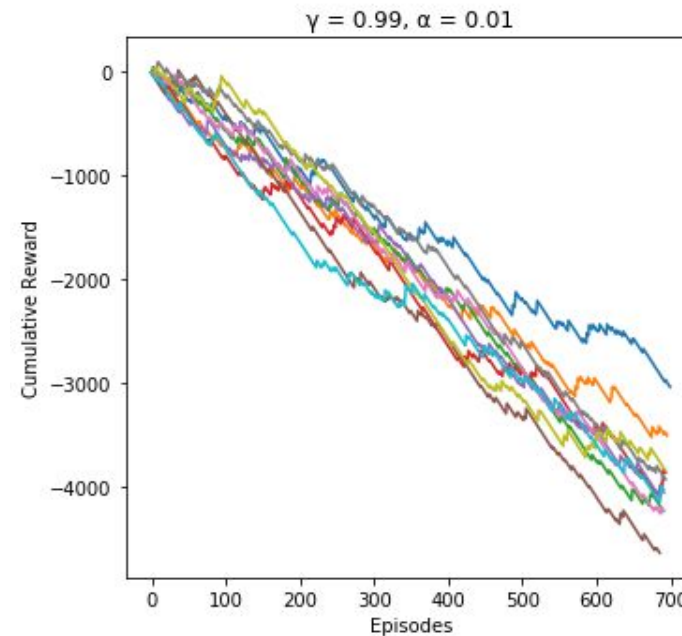
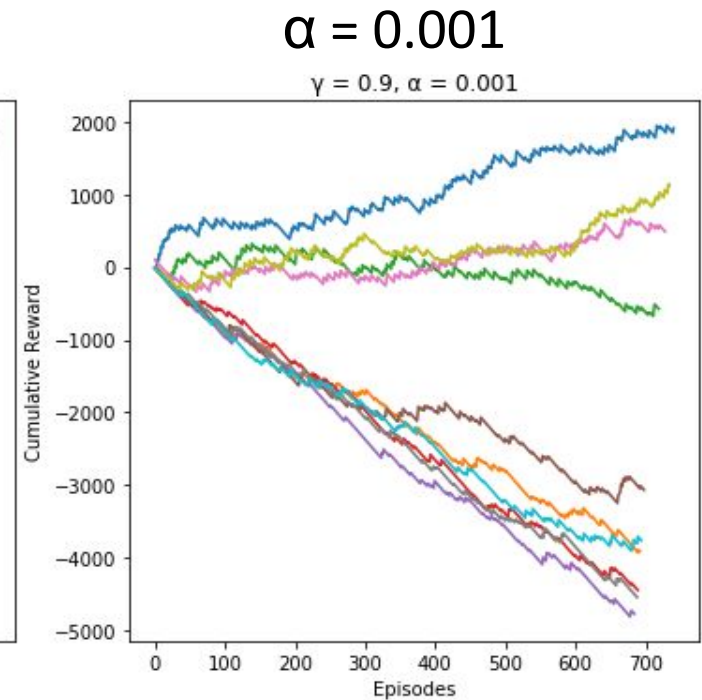
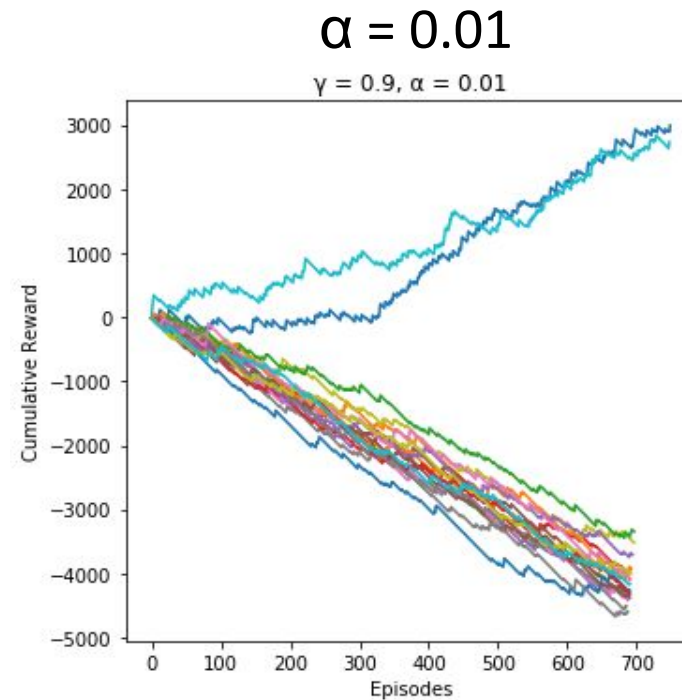
- Plot of Cumulative Reward (moving average of 100 episodes):

```
end of step 9, action is ['T', 0], inner_product is 0.4999999999999998
res is 3
end of step 8, action is ['T', 0], inner_product is 0.4999999999999999
end of step 1, action is ['X', 0], inner_product is 0.0
end of step 2, action is ['T', 0], inner_product is 0.0
end of step 3, action is ['H', 0], inner_product is 0.24999999999999983
end of step 4, action is ['H', 1], inner_product is 3.8814879110195774e-33
end of step 5, action is ['T', 0], inner_product is 0.87322338478336389
end of step 6, action is ['X', 1], inner_product is 0.87322338478336389
end of step 7, action is ['H', 1], inner_product is 0.24999999999999978
end of step 8, action is ['Z', 0], inner_product is 0.24999999999999978
end of step 9, action is ['V', 0], inner_product is 0.24999999999999999
res is 4
end of step 8, action is ['H', 0], inner_product is 0.25
end of step 1, action is ['V', 0], inner_product is 0.24999999999999999
end of step 2, action is ['V', 0], inner_product is 0.25
end of step 3, action is ['T', 1], inner_product is 0.25
end of step 4, action is ['H', 1], inner_product is 0.49999999999999999
end of step 5, action is ['T', 0], inner_product is 0.42677669529663675
end of step 6, action is ['Z', 0], inner_product is 0.87322338478336389
end of step 7, action is ['CX', [1, 0]], inner_product is 0.49999999999999999
end of step 8, action is ['Z', 0], inner_product is 3.8814879110195774e-33
end of step 9, action is ['T', 1], inner_product is 0.87322338478336389
res is 5
end of step 8, action is ['X', 0], inner_product is 0.0
end of step 1, action is ['X', 1], inner_product is 0.49999999999999999
end of step 2, action is ['V', 0], inner_product is 0.0
end of step 3, action is ['X', 1], inner_product is 0.49999999999999999
end of step 4, action is ['H', 1], inner_product is 0.25
end of step 5, action is ['Z', 1], inner_product is 0.25
end of step 6, action is ['Z', 1], inner_product is 0.25
```



Bell State Attempt 2: 8x8x12 Vanilla NN with 0.2 Dropout

- Very sensitive to initial conditions
- Lower learning rate of $\alpha = 0.001$ appears to have better success.



Bell State Attempt 3: LSTM 8x8x12 Example

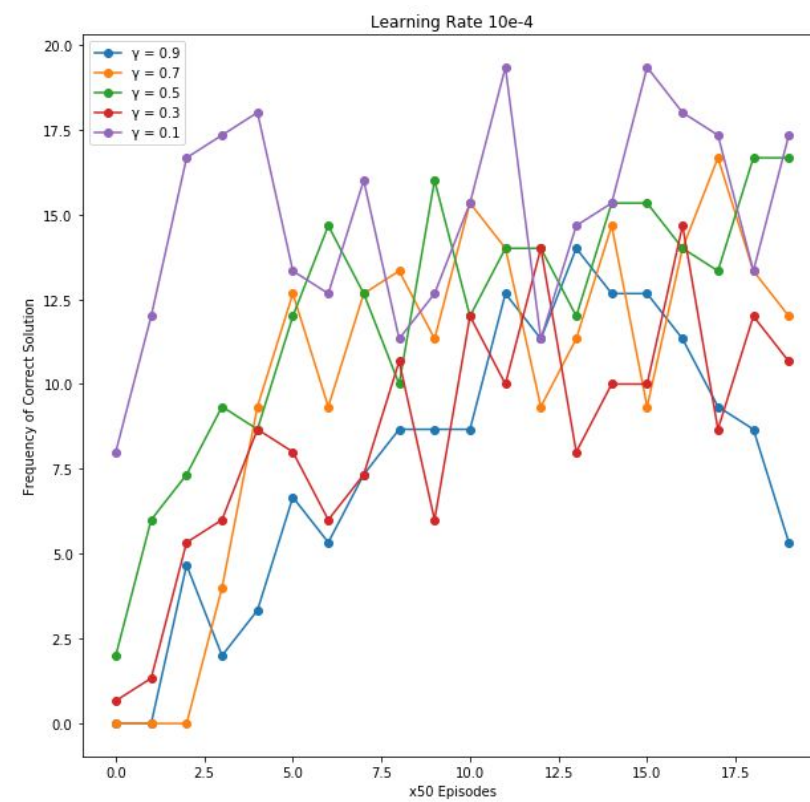
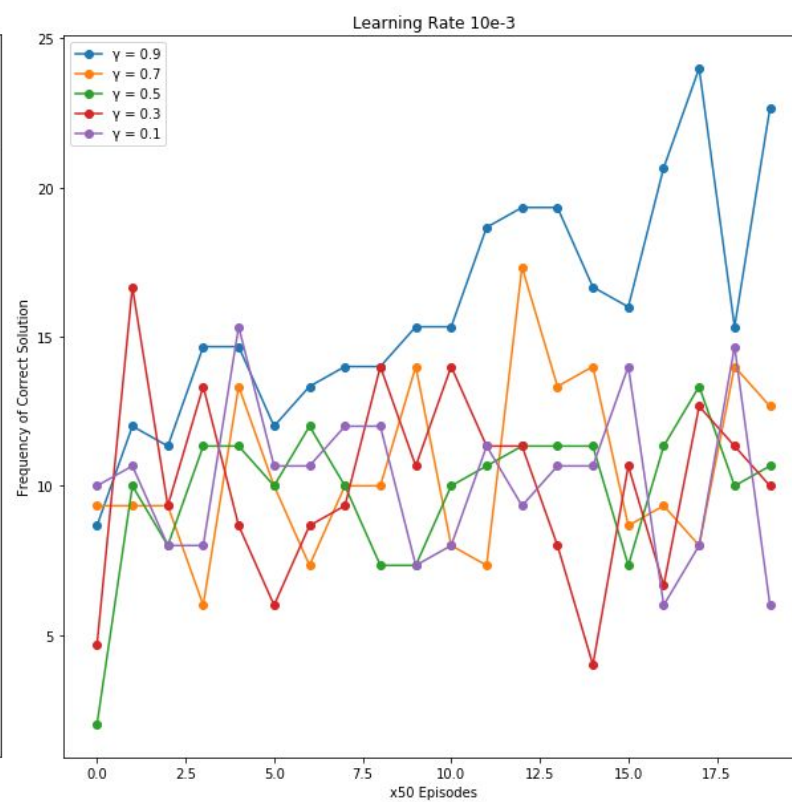
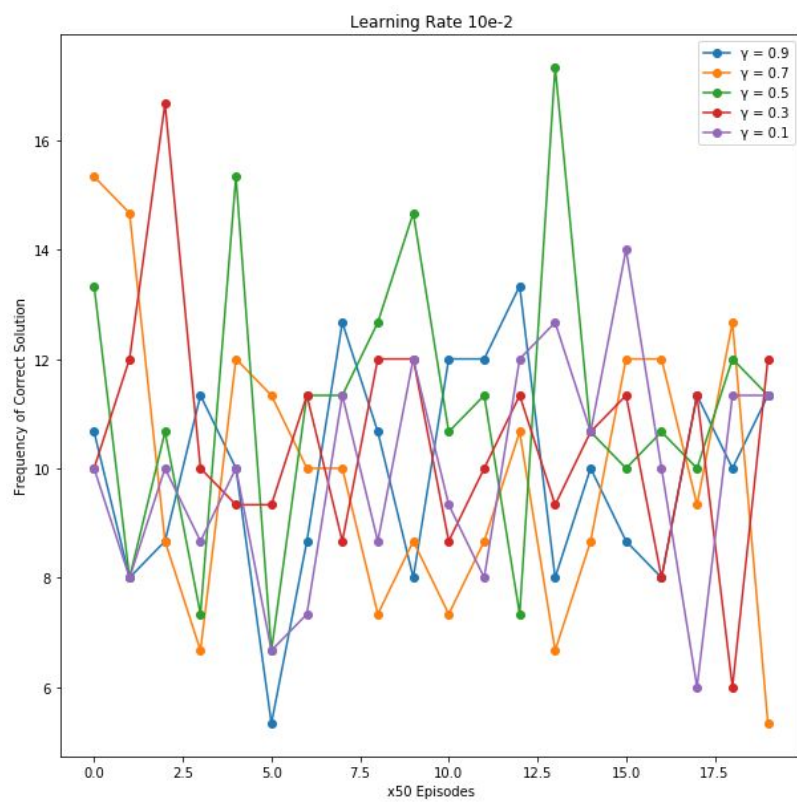
10e-2



Learning Rate Decreases



10e-4



Future Work

Applications

- Larger, more complicated circuits
- Application to Solovay-Kitaev on n -qubits.
- Compare with the Initialize function in Qiskit
- Benchmark and integrate into Qiskit

Improvements to RL

- Implementing other RL methods, such as Actor-Critic/ Soft Q-Learning
- Better Hyperparameter Tuning through Meta-Learning (iterate and improve!)
- Sparse Reward Signal:
 - Reward Shaping
 - Scoring
- Implement ***Experience Replay*** and ***Target Network***.

Thank you for
listening!!

