RECURRENT EXPERIENCE REPLAY

IN DISTRIBUTED REINFORCEMENT LEARNING

MOTIVATIONS

- Experience Replay
- Stack a fixed number of consecutive frames to overcome the partial observability.
- There is a **need for more advanced memory-based representations** for harder partially observable problems.
- Such as RNN (which is this paper).

CONTRIBUTION 1

- Identify the problem of Representation drift and Recurrent state staleness, with Q-value Discrepancy Metrics
- The effect is even worse in a distributed training setting.
- Diminishing training stability and performance.

CONTRIBUTION 2

Studies about the effect of several approaches to RNN training with experience replay, mitigating the problems mentioned in the previous slides.

CONTRIBUTION 3

Present an agent that integrates these findings to achieve significant advances in the state of the art on Atari-57 (Bellemare et al., 2013) and matches the state of the art on DMLab-30 (Beattie et al., 2016).

R2D2

- Recurrent Replay Distributed DQN (R2D2)
- To achieve good performance in partially observed environment, an RL agent requires a state representation that encodes the information about its state-action trajectory in addition to the current observation.
- This is achieved using an RNN, typically LSTM as a part of the state encoding.
- Learn long-term dependencies.
- State-action trajectories need to be stored in replay and used for training the network.

TWO STRATEGIES

Stored State: Storing the recurrent state in replay and using it to initialize the network at training time.

Pros:

Remedies the weakness of the zero start state strategy

Cons:

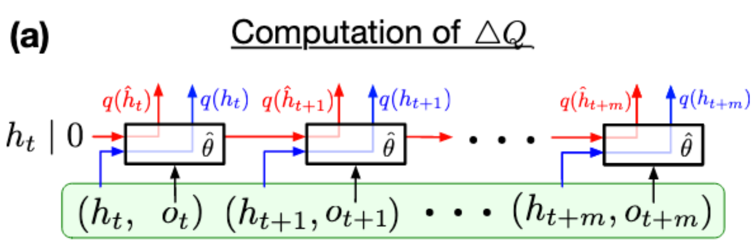
Suffer from the effect of “representational drift” leading to “recurrent state staleness”.

Burn-in: Allow the network a ‘burn-in period’ by using a portion of the replay sequence only for unrolling the network and producing a start state, and update the network only on the remaining part of the sequence.

Pros (by the author’s hypothesis):

Allows the network to partially recover from a poor start state and find itself in a better initial state before being required to produce accurate outputs.

IMPACTS OF THE TWO STRATEGIES



$$\Delta Q = \frac{\|q(\hat{h}_{t+i};\hat{\theta}) - q(h_{t+i};\hat{\theta})\|_2}{|\max_{a,j}(q(\hat{h}_{t+j};\hat{\theta}))_a|}$$

m: number of unrolling states

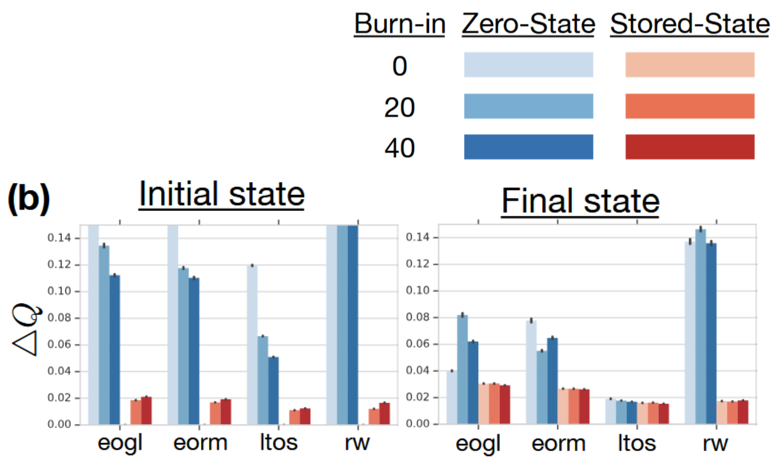
l: number of burn-in states

Q-value Discrepency: Difference between the Q value vector output using the hidden state and the stored recurrent states, normalized by the maximal Q-value

Note that we are not directly comparing the Q-values produced at acting and training time, q(ht; θ) and q(ˆht; θˆ), as these can naturally be expected to be distinct as the agent is being trained. Instead we focus on the difference that results from applying the same network (parameterized by θˆ) to the distinct recurrent states.

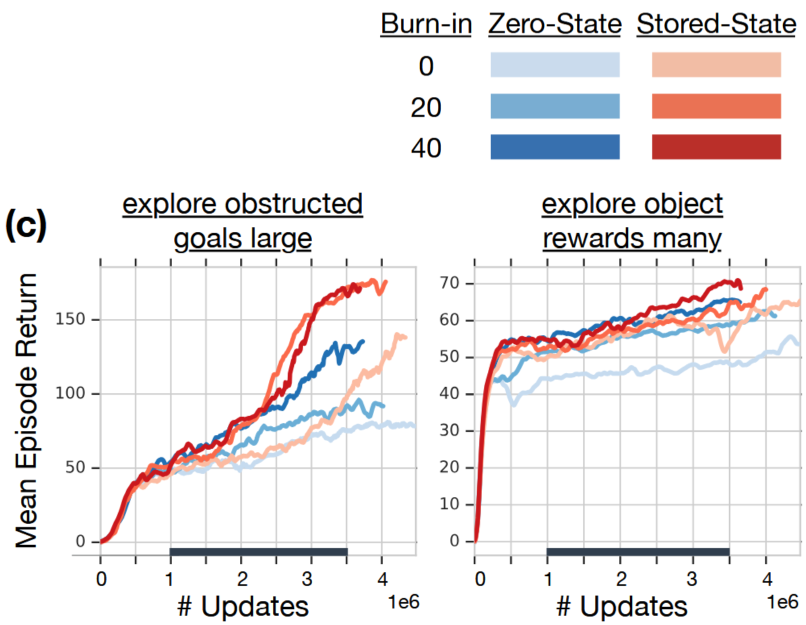
STORED STATE

- (b) Comparing agents trained with different strategies on several DMLab environments in terms of Q-value discrepancy.
- It can be seen that the zero start state heuristic results in a significantly more severe effect of recurrent state staleness.
- We observe that the burn-in strategy on its own partially mitigates the staleness problem on the initial part of replayed sequences, while not showing a significant effect on the Q-value discrepancy for later sequence states.

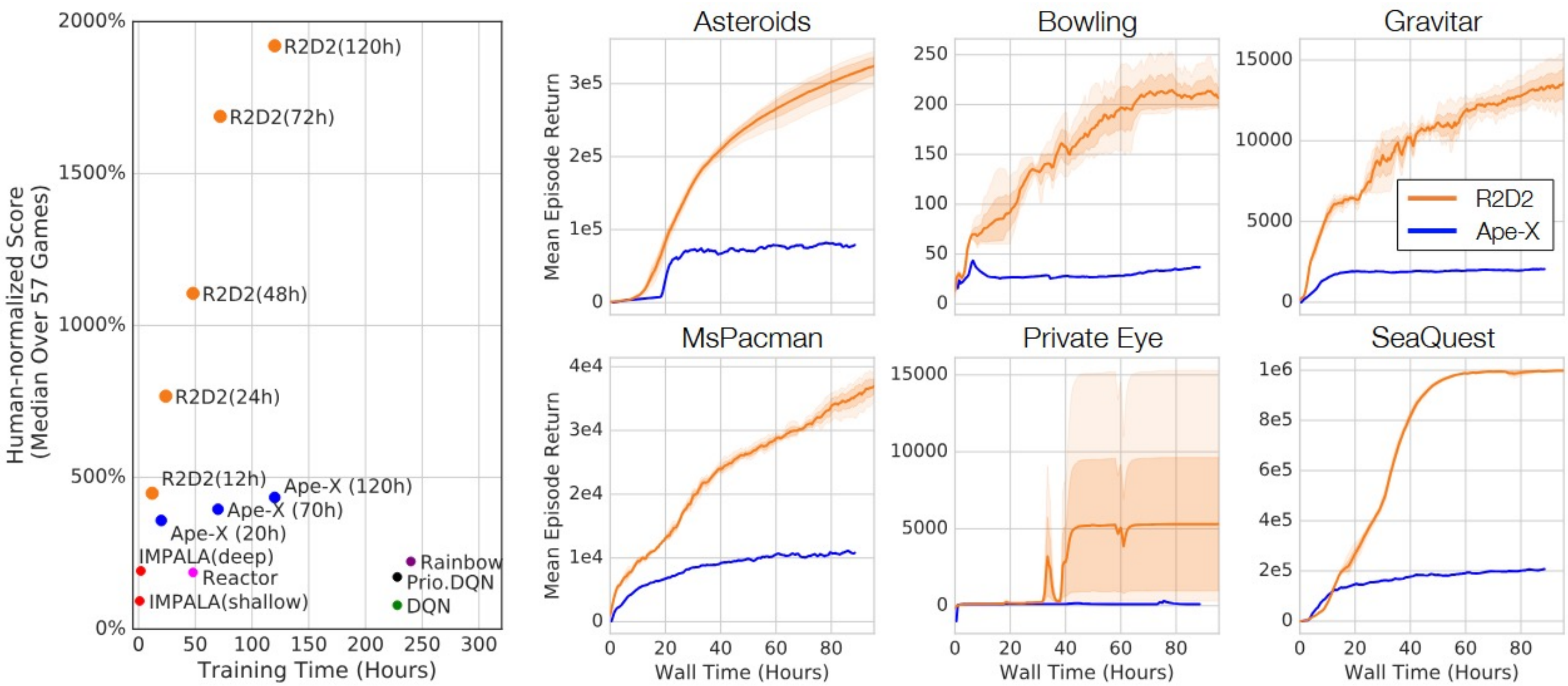


STORED STATE PERFORMANCE

- Empirically, this translates into noticeable performance improvements.
- This itself is noteworthy, as the only difference between the pure zero state and the burn-in strategy lies in the fact that the latter unrolls the network over a prefix of states on which the network does not receive updates.
- Conclusion: Stored-State strategy reduces the representation drift and improves the performance.



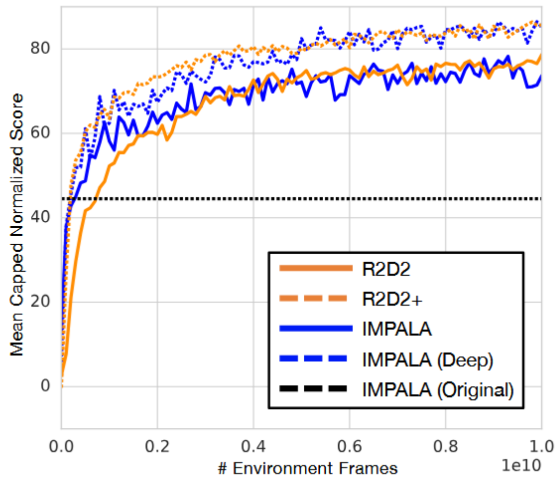
RESULTS ON ATARI-57

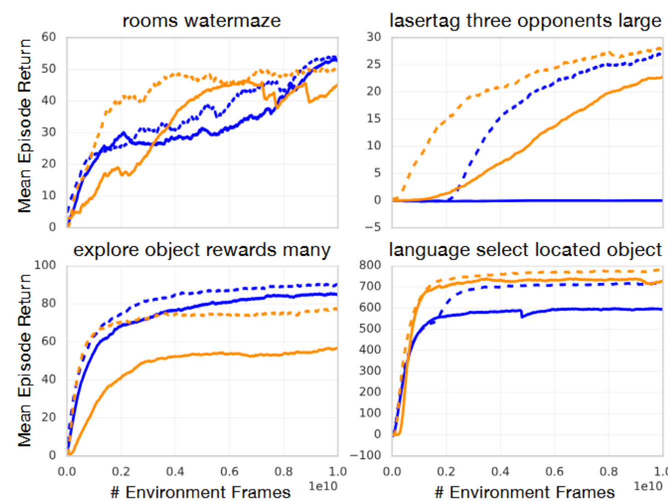


RESULTS ON DMLAB-30

While Atari can largely be approached with only frame-stacking, DMLab-30 requires long-term memory to achieve reasonable performance.

Compares R2D2 with IMPALA

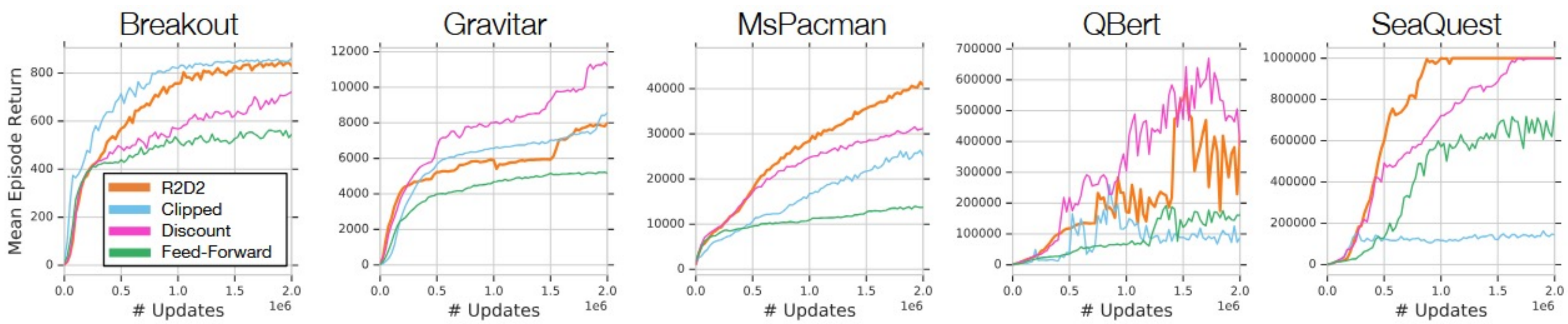




ABLATION STUDY

Ape-X + LSTM = R2D2

Study the role of LSTM.



EFFECT OF MEMORY

- Gradually decrease the history length k from full history to 0.
- Stored State performance decreases more rapidly w.r.t. the reduction in history length.
- This indicates the Stored State strategy is better at letting the agent use memory, compared to the zero state.