nature machine intelligence

Article

https://doi.org/10.1038/s42256-025-01029-3

Compositional pretraining improves computational efficiency and matches animal behaviour on complex tasks

Received: 28 February 2024

Accepted: 28 March 2025

Published online: 19 May 2025

Check for updates

David Hocker[1], Christine M. Constantinople[1,3] & Cristina Savin[1,2,3]

Recurrent neural networks (RNNs) are ubiquitously used in neuroscience to capture both neural dynamics and behaviours of living systems. However, when it comes to complex cognitive tasks, training RNNs with traditional methods can prove difficult and fall short of capturing crucial aspects of animal behaviour. Here we propose a principled approach for identifying and incorporating compositional tasks as part of RNN training. Taking as the target a temporal wagering task previously studied in rats, we design a pretraining curriculum of simpler cognitive tasks that reflect relevant subcomputations, which we term 'kindergarten curriculum learning'. We show that this pretraining substantially improves learning efficacy and is critical for RNNs to adopt similar strategies as rats, including long-timescale inference of latent states, which conventional pretraining approaches fail to capture. Mechanistically, our pretraining supports the development of slow dynamical systems features needed for implementing both inference and value-based decision making. Overall, our approach helps endow RNNs with relevant inductive biases, which is important when modelling complex behaviours that rely on multiple cognitive functions.

Recurrent neural networks (RNNs) are frequently used in neuroscience to study the role of coordinated network activity in supporting trained behaviours, with the hope of generating hypotheses about neural mechanisms that can be tested in animal models[1]. As cognitive tasks increase in complexity, training RNNs to mimic animals becomes increasingly difficult. First, RNNs may fail to learn the task, especially in reinforcement learning settings with salient suboptimal solutions, sparse rewards or long temporal dependencies[2]. Second, the solutions found by RNNs trained on complex tasks can fail to reflect the computational strategies in animals, limiting their utility as models for neuroscience.

Pretraining can boost learning by slowly increasing task complexity along a particular dimension (for example, gradually increasing the delay in a task requiring working memory) as a form of curriculum learning[3,4] or by using families of related tasks and harnessing shared structure, as in meta-learning[5–7]. In the animal domain, pretraining takes the form of behavioural shaping[8], which gradually exposes the animal to the functional subelements of a task, such as interacting with effectors such as levers. As with artificial agent pretraining, complexity is gradually increased by incorporating additional elements and the animal learns how to combine them in an appropriate way (for example, the lever needs to be pressed only when light is on). This approach can dramatically increase learning efficiency in animals but is rarely used in RNN training[9]. Another largely ignored form of pretraining is the experience that the animals bring with them to the experiment in the form of inductive biases. The subjects learn many cognitive tasks throughout their lifetime, and previously learned tasks can impact their behaviour in new ones[10–12]. Many factors including training history[13], early experiences[14–17] or inductive biases established by evolution[18] can influence learning outcomes and corresponding behavioural strategies.

We hypothesize that adopting pretraining procedures that better reflect the animal's experience will allow RNNs to learn more robustly

[1]Center for Neural Science, New York University, New York, NY, USA. [2]Center for Data Science, New York University, New York, NY, USA. [3]These authors jointly supervised this work: Christine M. Constantinople and Cristina Savin. e-mail: cs5360@nyu.edu

https://doi.org/10.1038/s42256-025-01029-3

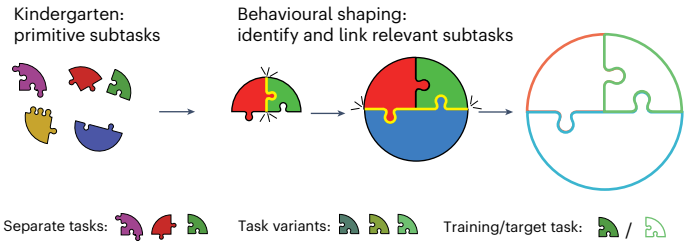

Fig. 1 | Modelling the animal's learning experience. Right: the target behaviour or task (open circle) can be thought of as composed of several computational subelements, some of which may need to be computed in parallel. Left: these subcomputations involve primitive skills that in the animals have probably been learned through past experience. Behavioural shaping guides learning by incrementally combining basic skills into more complex functions (linked puzzle pieces). For RNN training, we operationalize this idea by first training a set of fundamental subcomputations in a 'kindergarten' training phase, followed by behavioural shaping that mimics animal training.

and better match animal behaviour. To test this, we created a pretraining paradigm that explicitly trains a set of useful, basic computational skills (working memory, sensory evidence integration and so on) outside of the target task context (using simple tasks with separate loss functions). These skills are then combined via animal-matched behavioural shaping towards a target end goal (Fig. 1). We term this form of pretraining as 'kindergarten curriculum learning' (kCL). Our approach is similar to traditional curriculum learning in that the difficulty of the task increases over time and to meta-learning in that the curriculum tasks are assumed to share underlying computational structure with the target. It is unique in its decomposition of complex tasks into elements that need to be computed in parallel, pretrainable with simpler means.

We test our approach using a complex cognitive task previously performed in rats[19]. A simplified formalization of the task as a Markov decision process (MDP) allows us to identify key qualitative features of optimal behaviour in the task and the subtasks to include in the kindergarten curriculum. kCL substantially improves learning speed and performance compared with brute-force in-task training with or without behavioural shaping. Importantly, its computational strategies closely match animal behaviour, while other curricula's do not. While increasing the number of relevant kindergarten subtasks increases both performance and the strength of this match, good performance and adopting animal-like strategies are dissociable, further demonstrating the importance of good inductive biases in mimicking rats behaviour. Mechanistically, we find a diversity in the types of dynamical system features present in well-trained networks (that is, point attractors, saddles and line attractors), though typically well-performing networks possess attractor dynamics for latent states with strong beliefs and saddles for uncertain states with semistable beliefs. This solution is qualitatively distinct from high-performing networks not trained with kCL. Moreover, kCL yields networks with richer dynamics, which can be exploited for learning the target task.

Results

Behaving animals use inference in a temporal wagering task

To study the effects of curriculum learning on behaviour and neural activity, we sought a task that is difficult for RNNs to learn from experience alone. Specifically, we investigated a temporal wagering task performed by rats[19], where a simple, suboptimal strategy leads to reasonable rewards, but optimal strategies require inference of latent states over long timescales (Fig. 2a). Rats initiate trials by poking into a centre port, then they hear an auditory tone, the frequency of which denotes the volume of a water reward. A side port is randomly chosen as the rewarded side on each trial, indicated by light. The probabilistic reward ($p = 0.8$) arrives with a time delay drawn independently from an exponential distribution $p(t) = \lambda^{-1}\exp(-t/\lambda)$, where mean λ is 2.5 s. If the rat chooses the other port before reward delivery, the trial is terminated (an 'opt-out' trial), and a new trial can begin. The reward is withheld on a subset of trials ($p = 0.2$), termed 'catch trials', as a readout of how long rats are willing to wait before exercising the opt-out choice. There are five reward offers as well as a hidden block structure that is not cued to the rats (Fig. 2b). Each session starts with a mixed block (all offers), randomly followed by either a high (top three offers) or low block (bottom three offers) and then deterministically alternating between mixed and high/low blocks over the session. Importantly, 20 µl offers are present in all blocks, allowing a direct measure of context dependence.

Rat wait times during catch trials are sensitive to both the size of the reward offer and the latent block structure (Fig. 2c). Animals wait longer for larger offers and modulate their behaviour on the basis of the block. In addition, the difference in wait time for 20 µl offers when comparing low versus high blocks with mixed blocks is asymmetric (larger shifts for low blocks). To quantify behavioural sensitivity to the blocks, we calculated the ratio of wait times for 20 µl offers in high versus low blocks. Across a large population ($N = 291$ animals), rats modulate their wait times based on the block context (Fig. 2d). This state-dependent behaviour relies on an inferential strategy, in which rats infer the most likely state based on the current offer value and their prior beliefs about the current reward block[19,20], which has distinct behavioural signatures in particular after block transitions (Fig. 2e). Indeed, regressing wait times against previous reward and block over training shows that the block regressor increases with the same time course as the behavioural sensitivity to blocks, suggesting that contextual sensitivity reflects learned knowledge of block structure (Fig. 2f). Modeling of this inferential strategy predicts the probability of being in each block on a trial-by-trial basis (Fig. 2g).

Optimal task behaviour

Many rats exhibit linear sensitivity to log-reward offers and block-dependent responses, which we seek to understand from an optimal decision making perspective. While several formalizations are possible, we chose to describe the task as an MDP, in which agents must either wait for a reward or opt out of the trial (Fig. 2h), similar to ref. 21. In this formulation, the agents decide between 'wait' and 'opt-out' actions. Waiting either preserves the state or transitions to a terminal 'hit' state where reward is delivered, with probability given by the exponential delay distribution, $p(t)$, and overall reward probability, p_r. Opting out deterministically ends the trial with a reward penalty R_{oo}, while the wait action has a time penalty k. This MDP formulation treats the offered reward and time within trial as the state, which allows for a simple closed-form solution describing optimal behaviour for an agent with perfect knowledge of the task statistics, where the optimal wait time t^* is log linear in the reward offer R,

$$t^* = \lambda\left(\log[Rp_r] - \log[k\lambda]\right). \quad (1)$$

In foraging theory, where agents aim to maximize their reward rates, time penalties are often related to the opportunity cost or the average reward rate of the environment; thus, the wait-time decisions should incorporate the value of potential rewards missed out on by continuing to wait[22,23]. In our MDP formulation, this implies

$$k = \rho\Delta t = \mathbb{E}\left[\tilde{R}^{(B)}\right], \quad (2)$$

where ρ is the reward rate, Δt the time step size, $\tilde{R}^{(B)}$ the reward for a given block B and the expectation is taken over the reward probability for an offer in a specific environment, B. This approach identifies qualitative features of optimal behaviours in this task: linear

https://doi.org/10.1038/s42256-025-01029-3

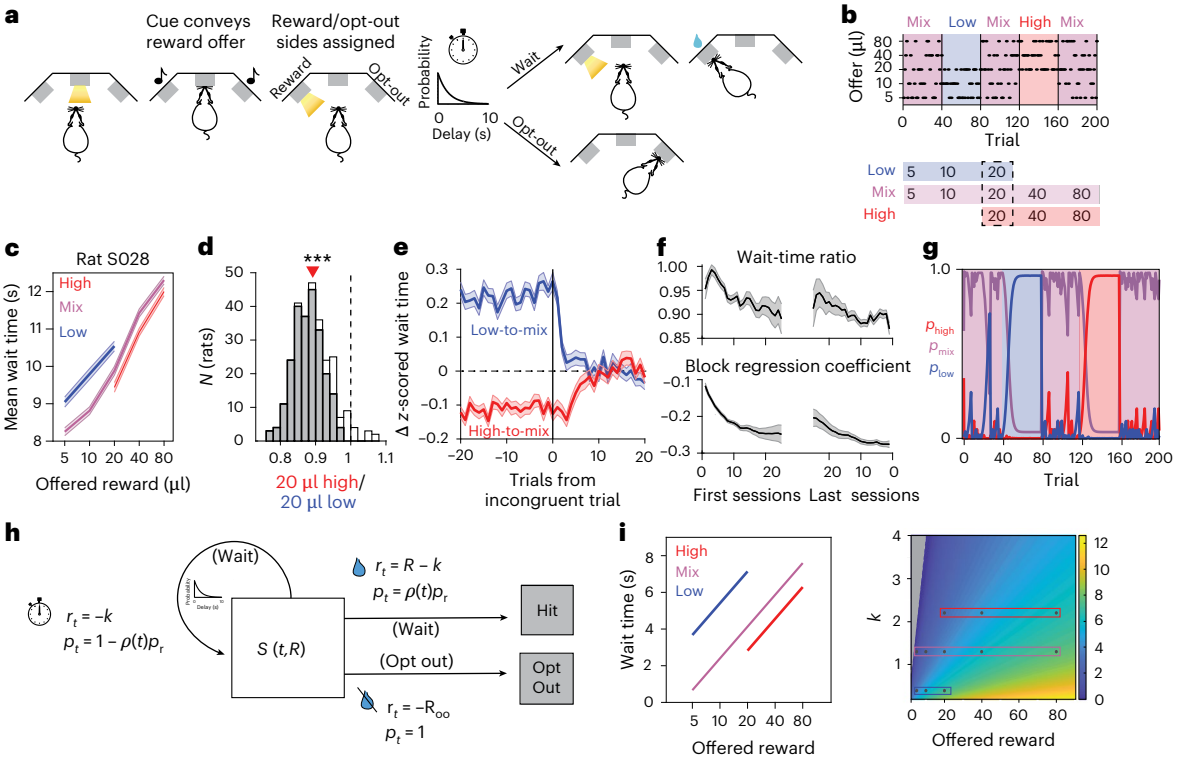

Fig. 2 | Rodent behaviour in temporal wagering task. a, The temporal wagering task. The rats wait for known reward offers conveyed by an auditory tone but with an uncertain reward delay. The rats may choose to wait for reward at illuminated side port until water arrives or poke into the other port to terminate the trial. **b**, The latent block structure of task that is uncued to rats. After every 40 successful trials, a transition between mixed blocks (purple) containing all potential reward offers and blocks with a subset of the highest (red) or lowest (low) offers. **c**, The example mean wait time behaviour on catch trials, where the reward is withheld for a single rat. The error bars denote the standard error of the mean (s.e.m.) over trials. **d**, The distribution of wait time ratios of high versus low for 20 µl offers on catch trials. ***$P = 8.26 \times 10^{-49}$, two-sided Wilcoxon signed-rank test. The histogram includes rats with significantly different wait-time ratios (shaded, $P < 0.05$, rank-sum test) and insignificant wait-time ratios (unshaded, $P > 0.05$, rank-sum test). **e**, The trial-by-trial changes in wait times across a block transition, averaged over rats; the error bars mark the s.e.m., and the colours reflect transitions from either a high (red) or low (block) block into a mix block, aligned to first incongruent trial that violates possible offers in the current block (for example, a 40-µl offer after being in a low block). **f**, The time course of the mean wait-time ratio (top) and mean regression coefficient for block in a linear regression of wait time (bottom) for catch trials. Left: the first 20 sessions after the block structure is introduced in the task. Right: the last 20 sessions of training. **g**, The predicted block probabilities of a Bayes-optimal model of the task. The background colours denote the true block. **h**, The MDP formalization of the task, showing states (boxes), state-transfer probabilities p and rewards r for each state–action pair. The rewards arrive within each trial on the basis of the delay distribution $\rho(t)$, incur a time penalty k at each time step, as well as penalty R_{oo} for opting out. **i**, Left: the MDP-defined optimal wait times for three different wait time penalties. Right: the wait times for all wait time penalties. The coloured boxes denote k for optimal wait times in the left plot. The grey region denotes the negative wait time. In **d**–**f**, the mean is over rats ($N = 291$), and the error bars in **e**–**f** mark the s.e.m.

sensitivity to log-offer, with a bias to wait longer in reward-sparse environments (Fig. 2i), an asymmetry in the sensitivity of wait times to high versus low blocks, as well as faster wait times with lower reward probabilities (see Supplementary Information for derivations and further details). Importantly, rats display all of these hallmarks (Fig. 2c,d).

Deep meta-reinforcement learning agents mimic rat behaviour

To study the impact of training on behavioural performance, we used deep RNNs trained with meta-reinforcement learning (Fig. 3a, long short-term memory (LSTM) architecture, all-to-all connected within each module), which allows for dynamic behavioural adaptation and has provided useful neural signatures in other tasks[24]. Our version includes a modular architecture, with one module implementing latent state inference ('inference'), analogous to regions such as the orbitofrontal cortex[25–27], and a second module responsible for action selection ('policy'), loosely analogous to the striatum[28].

We trained RNN agents on a simplified variant of the temporal wagering task (Fig. 3b). At every timepoint ($\Delta t = 50$ ms), the agent decides on the basis of its stochastic decision policy to either 'wait' or 'opt out' (abstracting away the trial by trial alternation in left versus right reward ports), with stochastic intertrial intervals out of the control of the agent. These changes preserve the core decision making process and cleanly map to the MDP, while simplifying the RNN architecture.

We used the MDP-derived expression for the Q value for waiting during the trial, to identify candidates for the kindergarten curriculum:

$$Q^*(\text{wait}, t) = p_r R e^{-t/\lambda} - k\lambda\left(1 + \log\left(\frac{p_r R}{k\lambda}\right) - \frac{t}{\lambda}\right) - R_{oo}. \quad (3)$$

In particular, optimal behaviour requires knowledge of three key quantities (R, t, k), each of which point to a form of basic skill training. First, a persistent representation of R throughout the trial can be encouraged by working memory training (report a transient input seen many time steps earlier). Second, knowledge of elapsed time within a trial, t, can be developed using a temporal counting task (output a ramp with slope 1). Finally, knowledge of the opportunity cost can come via either past reward averaging, trained as an integrator of stimulus input or through state inference trained via a classification task that reports latent states based solely on stimulus information. We trained each of these through separate learning objectives (Supplementary Fig. 1).

https://doi.org/10.1038/s42256-025-01029-3

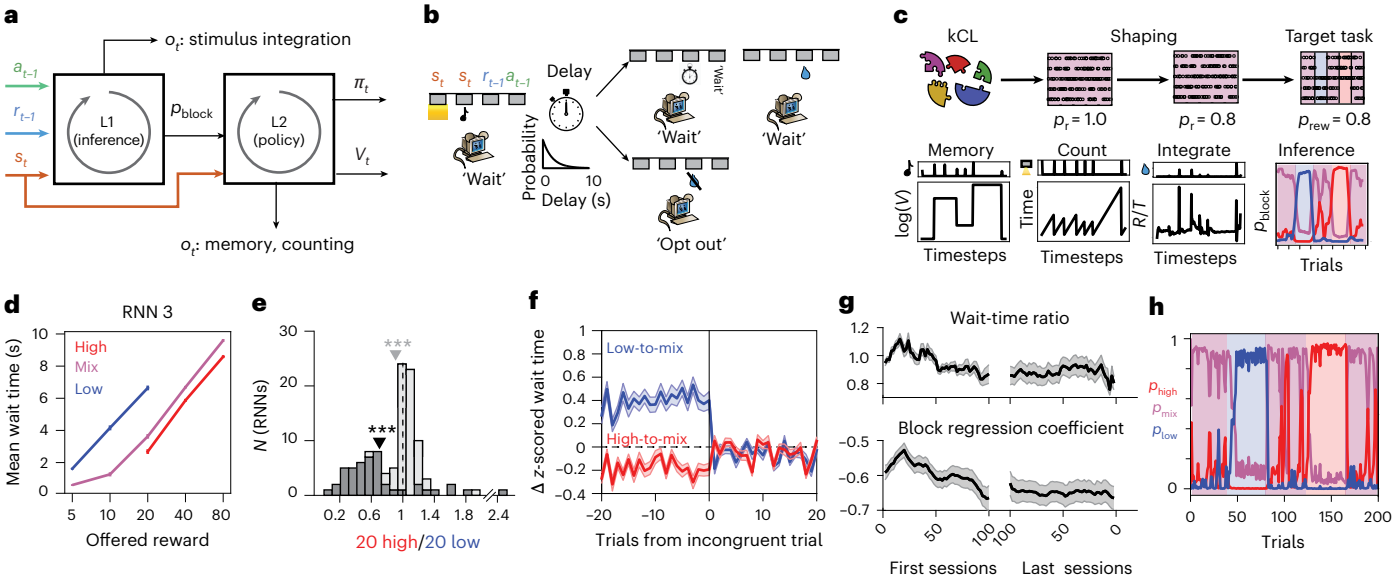

Fig. 3 | Deep meta-RL agents can learn the temporal wagering task. **a**, A schematic of deep meta-RL architecture. An LSTM-based RNN contains two modules, an inference layer that predicts latent task states and a policy layer that outputs decisions to wait or opt out of trials. The inputs include trial stimulus information and previous reward and action history (used in meta-RL to permit trial-by-trial learning). The policy module outputs the decision policy, π_t, and value function, V_t. The additional outputs o_t for kindergarten tasks are used for pretraining and in-task regularization. Moreover, p_{block} outputs from the inference module to the policy module are trained to infer the current block. **b**, The simplified temporal wagering tasks for RNN agents. **c**, The RNN training protocol, referred to as kindergarten + shaping curriculum learning (CL). Top: kindergarten supervised learning tasks outside the context of the temporal wagering task are performed first, followed by progressively harder variants of the target wait-time task that mimics the shaping procedure performed in rats. Bottom: the normative set of kindergarten tasks suggested from MDP solution to the temporal wagering task (equation (3)): working memory, maintaining an internal estimate of time, integrating input stimuli and inferring the latent state. **d**, The example mean wait time performance of an RNN agent. The error bars denote the standard error of the mean (s.e.m.) over trials (N = 10,000 trials). **e**, Wait-time ratio of population of N = 113 RNNs. The dark bars denote N = 46 RNNs trained with kindergarten + shaping CL, and the light bars denote other networks with simpler CL strategies (that is, shaping-only CL and target-task CL (see 'kCL supports near optimal task performance'). The dark grey arrow is the mean of kindergarten + shaping CL population, and the light grey arrow is the mean of entire population. ***$P < 0.001$. Kindergarten + shaping: $P = 1 \times 10^{-9}$; all RNNs: $P = 3 \times 10^{-20}$, two-sided Wilcoxon signed-rank test. **f**, The RNN wait-time dynamics, as in Fig. 2e. **g**, The wait-time ratio (top) and regression coefficient for a block in a regression of wait time (bottom) over training (Methods). In **f**–**g**, the results are the average over RNNs trained with kindergarten + shaping CL (N = 46), and the error bars are the s.e.m over networks. **h**, p_{block} during the temporal wagering task for network in **d**. The same colour convention as Fig. 2g.

After kindergarten (modelling pre-existing cognitive capacities that the animals bring with them to the experiment), we used the behavioural shaping protocol employed in rats: (1) an easy version of the wait-time task with deterministic rewards and no block structure first, followed by (2) introduction of stochastic rewards and finally (3) introduction of the blocks in the target wait-time task. This 'shaping' phase uses an adapted version of the meta-reinforcement learning loss from[24], with auxiliary losses corresponding to the kindergarten tasks (Fig. 3c, top, and Methods).

RNNs undergoing the complete training ('kindergarten+shaping' or 'full kindergarten') successfully learned to perform the temporal wagering task, exhibiting behavioural wait times on catch trials that resembled rat behaviour: the agents waited longer for larger offers, with wait times sensitive to the hidden block structure (Fig. 3d), with accurate block inference (Figs. 2g and 3h). These RNNs had strong block sensitivity across the population, as quantified by the wait-time ratio, whereas RNNs without full training generally failed to become sensitive to the blocks or did so in a manner inconsistent with rat behaviour (Fig. 3e). Finally, when comparing the time course of the wait time ratio and block sensitivity over training, we found that RNNs displayed similar learning dynamics to rats (Fig. 3g). The RNNs start with no block sensitivity, but over training wait times for 20 µl, offers become different across blocks, suggesting that knowledge of block structure is driving the difference in wait times, as in the rats (Fig. 2f). In summary, deep meta-reinforcement learning agents trained with kindergarten tasks and animal-like behavioural shaping display qualitative behaviour similar to rats, have similar inferential strategies and learn to become sensitive to latent structure that varies on long timescales in the same manner as animals.

kCL supports near optimal task performance

To determine the extent to which a structured curriculum was necessary for effective learning in the task, we investigated a family of curriculum learning sequences, varying the tasks in the kindergarten curriculum and their hyperparameters, with or without behavioural shaping. We first compared the performance of networks trained with both kindergarten and shaping to agents that experience the shaping procedure alone (Fig. 4a, orange) and directly training on the target task without any pretraining (Fig. 4a, red). In both cases, we allowed each procedure to undergo at least the same amount and often much longer total training (number of epochs) as kCL. We found that including kindergarten training outperformed both these alternatives (Fig. 4b–c) and lead to systematic differences in network behaviour (Fig. 4d). When training on the target task alone, networks adopted a simple, suboptimal strategy, where agents simply wait until a timeout penalty occurs, leading to a low sensitivity to both reward offer and context and opt-out rates below the true catch probability of the task (Supplementary Fig. 2a). Introducing shaping can lead to some mild sensitivity to the reward offer, though not nearly as strong as when also including kindergarten tasks. In addition, these agents did not learn to opt out as frequently. Importantly, the wait-time ratio in the shaping-only networks was reversed when compared with rats and kindergarten + shaping curriculum learning, meaning that these networks wait longer for 20 µl in high blocks, which is suboptimal according to the MDP. Overall, including kCL with shaping

https://doi.org/10.1038/s42256-025-01029-3

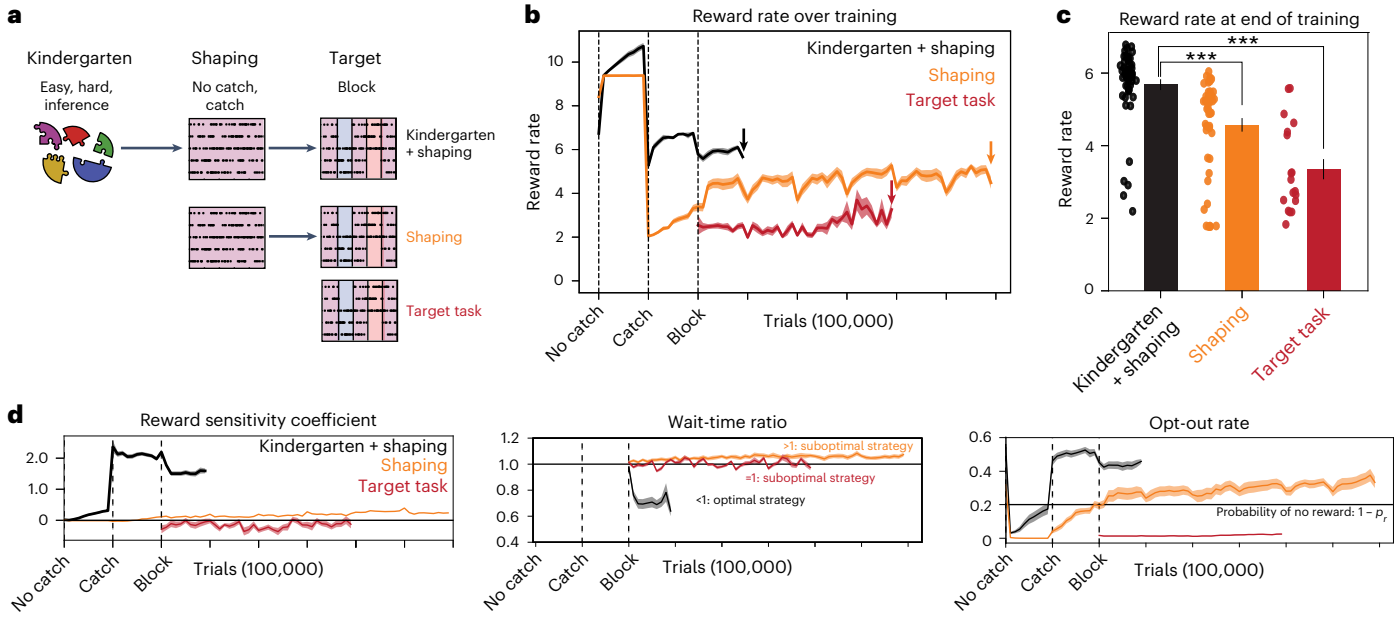

Fig. 4 | Performance of pretraining methods on the temporal wagering task. a, A schematic of pretraining methods, separated by kindergarten and shaping phases. The 'kindergarten + shaping' curriculum first trains on out-of-context supervised learning tasks (that is, kindergarten tasks), followed by the final stages of behavioural shaping that occur in rat training: first training on deterministic rewards in mixed blocks only ('no-catch' training), then mixed-block trials with stochastic rewards ('catch' training). Finally, the blocks are introduced ('block' training) (Supplementary Fig. 1). The 'shaping' curriculum omits kindergarten tasks and 'target task' only trains on the target task with the block structure. **b**, The average reward rate over training for each method. **c**, The reward at the end of the training stage, corresponding to the coloured arrows in **b**. The coloured dots denote the reward rate from each RNN (***kindergarten + shaping versus shaping: $P = 1 \times 10^{-7}$; kindergarten + shaping versus no shaping: $P = 5 \times 10^{-9}$, two-sided rank-sum test). **d**, The evolution of different behavioural features over training. Left: the reward sensitivity coefficient is the slope of a linear fit to the wait time versus the log of the reward offer. Middle: the wait-time ratio compares the average wait times for a 20 µl offer in high versus low blocks, optimally <1 under the MDP formulation. Right: the opt-out rates reflect the proportion of trials that an agent choses the opt-out option, compared with the catch probability (black line). For each inset in **d**, the *t*-tests at the end of training comparing kindergarten + shaping to other curriculum learning sequences were significantly different ($P << 0.001$). Shaping: reward sensitivity: $P = 3 \times 10^{-25}$; wait-time ratio: $P = 3 \times 10^{-16}$; opt-out rate: $P = 9 \times 10^{-4}$. Target task: reward sensitivity: $P = 1 \times 10^{-15}$; wait-time ratio: $P = 1 \times 10^{-5}$; opt-out rate: $P = 5 \times 10^{-27}$). **b**–**d** show the mean responses, with the shading denoting the standard error of the mean over networks (kindergarten + shaping: $N = 45$; shaping: $N = 45$; target task: $N = 18$).

is important to replicate qualitative patterns of behaviour that are consistent with the optimal strategy and rats.

The structure of the kindergarten curriculum matters

Having confirmed that the MDP-derived kindergarten curriculum is effective, we can now ask whether pretraining on all of the tasks was actually required or if training on simpler curricula would have worked just as well. As the space of options is combinatorially large, we chose to focus on the nature and number of tasks included in the kindergarten curriculum, with considerations such as task ordering and hyperparameter choices briefly explored in Supplementary Fig. 2. In all cases, we trained the networks for at least as much as the full set of four kindergarten tasks, to balance the total amount of parameter updating that could be performed. First, we asked if individual kindergarten tasks can achieve high-quality solutions on their own when paired with shaping. We found that single task kindergarten generally underperformed in terms of final reward rates compared to 'full kindergarten' (Fig. 5a, top) and also failed to capture its qualitative behaviour (Fig. 5a, bottom). A noteworthy exception was the memory task, which achieved equivalent reward rates. Behaviourally, the working-memory-only solution had comparable reward sensitivity and opt-out rates but showed significantly weaker context sensitivity, pointing to potential differences in the computational strategy adopted to solve the task, even among high-performing solutions. By contrast, the inference task allowed for good context sensitivity and weaker offer sensitivity.

We reasoned that adding more kindergarten tasks might have additive effects for behavioural matching and so we studied the effect of having two or three tasks as part of kCL (Fig. 5b). In particular, since the memory-only curriculum yielded offer sensitivity, and inference-only curriculum yielded good context sensitivity, we hypothesized that using them together might result in the desired task behaviour and high reward rates. We used these two tasks as the base pair, then optionally added a third task. All combinations retained high reward rates, but behaviourally, the solutions mapped to distinct strategies, with weaker reward sensitivity and a broad range of context sensitivity (including the suboptimal reverse context sensitivity, Fig. 2d). Adding in a third task got closer but could not fully recapitulate the behaviour of full kCL. Collectively, these results demonstrate that the content of the curriculum can measurably affect behavioural strategy, despite comparable task performance, and that the richer the (relevant) subcomputations, the closer the trained solution is to animal behaviour.

Would pretraining on tasks not part of the MDP solution also work by virtue of introducing task variability in training? To test this, we used a delay-to-non-match task[29,30] (Fig. 5c), which requires an agent to report if two presented stimuli that are lagged in time are the same (a match) or different (a non-match). We chose this task because it feasibly replaces the classification and working memory components learned in kCL while the essential inference component is missing. We studied two versions of such pretraining: using delay-to-non-match alone or additionally including the counting and stimulus integration tasks. Both underperformed compared with the original kCL and also displayed quite different behaviour.

Since kindergarten tasks play dual roles in training, we asked whether the benefits of kCL came from pretraining or in-task regularization (Fig. 5d). Specifically, we used the full set of kindergarten tasks but only as regularizers during in-task training (no pretraining

https://doi.org/10.1038/s42256-025-01029-3

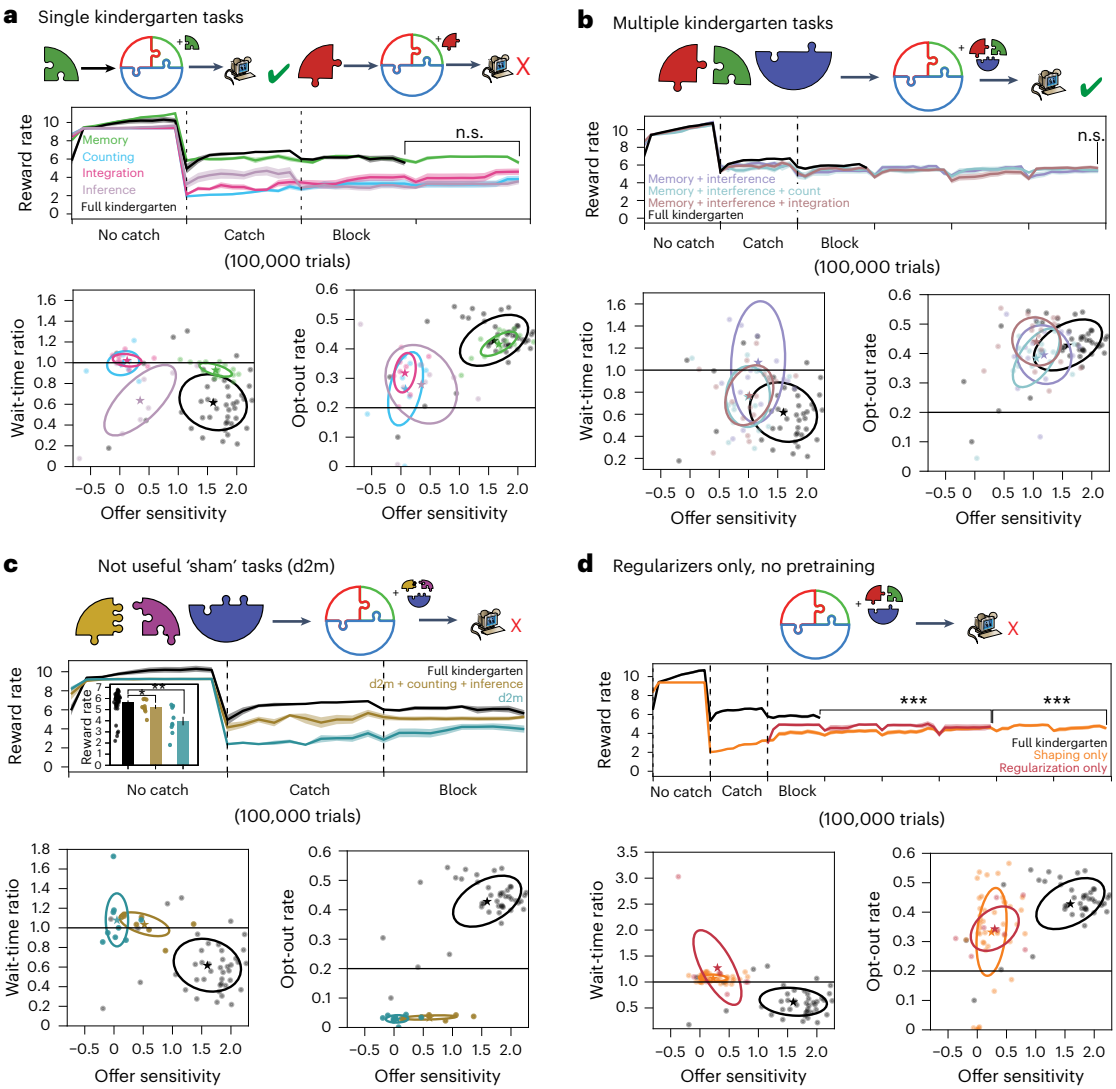

Fig. 5 | Variations of kCL. a, Single kindergarten tasks. Memory ($N = 18$, green; reward rate: $P = 0.33$; reward sensitivity: $P = 0.53$; wait-time ratio: $P = 1 \times 10^{-4}$; opt-out rate: $P = 0.44$). Counting ($N = 9$, blue; reward rate: $P = 5 \times 10^{-5}$; reward sensitivity: $P = 4 \times 10^{-10}$; wait-time ratio: $P = 3 \times 10^{-4}$; opt-out rate: $P = 3 \times 10^{-5}$). Stimulus integration ($N = 9$, pink; reward rate: $P = 0.002$; reward sensitivity: $P = 6 \times 10^{-10}$; wait-time ratio: $P = 2 \times 10^{-4}$; opt-out rate: $P = 0.002$). Inference ($N = 9$, purple; reward rate: $P = 5 \times 10^{-5}$; reward sensitivity: $P = 2 \times 10^{-7}$; wait-time ratio: $P = 0.55$; opt-out rate: $P = 7 \times 10^{-5}$). n.s., not significant. **b**, Multiple kindergarten tasks. Memory + inference ($N = 18$, purple; reward rate: $P = 0.15$; reward sensitivity: $P = 0.006$; wait-time ratio: $P = 1 \times 10^{-4}$; opt-out rate: $P = 0.19$). Memory + inference + counting ($N = 18$, blue; reward rate: $P = 0.09$; reward sensitivity: $P = 3 \times 10^{-4}$; wait-time ratio: $P = 0.08$; opt-out rate: $P = 0.17$). Memory + inference + integration ($N = 18$, red; reward rate: $P = 0.64$; reward sensitivity: $P = 4 \times 10^{-4}$; wait-time ratio: $P = 0.08$; opt-out rate: $P = 0.57$). **c**, A computationally irrelevant delay-to-non-match task (d2m). d2m ($N = 9$, yellow; reward rate: $P = 2 \times 10^{-4}$; reward sensitivity: $P = 1 \times 10^{-10}$; wait-time ratio: $P = 7 \times 10^{-5}$; opt-out rate: $P = 4 \times 10^{-18}$). d2m + counting + inference ($N = 9$, turquoise; reward rate: $P = 0.009$; reward sensitivity: $P = 6 \times 10^{-6}$; wait-time ratio: $P = 9 \times 10^{-5}$; opt-out rate: $P = 7 \times 10^{-18}$). **d**, Regularization-only training performs multitask learning with kindergarten tasks, without any curriculum learning (CL) pretraining or shaping ($N = 9$, red; reward rate: $P = 0.001$; reward sensitivity: $P = 3 \times 10^{-8}$; wait-time ratio: $P = 7 \times 10^{-5}$; opt-out rate: $P = 0.02$). **a**–**d** shows the set of results when altering the curriculum in some form. Top: schematic of the manipulation. Large puzzle pieces denote a kindergarten task, superscript version used for regularization. Middle: reward rate over training (mean and standard error of the mean over networks). Bottom: the distribution of behavioural metrics for each trained RNN. The stars mark the population mean and the ellipses are two standard deviations. All statistics compare full kindergarten, aka kindergarten + shaping ($N = 45$), versus other CL type (two-sided rank-sum test for reward rates, two-sided t-tests for the rest).

or shaping). Such regularization proved insufficient, and it could not achieve high-quality solutions nor could it recreate the behaviours seen in the animals, resembling solutions obtained from behavioural shaping alone (Supplementary Fig. 3). We briefly studied a few additional dimensions of the curriculum structure. Manipulations of the timescale of the inference task affected behaviour, with short inference timescales during pretraining failing to generate strong context sensitivity (Supplementary Fig. 3b–e). This suggests that its value to the curriculum may (at least partly) stem from introducing slow modes in the RNN dynamics. The location of the harder-to-train inference task in the sequence of subtasks also influenced the quality of the solution, consistent with the common prescription of learning simple tasks first. Finally, we tried decreasing the hyperparameters' similarity between kindergarten tasks and target, while preserving the nature of the computation (different offers for memory, a larger number of latent states for inference) (Supplementary Fig. 2f). This discrepancy in the task details also meant incorporating a separate output for inference, rather than tying it to the inference-to-policy module output. This kCL variant still leads to good context and wait-time sensitivity, although its solutions had less linear offer sensitivity (Supplementary Fig. 2f–j). Overall,

https://doi.org/10.1038/s42256-025-01029-3

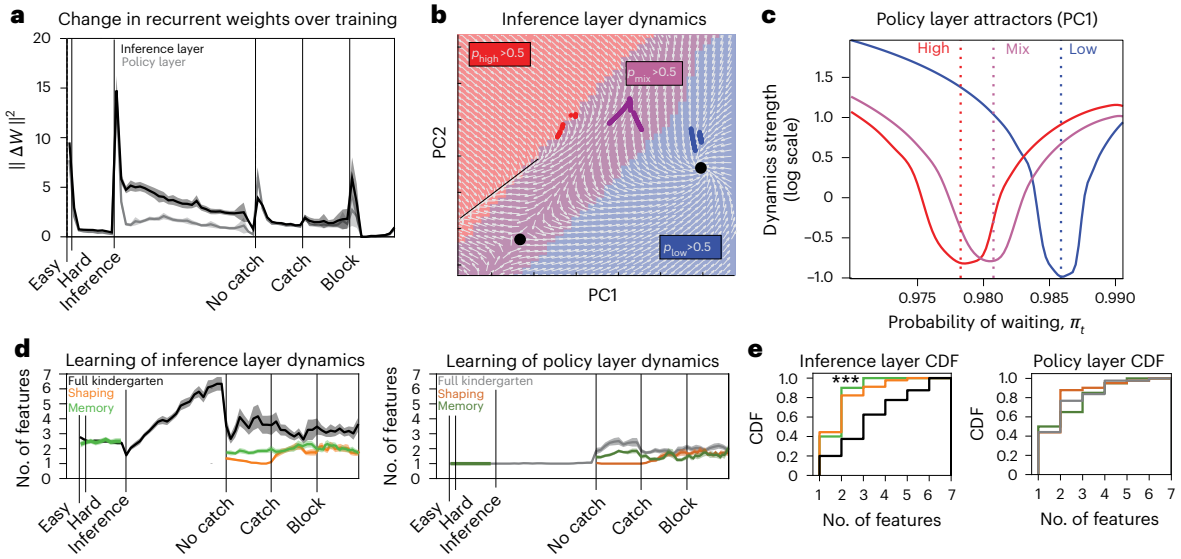

Fig. 6 | Dynamical systems features of kCL-trained networks. a, Change in the LSTM recurrent hidden-layer weights over training for kindergarten + shaping curriculum learning (CL) (Methods). **b**, Dynamical gradient flow field for the inference layer of a sample network. The background colour denotes region in which p_{block} was >50% for that respective block (same colour convention as in Figs. 2 and 3). The coloured curves denote the evolution of single-trial 20 µl offers from different blocks in which the agent ultimately opted out. The dynamical features are denoted in black (the line attractor in the high block, the saddle in the mix block and the point attractor in the low block). **c**, The dynamics of the policy layer along PC1 but isolated to the timepoint just before opting out for single trials from each block. The PC activity was labelled with π_t on the x axis, and the dynamics kinetic energy is plotted on a log scale on the y axis. **d**, The average number of dynamical systems features over training. The shaded area shows the standard error of the mean over networks. **e**, The cumulative distribution function (CDF) of the number of dynamical systems features at the end of training. A two-sided discrete Kolmogorov–Smirnov (KS) test compared the kCL distribution at end of training to other CL distributions[49]. ***$P < 0.001$. Shaping inference layer: $P = 1 \times 10^{-4}$, shaping policy layer: $P = 0.56$. Memory CL inference layer: $P = 3 \times 10^{-4}$, memory CL policy layer: $P = 0.57$. For kindergarten + shaping, $N = 45$, for shaping, $N = 45$ and for memory, $N = 18$.

these results suggest that the best learning is achieved by pretraining on kindergarten tasks that match the computational subelements of the target task and that a precise statistical match between kindergarten and target tasks is not strictly needed to see benefits from kCL.

kCL leads to distinct neural dynamics

What is the mechanism by which kCL helps target task training? Using the overall magnitude of weight changes as a coarse measure of network learning (Fig. 6a), we found network reorganization at the beginning of each training phase but most substantially when learning to perform inference of latent states. To determine how the network dynamics change in more functional terms across the training stages, we focused on the dynamical systems structure of the solution, determined by 'slow points'. Such slow dynamical systems features (for example, point attractors, saddles and line attractors) provide a natural mechanism for maintenance and integration of information at long timescales[31], which is critical for our task (reward offer within trials and opportunity cost across trials). Thus, identifying the computational strategy used by an RNN to solve our task translates to determining the number, type and geometry of its slow points[32].

First, we investigated maintenance and updating of latent beliefs in the inference layer of kCL-trained networks (Methods) and found that RNNs trained by kCL generally displayed a common motif involving three slow features (Fig. 6b): two stable features (fixed point or line attractor) and a semistable feature (saddle) separating them. Among networks that learned the task in a rat-like way, a large proportion (20 of 29 RNNs) displayed this common motif. Conversely, the networks that did not learn a rat-like strategy did not display the same motif (14 of 16). Partitioning the state space by most likely block according to the RNN's beliefs (the inference layer's main output), we find three contiguous subregions, with a slow feature in each (other rare motifs that still provide good wait-time behaviour are documented in Supplementary Fig. 4). The beliefs are updated by inputs from offers and rewards by pushing the network state along the direction orthogonal to the block boundaries (Supplementary Fig. 5). Importantly, stable features lie in areas of state space corresponding to low and high blocks, whereas the semistable feature is consistently located in the mixed-block region. The line or point attractors were equally likely in high blocks, but we invariably found a point attractor in the low-block region. This overall dynamical systems structure was specific to networks trained by kCL and almost never seen with shaping alone (2 of 47) or other curricula (Supplementary Fig. 6a,b). Importantly, the three-feature motif was not observed in RNNs with memory-only pretraining, (Supplementary Fig. 6c,d), despite the match in performance. This confirms the notion that different training procedures bias learned solutions towards different computational strategies for performing the task. It also generates testable predictions about the neural activity of brain regions thought to subserve inference (for example, orbitofrontal cortex).

The policy layer dynamics were also variable across RNNs but to a lesser degree than the inference layer. We often found one to two slow points, one of which was almost always a point attractor (41 of 45) and the sole to be functionally relevant. When visualizing the location of the fixed-point attractor along the first principal component (PC) of policy layer activity (Fig. 6c), we found that the location of the attractor was block specific, yielding varying wait-time probabilities, and that their relative location supported asymmetric wait time behaviour over blocks (Fig. 3d). Anecdotally, attractors did not change position across blocks in other curricula, which makes another testable prediction about a block-dependent shift of attractors within the manifold of population activity in the striatum.

Since the structure of slow points reflects key aspects of task computation, such as beliefs about blocks and associated wait times, we wondered how these features may be constructed over the course of training, in particular during kindergarten (Fig. 6d). We found that kCL-trained networks started building slow dynamical features in early stages and in particular during the inference kindergarten task. Once

the temporal wagering task was introduced, some of these features were pruned from the network's dynamic repertoire, stabilizing into the final motif described above. This expansion and then pruning of dynamical features over training has been previously reported in other tasks that rely on attractor-based dynamics[32] and might reflect a biologically relevant learning trait. Interestingly, while the policy layer had similar numbers of dynamical features across learning procedures, the total number of slow features in the inference layer was curriculum specific, with kCL showing richer dynamics than suboptimal networks trained just using shaping (Fig. 6e). In addition, the dynamics of kCL were also higher dimensional compared with other protocols (Supplementary Fig. 7). Overall, these results suggest that the benefits of kCL may stem from its ability to endow the network with richer dynamics that can then be exploited when learning the target task.

Discussion

It has been long recognized that the 'key to shaping is identifying the essential subcomponents in a task'[2]. Here, we argue that the same insight can be used to build inductive biases that steer RNNs towards learned solutions that are both computationally more efficient and a closer match to animal behaviour.

While we found a common dynamical systems motif in many of the well-performing networks, the exact geometry of the solution was variable across networks and curricula, unlike the 'universal' dynamical systems solutions found for simpler tasks, where all networks that do well are dynamically isomorphic[32,33]. Thus, the task complexity may open interesting avenues for exploring how across-animal behavioural variability can be traced back to variability in neural dynamics in related brain regions. This is particularly important in our task, as we see is a high degree of variability in wait-time behaviour among rats[19].

Our approach traces its roots to not only behavioural shaping[34] but also classic curriculum learning and meta-learning. If the idea of pretraining for RNNs is not new; previous work has focused on using incremental training to expand the trainable temporal horizon of RNNs[2] or as an assay to uncover learning principles on tasks where behaviour is difficult to analyse, but that could be optimally performed even without curriculum learning[35]. It is also qualitatively different from approaches that pretrain on human or animal behaviour directly in a supervised manner[36,37], which we see as complementary. By contrast, a precise design of the curriculum was critical for our task. Our approach also relates to forms of meta-learning used to account for structure learning[38], where invariant relationships in a task are stored and reused for more efficient learning. While sharing its emphasis on common computation that generalizes across tasks, kCL is distinct in its emphasis on task compositionality and, in particular, in decomposing the task into non-sequential subcomputations.

How well would our approach generalize to other tasks? The core idea is to break the target task into computational subelements, which is also reminescent of the decomposition of Q values into subtasks seen in previously in a mouse navigation task[39]. An important note is that, in our case, some of the computational elements need to be executed in parallel. This contrasts to traditional views of compositionality (inherited from hierarchical RL) where the goal is learning to stitch together a sequence of operations[40]. The choice of subtask here is not ad hoc but relies on the availability of a simplified ideal observer solution. Admittedly, a mathematically tractable approximation of the task may not always be available or relevant, as animals can display distinctly suboptimal behaviour in other tasks, such as sequential biases and lapses[41]. Nonetheless, it may be possible to intuit at least some of the computational elements involved, in the kindergarten sense. In particular, kCL pretraining targeted to improve the ability to process information over long timescales is likely to benefit other tasks with long temporal dependencies and sparse reward structure, echoing the idea that certain basic cognitive abilities might generally support the learning of complex behaviour[42,43]. More broadly, our work argues that modelling complex cognitive tasks in RNNs requires careful thinking about pre-existing knowledge that animals bring with them to an experiment.

Methods

Animal subjects and behaviour

Behavioural procedures have been published in detail elsewhere[19]. Briefly, a total of 291 Long-evans rats (184 male, 107 female) between the ages of 6 and 24 months were used for this study (*Rattus norvegicus*). The Long-evans cohort also included ADORA2A-Cre ($N = 10$), ChAT-Cre ($N = 2$), DRD1-Cre ($N = 3$) and TH-Cre ($N = 12$). The animal use procedures were approved by the New York University Animal Welfare Committee (no. 2021-1120) and carried out in accordance with National Institutes of Health standards. The animals were water restricted to motivate them to perform behavioural trials. From Monday to Friday, they obtained water during behavioural training sessions, which were typically 90 min per day, and a subsequent ad libitum period of 20 min. Following training on Friday until mid-day Sunday, they received ad libitum water. The rats were weighed daily. The rats were trained in a high-throughput behavioural facility in the Constantinople lab using a computerized training protocol. The rats were trained in operant boxes with three nose poke ports. The left and right ports contained speakers to generate audio tones and contained lick tubes to delivery water reward. All ports contained an internal light-emitting diode (LED) light inside the port.

An LED illumination from the centre port indicated that the animal could initiate a trial by poking its nose in that port—upon trial initiation, the centre LED turned off. While in the centre port, the rats needed to maintain centre fixation for a duration drawn uniformly from 0.8 to 1.2 s. During the fixation period, a tone played from both speakers, the frequency of which indicated the volume of the offered water reward for that trial (1, 2, 4, 8 and 16 kHz, indicating 5, 10, 20, 40, 80 µl rewards). Following the fixation period, one of the two side LEDs was illuminated, indicating that the reward might be delivered at that port; the side was randomly chosen on each trial. This event (side LED ON) also initiated a variable and unpredictable delay period, which was randomly drawn from an exponential distribution with mean of 2.5 s. The reward port LED remained illuminated for the duration of the delay period, and the rats were not required to maintain fixation during this period, although they tended to fixate in the reward port. When a reward was available, the reward port LED turned off, and the rats could collect the offered reward by nose poking in that port. The rat could also choose to terminate the trial (opt out) at any time by nose poking in the opposite, unilluminated side port, after which a tone was played, and new trial would immediately begin. On a proportion of trials (15–25%), the delay period would only end if the rat opted out (catch trials). If rats did not opt out within 100 s on catch trials, the trial would terminate.

The trials were self-paced: after receiving their reward or opting out, the rats were free to initiate another trial immediately. However, if rats terminated centre fixation prematurely, they were penalized with a white noise sound and a timeout penalty (typically 2 s, although adjusted to individual animals). Following premature fixation breaks, the rats received the same offered reward, to disincentivize premature terminations for small volume offers. We introduced semiobservable, hidden states in the task by including uncued blocks of trials with varying reward statistics: high and low blocks, which offered the highest three or lowest three rewards, respectively, and were interspersed with mixed blocks, which offered all volumes. There was a hierarchical structure to the blocks, such that high and low blocks alternated after mixed blocks (for example, mixed-high-mixed-low or mixed-low-mixed-high). The first block of each session was a mixed block. The blocks transitioned after 40 successfully completed trials. Because the rats prematurely broke fixation on a subset of trials, in practice, the block durations were variable.

Behavioural shaping. The shaping procedure was divided into eight stages. For stage 1, the rats learned to maintain a nose poke in the centre port, after which a 20 µl reward volume was delivered from a random illuminated side port with no delay. Initially, the rats needed to maintain a 5 ms centre poke. The centre poke time was incremented by 1 ms following each successful trial until the centre poke time reached 1 s, after which the rat moved to stage 2.

Stages 2–5 progressively introduced the full set of reward volumes and corresponding auditory cues. The rats continued to receive deterministic rewards with no delay after maintaining a 1 s centre poke. Each stage added one additional reward that could be selected on each trial—stage 2 added 40 µl, stage 3 added 5 µl, stage 4 added 80 µl and stage 5 added 10 µl. Each stage progressed after 400 successfully completed trials. All subsequent stages used all five reward volumes.

Stage 6 introduced variable centre poke times, uniformly drawn from 0.8 to 1.2 s. Moreover, stage 6 introduced deterministic reward delays. Initially, the rewards were delivered after a 0.1 s delay, which was incremented by 2 ms after each successful trial. After the rat reached delays between 0.5 and 0.8 s, the reward delay was incremented by 5 ms following successful trials. Delays between 0.8 and 1 s were incremented by 10 ms, and delays between 1 and 1.5 s were incremented by 25 ms. The rats progressed to stage 7 after reaching a reward delay of 1.5 s.

In stage 7, the rats experienced variable delays, drawn from an exponential distribution with mean of 2.5 s. Moreover, we introduced catch trials (see 'Animal subjects and behaviour' section), with a catch probability of 15%. Stage 7 terminated after 250 successfully completed trials. Finally, stage 8 introduced the block structure. We additionally increased the catch probably for the first 1,000 trials to 35%, to encourage the rats to learn that they could opt out of the trial. After 1,000 completed trials, the catch probability was reduced to 15–20%. All the animal data in Fig. 2 were from training stage 8. The conceptual changes that occur in stages 7 and 8 were used in shaping of the RNNs.

Behavioural analyses

The behavioural sessions from the rats required at least five catch trials to be included. In addition, the sessions were excluded if a linear regression of wait time did not have positive slope in a 2-day moving-window average or if it lacked a statistically significant positive slope coefficient for linear sensitivity on that day (F test, $P < 0.05$). Lastly, trials were excluded if the wait time was greater than two standard deviations from the mean wait time. These criteria were included primarily to exclude trials in which a rat was disengaged during the task.

For analysis of wait time ratio and regression of wait time over training, the data were aggregated over groups of two sessions either at the beginning of block training or the end of training ('first sessions' and 'last sessions', respectively, in Figs. 2f and 3g). The wait-time ratio was calculated on these aggregated sessions, and a regression of wait time with regressors for the current trial, one trial back and current block were used to regress the wait time on that trial. For this, the wait times for opt-out trials were z-scored, and an ordinal block was used ($B_{\text{low}} = 1$, $B_{\text{mixed}} = 2$ and $B_{\text{high}} = 3$).

The analysis of block transition dynamics followed from ref. 19. The wait times on catch trials of each rat and RNN were first z-scored separately for each volume, then the difference in these z-scored wait times were calculated for each volume, relative to the average z-scored wait time for that volume. This was calculated for each trial relative to an incongruent trial following a block transition. The change in wait time in Figs. 2e and 3f was an average of volume of these values. This approach was used to control for reward volume effects.

Approximate Bayesian inference of block. The Bayesian observer in Fig. 2g was calculated on the basis of methods from ref. 19. Briefly, the posterior belief of block was calculated according to Bayes rule

$$P(B_t|R_t) = \frac{P(R_t|B_t)P(B_t)}{P(R_t)}, \tag{4}$$

where B_t is the block on trial t and R_t is the reward on trial t. The likelihood $P(R_t|B_t)$ is the probability of the reward for each block (1/5 for all offers in mixed blocks and 1/3 or 0 for low and high blocks). The prior $P(B_t)$ is approximated using a posterior from the last trial as

$$P(B_t) \approx P(B_t|R_{t-1}) = \sum_{B_{t-1}} P(B_t|B_{t-1})P(B_{t-1}|R_{t-1}), \tag{5}$$

where $P(B_t|B_{t-1})$ is referred to as the hazard rate, which incorporates knowledge of the task structure, including the block length and block transition probabilities. For example, for blocks of length H, the hazard rate for low blocks would be

$$P(\text{low}\,|B_{t-1}) = \begin{cases} 1-H, & B_{t-1} = \text{low} \\ H, & B_{t-1} = \text{mix} \\ 0, & B_{t-1} = \text{high} \end{cases}, \tag{6}$$

where $H = 1/40$, to reflect the block length in this work. An extensive explanation motivating the assumption of a flat hazard rate is provided in ref. 19.

Details of MDP and RL formulation of task

The decision process (Fig. 2h) and RL task environment (Fig. 3b) were modelled as a simplified form of the animal task, without any left/right side choice information, requirement to persevere in the centre port or required action to initiate new trials. Formally, this was modelled as an episodic MDP with two potential actions (Wait, OptOut). The state at each point in a trial is minimally defined as a reward offer R and time within a trial, t: $S(R, t)$. The rewards are delivered probabilistically on each trial with probability $p_r = 0.8$. Within a trial, the reward is delivered at a time that is drawn from an exponential delay distribution $p(t) = \lambda^{-1}\exp(-t/\lambda)$ (where the mean wait time $\lambda = 2.5$ s). At every time step before reward is delivered, the agent received a small reward penalty $k = -0.05$, and when the opt-out penalty action is chosen, a reward penalty $R_{oo} = -2.0$ is provided. The opt-out penalty was used to discourage the suboptimal strategy of instantaneously opting out. Rather than instantaneously award the reward offer or opt-out penalty, the reward was instead evenly divided out across the time bins of the intertrial interval (ITI). ITIs were drawn from a uniform distribution from 50 ms to 1 s. This form of ITI better mimicked the experience in the rat task, in which animals were drinking water throughout most the ITI. It also creates longer timescale dependencies across trials, and receiving positive reward (rather than no reward) during the ITI improves value function estimates across trials. To avoid contaminating learning of the policies for Wait and OptOut, we enforced a required waiting during the ITI (waitITI) action that was given unit probability of occurring during the ITI and zero during the waiting epoch. We do not model violations in the RL formulation of the task nor do we model the decision to begin new trials. These reflect separate computational processes that are separate from the core decision to wait or opt out.

Reward offers followed an alternating block structure, as in the rat task, with random transitions. The same alternating block structure was used, with sessions always beginning with a mixed block, and then transitioning into a low block. Note, in the rat task, there is equal probability of high or low blocks being the second block, though this difference is probably negligible to RNN behaviour. The blocks were a minimum of 40 trials long, and after 40 trials, the block transitions occurred probabilistically, drawn from a binomial distribution with transition probability $p = 0.5$.

Model architecture

The RL agent is composed of a two-layer network, each with 256 LSTM units (Fig. 3a), with relevant gates, states and inputs denoted by (j) for layer j. The computational LSTM unit of the network uses the following gates to update the hidden states h_t and cell states c_t:

$$\mathbf{i}_t = \sigma\left(W_{ii}\mathbf{x}_t + \mathbf{b}_{ii} + W_{hi}\mathbf{h}_{t-1} + \mathbf{b}_{hi}\right) \tag{7}$$

$$\mathbf{f}_t = \sigma\left(W_{if}\mathbf{x}_t + \mathbf{b}_{if} + W_{hf}\mathbf{h}_{t-1} + \mathbf{b}_{hf}\right) \tag{8}$$

$$\mathbf{o}_t = \sigma\left(W_{io}\mathbf{x}_t + \mathbf{b}_{io} + W_{ho}\mathbf{h}_{t-1} + \mathbf{b}_{ho}\right) \tag{9}$$

$$\mathbf{g}_t = \tanh\left(W_{ig}\mathbf{x}_t + \mathbf{b}_{ig} + W_{hg}\mathbf{h}_{t-1} + \mathbf{b}_{hg}\right) \tag{10}$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \mathbf{g}_t \tag{11}$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t), \tag{12}$$

where $\odot$ is the elementwise Hadamard product and σ is a sigmoid nonlinearity. $\mathbf{S}_t^{(j)} \equiv [\mathbf{h}_t^{(j)}, \mathbf{c}_t^{(j)}]$ is a compact description of the activity of each layer, and successive states of the network are written here with a shorthand $S_{t+1} = \mathrm{RNN}(S_t)$ or $S_{t+1} = \mathrm{RNN}(\cdot)[(S_t)]$. The network inputs x_t are multidimensional and will generally consist of external task stimuli s_{start} and s_R, past time step reward r_{t-1}, past time step action a_{t-1}, outputs from the network o_t and states S_t from earlier LSTM layers. $s_{\mathrm{start}} = 1$ and $s_{\mathrm{R}} = \log(R)$ only at the start of the trial and are otherwise zero. By contrast, the remaining inputs can in general be non-zero and time-varying over the course of an entire trial.

Specifically, the first layer ('inference' layer) receives inputs about the task stimuli of trial start and reward offer, as well as the previous reward and previous action: $\mathbf{x}^{(\mathrm{inference})} = [s_{\mathrm{start},t}, s_{R,t}, r_{t-1}, a_{t-1}]$. The inference layer projects onto a three-unit linear projection head that outputs the log-probability of each block $o_t^{(\mathrm{block})} = \log[p_{\mathrm{block},t}]$, as well as a linear projection head for an auxiliary task that aims to output the average reward within a trial $o_t^{(\mathrm{int})}$. The second layer ('policy' layer) receives log-block probabilities from the inference layer as inputs, as well as the task stimuli for trial start and reward offer: $\mathbf{x}^{(\mathrm{policy})} = [s_{\mathrm{start},t}, s_{R,t}, o_t^{(\mathrm{block})}]$. The policy layer outputs onto linear projection heads for the (1) policy π_t, (2) value of current state V_t, (3) prediction of reward offer for a supervised memory task $o_t^{(\mathrm{mem})}$ and (4) prediction of time within a task for a supervised memory task $o_t^{(\mathrm{count})}$. The policy π is a three-unit RNN with a softmax nonlinearity, corresponding to probabilities for waiting within a trial (waitTrial), opting out of a trial (OptOut) or waitITI. During the trial, the only allowed options are waitTrial and Optout, with waitITI artificially set to low probability. During the the ITI period in which reward is delivered, the only option is waitITI. The remaining projection heads output scalar values.

Hidden and cell states of the LSTMs were initialized with random values drawn from a normal distribution ($\mathcal{N}(0,1)$). The weight parameters of the model were initialized to small values, drawn from a uniform distribution ($\mathcal{U}(-1/N, 1/N)$, where N is the number of hidden-layer units). The states were reset at the end of each training epoch, where a training epoch was over all data for kindergarten costs. The states were reset every 160 trials during the temporal wagering task. This reset corresponds to roughly the same timescale as progressing through all three block types, though it is not exactly aligned with block transitions.

Training

All RNN model training was conducted in PyTorch (v1.8.0, python v3.9.5). For all costs, the weights were updated using backpropagation through time with an Adam minimizer. The hyperparameters for training are provided in the Supplementary Information.

Kindergarten training. Kindergarten tasks were composed of supervised learning tasks: memory, counting, integration and inference tasks. The memory, counting and integration tasks were trained using supervised learning of a mean-squared error loss, where RNN outputs o_t were trained to match a target output z_t. The memory task trained the network to output the initial reward offer throughout the duration of a trial ($o_{t,\mathrm{targ}} = s_{R,0}$). The counting task trained networks to count time elapsed within a trial ($o_{t,\mathrm{targ}} = t$). The integration task trained networks to calculate a running average of the 'previous reward' stimulus input ($o_{t,\mathrm{targ}} = 1/t \sum_0^t r_{t-1}$). The inference task trained networks on a classification task using cross-entropy loss to categorize the latent block on the basis of reward offer.

These tasks used similar input and trial time statistics as the target temporal wagering task. Trial durations were on the same order of magnitude as the target task, and the inputs to the RNN used the same five target values as the log-reward offer stimulus input in the RL task. The target in the integration task was based off of typical rewards received from trials in the wait-time task. Importantly, while the input statistics were similar to the target task, these kindergarten tasks are supervised learning objectives in which the actions of the RL agent have no bearing. They are fundamentally a different class of learning than the final RL-based task.

We devised a cumulative curriculum for pretraining on the four tasks by introducing tasks into training, one at a time and adding onto previous tasks (Supplementary Fig. 1). We first introduced simple, single-trial variants in order of memory, counting and integration tasks (simple kindergarten). We then increased the complexity of these tasks by extending to multiple trials in a training batch but continuing to train on the three tasks simultaneously (hard kindergarten). Finally, we added in the inference task (inference kindergarten). We chose this ordering such that the successive tasks took more training time than previous ones.

The losses for each stage of kindergarten are described as

$$\begin{aligned}\mathcal{L}_{\mathrm{simple}} = \sum_t^T & \beta_{\mathrm{mem}}\left(o_t^{(\mathrm{mem})} - z_t^{(\mathrm{mem})}\right)^2 \\ & + \beta_{\mathrm{count}}\left(o_t^{(\mathrm{count})} - z_t^{(\mathrm{count})}\right)^2 + \beta_{\mathrm{int}}\left(o_t^{(\mathrm{int})} - z_t^{(\mathrm{int})}\right)^2,\end{aligned} \tag{13}$$

$$\mathcal{L}_{\mathrm{hard}} = \sum_{m=1}^{M} \mathcal{L}_{\mathrm{simple}}^{(m)}, \tag{14}$$

where $o_t^{(\cdot)}$ is the output signals from the RNN and $z_t^{(\cdot)}$ is the target outputs. $o_t^{(int)}$ comes from the inference layer, and $o_t^{(mem)}$ and $o_t^{(count)}$ come from the policy layer. t denotes time within a single trial, and m denotes the trial number. Training data for simple kindergarten was performed in a single batch, with 20 time steps of data per sample, ($T = 1$ s, $\Delta t = 0.05$), and a batch size of 1,000 samples. A weight update occurred after every epoch over the data and stopped when performance did not improve over 30 epochs. The second stage of kindergarten training (hard) simply expanded the time horizon of simple kindergarten, which optimized with target values from a single trial, m, to include multiple trials. We used $M = 10$ trials per sample, with variable trial times drawn from a uniform distribution between 1 and 5 s. The training data were again a single batch, with a batch size of 1,000. To maintain a single batch of data with the same amount of data per sample, each sample used ten trials of the same durations but in random order for each sample. A weight update was performed after each epoch over the data and completed after a threshold of $\mathcal{L}_{\mathrm{hard}} < 0.001$ was reached or until 10,000 epochs were performed. Rather than processing each contribution to the loss cumulatively as in simple kindergarten, all three costs were simultaneously optimized in hard kindergarten.

Following training on simple and hard kindergarten, which were tasks with squared error losses, the final stage of kindergarten (inference) was performed. Formally, this required the outputs of the inference layer head, p_{block} to minimize a cross-entropy categorization loss at every timepoint,

$$\mathcal{L}_{\text{inf}} = -\sum_{m,t} I^{(m)} \log\left[p^{(m)}_{\text{block},t}\right], \tag{15}$$

where $I^{(m)}$ is an indicator function for the true block on each trial, m, taking the value $I^{(m)} = 1$ for the true block type and 0 for the remaining block types. This final stage of kindergarten cumulatively optimized inference and the earlier kindergarten tasks

$$\mathcal{L}_{\text{kind}} = \beta_{\text{hard}}\mathcal{L}_{\text{hard}} + \beta_{\text{inf}}\mathcal{L}_{\text{inf}}, \tag{16}$$

where $\beta_{\text{hard}} = 1.0$ and $\beta_{\text{inf}} = 0.5$ during kindergarten pretraining.

The sham delay-to-non-match task used an augmented RNN with two additional inputs to the inference layer, as well as an additional output from the policy layer. The two inputs were chosen from the the potential offers provided during the task ([5, 10, 20, 40, 80]), and there was a temporal delay between the two offers drawn from a uniform distribution $\mathcal{U}(0.05\,\text{s}, 5\,\text{s})$. There was equal probability of both inputs being the same or different on each trial. As with the other supervised learning tasks, the trials were batched. The goal of the the agent was to output 'match' (1) if the two inputs matched, 'non-match' (2) if they were different and 'no cue' (0) otherwise. The task was optimized using a cross-entropy loss as in equation (15).

For the variations of the curriculum studied in Fig. 5, the ordering of tasks also proceeded as performing kindergarten tasks first, followed by shaping and then the target task. Any task used in the pretraining stages was also used as a regularizer in during target task training, and all others were omitted as regularizers. The only exception was the 'regularization-only' curriculum in Fig. 5d. The ordering of kindergarten tasks proceeded as performing mean-squared-error-based kindergarten tasks (memory, counting and integration) before the inference task. One exception was the 'inference first' curriculum in Supplementary Fig. 2b–e.

Deep meta-RL loss. Following previous work[24], the temporal wagering task described above was optimized with a deep meta-learning framework that optimized an actor-critic loss that was regularized with an entropy loss to encourage exploration, as well as the full kindergarten loss,

$$\begin{aligned}\mathcal{L} = \sum_t & -\beta_{\text{actor}} \log(\pi_t)\left[G_t - V(S_t)\right] + \beta_{\text{critic}}\left[G_t - V(S_t)\right]^2 \\ & - \beta_H \pi_t \log(\pi_t) + \mathcal{L}_{\text{kind}},\end{aligned} \tag{17}$$

where G_t is the empirical, total discounted future reward

$$G_t = r_t + \gamma G_{t+1}. \tag{18}$$

This Actor Critic approach trains networks to generate optimal decision policies π_t and value function estimates V_t but does so in a way that allows trial-by-trial learning to occur through persistent dynamical activity, as opposed to continual parameter updating of the connection weights between units. The key architectural ingredient of teaching this network to generate an internal RL procedure via its dynamics (the 'meta' component) is to provide the network with explicit feedback about the past action taken and last reward received[24]. In our two-layer architecture, this feedback is provided only to the inference layer.

Network analyses

To determine which phases of training created large structural changes in the network, we calculated the change in weights ΔW over training. Specifically, we investigated the squared L2 norm of the difference of a concatenated recurrent weight matrix $W \equiv [W_{hf}, W_{hg}, W_{ho}, W_{hi}]$ over training. Different phases of training had different learning rates, so to meaningfully compare the magnitude of changes over training, we sampled the network at different rates across training, proportional to their learning rates. Intuitively, this means we sampled training phases less frequently when only small parameter updates were possible (smaller learning rate) and conversely sampled more frequently in training epochs where large parameter updates were possible (larger learning rate). Specifically, the change $\|\Delta W\|^2$ in each stage is given by

$$\|\Delta \mathbf{W}_s\|_L^2 = \|\mathbf{W}_{s+N_L} - \mathbf{W}_s\|_L^2, \tag{19}$$

$$N_L = \frac{0.1}{\alpha^{(e)}}. \tag{20}$$

N_L is the number of gradient steps between samples of network weights and was empirically chosen on the basis of the relative learning rates $\alpha^{(e)}$ in each stage. For simple kindergarten, hard kindergarten and the temporal wagering task, $\alpha^{(e)} = 1 \times 10^{-4}$ ($N = 1{,}000$ steps). For the inference stage of kindergarten, $\alpha^{(e)} = 0.005$ ($N = 2$ steps). The wait-time task has variable numbers of gradient steps across trials, but empirically, we observed that 1,000 gradient steps spanned approximately 10,000 trials. Thus, we sampled the network every 10,000 trials for the temporal wagering task.

Reduced dynamics flow fields

A low-dimensional manifold of activity was found by performing a principle component analysis (PCA) on $S^{(i)}$ from a session of 1,000 trials from the temporal wagering task. A PCA was performed separately for each layer of the network to give a reduced set of activity $P_t^{(i)} = [S_t^{(i)} - \hat{S}^{(i)}]M^{(i)}$, where $M^{(i)}$ are the principal components, and $\hat{S}^{(i)}$ is the mean activity of network in layer i, respectively. Networks trained with full kCL had ~90% variance explained in the first two to three principal components. Thus, we visualized the dynamics in a two-dimensional space; however, an analysis of the low-dimensional dynamics was always performed in the N-dimensional space that captured at least 90% variance.

Dynamical flow fields reflect the instantaneous change of the network state over time, due to network activity. We approximate this temporal gradient with an empirical difference F_t across successive states as

$$\dot{S}_t \equiv \frac{\partial S_t}{\partial t} \approx \frac{S_{t+1} - S_t}{\Delta t} \equiv F_t, \tag{21}$$

$$F_t = \frac{(RNN(\cdot) - \mathbb{I})[S_t]}{\Delta t}. \tag{22}$$

The operator $RNN(\cdot)$ denotes propagating the state S_t by one time step, and $\mathbb{I}$ is the identity operator.

The flow field for behaviourally relevant low-dimensional dynamics in a PC space is equivalently calculated by first projecting back into neural activity space with an inverse PCA transform, calculating F_t, then projecting this gradient back into PC space. Assuming $\hat{S}^{(i)} = 0$ for simplicity, this dynamical gradient $F_{PC,t}$ is given by

$$\dot{P}_t \equiv \frac{\partial P_t}{\partial t} \approx \frac{P_{t+1} - P_t}{\Delta t} \equiv F_{PC,t}, \tag{23}$$

$$F_{PC,t} = \frac{(RNN(\cdot) - \mathbb{I})\left[P_t M^T\right] M}{\Delta t}. \tag{24}$$

In practice, single-trial trajectories in these two dimensions tended to follow the flow fields of then two-dimensional dynamics quite well and served as a sufficient space to analyse aspects of the dynamics that directly contribute to behaviour.

Dynamics are only well defined for static inputs. For each layer, the inputs used the wait-time epoch used in Fig. 6 are $\mathbf{x}^{(\text{inference})} = [s_{\text{start},t} = 0, s_{\text{R},t} = 0, r_{t-1} = -k, a_{t-1} = 0]$ and $\mathbf{x}^{(\text{policy})} = [s_{\text{start},t} = 0,\ s_{\text{R},t} = 0, o_{\text{block}}]$. To calculate the reduced dynamics for the policy layer with a static input, we chose o_{block} values the corresponded to the block probability estimates output from the RNN during characteristic opt-out trials from each block, just before the RNN opted out.

Linearized dynamics

The dynamics of an RNN are in general nonlinear and can be approximated by a Taylor series expansion around the fixed-point S_0 as

$$\dot{\mathbf{S}}_t \approx \nabla[\dot{\mathbf{S}}_t]^T\Big|_{\mathbf{S}_0}(\mathbf{S}_t - \mathbf{S}_0) + \frac{1}{2}(\mathbf{S}_t - \mathbf{S}_0)^T \nabla^2[\dot{\mathbf{S}}_t]\Big|_{\mathbf{S}_0}(\mathbf{S}_t - \mathbf{S}_0) + \text{h.o.t.}, \quad (25)$$

where h.o.t. means higher order terms. Analysis of the linearized dynamics matrix $J \equiv \nabla[\dot{\mathbf{S}}_t]^T_{i,j} = \partial\dot{\mathbf{S}}_i/\partial\mathbf{S}_j$ (that is, Jacobian of the temporal gradient) provides a compact description of the underlying dynamical system driving behaviour. The Jacobian for an LSTM only requires derivatives with respect to the cell state and hidden state. We again approximate the temporal gradient with an empirical difference and calculate the Jacobian J as

$$\nabla[\dot{\mathbf{S}}_t]^T \approx \nabla[\mathbf{F}_t]^T \equiv J = \left(\frac{\partial \mathbf{F}_t}{\partial \mathbf{h}_t}, \frac{\partial \mathbf{F}_t}{\partial \mathbf{c}_t}\right) \quad (26)$$

$$= \frac{1}{\Delta t}\begin{pmatrix} \frac{\partial \mathbf{h}_{t+1}}{\partial \mathbf{h}_t} - \mathbb{I} & \frac{\partial \mathbf{h}_{t+1}}{\partial \mathbf{c}_t} \\ \frac{\partial \mathbf{c}_{t+1}}{\partial \mathbf{h}_t} & \frac{\partial \mathbf{c}_{t+1}}{\partial \mathbf{c}_t} - \mathbb{I} \end{pmatrix}, \quad (27)$$

where we have used the fact that $S_{t+1} = \text{RNN}(\cdot)[(S_t)]$. The Jacobian can be written more compactly as

$$J = \frac{1}{\Delta t}\begin{pmatrix} h_h^T - \mathbb{I} & h_c^T \\ c_h^T & c_c^T - \mathbb{I} \end{pmatrix}. \quad (28)$$

We found that the Jacobian was typically non-normal (that is, when $JJ^T \neq J^TJ$), so we characterized the linearized dynamics by their spectral properties by using the Schur decomposition $J = WTW$ (ref. 44). The Schur decomposition is analogous to a standard eigenvalue decomposition but returns orthogonal modes, even for non-normal matrices. We characterized fixed points of the dynamics by the eigenvalues of J (diag(T)) and the Schur modes W. Additional details for the linearization can be found in the Supplementary Information.

Locating and classifying dynamical fixed points

Analysis of the dynamics for each network numerically located fixed points (F_t or $F_t^{(\text{PC})} = 0$), as well as 'slow points', where the dynamics contained local minima, but $F_t^{(\text{PC})} > 0$. These points were located through a minimization of the kinetic energy of the system, where kinetic energy is defined by

$$\text{KE} = \frac{1}{2}F_{\text{PC},t}^2. \quad (29)$$

To locate the fixed points in the behaviourally relevant subspace of RNN activity, we constrained the minimization of kinetic energy $F^{(\text{PC})}$ in the top dimensions that explained at least 90% of variance. Thus, we searched for kinetic energy minima on the N-dimensional PCA manifold. While a fully unconstrained minimization would identify all of the fixed points of the RNN dynamics, constraining to PC space restricts our analysis to the network dynamics driving behaviour. To choose initial conditions to the minimization, we used a uniform sampling for 50 points per dimension when $N = 1$ or $N = 2$. For higher dimensionality, to avoid exponential scaling issues we used a biased grid search proportional to total variance explained by each dimension. The same data and network inputs for calculating flow fields were used to calculate kinetic energy.

Once minima were located, a clustering procedure (DBSCAN, scipy.clustering.DBSCAN) was performed to determine the effective number of fixed and slow points, as well as remove any outliers. The hyperparameters were min_samples = 10 and epsilon. Epsilon was chosen individually for each network, as 1% of the range of support of PC1 activity. The fixed points were defined as identified minima where $KE < 1 \times 10^{-4}$. All other minima were termed 'slow points'. The network dynamics were then linearized at the identified fixed and slow points, then a Schur decomposition was performed on the Jacobian to retrieve the eigenvalues and Schur modes of the system. The dynamical systems features were categorized in the top two dimensions of PC space. Based on their eigenvalues λ_1 and λ_2 found by the Schur decompositions, we used the following scheme:

- $\lambda_1 < 0.999, \lambda_2 < 0.999$: point attractor
- $\lambda_1 \in [0.999, 1.001]$ and $\lambda_2 < 1.001$: stable line/plane attractor
- $\lambda_1 \in [0.999, 1.001]$ and $\lambda_2 > 1.001$: unstable line
- $\lambda_1 < 0.999$ and $\lambda_2 > 1.001$: saddle point

To characterize the dynamics across RNNs, we identified a 'common motif' in the dynamics of the inference layer if the following features were observed: the dynamics contain a stabilizing feature (point or line attractor) in high and low-block regions of state space, as well as an unstable feature (typically a saddle) in the mixed-block region of state space. In addition, a two-dimensional PC space needed to contain three discrete and continuous regions of block confidence, assessed using o_{block}. The networks were considered to have 'rat-like' strategies if they possessed linear sensitivity to reward offers, as well sensitivity to block context for all blocks in the same ordering as rats (longer wait times in low blocks). The flow field visualization was performed in Matlab (R2023b).

Reporting summary

Further information on research design is available in the Nature Portfolio Reporting Summary linked to this article.

Data availability

The rat behavioural data and the statistical model of behaviour are detailed in ref. 19 and are available via Zenodo at https://doi.org/10.5281/zenodo.10031483 (ref. 45). The data used to generate figures in this manuscript, as well as a repository of pretrained RNN model files for different curriculum learning strategies, are available via Zenodo at https://doi.org/10.5281/zenodo.14907819 (ref. 46).

Code availability

The data were analysed with code written in Python (Python v3.9.5, Pytorch v1.8.0), as well as Matlab (v2023b). The code used to train RNNs, analyse data and generate figures is available at GitHub via https://github.com/Savin-Lab-Code/kind_cl (ref. 47) and as CodeOcean capsule[48].

References

1. Yang, G. R. & Wang, X.-J. Artificial neural networks for neuroscientists: a primer. *Neuron* **107**, 1048–1070 (2020).
2. Krueger, K. A. & Dayan, P. Flexible shaping: How learning in small steps helps. *Cognition* **110**, 380–394 (2009).
3. Narvekar, S. et al. Curriculum learning for reinforcement learning domains: a framework and survey. *J. Mach. Learn. Res.* **21**, 181 (2020).
4. Bengio, Y., Louradour, J., Collobert, R. & Weston, J. Curriculum learning. In *Proc. 26th Annual Int. Conference on Machine Learning* 41–48 (Association for Computing Machinery, 2009).

5. Finn, C., Abbeel, P. & Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. In *Int. Conference on Machine Learning*, 1126–1135 (PMLR, 2017).
6. Thrun, S. & Pratt, L. in *Learning to Learn* (eds Thrun, S. & Pratt, L.) 3–17 (Springer, 1998).
7. Harlow, H. F. The formation of learning sets. *Psychol. Rev.* **56**, 51–65 (1949).
8. Skinner, B. F. How to teach animals. *Sci. Am.* **185**, 26–29 (1951).
9. Savin, C. & Triesch, J. Emergence of task-dependent representations in working memory circuits. *Front. Comput. Neurosci.* **8**, 57 (2014).
10. McAndrew, R. & Helms Tillery, S. I. Laboratory primates: their lives in and after research. *Temperature* https://doi.org/10.1080/23328940.2016.1229161 (2016).
11. Chowdhury, S. A. & DeAngelis, G. C. Fine discrimination training alters the causal contribution of macaque area mt to depth perception. *Neuron* **60**, 367–377 (2008).
12. Arlt, C. et al. Cognitive experience alters cortical involvement in goal-directed navigation. *eLife* **11**, e76051 (2022).
13. Wang, J. X. Meta-learning in natural and artificial intelligence. *Curr. Opin. Behav. Sci.* **38**, 90–95 (2021).
14. Vanderschuren, L. J., Achterberg, E. M. & Trezza, V. The neurobiology of social play and its rewarding value in rats. *Neurosci. Biobehav. Rev.* **70**, 86–105 (2016).
15. Vanderschuren, L. J. & Trezza, V. What the laboratory rat has taught us about social play behavior: role in behavioral development and neural mechanisms. In *The Neurobiology of Childhood* (eds Andersen, S. L. & Pine, D. S.) 189–212 (Springer, 2014).
16. Baarendse, P. J., Limpens, J. H. & Vanderschuren, L. J. Disrupted social development enhances the motivation for cocaine in rats. *Psychopharmacology* **231**, 1695–1704 (2014).
17. Einon, D. F. & Morgan, M. A critical period for social isolation in the rat. *Dev. Psychobiol. J. Int. Soc. Dev. Psychobiol.* **10**, 123–132 (1977).
18. Zador, A. M. A critique of pure learning and what artificial neural networks can learn from animal brains. *Nat. Commun.* **10**, 3770 (2019).
19. Mah, A., Schiereck, S. S., Bossio, V. & Constantinople, C. M. Distinct value computations support rapid sequential decisions. *Nat. Commun.* **14**, 7573 (2023).
20. Schiereck, S. S. et al. Neural dynamics in the orbitofrontal cortex reveal cognitive strategies. Preprint at *bioRxiv* https://doi.org/10.1101/2024.10.29.620879 (2024).
21. Constantino, S. M. & Daw, N. D. Learning the opportunity cost of time in a patch-foraging task. *Cog. Affect. Behav. Neurosci.* **15**, 837–853 (2015).
22. Charnov, E. L. Optimal foraging, the marginal value theorem. *Theor. Popul. Biol.* **9**, 129–136 (1976).
23. McNamara, J. M. & Houston, A. I. Optimal foraging and learning. *J. Theor. Biol.* **117**, 231–249 (1985).
24. Wang, J. X. et al. Prefrontal cortex as a meta-reinforcement learning system. *Nat. Neurosci.* **21**, 860–868 (2018).
25. Wilson, R. C., Takahashi, Y. K., Schoenbaum, G. & Niv, Y. Orbitofrontal cortex as a cognitive map of task space. *Neuron* **81**, 267–279 (2014).
26. Schuck, N. W., Cai, M. B., Wilson, R. C. & Niv, Y. Human orbitofrontal cortex represents a cognitive map of state space. *Neuron* **91**, 1402–1412 (2016).
27. Bartolo, R. & Averbeck, B. B. Inference as a fundamental process in behavior. *Curr. Opin. Behav. Sci.* **38**, 8–13 (2021).
28. Averbeck, B. & O'Doherty, J. P. Reinforcement-learning in fronto-striatal circuits. *Neuropsychopharmacology* **47**, 147–162 (2022).
29. Miyashita, Y. & Chang, H. S. Neuronal correlate of pictorial short-term memory in the primate temporal cortexyasushi miyashita. *Nature* **331**, 68–70 (1988).
30. Dubreuil, A., Valente, A., Beiran, M., Mastrogiuseppe, F. & Ostojic, S. The role of population structure in computations through neural dynamics. *Nat. Neurosci.* **25**, 783–794 (2022).
31. Khona, M. & Fiete, I. R. Attractor and integrator networks in the brain. *Nat. Rev. Neurosci.* **23**, 744–766 (2022).
32. Marschall, O. & Savin, C. Probing learning through the lens of changes in circuit dynamics. Preprint at *bioRxiv* https://doi.org/10.1101/2023.09.13.557585 (2023).
33. Sussillo, D. & Barak, O. Opening the black box: low-dimensional dynamics in high-dimensional recurrent neural networks. *Neural Comput.* **25**, 626–649 (2013).
34. Elman, J. L. Learning and development in neural networks: the importance of starting small. *Cognition* **48**, 71–99 (1993).
35. Kepple, D., Engelken, R. & Rajan, K. Curriculum learning as a tool to uncover learning principles in the brain. In *Int. Conference on Learning Representations* (2022).
36. Silver, D. et al. Mastering the game of go with deep neural networks and tree search. *Nature* **529**, 484–489 (2016).
37. Jensen, K. T., Hennequin, G. & Mattar, M. G. A recurrent network model of planning explains hippocampal replay and human behavior. *Nat. Neurosci.* **27**, 1340–1348 (2024).
38. Braun, D. A., Mehring, C. & Wolpert, D. M. Structure learning in action. *Behav. Brain Res.* **206**, 157–165 (2010).
39. Makino, H. Arithmetic value representation for hierarchical behavior composition. *Nat. Neurosci.* **26**, 140–149 (2023).
40. Driscoll, L. N., Shenoy, K. & Sussillo, D. Flexible multitask computation in recurrent networks utilizes shared dynamical motifs. *Nat. Neurosci.* **27**, 1349–1363 (2024).
41. Gupta, D., DePasquale, B., Kopec, C. D. & Brody, C. D. Trial-history biases in evidence accumulation can give rise to apparent lapses in decision-making. *Nat. Commun.* **15**, 662 (2024).
42. Richards, B. A. et al. A deep learning framework for neuroscience. *Nat. Neurosci.* **22**, 1761–1770 (2019).
43. Ma, W. J. & Peters, B. A neural network walks into a lab: towards using deep nets as models for human behavior. Preprint at https://arxiv.org/abs/2005.02181 (2020).
44. Goldman, M. S. Memory without feedback in a neural network. *Neuron* **61**, 621–634 (2009).
45. Mah, A., Schiereck, S., Bossio, V. & Constantinople, C. Distinct value computations support rapid sequential decisions (version v1). *Zenodo* https://doi.org/10.5281/zenodo.10031483 (2023).
46. Hocker, D., Constantinople, C. M. & Savin, C. Composition of simple computational tasks captures the inductive biases of animals in network models (version v1). *Zenodo* https://doi.org/10.5281/zenodo.14907819 (2025).
47. Hocker, D., Constantinople, C. M. & Savin, C. Savin-Lab-Code/kind_cl: Nature Machine Intelligence code (v1.0.0). *Zenodo* https://doi.org/10.5281/zenodo.14907734 (2025).
48. Hocker, D., Constantinople, C. M. & Savin, C. Compositional pretraining improves computational efficiency and matches animal behavior on complex tasks. *Code Ocean Capsule* https://doi.org/10.24433/CO.3440797.v1 (2025).
49. Arnold, T. B. & Emerson, J. W. Nonparametric goodness-of-fit tests for discrete null distributions. *R J.* **3**, 34–39 (2011).

Acknowledgements

We thank L. Driscoll, P. Glimcher, V. Goudar, O. Marschall, K. Miller and J. Wang for helpful discussions and comments on the manuscript. We also thank M. Chittireddy for fixed-point characterization of shaping-trained RNNs. This work was supported by the NIMH (grant nos. 1K01MH132043-01A1 to D.H. and 1R01MH125571-01 to C.M.C. and C.S.). This work was supported in part through the NYU IT High Performance Computing resources, services and staff expertise. We gratefully acknowledge use of the research computing resources of

the Empire AI Consortium, Inc., with support from the State of New York, the Simons Foundation and the Secunda Family Foundation.

Author contributions

C.M.C. designed the behavioural task. D.H., C.M.C. and C.S. designed the curriculum training protocol. D.H. created the RNN model. D.H., C.M.C. and C.S analysed the data. D.H. prepared the figures. D.H., C.M.C. and C.S. wrote the manuscript. C.M.C. and C.S. supervised the project.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s42256-025-01029-3.

Correspondence and requests for materials should be addressed to Cristina Savin.

Peer review information *Nature Machine Intelligence* thanks Maria Eckstein, and the other, anonymous, reviewer(s) for their contribution to the peer review of this work.

nature portfolio

Corresponding author(s): Cristina Savin

Last updated by author(s): Feb 26, 2025

Reporting Summary

Nature Portfolio wishes to improve the reproducibility of the work that we publish. This form provides structure for consistency and transparency in reporting. For further information on Nature Portfolio policies, see our Editorial Policies and the Editorial Policy Checklist.

Statistics

For all statistical analyses, confirm that the following items are present in the figure legend, table legend, main text, or Methods section.

n/a	Confirmed	
☐	☒	The exact sample size (*n*) for each experimental group/condition, given as a discrete number and unit of measurement
☒	☐	A statement on whether measurements were taken from distinct samples or whether the same sample was measured repeatedly
☐	☒	The statistical test(s) used AND whether they are one- or two-sided *Only common tests should be described solely by name; describe more complex techniques in the Methods section.*
☐	☒	A description of all covariates tested
☒	☐	A description of any assumptions or corrections, such as tests of normality and adjustment for multiple comparisons
☐	☒	A full description of the statistical parameters including central tendency (e.g. means) or other basic estimates (e.g. regression coefficient) AND variation (e.g. standard deviation) or associated estimates of uncertainty (e.g. confidence intervals)
☐	☒	For null hypothesis testing, the test statistic (e.g. *F*, *t*, *r*) with confidence intervals, effect sizes, degrees of freedom and *P* value noted *Give P values as exact values whenever suitable.*
☒	☐	For Bayesian analysis, information on the choice of priors and Markov chain Monte Carlo settings
☒	☐	For hierarchical and complex designs, identification of the appropriate level for tests and full reporting of outcomes
☒	☐	Estimates of effect sizes (e.g. Cohen's *d*, Pearson's *r*), indicating how they were calculated

Our web collection on statistics for biologists contains articles on many of the points above.

Software and code

Policy information about availability of computer code

Data collection	No data collection involved.
Data analysis	Data was analyzed with code written in Python (Python v 3.9.5, Pytorch v 1.11.0), as well as Matlab (v2023b). This code is publicly available at https:// github.com/Savin-Lab-Code/kind_cl. This information has been included in the Data and Code Availability section of the paper.

For manuscripts utilizing custom algorithms or software that are central to the research but not yet described in published literature, software must be made available to editors and reviewers. We strongly encourage code deposition in a community repository (e.g. GitHub). See the Nature Portfolio guidelines for submitting code & software for further information.

Data

Policy information about availability of data

All manuscripts must include a data availability statement. This statement should provide the following information, where applicable:

- Accession codes, unique identifiers, or web links for publicly available datasets
- A description of any restrictions on data availability
- For clinical datasets or third party data, please ensure that the statement adheres to our policy

Rat behavioral data published in Mah et al Nature Communications 2023 is publicly available on Zenodo at https://zenodo.org/records/10031483. The data used to generate figures in this manuscript, as well as a repository of pretrained RNN model files for different curriculum learning strategies, is available at a separate Zenodo database: https://zenodo.org/records/14907819 . This information has been included in the Data Availability section of the paper.

Human research participants

Policy information about studies involving human research participants and Sex and Gender in Research.

Reporting on sex and gender	N/A
Population characteristics	N/A
Recruitment	N/A
Ethics oversight	N/A

Note that full information on the approval of the study protocol must also be provided in the manuscript.

Field-specific reporting

Please select the one below that is the best fit for your research. If you are not sure, read the appropriate sections before making your selection.

☒ Life sciences ☐ Behavioural & social sciences ☐ Ecological, evolutionary & environmental sciences

For a reference copy of the document with all sections, see nature.com/documents/nr-reporting-summary-flat.pdf

Life sciences study design

All studies must disclose on these points even when the disclosure is negative.

Sample size	Rat behavioral data from the previosuly published dataset (Mah et al, Nat. Comm 2023) included N=291 rats to capture as much behavioral variability as possible across a large population of rats. We trained N=46 networks with full curriculum learning, as well as N=50 networks of partial curriculum, and N=50 networks with no curriculum learning. Additionally, we simulated N=10 networks with a sham form of kindergarten curriculum learning that included a delay-to-non-match task. We simulated from this large size of networks to obtain as much variability as possible in network activity, while also balancing computational resources required to train deep RL models.
Data exclusions	Rat behavioral data has behavioral sessions excluded where wait time and log(reward) were not linearly related, for at least 2 consecutive sessions. Wait times were removed on a session-by-session basis if they were greater than 2 standard deviations above the mean. Finally, sessions were excluded if they contained less than 5 catch trials. These restrictions were used to remove data in which the rat was not engaged int he task RNNs used the same exclusion criteria in replicating behavioral analyses (Fig 3 e,f). Such criteria are not generally necessary for RNNs though, as they do not become disengaged in the task. In order to perform time warping, analysis of linearized dynamics in Fig. 4 excluded trials if they had wait times less then 5 time points (250ms), or ITI of less than 4 time points (400 ms).
Replication	All rats received the same training protocol. Where applicable, plotted data from individual rats that were representative of the population. Sets of RNNs received the same curriculum learning of tasks, with the same hyperparameters in all cases. The structure of the tasks were the same for each RNN, but random seeds that set the initial RNN weights, and ordering of trials within each training stage were different for each RNN. These seeds are included as explicit input parameters to the simulation, to allow for replication of results. Replication of RNN results from a given seed was used only for internal purposes, to check code integrity. All of these internal replication checks were sucessful.
Randomization	The number of random RNN seeds varied for each curriculum studied, ranging from 10-50 random seeds. The number of random seeds is reported in each relevant figure caption (Figs 3-6)
Blinding	A human researcher classified the "shaping only" dynamics to classify the number and type of fixed point in each network. This researcher was not cued to the common motif located in "kindergarten + shaping" dynamics (e.g. Fig 5) , so as not to skew their analysis when comparing across curriculum types.

Reporting for specific materials, systems and methods

We require information from authors about some types of materials, experimental systems and methods used in many studies. Here, indicate whether each material, system or method listed is relevant to your study. If you are not sure if a list item applies to your research, read the appropriate section before selecting a response.

Materials & experimental systems

n/a	Involved in the study
☒	☐ Antibodies
☒	☐ Eukaryotic cell lines
☒	☐ Palaeontology and archaeology
☐	☒ Animals and other organisms
☒	☐ Clinical data
☒	☐ Dual use research of concern

Methods

n/a	Involved in the study
☒	☐ ChIP-seq
☒	☐ Flow cytometry
☒	☐ MRI-based neuroimaging

Animals and other research organisms

Policy information about studies involving animals; ARRIVE guidelines recommended for reporting animal research, and Sex and Gender in Research

Laboratory animals	Long-evans rats (184 male, 107 female) between ages of 6 and 24 months were used for this study (Rattus norvegicus)
Wild animals	This study did not involve wild animals.
Reporting on sex	Data was collected for both male and female rats.
Field-collected samples	This study did not contain samples collected from the field.
Ethics oversight	Animal use procedures were approved by the New York University Animal Welfare Committee (UAWC #2021-1120) and carried out in accordance with National Institutes of Health standards.

Note that full information on the approval of the study protocol must also be provided in the manuscript.