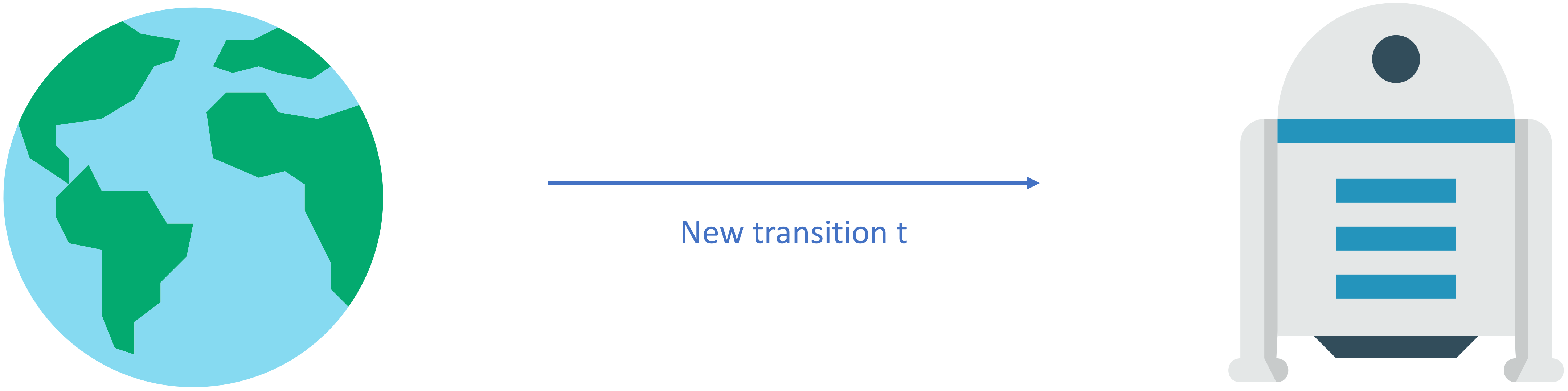
A Deeper Look at Experience Replay (17.12)

Seungjae Ryan Lee

endtoend.ai

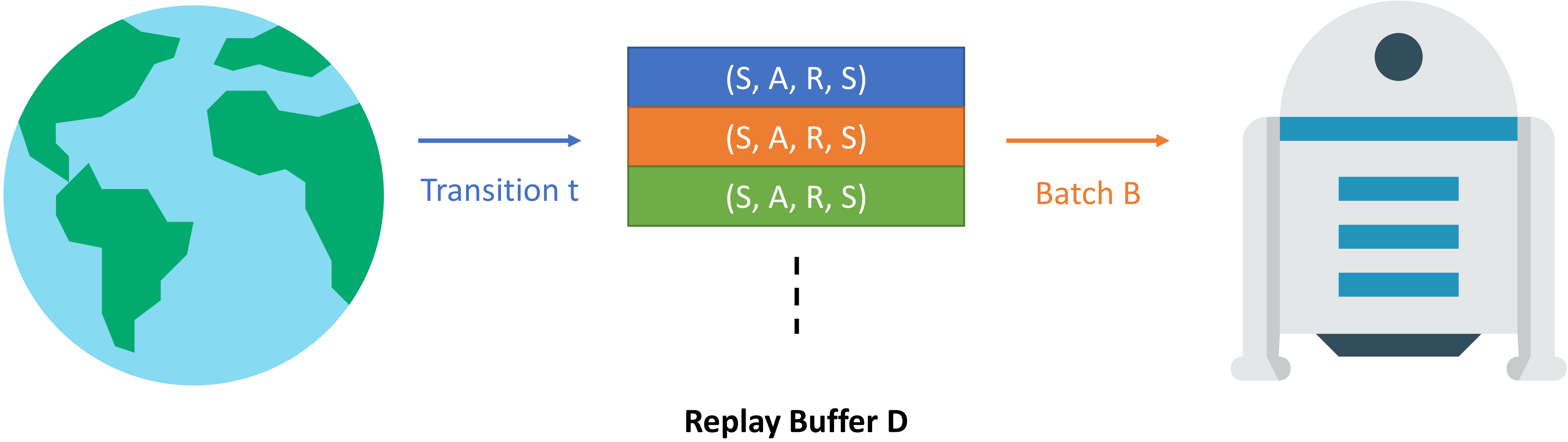
Online Learning

- Learn directly from experience
- Highly correlated data



Experience Replay

- Save transitions $(S_t, A_t, R_{t+1}, S_{t+1})$ into buffer and sample batch B
- Use batch B to train the agent



endtoend.ai

Effectiveness of Experience Replay

- Only method that can generate *uncorrelated* data for online RL
  - Except using multiple workers (A3C)
- Significantly improves data efficiency
- Norm in many deep RL algorithms
  - Deep Q-Networks (DQN)
  - Deep Deterministic Policy Gradient (DDPG)
  - Hindsight Experience Replay (HER)

Problem with Experience Replay

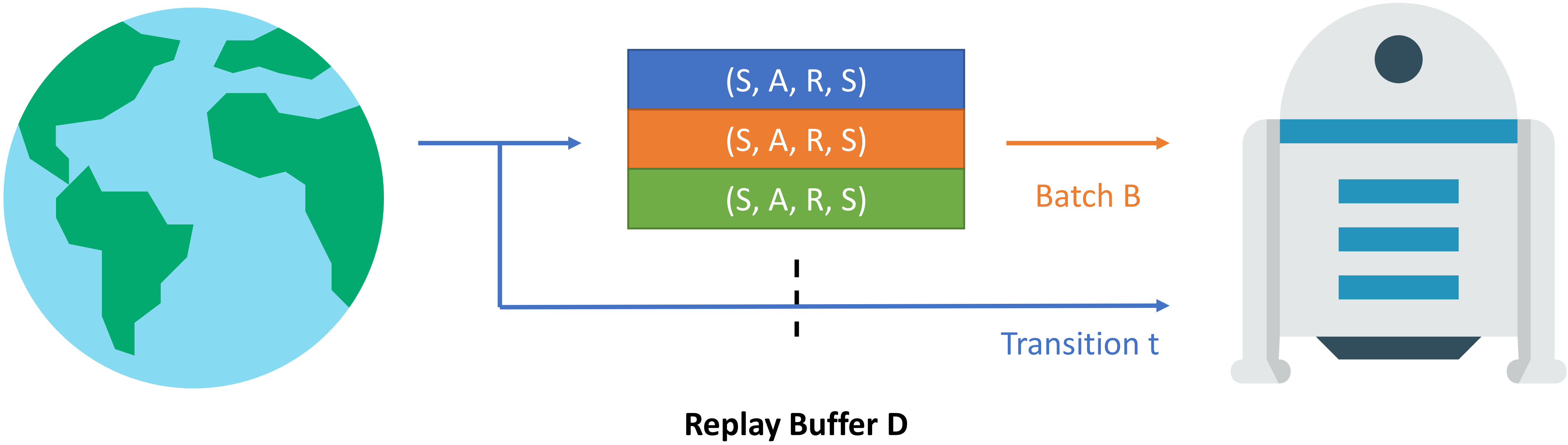
- There has been *default capacity* of 10^6 used for:
  - Different algorithms (DQN, PG, etc.)
  - Different environments (retro games, continuous control, etc.)
  - Different neural network architectures

Result 1

Replay buffer capacity can have significant negative impact on performance if too low or too high.

Combined Experience Replay (CER)

- Save transitions $(S_t, A_t, R_{t+1}, S_{t+1})$ into buffer and sample batch B
- Use batch B to and online transition t to train the agent



endtoend.ai

Combined Experience Replay (CER)

Result 2

CER can remedy the negative influence of a large replay buffer with $O(1)$ computation.

CER vs. Prioritized Experience Replay (PER)

- Prioritized Experience Replay (PER)
  - **Stochastic** replay method
  - Designed to replay the buffer more efficiently
  - Always expected to improve performance
  - $O(N \log N)$
- Combined Experience Replay (CER)
  - Guaranteed to use newest transition
  - Designed to remedy negative influence of a large replay buffer
  - Does not improve performance for good replay buffer sizes
  - $O(1)$

Test agents

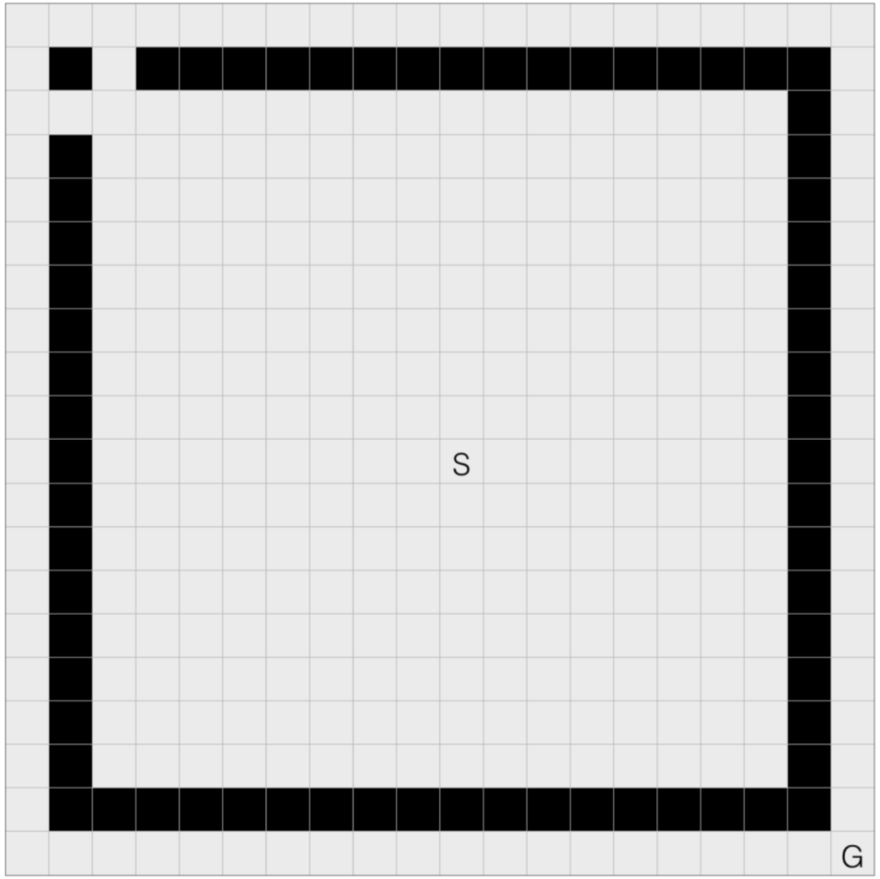
1. Online-Q
   - Q-learning with online transitions t
2. Buffer-Q
   - Q-learning with the replay buffer B
3. Combined-Q
   - Q-learning with both the replay buffer B and online transitions t

Testbed Environments

- 3 environments for 3 methods
  - Tabular, Linear and Nonlinear approximations
- Introduce "timeout" to all tasks
  - Episode ends automatically after T timesteps (large enough for each task)
  - Prevent episode being arbitrarily long
  - Used partial-episode-bootstrap (PEB) to minimize negative side-effects

Testbed: Gridworld

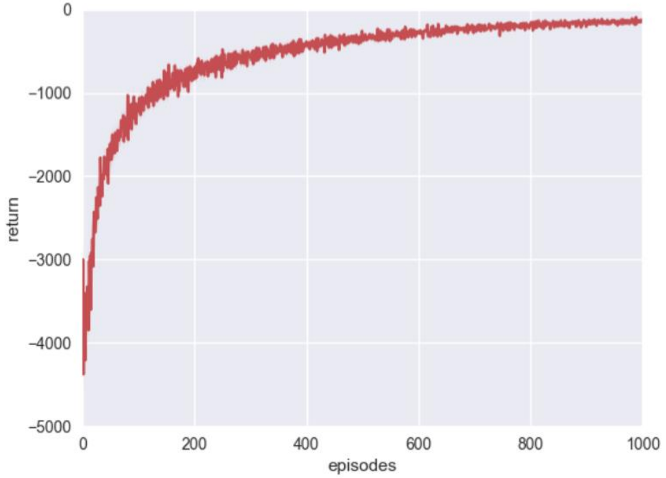
- Agent starts in S and has a goal state G
- Agent can move left, right, up, down
- Reward is -1 until goal is reached
- If the agent bumps into the wall (black), it remains in the same position



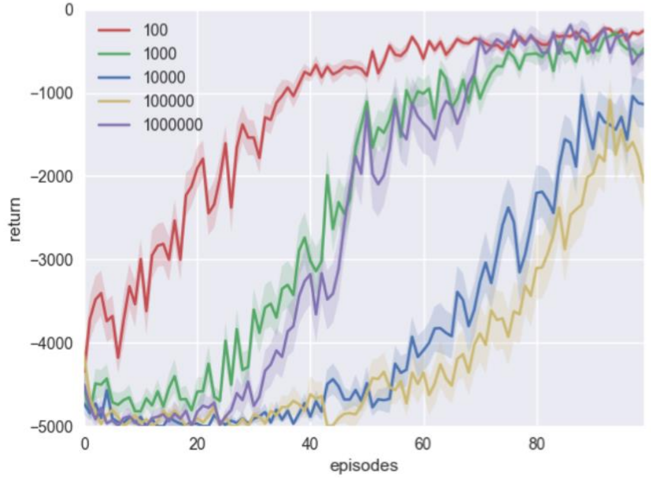
endtoend.ai

Gridworld Results (Tabular)

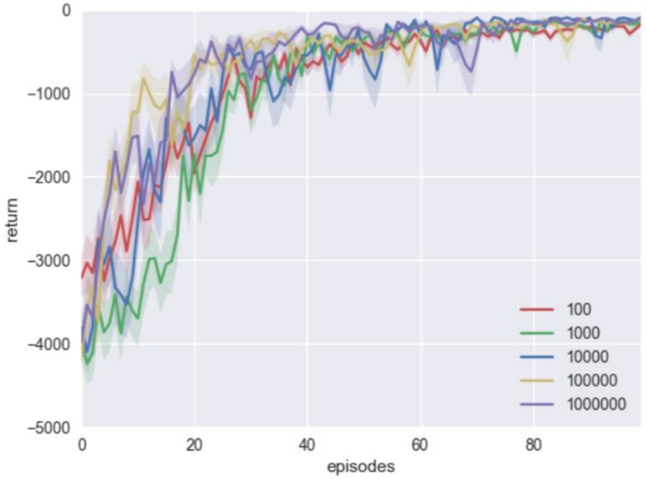
- Online-Q solves task very slowly
- Buffer-Q shows worse performance / speed for larger buffers
- Combined-Q shows slightly faster speed for larger buffers



(a) Online-Q



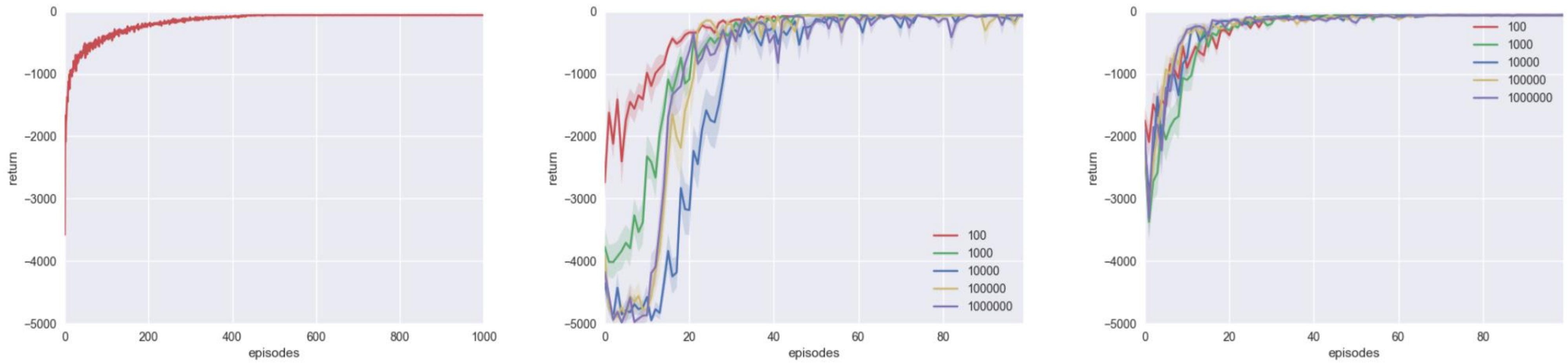
(b) Buffer-Q



(c) Combined-Q

Gridworld Results (Linear)

- Buffer-Q shows worse learning speed for larger buffers
- Combined-Q is robust for varying buffer size

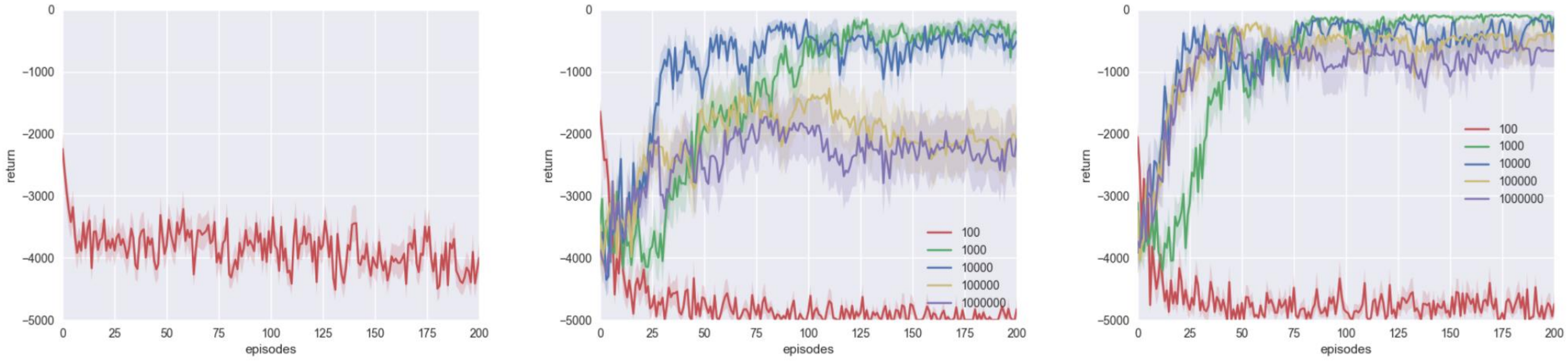


(a) Online-Q (b) Buffer-Q (c) Combined-Q

endtoend.ai

Gridworld Results (Nonlinear)

- Online-Q fails to learn
- Combined-Q significantly speeds up learning

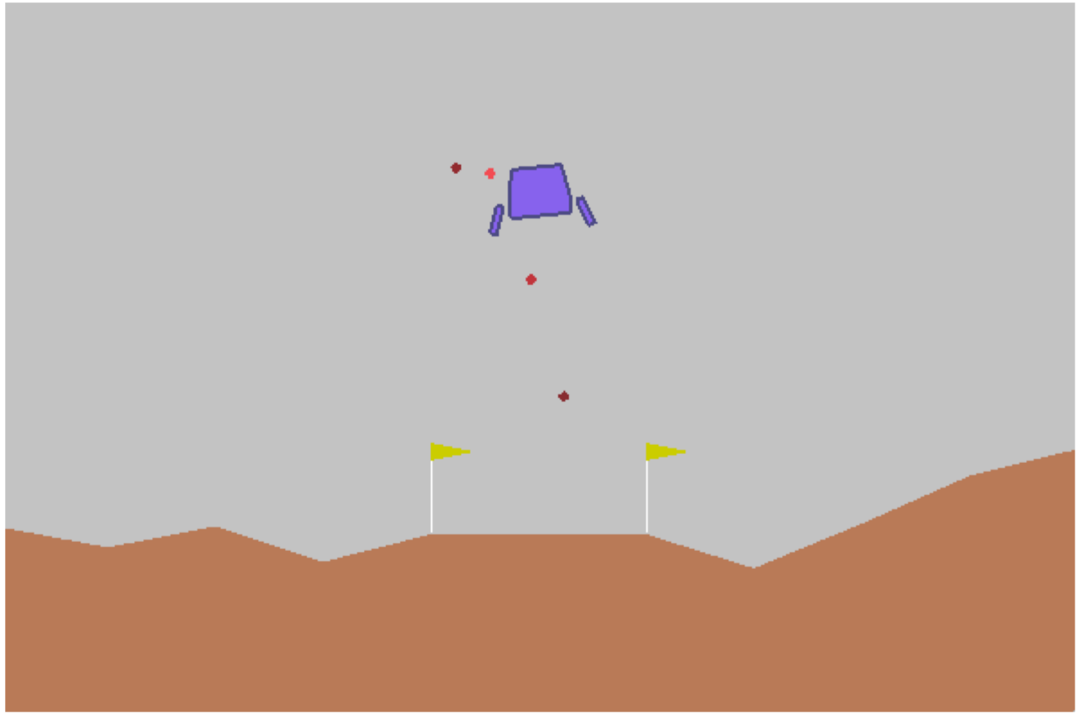


(a) Online-Q (b) Buffer-Q (c) Combined-Q

endtoend.ai

Testbed: Lunar Lander

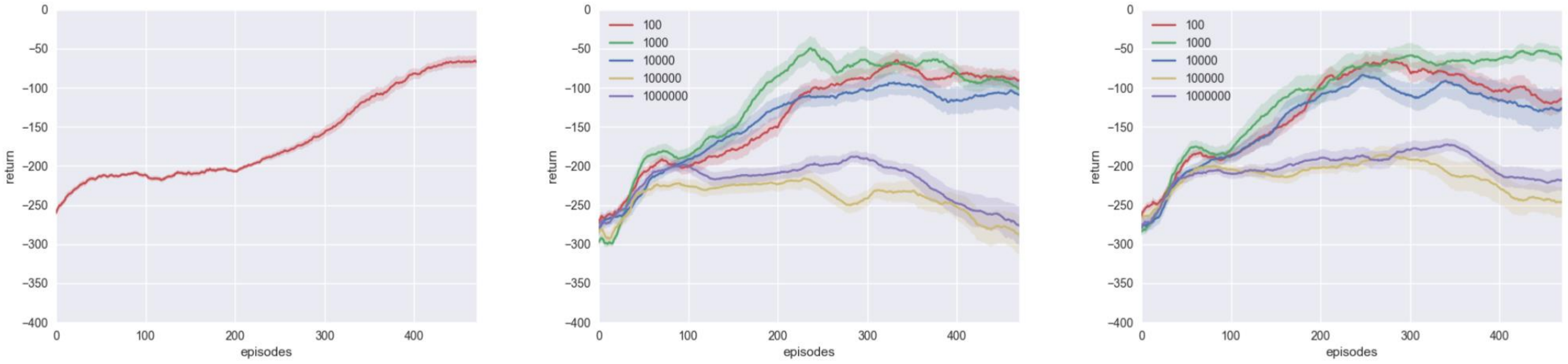
- Agent tries to land a shuttle on the moon
- State space: R^8
- 4 discrete actions



endtoend.ai

Lunar Lander Results (Nonlinear)

- Online-Q achieves best performance
- Combined-Q shows marginal improvement to Buffer-Q
- Buffer-Q and Combined-Q overfits after some time

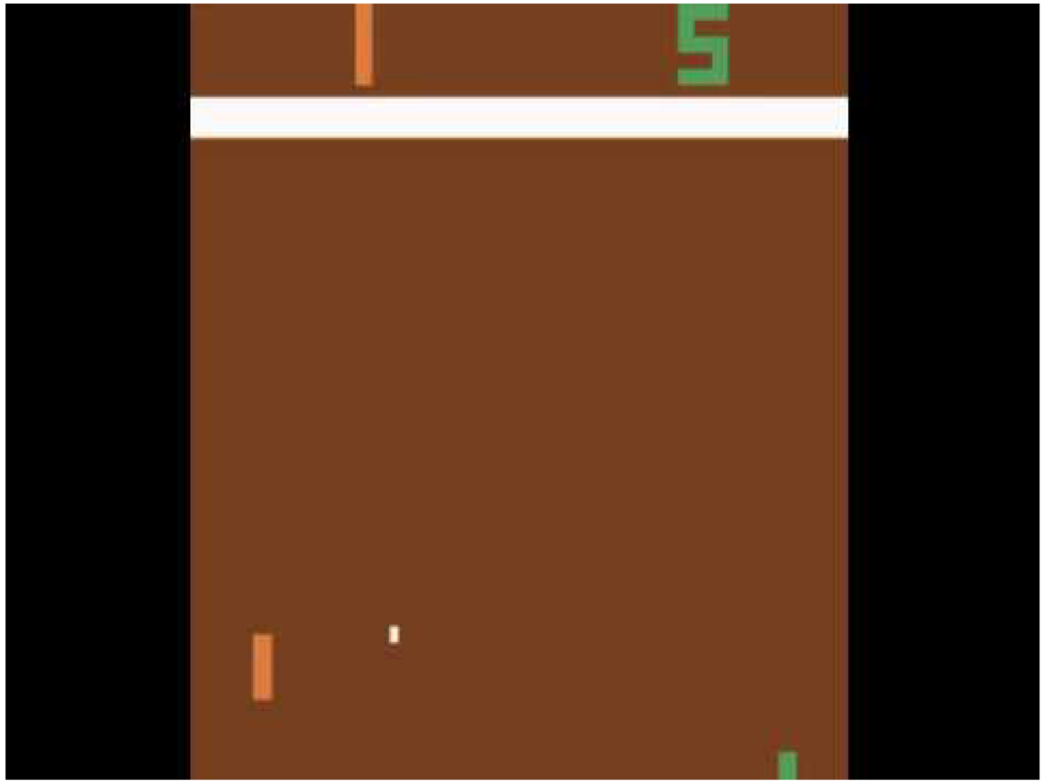


(a) Online-Q (b) Buffer-Q (c) Combined-Q

endtoend.ai

Testbed: Pong

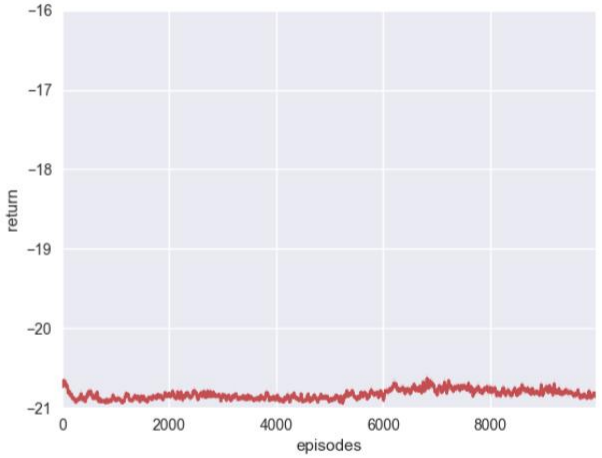
- RAM states used instead of raw pixels
  - More accurate state representation
  - State space: $\{0, \ldots, 255\}^{128}$
- 6 discrete actions



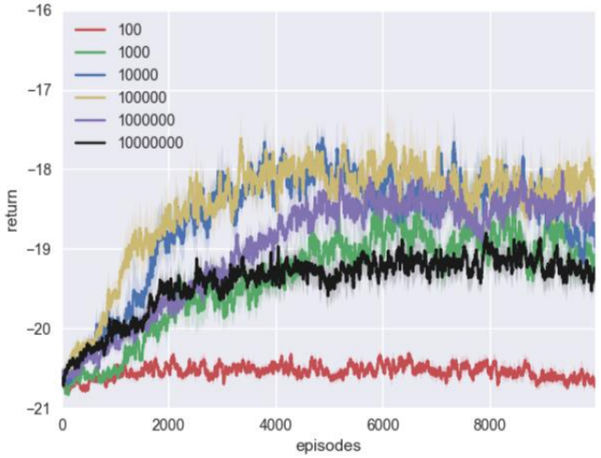
endtoend.ai

Pong Results (Nonlinear)

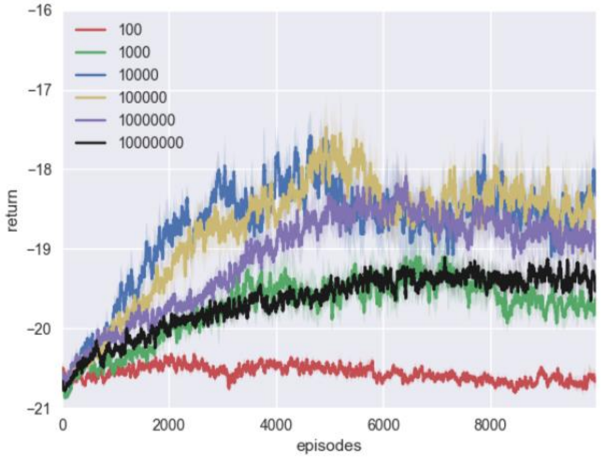
- All 3 agents fail to learn with a simple 1-hidden-layer network
- CER does not improve performance or speed



(a) Online-Q



(b) Buffer-Q



(c) Combined-Q

endtoend.ai

Limitations of Experience Replay

- Important transitions have delayed effects
  - Partially mitigated with PER, but has a cost of $O(N \log N)$
  - Partially mitigated with correct buffer size or CER
- Both are **workarounds**, not solutions

- Experience Replay itself is flawed
- Focus should be on **replacing** experience replay

Thank you!

Original Paper: https://arxiv.org/abs/1712.01275

Paper Recommendations:

- Prioritized Experience Replay
- Hindsight Experience Replay
- Asynchronous Methods for Deep Reinforcement Learning

You can find more content in www.endtoend.ai/slides