MULTITASK INTERACTIONS IN RECURRENT DYNAMICS ACROSS 100 COGNITIVE TASKS

Anonymous authors
Paper under double-blind review

ABSTRACT

The brain continually learns tasks with very different computational demands. Recurrent neural networks (RNNs) have been used as dynamical surrogates for studying task solutions, with recent work extending this approach to joint training on multitask batteries. However, it remains unclear how tasks interact as these batteries grow in scale and heterogeneity: under what conditions do they transfer or interfere with one another, and how do these effects relate to task complexity and model structure? We study this question by simultaneously training hierarchical Almost-Linear RNNs (AL-RNNs) on 100 tasks spanning a wide range of cognitive demands. AL-RNNs allow us to vary the amount of nonlinearity in the recurrence, while hierarchical task features provide a low-dimensional interface for task-specific adaptation within shared dynamics. On this task battery, performance improves with both nonlinear capacity and task-feature dimensionality, with the joint hierarchical model outperforming both individually trained models and a single unstructured joint model. Many tasks are individually solvable with linear dynamics, yet cannot be solved together by a shared linear model. Through hierarchical task features, task-specific linear region usage, and task geometry, the trained model provides several related but distinct measures of task similarity that capture how computations are separated and combined within the shared backbone. Across a series of experiments, we relate these representations to individual task complexity, joint solvability, pairwise transfer learning, and leave-one-out adaptation. Together, our experiments characterize task interactions in multitask recurrent learning while demonstrating both the potential and the challenges of reusable dynamical backbones for cognitive computation.

1 INTRODUCTION

The mammalian brain is a remarkably flexible learning system, acquiring a large repertoire of cognitive tasks by both efficiently leveraging shared structure across tasks (transfer learning) while preventing new learning from overwriting existing solutions (catastrophic forgetting). Achieving this requires a dynamical substrate supporting reusable motifs and task-specific separation. However, isolating and interpreting these mechanisms from recordings of complex nonlinear brain dynamics remains challenging. To better understand the dynamical mechanism underpinning task solutions, task-trained recurrent neural networks (RNNs) have emerged as dynamical surrogate models that allow researchers to analyze the resulting attractor dynamics and compare them to neural recordings (Mante et al., 2013; Sussillo and Barak, 2013). Historically focused on single-task settings, this approach has identified task-specific motifs but leaves open how they coexist when multiple tasks are embedded in a shared system.

In domains like vision and language, where data modalities are structurally homogeneous, positive transfer is often the default expectation (Raffel et al., 2020). However, cognitive tasks vary substantially in input structure, delay periods, and computational demands, creating a real risk of interference, requiring the model to actively separate conflicting computations within the shared dynamics. Recent work has begun to compile task batteries and investigated joint training and dynamical mechanisms (Yang et al., 2019; Turner and Barak, 2023; Driscoll et al., 2024). So far, these task suites have usually been relatively small and structured, with many of their tasks admitting surprisingly simple solutions (Khona et al., 2023). This motivates larger and more heterogeneous batteries (Khona et al., 2023) for

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

asking how task interactions scale when the shared model must accommodate diverse computational demands.

Understanding these interactions requires a framework that can explicitly control the capacity for nonlinear computation and task-specific adaptation. In this work, we employ hierarchical Almost-Linear RNNs (AL-RNNs) as an interpretable experimental framework to study multitask interactions within a shared recurrent dynamical backbone. The AL-RNN architecture allows us to vary the amount of piecewise-linear nonlinear computation, while hierarchical task features provide a low-dimensional interface for task-specific adaptation. We train this model jointly on a heterogeneous battery of 100 supervised cognitive tasks, combining paradigms adapted from prior multitask studies (Yang et al., 2019; Driscoll et al., 2024; Khona et al., 2023) with additional tasks compiled from the cognitive and neuroscientific literature. We compare joint hierarchical models with individually trained networks and flat joint models to characterize how heterogeneous tasks interact under shared recurrent dynamics. The hierarchical AL-RNN provides a parameter-efficient compromise that outperforms matched-size individual models and unstructured joint models. Across several analyses, we find that nonlinearity, low-dimensional task features, and recurrent state geometry provide complementary forms of task organization, while jointly learning more heterogeneous tasks requires increasingly complex models. These representations define interpretable task-similarity spaces that relate to transfer learning: some tasks systematically produce negative transfer when used as initializations, and a task's isolation in the learned space was predictive of poorer transfer and poorer leave-one-out adaptation to it. Together, these findings demonstrate that effective multitask computation is possible but requires actively structured capacity to manage interference within shared dynamics.

2 Related Work

Recurrent neural networks have become standard tools for modeling cognitive dynamics, successfully replicating neural activity and behavior in single-task settings (Mante et al., 2013; Sussillo and Barak, 2013; Yang et al., 2019). While recent work has scaled cognitive modeling, e.g. using large language models fine-tuned on broad behavioral datasets (Binz et al., 2025), RNNs remain unique in providing interpretable, low-dimensional dynamical systems directly comparable to neural recordings (Khona and Fiete, 2022; Durstewitz et al., 2023). However, most such models focus on isolated tasks, and how heterogeneous tasks interact within a single network remains poorly understood.

Recent work has increasingly examined recurrent networks trained on multiple cognitive tasks simultaneously. Task suites such as NeuroGym (Crocioni et al., 2026) and Mod-Cog (Khona et al., 2023) make such studies possible, while highlighting that many of their tasks admit simple solutions, from sparse or even diagonal connectivity to linear dynamics (Khona et al., 2023). Yang et al. (2019) trained one RNN on 20 cognitive tasks and found clustered units and compositional task representations, and Driscoll et al. (2024) showed that related tasks reuse shared dynamical motifs. Duncker et al. (2020) organize dynamics into task-specific, orthogonal subspaces while reusing motifs across similar computations. Turner and Barak (2023) call this tendency of RNNs to develop shared common representations across tasks a "simplicity bias", linking it to the emergence of shared attractor structure during training. Flesch et al. (2022) link such orthogonal, low-dimensional representations to robustness across contexts in networks and brains. Together, this literature has mainly characterized how structured task relationships enable sharing and compositionality.

Indeed, where tasks are structurally homogeneous, sharing parameters across tasks typically helps (Raffel et al., 2020). However, sharing is not necessarily beneficial, and in multitask settings with heterogeneous tasks, these can compete for shared capacity and can degrade one another, leading to the concept of "negative transfer", which is well established across the wider multitask learning literature (Standley et al., 2020; Zhang et al., 2023), but has received less systematic attention in dynamical models of cognitive task batteries. One line of work treats such interference as an optimization problem of conflicting gradient directions between tasks (Yu et al., 2020; Sener and Koltun, 2018), though the need for such specialized optimizers is contested (Kurin et al., 2022). A second line of work addresses interference through model structure, explicitly balancing shared and task-specific computation, with modularity constituting a useful inductive bias (Goyal et al., 2020). A number of approaches separate shared from task-specific computation explicitly, e.g. through learned routing between function blocks (Rosenbaum et al., 2018) or task-specific gates over shared experts

(Ma et al., 2018), while a number of other approaches have observed these structures emerge in trained networks (Yang et al., 2019), and leveraged in neural populations (Dubreuil et al., 2022).

Task conditioning provides another mechanism for combining sharing and separation, and can create a shared representational geometry that supports generalization to unseen tasks (Riveland and Pouget, 2024). Task identity can be characterized during joint training (Fifty et al., 2021) or post hoc, for instance through Fisher-information-based embeddings (Achille et al., 2019). Hypernetworks generate model parameters from task features and can mitigate forgetting in continual learning (von Oswald et al., 2019; Ehret et al., 2020; Durstewitz et al., 2025), while hierarchical models combine low-dimensional task-specific features with shared parameters (Brenner et al., 2025). These learned representations also define task spaces in which similarity can be quantified and related to transfer, providing a bridge between structured multitask models and the study of task interactions.

3 METHODS

Studying multitask interactions requires a model that allows us to quantify task complexity, distinguish shared from task-specific computation, and remain tractable as the number and heterogeneity of tasks increase. It should also provide an adaptation interface that allows new tasks to be learned without fitting an independent network for each task or modifying the shared dynamics. These requirements motivate a model that is efficient in both latent dimensionality and dynamical complexity, while retaining sufficient task-specific flexibility to avoid interference. We combine three components to meet these requirements: i) an Almost-Linear RNN (AL-RNN) (Brenner et al., 2024), in which latent dimension M and nonlinearity P are decoupled and can be tuned independently (Sec. 3.1), ii) a hierarchical inference framework (Brenner et al., 2025), which generates task-specific recurrent parameters from low-dimensional task features and shared group-level parameters, providing a common dynamical backbone with an interpretable task interface (Sec. 3.2), and iii), a shared input-output interface that accommodates heterogeneous task formats while keeping the recurrent parameterization consistent across all tasks (Appx. A.1).

3.1 ALMOST-LINEAR RNNS

The AL-RNN allows the two principal capacity axes (latent dimensionality M and the number of nonlinear units P) to be varied independently. This allows us to distinguish tasks that require rich state-space structure despite being linearly solvable from tasks that require nonlinear computation even in relatively simple geometries (Brenner and Koppe, 2026). The AL-RNN evolves a latent state $\boldsymbol{z}_t \in \mathbb{R}^M$ according to

$$\boldsymbol{z}_t = \boldsymbol{A}\boldsymbol{z}_{t-1} + \boldsymbol{W}\,\Phi^*(\boldsymbol{z}_{t-1}) + \boldsymbol{C}\boldsymbol{s}_t + \boldsymbol{h}, \tag{1}$$

with diagonal $\boldsymbol{A} \in \mathbb{R}^{M\times M}$, off-diagonal $\boldsymbol{W} \in \mathbb{R}^{M\times M}$, input matrix $\boldsymbol{C} \in \mathbb{R}^{M\times N}$, bias $\boldsymbol{h} \in \mathbb{R}^M$, and external input $\boldsymbol{s}_t$. Nonlinearity is applied to the last $P \leq M$ units only,

$$\Phi^*(\boldsymbol{z}_t) = \left[z_{1,t},\, \ldots,\, z_{M-P,t},\, \phi(z_{M-P+1,t}),\, \ldots,\, \phi(z_{M,t})\right]^\top, \tag{2}$$

with $\phi(z) = \max(0, z)$ as in the original formulation. For $P = 0$ the model is a linear state space model, while for $P = M$ it reduces to a standard piecewise-linear RNN (Durstewitz, 2017). The latent states are read out using a linear layer $\boldsymbol{B} \in \mathbb{R}^{5\times M}$ with $\boldsymbol{y} = \boldsymbol{B}\boldsymbol{z}$, with task-specific masks selecting the relevant timesteps and readout channels.

The piecewise-linear structure makes the model interpretable in terms of its linear region usage. For a given activation pattern $\boldsymbol{b}_t \in \{0,1\}^P$, where $b_{j,t} = \mathbf{1}\{z_{M-P+j,t} > 0\}$, the nonlinear units contribute the diagonal gating matrix $\widetilde{\boldsymbol{D}}(\boldsymbol{b}_t) = \mathrm{diag}(\boldsymbol{b}_t)$, where $\widetilde{\boldsymbol{D}}(\boldsymbol{b}_t) \in \mathbb{R}^{M\times M}$ is diagonal, with zeros in the first $M - P$ entries and $\boldsymbol{b}_t$ in the final P entries. Within a fixed region, the dynamics therefore reduce to the linear state-space system $\boldsymbol{z}_t = \left(\boldsymbol{A} + \boldsymbol{W}\widetilde{\boldsymbol{D}}(\boldsymbol{b}_t)\right)\boldsymbol{z}_{t-1} + \boldsymbol{C}\boldsymbol{s}_t + \boldsymbol{h}$. Each binary activation pattern, i.e. its bitcode, such as $\boldsymbol{b}_t = (0, 1, 1, \ldots, 0)$, defines one of the 2^P piecewise-linear regions. The bitcodes visited during a task then encode how the model uses nonlinearity to solve the task (Brenner and Koppe, 2026).

3.2 Hierarchical Multitask Inference

A joint model should share structure wherever tasks use related computations, while retaining enough task-specific flexibility to prevent interference. To achieve this, we use a hierarchical formulation (Brenner et al., 2025) that places task-specific variation in a low-dimensional space while generating the parameters of a task-specific dynamical system (DS) model from shared group-level parameters, depicted in Fig. 1. Each task $k \in \{1, \ldots, K\}$ is assigned an embedding $\boldsymbol{p}^{(k)} \in \mathbb{R}^{N_{\text{feat}}}$, with $N_{\text{feat}} \ll \dim(\boldsymbol{\theta}_{\text{DSR}})$, and a shared hypernetwork $G_{\boldsymbol{\theta}_{\text{group}}}$ maps this embedding to the parameters of its task-specific DS, parameterized e.g. by an AL-RNN:

$$\boldsymbol{\theta}_{\text{DSR}}^{(k)} = G_{\boldsymbol{\theta}_{\text{group}}}\left(\boldsymbol{p}^{(k)}\right), \qquad \boldsymbol{\theta}_{\text{DSR}}^{(k)} = \{\boldsymbol{A}^{(k)}, \boldsymbol{W}^{(k)}, \boldsymbol{C}^{(k)}, \boldsymbol{B}^{(k)}, \boldsymbol{h}^{(k)}\}. \tag{3}$$

We use a linear mapping for G, so that all task-specific models lie in an N_{feat}-dimensional linear subspace of parameter space. Since the group parameters $\boldsymbol{\theta}_{\text{group}}$ receive gradients from every task, any dynamical structure that is useful more than once should be preferentially represented there, while the only degrees of freedom individual to a task are the N_{feat} coordinates of $\boldsymbol{p}^{(k)}$. Task-specific adaptation is therefore confined to a low-dimensional subspace. This formulation also automatically supplies task identity, such that all task-specific structure the model discovers is localized in one object.

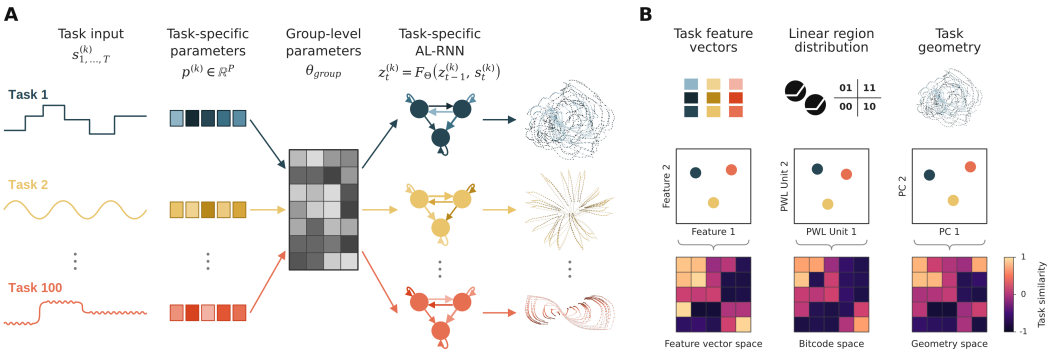


Figure 1: Hierarchical task conditioning and task-similarity spaces. **A**: Every task receives low-dimensional task features that modulate group-level parameters, producing task-conditioned AL-RNNs within a shared recurrent backbone. **B**: The learned model supports complementary task representations based on task features, piecewise-linear region usage, and state geometry.

3.3 Task Suite

We train on a heterogeneous battery of $K = 100$ supervised cognitive tasks spanning a diverse set of requirements, such as working memory, decision-making, perception, attention, timing, reasoning, and motor response. The battery combines task paradigms and variants adapted from prior multitask cognitive RNN studies (Yang et al., 2019; Khona et al., 2023; Driscoll et al., 2024), with additional tasks compiled from the cognitive-science and systems-neuroscience literature. Following the compositional construction of Khona et al. (2023), selected base tasks are expanded with modifiers that add delay, sequence generation, interval integration, or combinations thereof. Each trial may contain context, stimulus, delay, and response phases. The final battery contains 66 base task identities and 34 compositional variants, each treated as a separate task with its own task embedding. The complete task list, readout types, source references, and modifier definitions are provided in Appx. C.

All tasks are jointly trained using a common masked mean-squared-error objective (Eq. 4, see Appx. A.2 for details). Accuracy is evaluated separately using task-appropriate trial-level criteria: final-step accuracy for single-response tasks, response-step accuracy for reaction-time tasks, and the fraction of correctly scored timesteps for sequential tasks. Directional outputs are evaluated by angular error, while scalar and vector outputs use calibrated tolerances. The choice of tolerances and robustness analyses are described in Appx. A.1. We report both mean accuracy across tasks and the number of solved tasks, where a task is considered solved above an 80% accuracy threshold.

4 RESULTS

4.1 PERFORMANCE ACROSS THE TASK BATTERY

We trained three model variants on the full battery of 100 cognitive tasks: i) the task-conditioned hierarchical AL-RNNs, ii) flat AL-RNNs (a single model trained on all tasks at once), and iii) individually trained AL-RNNs (one model per task), each with matched parameter budgets. We further compared to hierarchical variants of vanilla RNNs, GRUs and LSTMs. These results are summarized in Table 1 for 200 samples per task. We find that hierarchical formulations of the AL-RNN, GRU and LSTM perform best, reaching around 96% test accuracy, clearly outperforming both individually trained models, reaching 91%, and the flat model, performing worst at around 81% accuracy. Thus, joint training provides a substantial sharing benefit, even though many tasks are already solvable in isolation, while the structure of the sharing is important in order to avoid the negative interference observed in the flat model. These effects were also largely conserved across different sample sizes, with overall worse performance for smaller sample sizes (Fig. 6).

Table 1: Performance of the best nonlinear model for each configuration and the corresponding linear models, with matched parameter counts and 200 samples per task. A task is counted as solved when its seed-averaged test accuracy is at least 80% in the joint models, and if one single seed reaches 80% for the individual models.

Model	M	P	N_{feat}	Test accuracy (%)	Tasks solved	Parameters
Nonlinear models						
LSTM	32	–	16	96.3 ± 0.1	94	87K
GRU	32	–	16	95.6 ± 0.7	93	66K
Vanilla RNN	64	–	16	93.8 ± 0.4	90	81K
Hierarchical AL-RNN	64	32	16	96.5 ± 0.3	94	82K
Flat joint	234	16	–	80.9 ± 4.3	63	81K
Individual	23	16	–	91.3 ± 0.6	86	84K
Linear models						
Hierarchical AL-RNN	64	0	16	70.4 ± 0.3	53	82K
Flat joint	234	0	–	51.5 ± 0.8	16	81K
Individual	23	0	–	64.1 ± 1.2	53	84K

Individual task complexity varies across the battery To assess how model complexity affects joint performance, we varied the number of nonlinear units and the dimensionality of the task features, including a comparison to fully linear RNNs. Performance in the hierarchical model improved smoothly along both dimensions (Fig. 2**A**), but relatively little capacity was needed to approach the performance ceiling. With only $N_{\text{feat}} = 4$ task features, the model reached approximately 95% accuracy at sufficient nonlinear capacity, while introducing just two nonlinear units increased performance from roughly 70% to 89% with sufficient hierarchical features. This high performance of simple models is striking given the apparently demanding cognitive and computational requirements represented in the battery, and only a small subset of tasks requires substantial model complexity. For instance, linear hierarchical and individually trained models already solve 53 tasks, and a small number of nonlinear units suffices to solve most other tasks. This extends the observation of Khona et al. (2023) that commonly used cognitive tasks often admit surprisingly simple recurrent solutions. Other tasks, like the classic neuroscientific *head direction integration* paradigm, require nonlinear dynamics to implement more complex attractor structures (Fig. 5). We found that linear solvability was also a proxy for how fast a task was learned. We measured the time required for each task to reach $> 80\%$ test accuracy and compared it with the task's performance in an individually trained linear model. Tasks well solved by linear models converged substantially faster than tasks requiring nonlinearity. For example, in the joint model with $P = 16$ nonlinear units, the tasks classified as individually linear were learned approximately four to five times faster than the remaining tasks (Fig. 2**B**). This relationship was observed for both individually trained models and the hierarchical joint model, whose convergence times were highly consistent across tasks (Fig. 2**B**, Spearman $\rho \approx 0.86$). The flat joint model instead exhibited several pronounced task-specific outliers, indicating that the

unstructured shared parameterization introduces additional optimization difficulty for some tasks, particularly without sufficient nonlinear capacity.

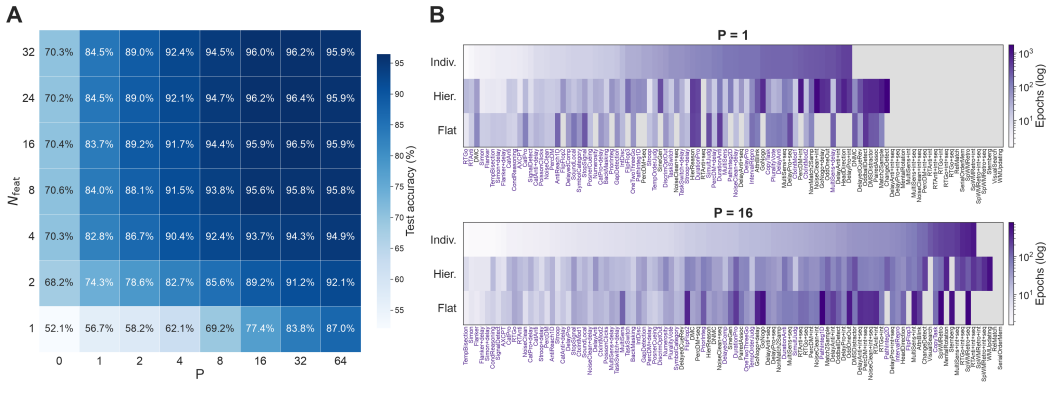


Figure 2: **A**: Test accuracy of the hierarchical model (AL-RNN, $M = 64$) across task-embedding dimension N_{feat} and nonlinear units P, averaged across 5 seeds. **B**: Epochs to reach $> 80\%$ test accuracy per task at $P = 1$ and $P = 16$ for individual models ($M = 64$), the hierarchical model ($N_{\text{feat}} = 16$) and a parameter-matched flat model ($M = 234$), averaged over the seeds that reached threshold. Grey: never reached. Purple names: tasks solved by a fully linear individual model.

4.2 Mechanisms for Task Separation

A shared recurrent model can separate tasks in at least three ways: by assigning them different task-specific features, by routing them through different linear regions, or by embedding their computations in different geometric subspaces of the state space. The model comparisons in Table 1 allow us to examine these mechanisms separately. The flat linear model has access only to geometric separation and solves just 16 tasks, although 53 tasks are individually linearly solvable. Adding hierarchical task features recovers the individually linear solutions, but cannot solve tasks that require nonlinearity in isolation. Conversely, adding nonlinear units to the flat model both recovers individually linear tasks and enables some intrinsically nonlinear tasks, reaching 63 solved tasks at its best-performing setting. However, performance peaks with sparse nonlinearity (16/234 nonlinear units), and declines as further nonlinear units are added (Fig. 6, mirroring similar findings by Brenner and Koppe, 2026). The hierarchical nonlinear model, which combines all three forms of organization, performs best, while retaining an individual DS model per task that can be analyzed in terms of its dynamical mechanism.

Individual vs. joint solvability We next tested the relationship between individual and joint complexity explicitly by combining sets of tasks that were each individually linearly solvable. As the number of jointly trained tasks increased, more nonlinear units were required to solve the complete set (Fig. 7). The AL-RNN lets us examine how nonlinear separation is implemented. Let R denote the active piecewise-linear region, observed through its bitcode $\boldsymbol{b}_t$ at time t, and let Q denote task identity. The conditional entropy $H(R \mid Q)$ measures variation in region usage within tasks, while the mutual information $I(Q;R) = H(R) - H(R \mid Q)$ measures how much task identity can be inferred from the active region alone. High $I(Q;R)$ therefore indicates that tasks occupy distinct linear subregions, whereas low $I(Q;R)$ indicates larger overlap. In both model classes, $I(Q;R)$ increased with nonlinear capacity. However, $I(Q;R)$ was generally much larger in the flat model than in the hierarchical one (e.g. for $P = 16$, 1.6 in the hierarchical model but approximately 5.0 bits in the flat model, close to the task entropy ceiling of ≈ 6.64 bits). The flat model therefore relies more heavily on leveraging different linear regions to separate tasks, whereas the hierarchical model can implement part of this separation through its task features. We find a similar pattern in recurrent-state geometry. Task-specific overlap of the first five principal components (PCs) is larger in the hierarchical than in the flat model (approximately 0.26 versus 0.14, Fig. 9), indicating greater sharing of task subspaces. At the same time, the first five PCs of the global state space explain more variance in the hierarchical model (0.43 versus 0.32), consistent with a more compact overall geometry. Thus, the flat model separates tasks more strongly in both linear region usage and state

space geometry, whereas the hierarchical model can additionally leverage the flexibility of the task features to enable sharing and separation more effectively.

4.3 Task Similarity Spaces

The preceding results identify three mechanisms through which tasks can be organized within the shared model: hierarchical task features, linear region usage, and the geometric overlap of task-specific activity subspaces. Each mechanism defines a pairwise similarity measure between tasks (cosine similarity of task features, overlap of linear region distributions, and overlap of the first five activity PCs) which we interpret as complementary *task-similarity spaces* (Fig. 3). We next investigate their robustness, agreement, and relationship to transfer and interference.

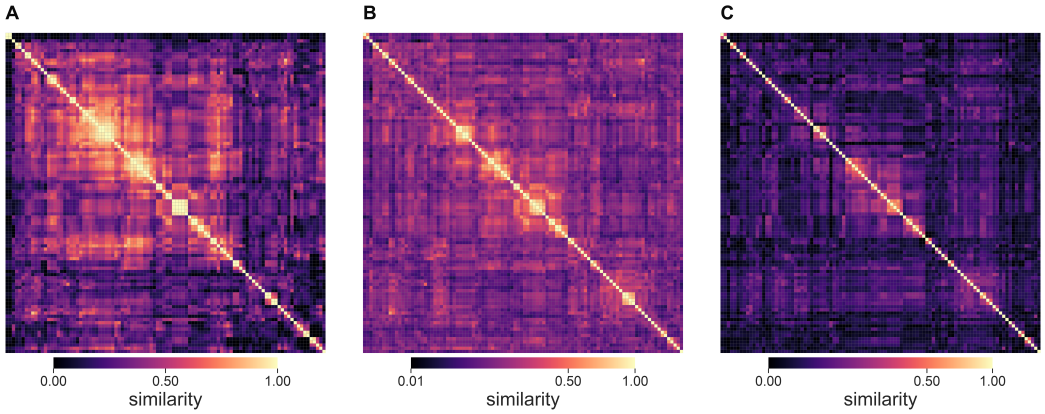


Figure 3: Task-similarity spaces learned by the hierarchical AL-RNN for **A** task features, **B** geometry, and **C** linear region usage, for $N_{\text{feat}} = 16$ and $P = 16$. Tasks are ordered by clustering the hierarchical task features (clustering details and task labels reported in Appx. A.2).

We first assessed reproducibility by correlating the off-diagonal entries of the 100×100 task-similarity matrices across random seeds. All three AL-RNN spaces were consistent across seeds ($r \approx 0.76$-0.78 for $N_{\text{feat}} = 16, P = 16$; Table 4), with similar values in LSTMs and GRUs (Table 5). Agreement across the different AL-RNN spaces was lower ($r = 0.47$-0.70), indicating that they capture related but non-identical aspects of task organization. Feature-similarity structure also generalized moderately across architectures (AL-RNN–LSTM: $r = 0.53$; AL-RNN–GRU: $r = 0.57$). Thus, task organization depends partly on the recurrent architecture, but substantial structure remains reproducible across both initializations and model classes.

Task spaces reveal interpretable structure The three task spaces can be queried along different dimensions to examine how the model organizes task solutions. We provide three illustrative, non-exhaustive examples. Task geometry similarity reveals that tasks with related dynamical computations occupy visually similar geometric organizations, including recurring rotations, trajectories, and other low-dimensional motifs (Fig. 8). For example, HeadDirection and its nearest neighbours in both feature and geometric space, and among the closest tasks in linear region usage, NoiseCleaner+int and PerceptualDM+int, reuse a very similar piecewise-linear approximation to a ring attractor (Fig. 4**A**, see also Fig. 5). Thus, proximity in the learned task spaces can reflect shared dynamical objects that are repurposed across tasks. Second, compositional variants of the same tasks cluster in the hierarchical feature space. For the 49 tasks belonging to 15 base families with at least one modified counterpart, the mean cosine similarity distance to variants was 0.21 ± 0.02, while the mean distance to other tasks was almost three-fold, 0.62 ± 0.03, $p < .001$ for all seeds (Fig. 4**B**). Finally, pro-anti pairs of the same tasks are often distributed along different linear subregions, in line with the observation that the model requires nonlinearity to jointly solve similar tasks with opposing response rules (Fig. 4**C**, see also Fig. 10).

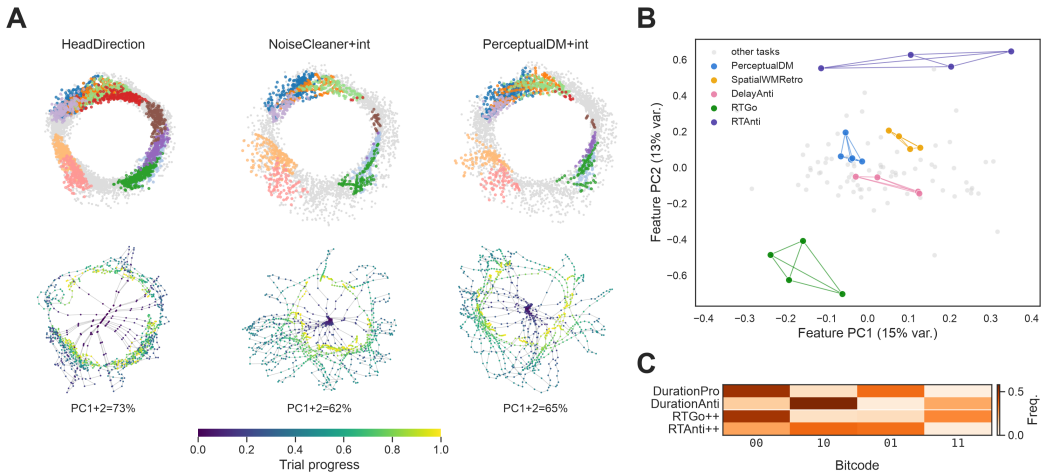


Figure 4: **A**: Example of three tasks (HeadDirection, NoiseCleaner+int and PerceptualDM+int) close in the task similarity spaces ($P = 16$, $N_{\text{feat}} = 16$) leveraging shared attractor structure. Top: Network states during the integration/delay phase, projected onto the first two PCs of HeadDirection's state space. The ring attractor is approximated by piecewise linear subregions (see also Fig. 5). Points of the 12 most visited regions, coloured by the linear region they occupy (grey for all others), appear at similar positions on all three rings. Bottom: 30 example trials per task, coloured by trial progress (dark = trial start, yellow = trial end). **B**: Five tasks with their respective task modifiers in the hierarchical feature space cluster together. **C**: Pro and Anti variants of the same tasks are placed in opposing linear subregions. ++ denotes +int+seq modifier.

4.4 Task representations predict transfer and adaptation

Task isolation predicts individual-model transfer We then evaluated transfer between individually trained models by initializing each target task from every other task's trained solution and measuring early learning over the first 100 epochs (see Appx. A.2). Pretraining improved learning relative to random initialization in approximately 70% of source-target pairs, but several targets were learned more effectively from scratch than from most pretrained sources. Thus, pretrained task solutions provide useful priors on average, but can act as negative priors for particular tasks. We then asked whether this source quality was related to the task similarity spaces learned by the joint model ($P = 16$, $N_{\text{feat}} = 16$). Across tasks, more isolated representations were modestly associated with poorer transfer sources in both task geometry ($\rho = -0.26$, $p = 0.01$) and linear region usage ($\rho = -0.23$, $p = 0.02$, also see Fig. 11). These effects do not fully explain transfer, but provide preliminary evidence that task organization in the joint model reflects aspects of transferability between independently trained task solutions.

Leave-one-out adaptation We finally tested whether a model pretrained on 99 tasks could learn to solve the remaining task by adapting only a task-specific feature vector ($N_{\text{feat}} = 16$). For each held-out task, we froze the shared parameters and initialized the feature vector either from the mean of the 99 pretrained task features or from the best-performing source vector among them. This procedure preserves the pretrained tasks by construction and tests whether the shared backbone already contains a suitable solution that can be accessed through its low-dimensional task interface. Adaptation reached $92.1\% \pm 0.1\%$ accuracy when initialized from the best source vector, close to the 96% achieved by the fully trained joint model, and even slightly outperforming individually trained full models ($\approx 91\%$, Table 1). In contrast, initialization from the mean feature vector reached only $84.5\% \pm 1.9\%$. The best-source result should be viewed as an oracle upper bound and not a realistic initialization strategy, but it shows that the pretrained backbone in principle already supports near-best performance, even though only the 16 task-feature parameters need to be adapted, rather than the roughly 800 parameters of an equivalent individually trained task model. However, the gap from mean initialization further shows that this solution is not generally recovered by feature optimization from an arbitrary point, including across all learning rates $\eta \in \{10^{-4}, 10^{-3}, 5 \times 10^{-3}, 10^{-2}, 5 \times 10^{-2}\}$ tested, indicating a non-smooth loss landscape.

Since the same procedure (initializing fine-tuning from another task's feature vector) can be repeated for every source-target pair, it also provides a measure of how useful each task representation is as an initialization, where we quantified training benefit using the same measure from the previous section (Appx. A.2). We then related source quality to task isolation, defined as the mean distance of a task from the other tasks in each similarity space. Source quality was negatively correlated with isolation in all three spaces: $\rho = -0.74$ for task geometry, $\rho = -0.86$ for task features, and $\rho = -0.53$ for linear region usage, with $p < .001$ in all cases. The relationship in the hierarchical feature space is partly expected because adaptation is performed by optimizing coordinates in that space. However, the corresponding associations with independently measured geometry and linear region usage similarity show that task positions learned during joint training more generally carry information about the functional reusability of those positions for adapting to other tasks. Thus, the same pre-training and adaptation framework both supports learning new tasks without forgetting and indicates which task representations are useful for successful transfer learning.

5 Discussion

We trained a hierarchical AL-RNN jointly on 100 heterogeneous cognitive tasks, using it to study how task computations interact within a shared dynamical backbone. We identified different mechanisms for separating and combining tasks through hierarchical task features, linear region usage, and task geometry. Together, these mechanisms allow the model to reuse shared structure while avoiding negative interference. The resulting task spaces provide several reproducible but distinct measures of task similarity. This offers a representation-based alternative to relying on task names or verbal taxonomies, which may obscure the computational relationships between tasks, and a common interface for comparing task representations across models.

Our results also highlight challenges that become especially important as multitask recurrent models are scaled. The framework reveals substantial heterogeneity in individual task complexity, particularly in the amount of nonlinear capacity required for learning, even when tasks appear similarly demanding from their verbal descriptions. Individual tasks can be simple while their joint implementation requires additional model complexity, and increasing the number and diversity of tasks can increase the need for such capacity. Transfer is consequently not uniformly positive, with some tasks systematically interfering with others. Task isolation in the derived task similarity spaces relates to negative transfer in pairwise pre-training. We further find that task-conditioned parameterization provides an efficient adaptation mechanism, where new tasks can be learned by modifying only a low-dimensional feature vector, avoiding catastrophic forgetting by construction. This suggests a route toward reusable dynamical backbones for cognitive computation, although the loss landscape for this adaptation is sufficiently irregular that success depends strongly on initialization. The results on negative transfer present an interesting avenue for comparing model-predicted task interactions with neural data: for example, do isolated tasks in the model also predict slower adaptation when humans or animals learn them sequentially?

To facilitate adaptation and identify compatible task pairings, future work could learn an encoder that provides informative priors for task-feature initialization (Vermani et al., 2024), building on task-embedding methods such as Task2Vec (Achille et al., 2019). Task features could also be organized according to dynamical properties (Trede et al., 2026), potentially improving both adaptation and interpretability. Finally, comparing the learned task representations with complementary characterizations based directly on dynamics may clarify which notions of task similarity generalize across methods (Ostrow et al., 2023; Huang et al., 2026).

A limitation of the present study is that its conclusions are based on a particular task battery and model class, while the field still lacks standardized benchmarks for evaluating transfer and interference across cognitive tasks. Moreover, the same task may admit multiple dynamical solutions (Qian and Pehlevan, 2025; Tolmachev and Engel, 2025), and the representations identified here may depend on architecture and optimization. We therefore view the task suite, evaluation, and pairwise transfer protocols introduced here as a contribution towards a reproducible basis for comparing multitask models. Extending these comparisons across architectures, learning rules, and task batteries will be necessary to determine which findings generalize.

Reproducibility Statement

The code for this paper is available at https://github.com/anonymous-neuroscience-capibara/multitask_interactions_100_tasks.git.

AI use statement

In this work, we used generative AI tools to assist with the initial generation and implementation of candidate task paradigms. Candidate tasks, their literature sources, and their final implementations were reviewed and verified by the authors. The model architecture and training framework were built on existing implementations from the literature, and AI assistance was used primarily for processing results and implementing visualizations. For writing, generative AI was used for editing and phrasing suggestions based on author-provided structure and content. We reviewed all AI-assisted code, analyses, figures, and text, and take full responsibility for the final content, claims, and artifacts of this work.

References

Alessandro Achille, Michael Lam, Rahul Tewari, Avinash Ravichandran, Subhransu Maji, Charless Fowlkes, Stefano Soatto, and Pietro Perona. Task2Vec: Task Embedding for Meta-Learning, February 2019. URL http://arxiv.org/abs/1902.03545. arXiv:1902.03545 [cs.LG].

David Badre and Mark D'Esposito. Functional magnetic resonance imaging evidence for a hierarchical organization of the prefrontal cortex. *Journal of Cognitive Neuroscience*, 19:2082–2099, 2007. doi: 10.1162/jocn.2007.19.12.2082.

David Badre and Mark D'Esposito. Is the rostro-caudal axis of the frontal lobe hierarchical? *Nature Reviews Neuroscience*, 10:659–669, 2009. doi: 10.1038/nrn2667.

Andrea Banino, Caswell Barry, Benigno Uria, Charles Blundell, Timothy Lillicrap, et al. Vector-based navigation using grid-like representations in artificial agents. *Nature*, 557:429–433, 2018. doi: 10.1038/s41586-018-0102-6.

Ziv Bar-Joseph, David K. Gifford, and Tommi S. Jaakkola. Fast optimal leaf ordering for hierarchical clustering. *Bioinformatics*, 17(suppl):S22–S29, 2001. doi: 10.1093/bioinformatics/17.suppl_1.S22.

Marcel Binz, Elif Akata, Matthias Bethge, Franziska Brändle, Fred Callaway, Julian Coda-Forno, Peter Dayan, Can Demircan, Maria K. Eckstein, Noémi Éltető, Thomas L. Griffiths, Susanne Haridi, Akshay K. Jagadish, Li Ji-An, Alexander Kipnis, Sreejan Kumar, Tobias Ludwig, Marvin Mathony, Marcelo Mattar, Alireza Modirshanechi, Surabhi S. Nath, Joshua C. Peterson, Milena Rmus, Evan M. Russek, Tankred Saanum, Johannes A. Schubert, Luca M. Schulze Buschoff, Nishad Singhi, Xin Sui, Mirko Thalmann, Fabian J. Theis, Vuong Truong, Vishaal Udandarao, Konstantinos Voudouris, Robert Wilson, Kristin Witte, Shuchen Wu, Dirk U. Wulff, Huadong Xiong, and Eric Schulz. A foundation model to predict and capture human cognition. *Nature*, 644 (8078):1002–1009, August 2025. ISSN 1476-4687. doi: 10.1038/s41586-025-09215-4. URL https://www.nature.com/articles/s41586-025-09215-4.

Bruno G. Breitmeyer and Lawrence Ganz. Implications of sustained and transient channels for theories of visual pattern masking, saccadic suppression, and information processing. *Psychological Review*, 83:1–36, 1976. doi: 10.1037/0033-295X.83.1.1.

Manuel Brenner and Georgia Koppe. Uncovering the Computational Roles of Nonlinearity in Sequence Modeling Using Almost-Linear RNNs. *Transactions on Machine Learning Research*, 2026. ISSN 2835-8856. URL https://openreview.net/forum?id=qI2Vt9P9rl.

Manuel Brenner, Christoph Jürgen Hemmer, Zahra Monfared, and Daniel Durstewitz. Almost-Linear RNNs Yield Highly Interpretable Symbolic Codes in Dynamical Systems Reconstruction. *Advances in Neural Information Processing Systems*, 37:36829–36868, December 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/40cf27290cc2bd98a428b567ba25075c-Abstract-Conference.html.

Manuel Brenner, Elias Weber, Georgia Koppe, and Daniel Durstewitz. Learning Interpretable Hierarchical Dynamical Systems Models from Time Series Data. In *International Conference on Learning Representations (ICLR)*, 2025. URL https://openreview.net/forum?id=Vp2OAxMs2s.

Kenneth H. Britten, Michael N. Shadlen, William T. Newsome, and J. Anthony Movshon. The analysis of visual motion: a comparison of neuronal and psychophysical performance. *Journal of Neuroscience*, 12(12):4745–4765, 1992. doi: 10.1523/JNEUROSCI.12-12-04745.1992.

Benjamin W. Brunton, Matthew M. Botvinick, and Carlos D. Brody. Rats and humans can optimally accumulate evidence for decision-making. *Science*, 340:95–98, 2013. doi: 10.1126/science.1233912.

Russell M. Church and Mary Z. Deluty. Bisection of temporal intervals. *Journal of Experimental Psychology: Animal Behavior Processes*, 3:216–228, 1977. doi: 10.1037/0097-7403.3.3.216.

Jonathan D. Cohen, Deanna M. Barch, Cameron Carter, and David Servan-Schreiber. Context-processing deficits in schizophrenia: Converging evidence from three theoretically motivated cognitive tasks. *Journal of Abnormal Psychology*, 108:120–133, 1999. doi: 10.1037/0021-843X.108.1.120.

Giulia Crocioni, Alexander Hadjiivanov, Nathan Cloos, Caterina Barezzi, Sara Fuentes, Joao Barbosa, Dani L Bodor, Parva Alavian, Jordi Pastor-Ciurana, Marta Fradera, RU-YUAN ZHANG, Jeremy Forest, Jorge del Pozo Lerida, Li Ji-An, Christopher J Cueva, Jaime de la Rocha, Devika Narain, Guangyu R Yang, Manuel Molano-Mazon, and Jorge F Mejias. NeuroGym: An open resource for developing and sharing neuroscience tasks, June 2026. URL https://osf.io/preprints/psyarxiv/aqc9n_v2/.

Christopher J. Cueva and Xue-Xin Wei. Emergence of grid-like representations by training recurrent neural networks to perform spatial localization. In *International Conference on Learning Representations*, 2018.

Laura N. Driscoll, Krishna Shenoy, and David Sussillo. Flexible multitask computation in recurrent networks utilizes shared dynamical motifs. *Nature Neuroscience*, 27(7):1349–1363, July 2024. ISSN 1546-1726. doi: 10.1038/s41593-024-01668-6. URL https://www.nature.com/articles/s41593-024-01668-6.

Alexis Dubreuil, Adrian Valente, Manuel Beiran, Francesca Mastrogiuseppe, and Srdjan Ostojic. The role of population structure in computations through neural dynamics. *Nature Neuroscience*, 25(6):783–794, June 2022. ISSN 1546-1726. doi: 10.1038/s41593-022-01088-4. URL https://www.nature.com/articles/s41593-022-01088-4.

Lea Duncker, Laura Driscoll, Krishna V Shenoy, Maneesh Sahani, and David Sussillo. Organizing recurrent network dynamics by task-computation to enable continual learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 14387–14397. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/hash/a576eafbce762079f7d1f77fca1c5cc2-Abstract.html.

Daniel Durstewitz. A state space approach for piecewise-linear recurrent neural networks for identifying computational dynamics from neural measurements. *PLOS Computational Biology*, 13(6):e1005542, June 2017. ISSN 1553-7358. doi: 10.1371/journal.pcbi.1005542. URL https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1005542. Publisher: Public Library of Science.

Daniel Durstewitz, Georgia Koppe, and Max Ingo Thurm. Reconstructing computational system dynamics from neural data with recurrent neural networks. *Nature Reviews Neuroscience*, 24(11):693–710, November 2023. ISSN 1471-0048. doi: 10.1038/s41583-023-00740-7. URL https://www.nature.com/articles/s41583-023-00740-7. Publisher: Nature Publishing Group.

Daniel Durstewitz, Bruno Averbeck, and Georgia Koppe. What Neuroscience Can Teach AI About Learning in Continuously Changing Environments, July 2025. URL http://arxiv.org/abs/2507.02103. arXiv:2507.02103 [cs.AI].

Stephen W. Egger, Evan D. Remington, Chia-Jung Chang, and Mehrdad Jazayeri. Internal models of sensorimotor integration regulate cortical dynamics. *Nature Neuroscience*, 22:1871–1882, 2019. doi: 10.1038/s41593-019-0500-6.

Benjamin Ehret, Christian Henning, Maria Cervera, Alexander Meulemans, Johannes von Oswald, and Benjamin F. Grewe. Continual learning in recurrent neural networks. In *International Conference on Learning Representations*, October 2020. URL https://openreview.net/forum?id=8xeBUgD8u9.

Barbara A. Eriksen and Charles W. Eriksen. Effects of noise letters upon the identification of a target letter in a nonsearch task. *Perception & Psychophysics*, 16:143–149, 1974. doi: 10.3758/BF03203267.

Joël Fagot, Edward A. Wasserman, and Michael E. Young. Discriminating the relation between relations: The role of entropy in abstract conceptualization by baboons (Papio papio) and humans (Homo sapiens). *Journal of Experimental Psychology: Animal Behavior Processes*, 27(4):316–328, 2001. doi: 10.1037/0097-7403.27.4.316.

Chris Fifty, Ehsan Amid, Zhe Zhao, Tianhe Yu, Rohan Anil, and Chelsea Finn. Efficiently Identifying Task Groupings for Multi-Task Learning. In *Advances in Neural Information Processing Systems*, volume 34, pages 27503–27516. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper/2021/hash/e77910ebb93b511588557806310f78f1-Abstract.html.

Timo Flesch, Keno Juechems, Tsvetomira Dumbalska, Andrew Saxe, and Christopher Summerfield. Orthogonal representations for robust context-dependent task performance in brains and neural networks. *Neuron*, 110(7):1258–1270.e11, April 2022. ISSN 0896-6273. doi: 10.1016/j.neuron.2022.01.005. URL https://www.sciencedirect.com/science/article/pii/S0896627322000058.

David J. Freedman and John A. Assad. Experience-dependent representation of visual categories in parietal cortex. *Nature*, 443(7107):85–88, 2006. doi: 10.1038/nature05078.

Shintaro Funahashi, Charles J. Bruce, and Patricia S. Goldman-Rakic. Mnemonic coding of visual space in the monkey's dorsolateral prefrontal cortex. *Journal of Neurophysiology*, 61:331–349, 1989. doi: 10.1152/jn.1989.61.2.331.

Apostolos P. Georgopoulos, Joseph T. Lurito, Michael Petrides, Andrew B. Schwartz, and Joe T. Massey. Mental rotation of the neuronal population vector. *Science*, 243(4888):234–236, 1989. doi: 10.1126/science.2911737.

Anirudh Goyal, Alex Lamb, Jordan Hoffmann, Shagun Sodhani, Sergey Levine, Yoshua Bengio, and Bernhard Schölkopf. Recurrent Independent Mechanisms. October 2020. URL https://openreview.net/forum?id=mLcmdlEUxy-.

David M. Green and John A. Swets. *Signal Detection Theory and Psychophysics*. Wiley, New York, 1966.

I. C. Griffin and Anna C. Nobre. Orienting attention to locations in internal representations. *Journal of Cognitive Neuroscience*, 15:1176–1194, 2003. doi: 10.1162/089892903322598139.

Michael J. Hacker. Speed and accuracy of recency judgments for events in short-term memory. *Journal of Experimental Psychology: Human Learning and Memory*, 6(6):651–675, 1980. doi: 10.1037/0278-7393.6.6.651.

Doug P. Hanes, William F. Patterson, and Jeffrey D. Schall. Role of frontal eye fields in countermanding saccades: Visual, movement, and fixation activity. *Journal of Neurophysiology*, 79:817–834, 1998. doi: 10.1152/jn.1998.79.2.817.

Timothy D. Hanks, Charles D. Kopec, Benjamin W. Brunton, Caroline A. Duan, Jeffrey C. Erlich, and Carlos D. Brody. Distinct relationships of parietal and prefrontal cortices to evidence accumulation. *Nature*, 520:220–223, 2015. doi: 10.1038/nature14066.

Ira J. Hirsh and Carl E. Sherrick. Perceived order in different sense modalities. *Journal of Experimental Psychology*, 62(5):423–432, 1961. doi: 10.1037/h0045283.

Ann Huang, Mitchell Ostrow, Satpreet H. Singh, Leo Kozachkov, Ila Fiete, and Kanaka Rajan. InputDSA: Demixing, then comparing recurrent and externally driven dynamics. *International Conference on Learning Representations*, 2026:55010–55053, April 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/5a437d77270d5588a84d63a851a3fdeb-Abstract-Conference.html.

Mehrdad Jazayeri and Michael N. Shadlen. Temporal context calibrates interval timing. *Nature Neuroscience*, 13:1020–1026, 2010. doi: 10.1038/nn.2590.

Philip N. Johnson-Laird and Jane Tagart. How implication is understood. *American Journal of Psychology*, 82:367–373, 1969. doi: 10.2307/1420752.

Mikail Khona and Ila R. Fiete. Attractor and integrator networks in the brain. *Nature Reviews Neuroscience*, 23(12):744–766, December 2022. ISSN 1471-0048. doi: 10.1038/s41583-022-00642-0. URL https://www.nature.com/articles/s41583-022-00642-0. Publisher: Nature Publishing Group.

Mikail Khona, Sarthak Chandra, Joy J. Ma, and Ila R. Fiete. Winning the Lottery With Neural Connectivity Constraints: Faster Learning Across Cognitive Tasks With Spatially Constrained Sparse RNNs. *Neural Computation*, 35(11):1850–1869, October 2023. ISSN 0899-7667. doi: 10.1162/neco_a_01613. URL https://doi.org/10.1162/neco_a_01613.

Sung Soo Kim, Hervé Rouault, Shaul Druckmann, and Vivek Jayaraman. Ring attractor dynamics in the Drosophila central brain. *Science*, 356:849–853, 2017. doi: 10.1126/science.aal4835.

Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *arXiv:1412.6980 [cs]*, January 2017. URL http://arxiv.org/abs/1412.6980. arXiv: 1412.6980.

Eric I. Knudsen and Masakazu Konishi. A neural map of auditory space in the owl. *Science*, 200: 795–797, 1978. doi: 10.1126/science.644324.

Vitaly Kurin, Alessandro De Palma, Ilya Kostrikov, Shimon Whiteson, and Pawan K Mudigonda. In Defense of the Unitary Scalarization for Deep Multi-Task Learning. In *Advances in Neural Information Processing Systems*, volume 35, pages 12169–12183. Curran Associates, Inc., 2022. doi: 10.52202/068431-0884. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/4f301ae934f396086bfefd1139039dbd-Abstract-Conference.html.

Matthew I. Leon and Michael N. Shadlen. Representation of time by neurons in the posterior parietal cortex of the macaque. *Neuron*, 38:317–327, 2003. doi: 10.1016/S0896-6273(03)00185-5.

Gordon D. Logan and William B. Cowan. On the ability to inhibit thought and action: A theory of an act of control. *Psychological Review*, 91:295–327, 1984. doi: 10.1037/0033-295X.91.3.295.

Steven J. Luck and Edward K. Vogel. The capacity of visual working memory for features and conjunctions. *Nature*, 390:279–281, 1997. doi: 10.1038/36846.

Jiaqi Ma, Zhe Zhao, Xinyang Yi, Jilin Chen, Lichan Hong, and Ed H. Chi. Modeling Task Relationships in Multi-task Learning with Multi-gate Mixture-of-Experts. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, pages 1930–1939, New York, NY, USA, July 2018. Association for Computing Machinery. ISBN 978-1-4503-5552-0. doi: 10.1145/3219819.3220007. URL https://dl.acm.org/doi/10.1145/3219819.3220007.

Valerio Mante, David Sussillo, Krishna V. Shenoy, and William T. Newsome. Context-dependent computation by recurrent dynamics in prefrontal cortex. *Nature*, 503(7474):78–84, November 2013. ISSN 1476-4687. doi: 10.1038/nature12742. URL https://www.nature.com/articles/nature12742. Publisher: Nature Publishing Group.

David E. Meyer and Roger W. Schvaneveldt. Facilitation in recognizing pairs of words: Evidence of a dependence between retrieval operations. *Journal of Experimental Psychology*, 90:227–234, 1971. doi: 10.1037/h0031564.

Earl K. Miller, Charles A. Erickson, and Robert Desimone. Neural mechanisms of visual working memory in prefrontal cortex of the macaque. *Journal of Neuroscience*, 16:5154–5167, 1996. doi: 10.1523/JNEUROSCI.16-16-05154.1996.

Akira Miyake, Naomi P. Friedman, Michael J. Emerson, Alexander H. Witzki, Amy Howerter, and Tor D. Wager. The unity and diversity of executive functions and their contributions to complex "frontal lobe" tasks: A latent variable analysis. *Cognitive Psychology*, 41:49–100, 2000. doi: 10.1006/cogp.1999.0734.

Douglas P. Munoz and Stefan Everling. Look away: The anti-saccade task and the voluntary control of eye movement. *Nature Reviews Neuroscience*, 5(3):218–228, 2004. doi: 10.1038/nrn1345.

Risto Näätänen, A. W. K. Gaillard, and S. Mäntysalo. Early selective-attention effect on evoked potential reinterpreted. *Acta Psychologica*, 42:313–329, 1978. doi: 10.1016/0001-6918(78)90006-9.

Andreas Nieder, Ilka Diester, and Ovidiu Tudusciuc. Temporal and spatial enumeration processes in the primate parietal cortex. *Science*, 313:1431–1435, 2006. doi: 10.1126/science.1130308.

Mitchell Ostrow, Adam Eisen, Leo Kozachkov, and Ila Fiete. Beyond Geometry: Comparing the Temporal Structure of Computation in Neural Circuits with Dynamical Similarity Analysis. In *Advances in Neural Information Processing Systems*, volume 36, pages 33824–33837. Curran Associates, Inc., 2023. doi: 10.52202/075280-1467. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/6ac807c9b296964409b277369e55621a-Abstract-Conference.html.

Reinier Plomp. Rate of decay of auditory sensation. *The Journal of the Acoustical Society of America*, 36(2):277–282, 1964. doi: 10.1121/1.1918946.

Michael I. Posner. Orienting of attention. *Quarterly Journal of Experimental Psychology*, 32:3–25, 1980. doi: 10.1080/00335558008248231.

William Qian and Cengiz Pehlevan. Discovering alternative solutions beyond the simplicity bias in recurrent neural networks, September 2025. URL http://arxiv.org/abs/2509.21504. arXiv:2509.21504 [q-bio.NC].

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. ISSN 1533-7928. URL http://jmlr.org/papers/v21/20-074.html.

Jane E. Raymond, Kenneth L. Shapiro, and Karen M. Arnell. Temporary suppression of visual processing in an RSVP task: An attentional blink? *Journal of Experimental Psychology: Human Perception and Performance*, 18:849–860, 1992. doi: 10.1037/0096-1523.18.3.849.

Reidar Riveland and Alexandre Pouget. Natural language instructions induce compositional generalization in networks of neurons. *Nature Neuroscience*, 27(5):988–999, May 2024. ISSN 1546-1726. doi: 10.1038/s41593-024-01607-5. URL https://www.nature.com/articles/s41593-024-01607-5.

Ranulfo Romo, Carlos D. Brody, Adrián Hernández, and Luis Lemus. Neuronal correlates of parametric working memory in the prefrontal cortex. *Nature*, 399(6735):470–473, 1999. doi: 10.1038/20939.

Clemens Rosenbaum, Tim Klinger, and Matthew Riemer. Routing Networks: Adaptive Selection of Non-Linear Functions for Multi-Task Learning. In *International Conference on Learning Representations*, February 2018. URL https://openreview.net/forum?id=ry8dvM-R-.

Herbert Rubenstein, Linda Garfield, and Janet A. Millikan. Homographic entries in the internal lexicon. *Journal of Verbal Learning and Verbal Behavior*, 9:487–494, 1970. doi: 10.1016/S0022-5371(70)80091-3.

Katsuyuki Sakai. Task set and prefrontal cortex. *Annual Review of Neuroscience*, 31:219–245, 2008. doi: 10.1146/annurev.neuro.31.060407.125642.

Keisuke Sakai and Yasushi Miyashita. Neural organization for the long-term memory of paired associates. *Nature*, 354:152–155, 1991. doi: 10.1038/354152a0.

Dominik Schmidt, Georgia Koppe, Zahra Monfared, Max Beutelspacher, and Daniel Durstewitz. Identifying nonlinear dynamical systems with multiple time scales and long-range dependencies. In *International Conference on Learning Representations (ICLR)*, 2021. URL https://openreview.net/forum?id=_XYzwxPIQu6.

Ozan Sener and Vladlen Koltun. Multi-Task Learning as Multi-Objective Optimization. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper/2018/hash/432aca3a1e345e339f35a30c8f65edce-Abstract.html.

David Servan-Schreiber, Jonathan D. Cohen, and Scott Steingard. Schizophrenic deficits in the processing of context: A test of a theoretical model. *Archives of General Psychiatry*, 53:1105–1112, 1996. doi: 10.1001/archpsyc.1996.01830120037008.

H. Sebastian Seung. How the brain keeps the eyes still. *Proceedings of the National Academy of Sciences*, 93:13339–13344, 1996. doi: 10.1073/pnas.93.23.13339.

Roger N. Shepard and Jacqueline Metzler. Mental rotation of three-dimensional objects. *Science*, 171:701–703, 1971. doi: 10.1126/science.171.3972.701.

James R. Simon. Reactions toward the source of stimulation. *Journal of Experimental Psychology*, 81:174–176, 1969. doi: 10.1037/h0027448.

Min-Ho Sohn, Stefan Ursu, John R. Anderson, Victor A. Stenger, and Cameron S. Carter. The role of prefrontal cortex and posterior parietal cortex in task switching. *Proceedings of the National Academy of Sciences*, 97:13448–13453, 2000. doi: 10.1073/pnas.240460497.

Robert R. Sokal and Charles D. Michener. A statistical method for evaluating systematic relationships. *University of Kansas Science Bulletin*, 38:1409–1438, 1958.

Trevor Standley, Amir Zamir, Dawn Chen, Leonidas Guibas, Jitendra Malik, and Silvio Savarese. Which Tasks Should Be Learned Together in Multi-task Learning? In *Proceedings of the 37th International Conference on Machine Learning*, pages 9120–9132. PMLR, November 2020. URL https://proceedings.mlr.press/v119/standley20a.html.

Saul Sternberg. High-speed scanning in human memory. *Science*, 153:652–654, 1966. doi: 10.1126/science.153.3736.652.

John R. Stroop. Studies of interference in serial verbal reactions. *Journal of Experimental Psychology*, 18:643–662, 1935. doi: 10.1037/h0054651.

David Sussillo and Omri Barak. Opening the black box: low-dimensional dynamics in high-dimensional recurrent neural networks. *Neural Computation*, 25(3):626–649, March 2013. ISSN 1530-888X. doi: 10.1162/NECO_a_00409.

Pavel Tolmachev and Tatiana A. Engel. Single-unit activations confer inductive biases for emergent circuit solutions to cognitive tasks. *Nature Machine Intelligence*, 7(10):1742–1754, October 2025. ISSN 2522-5839. doi: 10.1038/s42256-025-01127-2. URL https://www.nature.com/articles/s42256-025-01127-2.

Georg Trede, Charlotte Ricarda Doll, Elias Weber, and Daniel Durstewitz. Topological Out-of-Domain Generalization in Dynamical Systems Reconstruction, June 2026. URL http://arxiv.org/abs/2606.22969. arXiv:2606.22969 [cs.LG].

Anne M. Treisman and Garry Gelade. A feature-integration theory of attention. *Cognitive Psychology*, 12:97–136, 1980. doi: 10.1016/0010-0285(80)90005-5.

Elia Turner and Omri Barak. The Simplicity Bias in Multi-Task RNNs: Shared Attractors, Reuse of Dynamics, and Geometric Representation. In *Advances in Neural Information Processing Systems*, volume 36, pages 25495–25507. Curran Associates, Inc., 2023. doi: 10.52202/075280-1109. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/50d6dbc809b0dc96f7f1090810537acc-Abstract-Conference.html.

Nachum Ulanovsky, L. Las, and Israel Nelken. Processing of low-probability sounds by cortical neurons. *Nature Neuroscience*, 6:391–398, 2003. doi: 10.1038/nn1032.

Ayesha Vermani, Josue Nassar, Hyungju Jeon, Matthew Dowling, and Il Memming Park. Meta-Dynamical State Space Models for Integrative Neural Data Analysis. In *International Conference on Learning Representations*, October 2024. URL https://openreview.net/forum?id=SRpq5OBpED.

Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, et al. SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nature Methods*, 17:261–272, 2020. doi: 10.1038/s41592-019-0686-2.

Johannes von Oswald, Christian Henning, Benjamin F. Grewe, and João Sacramento. Continual learning with hypernetworks. In *International Conference on Learning Representations*, September 2019. URL https://openreview.net/forum?id=SJgwNerKvB.

Peter C. Wason. Reasoning about a rule. *Quarterly Journal of Experimental Psychology*, 20:273–281, 1968. doi: 10.1080/14640746808400161.

Guangyu Robert Yang, Madhura R. Joglekar, H. Francis Song, William T. Newsome, and Xiao-Jing Wang. Task representations in neural networks trained to perform many cognitive tasks. *Nature Neuroscience*, 22(2):297–306, February 2019. ISSN 1546-1726. doi: 10.1038/s41593-018-0310-2.

Douglas B. Yntema. Keeping track of several things at once. *Human Factors*, 5:7–17, 1963. doi: 10.1177/001872086300500102.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient Surgery for Multi-Task Learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 5824–5836. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/hash/3fe78a8acf5fda99de95303940a2420c-Abstract.html.

Wen Zhang, Lingfei Deng, Lei Zhang, and Dongrui Wu. A Survey on Negative Transfer. *IEEE/CAA Journal of Automatica Sinica*, 10(2):305–329, 2023. ISSN 2329-9266. doi: 10.1109/JAS.2022.106004. URL https://www.ieee-jas.net/en/article/doi/10.1109/JAS.2022.106004.

A Methodological Details

A.1 Task Suite

We train on a battery of 100 supervised cognitive tasks spanning working memory, decision-making, perception, attention, timing, reasoning, and motor response. The battery comprises 66 base task identities and 34 compositional variants generated from selected bases using four modifiers. Each modified variant is treated as a separate task with its own task embedding.

All tasks share a common trial interface. Inputs are zero-padded to a common input dimension, and targets are represented in a fixed five-channel readout comprising a fixation channel and task-dependent behavioural channels. Trials may contain context, stimulus, delay, and response phases, with phase durations resampled across trials.

Table 6 lists the 66 base task identities and their readout types. The modifiers are defined in Sec. C.1. Here, *angular* denotes a directional output represented by $(\cos\theta, \sin\theta)$, *binary* a thresholded decision, *scalar* a one-dimensional continuous output, *vector* multiple continuous coordinates, *categorical* a choice among multiple alternatives, and *bits* the signs of multiple binary outputs. A star (*) marks outputs scored at every response timestep.

Because the task battery contains tasks with different temporal structures and readouts, we use a common masked-MSE objective for joint training, with task-specific masks selecting the behaviourally relevant timesteps and readout channels, including the fixation channel:

$$\mathcal{L} = \frac{1}{B}\sum_{b=1}^{B} \frac{1}{L^{(0)}_{k(b)}} \frac{1}{|M_b|\,|C_b|} \sum_{t\in M_b}\sum_{c\in C_b} \left(\hat{y}_{b,t,c} - y_{b,t,c}\right)^2, \tag{4}$$

where M_b denotes the scored time steps, C_b the scored output channels for trial b, $k(b)$ is the task of trial b and $L^{(0)}_k$ is the mean of the unweighted per-trial loss (the inner term of Eq. 4) over task k's training trials at initialization, computed once before training and held fixed. Normalizing each trial by the number of scored time-channel entries prevents tasks with longer sequences or more output channels from dominating the joint gradient, and dividing by $L^{(0)}_k$ expresses each task's loss relative to its value at initialization, so that tasks whose targets differ in magnitude contribute on a comparable scale. We use no task-specific loss function.

We report accuracy separately from the training loss using a task-specific, trial-level success criterion. For single-response tasks, a trial is counted as correct if the final scored output lies within the prescribed tolerance of the target. Reaction-time tasks are evaluated at the response timestep following stimulus onset. For sequential tasks, accuracy is the fraction of scored timesteps satisfying the corresponding criterion. Following Yang et al. (2019), we set the angular threshold to 36°. To make scores comparable across response types, we use angular tolerances for directional outputs and a scalar/vector tolerance set to $0.35\,\sigma$ of the task's target distribution. This mapping between output tolerances and trial success is necessarily a modeling assumption, so we assess the robustness of this choice by recomputing battery accuracy across a range of angular thresholds (Table 3). Although the absolute accuracy changes and the chosen 36° criterion produces a mild ceiling effect, the relative ordering of model capacities is preserved across thresholds. In addition to mean accuracy across tasks, we report the number of solved tasks, defining a task as solved when its accuracy $> 80\%$.

A.2 Model Training

For each task, we used 200 training trials and 256 evaluation trials per task, with balanced task indices ensuring that every task contributed equally. Trials were sampled in batches of 64 and each epoch traversed the complete training pool once without replacement. Since trials differed in length and channel dimensionality, batches were zero-padded to their maximum dimensions and accompanied by timestep masks, so padding did not contribute to the masked-MSE objective defined above.

The principal model-capacity hyperparameters were the latent dimension M, the number of nonlinear units P, and the task-feature dimension N_{feat}. We varied P and N_{feat} in the main capacity experiments. In preliminary experiments, performance largely plateaued at $M = 64$, and larger latent

Response type	Correct if
Angular	$\lvert\theta - \theta^*\rvert < 36°$
Binary	output and target on same side of 0.5
Scalar / vector	readout error $< 0.35\,\sigma$
K-way categorical	argmax over readout channels matches target
Sequence production	$\lvert\theta_t - \theta_t^*\rvert < 36°$ at each step
Flip-flop	correct sign on every bit channel at each step

Table 2: Trial-level accuracy criteria by response type. θ denotes the angle decoded from the $(\cos\theta, \sin\theta)$ output pair; σ the standard deviation of the task's target distribution. Single-response tasks are evaluated at the last scored timestep; sequential tasks are scored as the fraction of scored timesteps meeting the criterion.

spaces yielded similar accuracy. We therefore fixed $M = 64$ for the main experiments. Initialization and optimization settings were chosen following the AL-RNN training procedure of Brenner et al. (2025).

For the hierarchical models, task features and task-specific noise covariances were optimized jointly with the shared parameters using Adam (Kingma and Ba, 2017). We implemented separate parameter groups for task-specific and shared parameters, with a common learning rate of 5×10^{-4} and weight decay of 10^{-4}. The shared recurrent projection was initialized with Xavier gain 0.05.

For training the AL-RNN, the objective also included the manifold attractor regularization (MAR, Schmidt et al., 2021) with ($\tau = 0.01$), applied to the first half of the recurrent units. This regularization encourages self-recurrent weights toward one and off-diagonal recurrent weights and biases toward zero, implementing slow time scales commonly useful in computational tasks (Schmidt et al., 2021), and incorporated as an architectural bias into LSTMs and GRUs. The total objective was Eq. 4 plus this regularization term. Gradients were clipped to a global norm of 10, and updates were skipped if the post-clipping norm was non-finite.

Learning rates were linearly warmed up over 100 epochs and then exponentially decayed by a factor of 0.999 per epoch. If a non-finite loss or gradient was detected, the learning-rate scale was halved for the remainder of the run. We evaluated the complete validation set after every epoch and tracked total loss and per-task accuracy. The checkpoint with the lowest validation loss was retained. After the first 300 epochs, we used early stopping with a patience of 300 epochs without improvement, training for up to 3000 epochs in total. All reported accuracies were scored on an independent test set with 256 samples per task.

For the leave-one-out experiments, we trained one hierarchical model per held-out task on the remaining 99 tasks ($M = 64$, $P = 16$, $N_{\text{feat}} = 16$), using the same data, optimization settings and stopping criteria as above, for three seeds each (300 models in total). To adapt a model to its held-out task, we froze all shared parameters and optimized only the new task's feature vector, with a learning rate of 10^{-2} and a batch size of 64 on 200 training trials, for up to 700 epochs with early stopping, evaluating on 128 test trials per epoch.

The flat joint models consisted of a single AL-RNN with one set of shared parameters trained on all 100 tasks. Task identity was provided as a one-hot vector appended to the input. They were trained with the same optimization settings as the hierarchical models, except that 128 test trials per task were used.

Individual models were trained separately on each task as flat AL-RNNs with a single set of shared parameters and no task input ($M = 64$, $P \in \{0, 1, 2, 4, 8, 16\}$, 20, 50 or 200 training trials, five seeds). They used the same learning rate, weight decay, gradient clipping and MAR regularization as the hierarchical models, no learning-rate warm-up and up to 500 epochs. Early stopping was used with a patience of 150 epochs. For the pairwise transfer experiments, each trained source model was used to initialize all parameters of a model trained on each of the other 99 targets, using the same settings as the individual models, for five seeds.

Clustering Algorithm For the results in Figs. 3 and 9, we leveraged a joint hierarchical clustering across all three similarity spaces. Tasks were ordered by agglomerative hierarchical clustering with

average linkage (UPGMA, Sokal and Michener, 1958) on the dissimilarity $1 - s$, where s is the mean of the three seed-averaged similarity matrices, with each min-max normalized on its off-diagonal entries. The leaves were then reordered with optimal leaf ordering (Bar-Joseph et al., 2001), using the default implementation in SciPy (Virtanen et al., 2020). The resulting clustering-based task order used in Figs. 3 and 9 is: 1. FlipFlop2; 2. FlipFlop3; 3. CopyTask; 4. GoNogo; 5. GoNogo+delay; 6. HierReason; 7. TaskSwitch; 8. TaskSwitch+delay; 9. GapDetection; 10. DelayedComparison; 11. IntDisc; 12. DurationAnti; 13. DurationPro; 14. StopSignal; 15. CatPro; 16. CatPro+delay; 17. DMC; 18. DMSDistractor; 19. Match2Sample; 20. NonMatch2Sample; 21. CatAnti+delay; 22. CatAnti; 23. SymbolCategory; 24. SimultaneityJudgment; 25. TemporalOrderJudgment; 26. BackwardMasking; 27. DelayedCueReversal; 28. Stroop+delay; 29. Stroop; 30. Flanker; 31. Simon; 32. Flanker+delay; 33. Simon+delay; 34. AXCPT; 35. ConditionalReasoning; 36. DNMC; 37. PairedAssoc; 38. CtxIntMod1; 39. DelayPro; 40. PerceptualDM; 41. NoiseCleaner; 42. NoiseCleaner+delay; 43. PerceptualDM+delay; 44. MultiSens; 45. MultiSens+delay; 46. CtxIntMod2; 47. PriorIntegration; 48. NoiseCleaner+seq; 49. PerceptualDM+seq; 50. MultiSens+seq; 51. MultiSens+int+seq; 52. MultiSens+int; 53. PerceptualDM+int+seq; 54. NoiseCleaner+int+seq; 55. NoiseCleaner+int; 56. PerceptualDM+int; 57. HeadDirection; 58. DelayPro+seq; 59. DelayPro+int+seq; 60. DelayPro+int; 61. DelayAnti+int; 62. DelayAnti+int+seq; 63. DelayAnti+seq; 64. DelayAnti; 65. IntervalReproduction; 66. OneTwoThreeGo; 67. Numerosity; 68. AntiReach1D; 69. SoundLocalization; 70. PoissonClicks; 71. PluralityVote; 72. PathIntegration1D; 73. PathIntegration2D; 74. PosnerCueing; 75. WMUpdating; 76. RelationalMatch; 77. MentalRotation; 78. Sternberg; 79. ChangeDetect; 80. SerialOrderMemory; 81. VisualSearch; 82. SpatialWMRetro+int+seq; 83. SpatialWMRetro+int; 84. SpatialWMRetro; 85. SpatialWMRetro+seq; 86. DiscriminationOptOut; 87. SignalDetection; 88. TemporalBisection; 89. OddballDetection; 90. SineGen; 91. AttentionalBlink; 92. RTAnti; 93. RTAnti+seq; 94. RTAnti+int+seq; 95. RTAnti+int; 96. RTGo+int; 97. RTGo+int+seq; 98. RTGo+seq; 99. RTGo; 100. OddOneOut.

Although the three similarity spaces show moderate pairwise agreement, this shared structure does not support a single robust partition of the full task battery. Clustering the spaces independently yielded only weak agreement across cluster numbers (ARI $\approx$ 0.1-0.35 for k = 2-20). This partly reflects their different similarity distributions: feature similarity has higher and more structured off-diagonal values than geometry or linear region usage (see Fig. 3, and consequently contributes more strongly to the joint clustering. Moreover, silhouette analyses did not identify a stable preferred number of clusters. Even in the feature space, the highest scores were obtained by separating a small number of outlier tasks rather than by revealing a meaningful multiway partition. Nevertheless, targeted comparisons, such as the proximity of task variants to their base tasks, recover expected local structure, indicating that the spaces capture meaningful task relationships despite the absence of a single globally robust clustering.

Pairwise transfer between individually trained models To assess transfer independently of the shared multitask backbone, we evaluated all ordered pairs of individually trained task models. For each source task A and target task B, we first trained a model on A to convergence and then used its parameters to initialize training on B. We compared these source-initialized runs with training on B from the same random initialization distribution. All target runs used the same architecture, training data, evaluation data, and optimization schedule; only the source initialization differed. We quantified early learning using the negative mean log test loss over epochs 6–100,

$$S(A \to B) = -\frac{1}{95} \sum_{e=6}^{100} \log \mathcal{L}_{\text{test}}^{(e)}(A \to B), \tag{5}$$

such that larger values indicate lower test loss and more effective early adaptation. The first five epochs were excluded because they contained large initialization-dependent transients. For each target B, we standardized the 99 source-specific scores, $z(A \to B) = \frac{S(A \to B) - \mu_B}{\sigma_B}$, where μ_B and σ_B are the mean and standard deviation of $S(A \to B)$ across all 99 source tasks. Scores were computed for five random seeds and aggregated using the median across seeds. We define the *source quality* of task A as the mean of $z(A \to B)$ across all target tasks $B \neq A$.

We additionally evaluated training from random initialization as a reference source. Pretrained task models improved early learning relative to random initialization in approximately 70% of source–target pairs. However, this benefit was task-dependent: for some targets, training from scratch

outperformed initialization from most source tasks. Thus, transfer was beneficial on average but could act as a negative prior for particular target tasks.

To relate this result to the representations learned in the joint model, we quantified task isolation separately in each task-similarity space. For a task A, isolation was defined as its mean distance to all other tasks, standardized across the 100 tasks. We then computed Spearman correlations between source quality and isolation. Source quality was negatively associated with isolation in recurrent-state geometry ($\rho = -0.26$, $p = 0.01$) and nonlinear-region usage ($\rho = -0.23$, $p = 0.02$), indicating that tasks occupying more isolated representations in the joint model tended to provide less useful initializations for independently trained task models (see Fig. 11). These relationships were modest and do not fully explain pairwise transfer, but indicate that joint-model task organization captures some aspects of transferability.

Table 3: Robustness of mean battery accuracy to the angular accuracy threshold, averaged across 5 seeds. Values show mean test accuracy (%) for models with $N_{\text{feat}} = 16$. The 36° threshold is used in the main text. Although absolute accuracy increases with the tolerance, the relative ordering of nonlinear capacities is largely preserved. The final column reports the change between the most and least permissive thresholds.

	15°	20°	25°	30°	36°	40°	45°	50°	$\Delta 50 - 15$
P									
1	72.9	76.5	79.4	81.6	83.7	84.9	86.1	87.1	14.2
2	78.0	82.2	85.1	87.3	89.2	90.1	91.1	91.8	13.8
4	81.6	85.5	88.2	90.1	91.7	92.5	93.3	93.8	12.2
8	85.0	89.0	91.5	93.1	94.4	95.0	95.5	95.9	10.9
16	89.0	92.4	94.1	95.2	95.8	96.2	96.4	96.6	7.6
32	90.2	93.3	94.9	95.8	96.4	96.7	96.9	97.0	6.8
64	89.0	92.2	94.1	95.2	95.9	96.2	96.4	96.6	7.6

B FURTHER RESULTS

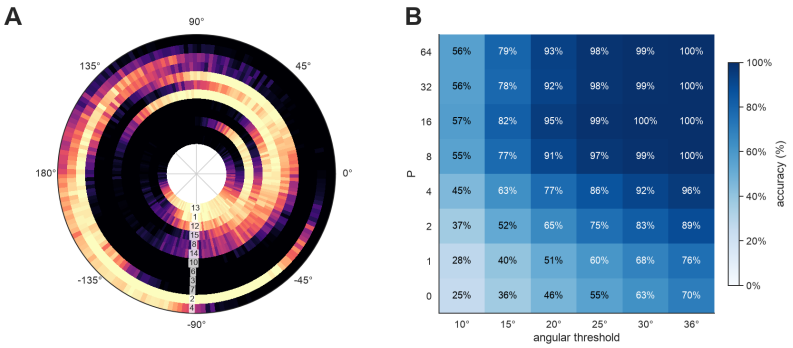


Figure 5: Hierarchical AL-RNN ($P = 16, N_{\text{feat}} = 16$) trained on the HeadDirection task. **A**: Probability that each ReLU unit is active, $P(\text{unit} > 0)$, as a function of heading (one concentric ring per unit, ordered by the onset of its active arc). Each unit switches on over a single contiguous arc, partitioning the circle into sectors whose boundaries tile the ring attractor (Fig. 4). An exactly continuous attractor would require a one-dimensional manifold of fixed points with marginal stability ($|\lambda| = 1$) everywhere along it. This is a measure-zero condition that is not learnable by gradient descent in practice. The network converges instead to a robust piecewise-linear approximation, where local dynamics within each region are either near-marginal (rotation-like, carrying the heading forward) or, in a few regions, weakly contracting toward an embedded fixed point (3 stable nodes, 2 saddles), producing slow drift toward these discrete resting headings. This tiling is fine enough to preserve near-perfect heading decoding ($R^2 \approx 1$ from the full state, $R^2 = 0.97$ from the bitcode alone) and the resulting drift remains small within typical trial lengths ($\sim$40 timesteps). **B**: Task accuracy as a function of the angular error threshold, for different numbers of nonlinear units, averaged across 5 seeds. As in our other threshold-ablation analyses (Table 3), the choice of accuracy threshold affects how well different architectures are distinguished, where progressively tighter thresholds reveal a graded improvement in ring-attractor approximation quality with increasing nonlinearity.

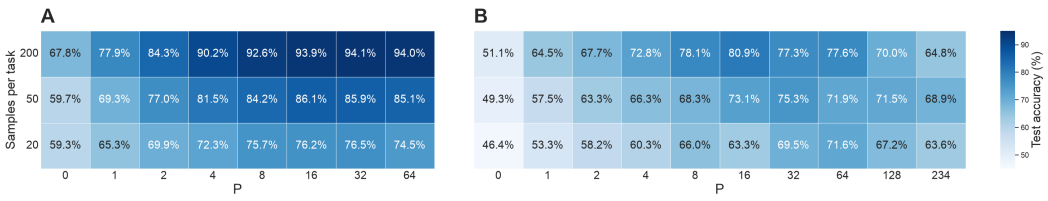


Figure 6: Test accuracy (%) as a function of nonlinearity P and training sample size per task, for **A** individual models (hidden size $M = 64$) trained separately on each of 100 tasks and **B** a single flat model (hidden size $M = 234$) jointly trained on all 100 tasks. Values are means over 5 seeds.

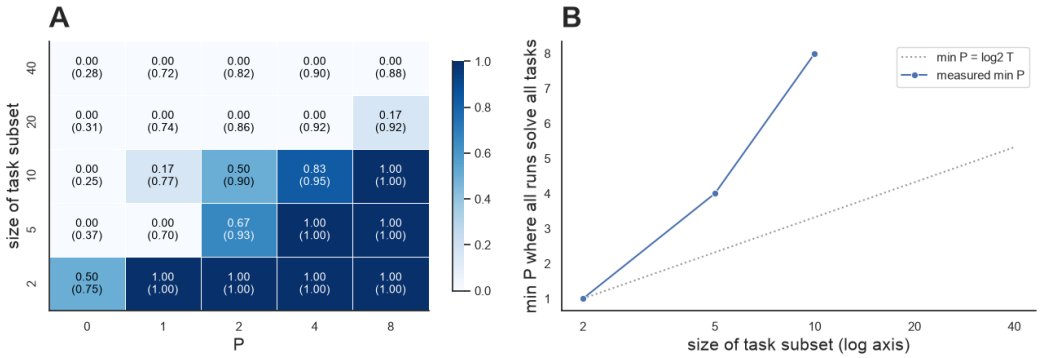


Figure 7: Individually linear tasks require nonlinear capacity when learned jointly. We constructed subsets of T tasks for which every task was solved by an individually trained linear AL-RNN, and trained joint models with varying numbers of nonlinear units. **A:** Each cell reports the fraction of six runs that solved all T tasks, values in parentheses show the mean fraction of tasks solved. **B**: Minimum number of nonlinear units required to solve every task in a subset reliably. The dotted line shows the idealized lower bound $\log_2 T$ obtained by assigning each task a distinct nonlinear region. The larger number required in practice likely reflects the additional constraint of shared recurrent parameterization in the AL-RNN.

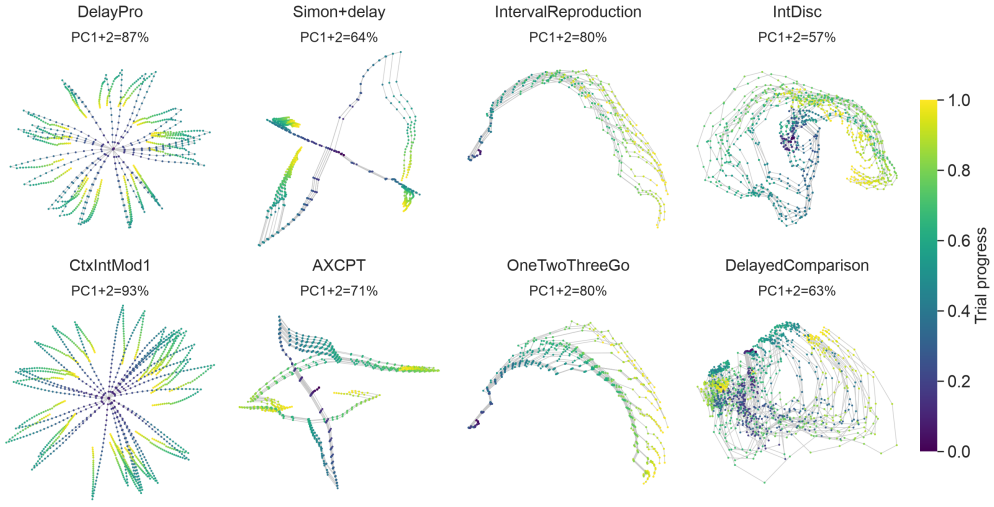


Figure 8: Four example tasks, each shown with one of its closest neighbours by geometric overlap. Tasks placed close together also tend to produce trajectories of similar shape.

Table 4: Robustness and overlap of task-similarity spaces across random seeds. Entries report mean seed-pair Pearson correlations, with $\Delta r = (r_{\max} - r_{\min})/2$ denoting half the observed range. The upper panel measures reproducibility within each similarity space, while the lower panel measures correlations between spaces. The robustness of a task representation is partly coupled to task performance, since poor performance constrains the feature vectors and dynamical activity less. On the other hand, particularly for the nonlinearity usage, we found that increasing capacity enforces less sharing between model components, and tasks are separated into increasingly distinct regions, reducing off-diagonal similarity and making the nonlinear usage matrix approach an identity-like structure. Accordingly, the cross-space agreement was generally highest at intermediate nonlinear capacity, consistent with sparse nonlinearity providing task differentiation without over-separating tasks into distinct linear regions.

	$N_{\text{feat}} = 8$			$N_{\text{feat}} = 16$		
Similarity space / pair	$P = 8$	$P = 16$	$P = 32$	$P = 8$	$P = 16$	$P = 32$
Within-space reproducibility						
Hierarchical features	0.56 ± 0.06	0.63 ± 0.06	0.64 ± 0.06	0.76 ± 0.02	0.78 ± 0.04	0.82 ± 0.01
Task geometry	0.67 ± 0.04	0.74 ± 0.04	0.72 ± 0.04	0.77 ± 0.03	0.78 ± 0.02	0.82 ± 0.02
Nonlinear usage	0.66 ± 0.06	0.78 ± 0.05	0.85 ± 0.05	0.71 ± 0.07	0.76 ± 0.03	0.81 ± 0.05
Between-space similarity						
Geometry–features	0.69 ± 0.04	0.69 ± 0.02	0.65 ± 0.05	0.65 ± 0.04	0.65 ± 0.04	0.63 ± 0.03
Geometry–nonlinearity	0.54 ± 0.04	0.70 ± 0.04	0.53 ± 0.03	0.56 ± 0.07	0.68 ± 0.05	0.52 ± 0.03
Features–nonlinearity	0.37 ± 0.08	0.47 ± 0.04	0.32 ± 0.04	0.37 ± 0.11	0.47 ± 0.07	0.34 ± 0.05

Table 5: Seed reproducibility and cross-architecture agreement for task representations across 5 seeds. The nonlinear feature space is only feasible for the AL-RNN, and hence omitted here. Models with settings as in Table 1 ($M = 64$, $N_{feat} = 16$).

Model	Geometry	Task features
LSTM seed consistency	0.81	0.76
GRU seed consistency	0.78	0.74
AL-RNN–LSTM feature similarity	0.53	
AL-RNN–GRU feature similarity	0.57	

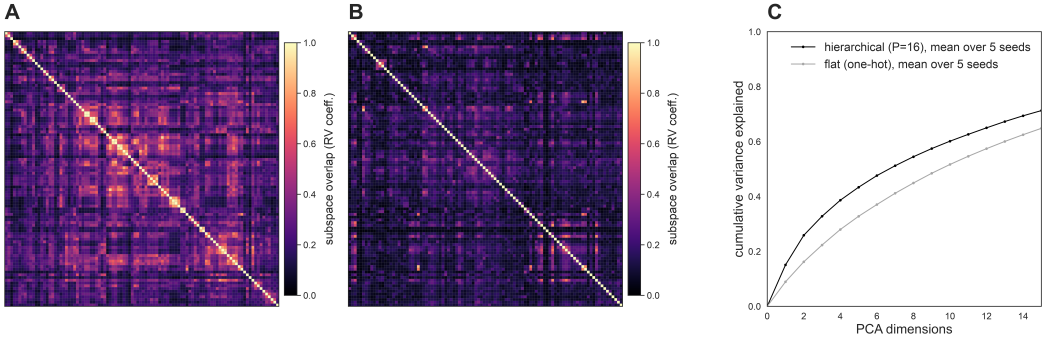


Figure 9: Geometric task similarity matrices for the hierarchical (**A**) and flat models (**B**) with $M = 64$ latent dimensions and the same task order in both. **C**: Cumulative variance explained via the first 15 principal components. The hierarchical model compresses activity into fewer dimensions, retaining substantially more overlap among tasks.

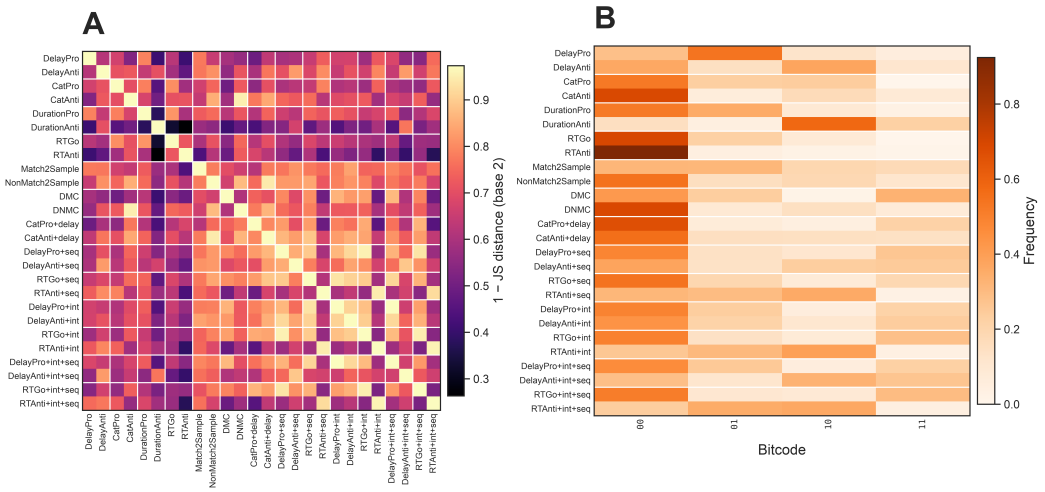


Figure 10: **A**: Linear region usage for a hierarchical AL-RNN with $P = 2$, leading to $2^2 = 4$ linear subregions. **B**: Pro-Anti task pairs are separated in nonlinear space. Each row shows the frequency with which a task visits each linear region of the AL-RNN. Pro and anti variants share some regions but also exhibit distinct usage patterns, indicating that the model reuses shared nonlinear structure while separating computations with opposing response rules.

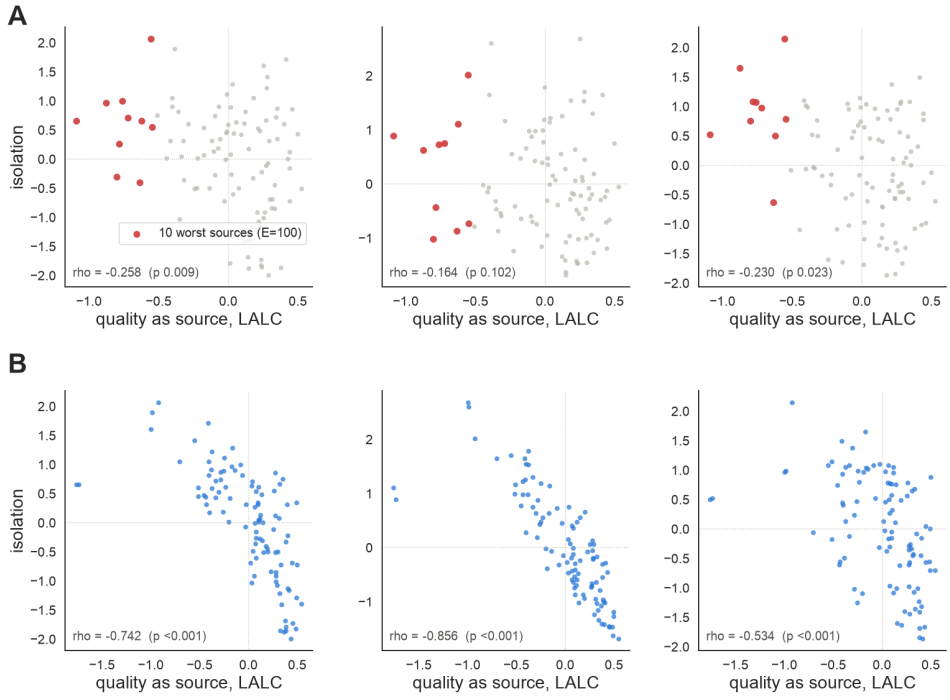


Figure 11: Isolation in the hierarchical AL-RNN ($N_{\text{feat}} = 16, P = 16$) feature spaces (geometry, task features and linear region usage from left to right), plotted against two different measures of source quality. **A**: Source quality is computed from the individual-model transfer matrix: one network per task ($P = 16$), fine-tuned onto every other task. Red points are the 10 worst sources when considering the first 100 training epochs (E100). **B**: Source quality computed from the leave-one-out experiments, where the task feature for the target task was fine-tuned initialized from the source task's task feature.

C TASK BATTERY

The task descriptions and names were adapted from established paradigms in cognitive psychology and systems neuroscience where available. Citations indicate the source paradigm and conceptual inspiration. Our implementations are standardized for the shared low-dimensional interface and therefore do not always reproduce the original experimental stimuli or modalities exactly. For example, auditory, visual, and motor variables are represented by abstract low-dimensional input and output channels rather than their natural sensory or behavioural formats.

Observations are zero-padded to a common input dimension, targets live in a fixed 5-dimensional readout, and a masked mean-squared error restricts the loss to the behaviourally relevant timesteps of each trial. Trials are organised into phases (*context*, *stimulus*, *delay*, *response*) whose durations are resampled per trial, and a fixation signal is maintained until the response window.

Table 6 lists the 66 distinct paradigms and Sec. C.1 the four modifiers that expand them to the 100 task identities used in the experiments, based on the Mod-Cog suite (Khona et al., 2023). Throughout, *angular* denotes a unit vector $(\cos\theta, \sin\theta)$ scored by angular error, *binary* a thresholded decision, *scalar* a continuous magnitude, *vector* two continuous coordinates, *categorical* an $\arg\max$ over K alternatives, and *bits* the signs of n outputs. A star (*) marks readouts scored at every response timestep rather than only at the report step.

C.1 MODIFIERS

The paradigms of Table 6 are expanded by four modifiers, each of which wraps a base task into a separate task identity with its own embedding $\boldsymbol{p}^{(k)}$, giving $66+34 = 100$ tasks in total. The bases they apply to are {GoNogo, PerceptualDM, CatPro, CatAnti, NoiseCleaner, MultiSens, Stroop, Flanker, Simon, TaskSwitch} for +delay, and the eight single-target angular tasks {DelayPro, DelayAnti, NoiseCleaner, MultiSens, PerceptualDM, RTGo, RTAnti, SpatialWMRetro} for +seq, +int and their composition.

Modifier	**Description**	**Readout**
+delay	A blank retention gap (fixation on, stimulus off, unscored) is spliced in immediately before the response window, forcing the pre-response state to be held across the delay.	as base
+seq	The reported direction drifts at a cued angular speed ω over the response window, so a static angular response becomes a rotating output trajectory.	angular*
+int	A delay of variable length is inserted before the response during which the remembered direction rotates at ω, so the reported angle is $\theta + \omega T_{\text{delay}}$ and angular velocity must genuinely be integrated.	angular
+int+seq	Composition of the two: the memory rotates across the inserted delay *and* the output moves during the response window.	angular*

Table 6: The 66 paradigms of the task battery

Task	**Description**	**Readout**
DelayPro	A direction cue is presented, followed by a delay phase. After delay, the network reports it once the fixation cue is removed (Yang et al., 2019).	angular
DelayAnti	As DelayPro, but the reported direction is opposite to the cued one (Yang et al., 2019; Munoz and Everling, 2004).	angular
RTGo	The stimulus direction is reported as soon as it appears, with no intervening delay (Yang et al., 2019).	angular
RTAnti	As RTGo, but reporting the direction opposite the stimulus (Yang et al., 2019; Munoz and Everling, 2004).	angular

continued on next page

Table 6, continued

Task	Description	Readout
PerceptualDM	Noisy directional evidence is presented at varying coherence and the net direction must be integrated out of the noise within a given tolerance level (Britten et al., 1992).	angular
NoiseCleaner	A true direction must be estimated from a stream of noisy angular samples within a given tolerance.	angular
MultiSens	Two modalities are shown simultaneously with independent strengths. After a delay period, the response is the direction of their vector sum (Yang et al., 2019).	angular
CtxIntMod1	Integrate modality 1 and ignore modality 2, as instructed by a context cue shown at the start (Mante et al., 2013; Yang et al., 2019).	angular
CtxIntMod2	Integrate modality 2 and ignore modality 1 (Mante et al., 2013; Yang et al., 2019).	angular
PriorIntegration	A prior cue indicating the likely direction is followed by noisy evidence of varying strength. The network reports the true direction.	angular
SpatialWMRetro	Two locations are presented in a sequence, followed by a delay and retro-cue. The network reports the cued location (Funahashi et al., 1989; Griffin and Nobre, 2003).	angular
WMUpdating	A stream of six items, each tagged with one of two categories, is followed by a query naming a category. The network reports the most recent item of that category (Yntema, 1963; Miyake et al., 2000).	angular
PosnerCueing	A spatial cue indicates the likely target direction. After a delay, a noisy target appears and the network reports its true direction (Posner, 1980).	angular
CopyTask	A sequence of symbols is presented one per step and must be immediately reproduced in order after input ends.	angular*
HeadDirection	An initial heading is cued and a stream of angular velocities must be integrated into a running heading reported at every step (Kim et al., 2017; Cueva and Wei, 2018).	angular*
SineGen	A cued frequency must be turned into a sustained, amplitude-stable oscillation, i.e. a tunable limit cycle (Sussillo and Barak, 2013).	angular*
AntiReach1D	A target appears at scalar position x on a line and the network must reach to $-x$ (Munoz and Everling, 2004).	scalar
SoundLocalization	Source azimuth must be recovered from a noisy level difference across two "ear" inputs (Knudsen and Konishi, 1978).	scalar
IntervalReproduction	Two flashes mark an interval; the network reports its length as a scalar once the second flash ends (Jazayeri and Shadlen, 2010).	scalar
OneTwoThreeGo	Three equally spaced flashes mark a repeated interval. The network reports its length as a scalar after the third flash (Egger et al., 2019).	scalar
PathIntegration1D	A stream of 1D velocities from a fixed origin; report the final position without further input (Cueva and Wei, 2018; Seung, 1996).	scalar
Numerosity	A number of discrete pulses is presented and, after a go cue, the count is reported as a continuous quantity (Nieder et al., 2006).	scalar
PathIntegration2D	As PathIntegration1D, with 2D velocities (vx, vy). Report the final (x, y) position (Cueva and Wei, 2018; Banino et al., 2018).	vector
GoNogo	Respond only if the stimulus amplitude exceeds a threshold, otherwise maintain fixation and withhold the response.	binary
StopSignal	A go stimulus appears, followed on a fraction of trials by a stop signal. Respond unless a stop signal occurred (Logan and Cowan, 1984; Hanes et al., 1998).	binary

continued on next page

Table 6, continued

Task	Description	Readout
CatPro	Classify the stimulus direction according to which half of the ring it falls in (Freedman and Assad, 2006).	binary
CatAnti	As CatPro with the category-to-response mapping reversed (Freedman and Assad, 2006).	binary
PoissonClicks	Two Poisson click streams arrive on separate input lines. The network has to report which side delivered more clicks (Brunton et al., 2013; Hanks et al., 2015).	binary
SignalDetection	A weak constant signal is present in a noisy stream on half of the trials. The network reports whether the signal was present or not (Green and Swets, 1966).	binary
BackwardMasking	A brief category signal is immediately followed by a high-noise mask on the same input line. The network reports the category (Breitmeyer and Ganz, 1976).	binary
Match2Sample	The network reports whether a test direction matches the sample held across a delay (Yang et al., 2019).	binary
NonMatch2Sample	As Match2Sample, responding positively to non-match instead (Yang et al., 2019).	binary
DMC	A sample direction is followed, after a delay, by a second direction. The network reports whether both fall in the same half of the ring (Freedman and Assad, 2006; Yang et al., 2019).	binary
DNMC	As DMC, but responding positively for different halves (Freedman and Assad, 2006; Yang et al., 2019).	binary
DMSDistractor	Delayed match-to-sample in which an irrelevant distractor direction appears during the delay and must be ignored (Miller et al., 1996).	binary
Sternberg	A set of items is presented, followed by a delay and single probe. The network reports whether the probe was in the set (Sternberg, 1966).	binary
PairedAssoc	A cue direction is followed, after a delay, by a second cue. The network reports whether the second is the designated associate of the first (Sakai and Miyashita, 1991).	binary
ChangeDetect	A sample array is shown and, after a delay, followed by a test array. The network reports whether any item has changed (Luck and Vogel, 1997).	binary
SerialOrderMemory	Report whether one probe item occurred before another in the previously presented list (Hacker, 1980).	binary
DelayedComparison	Two noisy scalar magnitudes separated by a delay. Report whether the first is stronger (Romo et al., 1999).	binary
DurationPro	Judge whether the interval between two flashes is short or long relative to a learned midpoint (Church and Deluty, 1977).	binary
DurationAnti	As DurationPro with the response mapping reversed (Church and Deluty, 1977).	binary
IntDisc	Two stimuli are presented, separated by a delay. The network reports whether the first lasted longer than the second.	binary
TemporalBisection	A single stimulus duration is classified as short or long about a fixed, learned boundary (Church and Deluty, 1977; Leon and Shadlen, 2003).	binary
TemporalOrderJudgment	Two pulses separated by a variable onset asynchrony arrive on different input lines. The network reports which came first (Hirsh and Sherrick, 1961).	binary

continued on next page

Table 6, continued

Task	Description	Readout
SimultaneityJudgment	Report whether two pulses were simultaneous or separated by a small onset asynchrony (Hirsh and Sherrick, 1961).	binary
GapDetection	An ongoing stimulus is presented and the network reports whether a brief gap interrupted it (Plomp, 1964).	binary
Stroop	Two features are shown at once and a rule cue selects which to report. On incongruent trials the features disagree (Stroop, 1935).	binary
Flanker	A target value and an irrelevant flanker value are shown simultaneously on separate input lines. The network reports the target (Eriksen and Eriksen, 1974).	binary
Simon	Respond according to stimulus identity while ignoring an irrelevant location that may map to the opposite response (Simon, 1969).	binary
TaskSwitch	A rule cue selects whether to judge the sign or the magnitude of the same scalar stimulus (Sohn et al., 2000; Sakai, 2008).	binary
AXCPT	A cue (A or B) is followed, after a delay, by a probe (X or Y). The network responds only to the A-then-X sequence (Servan-Schreiber et al., 1996; Cohen et al., 1999).	binary
DelayedCueReversal	A rule cue may appear early and is withdrawn before the stimulus. The network reports the stimulus sign, inverted if the cue appeared.	binary*
VisualSearch	A target feature is cued and the network reports whether it is present in a sequentially presented array (Treisman and Gelade, 1980).	binary
AttentionalBlink	A rapid stream contains two flagged targets at variable lag. The network reports the category of the second (Raymond et al., 1992).	binary
OddballDetection	A stream of repeated cues is presented and the network reports whether a single cue broke the regularity (Näätänen et al., 1978; Ulanovsky et al., 2003).	binary
HierReason	A scalar stimulus is presented with two cues: one selects whether to judge its sign or its magnitude, the other whether to invert that judgment. The network reports the result (Badre and D'Esposito, 2007; 2009).	binary
ConditionalReasoning	An antecedent and a consequent are presented and the network reports whether the case is consistent with the material conditional, i.e. detects the falsifying combination (Wason, 1968; Johnson-Laird and Tagart, 1969).	binary
RelationalMatch	Two pairs of items are presented in a sequence, each either same or different, and the network reports whether both pairs share the same relation (Fagot et al., 2001).	binary
MentalRotation	A sample direction is followed by a delay and a probe, with the rotation amount cued from trial onset. The network reports whether the probe equals the sample rotated by the cued amount (Shepard and Metzler, 1971; Georgopoulos et al., 1989).	binary
SymbolCategory	A short string of symbols is presented one symbol at a time and must be classified as a member of a fixed lexicon or not (Rubenstein et al., 1970; Meyer and Schvaneveldt, 1971).	binary
DiscriminationOptOut	Noisy scalar evidence of variable strength is presented. The network chooses left or right when the underlying signal is strong and a third "opt-out" option when it is weak.	categorical
PluralityVote	A stream of six symbols, each one of three categories, is presented and followed by a delay. The network reports the most frequent category.	categorical
OddOneOut	K scalar items are shown at once and exactly one deviates from the common level (up or down). The network reports which one.	categorical

continued on next page

Table 6, continued

Task	Description	Readout
FlipFlop3	Three bits, each set by a brief ± 1 pulse, must be held and reported at every timestep (Sussillo and Barak, 2013).	3 bits*
FlipFlop2	As FlipFlop3 with two bits, giving four stable configurations.	2 bits*