Minerva-Q: A Multiple-Trace Memory System for Reinforcement Learning

Renan Ozen (renanozen@cmail.carleton.ca)
Robert L. West (robert.west@carleton.ca)
M. Alex Kelly (alex.kelly@carleton.ca)
Department of Cognitive Science, Carleton University
1125 Colonel By Drive, Ottawa, Ontario, K1S 5B6 Canada

Abstract

We propose Minerva-Q, a multiple-trace memory model capable of perceptual-motor reinforcement learning. This model combines Q-learning with the Minerva family of memory models. In our simulations we found our Minerva-Q agent learned increasingly optimal solutions to the Cart Pole task and reproduced human-like performance when presented with minimal expert training examples in a sparse reward task.

Keywords: Minerva; reinforcement learning; memory; perception; motor; artificial intelligence; Q-learning

Introduction

Perceptual-motor learning can be observed in the behaviours of many animal species and may be responsible for the development of embodied skills including tool use, spatial navigation, hunting and foraging, or in humans, skills such as riding a bicycle or playing an instrument. Learning more broadly is known to require some form of memory and entails adaptive processes to reinforce, inhibit, or alter information stored in memory. We may thus posit a minimal perceptual-motor learning agent as consisting of a set of adaptive learning and action-selection processes and a set of sensory inputs and motor outputs connected to a memory store. In the fields of cognitive science, artificial intelligence, and robotics, we hold that it is useful to consider and study such minimal configurations, as most living organisms do not possess the "higher" cognitive skills of humans such as those related to semantic reasoning, yet non-human organisms are capable of completing tasks our best models and intelligent systems are either incapable of handling at present or may require many orders of magnitude more computational resources to perform. Moreover, human motor learning may be more akin to this type of system.

Towards this end, we propose Minerva-Q, a minimal memory model capable of perceptual-motor skill acquisition via an implementation of Q-learning (Watkins, 1989), which is a form of temporal-difference (TD) learning algorithm motivated by classical conditioning theories of animal skill acquisition. "Minerva" in Minerva-Q refers to a family of multiple-trace memory models largely based on MINERVA 2 (Hintzman 1984; Hintzman, 1986), which have been used to model associative learning (Jamieson et al., 2012) and decision-making (Dougherty et al., 1999), among a large set of other cognitive processes (see Jamieson et al., 2022). Given the versatility of Minerva models, we hypothesized that a Minerva-like model of perceptual-motor memory could be applied to the domain of reinforcement learning (RL). We found that our Minerva-Q RL agent could solve a dense-reward and a sparse-reward RL task and observed its human-like capacity to learn from a minimal number of expert examples. This latter feature sets Minerva-Q apart from models like the so-called "deep Q-network" (Mnih et al., 2015) that may require hundreds of training examples (or more) to reproduce expert task performance.

In this paper, we first describe the specifications of the Minerva family of memory models that set precedence for Minerva-Q. Next, we provide a detailed account of the structure and mechanisms of Minerva-Q. Finally, we outline the results of a Minerva-Q agent in two simulated RL tasks and discuss the wider implications of our model to the cognitive sciences.

MINERVA 2

In MINERVA 2, each experience is stored as a separate item in memory, known as a memory trace (hence, multiple-trace memory). More specifically, MINERVA 2 consists of two memory subsystems: primary memory (PM), which is a limited temporary memory store analogous to working memory, and a long-term secondary memory (SM) containing all memory traces. Information that passes into PM is sent to SM as a "probe," which returns a single "echo" to PM. This echo represents a retrieved memory instance and is constructed during each retrieval as the sum of activated traces. Thus, retrieval cannot be understood as a lookup process, and the addition of identical traces to SM has the effect of strengthening the influence of these traces during activation.

Memory traces in MINERVA 2 are represented as integer-valued vectors with random values $\mathrm{v} \in \{-1,0,1\}$. These traces are stored in an $m \times n$ matrix M, where m is the number of memory traces and n is the number of values in each trace. In MINERVA 2, the similarity s between a probe p and a memory trace is computed as a normalized dot product,

$$s_{\mathrm{i}} = \frac{1}{n_{\mathrm{R}}} \sum_{\mathrm{j}=1}^{\mathrm{n}} p_j M_{\mathrm{i,j}} \qquad (1)$$

where n_{R} is the maximum number of nonzero values in the probe or memory trace M_i. This similarity metric is cubed to increase the signal-to-noise ratio in the echo, producing an activation vector a of the trace given the probe:

$$a_{\mathrm{i}} = s_{\mathrm{i}}^3 \qquad (2)$$

Finally, the echo c may be computed as the sum of activated traces:

$$c_{\mathrm{j}} = \sum_{\mathrm{i}=1}^{m} a_{\mathrm{i}} M_{\mathrm{i,j}} \tag{3}$$

Learning in MINERVA 2 is probabilistic, where each nonzero feature in the probe is added to memory with probability L. Forgetting is treated as the opposite of learning, where each nonzero feature has a probability F of being set to 0. According to Hintzman (1986), learning with $L = 0.25$ is equivalent to learning with $L = 1.00$ and forgetting with $F = 0.75$.

MINERVA 2 Variants

Variants of MINERVA 2 typically commit to Hintzman's (1984; 1986) assumptions but may propose different functions with respect to encoding, activation, learning, and forgetting to model their targeted phenomena. For example, Jamieson et al. (2018) offer a Minerva variant that combines the retrieval operations of MINERVA 2 with the encoding scheme of BEAGLE (Jones & Mewhort, 2007), and compute activation under a cosine similarity function. Likewise, Collins et al. (2020) formalize a probabilistic learning mechanism that prioritizes surprise, conceptually grounded in the discrepancy encoding of Jamieson et al. (2012).

These examples set precedence for many of the formalizations developed for Minerva-Q, which we argue should be considered a Minerva-*like* model as it retains the core operations of MINERVA 2 but departs somewhat from Hintzman's (1984) theoretical assumptions about memory.

Minerva-Q

Q-Learning

Minerva-Q implements Q-learning (Watkins, 1989) to handle perceptual-motor tasks. Q-learning is a model-free reinforcement learning method. More formally, Q-learning, when implemented in a table of Q-values, converges on an optimal action-selection policy for finite, discrete-time Markov decision processes (Watkins & Dayan, 1992). At each time step t, a Q-learning agent has the task of selecting some action $A_t \in A$ given some state $S_t \in S$, where S is a finite set of discrete states and A is a finite set of discrete actions. After taking said action, the agent receives a reward r, which is used to update its policy according to the formula,

$$\begin{aligned} Q_{\mathrm{new}}(S_t, A_t) \leftarrow (1-\alpha) Q(S_t, A_t) \\ + \alpha\left(r + \gamma \max\left(Q(S_{t+1}, A)\right)\right) \end{aligned} \tag{4}$$

where $Q(S_t, A_t)$ (the Q-value) is the expected discounted reward for selecting action A_t given state S_t under the current policy, γ is a discount factor that evaluates rewards received earlier as higher than those received later (if $\gamma < 1$) by a factor of γ^t, and α is a learning rate which adjusts the impact of Q-value updates at each time step.

Retrieval

At each time step during a task, a state vector is passed into Minerva-Q's PM which is then encoded into a probe for echo retrieval from SM. Memory traces in SM are a concatenation of a state vector and n_A Q-values (the "action vector"), where n_A is the number of discrete actions the Minerva-Q agent can perform during the task. SM in Minerva-Q is thus represented as an $m \times (n_S + n_A)$ matrix M, where m is the number of stored traces and n_S is the dimension of the encoded probe p.

Minerva-Q differs from other Minerva models in that the dimension of its probes is not equal to the dimension of its traces. This is handled by activating only the part of each trace vector corresponding to the probe, where the activation vector a is computed under a cubed cosine similarity function:

$$a_{\mathrm{i}} = \left(\frac{\sum_{\mathrm{j}=1}^{n_S} p_{\mathrm{j}} M_{\mathrm{i,j}}}{\sqrt{\sum_{\mathrm{j}=1}^{n_S} p_{\mathrm{j}}^2} \sqrt{\sum_{\mathrm{j}=1}^{n_S} M_{\mathrm{i,j}}^2}} \right)^3 \tag{5}$$

We may justify this decision with the reasoning that Q-values do not exist as objects in the environment, rather they are internal representations of the expected values of particular actions and therefore should not be considered in similarity measures between stored memory traces and what the agent is experiencing (or observing). Under this interpretation, we see the probe as fundamentally tied to perception, or more specifically to the observation and transduction of a state vector, and activation as an associative memory process, the latter position informed by Hintzman (1990). It may therefore be more helpful to understand the stored Q-values in the action vector as metadata attached to an observed state that influences action-selection rather than playing a role in associative perceptual memory processes. Further, we note precedence for this kind of partial activation in other Minerva models (e.g., Johns et al., 2016).

Given this interpretation, trace activation in Minerva-Q still must be understood as influencing all information in a stored memory trace, thus the retrieved echo vector c has a dimension of $n_S + n_A$, and is computed like in MINERVA 2:

$$c_{\mathrm{i}} = \sum_{\mathrm{j}=1}^{n_S + n_A} a_{\mathrm{j}} M_{\mathrm{i,j}} \tag{6}$$

Perceptual-Motor Learning

A Minerva-Q agent learns as follows. First, an initial state observation is passed into PM and copied into the bottom slot of a Q-buffer, which is a matrix B with a dimension of $2 \times (n_S + n_A)$ and is initialized to all zeros. For the sake of clarity, we refer to the first row of B as its top slot and the second row as its bottom slot. Note that since the observation

probe has a dimension of n_S, the latter n_A dimensions of the top slot are left unchanged. Then, the probe is used to retrieve an echo (all zeros if memory is empty) from SM and the latter n_A values (the Q-values, or action vector) from the echo are copied into the latter n_A values of B's bottom slot.

Next, the Minerva-Q agent is tasked with selecting an action. Here we leave the particulars up to modelers, as various heuristics may be implemented across different tasks to balance between exploration of an environment and exploitation of the learned policy (see Amin et al., 2021). However, in our simulations we used a decaying ϵ-greedy heuristic which selects actions according to a random uniformly distributed variable $x \in [0,1]$ and a variable $\epsilon \in [0,1]$. At each time step t, x is randomly chosen and an action A_t is selected according to the strategy,

$$A_t = \begin{cases} maxarg\left(\left[B_{2,n_S+1} \dots B_{2,n_S+n_A}\right]\right), \text{if } x < \epsilon \\ randarg\left(\left[B_{2,n_S+1} \dots B_{2,n_S+n_A}\right]\right), \text{otherwise} \end{cases} \quad (7)$$

where the function *maxarg* returns the index of the maximum value of the action vector, and the function *randarg* returns a random index of the action vector. Then, ϵ decays at each time step by some fixed amount until it reaches a predetermined minimum value.

Before performing the selected action, the contents of B's bottom slot are copied to its top slot. After performing the action at time t, the Minerva-Q agent receives a reward and observes a new state which is then copied to B's bottom slot and used to probe memory to retrieve an echo c_{t+1}. As before, the action vector portion from c_{t+1} is copied into B's bottom slot.

At this point, having performed its first action and received a reward r, the Minerva-Q agent can update the Q-value in B's top slot corresponding to A_t using the formula,

$$\begin{aligned} B_{1,n_S+A_t} \leftarrow (1-\alpha)B_{1,n_S+A_t} \\ + \alpha\big(r \\ + \gamma \max\left(\left[B_{2,n_S+1} \dots B_{1,n_S+n_A}\right]\right)\big) \end{aligned} \quad (8)$$

where α and γ represent the same parameters as in equation (4). Next, the action vector in B's top slot is normalized to promote numerical stability, then finally the agent forgets (as explained below), and the top slot is added as a new item to SM without any information loss. From here, the algorithm may loop from the point of action selection.

Forgetting

Since there is no formal mechanism by which stored memory traces can be updated in the Minerva framework and thus no obvious way to update Q-values akin to replacing them in a table, Minerva-Q leverages forgetting to probabilistically clear memory traces most similar to what is being added (again, considering only the first n_S dimensions during activation) allowing for a somewhat noisy yet effective method of updating stored Q-values. This forgetting is implemented as logistic function similar to the discrepancy encoding function of Minerva-DE (Collins et al., 2020) but modified to enable targeted forgetting. Before an item is added to memory, each element of trace M_i (excluding those containing Q-values) has a probability F of being set to 0. This probability is determined by the trace's activation a_i to the item being added according to the function,

$$F_i = \frac{1}{1 + e^{-\phi a_i + \beta}} \quad (9)$$

where the parameters ϕ and β adjust the slope and bias respectively. Tuning these parameters allows for implementations ranging from those with hard, precise forgetting thresholds to those with softer, more broad targeting to manage memory capacity, as traces containing all 0's in their first n_S dimensions can safely be removed from memory.

Trace Encoding

As outlined, each instance stored in memory is a concatenation of an observation and a normalized action vector containing Q-values. However, due to the use of vector cosine as a similarity metric in Minerva-Q, observation state spaces with a low dimension might result in suboptimal learning. In the most extreme case, a task providing only a single floating-point number as an observation will restrict the similarity function to the set of possible activations: {-1, 1}. Thus, it may be useful to process (i.e., transduce) an observation prior to its delivery to PM. In our simulations, observations are expanded such that each constituent floating-point value is represented as a bit vector with a dimension of $b - 1$, where b is the value's number of bits of precision (more accurately, the width of its exponent plus the width of its mantissa). The value's sign bit is not included in this expansion but is used to populate the delivered probe vector with either 1's and 0's or -1's and 0's. This preserves the semantics of the similarity metric (i.e., x and $-x$ will have a similarity of -1.0). Thus, the dimension of each trace in memory in our simulations is $(b-1) \times n_S + n_A$.

This encoding method, though effective in our simulations, is likely suboptimal, as our similarity function is not sensitive to magnitude. For example, the similarity between 3.14 and 3.15 in our implementation is 0.58, whereas the similarity between 3.14 and 1.268×10^{-308} is 0.60. Intuitively, we might expect numbers closer together to be more similar to each other than to numbers that differ by many orders of magnitude. We encourage future research to develop different representational schemes and/or similarity functions that improve our approach and preserve this expectation. A possible approach for future investigation may involve the use of fractional binding to represent continuous spaces (e.g., Komer et al., 2019).

Simulations

For our simulations, we implemented Minerva-Q in Python 3.10.2 using PyTorch (Paszke, et al., 2019). We chose the OpenAI Gym library (Brockman et al., 2016) environments

CartPole-v1 (Cart Pole) and *MountainCar-v0* (Mountain Car) to simulate our tasks. Notably, OpenAI Gym has been designed specifically for reinforcement learning and is used across the artificial intelligence community as a benchmarking tool.

To improve learning, after each trial (or episode) we compared the agent's results to previous trials and if we found no significant improvement, cleared its memory up to the last best trial (or to the last trial better than some minimum performance threshold). Though this procedure is not mandatory for Minerva-Q to learn satisficing strategies, we decided to include it to demonstrate the potential of using Minerva-Q to rapidly optimize a strategy. We conjecture based on our limited tests that given enough time, Minerva-Q will tend towards increasingly optimal solutions, however this notion requires further corroboration and testing. As precedence, a conceptually similar approach is taken in instance-based cognitive models (see e.g., Gonzalez et al., 2003). Under this frame, our approach may be understood as setting the activation or utility of the memory chunks stored during a suboptimal trial to 0.

Cart Pole Task

The Cart Pole (also known as inverted pendulum) task is a classic control problem that requires an agent controlling a cart to keep upright a pole attached to a joint fixed to the cart's center. In this environment, a reward of 1 is given at each time step. The task terminates if the pole falls past 12 degrees in either direction, when the cart moves past a certain position in either direction, or when the agent reaches the maximum of 500 steps. At each time step, the agent observes the cart position, cart velocity, the pole's angle, and its angular velocity, and must choose between one of two discrete actions: pushing the cart either left or right.

For this task, we set ϕ to 11, and β to 8 in our forgetting function, informed by the parameters used in the Minerva-DE (Collins et al., 2020) simulations. We set α to 0.8 and γ to 0.99 in our Q function to promote a long-term time-horizon.

At each trial 1, the Minerva-Q agent has zero knowledge of the task stored in memory, and chooses actions based on a decaying ϵ-greedy policy. Figure 1 shows our results after 200 trials, averaged over 50 sets of trials; ϵ values are shown in red, beginning at 0.99 and decaying to a minimum of 0.02.

Though our results do not show the agent achieving a maximum score of 500, we see the number of steps taken before the task terminates increasing as ϵ decays, continuing in a clear upward trend toward trial 200, demonstrating learning.

Mountain Car Task

The Mountain Car task originally appears in Moore (1990) and consists of a car on curved mountain-like one-dimensional surface, which is consistent across trials. Starting at a random location at the bottom of a valley, an agent must maneuver the car to reach a goal position at the

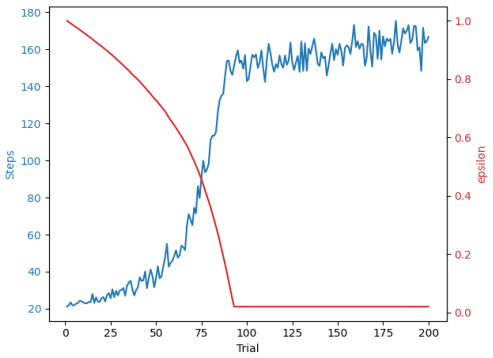

Figure 1: Average Cart Pole results across 50 sets of 200 trials. Blue indicates the average number of steps per trial (higher is better); red indicates the average ϵ value per trial.

mountaintop by accelerating back and forth across the valley until enough speed is gained to drive up a steep incline.

At each time step, the agent receives an observation describing the car's position along the x-axis and its velocity and must choose between three actions: accelerate to the right, to the left, or do nothing. Interestingly, Moore's solution to this problem, like Minerva-Q, stores each experience in its memory, though implements a much different learning algorithm and overall architecture.

In the Cart Pole task, the agent receives "dense" rewards (i.e., at each time step), but in the Mountain Car task is only rewarded once the car reaches the goal position. This is a subtle yet consequential distinction between the two tasks, as "sparse" reward paradigms like the Mountain Car task may necessitate more exploration of their state space to find rewarding solutions.

Using our memory clearing approach therefore does not make much sense for tasks with sparse rewards, at least until a satisficing solution is found from which a policy may be optimized. Thus, to assess Minerva-Q's performance on this task, we initialized our agent's memory with four expert trials of the task, which provided enough rewarding experience to optimize further in unsupervised trials.

Properly integrating this expert knowledge required a change of the task reward values. By default, this task environment returns a reward of -1 at each time step (up to a maximum of 200 steps, terminating the task) and a reward of 0 upon reaching the goal position. However, since our ϵ-greedy selection targets the maximum Q-value, we found that the agent actively avoided taking the actions made by expert players, as the combination of sparseness and negative rewards disincentivized selecting these actions. To resolve this, we modified the rewards such that the agent received a maximum reward of 1 at the goal position, otherwise it would receive a reward of 0. This resulted in the expected behaviour of reproducing the expert knowledge. Thus, we may conclude our model exhibits human-like motor learning in its capacity to learn from limited examples, given an appropriate reward structure.

Discussion

Our results suggest that human-like perceptual-motor learning is possible to model with a minimal set of memory structures and functions, more specifically those of the Minerva family. As acknowledged, there are key differences in our implementation of Q-learning that necessitated an extension of the MINERVA 2 framework, though we hold that our solutions in Minerva-Q provide modest innovations that architectural purists are likely to find acceptable. More broadly, where other Minerva models were able to show successful empirical results for episodic and semantic memory tasks, we showed qualitative successes related to perceptual-motor memory tasks. This may have important theoretical implications to be explored in future works. However, we note some more immediate points for discussion.

First, we want to address the notion of optimization, which at least in the cognitive sciences is less of a priority compared to plausible satisficing strategies. Q-learning is typically implemented using a table of Q-values and in this form is guaranteed to converge on an optimal policy (Watkins, 1989; Watkins & Dayan 1992). More recently, Q-learning has been implemented in a deep Q-network (DQN; Mnih et al., 2015) that achieved human-level performance on 29 Atari games. In DQN, an approximation of the Q-learning policy is learned via iterated backpropagation. In response to stability and divergence issues in using nonlinear function approximators to derive Q-values (see e.g., Baird, 1995; Tsitsiklis & Van Roy, 1997), DQN employs "experience replay" and a novel loss function claimed to mitigate problematic correlations in training data that result in divergence from an optimal policy, though these are imperfect solutions.

There are important structural differences between Minerva and fully-connected artificial neural networks that suggest the divergence issues of DQN may not be relevant to Minerva-Q. According to Hintzman (1990), Minerva models can be understood as nonlinear localist neural networks. More specifically, Kelly et al. (2017) show that MINERVA 2 is equivalent to a distributed Hebbian associative memory. Conversely, neural networks like DQN utilize backpropagation via gradient descent on distributed representations and are more formally understood as universal function approximators given a sufficient number of hidden layers (Hornik et al., 1989). Though powerful, this kind of backpropagation is prone to issues like catastrophic interference and is largely responsible for divergence issues when approximating the Q-value function (Baird, 1995; Tsitsiklis & Van Roy, 1997).

Critically, Minerva-Q does not update its Q-values using an approximated function, rather the function is directly implemented to store new Q-values in memory. Despite this advantage over DQN, Minerva models are not equivalent to lookup tables, therefore we cannot assume without further investigation that Watkins and Dayan's (1992) convergence proofs apply to Minerva-Q in its current form. Nevertheless, it is encouraging that Minerva-Q appears to at least satisfice, and its connection to other Minerva models suggest it is a worthwhile effort of cognitive science to study questions of convergence in greater depth.

However, there appears to be, as a first approximation, an isomorphism between the Minerva family of memory models (including Minerva-Q) and the transformer class of neural network architectures. Specifically, we conjecture there are similarities between the so-called "attention" functions of transformer networks (see: Vaswani et al., 2017) and the Minerva activation and echo construction mechanisms. Though this notion is yet to be corroborated, if true, it may help ground currently state-of-the-art transformer models in more psychologically plausible models of memory.

Lastly, we note that the presented Minerva-Q model is subject to change, as it is under active development to accommodate theoretical considerations, incorporate more cognitively plausible mechanisms, and improve performance. For example, the current iteration is limited to discrete action spaces and thus cannot handle tasks requiring continuous-valued inputs. Likewise, there may be more plausible action-selection heuristics, such as ones motivated by optimism or surprise. Finally, we note that although our learning optimization approach (i.e., the deletion of suboptimal trials from memory) is not mandatory and has some conceptual precedence, it introduces structural changes that may be theoretically problematic. Thus, future iterations should explore solutions that achieve this behaviour in a more parsimonious and tenable manner.

References

Amin, S., Gomrokchi, M., Satija, H., Van Hoof, H., & Precup, D. (2021). *A survey of exploration methods in reinforcement learning.* arXiv. https://doi.org/10.48550/arXiv.2109.00157

Baird, L. (1995). Residual algorithms: Reinforcement learning with function approximation. *Machine Learning Proceedings 1995,* 30-37. https://doi.org/10.1016/B978-1-55860-377-6.50013-X

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., & Zaremba W. (2016). *OpenAI Gym.* arXiv. https://doi.org/10.48550/arXiv.1606.01540

Collins, R. N., Milliken, B., & Jamieson R. K. (2020). MINERVA-DE: An instance model of the deficient processing theory. *Journal of Memory and Language, 115*(2020:104151), 1-13. https://doi.org/10.1016/j.jml.2020.104151

Dougherty, M. R. P., Gettys, C. F., & Ogden, E. E. (1999). MINERVA-DM: A memory processes model for judgments of likelihood. *Psychological Review, 106*(1) 180-209. https://doi.org/10.1037/0033-295X.106.1.180

Gonzalez, C., Lerch, J. F., & Lebiere, C. (2003). Instance-based learning in dynamic decision-making. *Cognitive Science, 27*(2003), 591-635. https://doi.org/10.1016/S0364-0213(03)00031-4

Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks, 2*(5), 359-366. https://doi.org/10.1016/0893-6080(89)90020-8

Hintzman, D. L. (1984). MINERVA 2: A simulation model of human memory. *Behavior Research Methods, Instruments, & Computers, 16*(2), 96-101. https://doi.org/10.3758/BF03202365

Hintzman, D. L. (1986). "Schema abstraction" in a multiple-trace memory model. *Psychological Review, 93*(4), 411-428. https://doi.org/10.1037/0033-295X.93.4.411

Hintzman, D. L. (1990). Human learning and memory: Connections and dissociations. *Annual Review of Psychology, 41*(1990), 109-139. https://doi.org/10.1146/annurev.ps.41.020190.000545

Jamieson, R. K., Crump, M. J. C., & Hannah, S. D. (2012). An instance theory of associative learning. *Learning & Behavior, 40*(2012), 61-82. https://doi.org/10.3758/s13420-011-0046-2

Jamieson, R. K., Avery, J. E., Johns, B. T., & Jones, M. N. (2018). An instance theory of semantic memory. *Computational Brain & Behavior, 2018*(1), 119-136. https://doi.org/10.1007/s42113-018-0008-2

Jamieson, R. K., Johns, B. T., Vokey, J. R., & Jones, M. N. (2022). Instance theory as a domain-general framework for cognitive psychology. *Nature Reviews Psychology, 1*(2022), 174-183. https://doi.org/10.1038/s44159-022-00025-3

Johns, B. T., Jamieson, R. K., Crump, M. J. C., Jones, M. N., & Mewhort, D. J. K. (2016). The combinatorial power of experience. In Papafragou, A., Grodner, D., Mirman, D., & Trueswell, J.C. (Eds.) *Proceedings of the 38th Annual Conference of the Cognitive Science Society.* (pp. 1325-1330). Austin, TX: Cognitive Science Society.

Jones, M. N., & Mewhort, D. J. K. (2007). Representing word meaning and order information in a composite holographic lexicon. *Psychological Review, 114*(1), 1-37. https://doi.org/10.1037/0033-295X.114.1.1

Kelly, M. A., Mewhort, D. J. K., & West, R. L. (2017). The memory tesseract: Mathematical equivalence between composite and separate storage memory models. *Journal of Mathematical Psychology, 77*(April 2017), 142-155. https://doi.org/10.1016/j.jmp.2016.10.006

Komer, B., Stewart, T. C., Voelker, A. R., & Eliasmith, C. (2019). A neural representation of continuous space using fractional binding. *Proceedings of the 41st Annual Meeting of the Cognitive Science Society, 2019,* 2038-2043.

Moore, A. W. (1990). *Efficient memory-based learning for robot control.* [Ph.D. thesis]. University of Cambridge.

Mnih, V., Kavukcuoglu, K., Silver., D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglu, I., King, H., Kumaran, D., Wierstra, D., Legg, S., & Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature, 518,* 529-542. https://doi.org/10.1038/nature14236

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Kileen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., & Chintala, S. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. *33rd Conference on Neural Information Processing Systems, 721*(2019), 8026-8037.

Tsitsiklis, J. N., & Van Roy., B. (1997). An analysis of temporal-difference learning with function approximation. *IEEE Transactions on Automatic Control, 42*(5), 674-690. https://doi.org/10.1109/9.580874

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is all you need. *31st Conference on Neural Information Processing Systems (NIPS 2017)*, 1-11. https://doi.org/10.48550/arXiv.1706.0376

Watkins, C. J. C. H. (1989). *Learning from delayed rewards.* [Ph.D. thesis]. King's College.

Watkins, C. J. C. H, & Dayan, P. (1992). Technical note: Q-learning. *Machine Learning, 8*(1992), 279-292. https://doi.org/10.1023/A:1022676722315