Teaching a neural network to count: reinforcement learning with "social scaffolding"

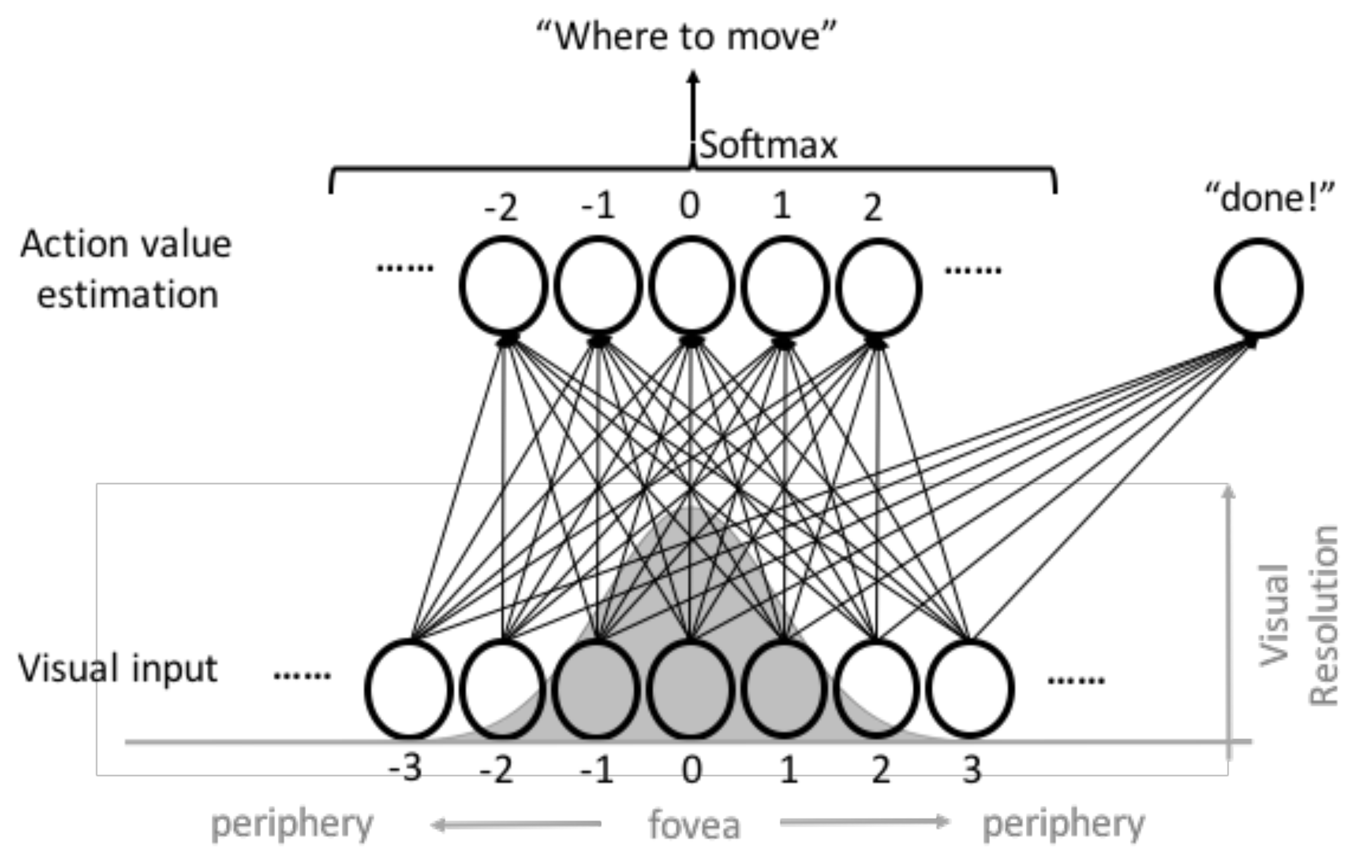
Qihong Lu, University of Wisconsin-Madison
James L. McClelland, Stanford University

Background

Counting skill is a foundation for more sophisticated math concepts. The current project tries to capture a sub-task of counting: given an array of objects, touch every object exactly once, which is related to the one-to-one principle [1].

Although some have considered aspects of counting principles as innate, we examine how these skills might be learned from social support, such as enriched feedback and teacher demonstration.

Modeling Detail

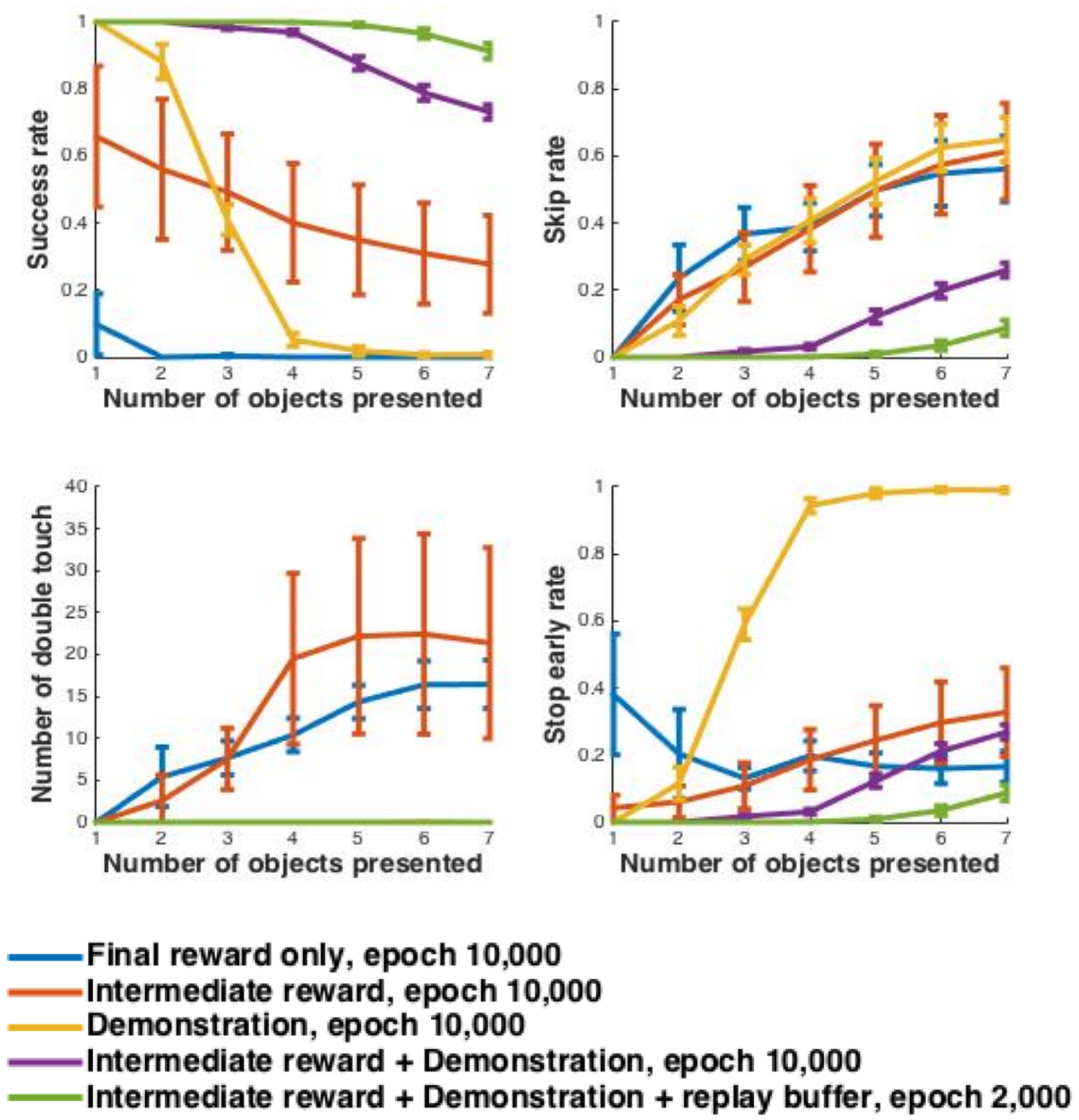


$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\left[R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)\right]$$

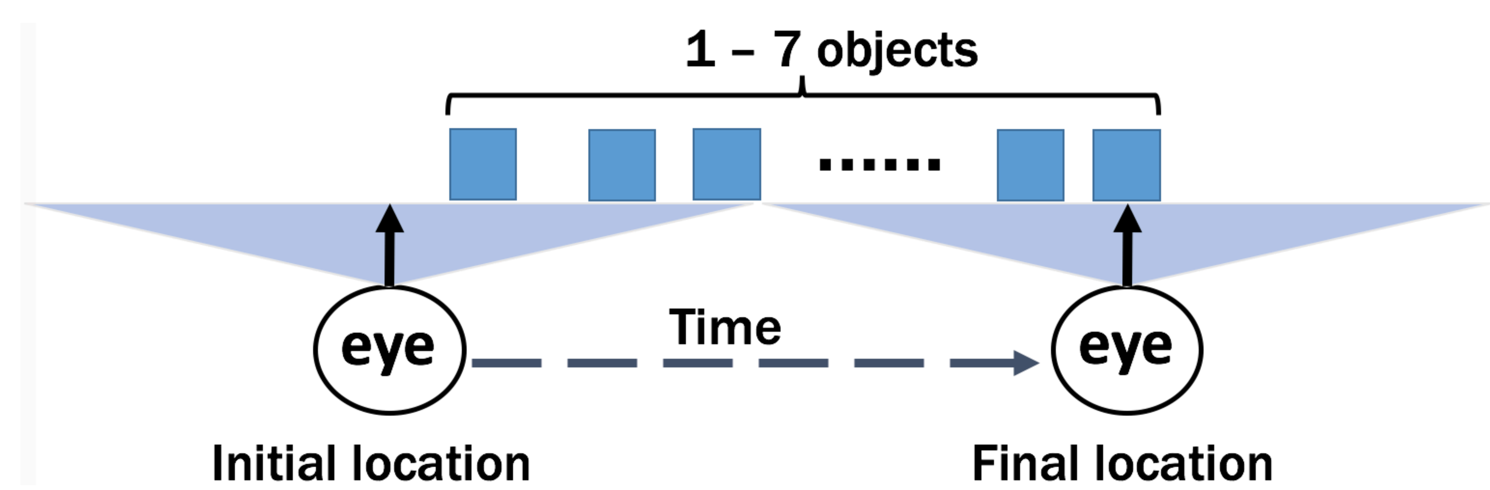
In every trial, a one-dimensional object array is randomly initialized on the right-hand side of the agent's initial fixation point. The goal is to make a series of movements, touching every object exactly once in order and then touch "done" in no more than 100 steps. The model has a limited visual field. At each time step, the model can make a virtual 'eye+hand' movement determining its visual input at the next time step and where it is touching. The input vector encodes objects in a viewer-centered frame, represented by some Gaussian bumps (see model environment).

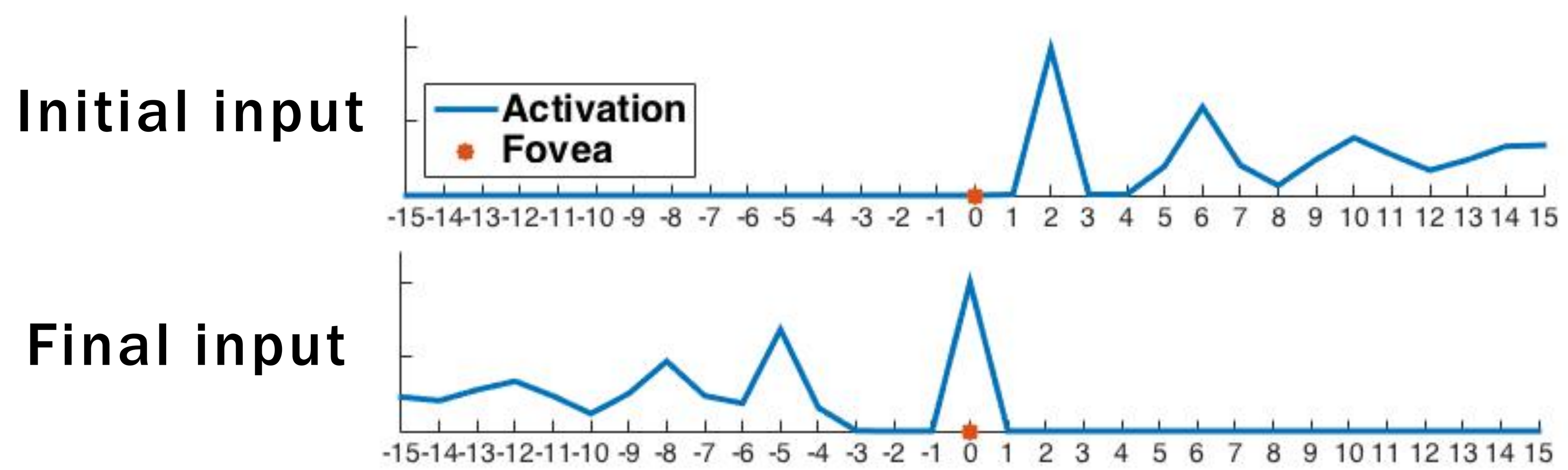
The model estimates action values with a one-layered neural network with softmax non-linearity, using the Q-learning rule [2], listed above. The model can be augmented with an experience replay buffer [3] and a target network [4] (also see [5]).

Compare different teaching strategies



Model environment





Reward policy & Teaching strategies

Error types shown in graphs:
1: Stop early – say "done" when there exists an untouched object
2: Double touch – touching the same object twice in a trial
3: Skip – touching the (k+1)th object before touching the kth object

Note: These errors are not mutually exclusive. Also, touching empty space is not in itself an error though it can lead to delay in receipt of reward.

Teaching strategies:
There is always a final reward on successful completion. We also considered two types of "social scaffolding" and their combination:
1. Intermediate reward: Reward the agent for touching the correct next object. The reward magnitude is one-half of the final reward.
2. Demonstration: Force the agent to execute the maximally efficient action sequence and provide the corresponding reward. In this condition, we alternate demonstration trials and the regular self-exploration trial.
3. Intermediate reward + Demonstration: Combination of (1) and (2)

Summary

Social scaffolding made learning easier, because:
- Intermediate feedback makes the task more supervised.
- Demonstration forces exposure to the optimal solution.

These results provide insights about how social scaffoldings support learning from a computational perspective. Further research will extend these explorations to multi-layer recurrent architectures and more complex task settings.

References

[1] Gelman, R., Gallistel, C. R. (1978). The child's understanding of number. Harvard U. press.
[2] Sutton, R. S., & Barto, A. G. (1998). *Reinforcement learning*: An introduction. MIT press.
[3] Lin, L.-J. (1993). Reinforcement learning for robots using neural networks. Technical Report, DTIC Document.
[4] Van Hasselt, H., Guez, A., Silver, D. (2015). Deep Reinforcement Learning with Double Q-learning. arXiv.
[5] Mnih, V., et al. (2015). Human-level control through deep reinforcement learning. *Nature*, 518(7540), 529–533.

Simulation source code:
https://github.com/QihongL/mathCognition_PDP_RL

Acknowledgement: This research is supported by the Stanford Center for the Study of Language and Information summer research program and the Rumelhart Emergent Cognition Fund.

qihong.lu@wisc.edu