Prioritized memory access explains planning and hippocampal replay

Mattar, M. G., & Daw, N. D. (2018).

A project by Hector Kohler

Supervised by Mehdi Khamassi

Summary:

1. Article background and problematic
2. Article main results
3. Article model
4. Experiments
5. Discussion

1) Article background and problematic

- Objectives: optimal decision making in MDP, better understanding of replay.
- Framework: Dyna-Q.

  Dyna-Q :
  - Access experiences (a tuple (s, a, r, s')) during interactions with environment or access passed experiences from a learned model during planning.
  - Perform bellman backups on experiences to update State-Action values.

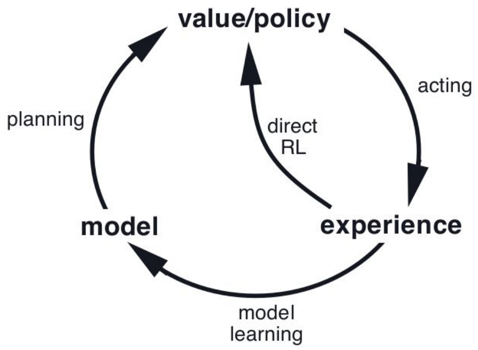


Fig 0 : Relationships between acting planning and learning
(credit Sutton & Barto)

- **Problematic: which experiences should the agent consider at each moment to set the stage for the most rewarding future decisions?**

2) Article main results

- Authors derive from first principles, the utility of retrieving each individual experience at each moment. (Expected Value of Backup)

- Propose that all patterns of replay reflect different instances of a general state-retrieval operation that integrates experiences across space and time to propagate reward information and guide decisions.

- Show prioritized memory access speeds learning.

- Show the existence and balance between forward and reverse replay.

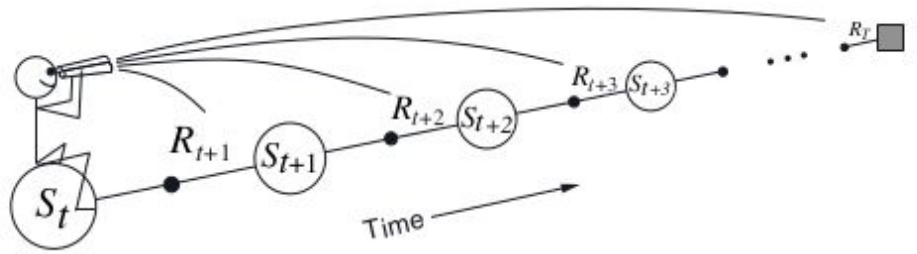


Figure 7.5: The forward or theoretical view. We decide how to update each state by looking forward to future rewards and states.

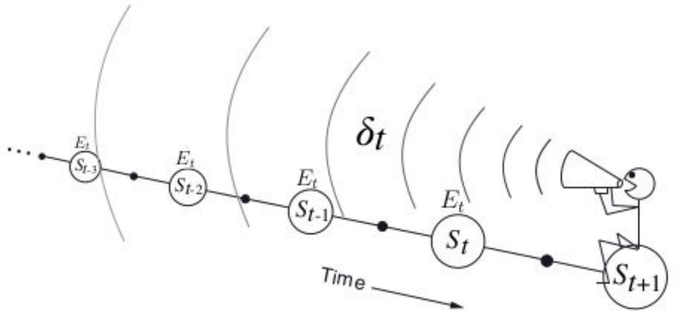


Figure 7.8: The backward or mechanistic view. Each update depends on the current TD error combined with eligibility traces of past events.

Fig 0.1 : Forward and Backward views of replay (credit Sutton & Barto)

3.1) Studied Model

- Dyna-Q base
- At each planning step, the experiences ek = (sk, ak, rk, s'k) to access are chosen to maximise:

$$EVB(s_k, a_k) = Gain(s_k, a_k) \times Need(s_k).$$

$$Gain(s_k, a_k) = \sum_{a \in \mathcal{A}} Q_{\pi_{new}}(s_k, a)\pi_{new}(a|s_k) - \sum_{a \in \mathcal{A}} Q_{\pi_{new}}(s_k, a)\pi_{old}(a|s_k),$$

$$Need(s_k) = \sum_{i=0}^{\infty} \gamma^i \delta_{S_{t+i}, s_k},$$

- **Gain : "quantifies the increase in discounted future reward expected from a policy change at the target state" .**
- **Need : "quantifies the number of times the agent is expected to harvest the gain by visiting the target state in the future" .**

3.2) Studied Model: algorithms

Initialize $Q(s,a)$ and $Model(s,a)$ for all $s \in \mathcal{S}$ and $a \in \mathcal{A}(s)$
Do forever:
(a) $S \leftarrow$ current (nonterminal) state
(b) $A \leftarrow \epsilon$-greedy(S,Q)
(c) Execute action A; observe resultant reward, R, and state, S'
(d) $Q(S,A) \leftarrow Q(S,A) + \alpha[R + \gamma \max_a Q(S',a) - Q(S,A)]$
(e) $Model(S,A) \leftarrow R, S'$ (assuming deterministic environment)
(f) Repeat n times:
$S \leftarrow$ random previously observed state
$A \leftarrow$ random action previously taken in S
$R, S' \leftarrow Model(S,A)$
$Q(S,A) \leftarrow Q(S,A) + \alpha[R + \gamma \max_a Q(S',a) - Q(S,A)]$

Algo 1: Dyna-Q from Sutton & Barto

Initialize $Q(s,a)$, $Model(s,a)$ and $T(s,s')$ for all $s, s' \in S$ and $a \in A(s)$.
Do forever:
(a) $S \leftarrow$ current (nonterminal) state
(b) $A \leftarrow \epsilon$-greedy(S,Q)
(c) Execute action A; observe resultant reward, R, and state, S'
(d) $Q(S,A) \leftarrow Q(S,A) + \alpha[R + \gamma \max_a Q(S',a) - Q(S,A)]$
(e) $Model(S,A) \leftarrow R, S'$
(f) $T(S,:) \leftarrow T(S,:) + \tau(\mathbb{1}[S = S'] - T(S,:))$
(g) Repeat n times:
$maxEVB \leftarrow 0$
$S^* \leftarrow \emptyset, A^* \leftarrow \emptyset, R^* \leftarrow \emptyset, S'^* \leftarrow \emptyset,$
For S in $Model$ do:
Compute Need(S)
For A in $Model(S,:)$ do:
Compute Gain(S,A)
$EVB(S,A) \leftarrow$ Need(S) $\times$ Gain(S,A)
if $EVB(S,A) > maxEVB$:
$maxEVB \leftarrow EVB(S,A)$
$S^* \leftarrow S, A^* \leftarrow A, R^*, S'^* \leftarrow Model(S,A)$
$Q(S^*,A^*) \leftarrow Q(S^*,A^*) + \alpha[R^* \gamma \max_a Q(S'^*,a) - Q(S^*,A^*)]$

Algo 2: Dyna-Q with EVB based planning

4.1) Validation of implementation: study of learning performances

experiment params: simulations: 20 ; episodes:50 ; action policy:e-greed ; planning policy:softmax ; eps:1 ; temperature:0.2 ; gamma:0.9 ; alpha:1 ; planning steps:20 ; transition matrix lr:0.9 ; mdp:6x9maze.

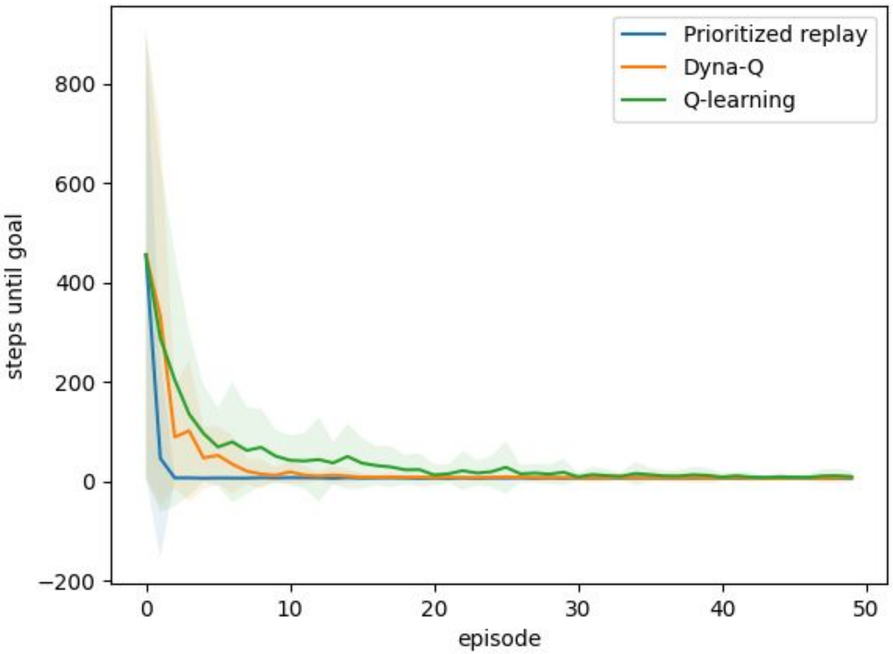


Fig 1: Reproduction of figure 1.d

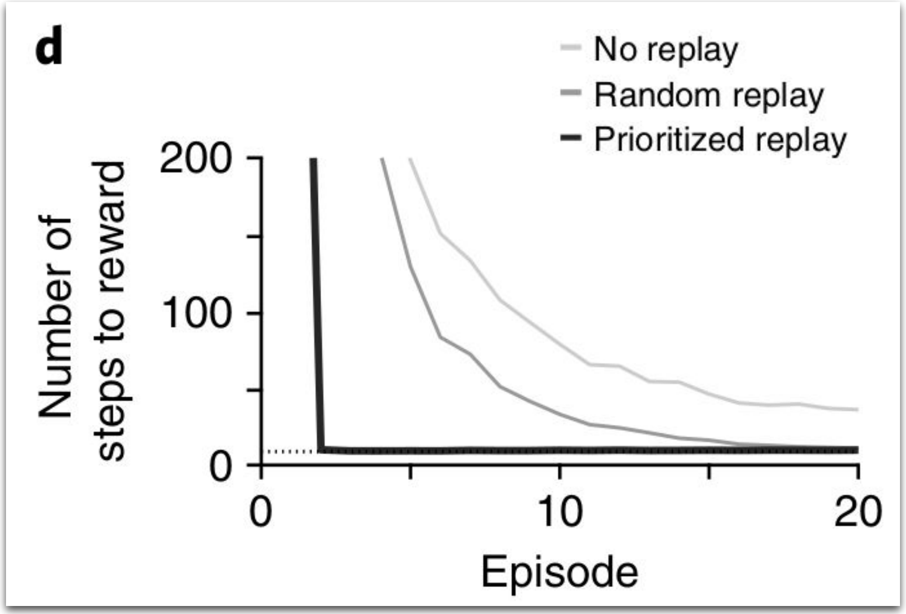


Fig 2 : Original Figure 1.d from the article

4.2) Another view of learning performances

experiment params: simulations: 10 ; episodes:10 ; action policy:e-greed ; planning policy:softmax ; eps:1 ; temperature:0.2 ; gamma:0.9 ; alpha:1 ; planning steps:20 ; transition matrix lr:0.9 ; mdp:6x9maze.

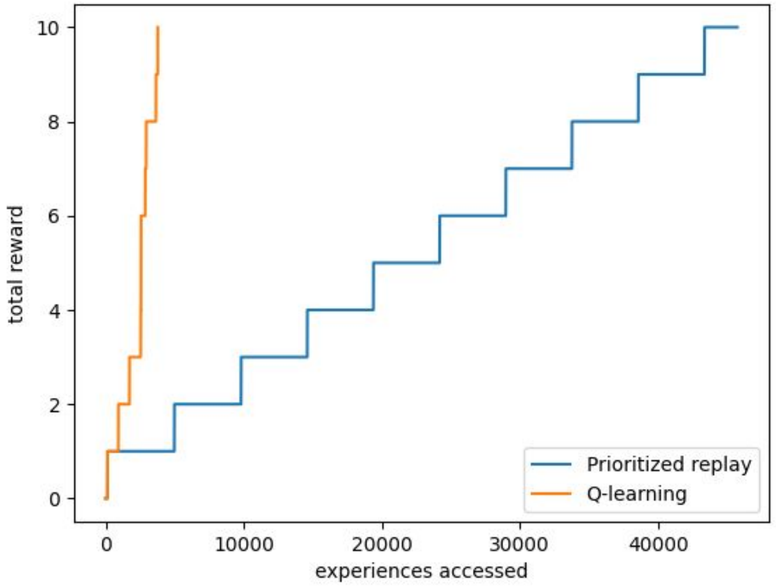


Fig 3 : Comparison of learning perf in terms of reward per experiences accessed

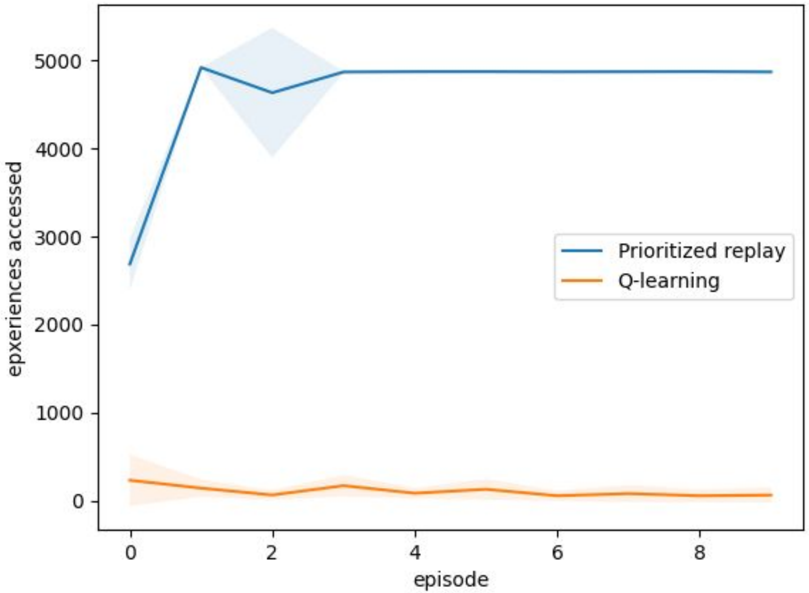
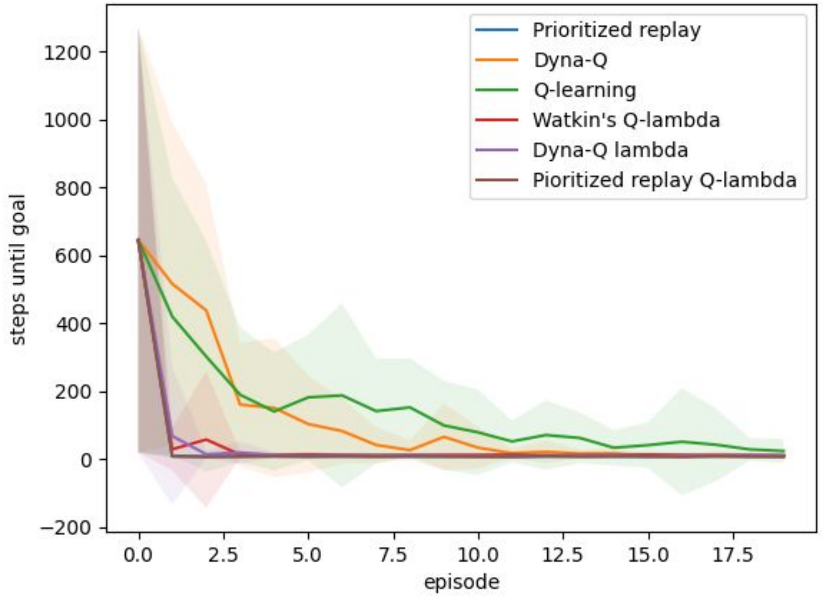


Fig 4 : Comparison of average experiences accessed per episode

4.3) Comparison of TD-learning methods

experiment params: simulations: 20 ; episodes:20 ; action policy:e-greed ; planning policy:softmax ; eps:1 ; temperature:0.2 ; gamma:0.9 ; alpha:1 ; planning steps:20 ; transition matrix lr:0.9 ; mdp:6x9maze ; lambda:0.9



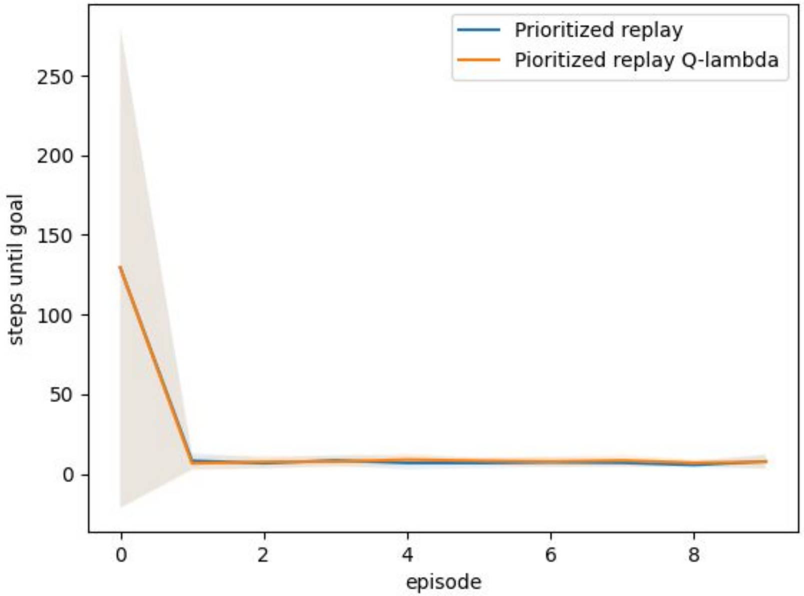
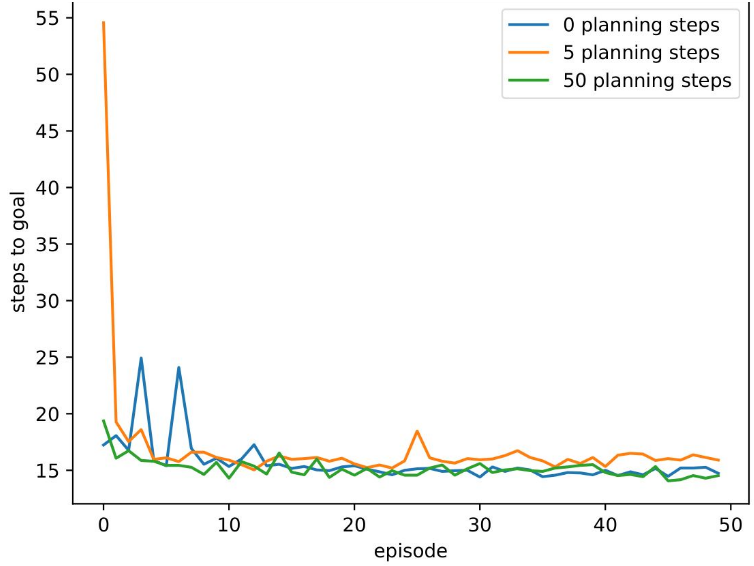


Fig 8 : Comparison of learning performances of TD-learning methods

Fig 9 : Using eligibility traces does not affect Prioritized replay with EVB

4.4) Bonus: base Dyna-Q study of learning performances

experiment params: simulations:30 ; episodes 50 ; action policy = planning policy = e-greedy ; eps:0.1 ; alpha:0.1 ; gamma:0.95 : mdp:maze6x9



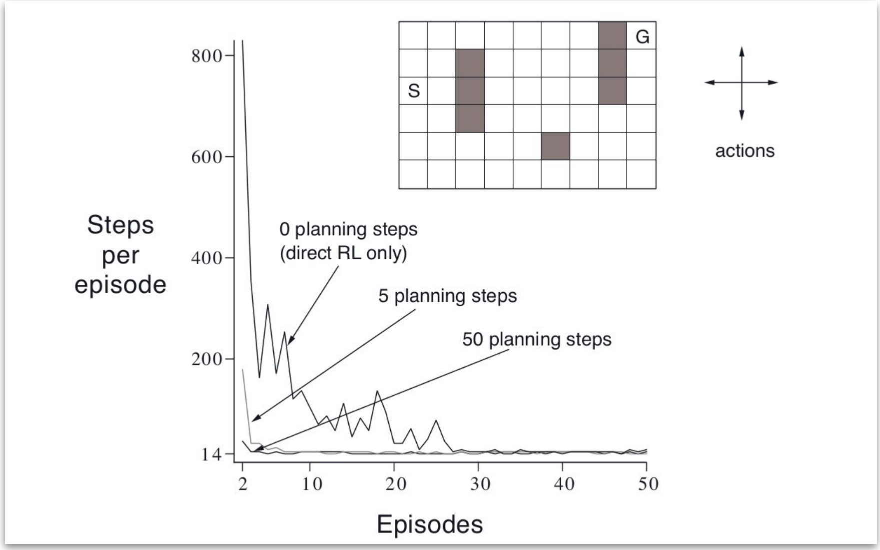


Fig 10 : Reproduction of S&B figure 8.5

Fig 11 :Original figure 8.5 from S&B

5) Discussion

- Many of the fundamentals idea of the article were already mentioned by Sutton & Barto in 2014.
- For future work on model-based and planning methods, maybe learning performances should be measured against the number of experiences accessed.
- It would also be interesting to try the model in the continuous domain. To better emulate the role of place cells.
- Further study why Prioritized replay with EVB learning is not affected by the use of Elig Traces.

References

- SimpleMazeMDP https://github.com/osigaud/SimpleMazeMDP
- Matlab code of the article https://github.com/marcelomattar/PrioritizedReplay
- The article: Mattar, M. G., & Daw, N. D. (2018). Prioritized memory access explains planning and hippocampal replay
- Sutton & Barto (2014-2015). Introduction to Reinforcement Learning, 2nd edition

Annexes

experiment params: simulations: 20 ; episodes:50 ; action policy:softmax ; planning policy:softmax ; temperature:0.2 ; gamma:0.9 ; alpha:1 ; planning steps:20 ; transition matrix lr:0.9 ; mdp:5x5maze.

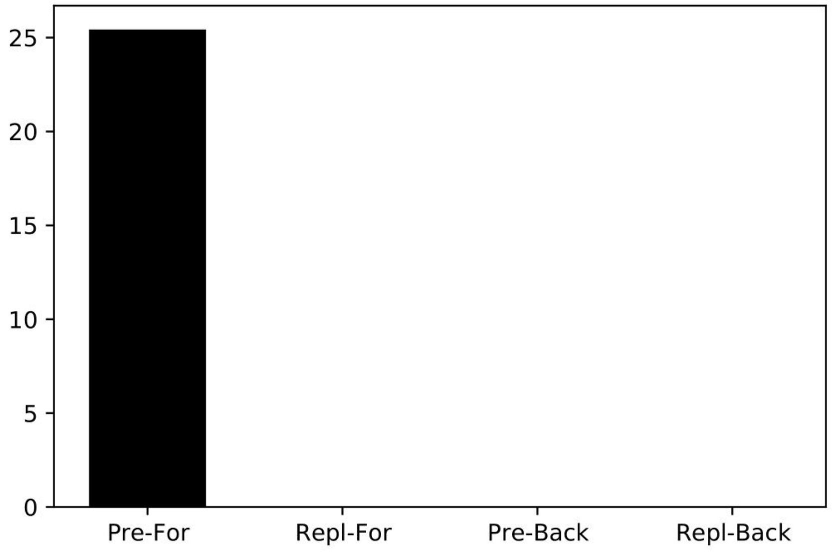


Fig 4 Reproduction (failed) of figure 3.a

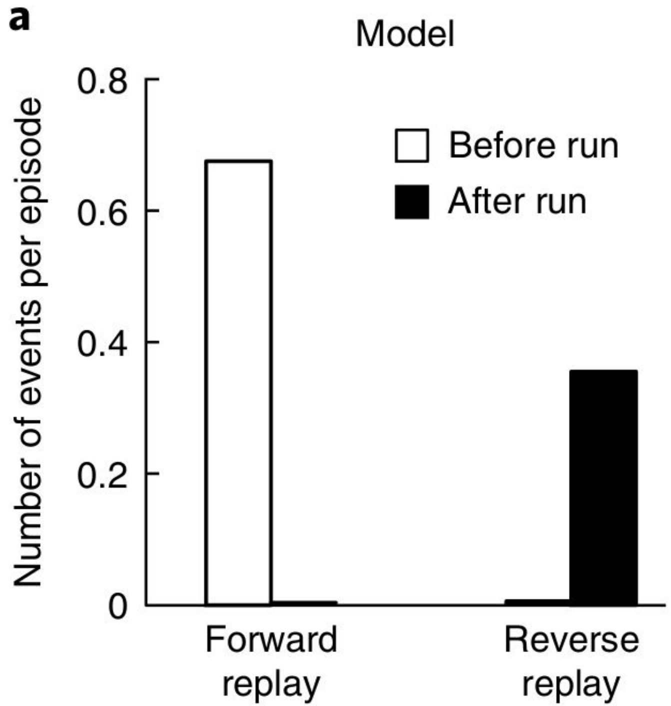


Fig 5 : Original figure 3.a from the article

Study of Forward/Reverse sequences balance

Annexes

experiment params: simulations: 20 ; episodes:50 ; action policy:softmax ; planning policy:softmax ; temperature:0.2 ; gamma:0.9 ; alpha:1 ; planning steps:20 ; transition matrix lr:0.9 ; mdp:5x5maze ; rew multip:4 ; multip proba:0.5;

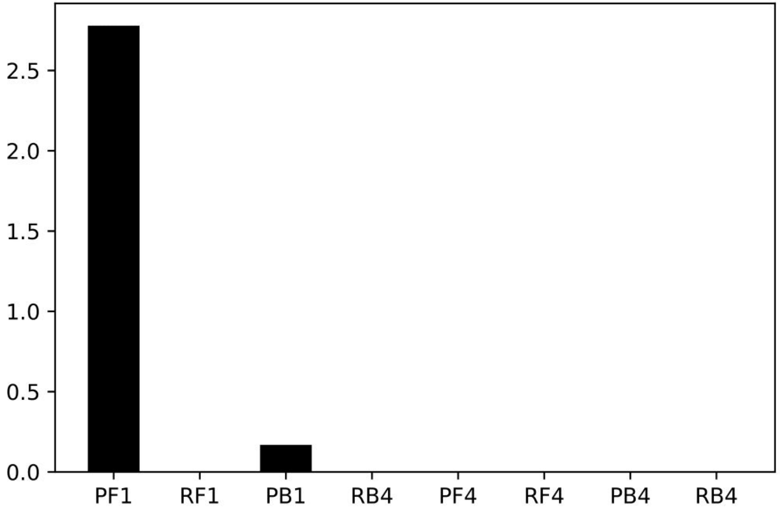


Fig 6 : Reproduction (failed) of figure 5.c

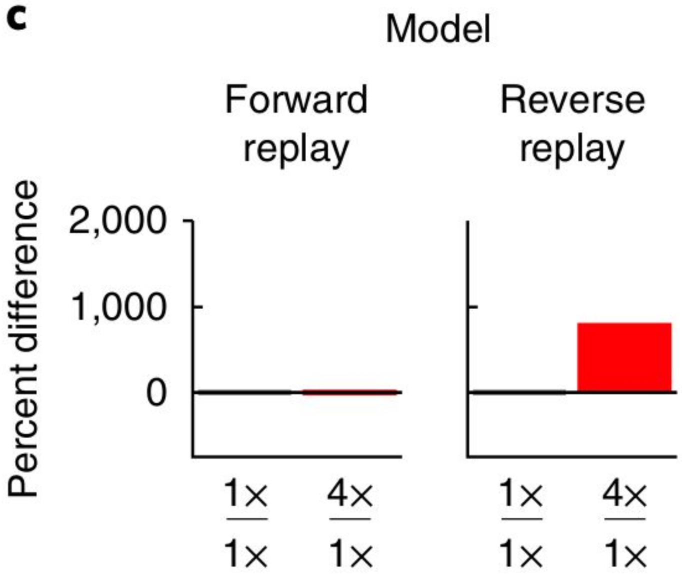


Fig 7 : Original figure 5.c

Study of Forward/Reverse sequences balance w/ reward shift