

Task-structured modularity emerges in artificial networks and aligns with brain architecture

1 Contents

2	1	Details of Involved Tasks	2
3	2	Tasks-elicited Modularity	3
4	2.1	Task-specific modularity	3
5	2.2	Multi-task v.s. single task modularity	5
6	3	Results of Other Types of Networks	7
7	4	Mask Generation Protocol and the Control Experiments	12
8	5	Perturbation Analysis of models in Fig2. b	13
9	6	Statistical Model Comparison via Information Criteria	13
10	7	Mechanistic Account of Task-Driven Modularity and Learning Efficiency	17

11 Supplementary Note 1: Details of Involved Tasks

12 To investigate the differences in emerged structures elicited by various cognitive demands, we implemented a
 13 functional classification of the 20 tasks used in our study. This classification is not novel but strictly adheres
 14 to the original design framework established for this task set ([1]). Based on the similarities and differences in
 15 their core computational requirements, the tasks were organized into five "families": the Go, Anti, Decision
 16 Making (DM), Delayed Decision Making (Dly DM), and Matching families:

Supplementary Table. 1. Cognitive Task Requirements and Elicited Representations

Task Family	Task Member	Core Computational Primitive	Required Representation Type
Go Family	fdgo, reactgo, delaygo	Sensorimotor Mapping, Inhibitory Control (in 'fdgo'), Working Memory (in 'delaygo')	Sensorimotor & Memory: Representations must map a sensory location to a motor plan. 'delaygo' specifically requires maintaining this location trace over a delay (fading memory). 'fdgo' requires inhibiting a prepotent response.
Anti Family	fdanti, reactanti, delayanti	Flexible Sensorimotor Re-mapping, Inhibitory Control, Working Memory (in 'delayanti')	Rule-based Transformation: Representations must apply a contextual "Anti" rule to transform the sensory location into a new, opposite motor plan, requiring high separation of sensory input from motor output.
Decision Making (DM) Family	dm1, dm2, contextdm1, contextdm2, multidm	Spatial Evidence Integration, Selective Attention/Gating (context tasks)	Integrator/Accumulator: Representations must simultaneously compare and accumulate analog evidence (stimulus strength) from two competing spatial locations. Context tasks require this integration to be gated by a rule.
Delayed DM Family	delaydm1, delaydm2, contextdelaydm1, contextdelaydm2, multidelaydm	Temporal Evidence Integration, Analog Working Memory, Selective Attention/-Gating	Analog Working Memory: Representations must store an *analog* value (strength of stim 1) over a delay and then compare it to a second sequential input. This demands robust memory of a continuous variable.
Matching Family	dmsgo, dmsnogo, dmcgo, dmcnogo	Feature-based Working Memory, Categorization (dmc tasks), Comparison	Abstract/Categorical: Representations must store discrete features (location/modality) in 'dms'. For 'dmc', they must further abstract a continuous location into a binary category, maintain this abstract label, and compare it to a subsequent stimulus category.

17 The specifications of the five task sets used in Fig. 2 are as follows:

18 3 tasks: {Go Family}

19 6 tasks: {Go Family, Anti Family}

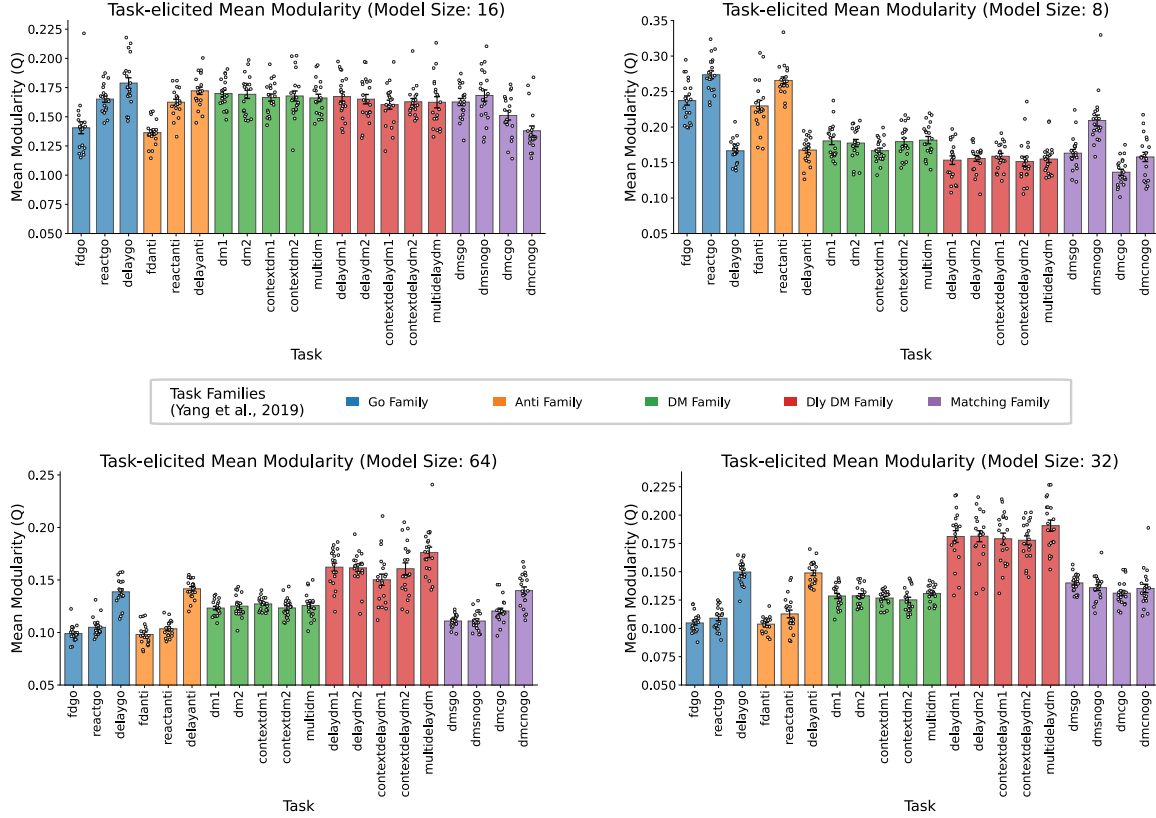
20 11 tasks: {Go Family, Anti Family, Decision Making Family}

21 16 tasks: {Go Family, Anti Family, Decision Making Family Delayed Decision Making Family }

22 20 tasks: {Go Family, Anti Family, Decision Making Family, Delayed Decision Making Family, Matching Family }

23 Supplementary Note 2: Tasks-elicited Modularity

24 2.1 Task-specific modularity



Supplementary Fig. 1. Task-specific modularity reveals an interaction between computational demand and model capacity. Mean modularity (Q) of networks trained on each of the 20 single tasks, grouped by model size (# Hidden Neurons: 8, 16, 32, 64). Tasks are color-coded by their functional family. Bars show mean values and error bars show SEM; overlaid dots indicate individual independently trained networks. For each task and model size, n=20 independent experiments.

Supplementary Fig.1 provides a detailed breakdown of the modularity elicited by each of the 20 tasks when trained individually. To further avoid variability associated with random initialization, these experiments adopted the same initialization protocol as that used in Fig. 4. This analysis reveals broad, consistent patterns that directly link a task’s computational requirements to the network’s emergent structure, and it highlights a crucial interaction with model capacity.

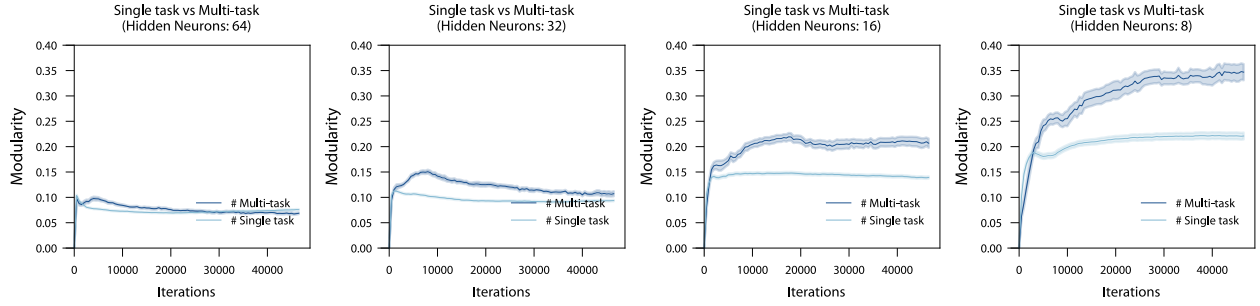
In larger models (# Hidden Neurons: 64 and 32), a clear hierarchy emerges. The Delayed Decision Making (Dly DM) family (red) consistently yields the highest median modularity. These tasks are arguably the most computationally complex, requiring the integration of context, evidence accumulation, and working memory over a delay. Conversely, tasks requiring simpler, immediate sensorimotor mapping (the fdgo, reactgo,

34 fdanti, and reactanti tasks) produce the lowest median modularity. The Decision Making (DM) and Match-
35 ing families fall between these two extremes. Notably, a "delay" component consistently induces greater
36 modularization than its non-delay counterpart (e.g., delaygo vs. fdgo, delayanti vs. fdanti, and the Dly DM
37 family vs. the DM family), indicating that the demand for working memory introduces significant compu-
38 tational pressure. Furthermore, within the Go and Anti families, tasks requiring reactive control (reactgo,
39 reactanti) tend to elicit slightly higher modularity than their fixed-duration counterparts (fdgo, fdanti). This
40 subtle increase suggests that even the necessity for precise, reaction-time-based control over a fixed, internal
41 temporal mapping contributes marginally to the pressure for structural segregation.

42 This pattern is strongly modulated by model capacity. In the 16-neuron models, the trend largely holds:
43 delay tasks still induce higher modularity than their non-delay counterparts. However, this phenomenon
44 dramatically inverts in the most severely constrained models (# Hidden Neurons: 8). Here, the simpler fdgo
45 and fdanti tasks produce higher modularity than their more complex delaygo and delayanti counterparts.
46 This inversion suggests a threshold effect: when network capacity is too limited, it cannot form the necessary
47 complex, modular solution for the delay tasks. The network can, however, still adequately solve the simpler
48 tasks, for which a moderately modular structure is sufficient and achievable.

49 Taken together, these results support our central hypothesis. Modularity is not a default property but
50 an emergent structural adaptation to specific computational demands. The strongest demand appears to be
51 the need to maintain information and segregate computational processes over time—directly engaging the
52 "fading memory" and "separation" properties—as seen in all delay-based tasks. These complex requirements
53 drive the network to find more modular solutions, but this adaptation is contingent on the network having
54 sufficient representational capacity to find and implement them.

55 2.2 Multi-task v.s. single task modularity

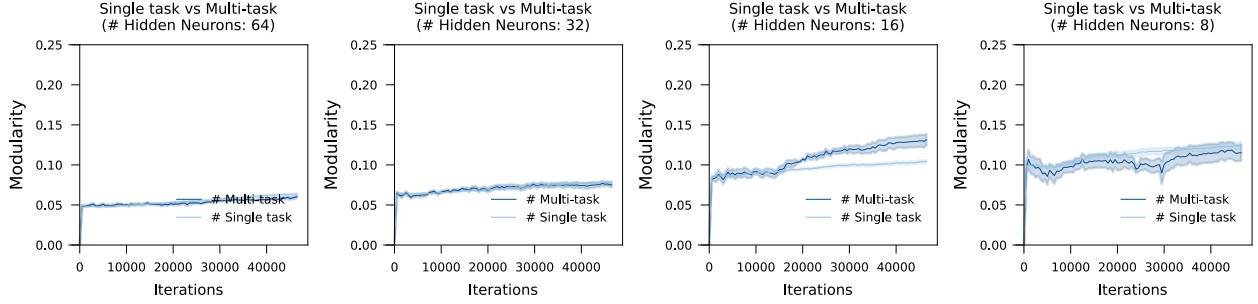


Supplementary Fig. 2. The complete results across models of all sizes. Due to space limitations, Fig. 2d in the main text only shows the results for $N=16$. This figures presents a comparison of models of all sizes under training mode I and training mode II. Solid lines indicate the mean and shaded areas indicate the s.e.m. For the multi-task condition, $n = 20$ independently trained model instances were used. For the single-task condition, $n = 20$ replicate-level means were used, each calculated by averaging modularity across the 20 independently trained task-specific models within that replicate.

Supplementary Table. 2. Statistical comparisons corresponding to Supplementary Fig. 2. Exact P values from two-sided Welch's t -tests comparing modularity between the single-task and multi-task training conditions at the indicated training checkpoints. For each comparison, $n = 20$ multi-task model instances and $n = 20$ single-task replicate-level means were used; each single-task replicate-level mean was calculated across 20 independently trained task-specific models.

Iterations	15000	20000	25000	30000	35000	40000
# 64	1.80×10^{-8}	1.12×10^{-5}	0.01	0.41	0.50	0.20
# 32	1.30×10^{-9}	1.97×10^{-10}	4.69×10^{-9}	2.18×10^{-9}	1.70×10^{-7}	4.77×10^{-6}
# 16	4.43×10^{-10}	3.58×10^{-10}	2.98×10^{-12}	4.07×10^{-11}	2.60×10^{-10}	7.51×10^{-11}
# 8	1.69×10^{-9}	6.52×10^{-9}	2.27×10^{-9}	2.06×10^{-9}	9.03×10^{-10}	1.85×10^{-10}

56



Supplementary Fig. 3. Control experiment examining the effect of recurrent-weight scaling at initialization. In the experimental condition shown in Supplementary Fig. 2, orthogonally initialized recurrent weights were scaled to reduce the influence of modularity arising from random initial connectivity. Here, this scaling was omitted, resulting in a less pronounced separation in modularity between the multi-task and single-task conditions and indicating that initial recurrent connectivity can obscure task-driven modular organization. Solid lines indicate the mean and shaded areas indicate the s.e.m. For the multi-task condition, $n = 10$ independently trained model instances were used. For the single-task condition, $n = 20$ replicate-level means were used, each averaged across 10 independently trained task-specific models. Exact P values are reported in Supplementary Table. 3.

Supplementary Table. 3. Statistical comparisons corresponding to Supplementary Fig. 3. Exact P values from two-sided Welch’s t -tests comparing modularity between the single-task and multi-task training conditions at the indicated training checkpoints. For each comparison, $n = 10$ multi-task model instances and $n = 20$ single-task replicate-level means were used; each single-task replicate-level mean was calculated across 10 independently trained task-specific models.

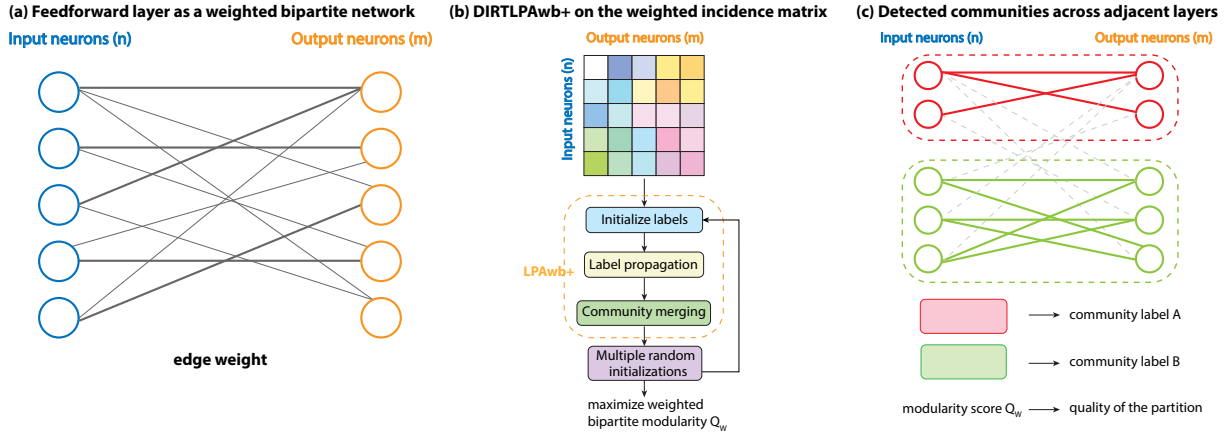
Iterations	15000	20000	25000	30000	35000	40000
# 64	0.96	0.09	0.12	0.09	0.01	0.00
# 32	0.35	0.81	0.80	0.42	1.00	0.37
# 16	0.65	1.88×10^{-5}	5.76×10^{-4}	0.00	0.01	0.01
# 8	0.63	0.37	0.18	0.24	0.56	0.62

Supplementary Table. 4. Statistical comparisons corresponding to Fig. 5b. Exact unadjusted P values from two-sided Welch’s two-sample t -tests comparing modularity between the incremental and all-at-once training conditions at the indicated training checkpoints. For each condition, $n=20$ independently trained model instances.

Iterations	15000	20000	25000	30000	35000	40000
# 84	1.49×10^{-24}	1.13×10^{-26}	4.01×10^{-25}	2.17×10^{-24}	1.60×10^{-21}	2.66×10^{-16}

Supplementary Note 3: Results of Other Types of Networks

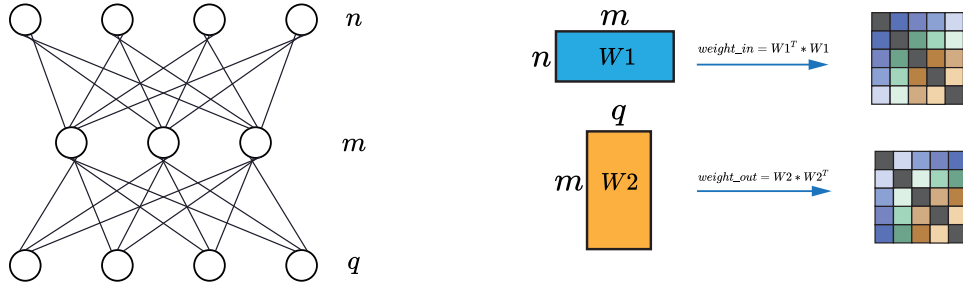
In addition to the recurrent architectures, we also conducted experiments using feed-forward neural models that lack recurrent connections, such as convolutional neural networks (CNNs) and graph neural networks (GNNs). For recurrent layers, the recurrent weight matrix is a square matrix defined over the same set of recurrent units, with each matrix entry directly describing the connection between a pair of nodes within that layer. Such a representation can therefore be naturally interpreted as a weighted network for community detection. In contrast, the weight matrix of a feed-forward layer describes connections between two distinct sets of units in adjacent layers and is generally rectangular, rather than a square connectivity matrix defined over a single set of nodes. Consequently, how modular structure should be defined and detected for feed-forward neural networks is less straightforward. This remains an open methodological question, and a comprehensive theoretical treatment or a new formal definition is beyond the scope of our study.



Supplementary Fig. 4. Community detection of feed-forward layers using DIRTLPawb+ algorithm. Feed-forward connections between two adjacent layers can be represented as a weighted bipartite network. The corresponding feed-forward weight matrix is treated as the weighted incidence matrix for DIRTLPawb+. Within each execution of LPAwb+, node labels are iteratively updated through label propagation to increase weighted bipartite modularity, followed by community merging when merging produces a further increase in modularity. DIRTLPawb+ repeats this optimization using multiple initializations and retains the partition with the greatest weighted bipartite modularity, Q_w . **c**, The resulting partition assigns community labels jointly to neurons from both adjacent layers. Neurons sharing the same community label constitute a common bipartite module across the two node sets, and Q_w quantifies the modularity of the resulting partition.

Nevertheless, for exploratory purposes, we investigated two possible approaches. First, since feed-forward connections naturally form a weighted bipartite graph between adjacent layers, we applied the DIRTLPawb+ algorithm proposed by [2] for community detection in weighted bipartite networks (Supplementary Fig. 4). Specifically, neurons from two adjacent layers were treated as two disjoint node sets, where the feed-forward

weights between them define weighted edges of the bipartite graph. The corresponding weight matrix was used as the weighted incidence matrix provided to the algorithm. DIRTLPawb+ optimizes weighted bipartite modularity by iteratively updating node labels to maximize modularity, merging communities when this increases modularity, and repeating the optimization from multiple random initializations to improve robustness against local optima. The approach directly operates on asymmetric feed-forward connectivity and simultaneously assigns community labels to both node sets. The resulting partitions therefore characterize modular organization across pairs of adjacent feed-forward layers without requiring the original weight matrix to be symmetrized.



Supplementary Fig. 5. Construction of symmetric matrices for community detection in feed-forward networks. For three consecutive feed-forward layers containing n , m , and q units, respectively, the adjacent asymmetric weight matrices are denoted by $W1 \in \mathbb{R}^{n \times m}$ and $W2 \in \mathbb{R}^{m \times q}$. For the m neurons in the intermediate layer, symmetric matrices describing their incoming and outgoing connection patterns were constructed as $\text{weight_in} = W1^T W1$ and $\text{weight_out} = W2 W2^T$, respectively. Both resulting matrices are of size $m \times m$ and were used for subsequent community detection.

Second, we considered an alternative approach in which the asymmetric weight matrices of adjacent feed-forward layers were transformed into symmetric matrices defined over neurons within the same intermediate layer. Consider three consecutive layers containing n , m , and q units, respectively, with adjacent feed-forward weight matrices $W1 \in \mathbb{R}^{n \times m}$ and $W2 \in \mathbb{R}^{m \times q}$ (Supplementary Fig. 5). For the m neurons in the intermediate layer, we characterized their pairwise relationships from both their incoming and outgoing connection patterns. Specifically, we constructed the symmetric incoming-connectivity matrix as $\text{weight_in} = W1^T W1$ and the symmetric outgoing-connectivity matrix as $\text{weight_out} = W2 W2^T$. Both weight_in and weight_out are therefore $m \times m$ symmetric matrices defined over the same set of intermediate-layer neurons. More specifically, each element (i, j) of weight_in is the inner product between the incoming weight vectors of intermediate-layer neurons i and j , such that $(\text{weight_in})_{ij} = \sum_{k=1}^n W1_{ki} W1_{kj}$. Thus, two intermediate-layer neurons have a larger value in weight_in when their weighted connection profiles from the preceding layer are more strongly aligned. Similarly, $(\text{weight_out})_{ij} = \sum_{k=1}^q W2_{ik} W2_{jk}$ is the inner product between the outgoing weight vectors of neurons i and j , characterizing the extent to which they project to the subsequent

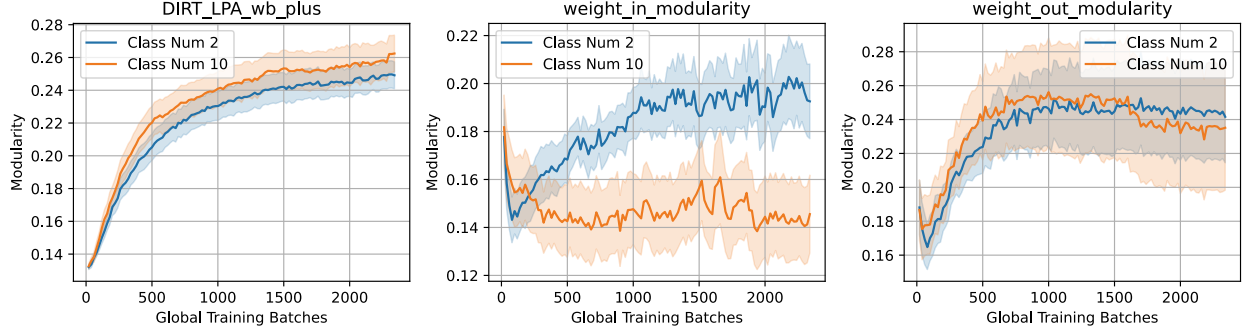
layer with aligned weight patterns. In this way, the original rectangular and asymmetric feed-forward weight matrices are projected onto symmetric representations of pairwise relationships among neurons within the intermediate layer. Standard community-detection methods can then be applied to these derived symmetric matrices to identify potential modular organization among the intermediate-layer neurons. The corresponding results are shown in Supplementary Fig. 6 and Supplementary Fig. 7.

For the CNN experiments, we used the CIFAR-10 dataset [3] to build a contrastive learning task. Specifically, we selected the first two (Clas Num 2 in Supplementary Fig. 6) and all ten categories (Clas Num 10 in Supplementary Fig. 6) from CIFAR-10, respectively, and trained a Siamese CNN to determine whether two input images belong to the same class.

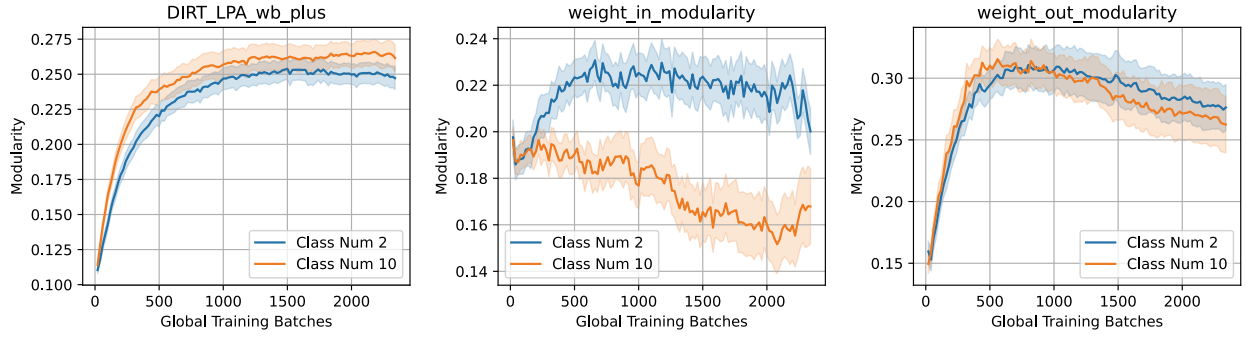
For the GNN experiments, we used the Tox21 dataset, which contains 12,060 training and 647 test samples of chemical compounds, each annotated with 12 binary labels indicating activity or inactivity across 12 toxicological assays. We trained both single-task and multi-task GNN models: the former predicts one toxicity label at a time (Single Task Learning in Supplementary Fig. 7), while the latter simultaneously predicts all 12 labels (Multiple Tasks Learning in Supplementary Fig. 7).

Convolutional Neural Networks During Contrastive Learning

(# channel_in: 8, channel_out: 8)



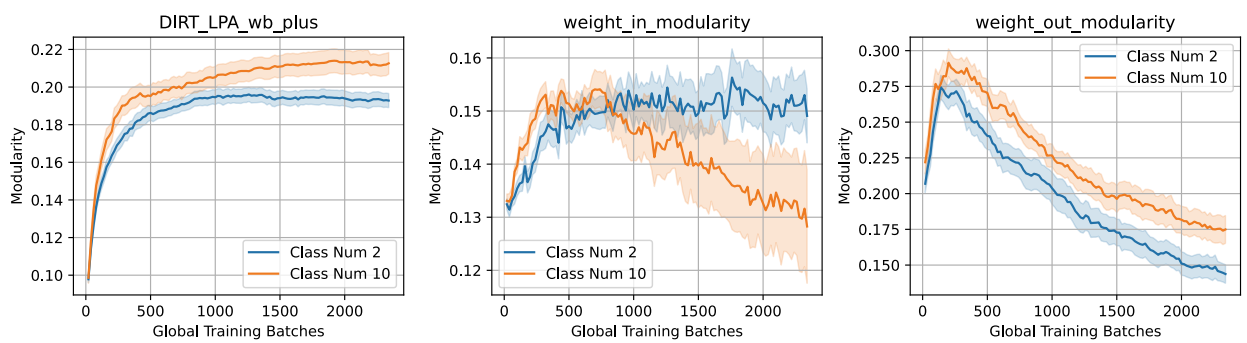
(# channel_in: 16, channel_out: 16)



(# channel_in: 32, channel_out: 32)

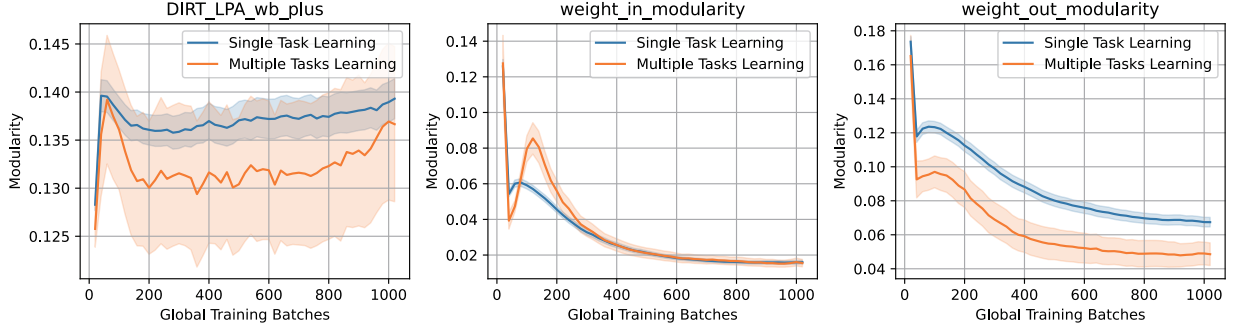


(# channel_in: 64, channel_out: 64)

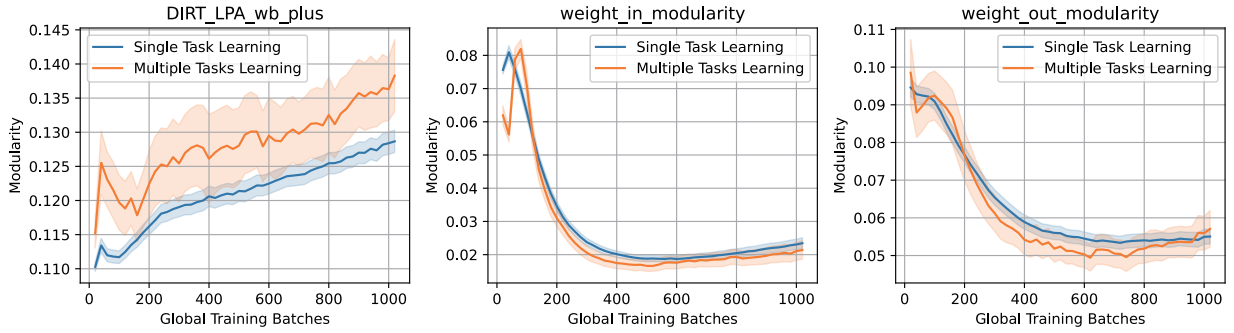


Supplementary Fig. 6. Modular structure for CNNs. We applied bipartite network community detection (DIRT_LPA_wb_plus [2]) or community detection on CNN models. Solid lines are the mean over 10 independent runs. Shaded areas indicate the standard error.

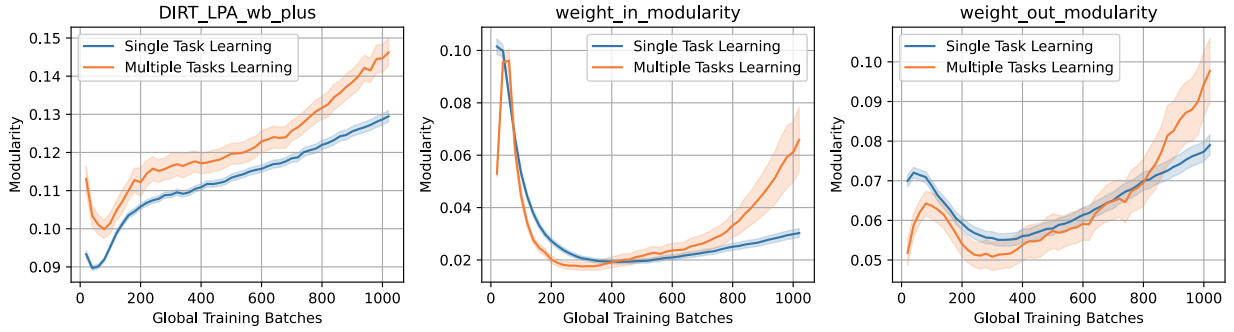
Graph Neural Networks During Training on Tox21 Dataset
(# hidden_channels: 16)



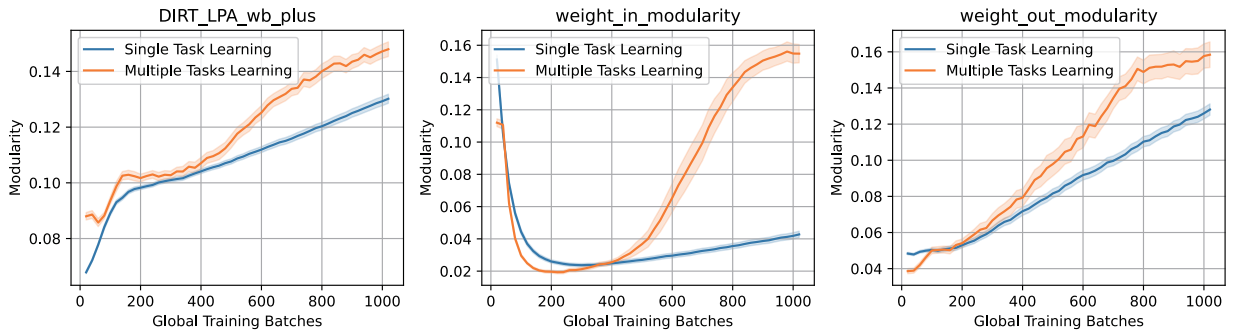
(# hidden_channels: 32)



(# hidden_channels: 64)



(# hidden_channels: 128)



Supplementary Fig. 7. Modular Structure for GNNs. We applied bipartite network community detection(DIRT_LPA_wb_plus [2]) or community detection on GNN models. Solid lines are the mean over 10 independent runs. Shaded areas indicate the standard error.

109 **Supplementary Note 4: Mask Generation Protocol and the Control Experiments**

110 To recapitulate the biological growth of brain networks, we leveraged empirical human brain data from the
111 Human Connectome Project [4] and selected a parcellation atlas of 84 cortical areas [5]. We then extracted
112 both the mean inter-regional distance matrix and the structural connectivity (SC) matrix. To determine the
113 appropriate network sparsity, we analyzed binarized SC matrices from 758 participants, identifying an average
114 of approximately 2000 structural connections. Accordingly, the final connections in the RNN recurrent layer
115 were constrained by a 84×84 binary mask fixed at 2000 connections. The topology of this mask was
116 derived from the inter-regional distance matrix, where the connection probability between nodes u and v is
117 defined as $P(u, v) = E(u, v)^\eta$, with $E(u, v)$ representing the Euclidean distance and η governing the decay
118 of connection probability with distance. For the experiments presented in Fig. 4b, we set $\eta = -3.0$, and add
119 25 connections every 500 steps; additional control experiments examining connection counts, randomness
120 levels, and addition schedules are detailed as follows.

121 To isolate the effect of the incremental mechanism from other confounding factors, we conducted a series
122 of additional control experiments to rule out potential influences such as the total number of connections,
123 the degree of randomness, and the schedule for adding connections.

124 **Selection Strategy and Randomness.** We investigated the connection selection strategy. Here, we varied
125 the parameter η to control the importance of distance (e.g. the degree of randomness). As shown in the 1st
126 row in Supplementary Fig. 8, the magnitude of η significantly influences the overall height of the modularity
127 curves, confirming that stronger spatial constraints correlate with higher modularity.

128 **Robustness to Connection Schedules.** To test if the specific order of addition mattered, we explored
129 different schedules for adding connections while keeping the total number of added edges constant. As shown
130 in the 2nd row in Supplementary Fig. 8, we found no obvious difference between these schedules.

131 **Impact of Total Connection Density.** To study the impact of the total number of connections, we tested
132 2,000, 4,000, 5,000, 6,000, and the fully connected 7,056 edges while holding other conditions (like η) constant
133 (3rd row in Supplementary Fig. 8). As the total number of connections increases and the model transitions
134 from sparse to dense, modularity decreases in both training modes. In the all-at-once mode, modularity
135 gradually falls below 0.1 as the network becomes fully connected. This aligns with the results for the "Model
136 Size=64" case (which starts fully connected), effectively validating the consistency of our results.

137 **Controlling for Training Samples and Update Steps.** To study the impact of the total number of
138 training samples and update steps, we trained the models under the same training plan in Fig. 4b but for a
139 4x longer iterations. In 4th row of Supplementary Fig. 8, the vertical green dashed line marks a critical point:
140 after this point, the models in both curves possess identical connectivity (the same number and exact set of

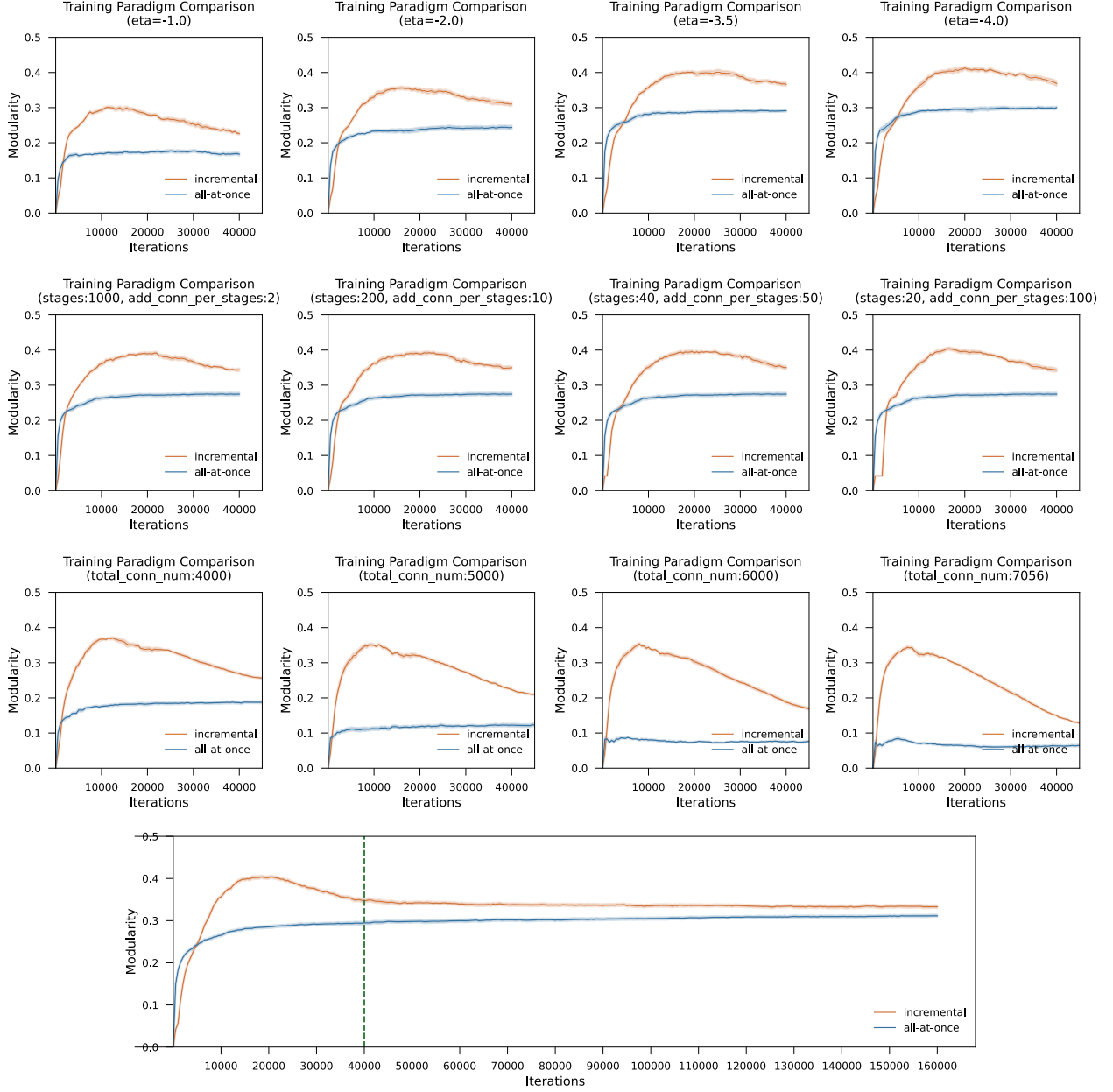
connections). It can be seen that even when the topology and subsequent training plan are identical (post-green line), the "incremental" method maintains higher modularity. Although the gap tends to narrow over time—which is expected, as the structural connectivity are identical after the green line—the persistence of higher modularity demonstrates that the difference is inherited from the structural characteristics formed during the earlier incremental phase, rather than being a byproduct of mismatched sample sizes or update counts.

Supplementary Note 5: Perturbation Analysis of models in Fig2. b

This analysis is specifically designed to quantify the weight perturbation robustness of the large-capacity (over-parameterized) model relative to smaller-capacity models. We conducted the perturbation by adding Gaussian white noise ($\mathcal{N}$) to the recurrent connection weight matrix (W_{rec}), simulating the effect of various random errors or noise that the model might encounter. To precisely control the perturbation intensity, we utilized the Signal-to-Noise Ratio (SNR), a scientific and standard technique in signal processing. This approach links the noise power to the power of the original weights (the signal), ensuring that the perturbation strength is independent of the model's intrinsic scale. This allows for a fair comparison of robustness across models of different capacities. We define the signal power (P_{signal}) as the total energy (sum of squares) of the original weight matrix, $P_{signal} = \sum W_{rec}^2$. For a given SNR (in dB), the required noise power P_{noise} is calculated as: $P_{noise} = P_{signal}/10^{(SNR_{dB}/10)}$. The randomly generated Gaussian noise is then scaled to possess exactly the required P_{noise} . We measure the incremental change in task loss ($\Delta Loss = Loss_{perturbed} - Loss_{original}$) after the perturbation. A smaller $\Delta Loss$ indicates greater robustness to that level of perturbation. The Supplementary Fig. 9 clearly demonstrates the following trend: for a given training step and SNR perturbation, models with larger capacity (larger N_{RNN}) show a significantly lower variance in $\Delta Loss$ compared to smaller-capacity models. Furthermore, the median $\Delta Loss$ is also lower in most cases for the larger models. This robustly supports our argument that a certain degree of over-parameterization endows the model with inherent robustness.

Supplementary Note 6: Statistical Model Comparison via Information Criteria

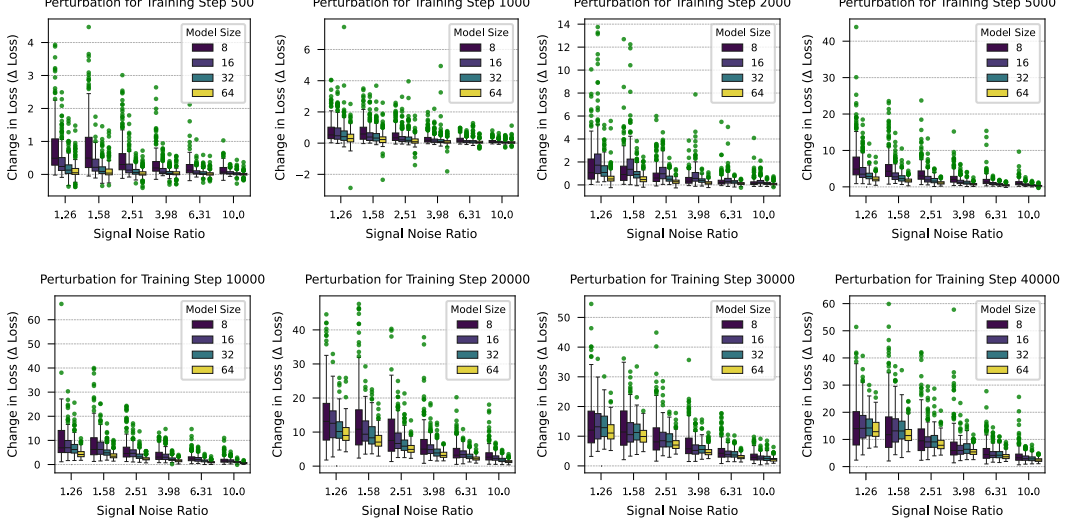
To disentangle the advantage conferred by additional degrees of freedom from genuine improvements in model fit, we calculated the Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC) for all generative models. Generative Network Models (GNMs) are evaluated based on their ability to reproduce the distributions of topological features rather than exact edge-to-edge overlaps. Consistent with this framework,



Supplementary Fig. 8. Ablation study for the comparison between “All-at-once” and “Incremental” strategies. Control experiments are conducted to ablate the effects of the degree of randomness (1st row), the connection-incremental schedule (2nd row), total connection number (3rd row) and total number of training iterations (4th row). Solid lines are the mean over 8 independent runs. Shaded areas indicate the standard error.

we estimated the likelihood of observing the empirical network’s topology under the distribution generated by the model parameters $\theta = \{\eta, \gamma\}$.

We characterized the topology of any given network using distributions of four key metrics: node degree (k), clustering coefficient (c), betweenness centrality (b), and edge length (e). For each metric, we extracted a vector of 7 summary statistics: the mean, variance, and 5 percentiles (10th, 30th, 50th, 70th, 90th).



Supplementary Fig. 9. Larger-capacity models show greater robustness to recurrent-weight perturbations. Models in Fig. 2b were perturbed at the indicated training checkpoints by adding Gaussian noise to the recurrent weights at different signal-to-noise ratios (SNRs). Perturbation sensitivity was measured as $\Delta \text{Loss} = \text{Loss}_{\text{perturbed}} - \text{Loss}_{\text{original}}$, with smaller values indicating greater robustness. For each model size, training checkpoint and SNR, $n = 20$ independently trained model instances were evaluated, with 10 independent noise realizations per model. Boxes show the interquartile range, center lines indicate the median, whiskers extend to 1.5 times the interquartile range, and colors denote model size.

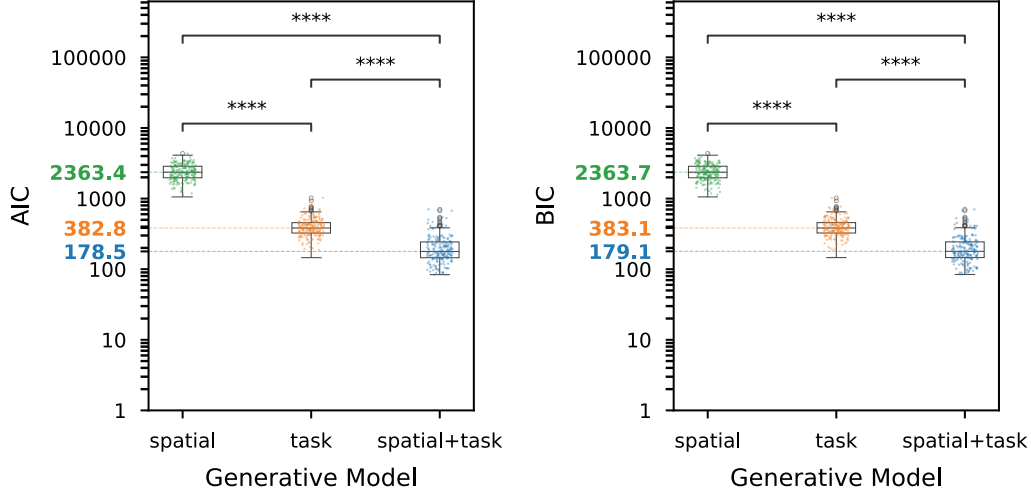
Concatenating these yielded a raw feature vector $\mathbf{x} \in \mathbb{R}^{28}$. To address the high collinearity among these features, we applied Principal Component Analysis (PCA) to the feature space of the simulated networks. We retained the first $m = 10$ principal components, denoted as vector $\mathbf{s}$, which accounted for $> 98\%$ of the total variance. For a given subject and parameter set θ , we generated $N = 200$ independent synthetic networks. The distribution of the dimensionality-reduced statistics $\mathbf{s}_{syn}$ was modeled as a multivariate Gaussian $\mathbf{s}_{syn} \sim \mathcal{N}(\boldsymbol{\mu}_\theta, \boldsymbol{\Sigma}_\theta)$ where $\boldsymbol{\mu}_\theta$ and $\boldsymbol{\Sigma}_\theta$ are the sample mean and covariance matrix estimated from the simulations. The validity of this Gaussian assumption was confirmed using the Henze-Zirkler multivariate normality test.

The log-likelihood of the empirical subject's data $\mathbf{s}_{gt}$ given parameters θ is:

$$\ln(\mathcal{L}(\theta|\mathbf{s}_{gt})) = -\frac{1}{2} [(\mathbf{s}_{gt} - \boldsymbol{\mu}_\theta)^T \boldsymbol{\Sigma}_\theta^{-1} (\mathbf{s}_{gt} - \boldsymbol{\mu}_\theta) + m \ln(2\pi) + \ln |\boldsymbol{\Sigma}_\theta|] \quad (1)$$

Calculation of AIC and BIC. We optimized θ to find the maximum likelihood estimate $\hat{\mathcal{L}}$. The Information Criteria were defined as: $AIC = 2k - 2\ln(\hat{\mathcal{L}})$ and $BIC = k \ln(n) - 2\ln(\hat{\mathcal{L}})$. Here, k represents the number of free parameters in the generative model, where we have $k = 1$ for the Single-constraint models (Spatial or Task) and $k = 2$ for the Combined model (Spatial + Task). The term n refers to the number of effective independent observations used in the likelihood evaluation. In this dimensionality-reduced feature space, $n = m = 10$.

189 **Statistical Results.** The “Spatial + Task” model consistently yielded the lowest AIC and BIC scores across
190 the cohort (Supplementary Fig. 10), demonstrating that the improvement in topological fidelity outweighs
191 the penalty for model complexity. Specially, the AIC value for “Spatial + Task” is significantly lower than
192 that for both “Spatial” ($t = 49.66, p < 0.0001$) and “Task” ($t = 17.61, p < 0.0001$). And the BIC value
193 for “Spatial + Task” is significantly lower than that for both “Spatial” ($t = 49.65, p < 0.0001$) and “Task”
194 ($t = 17.58, p < 0.0001$). For more detailed statistical results, see Supplementary Table. 5.



Supplementary Fig. 10. Model selection via Information Criteria. Comparison of Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC) across the three generative model families: Spatial-only (green), Task-only (orange), and Spatial+Task (blue). Lower values indicate a better trade-off between goodness-of-fit and model complexity. The combined “Spatial+Task” model consistently achieves the lowest scores, statistically outperforming single-constraint baselines even after penalizing for the additional parameter ($k = 2$ vs $k = 1$). Statistical significance was assessed via two-sample t -tests.

Supplementary Table. 5. Detailed statistical results for model comparison using information criteria. This table reports the precise statistics for the independent two-sample t -tests performed on AIC and BIC values across the three generative model families. For each comparison, the t -statistic, degrees of freedom (df), and exact p -value are provided. Sample size $N = 200$ represents the number of subjects (To balance the robustness of the statistical results and computational feasibility, we determined $N=200$ instead of 758.); here $df = 2N - 2$.

Comparison	Metric	t -statistic	df	p -value
Spatial vs. Task	AIC	44.90	398	7.04×10^{-158}
Task vs. Spatial+Task	AIC	17.61	398	9.55×10^{-52}
Spatial vs. Spatial+Task	AIC	49.66	398	1.18×10^{-172}
Spatial vs. Task	BIC	44.90	398	7.04×10^{-158}
Task vs. Spatial+Task	BIC	17.58	398	1.25×10^{-51}
Spatial vs. Spatial+Task	BIC	49.65	398	1.24×10^{-172}

195 **Supplementary Note 7: Mechanistic Account of Task-Driven Modularity and** 196 **Learning Efficiency**

197 This note provides an idealized mechanistic account for the emergence and utility of modular structure in
198 multitask recurrent neural networks. The goal is not to prove a universal theorem for all nonlinear trained
199 RNNs, but to formalize the main empirical chain observed in the manuscript:

task statistics $\rightarrow$ family-structured hidden representations
 $\rightarrow$ block-biased recurrent updates $\rightarrow$ modular recurrent weights
 $\rightarrow$ reduced destructive cross-task interference $\rightarrow$ improved learning efficiency.

200 This derivation follows the style of idealized model-based explanations used in recent work on modular
201 structure in artificial neural networks [6]. We distinguish between three levels of claims: (i) formal derivations
202 under idealized assumptions, (ii) toy-model closures that make the modularity statistic interpretable, and
203 (iii) mechanistic interpretations for incremental connectivity and temporal extensions. This distinction is
204 important because the following derivations are intended to support the interpretation of our empirical
205 findings, not to replace them with a fully general theory.

206 **Setup and local linearization.** We start from the recurrent network in the manuscript and consider
207 a local linearization around a stable operating regime, a standard simplification for analyzing recurrent
208 dynamics near a trained trajectory or fixed operating point [7, 8]. Let the hidden state obey the discrete-time
209 approximation

$$h_{t+1} = Jh_t + Bx_t + u_{m(t)} + \xi_t, \quad (2)$$

210 where $h_t \in \mathbb{R}^N$ is the hidden state, x_t is the stimulus drive, $u_{m(t)}$ is the task-dependent context drive,
211 $m(t) \in \{1, \dots, M\}$ indexes the task, ξ_t is zero-mean noise, and J is the linearized recurrent operator. We
212 assume $\rho(J) < 1$, so that the linearized dynamics are stable. A key point is that the raw one-hot task
213 label covariance is not itself block-structured. Any block structure must arise from task-conditioned hidden
214 statistics induced by learning, rather than from the one-hot rule input alone. We assume that tasks can be
215 grouped into a small number of compatible task families, reflecting the empirical observation that some task
216 pairs require disjoint representations while others share representations partially or almost fully.

217 **Hidden-state second moments inherit task-family structure.** For a fixed task m , define $\mu_m = \mathbb{E}[h_t \mid m]$
 218 and $\Sigma_h^{(m)} = \text{Cov}(h_t \mid m)$. Under Eq. (2), the task-conditional covariance satisfies the discrete Lyapunov
 219 equation $\Sigma_h^{(m)} = J\Sigma_h^{(m)}J^\top + B\Sigma_x^{(m)}B^\top + \Sigma_\xi$. When tasks are mixed with probabilities π_m , the law of total
 220 covariance gives

$$\Sigma_{\text{mix}} = \sum_{m=1}^M \pi_m \Sigma_h^{(m)} + \sum_{m=1}^M \pi_m (\mu_m - \mu)(\mu_m - \mu)^\top, \quad \mu = \sum_{m=1}^M \pi_m \mu_m. \quad (3)$$

221 Thus the mixed hidden covariance decomposes into within-task variability and between-task mean geometry.
 222 If related tasks induce nearby hidden means and similar within-task covariances, Σ_{mix} acquires a family-
 223 wise block structure. This formalizes the hypothesis that task co-embedding can modularize hidden-layer
 224 functional covariance. Importantly, this argument does not assume modular recurrent weights a priori; it
 225 only assumes that learning organizes related tasks into nearby hidden-state statistics.

226 **Capacity limits make full task separation impossible.** We next explain why the effect should be
 227 strongest when task load is high relative to representational capacity. Let $a_1, \dots, a_M \in \mathbb{R}^d$ be normalized
 228 task representations in a true latent subspace of dimension d , and define the Gram matrix $G_{mn} = \langle a_m, a_n \rangle$.
 229 The frame-potential bound, also known as the Welch bound in signal design [9], gives

$$\sum_{m,n=1}^M |\langle a_m, a_n \rangle|^2 = \text{tr}(G^2) \geq \frac{\text{tr}(G)^2}{\text{rank}(G)} \geq \frac{M^2}{d}. \quad (4)$$

230 Consequently,

$$\frac{1}{M(M-1)} \sum_{m \neq n} |\langle a_m, a_n \rangle|^2 \geq \frac{M-d}{d(M-1)}. \quad (5)$$

231 When $M > d$, this bound gives a positive lower bound on average task overlap; when $M \leq d$, the bound
 232 is non-positive and therefore only states the trivial fact that the average squared overlap is non-negative.
 233 Thus, as the task-to-capacity ratio grows beyond one, some task overlap becomes unavoidable. The bound
 234 itself does not prove that family-wise clustering is the unique optimum. Rather, it establishes that full task
 235 separation becomes impossible under limited capacity; clustered reuse among compatible tasks is then a
 236 natural low-interference compromise. In the manuscript, the task-to-dimensionality ratio uses PCA-based
 237 effective dimensionality and should be interpreted as an empirical proxy for this latent-capacity ratio, not
 238 as the exact theorem variable.

239 **From family-structured hidden activity to block-biased recurrent updates.** The previous sections
 240 explain why task-family structure can appear in hidden-state statistics. We now show how such repre-
 241 sentational structure can be translated into modular recurrent wiring through learning. Let P_g be the
 242 projection onto the dominant hidden subspace for task family g . For a task m in family g , assume
 243 $h_t^{(m)} = P_g z_t^{(m)} + r_t^{(m)}$, where $r_t^{(m)}$ is a small residual. Likewise, let the backpropagated recurrent error decom-
 244 pose as $\delta_{t+1}^{(m)} = Q_g e_t^{(m)} + s_t^{(m)}$, where Q_g is the dominant error-support projector for family g , approximately
 245 aligned with P_g , and $s_t^{(m)}$ is a small residual. This error-structure assumption is necessary: hidden-state
 246 block structure alone does not automatically imply block-structured weight updates. For recurrent weights,
 247 the gradient has the standard outer-product form

$$\nabla_{W_{\text{rec}}} L_m = \sum_t \delta_{t+1}^{(m)} h_t^{(m)\top}. \quad (6)$$

248 Substituting these decompositions gives

$$\begin{aligned} \delta_{t+1}^{(m)} h_t^{(m)\top} &= Q_g e_t^{(m)} z_t^{(m)\top} P_g + Q_g e_t^{(m)} r_t^{(m)\top} \\ &\quad + s_t^{(m)} z_t^{(m)\top} P_g + s_t^{(m)} r_t^{(m)\top}. \end{aligned} \quad (7)$$

249 Taking expectations,

$$\mathbb{E}[\Delta W_{\text{rec}}^{(m)}] = -\eta_{\text{lr}} \sum_t Q_g \mathbb{E}[e_t^{(m)} z_t^{(m)\top}] P_g + E_g, \quad (8)$$

250 where E_g collects residual terms and is small when the residual energies of $r_t^{(m)}$ and $s_t^{(m)}$ are small. Therefore,
 251 under family-structured hidden activity and family-structured error propagation, expected recurrent updates
 252 are block-biased: within-family update magnitude dominates cross-family update magnitude. Over training,
 253 these biased updates accumulate into block-dominant recurrent weights, providing a learning-dynamics bridge
 254 from functional co-embedding to structural modularity.

255 **Gradient factorization and quadratic suppression of interference.** We next explain why modular
 256 recurrent structure improves multitask optimization. Consider the steady-state linearized RNN

$$r_k = (I - \widetilde{W})^{-1} c_k, \quad \widetilde{W} = \alpha I + \beta W_{\text{rec}}, \quad (9)$$

257 and define the effective recurrent error

$$\widetilde{\delta}_k = (I - \widetilde{W}^\top)^{-1} \delta_k. \quad (10)$$

258 Differentiating the steady state with respect to W_{rec} gives

$$\frac{\partial L_k}{\partial W_{\text{rec}}} = \beta \tilde{\delta}_k r_k^\top. \quad (11)$$

259 Therefore, for tasks i and j , the inner product of vectorized recurrent gradients factorizes as

$$G_{ij} = \left\langle \text{vec} \left(\frac{\partial L_i}{\partial W_{\text{rec}}} \right), \text{vec} \left(\frac{\partial L_j}{\partial W_{\text{rec}}} \right) \right\rangle = \beta^2 (\tilde{\delta}_i^\top \tilde{\delta}_j) (r_i^\top r_j). \quad (12)$$

260 This factorization shows that cross-task interference is small whenever either forward activity overlap or

261 backward error overlap is small. Under modular structure, both overlaps are small across task families.

262 To quantify this point, write

$$W_{\text{rec}} = W_D + \gamma \widehat{W}_O, \quad (13)$$

263 where W_D is block diagonal, $\widehat{W}_O$ is the normalized inter-family perturbation, and γ is the perturbation

264 magnitude. Let $P = (I - \widetilde{W}_D)^{-1}$. If $\beta\gamma\|P\|\|\widehat{W}_O\| < 1$, the Neumann expansion gives

$$(I - \widetilde{W})^{-1} = \sum_{n=0}^{\infty} (\beta\gamma P \widehat{W}_O)^n P. \quad (14)$$

265 Thus, for two tasks from different dominant families, cross-family leakage in both r_k and $\tilde{\delta}_k$ is of order $O(\gamma)$,

266 so

$$r_i^\top r_j = O(\gamma), \quad \tilde{\delta}_i^\top \tilde{\delta}_j = O(\gamma). \quad (15)$$

267 Combining Eq. (15) with Eq. (12),

$$|G_{ij}| \leq C\gamma^2 + O(\gamma^3), \quad (16)$$

268 for some constant C depending on signal norms and the stability of the recurrent dynamics. The exact

269 prefactor is not essential; the key conclusion is the quadratic suppression of cross-family gradient interference

270 as inter-family coupling decreases.

271 **Reduced destructive interference improves learning efficiency.** Let the multitask objective be

$$\mathcal{L}(\theta) = \sum_{m=1}^M \pi_m \ell_m(\theta), \quad g_m = \nabla \ell_m(\theta), \quad \bar{g} = \sum_{m=1}^M \pi_m g_m. \quad (17)$$

272 If $\mathcal{L}$ is locally smooth, one gradient step gives [10]

$$\mathcal{L}(\theta - \eta_{\text{lr}} \bar{g}) \leq \mathcal{L}(\theta) - \eta_{\text{lr}} \left(1 - \frac{L\eta_{\text{lr}}}{2}\right) \|\bar{g}\|^2, \quad (18)$$

273 while

$$\|\bar{g}\|^2 = \sum_m \pi_m^2 \|g_m\|^2 + \sum_{m \neq n} \pi_m \pi_n \langle g_m, g_n \rangle. \quad (19)$$

274 Negative cross-task inner products create destructive interference and reduce local descent. Modular structure
 275 is useful when it suppresses destructive cross-family interference while preserving within-family sharing. This
 276 point is important: the argument does not claim that all cross-task alignment should be removed. Positive
 277 within-family alignment can be beneficial, whereas negative cross-family interference is costly, consistent
 278 with the motivation behind multi-task gradient-conflict methods [11].

279 If tasks are sampled stochastically during training, the gradient variance also depends on cross-task gradient
 280 inner products. For uniform sampling over K tasks, let $G_k = \nabla L_k$ and $\bar{G} = K^{-1} \sum_k G_k$. Then

$$\mathbb{E}_k \|G_k - \bar{G}\|^2 = \frac{K-1}{K^2} \sum_k \|G_k\|^2 - \frac{2}{K^2} \sum_{i < j} \langle G_i, G_j \rangle. \quad (20)$$

281 Thus, suppressing destructive cross-family interference improves learning through two coupled channels: it
 282 reduces cancellation in the average update and stabilizes stochastic multitask optimization. This mechanism
 283 explains why mid-training modularity is most predictive in capacity-limited networks, where task overlap
 284 cannot be avoided simply by increasing representational dimension.

285 **Capacity dependence, incremental connectivity, and empirical diagnostics.** Combining the capac-
 286 ity bound and the interference analysis gives the following interpretation. When latent capacity is high relative
 287 to task count, near-separated representations may be possible even without strong structural segregation.
 288 When task count exceeds effective representational capacity, average overlap becomes unavoidable, and clus-
 289 tered reuse among compatible tasks becomes a low-interference solution. In that regime, modularity is not
 290 merely descriptive; it is a structural means of reducing destructive cross-task interference. The connection-
 291 incremental results can be interpreted through the same lens, but should not be overstated as a theorem.
 292 Incremental connectivity defines a continuation process: early sparse subproblems encourage low-interference
 293 substructures, and later connection growth can preserve and amplify them through path dependence. Simi-
 294 larly, the steady-state derivation suggests an analogous temporal statement under BPTT if family support
 295 is approximately preserved through recurrence and backpropagation, but a complete temporal theorem is
 296 beyond the scope of this note.

297 The framework also yields direct empirical diagnostics. First, using modules estimated from neuron-similarity
 298 or recurrent-weight communities, one can decompose task gradients into intra- and inter-module energy:

$$E_{\text{intra}}(m) = \sum_g \|P_g G_m P_g\|_F^2, \quad E_{\text{inter}}(m) = \sum_{g \neq g'} \|P_{g'} G_m P_g\|_F^2, \quad (21)$$

299 where $G_m = \nabla_{W_{\text{rec}}} L_m$. The theory predicts that $E_{\text{intra}}/E_{\text{inter}}$ should increase as modular organization forms.
 300 Second, within-family and cross-family gradient cosine similarities can be compared over training. The theory
 301 predicts preserved or positive within-family alignment but reduced destructive cross-family interference,
 302 especially in capacity-limited models.

303 Overall, such a reasoning procedure explains why task-driven modularity emerges as a useful structural
 304 adaptation under multitask pressure and limited representational capacity, while keeping the theoretical
 305 claims appropriately scoped to idealized assumptions.

306 References

- 307 [1] Yang, G. R., Joglekar, M. R., Song, H. F., Newsome, W. T. & Wang, X.-J. Task representations in
 308 neural networks trained to perform many cognitive tasks. *Nature neuroscience* **22**, 297–306 (2019).
- 309 [2] Beckett, S. J. Improved community detection in weighted bipartite networks. *Royal Society open science*
 310 **3**, 140536 (2016).
- 311 [3] Krizhevsky, A., Hinton, G. *et al.* Learning multiple layers of features from tiny images (2009).
- 312 [4] Van Essen, D. C. *et al.* The wu-minn human connectome project: an overview. *Neuroimage* **80**, 62–79
 313 (2013).
- 314 [5] Rolls, E. T., Huang, C.-C., Lin, C.-P., Feng, J. & Joliot, M. Automated anatomical labelling atlas 3.
 315 *Neuroimage* **206**, 116189 (2020).
- 316 [6] Gu, S., Mattar, M. G., Tang, H. & Pan, G. Emergence and reconfiguration of modular structure for
 317 artificial neural networks during continual familiarity detection. *Science Advances* **10**, eadm8430 (2024).
- 318 [7] Sussillo, D. & Barak, O. Opening the black box: Low-dimensional dynamics in high-dimensional
 319 recurrent neural networks. *Neural Computation* **25**, 626–649 (2013).

- 320 [8] Saxe, A. M., McClelland, J. L. & Ganguli, S. Exact solutions to the nonlinear dynamics of learning in
321 deep linear neural networks. *arXiv preprint arXiv:1312.6120* (2013).
- 322 [9] Welch, L. R. Lower bounds on the maximum cross correlation of signals. *IEEE Transactions on*
323 *Information Theory* **20**, 397–399 (1974).
- 324 [10] Bottou, L., Curtis, F. E. & Nocedal, J. Optimization methods for large-scale machine learning. *SIAM*
325 *Review* **60**, 223–311 (2018).
- 326 [11] Yu, T. *et al.* Gradient surgery for multi-task learning. *Advances in neural information processing*
327 *systems* **33**, 5824–5836 (2020).