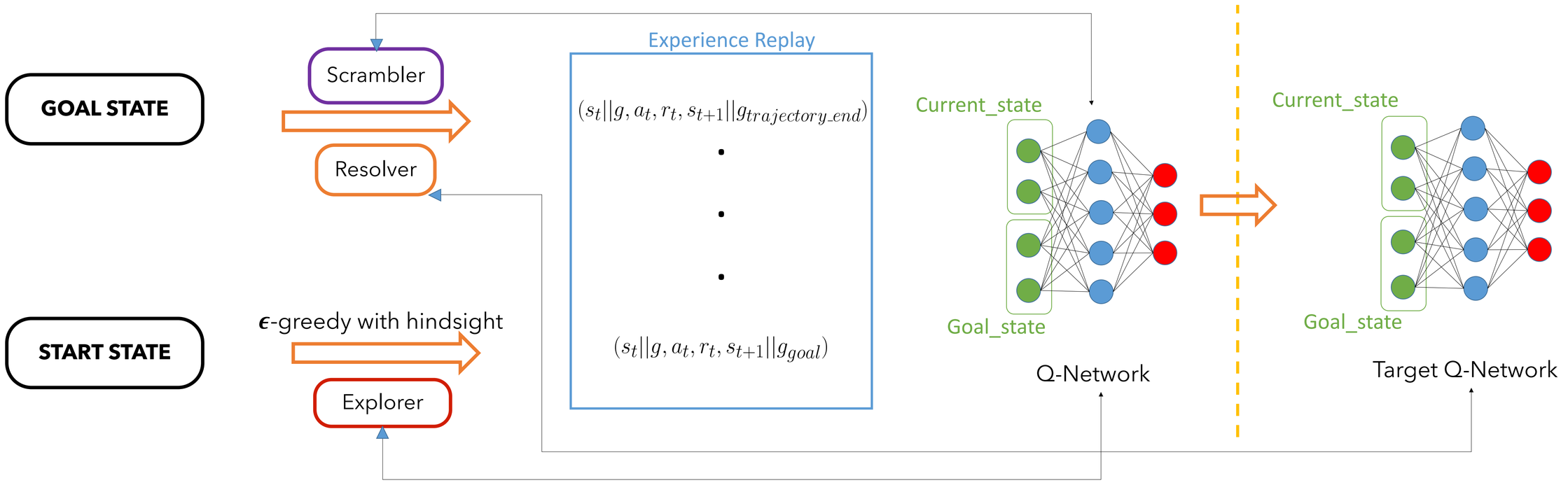
Scrambler-Resolver-Explorer:
A Hindsight based Backward & Forward Exploration Strategy for State Space Search Problems

Vihaan Akshaay
vihaanakshaay@ucsb.edu

Github Repository

Overall Learning Flowchart



Agent Architecture

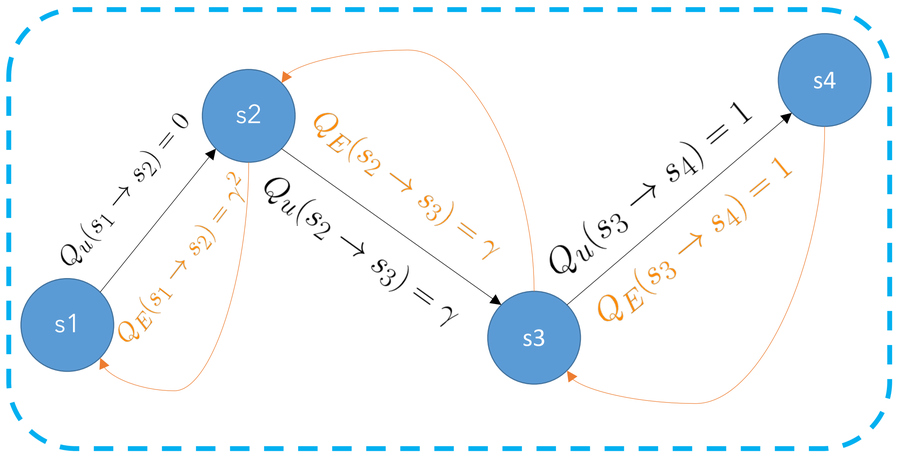
Existing Exploration Algorithms

➢ ϵ-Greedy Exploration

While agent collects experiences from the environment (during the learning phase) it chooses a mix of random and greedy actions to ensure good distribution.

$$AgentAction = \begin{cases} \arg_a \max Q_t(state), & \text{with probability } 1 - \epsilon \\ \text{Random Action}, & \text{with probability } \epsilon \end{cases}$$

➢ Backward Episodic Update (BEU)



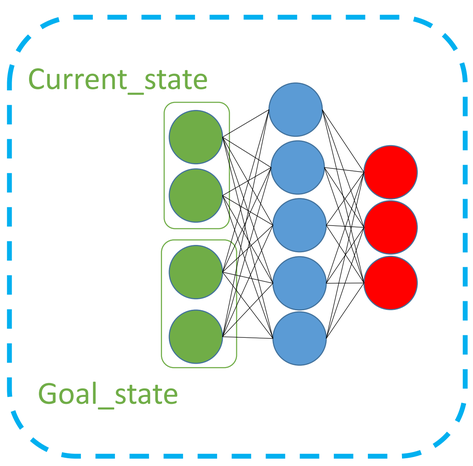
Updates are done by picking a trajectory and using transitions in reverse order.

This way sparse reward systems get state values passed on successively with better sample efficiency.

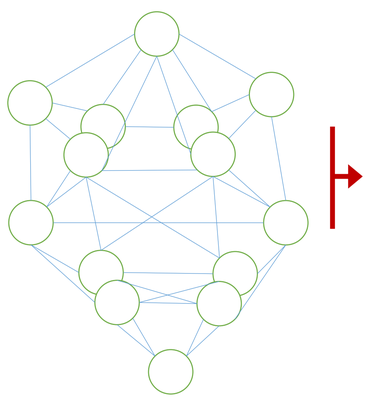
➢ Hindsight Experience Replay (HER)

To make use of every exploration trajectory, the policy NN is modified to take both the 'state' and the 'goal' as inputs and trained to predict the optimal action.

Trajectories that don't reach the goal, we take the final state as a 'pseudo goal' and train the function with these augmented data points.



Rubik's Cube: State Sphere & Trajectories



Rubik's cube configurations can be modeled as a graph with,

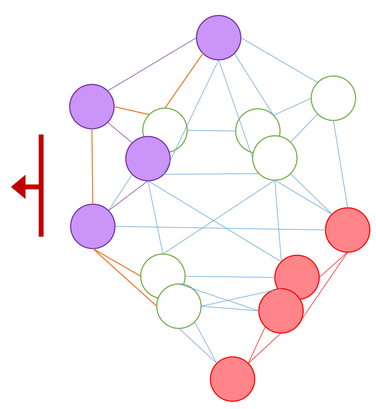
- Each node as a configuration (4.3 * 10^19 nodes)
- Each edge is an action on the cube. (18 actions for every state)
- The upper bound on the shortest path between any two nodes is 20.
- The extreme opposite states in the scramble state space is called the super-flip which is guaranteed to take 20 moves to reach the goal state.

The proposed **Scrambler-Resolver-Explorer** Algorithm has the following components:

- SCRAMBLER:
Starts the environment from the goal state. Generates trajectories in reverse that are hard for the resolver to guess.

- RESOLVER:
From every scrambled state the agent tries to learn how to resolve back to the goal state. It uses the hindsight formulation to learn and match the explorer configuration.

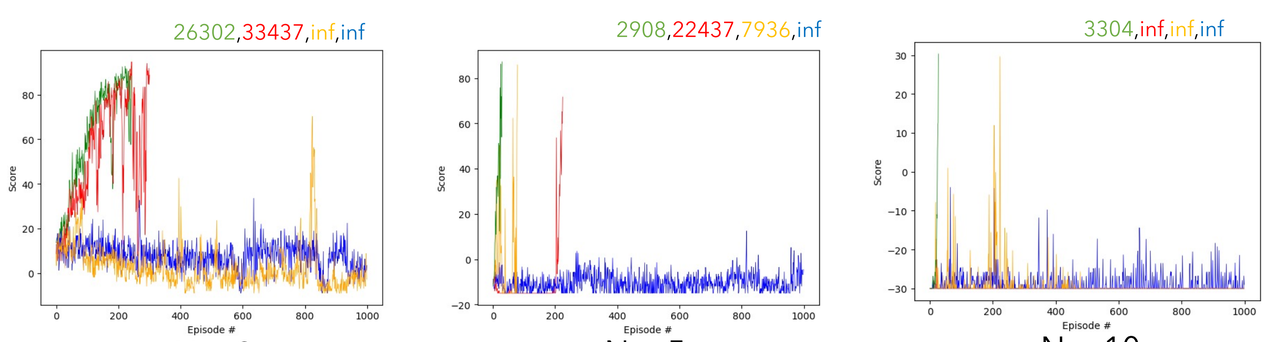
- EXPLORER:
This follows a traditional exploration strategy. It starts from the super-flip state, and tries to learn through hindsight experience replay to reach the goal state.



Experiments

A custom (nxn) grid environment was constructed and these 4 competing agents were made to learn the shortest path from start to goal state on opposite ends of the grid.

The agent receives a reward of +100 for reaching the goal and -1 for every step.



Conclusion & Future Work

- For smaller grids it competes with other standard explorations and beats them with a small margin, and as the state space keeps getting bigger, this shines as expected.
- For state search problems this exploration encorporates strengths from various other algorithms.
- Construct Graphs from Obtained Trajectories and run loop detection algorithm to improve samples.

Significant References

[1] S. Li, etc. Sample Efficient Deep Reinforcement Learning via Episodic Backward Update

[2] M. Andrychowicz, etc. Hindsight Experience Replay

[3] Z. Hong, etc. Topological Experience Replay