Brain-Like Processing Pathways Form in Models With Heterogeneous Experts

Jack Cook[1] **Danyal Akarca**[2] **Rui Ponte Costa**[1,*] **Jascha Achterberg**[1,*]
[1]Centre for Neural Circuits and Behaviour, University of Oxford
[2]Department of Electrical and Electronic Engineering, Imperial College London
[*]Joint senior authors

Abstract

The brain is made up of a vast set of heterogeneous regions that dynamically organize into pathways as a function of task demands. Examples of such pathways can be found in the interactions between cortical and subcortical networks during learning, or in sub-networks specializing for task characteristics such as difficulty or modality. Despite the large role these pathways play in cognition, the mechanisms through which brain regions organize into pathways remain unclear. In this work, we use an extension of the Heterogeneous Mixture-of-Experts architecture to show that heterogeneous regions do not form processing pathways by themselves, implying that the brain likely implements specific constraints which result in the reliable formation of pathways. We identify three biologically relevant inductive biases that encourage pathway formation: a routing cost imposed on the use of more complex regions, a scaling factor that reduces this cost when task performance is low, and randomized expert dropout. When comparing our resulting *Mixture-of-Pathways* model with the brain, we observe that the artificial pathways in our model match how the brain uses cortical and subcortical systems to learn and solve tasks of varying difficulty. In summary, we introduce a novel framework for investigating how the brain forms task-specific pathways through inductive biases, and the effects these biases have on the behavior of Mixture-of-Experts models.

1 Introduction

The brain is made up of many heterogeneous regions distinguished by features such as connectivity, cell types, neurotransmitters, and functional specialization [1–4]. To support complex behavior, the mammalian brain dynamically organizes these regions into diverse networks and processing pathways [5], allowing it to adapt to different inputs and task demands. This principle spans sensory systems [6–8], cognitive networks [9], emotion-related circuits [10], and face perception [11]. Notably, pathway formation is highly dynamic: regions can participate in many pathways, allowing cognitive processes to arise from the joint activations of specific groups of regions. While theoretical work has shown how heterogeneous regions and modules can develop within networks [12–14], how these organize into large-scale pathways remains poorly understood.

The importance of studying pathway formation and coordination extends beyond neuroscience: it is also becoming increasingly relevant in machine learning research. As models have evolved from small networks with a couple of layers to large system-level architectures, achieving complex function while maintaining efficiency has become critical. One recent development toward this goal is the Mixture-of-Experts (MoE) architecture [15, 16], which contains specialized experts that are selectively activated based on the current input. This should create pathways between experts that selectively respond to inputs of varying complexity [17] to make efficient use of computational resources [18, 19]. However, this theorized specialization of experts appears to be limited in practice [20, 21], making it difficult for specialized task-complexity-related pathways to form.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

These findings raise the question, how do stable and functionally relevant pathways form in networks of distributed heterogeneous experts? Do heterogeneous regions automatically group into such pathways, or are additional priors required? Once pathways develop in models, do they show the same context-aware adaptability that has been observed in the brain? To address these questions, we introduce a neural network architecture made up of heterogeneous experts to study the conditions under which processing pathways form, and the degree to which these pathways resemble those studied in the brain. Specifically, we adapt the Heterogeneous Mixture-of-Experts (HMoE) architecture [22, 23], in which information may be dynamically routed to computational experts, or regions, of varying sizes. In our model, unlike prior work, each expert is implemented as a recurrent network that could be considered a standalone model or brain region. We study the pathway formation in this architecture while we train models to learn 82 time-series-based cognitive tasks of varying difficulty [24]. Through these analyses, we find:

- Layers of heterogeneous experts do not automatically form recognizable pathways.
- Instead, inductive biases are required for pathways to form: (i) a routing cost that penalizes the model for using larger experts, (ii) scaling the routing cost based on task performance, and (iii) random expert dropout. These result in the formation of a *Mixture-of-Pathways*.
- The pathways that form in our new *Mixture-of-Pathways* architecture mirror the interactions between cortical and subcortical pathways in the brain during learning, and are in line with the dynamics of the brain's multiple-demand system [9, 25].

To arrive at these findings, in the following we start by analyzing the usage of experts of a baseline model with HMoE layers. We then develop specific inductive biases that encourage pathways to form, before finally comparing these pathways to observations made in the brain.

2 Related Work

Brain-like modularity and regional heterogeneity can be induced in neural networks through priors and training procedures to explain how such features develop in the brain [14, 26–30]. The priors that are especially relevant in the context of this work relate to metabolic cost and energy efficiency, which are crucial in determining the brain's circuitry and function [13, 31–34].

While the above work often focuses on starting from fully-connected networks to observe the formation of modules and regions, modeling multi-region interactions has also become possible with modern methods [35–40]. This line of work has revealed how joint computation can be implemented through interactions between independent modules [5]. However, these multi-region models generally predefine a specific circuit structure with a small set of regions, preventing further study on how regions come to interact in the first place. Notable work that allows for dynamic (non-fixed) interaction of multiple independent regions often assumes networks which are not able to learn tasks [41, 42], though [43] stands out with a trainable network made up of individual RNNs. Their multi-region networks can change their connectivity during learning, but cannot route information based on task context. The work most closely aligned with our goal is [44], which studies how spatial (metabolic) constraints in feed-forward networks can form visual processing streams. However, it too does not consider how regions may change their interaction as a function of context, and does not allow for solving standard time-series-based cognitive tasks.

In the context of artificial intelligence, the introduction outlined how the popular Mixture-of-Experts architecture [15, 16] is relevant to our question of how specialized regions dynamically organize into processing pathways. Recent efforts to build networks out of experts that vary in terms of their architecture [17, 22, 23] and function [20] are especially relevant here. Moreover, work has argued that routing pathways are a powerful method for handling the complex data-flow of these otherwise efficient architectures [19, 45], but without investigating how adaptable pathways can be encouraged to form. In neuromorphic computing it has been shown to be possible to implement brain-like visual processing pathways to achieve efficient processing [46], but with a predefined static architecture.

3 Methods

In this work, we aim to identify the mechanisms by which pathways form between heterogeneous regions, and how those pathways are used across a diverse set of tasks. To study this computationally,

we need a model made up of heterogeneous experts, analogous to brain regions, that work together to solve many different tasks. We create such a model by extending the Heterogeneous Mixture-of-Experts architecture [22] and training it on the Mod-Cog set of time-series-based cognitive tasks [24].

3.1 Model Architecture: Heterogeneous Mixture-of-Experts

Mixture-of-Experts models (MoEs) [15, 16] are characterized by their layers, which contain multiple smaller models alongside a router model, which decides which experts should process the input at each timestep. Specifically, the router determines the weight with which each expert contributes to the layer's final output. Experts can also be excluded, by setting an expert's weight to zero. In most modern MoEs, the experts are large scale MLPs placed in between attention layers, which are typically activated at every timestep. In Heterogeneous Mixture-of-Expert models [22] (HMoEs), the experts can vary in terms of their sizes and activation functions. We extend the HMoE architecture with several significant adaptations. Namely, each layer of our model contains three experts: two GRUs with 16 and 32 neurons respectively, and one skip connection, which allows the model to choose to perform no computation for a given timestep [17]. Our implementation uses GRUs with 64 neurons as routers and does not include any additional layers between HMoE layers. We use this setup as a baseline for our investigations (Figure 1). In the following sections, we will introduce additional inductive biases to this baseline architecture, resulting in our final Mixture-of-Pathways model. The algorithm for this full model is described in Algorithm 1.

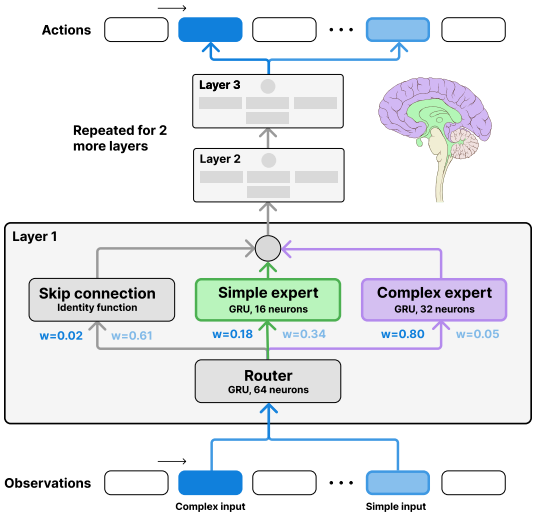


Figure 1: **Schematic of our baseline model architecture**. Information is passed through three layers, each of which can dynamically route information to experts of different computational complexity.

Algorithm 1: Mixture-of-Pathways training protocol. Full details in Appendix A.1.

Data: Task dataset D, Experts $E = \{e_1, e_2, \ldots, e_n\}$, Routers $R = \{r_1, r_2, \ldots, r_n\}$
Initialize routers and heterogeneous experts, set h_0 to task input;
for *each training step* **do**
  Sample batch b from D;
  for *each task i, timestep $t \in b$* **do**
    for *each layer l with experts $e_j \in l$* **do**
      Compute routing weights $w_j = \text{softmax}(r_l(h_{l-1}))$;
      Apply expert dropout (Not in baseline architecture; see Section 4.2) ;
      Compute expert activations w_j for each expert $e_j \in l$;
      Combine outputs: $h_l = \sum_j w_j h_{l,j}$;
    end
    Compute baseline model losses: L_{fix} and $L_{\text{response},i}$;
    Compute L_{routing} loss (Not in baseline architecture; see Section 4.1) ;
    Compute $L_{\text{total}} = L_{\text{fix}} + L_{\text{routing}} + \sum_i L_{\text{response},i}$;
  end
  Update parameters using Schedule-Free AdamW optimizer [47];
end

3.2 Evaluation with Cognitive Tasks

To evaluate how pathways are formed and used across tasks with different characteristics, we use the Mod-Cog task set, which contains 82 time-series-based cognitive tasks [24]. This is an expansion of

the popular NeuroGym framework [48], which contains tasks like Go-NoGo or two-stimuli integration tasks (see Appendix A.2 for task details). Importantly for us, the tasks vary in difficulty due to their varied inputs, decision rules, and delay lengths. Generally, tasks range from 0.8 to 3 seconds in duration, during which we sample information at timesteps 100 milliseconds apart. At each timestep, the model receives a 115-dimensional input made up of four components: a 1-dimensional fixation input, two 16-dimensional stimuli, and an 82-dimensional one-hot encoding of the active task, which we pass through a 16-dimensional learned embedding layer. While the fixation input is active, the model should always output zero. After the fixation period, the model needs to use the observed information to output the correct choices during the response period.

We train our models over 10 epochs, each containing 1000 training steps. At each training step, models are given a $128 \times 350 \times 115$ matrix of input data, representing 128 batches of task sequences that are 350 timesteps long, with 115 features at each timestep. These task sequences contain many individual tasks: the average task is about 20 timesteps long, meaning that in each batch, models observe about 27 trials of each task. Models are trained with a cross entropy loss $L_{\text{response},i}$ for the correct response during the response period of task i. An additional fixation loss L_{fix} encourages the model to output zero during the fixation period. All losses used in our analyses are detailed in Appendix A.3. Training one model takes roughly 1 hour on a single NVIDIA T4 GPU. Details on implementation and code access are outlined in Appendix A.1.

Once models have learned to solve each task by routing information between experts, we can study the conditions under which processing pathways form between layers. In the following we will first study the routing behavior of our baseline architecture. We will then show how the additional inductive biases described in Algorithm 1 result in the formation of pathways. Finally, we will test the degree to which these pathways resemble established processing pathways in the brain.

4 What Causes Pathways to Form?

In this section, we investigate the conditions under which pathways form between layers of heterogeneous experts. We set three criteria to determine whether pathways have formed:

1. Pathways should be **consistent** with respect to tasks, meaning that when two models are trained to solve the same tasks, they should have structurally similar pathways.
2. Pathways should be **self-sufficient**, meaning that when experts outside of a pathway are removed, then the model's overall performance should remain largely intact.
3. Pathways should be **distinct**, meaning that several different pathways should be used to solve groups of tasks with varying characteristics.

4.1 Pathway Consistency

To see if experts form consistent, task-driven pathways, we train 20 randomly initialized models with the same settings and compare their routing patterns on the same set of 82 cognitive tasks. This allows us to examine whether models use similar sets of experts to solve the same tasks, such as whether smaller experts are reliably used to solve simpler tasks, or vice versa. We first do this with a baseline model made up of three HMoE layers, and then test each model on 50 trials of each task while recording the routing weights assigned to each expert at each timestep (w values in Figure 1). To test whether routing is stable across training runs, we use these weights to calculate each model's *Learned Pathway Complexity* for each task i (LPC_i) as follows:

$$LPC_i = \frac{1}{T_i} \sum_{t}^{T_i} \sum_{j}^{E} w_{i,j,t} s_j^2 \tag{1}$$

This metric is calculated by multiplying the weight $w_{i,j,t}$ assigned by the router to each expert j at each timestep t by the squared size s_j^2 of expert j while the model solves task i. This is then averaged across the total timesteps T_i of each task i to ensure that longer tasks are not biased toward having larger LPCs. This results in a LPC value for each of the 82 tasks and 20 model runs (see Appendix A.4 for an example calculation). The squaring of expert sizes is motivated by the $O(s_j^2)$ cost of storing each expert's weight matrix in memory, and we expand on the suitability of using s_j as

a measure of each expert's complexity in Appendix A.5. Skip connections are free to use. To measure pathway consistency, we can now correlate this list of LPC values across training runs. For the baseline model, we find that models are not consistent across training runs (mean pairwise correlation of 0.0324, Figure 2), suggesting that the baseline model does not form any stable and task-related processing pathways by default. Therefore, we next want to explore which specific inductive biases may result in such pathways.

Theories of metabolic optimization and cost minimization are core parts of our understanding of the brain's computations [33, 49, 50]. The reduction of energy consumption has been a powerful source of priors for building brain-like neural networks [13, 31, 34] and more generally achieving brain-inspired computing [46, 51]. Hence we hypothesize that regularizing the routing weights by making it more expensive to route to more complex experts might cause replicable pathways to develop, as observed in the brain. We do so by incorporating the LPC_i (from Equation 1) into the model's loss, making it more costly for the model to activate more complex experts. Finally, to avoid convergence on the local minimum of only using the smallest experts without solving any tasks[1], we add a normalization strategy, dividing each LPC_i by $L_{\text{response},i}$, the cross-entropy loss measuring the model's performance on task $i \in \mathcal{T}$. In addition to helping with convergence, this normalization term can also be viewed as helping our model more directly control the cognitive effort, or processing power, with which it solves a task. We discuss this further in Section 6.

Adding these additional components to the loss results in the following equation, where L_{fix} and $L_{\text{response},i}$ are the standard task-based losses described in Section 3.2. A small value ϵ is added to ensure that if the model solves tasks perfectly, the routing loss is not ∞. All loss calculations are outlined in detail in Appendix A.3 and A.4.

$$L = L_{\text{fix}} + \sum_{i}^{\mathcal{T}} (L_{\text{response},i} + \frac{\alpha LPC_i}{L_{\text{response},i} + \epsilon}) \tag{2}$$

We now evaluate the routing consistency of models trained with this custom loss function. Figure 2 shows that our expectations are confirmed: on its own, adding the LPC_i for each task creates more consistent routing (mean pairwise correlation of 0.15, significant over baseline with $p < 0.01$). Scaling this term by the model's performance on each task $L_{\text{response},i}$ amplifies this effect, encouraging models across training runs to converge on more consistent routing patterns (mean pairwise correlation of 0.71, significant over baseline with $p < 0.001$).

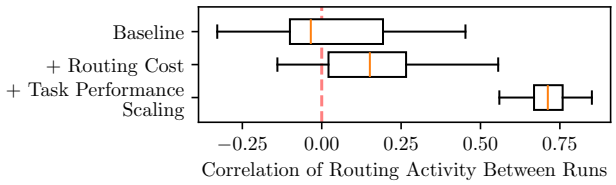


Figure 2: **Models trained with routing cost and task-based scaling exhibit more stable routing.** Correlations are calculated between the routing patterns across 20 training runs of each model setup.

4.2 Self-Sufficiency of Pathways

Our second criterion measures whether formed pathways are self-sufficient, meaning that removing an expert that is not part of the currently activated pathway should only minimally impact the performance of the model. To test for self-sufficiency, we first evaluate whether models are still able to perform tasks when they are prevented from using experts that have been assigned low routing weights. We find that our baseline models are extremely sensitive to this deactivation: if they are prevented from using experts with $w < 0.025$, which only contribute 2.5% or less to each layer's output, average task accuracy drops from 98.2% to 16.4%. This shows that while models learn replicable routing patterns, these are not yet pathways, as they rely on all the experts.

We speculate that pathway self-sufficiency can be achieved by stochastic dropout of experts with low routing weights. Dropout is especially interesting as it is an established principle for achieving more

[1] Note that routers converging to local minima is an established phenomenon in Mixture-of-Experts models, as there is a bias to rely on the expert that learns the task, or decreases the overall loss as in our case, first [16, 22, 52, 53].

robust neural networks [54] and has also been linked to the stochastic nature of signal processing in neuroscience [55, 56]. We implement *expert dropout* by randomly deactivating experts that contribute very little to the output during training. The probability p_j with which expert j is deactivated at a given timestep is determined as follows:

$$p_j = \begin{cases} \beta - \frac{\beta}{\gamma} w_j, & \text{if } w_j < \gamma \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

We set γ to 0.1, meaning that experts contributing 10% or more to the output of a layer are never deactivated. As this contribution weight decreases to zero, this probability increases linearly to β. To identify how much dropout is needed to improve robustness, we train 11 groups of models with β values ranging from $0, 0.1, ..., 0.9, 1$ using Equation 3, with 10 models in each group. We evaluate each model on 50 trials per task, blocking experts with routing weights below 11 values: $0, 0.025, \ldots, 0.225, 0.25$. Accuracy is averaged across the 10 models in each group. We find that expert dropout has a relatively minor impact on performance, while significantly improving the robustness of the pathways that form. Figure 3 shows that for models trained with a maximum dropout of 80% ($\beta = 0.8$), this drop in accuracy is small (from 85.8% to 74.4%). This motivates us to set $\beta = 0.8$ for our models in the remainder of this work. Note that routing consistency (Section 4.1) remains high with dropout, shown through an average pairwise correlation of 0.51, which is significant over the baseline with $p < 0.0001$ (see Appendix A.4).

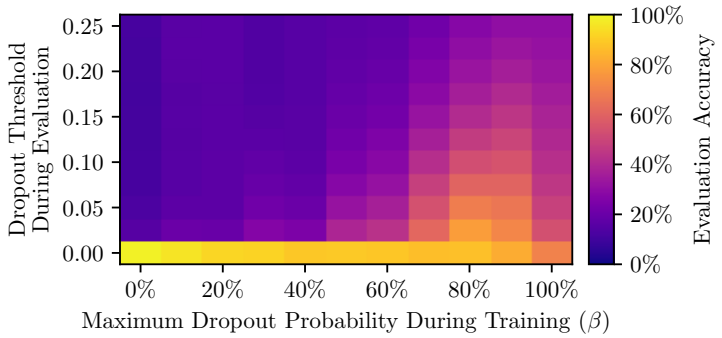


Figure 3: **Model accuracy after removing low-weighted experts across different training dropout levels.** If models trained without dropout ($\beta = 0$) are prevented from using experts that contribute very little to the output, accuracy drops precipitously, from 98.2% to 15.1%. By comparison, the accuracy of models trained with a maximum dropout value of $\beta = 0.8$ only drops from 86.5% to 77.7%.

4.3 Distinct Pathways Across and Within Tasks

For our final criterion, we want to identify whether meaningful patterns of expert usage develop across tasks and timescales within our model. To do this, we record the routing patterns across 50 trials of each task for both the baseline model and our final model, which is trained using the routing cost with task performance scaling and expert dropout. To visualize how routing varies across layers, tasks, and time, we average routing patterns for each task in three phases: (i) before the stimulus is shown, (ii) while the stimulus is shown, and (iii) during the response period. We apply K-means clustering ($k = 10$) to these matrices to identify groups of tasks that use similar pathways.

In our model, we observe a structured usage of expert pathways (Figure 4): during the pre-stimulus phase, models primarily rely on the cheap skip connections, as no information needs to be processed yet. For some tasks, increasingly complex experts are activated with the onset of stimuli. However, since the model still only needs to output zero during this phase, most tasks continue to leverage the cheap 'all-skip' pathway until the response phase. During this final phase, we see very rich dynamics of pathways being differentially activated across tasks and time periods in our model. This shows how processing pathways interact over the time of a trial, with different combinations of experts activated over tasks and time periods. The following sections analyze these dynamics in more detail, especially in comparison to the dynamics observed in the brain. Importantly, for the baseline model, clusters do not seem to employ very different combinations of experts across tasks. The more differentiated usage of pathways across clusters can be seen in the distribution of the numbers of tasks contained in a given cluster (Figure 4, left). Here, our model shows a very distinct power-law distribution of several large clusters containing > 20 tasks and many small clusters capturing task-specific pathway

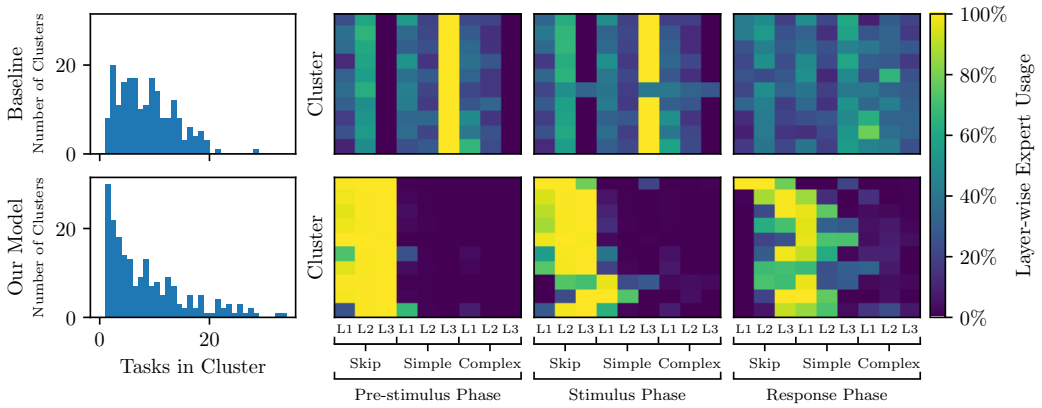


Figure 4: **Task clustering derived from expert usage patterns.** Clusters are averaged over three phases of each task: the pre-stimulus phase, during which the task is known but no input data is presented, the stimulus phase, during which input data is presented, and the response phase. Left: Sizes of clusters across training runs. Right: Average routing weights across training runs by cluster (y-axis) and task phase. In each phase, expert usages within each layer (across each layer's skip connection, simple expert, and complex expert) sum to 100%. The complex expert is rarely used in the first two phases, during which the model outputs zero, but has an average usage as high as 11% in the most complex cluster (see Appendix A.8). L1 / L2 / L3 = Layer 1 / Layer 2 / Layer 3.

usages. The baseline model, on the other hand, seems to distribute the tasks more evenly across clusters, indicating that there are much less distinct routing patterns. This can be shown quantitatively in the sizes of the largest clusters for each model run, which are significantly larger for our model than for the baseline model across 10 different random seeds ($p < 0.0001$). Further visualizations are provided in Appendices A.7 and A.8.

Our results so far show that pathways do not automatically develop from a heterogeneous mixture of experts. Training models with a routing-complexity loss, scaling it based on task performance, and adding expert dropout, all encourage stable pathways to form. These three features define our *Mixture-of-Pathways* (MoP) model. In the following section, we investigate whether the pathways that form in our MoP model mirror the pathways observed within the primate brain.

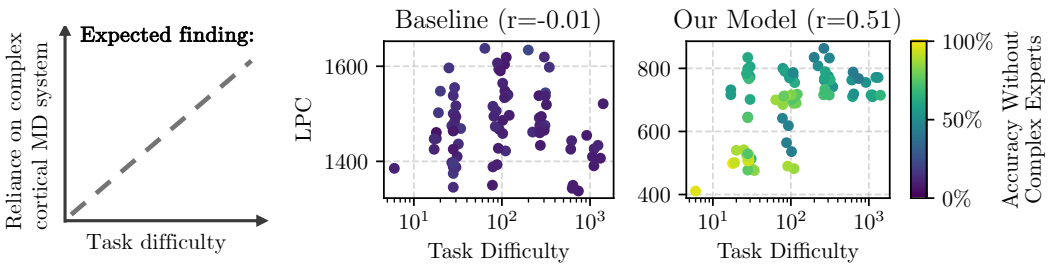
5 Do Artificial Pathways Behave Like the Brain's Pathways?

In the previous section, we showed how our brain-inspired architectural contributions resulted in the formation of a mixture of processing pathways in our model. Now, we will evaluate the degree to which these artificial pathways resemble the behavior of established processing pathways in the brain. Our analyses primarily focus on pathways and dynamics of the brain relating to task difficulty.

5.1 Solving Tasks of Varying Difficulty

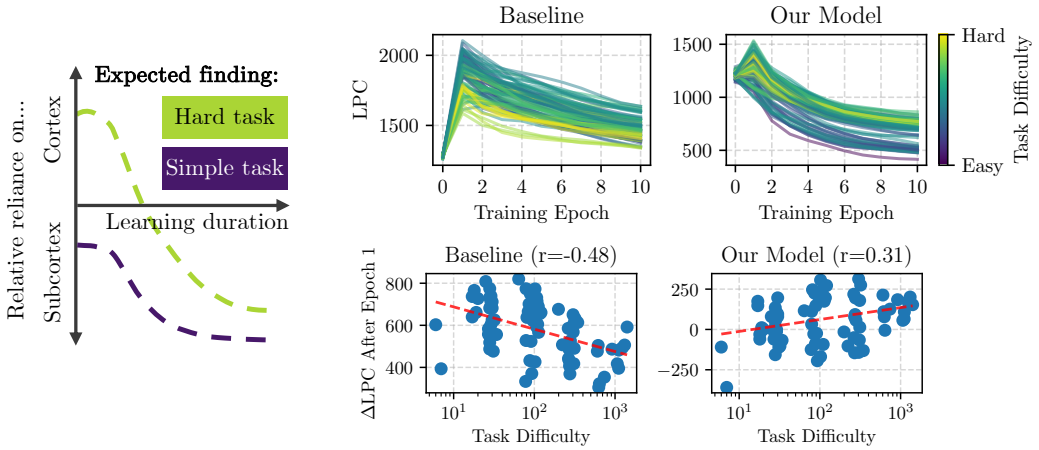
When analyzing brain activations across tasks with varying levels of difficulty, there is a distinct group of activations in a large frontoparietal network when solving complicated tasks. Since this network is activate while solving any difficult task, it was named the multiple-demand (MD) system [57, 58]. It can be identified both in humans and non-human primates [4, 59]. With this in mind, we now want to test whether the selection of experts used to solve a task is indicative of task difficulty.

To relate these findings from the MD system to our model, we expect that when solving tasks of increasing difficulty, our model should learn to activate increasingly complex regions (schematic in Figure 5). We test this by measuring the correlation between a task's difficulty and the learned pathway complexity (LPC) used by the model to solve the task. We quantify the difficulty of a task by the number of training steps it takes a standalone GRU to learn the task (see Appendix A.9 for details and alternative ways of quantifying task difficulty). We find that our full MoP model shows a positive significant relationship, whereas the baseline model does not, matching our expectations.



(a) Expected results (b) Empirical results

Figure 5: **Our model allocates less computation toward solving simpler tasks.** Additionally, when the most complex experts in each layer are disabled, our model is still able to solve the simplest tasks with high accuracy.



(a) Expected results (b) Empirical results

Figure 6: **Our model moves complex tasks to more complex pathways at the start of training to support the learning process.** (a) Conceptual schematic of cortical-subcortical interactions. (b) Top row shows the pathway complexity over learning per task averaged across 10 training runs. Lower row shows the change in pathway complexity between model initialization and the end of the first epoch as a function of task difficulty. Our model specifically relies on complex pathways to learn difficult tasks, similar to how the brain relies on complex pathways to support the acquisition of complex skills, even if they are gradually moved to simpler pathways later on.

Furthermore, it is known that patients with lesions to their MD system struggle solving difficult tasks but their ability to solve simple tasks usually is unaffected [60, 61]. We find that the same is true in our models: if the most complex expert in each layer is lesioned, our model can still solve simple tasks with high accuracy, but accuracy on difficulty tasks drops significantly. The same is not true of the baseline model: performance on all tasks drops to near-chance.

5.2 Learning Tasks of Varying Difficulty

A more nuanced view of pathways in the brain comes from observing how tasks of varying difficulty are learned over time. Here, an interesting distinction between complex and simple tasks is observed: while simple tasks can be learned through simpler (subcortical) regions alone, complex tasks require more complex (cortical) regions for learning [62–65] (see schematic in Figure 6a). However, as learning continues, even complex task skills are often "transferred" from complex to simple brain regions. This is possible despite the simple pathway not being sufficient to drive the learning process in the first place. We now want to test whether this phenomenon can be observed in our model.

Table 1: Task accuracy and pathway metrics for modified versions of our model.

Model	Accuracy	Fig. 5 Correlation	Fig. 6b Correlation
Baseline	91.1% ± 8.9%	-0.01	-0.49***
Our model†	83.0% ± 15.5%	0.54***	0.31**
Without dropout	90.1% ± 8.9%	0.55***	0.03
$\alpha = 1e^{-4}$	69.0% ± 20.1%	-0.57***	-0.37***
$\alpha = 1e^{-6}$	89.7% ± 9.0%	**0.62****	0.18
Without task embeddings	83.0% ± 14.2%	0.58***	**0.58****
Router dim = 32	83.2% ± 14.5%	0.46***	0.33**
Router dim = 128	81.9% ± 16.9%	0.35**	0.27*

† With dropout, $\alpha = 1e^{-5}$, task embeddings, and router dim = 64
* $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$

To study this effect, we track the complexities of the learned pathways across tasks over the duration of learning. Figure 6 shows these results across models: our MoP model seems to indeed learn complex tasks by first increasing their pathway complexity, but then reducing it gradually during learning. In contrast, very simple tasks do not increase in pathway complexity at all over learning. We now want to quantify this effect. To translate Figure 6a to our models, we can quantify to which degree the pathway complexity of a given task is increasing or decreasing after the first training epoch. This is a measure of how much the model specifically uses a more complex pathway to learn a given task. Based on findings from neuroscience, we would expect complex tasks to be explicitly moved to more complex pathways, relative to the random starting point, whereas this should not be necessary for simple tasks [35, 62, 66]. Figure 6b shows that this phenomenon can be observed in our model. The more difficult a task is, the more its pathway complexity increases at the start of learning, with the simplest tasks immediately getting routed toward simpler pathways ($r = 0.31$, $p = 0.0040$). This is not true in the baseline model, where we observe the opposite effect: the pathway complexity used to solve the most difficult tasks increases the least at the start of learning ($r = -0.48$, $p < 0.0001$). Upon further inspection, this happens because in an effort to minimize the routing loss, the baseline model learns to push tasks that it is unable to solve toward simpler pathways prematurely. As a result, the baseline model fails to learn the most difficult tasks until the very end of the training process.

5.3 Ablations

Lastly, we investigate how changes made to our model can alter the degree to which it resembles pathways in the brain, as discussed in Sections 5.1 and 5.2. Table 1 shows the effects of design parameters on the correlations reported in Figures 5 and 6. Notably, we found that when trained with our loss function that scales based on LPC but without dropout, our model exhibits the effect shown in Figure 5, but not the effect shown in Figure 6. This indicates that the finding in Figure 6 is due to an interaction between the LPC scaling in our loss function and dropout, and can not be explained by training with the LPC regularization alone. We speculate that expert dropout forces the model to be explicit about which pathway is used in learning, and is crucial for creating brain-like learning dynamics. This finding highlights how our model's behavior specifically results from the interplay of all three of our proposed inductive biases.

We also find that removing the task embedding layer improves the finding in Figure 6, however its removal drastically slows down training since the active task is represented with 82 dimensions at each timestep rather than 16 (see Section 3.2). Changing the router's hidden size does not meaningfully affect our results. Increasing or decreasing the penalty for using large experts (α) naturally has a meaningful effect on the results, where too strong of a penalty inhibits learning the tasks as well as general convergence of the model, and too weak of a penalty does not sufficiently motivate the model to reduce its usage of complex experts. All rows in the table are averaged over 10 runs.

6 Discussion

In this work, we adapted the heterogeneous Mixture-of-Experts architecture to investigate how brain-like processing pathways can form between layers of heterogeneous experts. While these experts do

not form pathways on their own, once trained with a routing-cost loss, task-performance scaling, and expert dropout, we find that they create a *Mixture of Pathways*. Our model provides an account of task-specific brain-wide pathways commonly observed in neuroscience.

These findings are relevant for neuroscience, as energetic and processing complexity related priors have been key explanatory mechanisms for how the structure and function of the brain arises [13, 32, 49, 50]. We show that incentivizing models to learn to prioritize simple experts drives the development of brain-like processing pathways from a heterogeneous set of expert models. Additionally, we show how stochasticity of signals is important for learning self-sufficient processing networks. Our model represents an exciting new architecture which can be expanded in the future to study additional heterogeneities present in the connectome, such as varying cell-types and neurotransmitters. Region-specific models, such as those for the hippocampus [67], could be integrated within our architecture.

The brain's implementation of the complexity-guided routing mechanism would likely be found in thalamic nuclei, which regulate information flow between cortical regions [68]. Two systems could modulate pathway selection based on metabolic costs: norepinephrine release from the locus coeruleus, which controls cognitive effort and processing power allocation [69–71], and hypocretin/orexin neurons in the hypothalamus, which govern metabolic resource budgets [72]. Both systems project strongly to thalamic nuclei and could influence routing decisions between simple and complex processing pathways. This suggests our model's routing-cost mechanism may reflect how the brain balances computational demands against metabolic constraints through neuromodulatory control of thalamocortical interactions. While mapping our router onto a specific brain region might feel natural, it should be added that mechanisms like predictive coding can implement routing and filtering operation between regions without the need of an explicit router region [73].

In the context of machine learning, our work builds on the recent widespread adoption of the Mixture-of-Experts architecture for building parameter-efficient large language models [15, 74]. Recent innovations specifically aim at allowing MoE models to process queries dynamically to reduce processing costs [17]. Our small-scale simulations show how it may be possible to use heterogeneous experts alongside a processing-cost loss function that allows the model to dynamically allocate resources to processing tokens. Finally, load balancing in MoEs prevents over-reliance on a single expert by encouraging distributed processing [15]. Our complexity loss serves as a task-driven form of load balancing.

6.1 Limitations and Extensions

There are several ways to expand our investigations. **On the neuroscience side**, our architecture introduces a new way of modeling multi-region interactions of the brain, but some key architectural characteristics are not yet taken into account. Most importantly, the primate brain has large loop structures which would allow signals to return to a region [27]. Our architecture only allows a forward progression of signal and does not allow signals to be routed back to earlier layers. At the same time, while our analyses demonstrated a link between experts and cortical and subcortical regions, we have not linked the router component of our HMoE layers to a specific component of the brain. Potential options are discussed earlier in Section 6, we do not make any explicit comparison to brain data yet. **On the ML side**, our training setup currently focuses on solving relatively simple tasks with a small model. To see whether our complexity-aware routing and load-balancing measures scale to larger networks, we would need to train larger models on more difficult tasks. Lastly, **on the identification of pathways**, we currently rely on three independent tests to see whether a model contains pathways. Future investigations would ideally identify one specific metric to quantify the degree to which a Mixture-of-Experts architecture has formed pathways.

7 Conclusion

In this paper, we introduced a modified Heterogeneous Mixture-of-Experts architecture that results in the formation of recognizable processing pathways. Analysis of these pathways during learning and problem solving revealed a similarity between these pathways and those observed in the brain. Our new *Mixture-of-Pathways* architecture serves as a new theoretical tool for neuroscience and can guide the search for future resource-efficient architectures in machine learning.

Acknowledgments and Disclosure of Funding

We thank our reviewers, the *Neural & Machine Learning Group* at the University of Oxford, and the *AI HW SW CoDesign Workstream* at the Open Compute Project (OCP) for helpful feedback. This research is supported by the EPSRC (EP/X029336/1) and an ERC-UKRA Frontier Research Guarantee Starting Grant (EP/Y027841/1) awarded to R.P.C. J.C.'s work was supported by a Rhodes Scholarship. J.A.'s work was partially supported by a Career Development Research Fellowship from St John's College, Oxford. We additionally thank Modal for compute credits.

References

[1] Zizhen Yao, Cindy TJ van Velthoven, Michael Kunst, Meng Zhang, Delissa McMillen, Changkyu Lee, Won Jung, Jeff Goldy, Aliya Abdelhak, Matthew Aitken, et al. A high-resolution transcriptomic and spatial atlas of cell types in the whole mouse brain. *Nature*, 624(7991):317–332, 2023.

[2] Matthew F Glasser, Timothy S Coalson, Emma C Robinson, Carl D Hacker, John Harwell, Essa Yacoub, Kamil Ugurbil, Jesper Andersson, Christian F Beckmann, Mark Jenkinson, et al. A multi-modal parcellation of human cerebral cortex. *Nature*, 536(7615):171–178, 2016.

[3] Stewart Shipp. Structure and function of the cerebral cortex. *Current Biology*, 17(12):R443–R449, 2007.

[4] Valentina Mione, Jascha Achterberg, Makoto Kusunoki, Mark J Buckley, and John Duncan. Neural dynamics of an extended frontal lobe network in goal-subgoal problem solving. *bioRxiv*, pages 2025–05, 2025.

[5] Maxwell A Bertolero, BT Thomas Yeo, and Mark D'Esposito. The modular and integrative functional architecture of the human brain. *Proceedings of the National Academy of Sciences*, 112(49):E6798–E6807, 2015.

[6] Kalanit Grill-Spector and Rafael Malach. The human visual cortex. *Annual Review of Neuroscience*, 27:649–677, 2004.

[7] Edward HF de Haan and Alan Cowey. On the usefulness of 'what'and 'where'pathways in vision. *Trends in cognitive sciences*, 15(10):460–466, 2011.

[8] Stephen R Arnott, Malcolm A Binns, Cheryl L Grady, and Claude Alain. Assessing the auditory dual-pathway model in humans. *Neuroimage*, 22(1):401–408, 2004.

[9] John Duncan. Construction and use of mental models: Organizing principles for the science of brain and mind. *Neuropsychologia*, 207:109062, 2025.

[10] Amit Etkin, Tobias Egner, and Raffael Kalisch. Emotional processing in anterior cingulate and medial prefrontal cortex. *Trends in Cognitive Sciences*, 15(2):85–93, 2011.

[11] Michal Bernstein and Galit Yovel. Two neural pathways of face processing: A critical evaluation of current models. *Neuroscience & Biobehavioral Reviews*, 55:536–546, 2015.

[12] 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, 2019.

[13] Jascha Achterberg, Danyal Akarca, D. J. Strouse, John Duncan, and Duncan E. Astle. Spatially embedded recurrent neural networks reveal widespread links between structural and functional neuroscience findings. *Nature Machine Intelligence*, 5(12):1369–1381, 2023. Publisher: Nature Publishing Group.

[14] Olaf Sporns and Richard F Betzel. Modular brain networks. *Annual review of psychology*, 67(1):613–640, 2016.

[15] Aixin Liu, Bei Feng, Bin Wang, Bingxuan Wang, Bo Liu, Chenggang Zhao, Chengqi Dengr, Chong Ruan, Damai Dai, Daya Guo, et al. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model. *arXiv preprint arXiv:2405.04434*, 2024.

[16] Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mixtral of Experts, 2024. arXiv:2401.04088 [cs].

[17] David Raposo, Sam Ritter, Blake Richards, Timothy Lillicrap, Peter Conway Humphreys, and Adam Santoro. Mixture-of-depths: Dynamically allocating compute in transformer-based language models. *arXiv preprint arXiv:2404.02258*, 2024.

[18] William Fedus, Jeff Dean, and Barret Zoph. A review of sparse expert models in deep learning. *arXiv preprint arXiv:2209.01667*, 2022.

[19] Paul Barham, Aakanksha Chowdhery, Jeff Dean, Sanjay Ghemawat, Steven Hand, Daniel Hurt, Michael Isard, Hyeontaek Lim, Ruoming Pang, Sudip Roy, et al. Pathways: Asynchronous distributed dataflow for ml. *Proceedings of Machine Learning and Systems*, 4:430–449, 2022.

[20] Nikolas Gritsch, Qizhen Zhang, Acyr Locatelli, Sara Hooker, and Ahmet Üstün. Nexus: Specialization meets adaptability for efficiently training mixture of experts. *arXiv preprint arXiv:2408.15901*, 2024.

[21] Matthew Lyle Olson, Neale Ratzlaff, Musashi Hinck, Man Luo, Sungduk Yu, Chendi Xue, and Vasudev Lal. Semantic specialization in moe appears with scale: A study of deepseek r1 expert specialization. *arXiv preprint arXiv:2502.10928*, 2025.

[22] An Wang, Xingwu Sun, Ruobing Xie, Shuaipeng Li, Jiaqi Zhu, Zhen Yang, Pinxue Zhao, JN Han, Zhanhui Kang, Di Wang, et al. Hmoe: Heterogeneous mixture of experts for language modeling. *arXiv preprint arXiv:2408.10681*, 2024.

[23] Ganesh Jawahar, Subhabrata Mukherjee, Xiaodong Liu, Young Jin Kim, Muhammad Abdul-Mageed, Laks VS Lakshmanan, Ahmed Hassan Awadallah, Sebastien Bubeck, and Jianfeng Gao. Automoe: Heterogeneous mixture-of-experts with adaptive computation for efficient neural machine translation. *arXiv preprint arXiv:2210.07535*, 2022.

[24] Mikail Khona, Sarthak Chandra, Joy J. Ma, and Ila Fiete. Winning the lottery with neural connectivity constraints: faster learning across cognitive tasks with spatially constrained sparse RNNs, 2023. arXiv:2207.03523 [q-bio].

[25] John Duncan. The multiple-demand (MD) system of the primate brain: mental programs for intelligent behaviour. *Trends in Cognitive Sciences*, 14(4):172–179, 2010.

[26] Guangyu Robert Yang and Manuel Molano-Mazón. Towards the next generation of recurrent network models for cognitive neuroscience. *Current opinion in neurobiology*, 70:182–192, 2021.

[27] Jascha Achterberg, Danyal Akarca, Moataz Assem, Moritz Heimbach, Duncan E Astle, and John Duncan. Building artificial neural circuits for domain-general cognition: a primer on brain-inspired systems-level architecture. *arXiv preprint arXiv:2303.13651*, 2023.

[28] Nicolas Perez-Nieves, Vincent CH Leung, Pier Luigi Dragotti, and Dan FM Goodman. Neural heterogeneity promotes robust learning. *Nature communications*, 12(1):5791, 2021.

[29] Gabriel Béna and Dan FM Goodman. Dynamics of specialization in neural modules under resource constraints. *Nature Communications*, 16(1):187, 2025.

[30] Cornelia Sheeran, Andrew S Ham, Duncan E Astle, Jascha Achterberg, and Danyal Akarca. Spatial embedding promotes a specific form of modularity with low entropy and heterogeneous spectral dynamics. *arXiv preprint arXiv:2409.17693*, 2024.

[31] Jake Patrick Stroud, Michal Wojcik, Kristopher Torp Jensen, Makoto Kusunoki, Mikiko Kadohisa, Mark J Buckley, John Duncan, Mark G Stokes, and Máté Lengyel. Effects of noise and metabolic cost on cortical task representations. *eLife*, 13:RP94961, 2025.

[32] Danyal Akarca, Simona Schiavi, Jascha Achterberg, Sila Genc, Derek K Jones, and Duncan E Astle. A weighted generative model of the human connectome. *bioRxiv*, pages 2023–06, 2023.

[33] Samuel J Gershman, Eric J Horvitz, and Joshua B Tenenbaum. Computational rationality: A converging paradigm for intelligence in brains, minds, and machines. *Science*, 349(6245):273–278, 2015.

[34] Abdullahi Ali, Nasir Ahmad, Elgar de Groot, Marcel Antonius Johannes van Gerven, and Tim Christian Kietzmann. Predictive coding is a consequence of energy efficiency in recurrent neural networks. *Patterns*, 3(12), 2022.

[35] Kevin GC Mizes, Jack Lindsey, G Sean Escola, and Bence P Ölveczky. The role of motor cortex in motor sequence execution depends on demands for flexibility. *Nature Neuroscience*, pages 1–10, 2024.

[36] Yang Zhou, Matthew C Rosen, Sruthi K Swaminathan, Nicolas Y Masse, Ou Zhu, and David J Freedman. Distributed functions of prefrontal and parietal cortices during sequential categorical decisions. *Elife*, 10:e58782, 2021.

[37] Joseph Pemberton, Paul Chadderton, and Rui Ponte Costa. Cerebellar-driven cortical dynamics can enable task acquisition, switching and consolidation. *Nature Communications*, 15(1):10913, 2024.

[38] Ching Fang and Kimberly L Stachenfeld. Predictive auxiliary objectives in deep rl mimic learning in the brain. *arXiv preprint arXiv:2310.06089*, 2023.

[39] Ted Moskovitz, Kevin J Miller, Maneesh Sahani, and Matthew M Botvinick. Understanding dual process cognition via the minimum description length principle. *PLOS Computational Biology*, 20(10):e1012383, 2024.

[40] Samuel Liebana Garcia, Aeron Laffere, Chiara Toschi, Louisa Schilling, Jacek Podlaski, Matthias Fritsche, Peter Zatka-Haas, Yulong Li, Rafal Bogacz, Andrew Saxe, et al. Striatal dopamine reflects individual long-term learning trajectories. *bioRxiv*, pages 2023–12, 2023.

[41] David G Clark and Manuel Beiran. Structure of activity in multiregion recurrent neural networks. *Proceedings of the National Academy of Sciences*, 122(10):e2404039122, 2025.

[42] Ulises Pereira-Obilinovic, Sean Froudist-Walsh, and Xiao-Jing Wang. Cognitive network interactions through communication subspaces in large-scale models of the neocortex. *bioRxiv*, 2024.

[43] Leo Kozachkov, Michaela Ennis, and Jean-Jacques Slotine. Rnns of rnns: Recursive construction of stable assemblies of recurrent neural networks. *Advances in neural information processing systems*, 35:30512–30527, 2022.

[44] Dawn Finzi, Eshed Margalit, Kendrick Kay, Daniel LK Yamins, and Kalanit Grill-Spector. A single computational objective drives specialization of streams in visual cortex. *bioRxiv*, pages 2023–12, 2023.

[45] Xin He, Shunkang Zhang, Yuxin Wang, Haiyan Yin, Zihao Zeng, Shaohuai Shi, Zhenheng Tang, Xiaowen Chu, Ivor Tsang, and Ong Yew Soon. Expertflow: Optimized expert activation and token allocation for efficient mixture-of-experts inference. *arXiv preprint arXiv:2410.17954*, 2024.

[46] Zheyu Yang, Taoyi Wang, Yihan Lin, Yuguo Chen, Hui Zeng, Jing Pei, Jiazheng Wang, Xue Liu, Yichun Zhou, Jianqiang Zhang, et al. A vision chip with complementary pathways for open-world sensing. *Nature*, 629(8014):1027–1033, 2024.

[47] Aaron Defazio, Xingyu Yang, Ahmed Khaled, Konstantin Mishchenko, Harsh Mehta, and Ashok Cutkosky. The road less scheduled. *Advances in Neural Information Processing Systems*, 37:9974–10007, 2024.

[48] Manuel Molano-Mazon, Joao Barbosa, Jordi Pastor-Ciurana, Marta Fradera, Ru-Yuan Zhang, Jeremy Forest, Jorge del Pozo Lerida, Li Ji-An, Christopher J Cueva, Jaime de la Rocha, et al. Neurogym: An open resource for developing and sharing neuroscience tasks. *OSF*, 2022.

[49] Wouter Kool and Matthew Botvinick. Mental labour. *Nature Human Behaviour*, 2(12):899–908, 2018. Publisher: Nature Publishing Group.

[50] Ed Bullmore and Olaf Sporns. The economy of brain network organization. *Nature reviews neuroscience*, 13(5):336–349, 2012.

[51] James B Aimone. A roadmap for reaching the potential of brain-derived computing. *Advanced Intelligent Systems*, 3(1):2000191, 2021.

[52] William Fedus, Barret Zoph, and Noam Shazeer. Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity, 2022. arXiv:2101.03961 [cs].

[53] Samyam Rajbhandari, Conglong Li, Zhewei Yao, Minjia Zhang, Reza Yazdani Aminabadi, Ammar Ahmad Awan, Jeff Rasley, and Yuxiong He. DeepSpeed-MoE: Advancing Mixture-of-Experts Inference and Training to Power Next-Generation AI Scale, 2022. arXiv:2201.05596 [cs].

[54] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014.

[55] Gustavo Deco, Edmund T. Rolls, and Ranulfo Romo. Stochastic dynamics as a principle of brain function. *Progress in Neurobiology*, 88(1):1–16, 2009.

[56] Hailiang Li, Jian Weng, Yijun Mao, Yonghua Wang, Yiju Zhan, Qingling Cai, and Wanrong Gu. Adaptive dropout method based on biological principles. *IEEE Transactions on Neural Networks and Learning Systems*, 32(9):4267–4276, 2021.

[57] John Duncan. Construction and use of mental models: Organizing principles for the science of brain and mind. *Neuropsychologia*, 207:109062, 2025.

[58] Evelina Fedorenko, John Duncan, and Nancy Kanwisher. Broad domain generality in focal regions of frontal and parietal cortex. *Proceedings of the National Academy of Sciences*, 110(41):16616–16621, 2013.

[59] Daniel J Mitchell, Andrew H Bell, Mark J Buckley, Anna S Mitchell, Jerome Sallet, and John Duncan. A putative multiple-demand system in the macaque brain. *Journal of Neuroscience*, 36(33):8574–8585, 2016.

[60] María Roca, Alice Parr, Russell Thompson, Alexandra Woolgar, Teresa Torralva, Nagui Antoun, Facundo Manes, and John Duncan. Executive function and fluid intelligence after frontal lobe lesions. *Brain*, 133(1):234–247, 2010.

[61] V. Goel and J. Grafman. Are the frontal lobes implicated in "planning" functions? Interpreting data from the Tower of Hanoi. *Neuropsychologia*, 33(5):623–642, 1995.

[62] Y Kate Hong, Clay O Lacefield, Chris C Rodgers, and Randy M Bruno. Sensation, movement and learning in the absence of barrel cortex. *Nature*, 561(7724):542–546, 2018.

[63] Andrew J Peters, Julie MJ Fabre, Nicholas A Steinmetz, Kenneth D Harris, and Matteo Carandini. Striatal activity topographically reflects cortical activity. *Nature*, 591(7850):420–425, 2021.

[64] Steffen B. E. Wolff, Raymond Ko, and Bence P. Ölveczky. Distinct roles for motor cortical and thalamic inputs to striatum during motor skill learning and execution. *Science Advances*, 8(8):eabk0231, 2022.

[65] Ray J Dolan and Peter Dayan. Goals and habits in the brain. *Neuron*, 80(2):312–325, 2013.

[66] Risa Kawai, Timothy Markman, Rajesh Poddar, Raymond Ko, Antoniu L Fantana, Ashesh K Dhawale, Adam R Kampff, and Bence P Ölveczky. Motor cortex is required for learning but not for executing a motor skill. *Neuron*, 86(3):800–812, 2015.

[67] Sarthak Chandra, Sugandha Sharma, Rishidev Chaudhuri, and Ila Fiete. Episodic and associative memory from spatial scaffolds in the hippocampus. *Nature*, pages 1–13, 2025.

[68] S. M. Sherman and R. W. Guillery. The role of the thalamus in the flow of information to the cortex. *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences*, 357(1428):1695–1708, 2002. Publisher: Royal Society.

[69] Gary Aston-Jones and Jonathan D Cohen. An integrative theory of locus coeruleus-norepinephrine function: adaptive gain and optimal performance. *Annu. Rev. Neurosci.*, 28(1):403–450, 2005.

[70] Susan J Sara and Sebastien Bouret. Orienting and reorienting: the locus coeruleus mediates cognition through arousal. *Neuron*, 76(1):130–141, 2012.

[71] Andrew Westbrook and Todd S Braver. Cognitive effort: A neuroeconomic approach. *Cognitive, Affective, & Behavioral Neuroscience*, 15(2):395–415, 2015.

[72] Alexander L Tesmer, Christine Dalla Pola, Dino Gilli, Nikola Grujic, Eva F Bracey, Tommaso Patriarchi, Daria Peleg-Raibstein, Rafael Polania, and Denis Burdakov. Neurometabolic signaling and control of policy complexity. *bioRxiv*, pages 2025–02, 2025.

[73] Kaitlyn M Gabhart, Yihan Sophy Xiong, and André M Bastos. Predictive coding: a more cognitive process than we thought? *Trends in Cognitive Sciences*, 2023.

[74] Llama Team. The Llama 4 herd: The beginning of a new era of natively multimodal AI innovation, 2025.

[75] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018.

[76] Stefano Recanatesi, Serena Bradde, Vijay Balasubramanian, Nicholas A. Steinmetz, and Eric Shea-Brown. A scale-dependent measure of system dimensionality. *Patterns*, 3(8):100555, 2022.

[77] Ana C. Lorena, Luís P. F. Garcia, Jens Lehmann, Marcilio C. P. Souto, and Tin K. Ho. How complex is your classification problem? a survey on measuring classification complexity, 2020.

[78] John Duncan, Alice Parr, Alexandra Woolgar, Russell Thompson, Peter Bright, Sally Cox, Sonia Bishop, and Ian Nimmo-Smith. Goal neglect and spearman's g: competing parts of a complex task. *J. Exp. Psychol. Gen.*, 137(1):131–148, February 2008.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes] .

   Justification: The abstract and introduction accurately represent the paper's contributions without overstatement. We claim that (1) heterogeneous experts do not automatically form pathways without specific architectural priors, (2) normalized routing complexity loss and expert dropout enable pathway formation, and (3) the resulting pathways mirror brain-like processing pathways in multiple ways. Our experimental results directly support these claims through quantitative evaluations of pathway stability, self-sufficiency, and task-related functionality. The limitations of our approach are acknowledged in the Discussion section, including the lack of recurrent connectivity and the current small scale of our model.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Our paper includes a dedicated "Limitations and Extensions" section that explicitly discusses the limitations of our work across three key areas: (1) Neuroscience limitations: We acknowledge that our architecture lacks important brain-like features such as recurrent connectivity loops, which are present in the primate brain, and that we have not explicitly linked the router networks to specific brain structures. (2) Machine learning limitations: We discuss that our current implementation focuses on relatively small models and simple tasks, raising questions about whether our approach would scale to larger networks with more complex tasks. (3) Methodology limitations: We note that our current approach for identifying pathways relies on two independent tests, rather than a single unified metric. Throughout the paper, we are also transparent about the computational requirements of our model and the number of training runs conducted. These limitations are presented alongside potential future research directions to address these constraints.

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.

- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

   Justification: Our paper does not include any theoretical proofs. It does make the theoretical assumption that the Mixture-of-Experts architecture is a suitable backbone for testing the development of pathways but this assumption is made very clear given our extensive discussion of the Mixture-of-Experts architecture.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We rely on commonly used and well defined building blocks both on the side of models and tasks that are openly available and clearly state the hyperparameters we rely on. Our code is also available for review as part of the supplemental material.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.

- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We provide detailed description on how to access and run our code in the appendix.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Our Methods section describes all details of the algorithm setup needed, and the appendix provides additional implementation details, specifically with regard to the custom loss functions that we use. Generally we rely on very established building blocks combined in a novel way, and our instructions on how we combined them and which hyperparameters we used are described in the main text.

   Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We provide significance tests for all main claims in the paper. All significance tests we use are two-sided.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We specifically describe the compute resources we use to run our experiments in the methods.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

Justification: Yes, our work fully conforms with the NeurIPS Code of Ethics.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [NA]

Justification: We conduct very small-scale simulations aimed at studying a theoretical phenomenon in MoE models and replicating features observed in the brain. The models trained here are in no way powerful enough to be deployed for the contexts discussed in the guidelines of this question.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: We only use simple simulated trials of cognitive tasks as data which were generated based on open-access packages and hence there is no risk of misuse. We do not release any pretrained models.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.

- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We appropriatly cite the source of the dataset package. All our simulations are implemented with PyTorch, as described in the extended implementation details in the appendix.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We provide anonymized simulation code for our paper which is released under a CC BY 4.0 license.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: We do not use any crowdsourcing or human subjects.

    Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: We do not use any crowdsourcing or human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (https://neurips.cc/Conferences/2025/LLM) for what should or should not be described.

A Appendix

A.1 Implementation Details and Code Access

While most details on the model implementation are described in Section 3, we provide some additional details here. The GRUs in all of our layers, including routers, use the ReLU activation function and are initialized from $\mathcal{U}(-\sqrt{k}, \sqrt{k})$, where k is the GRU's hidden size, as is standard in PyTorch. All models are optimized with the Schedule-Free variant of the AdamW optimizer [47] using a learning rate of 0.01, betas of $(0.9, 0.999)$, and no weight decay. As briefly described in Section 3, we use an additional embedding layer to transform the 82-dimensional one-hot encoding of the task identity into a 16-dimensional embedding vector, which is concatenated with the task stimuli before being provided to the model. We include these details in a complete version of our training algorithm below in Algorithm 2, expanding on the abbreviated algorithm introduced in Algorithm 1. We provide our implementation at `https://github.com/jackcook/mixture-of-pathways`.

Algorithm 2: Full Mixture-of-Pathways training protocol.

Data: Task dataset D, Experts $E = \{e_1, e_2, \ldots, e_n\}$, Routers $R = \{r_1, r_2, \ldots, r_n\}$
Initialize routers and heterogeneous experts, sampling weights and biases from $\mathcal{U}(-\sqrt{k}, \sqrt{k})$;
Process 82-dimensional one-hot task encoding with embedding layer;
Set h_0 to task input, consisting of a 1-dimensional fixation input, two 16-dimensional stimuli, and a 16-dimensional task embedding at each timestep;
for *each training step* **do**
  Sample batch b from D;
  for *each layer* l **do**
    Compute routing weights $w_{l,j} = \text{softmax}(r_l(h_{l-1}))$;
    Apply expert dropout (Not in baseline architecture; see Section 4.2) ;
    Compute expert activations $w_{l,j}$ for each expert in l ;
    Combine outputs: $h_l = \sum_j w_{l,j} \cdot h_{l,j}$;
  end
  Compute baseline model losses: L_{fix} and $L_{\text{response},i}$;
  Compute L_{routing} loss (Not in baseline architecture; see Section 4.1) ;
  Compute $L_{\text{total}} = L_{\text{fix}} + L_{\text{routing}} + \sum_i L_{\text{response},i}$;
  Update parameters using Schedule-Free AdamW optimizer [47];
end

A.2 Sample Task Visualizations and Descriptions

Section 3 briefly described the Mod-Cog task set [24]. Here, we provide a more detailed description of the tasks, alongside visualizations of sample trials.

The Mod-Cog task set consists of 82 time-series-based cognitive tasks that extend the original NeuroGym framework [48] through two primary modifications: integration tasks, which incorporate interval estimation based on delay periods, and sequence generation tasks, which require time-varying outputs with drifting directions. The original 20 cognitive tasks from NeuroGym serve as the foundation, with new integration-based tasks and new sequence generation tasks forming a set of 82 tasks in total. These tasks span a wide range of cognitive demands, from simple stimulus-response mappings to complex working memory and sequential decision-making challenges.

Tasks are presented as continuous time-series data, which we sample at 100-millisecond intervals. Each input consists of a 1-dimensional fixation signal, two 16-dimensional stimulus channels, and an 82-dimensional one-hot task identifier that passes through a learned embedding layer, as described in Appendix A.1. During the fixation period of each task, models must maintain an output of zero while processing incoming stimuli. During the subsequent response period, models must return task-specific output sequences. Task difficulty varies systematically according to several factors: the complexity of decision rules (from simple detection to multi-step integration), the duration of delay periods that tax working memory, the number of stimuli that must be simultaneously tracked, and whether responses require static outputs or dynamic sequential patterns. Figure 7 illustrates six representative tasks that demonstrate this range of complexity.

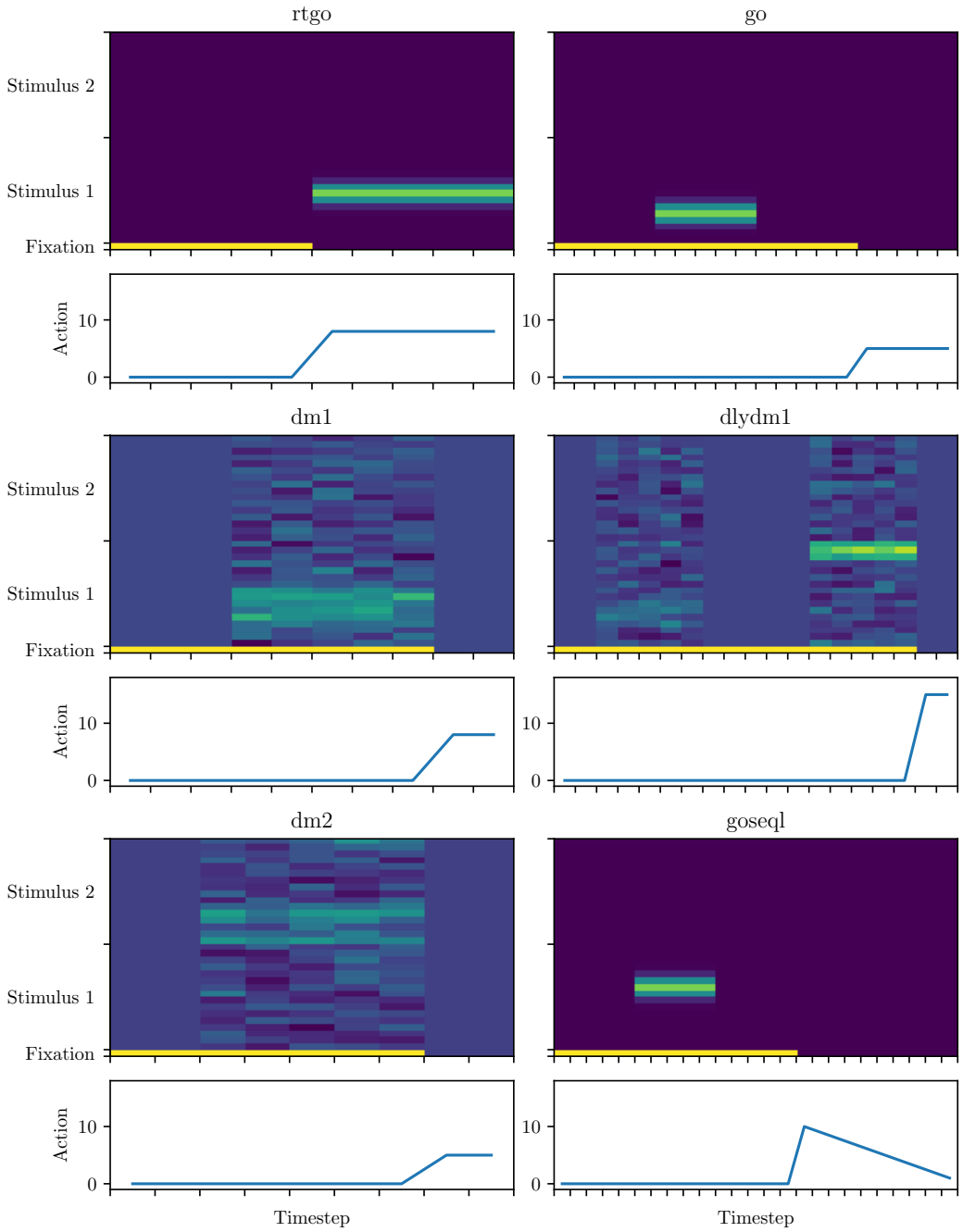


Figure 7: Model inputs and actions for the `rtgo`, `go`, `dm1`, `dlydm1`, `dm2`, and `goseqr` tasks. Appendix A.2 provides a detailed description of each task.

- **RTGo task** (`rtgo`): The model must immediately output the value presented in either input channel.
- **Go task** (`go`): The model observes two input channels and must respond with the value presented in either stimulus once the fixation period has ended.
- **Decision Making task** (`dm1`): The model observes brief stimulus presentations from both channels and must respond with the value in stimulus 1 that had the highest average intensity (values in stimulus 2 can be ignored). This requires integration and comparison of sensory evidence.
- **Delayed Decision Making task** (`dlydm1`): Similar to dm1, but includes a delay period between a decoy stimulus and the noisy stimulus, requiring working memory to maintain stimulus information.
- **Dual Decision Making task** (`dm2`): Similar to dm1, but the model must return the highest value from stimulus 2.
- **Sequential Decision Making task** (`goseql`): Based on the Go task but requiring a time-varying output that drifts in a specific direction (leftward) over the response period, combining stimulus detection with sequential motor control.

A.3 Loss Variants Used Across Model Types

In Section 4, we introduce architectural changes to the model that encourage pathways to form between layers. Here, we give an overview of the five different loss functions which are used to train our different model variants.

The loss function that we use to train the baseline model only includes the fixation and task response loss, as shown below in Equation 4. Equation 5 computes a mean-squared error for the fixation period where the correct output value is always zero. This loss function is not task-specific since it can be computed across all task inputs while the fixation input is active. Equation 6 computes a task-specific cross-entropy loss between the 16 possible output values and the model's 16 output logits at each timestep. When used to train a model, one of these response losses will be computed for each task in the set of tasks $\mathcal{T}$.

$$L_{\text{Baseline}} = L_{\text{fix}} + \sum_{i}^{\mathcal{T}} L_{\text{response},i} \tag{4}$$

$$L_{\text{fix}} = \frac{1}{T} \sum_{t}^{T} \hat{y}_t^2 \tag{5}$$

$$L_{\text{response},i} = -\frac{1}{T_i} \sum_{t}^{T_i} y_t \log(\hat{y}_t) \tag{6}$$

Here, $\hat{y}_t$ represents the model's output logits at timestep t during fixation, T is the total number of fixation timesteps, y_t is the true target output, $\hat{y}_t$ is the model's predicted output at timestep t, and T_i represents the total number of response timesteps for task i.

In Section 4.1, using this baseline loss function, we find that the baseline model by itself does not converge on consistent pathways across model runs. This motivates us to create a new loss function, L_{RC}, which introduces a routing cost that penalizes the model for using more complex experts, as shown in Equation 7.

$$L_{\text{RC}} = L_{\text{fix}} + \sum_{i}^{\mathcal{T}} (L_{\text{response},i} + \alpha LPC_i) \tag{7}$$

$$LPC_i = \frac{1}{T_i} \sum_{t}^{T_i} \sum_{j}^{E} w_{i,j,t} s_j^2 \tag{8}$$

In Equation 8, $w_{i,j,t}$ represents the routing weight assigned to expert j at timestep t for task i, s_j is the size of expert j, and E denotes the total number of experts in the model (in this manuscript, all models have three layers of three experts each, so $E = 9$). α is a hyperparameter that balances the trade-off between the model's performance on each task and the complexity of the experts used to solve that task. If α is too large, the model will reach a local minimum where it is unable to solve any task, but it can reduce its expert usage to zero by using the skip connections in each layer and performing no computation. On the other hand, if α is too small, the model will solve each task to a very high degree of accuracy, but not reduce the complexity of the experts used to solve each task, and fail to form brain-like pathways. We found that setting $\alpha = 10^{-5}$ balanced these priorities well, and used this value for all of the experiments in this work. However, future work may investigate a better method for selecting this hyperparameter.

While the routing consistency for the model with the loss in Equation 7 is improved as shown in Figure 2, we do find that it does not converge on a fully consistent routing pattern. This motivates the addition of a scaling factor based on task performance, which reduces the effect of the routing loss when task performance is low. The resulting loss shown in Equation 9 is the final loss we use to train our *Mixture-of-Pathways* model.

$$L = L_{\text{fix}} + \sum_{i}^{\mathcal{T}} (L_{\text{response},i} + \frac{\alpha LPC_i}{L_{\text{response},i} + \epsilon}) \tag{9}$$

The normalization term $L_{\text{response},i} + \epsilon$ uses the task-specific response loss $L_{\text{response},i}$ to scale the routing penalty, where ϵ is a small constant added to prevent division by zero when the model achieves perfect task performance.

A.4 Learned Pathway Complexity and Routing Consistency

A.4.1 Calculation Example

Here we provide a detailed example calculation of how the expert size penalty is calculated. This penalty is used within the calculation of the routing consistency described in Section 4.1 and also as part of the expert-usage loss from Equation 7.

In this equation, E is the set of all experts, s_j is the size of each expert, and $w_{i,j}$ is the weight assigned to expert j while solving task i. For example, imagine a model with three experts: a skip connection, a simple expert with 16 neurons, and a complex expert with 32 neurons. To solve a task i, imagine the model sets the weight of the skip connection to 31%, the simple expert to 43%, and the complex expert to 26%. The model's learned pathway complexity (LPC) for task i would be 376.3, as follows:

$$LPC_i = w_{i,0}s_0^2 + w_{i,1}s_1^2 + w_{i,2}s_2^2 = (0.31)(0)^2 + (0.43)(16)^2 + (0.26)(32)^2 = \boxed{376.3} \tag{10}$$

For simplicity, this equation shows how to calculate the model's LPC at a single timestep. To calculate the LPC for an entire task, this value should be calculated at and averaged over all of the task's timesteps.

A.4.2 Routing Consistency Across Model Types

Using the calculated LPC values for each task, we determine the consistency of the routing decisions made by different models when solving tasks. In Figure 8, we show a version of Figure 2 with our final model, which includes expert dropout. After the addition of expert dropout, the mean pairwise correlations of the models' routing consistency is 0.51 ($p < 0.0001$). This is a reduction when compared to our model trained only with our custom routing cost and task performance scaling, however, as explained in Section 4.2, the model with the additional dropout does develop self-sufficient pathways on top of the consistency criteria, so that the model including the dropout overall better matches the pathway formation criteria. Our main investigation on what we call the *Mixture-of-Pathways* model includes the expert dropout.

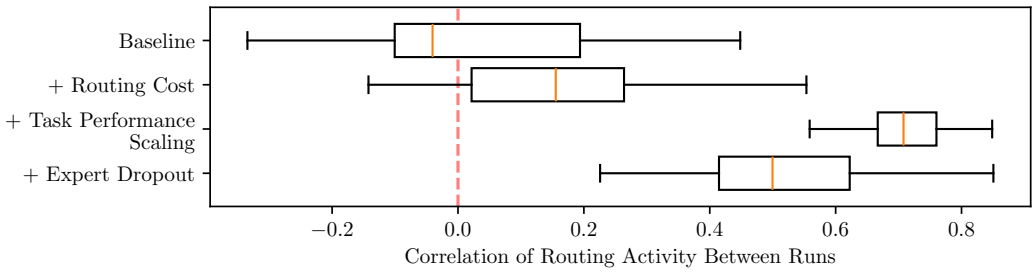


Figure 8: Routing consistency as measured by correlation of routing activity across model runs. This is a version of Figure 2 which also shows the model with expert dropout. Even with expert dropout the routing consistency is significantly above the baseline model ($p < 0.0001$), even though it is slightly reduced when compared to the model with routing cost and task performance scaling.

A.5 Effective Rank

To define the LPC, in Equation 1 we use s_j^2 as a penalty for using expert j, where s_j is the expert's hidden dimension, or zero in the case of a skip connection. Intuitively, this was motivated by the $O(s_j^2)$ cost of storing each expert's weight matrix in memory, however this only roughly captures the learning capabilities of a GRU with s_j neurons. For example, as demonstrated by the lottery ticket hypothesis, it is possible that experts with large hidden dimensions may converge on low-rank solutions more easily than experts with small hidden dimensions [75].

To ensure that we were appropriately penalizing experts relative to each other, we analyzed the effective rank of each expert's matrix, defined as the participation ratio of the squared sum of singular values to the sum of squared singular values, $\frac{(\sum \sigma_i)^2}{\sum \sigma_i^2}$, over the course of training [76]. This metric measures how evenly distributed the singular values are and thus how many dimensions the matrix effectively uses. These are shown in Figure 9. For both models, we find that the effective rank of large experts is roughly double that of small experts, both before and after training, so that large experts always have a much larger effective rank than small experts ($p < 0.0001$). We additionally find that all experts across both our model and the baseline model decrease slightly in effective rank over training, but the differences between these models' effective ranks tends to be small and never becomes significant ($p > 0.05$). This supports the conclusion that the hidden dimension is at least a good approximation of processing complexity, but we encourage future work to consider measuring this with a scalar value.

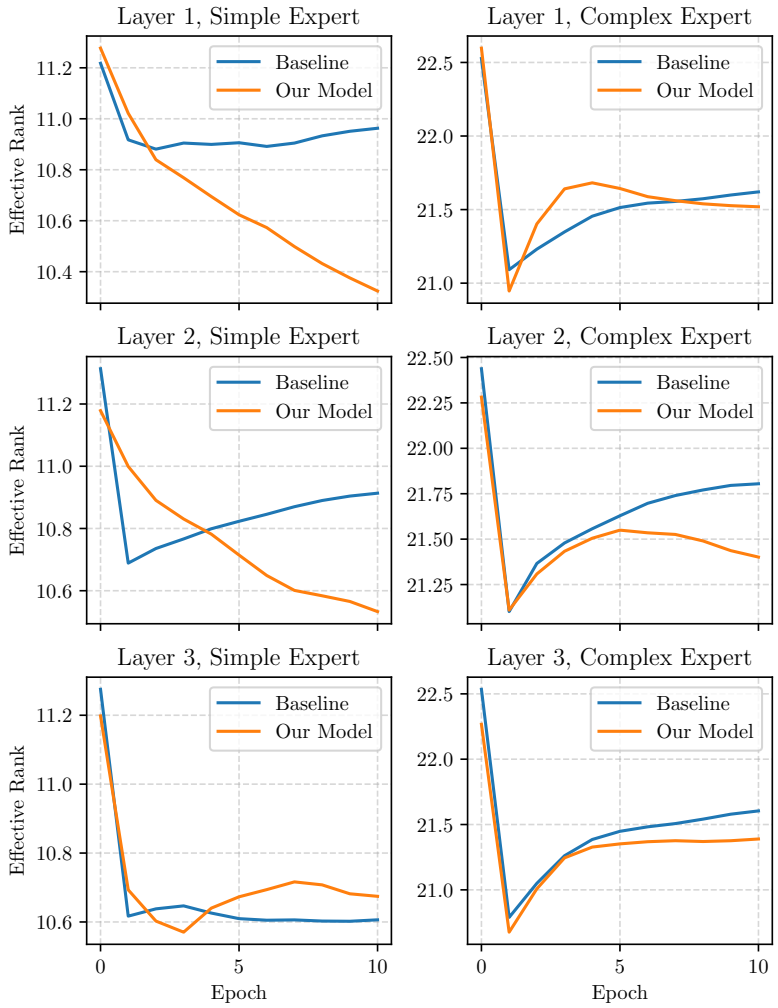


Figure 9: Changes in the effective rank of each expert's weight matrix over the course of training. Results for each configuration are averaged across 20 runs.

A.6 Per-Task Accuracy Metrics

In Table 2 below, we report accuracy metrics on each task for three models: the baseline HMoE model described in Section 3.1, our model, which includes the routing cost and task-performance scaling described in Section 4.1 and the expert dropout described in Section 4.2, and our model without expert dropout.

Table 2: Per-task accuracy metrics.

Task	**Baseline**	**Our Model**	**Without Dropout**
Mean	91.1%	83.0%	90.1%
Median	93.3%	87.5%	91.1%
`anti`	100.0%	100.0%	99.9%
`antiseql`	100.0%	99.8%	99.8%
`antiseqr`	100.0%	100.0%	100.0%
`ctxdlydm1`	99.5%	99.6%	99.5%
`ctxdlydm1intl`	99.8%	100.0%	99.3%
`ctxdlydm1intr`	99.6%	98.4%	99.6%

Task (cont.)	Baseline	Our Model	Without Dropout
ctxdlydm1seql	83.5%	79.9%	84.0%
ctxdlydm1seqr	83.2%	84.1%	82.3%
ctxdlydm2	75.5%	71.7%	76.3%
ctxdlydm2intl	78.2%	70.9%	79.3%
ctxdlydm2intr	81.2%	82.3%	84.2%
ctxdlydm2seql	95.4%	90.2%	94.3%
ctxdlydm2seqr	90.7%	92.7%	90.6%
ctxdm1	93.0%	90.0%	93.7%
ctxdm1seql	93.6%	88.6%	91.0%
ctxdm1seqr	82.2%	83.0%	76.4%
ctxdm2	98.7%	97.9%	99.3%
ctxdm2seql	99.5%	97.5%	98.5%
ctxdm2seqr	99.4%	92.1%	99.0%
dlyanti	99.7%	98.4%	99.5%
dlyantiintl	99.8%	39.1%	98.7%
dlyantiintr	99.6%	44.3%	94.1%
dlyantiseql	96.2%	38.4%	98.4%
dlyantiseqr	98.7%	37.6%	93.4%
dlydm1	91.8%	87.5%	88.6%
dlydm1intl	93.0%	88.5%	91.8%
dlydm1intr	91.7%	89.2%	88.7%
dlydm1seql	92.8%	86.0%	89.3%
dlydm1seqr	90.5%	90.6%	88.8%
dlydm2	90.2%	87.0%	90.5%
dlydm2intl	92.2%	84.1%	86.8%
dlydm2intr	92.1%	85.1%	88.1%
dlydm2seql	82.4%	78.0%	74.3%
dlydm2seqr	82.1%	78.9%	76.8%
dlygo	80.2%	59.9%	76.0%
dlygointl	78.5%	60.8%	79.6%
dlygointr	82.6%	61.5%	84.3%
dlygoseql	85.5%	60.4%	82.0%
dlygoseqr	74.3%	62.6%	76.8%
dm1	76.3%	61.7%	75.0%
dm1seql	78.8%	57.3%	78.4%
dm1seqr	81.0%	63.4%	76.6%
dm2	100.0%	98.0%	99.9%
dm2seql	99.7%	97.6%	99.2%
dm2seqr	100.0%	98.0%	99.9%
dmc	99.9%	95.5%	99.8%
dmcintl	99.8%	99.0%	99.4%
dmcintr	99.5%	98.0%	99.8%
dmcseql	81.9%	75.2%	81.1%
dmcseqr	81.5%	75.0%	81.7%
dms	75.2%	66.5%	77.2%
dmsintl	73.4%	64.8%	77.3%
dmsintr	79.7%	72.8%	86.0%
dmsseql	95.1%	87.6%	92.8%
dmsseqr	95.2%	88.6%	90.6%
dnmc	93.2%	87.3%	88.7%
dnmcintl	93.1%	86.2%	89.7%
dnmcintr	80.8%	79.0%	77.1%
dnmcseql	98.5%	93.0%	98.9%
dnmcseqr	99.7%	94.3%	98.3%
dnms	98.8%	91.3%	98.8%
dnmsintl	99.3%	93.6%	98.1%
dnmsintr	99.9%	99.4%	100.0%

Task (cont.)	Baseline	Our Model	Without Dropout
dnmsseql	100.0%	97.4%	99.6%
dnmsseqr	100.0%	99.1%	99.4%
go	100.0%	94.9%	99.7%
goseql	99.9%	96.7%	99.4%
goseqr	100.0%	94.0%	99.7%
multidlydm	81.8%	74.5%	81.9%
multidlydmintl	78.1%	77.3%	80.0%
multidlydmintr	74.1%	66.7%	76.0%
multidlydmseql	73.1%	66.6%	71.8%
multidlydmseqr	79.7%	71.6%	79.4%
multidm	94.0%	90.0%	90.7%
multidmseql	94.3%	88.9%	93.1%
multidmseqr	93.3%	85.9%	92.6%
rtanti	92.6%	86.1%	91.2%
rtantiseql	82.2%	79.5%	81.6%
rtantiseqr	99.7%	95.2%	98.9%
rtgo	98.7%	94.9%	98.9%
rtgoseql	99.3%	91.9%	98.0%
rtgoseqr	98.1%	93.1%	99.0%

A.7 Task-Specific Expert Usage Patterns

In Figures 10 and 11, we show two samples of tasks completed by our model. At each timestep, we plot the model's decisions in orange, which overlap with the blue ground truth values, indicating that in these trials, the model always returned the correct answer. We additionally plot the expert usage of the model at each timestep. At each timestep, three bars are shown for each layer, with their color indicating the usage of the skip connection, the simple expert, and the complex expert, in that order. In the go trial shown in Figure 10, the model primarily uses the skip connections at each timestep during the fixation period, in which it only needs to return zero. During the response period, the model moves its computations toward a more complex pathway, primarily using the simple experts in layers 1 and 2, and the skip connection in layer 3. In the more complicated dmcintr trial shown in Figure 11, the model switches between pathways multiple times depending on its needs, which vary between working memory and computation. This shows that our model has formed distinct pathway and modes of processing information which it dynamically switches between. As a result, the model opens up the possibility of analyzing the detailed dynamics of how pathways are combined over tasks with time courses that include varying computational demands, to learn which principles underlie such coordination processes.

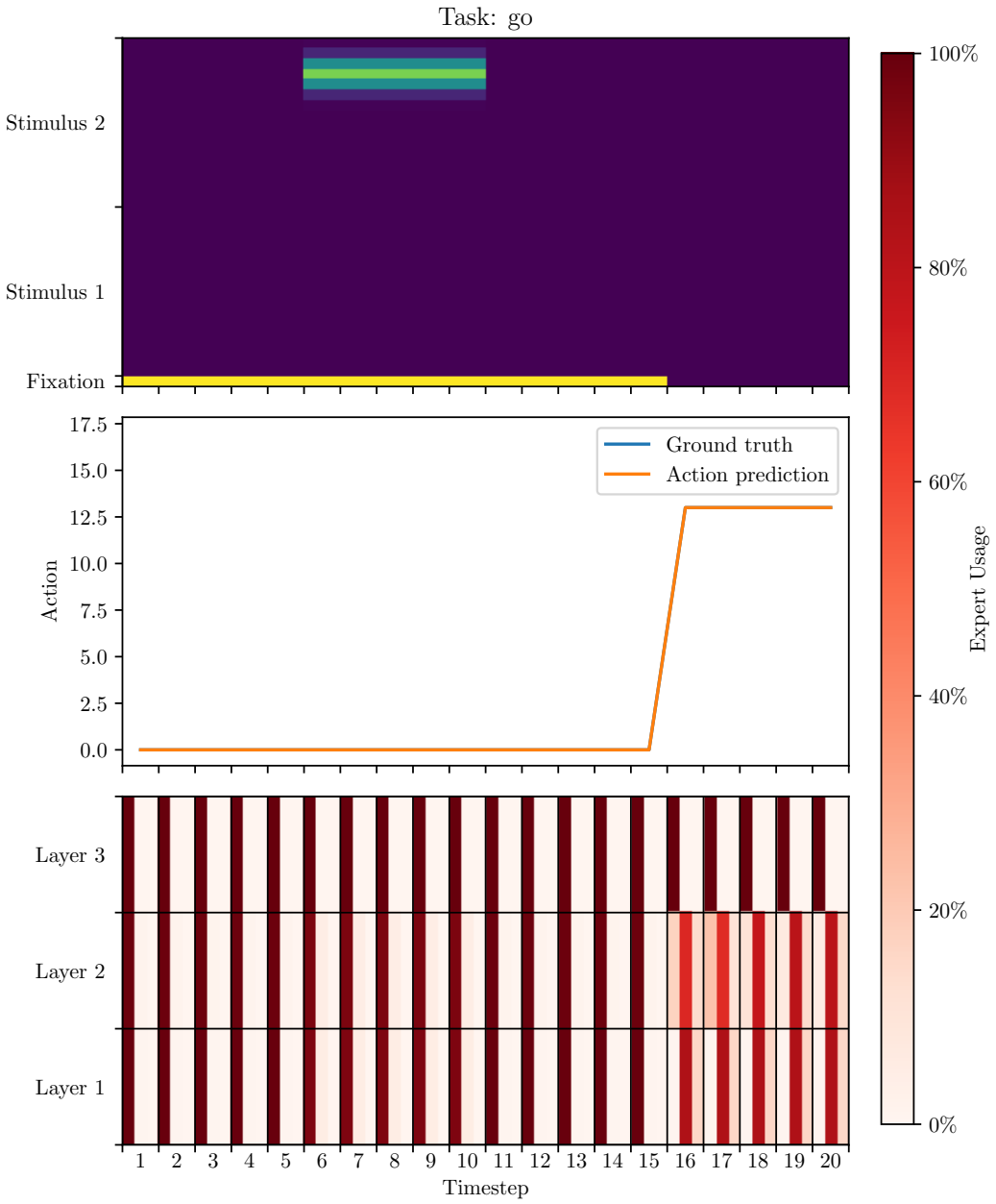


Figure 10: In the go task we see that models rely on extremely simple pathways for their decision making until activating model complex pathways during the response period.

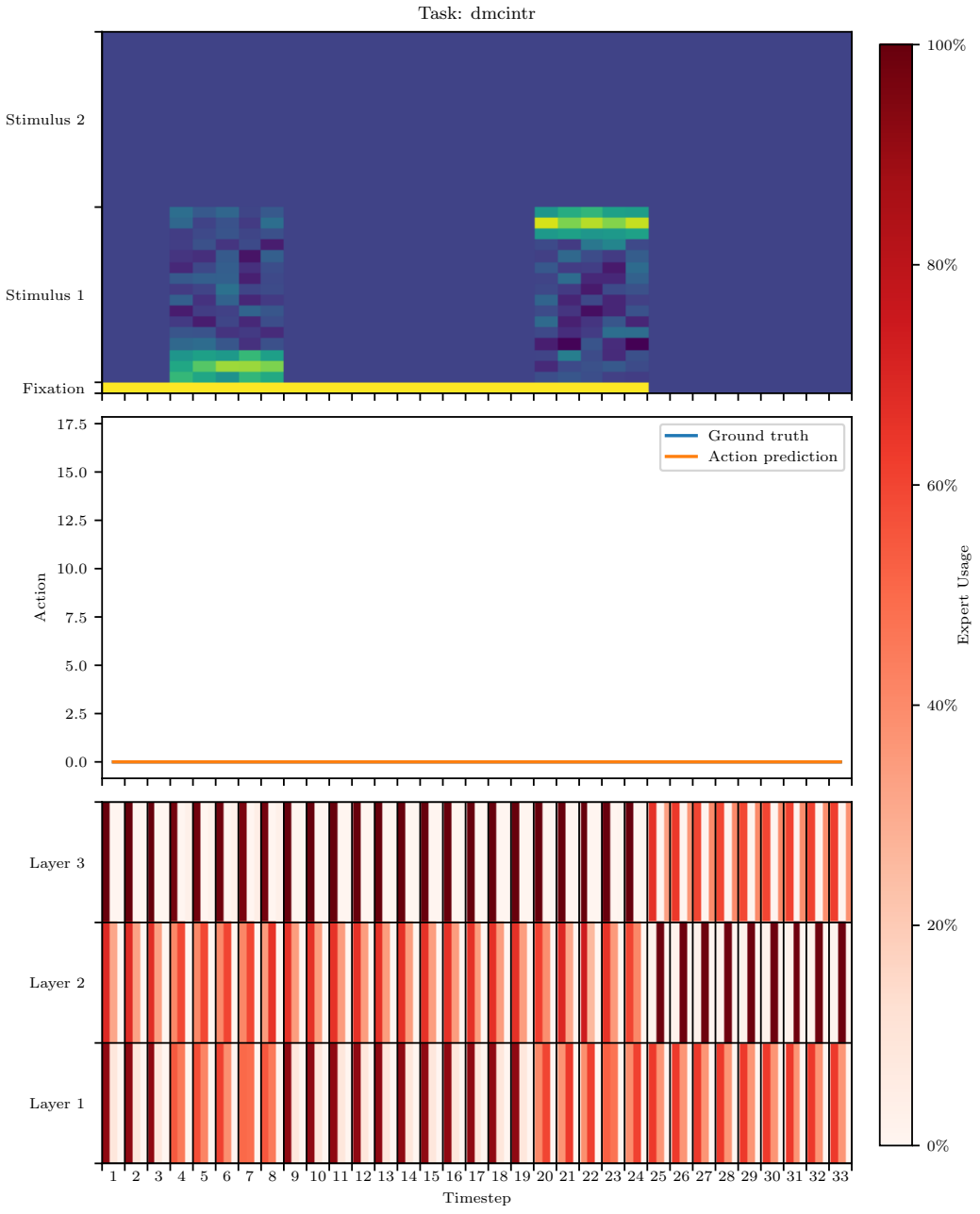


Figure 11: During more complex tasks such as dmcintr we see that models us a a very dynamic and rich set of pathways throughout the duration of a trial to return the correct response.

A.8 Unclustered Expert Usage Patterns Across Tasks

In Figures 13 and 14, we show the unclustered expert usages during three phases of each task. Each task has a different number of timesteps, so in order to condense expert usage into a single figure, we averaged expert usage over three phases which are shared by each task: a pre-stimulus phase, during which the task identity is known but input data has not yet been presented, a stimulus phase, during which the model is observing input data, and a response phase, during which the model needs to output its responses. In these figures, tasks are sorted based on the same clusters shown in Figure 4 for visual clarity. Notably, the complex expert is rarely used in the first two phases, during which the model outputs zero, but is commonly used during the response phase of complex tasks. When analyzing the average usage of the most complex experts over the clusters shown in Figure 4, we find that the cluster with the highest reliance of the most complex experts uses those with an average routing weight of 0.11. In Figure 12 we show the distribution of complex expert usage by layers of the model, over clusters derived in Figure 4, meaning each data points here is one of the 10 clusters.

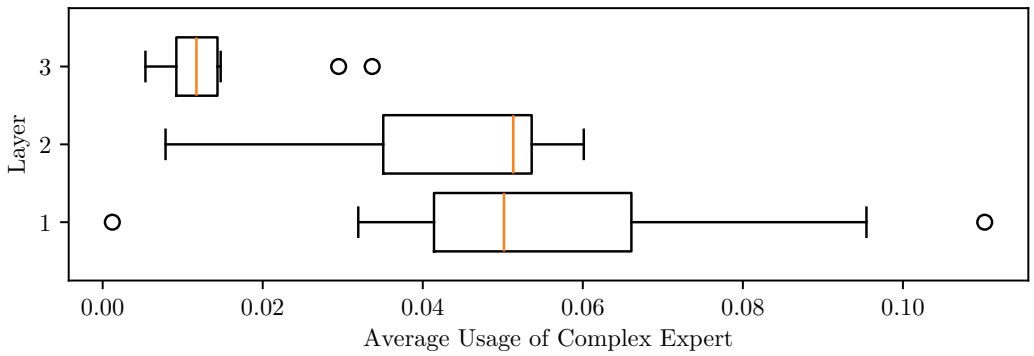


Figure 12: Average usage of the most complex experts for each cluster derived in Figure 12, split by layer index.

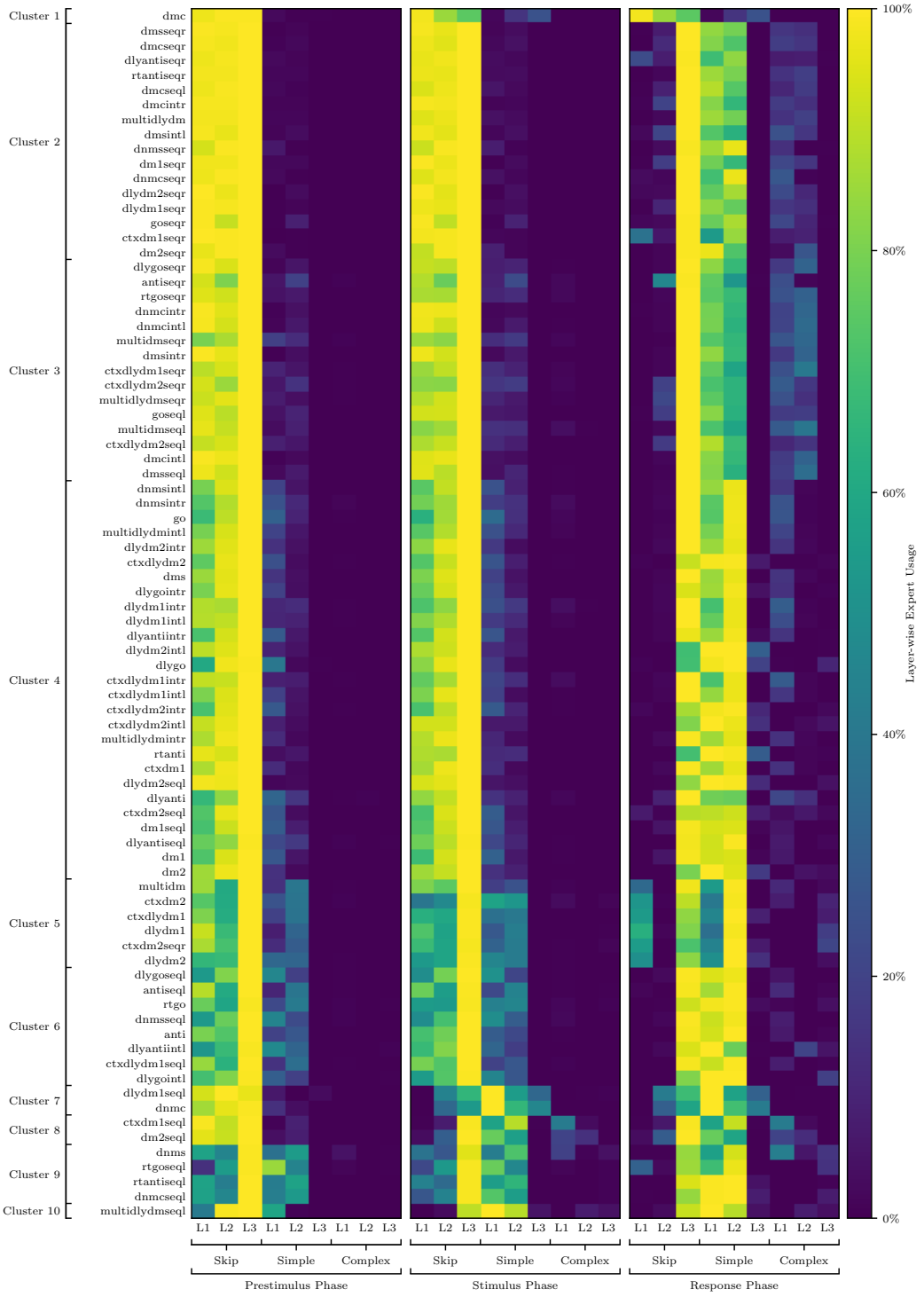


Figure 13: Layer-wise expert usage averaged over three phases of each task: the pre-stimulus phase, during which the task is known but no input data has been given to the model, the stimulus phase, during which input data is being given to the model, and the response phase, during which the model needs to calculate and return the correct response. In each phase, expert usages sum to 100% within each layer, i.e. usages of the skip connection, simple expert, and complex expert of layer 1, denoted by "L1", add up to 100%. Tasks are grouped into 10 clusters, shown along the left, based on similarities in their routing patterns. A contrasting version of this figure for a baseline model trained without our cost-based loss and dropout is shown in Figure 14.

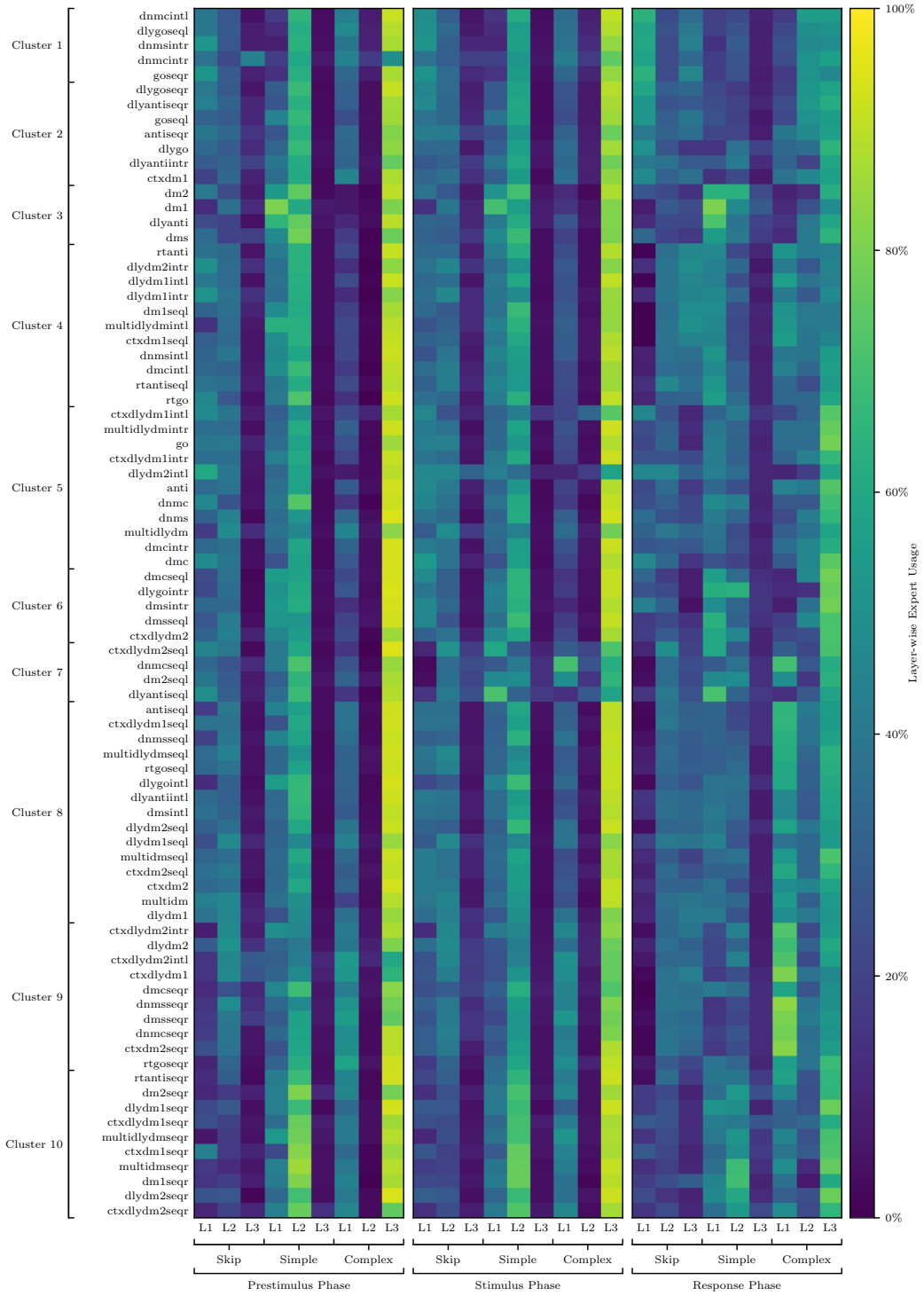


Figure 14: A separate version of Figure 13 for a model trained without our custom routing cost, loss normalization, and expert dropout. The pathways that form are much less distinct, and are also much less stable.

A.9 Task Difficulty

A.9.1 Number of Training Steps

To measure each task's difficulty, we train five recurrent neural networks (GRUs), each with 64 neurons, and record how many training steps it takes each model to solve that task to 99% accuracy. The task's difficulty is then reported as the median number of steps from these five training runs. Figure 15 shows these measurements for all 82 tasks in the Mod-Cog task suite [24].

A.9.2 Number of Rules

There are many ways to characterize the difficulty of learning problems [77], but no universal complexity measure has been developed to date. We believe "training steps needed to learn the task," as discussed above in Appendix A.9.1, is a sensible measure because this can be measured without any implicit biases, and takes into account task demands such as working memory and any other factors which make inference challenging [77]. At the same time, it remains unclear whether such a complexity measure would neatly map onto what humans or animals perceive to be "difficult tasks," which is often linked to the number of rules in a task [78]. However, this can be easily tested, as Mod-Cog tasks are created based on combinations of different motifs and rules. For example, Figure 7 shows that the "Delayed Decision Making" task (`dlydm1`) is an altered version of the standard "Decision Making" task (`dm1`) with the added "Delay" rule (`dly`).

We find that our difficulty metric is in fact correlated with the number of rules in each task ($r = 0.39$; $p < 0.0001$, shown in Figure 16), and that our model exhibits an even stronger correlation for the brain-like finding in Figure 5 when this is used as the difficulty metric ($r = 0.57$, $p < 0.0001$) compared to the baseline model ($r = -0.09$, $p = 0.4288$). At the same time, using "number of rules" as the actual difficulty metric has two downsides: (a) it is a discrete and ordinal measurement from 1 to 4 with less statistical power, and (b) some rules are harder to learn than others (i.e. `go` vs. `dm1`). This suggests to us that our current convergence-based complexity measure is, at least in this specific task environment, a difficulty metric providing a better link to both the brain and GRU-based machine learning models.

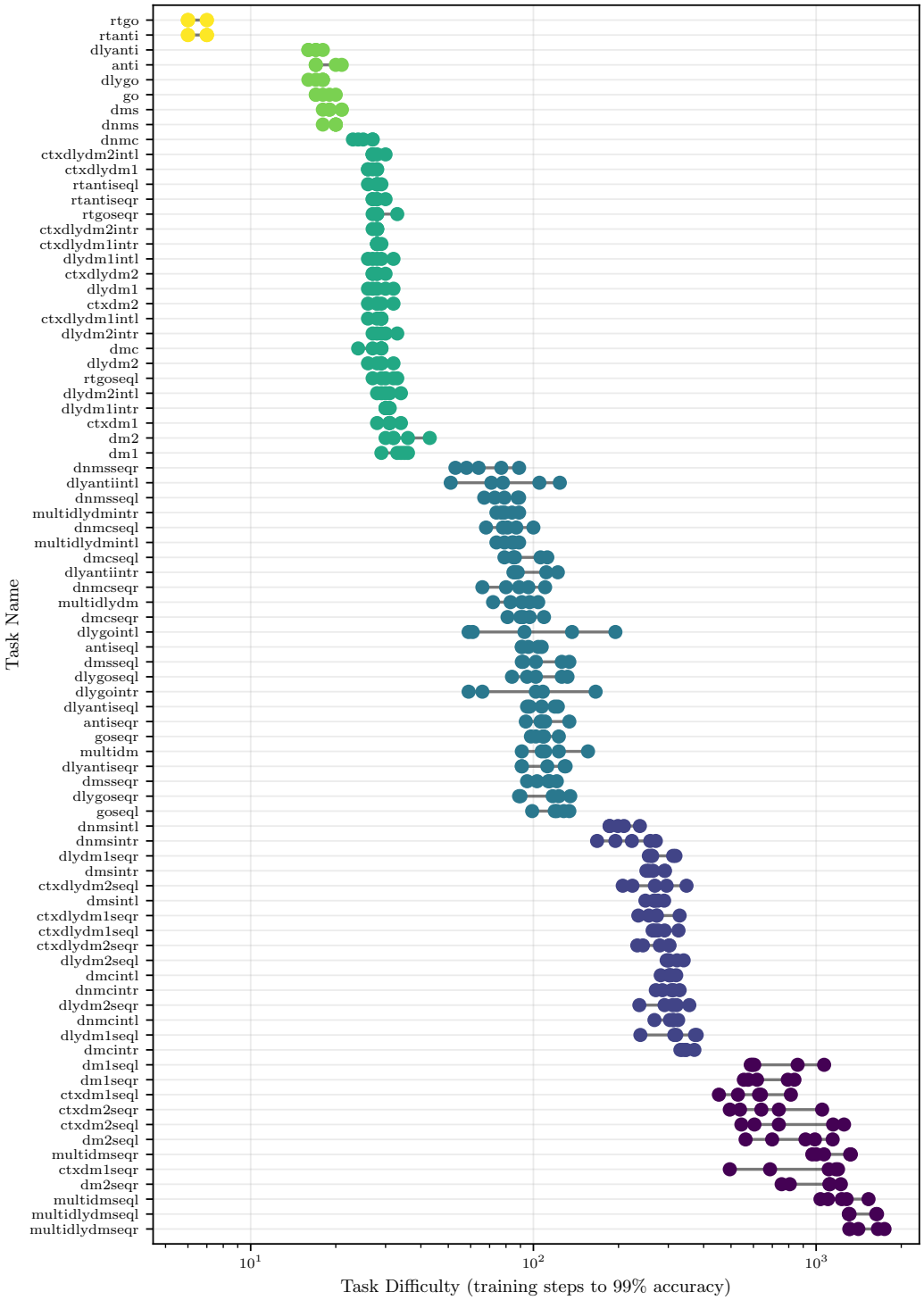


Figure 15: Five measurements of task difficulty made for each task in the Mod-Cog task suite [24]. Tasks are sorted by the median of the five measurements. Interestingly, the tasks form groups around similar difficulty levels, which we indicate with the color of each dot.

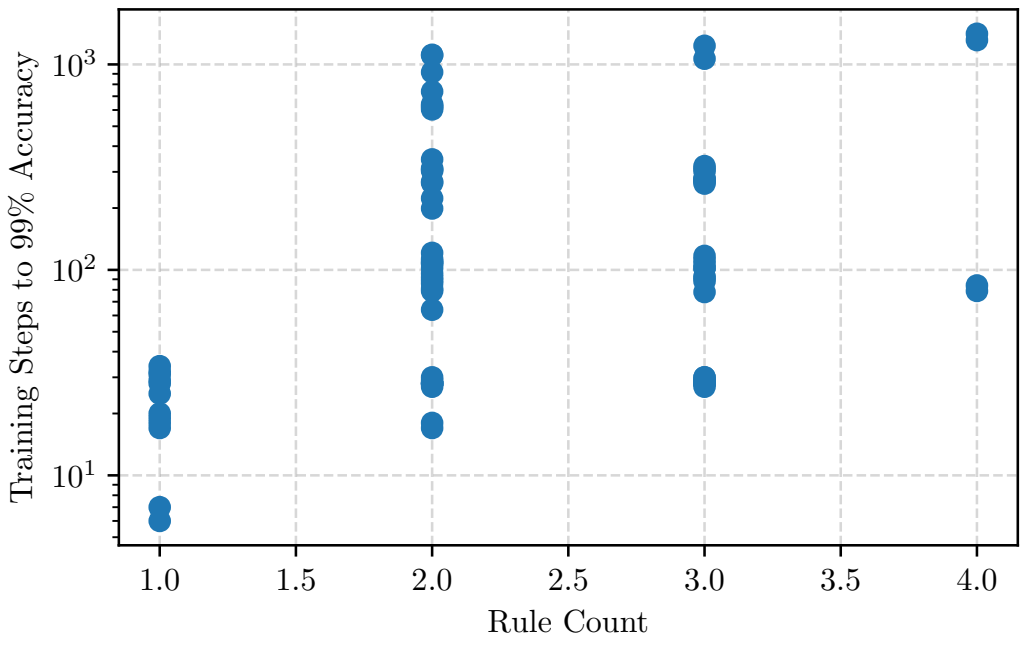


Figure 16: The number of rules that form a Mod-Cog task is correlated with the number of steps it takes a single RNN to learn the task ($r = 0.39$, $p < 0.0001$).