A Theory of Initialisation's Impact on Specialisation

Anonymous Author(s)
Affiliation
Address
email

Abstract

Prior work has demonstrated a consistent tendency in neural networks engaged in continual learning tasks, wherein intermediate task similarity results in the highest levels of catastrophic interference with prior learning. This phenomenon is attributed to the network's tendency to reuse learned features across tasks. However, this explanation heavily relies on the condition that such a neuron specialisation occurs, i.e. the emergence of localised representations. Our investigation challenges the validity of this assumption. Using theoretical frameworks for the analysis of neural networks, we show a strong dependence of specialisation on the initial condition. More precisely, we show that weight imbalance and high weight entropy can favour specialised solutions. We then apply these insights in the context of continual learning, first showing the emergence of a monotonic relation between task-similarity and forgetting in non-specialised networks, and, finally, assessing the implications on the commonly employed elastic weight consolidation regularisation technique.

1 Introduction

Theories of representation in biological neural networks span from highly localised representations in single neural units [1] to fully distributed or shared representations [2]. While shared representations offer greater resilience, specialised representations allow for more efficient encoding of information. Experimental evidence supports both ends of this spectrum, with different brain areas and tasks exhibiting distinct forms of representation [3, 4, 5, 6, 7]. Similarly, artificial neural networks display both shared [8, 9, 10] and specialised representations [11, 12], where a recent advancements in explainable AI, such as the Golden Gate Claude model [13], exemplify an extreme of the spectrum.

Given the trade-offs between shared and specialised representations, a critical research challenge lies in understanding how to guide neural networks towards one form or the other. This tension is especially relevant in contexts like disentangled representation learning [14] and multi-task learning [15], including continual learning and transfer learning. Specialised representations can facilitate faster adaptation and reduce catastrophic forgetting [16, 17], as they allow networks to rewire efficiently [18]. Rich Caruana's seminal work on multi-task learning [15] emphasised the value of specialisation in enhancing performance across multiple tasks. Recent efforts to mitigate catastrophic forgetting [19, 20] have led to the development of regularisation strategies that promote specialisation, such as elastic weight consolidation [21], synaptic intelligence [22], and learning without forgetting [23]. In disentangled representation learning, [24] highlighted that, despite the potential success of unsupervised approaches, disentanglement does not emerge naturally without an explicit inductive bias, underscoring the need for supervision to enforce such structures.

In this study, we investigate the role of initialisation in steering neural networks towards specialised or shared representations, providing a complementary perspective on both the lazy learning regime [25] and the rich learning regime [26, 27, 28]. Previous research [29, 30, 31] has showns that by interpolating between these regimes, we can transition from shared representations–characterised by random projections in the neural tangent kernels–to effective feature learning [32, 33, 34, 35, 36].

Submitted to the Mathematics of Modern Machine Learning Workshop at NeurIPS 2024. Do not distribute.

While our analysis remains within the feature learning regime, it adopts a distinct theoretical approach compared these studies, concentrating specifically on the impact of initialisation within standard synthetic frameworks for neural networks. This exploration reveals how initialisation can skew the learning dynamics towards either specialised or shared representations, thereby adding a new dimension to the study of learning dynamics in over-parameterised networks.

Our work makes the following **main contributions**:

- We study the impact of initialisation on specialisation through two theoretical frameworks:
  - We utilise the dynamics of **deep linear networks** to investigate the evolution of specialisation [37];
  - We extend this analysis to **high-dimensional mean-field neural networks**, drawing insights from stochastic gradient dynamics [38, 39, 40].
- Our findings challenge prevailing assumptions regarding the relationship between task similarity and catastrophic forgetting [41, 42, 43].
- Moreover, we identify specific initialisation schemes that promote specialised solutions by increasing the entropy of the readout weights and creating an imbalance between the first and last layers, akin to the findings of [35].
- Finally, we demonstrate the practical implications of our results on regularisation strategies, specifically analysing how Elastic Weight Consolidation (EWC) [21] is influenced by specialisation dynamics, highlighting potential pitfalls associated with regularisation methods in continual learning.

In Sec. 2, we introduce the concept of specialisation within the teacher-student framework and highlight the relevant literature. Sec. 3 explores this issue through the lens of deep linear dynamics, illustrating its impact on learned representations, particularly in the context of disentangled representation learning. Sec. 4 addresses the continual learning problem, revisiting existing theoretical frameworks and demonstrating how their conclusions may not hold under certain initialisation schemes. We conclude this section by discussing the implications for the EWC mitigation strategy. Finally, in Sec. 5, we reflect on the limitations of our work and propose future directions for research.

2 Specialisation in the teacher-student

The teacher-student framework is a generative model that allows for the controlled creation of synthetic datasets [44]. The framework involves two classifiers: the *teacher* and the *student*, for instance represented as neural networks as exemplified in Fig. 1a. The teacher, has fixed randomly drawn weights and maps random inputs $\boldsymbol{x}$ from a given distribution to labels, providing a rule for generating data. The student, on the other hand, updates its parameters through learning protocols like stochastic gradient descent (SGD) to approximate the teacher's outputs.

While a detailed quantitative characterisation of specialisation follows in the next sections, we briefly introduce the concept within the teacher-student framework. [38] showed that, when both teacher and student are modelled as committee machines, each student neuron specialises by aligning with a specific teacher neuron. Similarly, [45] observed that for certain activation functions in two-layer networks, an over-parameterised student will selectively use only a subset of those units to replicate the teacher's outputs. This phenomenon, termed specialisation, stands in contrast to a student redundantly sharing representations of the teacher across neurons. In this work we present a more comprehensive account of the factors underlying specialisation. In contrast to [45], we argue that initialisation—not the activation function—is chiefly responsible. We highlight this in Fig. 1b, by showing that with carefully chosen initialisations we can train a highly specialised ReLU student (bottom panels), and a non-specialising sigmoidal student (top), which represents the opposite of the conclusions presented in [45].

3 Specialisation explained using Linear Dynamics

As a first step towards understanding specialisation in neural networks we turn to the deep linear neural network paradigm [37]. While deep linear networks can only represent linear input-output mappings, they showcase intricate fixed point structure and nonlinear learning dynamics reminiscent

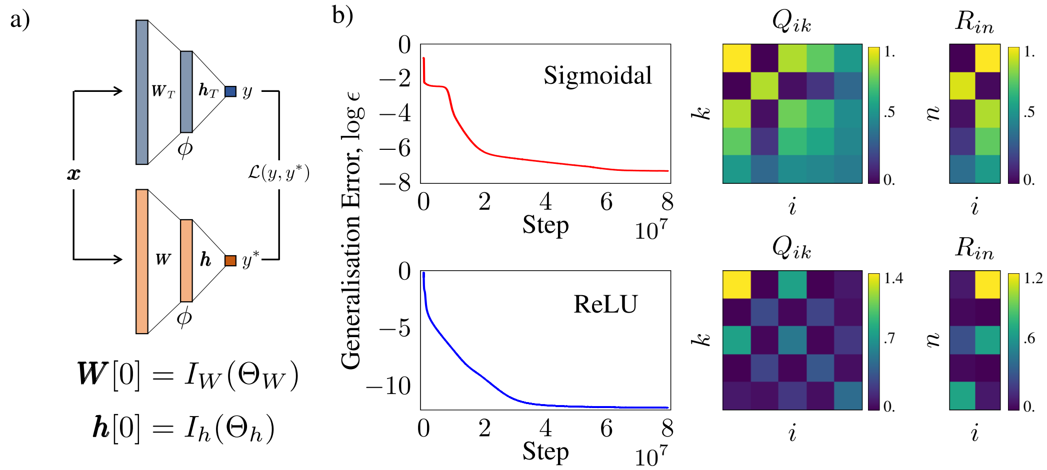

Figure 1: **Initialisation impacts specialisation.** a) In the teacher-student setup a student network is trained with labels generated by a fixed teacher network. Previous work established a relationship between the activation function ϕ and the propensity for the student nodes to specialise to teacher nodes. However we show in this work that this is an overly simplistic description; other factors including student weight initialisations I_W, I_h, parameterised by Θ_W, Θ_h arguably play a stronger role. b) Generalisation error curves for two simulations of the teacher-student setup, one with a ReLU activation function and one with a scaled error activation function. Θ_W and Θ_h are chosen to achieve a solution with ReLU that specialises—as indicated by sparser overlap matrices on the bottom right, and a scaled error function solution that does not specialise—as indicated by denser overlap matrices on the top right. A sparse (dense) Q matrix shows few (many) nodes are active, while a sparse (dense) R matrix shows student nodes are representing teacher nodes in a targeted (redundant) manner. Further details for the quantities described can be found in Sec. 4.

of phenomena seen in nonlinear networks. Deep linear networks have been successfully used to describe the effects of depth and nonlinearity, while showcasing the influence of initialisation [46, 47]. Here we construct a synthetic setup, to study the influence of initialisation on specialisation. In this work, we consider specialisation adhering to the definition of proposed by the statistical physics literature [45] which considers whether one neuron will account for all of the variance associated to one feature, while the others remain inactive. This is in contrast to other work on modularity [48] such as Neural Module Networks [49, 50, 51, 52], mixture-of-expert models [53, 54, 55], tensor product networks [56], among others [57, 58], which consider specialisation as a subset of a network or module performing a single "task" or only being activated by one interpretable feature in the dataset. Thus, these works are more concerned with *what* is learned and consider specialisation to imply feature sparsity [59]. While we are concerned with the manner in which learning is represented, a phenomenon closer to activation sparsity.

3.1 Specialisation in the deep linear network framework

To connect this framework to specialisation we use the notion of the "neural race" from [46]. The neural race hypothesis says that the pathways through a network are racing to explain the variance in the dataset (perform the input-output mapping). Thus, we consider the limited case of a network with two hidden neurons and one output neuron. Fig. 2 depicts the setup, notation and strategy for this section. We ask the question: "when will one pathway finish learning (reach it's hitting time t^*) before the other begins learning (reaches it's escaping time $\hat{t}$)". In cases when this occurs, the network would have specialised as only one pathway will have any activity and will explain all of the data. Similar to Sec. 4 we generate data by sampling the elements of a data point from a Gaussian distribution ($x_i \sim \mathcal{N}(0, 1)$) with $i = 1, \ldots, d$. We then define a ground-truth mapping ($\boldsymbol{W}_T$) and generate labels $y = \boldsymbol{W}_T \cdot \boldsymbol{x}$. We only consider regression tasks in this section, thus $y \in \mathbb{R}$. For P inputs we can form the input matrix $\boldsymbol{X} \in \mathbb{R}^{d \times P}$ and row vector of scalar outputs $\boldsymbol{y} \in \mathbb{R}^{1 \times P}$. The dataset statistics which drive learning are collected in the input and input-output correlation matrices, Σ^x and Σ^{yx}

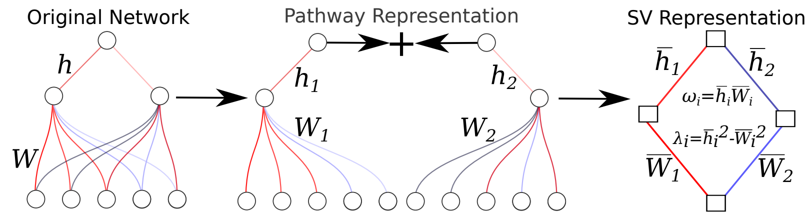

Figure 2: Summary of our setup, notation and strategy. a) The original network with two hidden neurons learning the regression task. b) We split the network into two separate pathways and consider their dynamics individually. Since both networks are learning the same task simultaneously, their dynamics are coupled. c) To obtain the dynamics of the two pathways and calculate their escaping and hitting time we track the pathway dynamics in terms of the network's effective singular values. The closed form dynamics for the pathway singular value are given in Eq. 3.

respectively. For the task described above the singular value decomposition of these matrices are:

$$\boldsymbol{\Sigma}^x = E[\boldsymbol{X}\boldsymbol{X}^T] = \boldsymbol{V}\boldsymbol{D}\boldsymbol{V}^T, \qquad \boldsymbol{\Sigma}^{yx} = E[\boldsymbol{y}\boldsymbol{X}^T] = us\boldsymbol{v}^T. \tag{1}$$

Here, $u \in \{-1, 1\}$, $\boldsymbol{v}$ is a vector such that $\boldsymbol{v}^T\boldsymbol{v} = 1$ and $\boldsymbol{V}$ is an orthogonal singular vector matrix. Correspondingly, s is the singular value for the rank 1 task and $\boldsymbol{D}$ is a diagonal matrix of singular values. Note that we assume that the correlation matrices are mutually diagonalisable (share the same $\boldsymbol{V}$) up to the rank of Σ^{yx}.

For this task we consider a single hidden layer network (Fig. 2 left) computing output $\hat{y} = \boldsymbol{h}\boldsymbol{W}\boldsymbol{x}$ with $\boldsymbol{h} \in \mathbb{R}^K$ and $\boldsymbol{W} \in \mathbb{R}^{K\times d}$ in response to an input $\boldsymbol{x} \in \mathbb{R}^d$. The network is trained to minimise the mean squared error loss using full batch gradient descent with a small learning rate η. To identify when specialisation will occur in this network, we split the network into two pathways with one hidden neuron each. The input and output dimensions remain the same (Fig. 2 middle). Finally we obtain the linear dynamics (ultimately depicted as Eq. 3) for each pathway (the full details and assumptions of the derivation are given in Appendix A). In this setting, the network's input-output mapping after t epochs of training is $\boldsymbol{h}(t)\boldsymbol{W}(t)$. Assuming that the network weights align to the singular vectors of the dataset from early in training, as described by the "silent alignment effect" [60], we perform a change of variables and write the network mapping in terms of the dataset singular vectors:

$$\boldsymbol{h}(t)\boldsymbol{W}(t) = u\omega(t)\boldsymbol{v}^T, \tag{2}$$

where $\omega(t)$ is the network pathway's scalar effective singular value and the only time-dependent component of the decomposed network mapping. While the alignment assumption is strong, linear paradigms with these assumptions have been used successfully in the past [47, 61, 62, 48, 35]. With the change of variables we can now obtain a closed form equation describing how ω evolves through time as:

$$\omega(t) = \frac{\lambda}{2}\sinh\left\{2\tanh^{-1}\left[\frac{K\left(C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)-1\right) - \lambda D\left(C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)+1\right)}{2S\left(C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)+1\right)}\right]\right\} \tag{3}$$

where C is a defined constant, $\tau = \frac{1}{\eta}$ is the learning time constant and $K = \sqrt{4S^2 + \lambda^2 D^2}$. Eq. 3 shows that K is the variable interacting with time (t) and as a consequence determines how quickly the network will learn. Three factors affect K fastening learning: 1. S the input-output correlation matrix singular value, 2. D the input correlation matrix singular value, and 3. $\lambda = h^2 - W^2$ which denotes the imbalance between the weights of the network. Notice that–as shown in Appendix A–λ is a conserved quantity and constant throughout training. Thus, given a dataset–which characterises the S and D matrices–the only property which can promote faster learning in the network is to increase the imbalance parameter. For our experiments we whiten the input data $\boldsymbol{x}$ such that $K = \sqrt{4S^2 + \lambda^2}$ to remove one of the interactions within K.

Fig. 3(a-c) show a confirmation of the validity our theory by comparing with simulations. Instead, Fig. 3d represents the main result of this section. We consider both network pathways and vary the weight imbalance for each (λ_{slow} for the pathway with the lower imbalance and λ_{high} for the pathway with the larger imbalance). We place these two values on the axes and in colour depict in log scale how close the slower pathway comes to reaching its escaping time across its training. When negative, it means that during training there is a timestep where the network is less than one

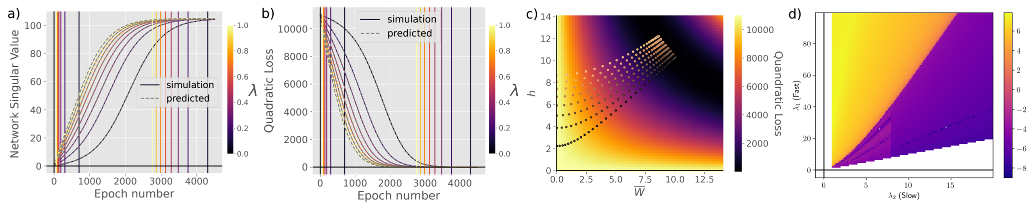

Figure 3: **Linear Dynamics from imbalanced initialisation leads to specialisation.** *Panels a-c)* Show agreement between our theoretical curves and simulations for the training dynamics of: (a) the network's singular values, (b) the network's loss, (c) and the network's movement in weight space. In (a,b) the colour indicates the singular value used for the input weights, while in (c) the colour represents the loss. *Panel d)* shows a phase diagram representing how pathways with different initial weight imbalances lead to specialisation. The two axis represent the initial singular values associated to the different pathways. The colour represents the amount of time it takes the slower pathways to learn in logscale. We see that the more inbalanced the fast pathway relative to the slower pathway, the more likely the network will specialise. The white region represents when the inbalance is reversed.

epoch from its escaping time (so it will learn). In this case there will not be specialisation as both pathways will learn some part of the input-output mapping. When the colour is positive it means there will be specialisation as the slower pathway is always at least a full epoch away from learning. It is important to note that the slower pathway's escaping time is moving constantly as the faster pathway accounts for variance in the data. This decreases the input-output singular value in K for this pathway and makes learning slower. Due to this coupling we are also unable to obtain completely closed form equations for the slower pathway in term's of the faster pathway's effective singular value. However, this phase diagram would not be computationally feasible without the closed-form escaping time, hitting time and training dynamics (see Appendix C for our process on constructing this plot). Finally, we only consider imbalances where the output layer is larger than the input layer. Recent work [33, 35] has shown that having larger input weight pushes the network towards lazy learning [30] while output heavy imbalance promotes feature learning. Since we are concerned with the latter in this work, we focus on the output heavy imbalanced setting for both pathways. From Fig. 3 we see that there is a clear phase transition from non-specialised representations to specialised ones. This occurs with increasing imbalance of the faster pathway. Increasing the imbalance of the slower pathway can similarly combat this specialisation pressure. Thus, the relative imbalance of the two pathway at initialisation will dictate whether specialised representations are learned.

3.2 Specialisation underlies disentanglement

We extend the results on inbalanced initialisation and applied them, beyond the limited setting of our framework, in the context of disentangled representation learning, where the goal is to separate latent factors. [14] introduced the importance of disentanglement for interpretability and generalisation. A seminal contribution to this domains came with the β-VAE model, where [63] demonstrated how increasing the KL-divergence term can enforce disentanglement by encouraging specialised latent representations. Many studies have built upon these foundational frameworks to enhance disentanglement performance, exploring different training regimes [64, 65] and loss functions [66, 67, 68]. Here we contribute to this literature by applying our theoretical insights and examining the impact of initialisation on disentanglement performance.

Specifically, we examine how initialisation impacts specialisation in disentanglement learning on the 3DShapes dataset [69] using the β-VAE model–widely adopted for such tasks [63, 70]. We implement a β-VAE model, employing the "DeepGaussianLinear" architecture for the decoder and the "DeepLinear" architecture for the encoder, as specified in [24]. Both architectures are composed of five fully connected layers with ReLU activations. The model is trained using the Adam optimiser, optimising a loss function that combines KL divergence and binary cross-entropy-based reconstruction loss. Additional details are given in Appendix D. In these experiments, we adjust the variance of the weights in a deep fully-connected encoder, by varying the constant gain of the Xavier initialisation [71]. Specifically, the first block of layers was initialised with gain g while the readout layer received a gain $1/g$. Notice that $g = 1$ represents the standard initialisation scheme.

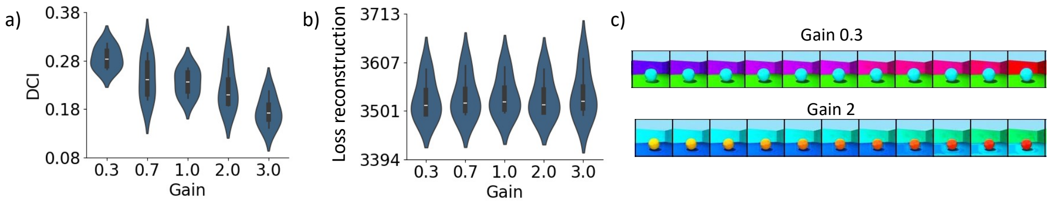

Figure 4: *Panel a)* Violin plots of the DCI values against the gain. *Panel b)* Violin plots of the reconstruction loss against the gain. The standard deviation was computed over four seeds. *Panel c)* Example Traverse of model with gain 2 and .3 respectively showcasing

Results are shown in Fig. 4, despite very similar levels of reconstruction loss, networks initialised with smaller gains improved disentanglement in the β-VAE network, as reflected in higher Disentanglement, Completeness, and Informativeness (DCI) scores [72]. This result confirms that modulating the initialisation gain can either enhance or reduce the network's disentanglement. Although the scope of these experiments is limited, they provide preliminary validation of our theoretical framework in more realistic contexts, encouraging further investigation into alternative initialisation schemes with varying levels of balance.

4 Continual Learning

As [15] noted, multi-task learning benefits significantly from task-specific specialisation, allowing the network to better preserve performance across multiple domains. In the context of continual learning, [41] and [42] observed that forgetting does not monotonically increase with task similarity. [43] provided a mechanistic explanation, showing that this phenomenon is due to the interplay between re-use of specialised neurons and activation of unused ones. In this section, we build on these findings and show that this phenomenology can be disrupted by initialisation schemes that disincentives specialisation.

4.1 Continual Learning in the two-layer teacher-student setup

We use a teacher-student framework, introduced in Sec. 2, which has been analysed in [42, 43]. This model consists of two randomly initialised teacher networks—one for an upstream task and one for a downstream task. Each teacher is represented by two-layer neural networks with P^* hidden units and weights $\boldsymbol{W}_T^{(1)}$, $\boldsymbol{h}_T^{(1)}$ for the upstream task, and $\boldsymbol{W}_T^{(2)}$, $\boldsymbol{h}_T^{(2)}$ for the downstream task. Given a random input $\boldsymbol{x} \in \mathbb{R}^d$, drawn i.i.d. from a Gaussian distribution $x_i \sim \mathcal{N}(0,1)$, the teachers generate labels according to the equation:

$$y^{(t)} = \boldsymbol{h}_T^{(t)} \cdot \phi\left(\frac{\boldsymbol{W}_T^{(t)}\boldsymbol{x}}{\sqrt{d}}\right) \quad \text{for } t = 1, 2, \tag{4}$$

where ϕ is a non-linear activation function, chosen here as $\phi(z) = \operatorname{erf}\left(z/\sqrt{2}\right)$. This setup allows us to generate two datasets $\mathcal{D}^{(1)}$ and $\mathcal{D}^{(2)}$, with controlled similarity between the tasks by manipulating the teacher weights. Specifically, we generate $\boldsymbol{W}_T^{(1)}$, $\boldsymbol{h}_T^{(1)}$, and $\boldsymbol{h}_T^{(2)}$ with i.i.d. Gaussian entries, while $\boldsymbol{W}_T^{(2)}$ is generated as:

$$\boldsymbol{W}_T^{(2)} = \gamma \boldsymbol{W}_T^{(1)} + \sqrt{1-\gamma^2}\boldsymbol{W}_T^{(\text{aux})}, \tag{5}$$

where $\boldsymbol{W}_T^{(\text{aux})}$ is an auxiliary weight matrix, and γ controls the correlation between tasks. The student is a two-layer neural network with P hidden units, using the same non-linearity ϕ. It is trained using online stochastic gradient descent on a squared error loss, with a shared first-layer weight matrix $\boldsymbol{W}$ and task-specific readout weights $\boldsymbol{h}^{(1)}$ and $\boldsymbol{h}^{(2)}$. For both layers, the initial weights are sampled i.i.d. from a Gaussian distribution, with the first-layer weights $\boldsymbol{W}$ having standard deviation σ_W. While most previous studies follow a similar scheme for the readout weights, we introduce a novel initialisation scheme using polar coordinates, as detailed in Eq. 11. The updates for $\boldsymbol{W}$ and $\boldsymbol{h}^{(t)}$ at iteration e, under SGD on the squared error loss, are given by:

$$\boldsymbol{W}[e+1] = \boldsymbol{W}[e] - \frac{\eta}{\sqrt{d}}\left(\boldsymbol{h}^{(t)} \cdot \phi\left(\frac{\boldsymbol{Wx}}{\sqrt{d}}\right) - y^{(t)}\right)\phi'\left(\frac{\boldsymbol{Wx}}{\sqrt{d}}\right)\boldsymbol{v}^{(t)}\boldsymbol{x}, \tag{6}$$

$$\boldsymbol{h}^{(t)}[e+1] = \boldsymbol{h}^{(t)}[e] - \frac{\eta}{d}\left(\boldsymbol{h}^{(t)} \cdot \phi\left(\frac{\boldsymbol{Wx}}{\sqrt{d}}\right) - y^{(t)}\right)\phi\left(\frac{\boldsymbol{Wx}}{\sqrt{d}}\right), \tag{7}$$

where η is the learning rate and $y^{(t)}$ is the target output from the teacher network for task t.

In the large input dimension limit $d \to \infty$, key observables, such as the generalisation error, can be captured by a few order parameters:

$$\boldsymbol{Q} = \frac{1}{d}\boldsymbol{W}\boldsymbol{W}^T, \qquad \boldsymbol{R}^{(t)} = \frac{1}{d}\boldsymbol{W}\boldsymbol{W}_T^{(t),T}, \qquad \boldsymbol{T}^{(t,t')} = \frac{1}{d}\boldsymbol{W}_T^{(t)}\boldsymbol{W}_T^{(t'),T}, \qquad \boldsymbol{h}^{(t)}, \qquad \boldsymbol{h}_T^{(t)}; \tag{8}$$

where $t, t' \in \{1, 2\}$ refer to the two tasks. The generalisation error for task t is then:

$$\begin{aligned}\epsilon^{(t)} &= \frac{1}{2}\mathbb{E}_{\boldsymbol{x}}\left[\left(\boldsymbol{h}^{(t)} \cdot \phi\left(\frac{\boldsymbol{Wx}}{\sqrt{d}}\right) - y^{(t)}\right)^2\right] \\ &= I_{21}(\boldsymbol{Q}, \boldsymbol{h}^{(t)}) + I_{21}(\boldsymbol{T}^{(t,t)}, \boldsymbol{h}_T^{(t)}) - \frac{1}{2}I_{22}(\boldsymbol{Q}, \boldsymbol{R}^{(t)}, \boldsymbol{T}^{(t,t)}, \boldsymbol{h}^{(t)}, , \boldsymbol{h}_T^{(t)}),\end{aligned} \tag{9}$$

where I_{21} and I_{22} are explicit functions of the order parameters, detailed in Appendix B. The evolution of these parameters throughout training can be tracked to study the learning dynamics, as first shown in [39, 40, 45]. For the specific case of continual learning, [42] derived the governing ordinary differential equations (ODEs), provided in Appendix B.

4.2 Specialisation relevance for continual learning

The continual learning results in the teacher-student setup, including the non-monotonic relationship between catastrophic forgetting and task similarity, often implicitly assume that the student has specialised to the teacher in the first task. This assumption allows for spare capacity to represent the second task. However, as shown in Fig. 1b, there are regimes where this assumption of specialisation is violated. Here, we expand on these findings and their implications for forgetting.

A student can effectively ignore a unit in two ways: either the unit's post-activation is near 0 (inactive), or the corresponding second-layer weight is 0. This motivates three measures for specialisation based on the definition of entropy–over the hidden units, head weights, and the product of both:

$$H_h = -\sum_i^P |\tilde{h_i}| \log |\tilde{h}_i|, \quad H_Q = -\sum_i^P \tilde{Q}_{ii} \log \tilde{Q}_{ii}, \quad H_m = -\sum_i^P \tilde{Q}_{ii}|\tilde{h}_i| \log(\tilde{Q}_{ii}|\tilde{h}_i|); \tag{10}$$

where the tilde denote normalisation, i.e. $|\tilde{h_i}| = \frac{|h_i|}{\sum_i^P |h_i|}$ and $\tilde{Q}_{ii} = \frac{Q_{ii}}{\sum_i^P Q_{ii}}$. Maximum entropy in these measures corresponds to no specialisation, while minimum entropy corresponds to maximum specialisation.

We can investigate how these measures vary as a function of different properties of the problem setup, in particular those related to initialisation. To simplify the analysis, we begin with the case where the optimal number of tasks is $P^* = 1$ and the network has $P = 2$ output units. This allows us to initialise the second layer weights in polar coordinates, with precise and interpretable control over scale and asymmetry of weights. Formally we parameterise our readout initialisations according to

$$\boldsymbol{h}^{(t)}[0; r^{(t)}, \theta^{(t)}] = (r^{(t)} \cos\theta^{(t)}, r^{(t)} \sin\theta^{(t)}). \tag{11}$$

Fig. 5 contain phase diagrams showing how the entropy measures in Eq. 10 vary with the initialisation parameters $r^{(t)}$, $\theta^{(t)}$, and σ_W. We can make several observations: (i) the strongest determinant of specialisation is the asymmetry in the second layer weights, i.e. the θ parameter. (ii) this is the case for both ReLU and sigmoidal activation functions, reinforcing the point made in the example from Fig. 1b. (iii) the scale of initialisations (parameters σ_W, r) are also important.

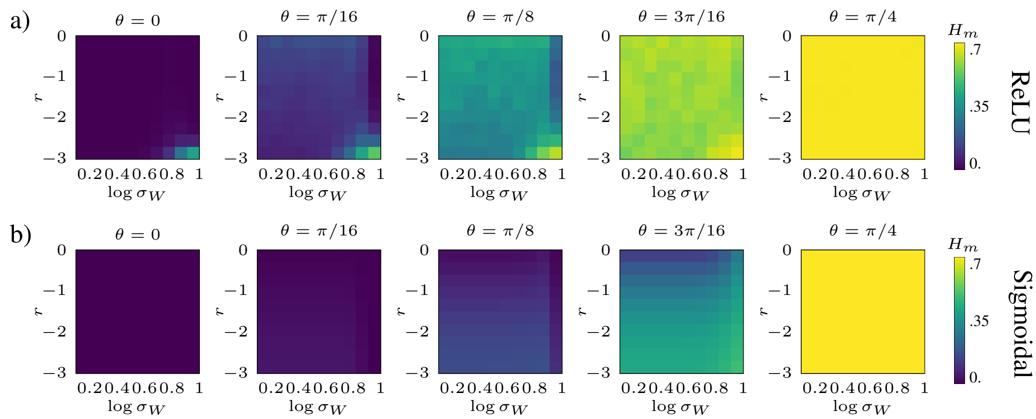

Figure 5: **Phase diagrams show significance of initialisation for specialisation.** The phase diagrams show with colour the aggregated entropy Eq. 10 evaluated for different initialisations. On the x-axis we span over the standard deviation of the first layer. The second layer is initialised using polar coordinates, and the y-axis represents the norm while the different panels give the angle spanning from orthogonal units ($\theta = 0$) to identical units ($\theta = \pi/4$). Specialisation is achieved by blue-leaning initialisations, while yellow-leaning ones exhibit high entropy and therefore non-specialised solutions. Additional results can be found in Appendix E. standard

4.3 Specialisation underlies Maslow's hammer

The phase diagrams in Fig. 5 demonstrate that initialisation can drastically change the type of solutions found by the student after training on one teacher. While this may be inconsequential if the generalisation error remains unaffected, in many cases, the precise nature of the learned representation can significantly impact downstream tasks.

In the worst case scenario, the student undergoes no specialisation during the first task. During the second task there is no notion of the trade-off between node re-use and node activation discussed in [43]; rather the student continues to find a non-specialised solution to the second teacher, effectively fully re-using it's entire representation for the second task. Consequently, the amount of forgetting with respect to the initial task decreases monotonically with task similarity, thereby breaking the U-shaped pattern characteristic of Maslow's hammer that has been observed in various continual learning setups [41]. This extreme case is illustrated in Fig. 6. Further, *even with* specialisation after the first task, large asymmetric initialisation in the second task readout weights can induce this monotonic relationship, again by pushing the student into re-use rather than activation.

In a broader context, a rich diversity of behaviours can emerge, driven by factors such as the initialisation schemes, the scale of weights in the first layer, and the readout heads for both tasks. A glimpse of this behavioural diversity is provided in Appendix F, where we further explore the interaction between these factors and their impact on forgetting in continual learning.

4.4 Specialisation underlies EWC

The findings relating specialisation to forgetting from subsection 4.3 have direct consequences for interference mitigation strategies such as EWC. EWC is a regularisation-based method that computes a measure of 'importance' for each weight with respect to a task via the Fischer information [21]. Subsequently a squared penalty scaled by this importance is applied to deviation of this weight during learning of future tasks as follows:

$$\mathcal{L}_{\text{EWC}}(\boldsymbol{W}) = \mathcal{L}(\boldsymbol{W}) + \frac{\xi}{2} \sum_i F_i (W_i - W_i^*), \tag{12}$$

where F is the Fischer information matrix, ξ is a regularisation strength parameter, and $\boldsymbol{W}^*$ are the weights at the end of training on the first task.

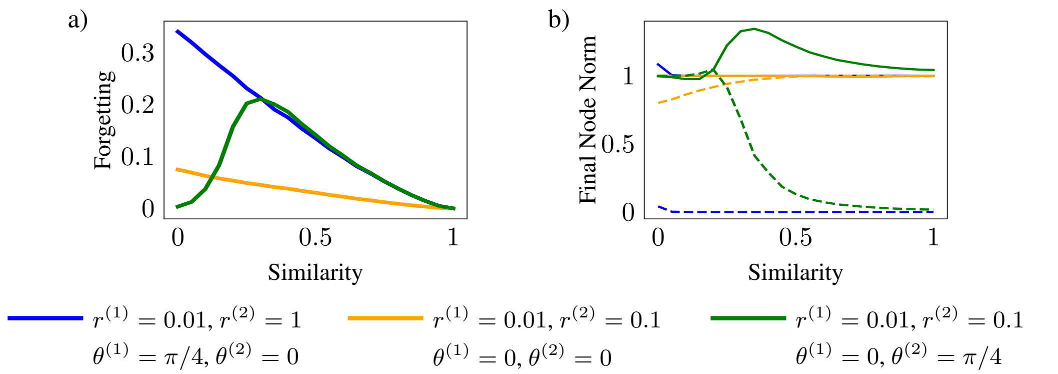

Figure 6: **Initialisation and specialisation properties can influence profile of forgetting vs. similarity.** *(a)* forgetting as a function of task similarity can be both monotonic, shown here for the cases of specialisation after the first task + large second head initialisation (blue), and no-specialisation during both tasks (orange); *or* non-monotonic (green, as characterised by Maslow's hammer [43]). *(b)* the final norm of the two nodes (one solid and one dashed), i.e. at the end of training on both tasks, as a function of task similarity. In the cases that lead to monotonic forgetting, nodes are fully re-used, either because the corresponding new head is initialised large (orange) or because the new head is symmetrically initialised and the nodes continue to represent redundant information during the second task (blue). *Params:* $N = 10000$, $\eta = 1$, $K^* = 1$, $K = 2$, $\sigma_w = 0.001$.

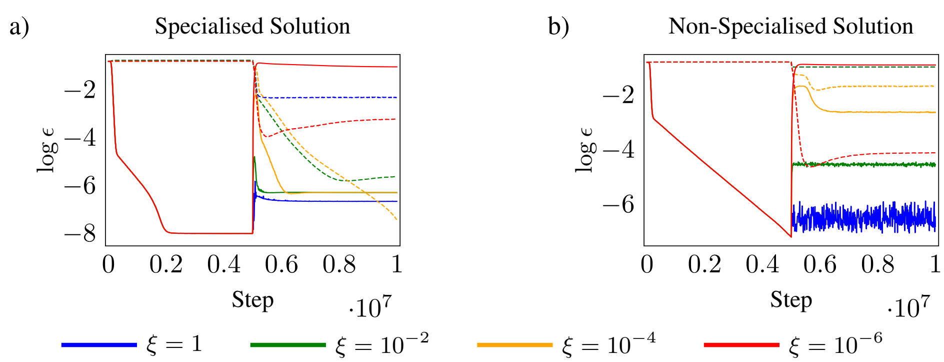

Figure 7: **EWC is strongly reliant on specialisation.** We show the generalisation error in the first (solid line) and second (dashed) task for different EWC regularisation strengths. *(a)* When the student finds a specialised solution to the first task, there is a range of EWC regularisation strength ξ for which the activated units can remain fixed and spare capacity can be used to learn the second task—leading to low generalisation error in both tasks ($\xi = 10^{-2}$, $\xi = 10^{-4}$ perform very well). *(b)* When the student does not specialise in the first task, EWC reduces to an inflexible regulariser that either penalises plasticity everywhere—leading to little forgetting but no further learning (e.g. $\xi = 1$), or does not penalise any plasticity—leading to catastrophic forgetting (e.g. $\xi = 10^{-6}$).

In cases where the network does not specialise, i.e. multiple student nodes learn redundant representations for a given teacher node, the nodes have equal importance. Consequently EWC cannot distinguish between these sets of weights and depending on the regularisation parameter λ either lets these nodes move during training on the second task (under-regularises) leading to forgetting, or lets none move (over-regularises) leading to no transfer. We show results illustrating this behaviour in the teacher-student setup in Fig. 7. In particular we show the regime of intermediate task similarity, wherein [43] previously argued that EWC should perform better than methods such as replay.

5 Limitations and Perspectives

This work operates within simplified frameworks, which–while widely used in the analysis of neural networks–do not fully capture the complexity of modern architectures and real-world data. Our experiments rely on Gaussian input data and simplified input-output relations, which are far removed from the intricacies of real-world scenarios. A natural next step is to extend our analysis to more realistic generative models, such as the hidden manifold model [73] or the superstatistical generative model [74], which offer more structured data distributions and better capture observations from real data experiments.

Another promising direction is to complement analytical approaches with numerical experiments on controlled real-world datasets. While this may sacrifice some analytical tractability, it brings us closer to addressing practical challenges. For instance, transfer learning settings, such as those explored in [75], provide a useful benchmark for testing our theoretical findings in more complex environments.

While the current work remains theoretical in nature, focusing on simplified models for analytical tractability, a thorough exploration of the practical implications of our findings, particularly in disentangled representation learning, is beyond the scope of this paper. However, we aim to address this in future work by shifting towards a more experimental approach. Specifically, we plan to explore a broader range of network architectures, datasets–such as Car3D [**?**] and dSprites [76]–and evaluation metrics—such as SAP [68, 63]. This future study will allow us to validate our theoretical insights and fully assess their relevance in real-world settings.

References

[1] Horace B Barlow. Single units and sensation: a neuron doctrine for perceptual psychology? *Perception*, 1(4):371–394, 1972.

[2] John J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, 79(8):2554–2558, 1982.

[3] Colin Blakemore, James PJ Muncey, and Rosalind M Ridley. Stimulus specificity in the human visual system. *Vision research*, 13(10):1915–1931, 1973.

[4] R Quian Quiroga, Leila Reddy, Gabriel Kreiman, Christof Koch, and Itzhak Fried. Invariant visual representation by single neurons in the human brain. *Nature*, 435(7045):1102–1107, 2005.

[5] Apostolos P Georgopoulos, Andrew B Schwartz, and Ronald E Kettner. Neuronal population coding of movement direction. *Science*, 233(4771):1416–1419, 1986.

[6] Alumit Ishai, Leslie G Ungerleider, Alex Martin, and James V Haxby. The representation of objects in the human occipital and temporal cortex. *Journal of cognitive neuroscience*, 12(Supplement 2):35–51, 2000.

[7] Bruno B Averbeck, Peter E Latham, and Alexandre Pouget. Neural correlations, population coding and computation. *Nature reviews neuroscience*, 7(5):358–366, 2006.

[8] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[9] Dumitru Erhan, Aaron Courville, Yoshua Bengio, and Pascal Vincent. Why does unsupervised pre-training help deep learning? In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 201–208. JMLR Workshop and Conference Proceedings, 2010.

[10] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? *Advances in neural information processing systems*, 27, 2014.

[11] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I 13*, pages 818–833. Springer, 2014.

[12] Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In Anna Korhonen, David R. Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 5797–5808. Association for Computational Linguistics, 2019.

[13] Adly Templeton. *Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet*. Anthropic, 2024.

[14] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

[15] Rich Caruana. Multitask learning. *Machine learning*, 28:41–75, 1997.

[16] Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier, 1989.

[17] Roger Ratcliff. Connectionist models of recognition memory: constraints imposed by learning and forgetting functions. *Psychological review*, 97(2):285, 1990.

[18] Steven C Suddarth and YL Kergosien. Rule-injection hints as a means of improving network performance and learning time. In *European association for signal processing workshop*, pages 120–129. Springer, 1990.

[19] German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural networks*, 113:54–71, 2019.

[20] Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Aleš Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE transactions on pattern analysis and machine intelligence*, 44(7):3366–3385, 2021.

[21] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

[22] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *International conference on machine learning*, pages 3987–3995. PMLR, 2017.

[23] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017.

[24] Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Raetsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. In *international conference on machine learning*, pages 4114–4124. PMLR, 2019.

[25] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems*, 31, 2018.

[26] Song Mei, Andrea Montanari, and Phan-Minh Nguyen. A mean field view of the landscape of two-layer neural networks. *Proceedings of the National Academy of Sciences*, 115(33):E7665–E7671, 2018.

[27] Lenaic Chizat and Francis Bach. On the global convergence of gradient descent for over-parameterized models using optimal transport. *Advances in neural information processing systems*, 31, 2018.

[28] Grant M Rotskoff and Eric Vanden-Eijnden. Neural networks as interacting particle systems: Asymptotic convexity of the loss landscape and universal scaling of the approximation error. *stat*, 1050:22, 2018.

[29] Lenaic Chizat, Edouard Oyallon, and Francis Bach. On lazy training in differentiable programming. *Advances in neural information processing systems*, 32, 2019.

[30] Mario Geiger, Stefano Spigler, Arthur Jacot, and Matthieu Wyart. Disentangling feature and lazy training in deep neural networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2020(11):113301, 2020.

[31] Blake Bordelon and Cengiz Pehlevan. Self-consistent dynamical field theory of kernel evolution in wide neural networks. *Advances in Neural Information Processing Systems*, 35:32240–32256, 2022.

[32] Salma Tarmoun, Guilherme Franca, Benjamin D Haeffele, and Rene Vidal. Understanding the dynamics of gradient flow in overparameterized linear models. In *International Conference on Machine Learning*, pages 10153–10161. PMLR, 2021.

[33] Daniel Kunin, Allan Raventós, Clémentine Dominé, Feng Chen, David Klindt, Andrew Saxe, and Surya Ganguli. Get rich quick: exact solutions reveal how unbalanced initializations promote rapid feature learning, 06 2024.

[34] Yizhou Xu and Liu Ziyin. When does feature learning happen? perspective from an analytically solvable model. *arXiv preprint arXiv:2401.07085*, 2024.

[35] Clémentine C. J. Dominé, Nicolas Anguita, Alexandra M. Proca, Lukas Braun, Daniel Kunin, Pedro A. M. Mediano, and Andrew M. Saxe. From lazy to rich: Exact learning dynamics in deep linear networks, 2024.

[36] Aditya Vardhan Varre, Maria-Luiza Vladarean, Loucas Pillaud-Vivien, and Nicolas Flammarion. On the spectral bias of two-layer linear networks. *Advances in Neural Information Processing Systems*, 36, 2024.

[37] Andrew M Saxe, James L McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv preprint arXiv:1312.6120*, 2013.

[38] David Saad and Sara A Solla. On-line learning in soft committee machines. *Physical Review E*, 52(4):4225, 1995.

[39] David Saad and Sara Solla. Dynamics of on-line gradient descent learning for multilayer neural networks. *Advances in neural information processing systems*, 8, 1995.

[40] Michael Biehl and Holm Schwarze. Learning by on-line gradient descent. *Journal of Physics A: Mathematical and general*, 28(3):643, 1995.

[41] Vinay V Ramasesh, Ethan Dyer, and Maithra Raghu. Anatomy of catastrophic forgetting: Hidden representations and task semantics. *arXiv preprint arXiv:2007.07400*, 2020.

[42] Sebastian Lee, Sebastian Goldt, and Andrew Saxe. Continual learning in the teacher-student setup: Impact of task similarity. In *International Conference on Machine Learning*, pages 6109–6119. PMLR, 2021.

[43] Sebastian Lee, Stefano Sarao Mannelli, Claudia Clopath, Sebastian Goldt, and Andrew Saxe. Maslow's hammer for catastrophic forgetting: Node re-use vs node activation. *arXiv preprint arXiv:2205.09029*, 2022.

[44] Elizabeth Gardner and Bernard Derrida. Three unfinished works on the optimal storage capacity of networks. *Journal of Physics A: Mathematical and General*, 22(12):1983, 1989.

[45] Sebastian Goldt, Madhu Advani, Andrew M Saxe, Florent Krzakala, and Lenka Zdeborová. Dynamics of stochastic gradient descent for two-layer neural networks in the teacher-student setup. *Advances in neural information processing systems*, 32, 2019.

[46] Andrew Saxe, Shagun Sodhani, and Sam Jay Lewallen. The neural race reduction: Dynamics of abstraction in gated networks. In *International Conference on Machine Learning*, pages 19287–19309. PMLR, 2022.

[47] Andrew M Saxe, James L McClelland, and Surya Ganguli. A mathematical theory of semantic development in deep neural networks. *Proceedings of the National Academy of Sciences*, 116(23):11537–11546, 2019.

[48] Devon Jarvis, Richard Klein, Benjamin Rosman, and Andrew M Saxe. On the specialization of neural modules. In *The Eleventh International Conference on Learning Representations*, 2023.

[49] Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. Neural module networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 39–48, 2016.

[50] Ronghang Hu, Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Kate Saenko. Learning to reason: End-to-end module networks for visual question answering. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 804–813, 2017.

[51] Ronghang Hu, Jacob Andreas, Trevor Darrell, and Kate Saenko. Explainable neural computation via stack neural module networks. In *Proceedings of the European conference on computer vision (ECCV)*, pages 53–69, 2018.

[52] Jacob Andreas. Measuring compositionality in representation learning. In *International Conference on Learning Representations*, 2018.

[53] Saeed Masoudnia and Reza Ebrahimpour. Mixture of experts: a literature survey. *Artificial Intelligence Review*, 42(2):275–293, 2014.

[54] Emmanuel Bengio, Pierre-Luc Bacon, Joelle Pineau, and Doina Precup. Conditional computation in neural networks for faster models. *arXiv preprint arXiv:1511.06297*, 2015.

[55] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

[56] Paul Smolensky, R Thomas McCoy, Roland Fernandez, Matthew Goldrick, and Jianfeng Gao. Neurocompositional computing: From the central paradox of cognition to a new generation of ai systems. *arXiv preprint arXiv:2205.01128*, 2022.

[57] Michael B Chang, Abhishek Gupta, Sergey Levine, and Thomas L Griffiths. Automatically composing representation transformations as a means for generalization. *arXiv preprint arXiv:1807.04640*, 2018.

[58] Anirudh Goyal, Alex Lamb, Jordan Hoffmann, Shagun Sodhani, Sergey Levine, Yoshua Bengio, and Bernhard Schölkopf. Recurrent independent mechanisms. *arXiv preprint arXiv:1909.10893*, 2019.

[59] Ishita Dasgupta, Erin Grant, and Tom Griffiths. Distinguishing rule and exemplar-based generalization in learning systems. In *International Conference on Machine Learning*, pages 4816–4830. PMLR, 2022.

[60] Alexander Atanasov, Blake Bordelon, and Cengiz Pehlevan. Neural networks as kernel learners: The silent alignment effect. *arXiv preprint arXiv:2111.00034*, 2021.

[61] A.K. Lampinen and S. Ganguli. An analytic theory of generalization dynamics and transfer learning in deep linear networks. In T. Sainath, editor, *International Conference on Learning Representations*, 2019. arXiv: 1809.10374.

[62] Lukas Braun, Clémentine Dominé, James Fitzgerald, and Andrew Saxe. Exact learning dynamics of deep linear networks with prior knowledge. *Advances in Neural Information Processing Systems*, 35:6615–6629, 2022.

[63] Irina Higgins, Loic Matthey, Arka Pal, Christopher P Burgess, Xavier Glorot, Matthew M Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. *ICLR (Poster)*, 3, 2017.

[64] Francesco Locatello, Ben Poole, Gunnar Rätsch, Bernhard Schölkopf, Olivier Bachem, and Michael Tschannen. Weakly-supervised disentanglement without compromises, 2020.

[65] Marco Fumero, Luca Cosmo, Simone Melzi, and Emanuele Rodola. Learning disentangled representations via product manifold projection. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 3530–3540. PMLR, 18–24 Jul 2021.

[66] Ricky T. Q. Chen, Xuechen Li, Roger Grosse, and David Duvenaud. Isolating sources of disentanglement in variational autoencoders, 2019.

[67] Hyunjik Kim and Andriy Mnih. Disentangling by factorising, 2019.

[68] Abhishek Kumar, Prasanna Sattigeri, and Avinash Balakrishnan. Variational inference of disentangled latent concepts from unlabeled observations, 2018.

[69] Chris Burgess and Hyunjik Kim. 3d shapes dataset. https://github.com/deepmind/3dshapes-dataset/, 2018.

[70] Christopher P Burgess, Irina Higgins, Arka Pal, Loic Matthey, Nick Watters, Guillaume Desjardins, and Alexander Lerchner. Understanding disentangling in β-vae. *arXiv preprint arXiv:1804.03599*, 2018.

[71] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings, 2010.

[72] Cian Eastwood and Christopher KI Williams. A framework for the quantitative evaluation of disentangled representations. In *6th International Conference on Learning Representations*, 2018.

[73] Sebastian Goldt, Marc Mézard, Florent Krzakala, and Lenka Zdeborová. Modeling the influence of data structure on learning in neural networks: The hidden manifold model. *Physical Review X*, 10(4):041044, 2020.

[74] Urte Adomaityte, Gabriele Sicuro, and Pierpaolo Vivo. Classification of superstatistical features in high dimensions. In *2023 Conference on Neural Information Procecessing Systems*, 2023.

[75] Federica Gerace, Diego Doimo, Stefano Sarao Mannelli, Luca Saglietti, and Alessandro Laio. How to choose the right transfer learning protocol? a qualitative analysis in a controlled set-up. *Transactions on Machine Learning Research*, 2024.

[76] Loic Matthey, Irina Higgins, Demis Hassabis, and Alexander Lerchner. dsprites: Disentanglement testing sprites dataset, 2017.

[77] Anchit Jain, Rozhin Nobahari, Aristide Baratin, and Stefano Sarao Mannelli. Bias in motion: Theoretical insights into the dynamics of bias in sgd training. *arXiv preprint arXiv:2405.18296*, 2024.

[78] Gerard Ben Arous, Reza Gheissari, and Aukosh Jagannath. High-dimensional limit theorems for sgd: Effective dynamics and critical scaling. *Advances in Neural Information Processing Systems*, 35:25349–25362, 2022.

[79] Amir H. Abdi, Purang Abolmaesumi, and Sidney Fels. Variational learning with disentanglement-pytorch. *arXiv preprint arXiv:1912.05184*, 2019.

[80] Xiaobiao Du, Haiyang Sun, Shuyun Wang, Zhuojie Wu, Hongwei Sheng, Jiaying Ying, Ming Lu, Tianqing Zhu, Kun Zhan, and Xin Yu. 3drealcar: An in-the-wild rgb-d car dataset with 360-degree views, 2024.

A Hyperbolic-Linear Dynamics

Consider a linear network performing a regression task with one hidden layer computing output $\hat{Y} = hWX$ in response to an input batch of data X, with P datapoints, and trained to minimize the quadratic loss using gradient descent:

$$L(W,h) = \sum_{i=1}^{P} \frac{1}{2}||y_i - hW\mathbf{x}_i||_2^2$$

This gives the learning rules for each layer with learning rate η as:

$$\Delta W = \eta P h^T(\Sigma^{yx} - hW\Sigma^x); \qquad \Delta h = \eta P(\Sigma^{yx} - hW\Sigma^x)W^T$$

These equations can be derived for a batch of data using the linearity of expectation, where $\Sigma^x = \mathbb{E}[XX^T]$ is the input correlation matrix and $\Sigma^{yx} = \mathbb{E}[YX^T]$ is the input-output correlation matrix, as follows:

$$\begin{aligned}
\Delta W &= \eta \frac{d}{dW} L(W,h) \\
&= \eta \frac{d}{dW} \sum_{i=1}^{P} \frac{1}{2}(Y_i - hWX_i)^T(Y_i - hWX_i) \\
&= \eta \sum_{i=1}^{P} h^T(Y_i - hWX_i)X_i^T \\
&= \eta P \frac{1}{P} \sum_{i=1}^{P} h^T(Y_i - hWX_i)X_i^T \\
&= \eta P \mathbb{E}[h^T(Y_iX_i^T - hWX_iX_i^T)] \\
&= \eta P h^T(\mathbb{E}[Y_iX_i^T] - hW\mathbb{E}[X_iX_i^T])] \\
&= \eta P h^T(\Sigma^{yx} - hW\Sigma^x)
\end{aligned}$$

$$\begin{aligned}
\Delta h &= \eta \frac{d}{dh} L(W,h) \\
&= \eta \frac{d}{dh} \sum_{i=1}^{P} \frac{1}{2}(Y_i - hWX_i)^T(Y_i - hWX_i) \\
&= \eta \sum_{i=1}^{P} (Y_i - hWX_i)(WX_i)^T \\
&= \eta P \frac{1}{P} \sum_{i=1}^{P} (Y_i - hWX_i)X_i^T W^T \\
&= \eta P \mathbb{E}[(Y_iX_i^T - hWX_iX_i^T)]W^T \\
&= \eta P(\mathbb{E}[Y_iX_i^T] - hW\mathbb{E}[X_iX_i^T])]W^T \\
&= \eta P(\Sigma^{yx} - hW\Sigma^x)W^T
\end{aligned}$$

By using a small learning rate η and taking the continuous time limit, the mean change in weights is given by:

$$\tau \frac{d}{dt} W = h^T(\Sigma^{yx} - hW\Sigma^x); \qquad \tau \frac{d}{dt} h = (\Sigma^{yx} - hW\Sigma^x)W^T$$

where $\tau = \frac{1}{P\eta}$ is the learning time constant. Here, t measures units of learning epochs. It is helpful to note that since we are using a small learning rate the full batch gradient descent and stochastic gradient descent dynamics will be the same.

[47] has shown that the learning dynamics depend on the singular value decomposition of:

$$\Sigma^{yx} = USV^T = \sum_{\alpha=1}^{r_y} \sigma_\alpha u^\alpha {v^\alpha}^T; \qquad \Sigma^{x} = VDV^T = \sum_{\alpha=1}^{r_x} \delta_\alpha u^\alpha {v^\alpha}^T$$

Here r_y and r_x denote the ranks of the matrices. To solve for the dynamics we require that Σ^{yx} and Σ^x are mutually diagonalizable such that the right singular vectors V of Σ^{yx} are also the singular vectors of Σ^x. We verify that this is true for the tasks considered in this work and assume it to be true for these derivations. We also assume that the network has at least r_y hidden neurons (the rank of Σ^{yx} which determines the number of singular values in the input-output covariance matrix) so that it can learn the desired mapping perfectly. If this is not the case then the model will learn the top n_h singular values of the input-output mapping where n_h is the number of hidden neurons [37]. To ease notation for the remainder of this section we will use n_h to denote both the number of hidden neurons and rank of Σ^{yx}. S and D then are diagonal matrices of the singular values of the input-output correlation and input correlation matrices respectfully.

We now perform a change of variables using the SVD of the dataset statistics. The purpose of this step is to decouple the complex dynamics of the weights of the network, with interacting terms, into multiple one-dimensional systems. Specifically we set:

$$h = U\overline{h}R^T; \qquad W = R\overline{W}V^T$$

where R is an arbitrary orthogonal matrix such that $R^T R = I$. Substituting this into the gradient descent update rules for the parameters above yields:

$$\begin{aligned}
\tau\frac{d}{dt}W =& h^T(\Sigma^{yx} - hW\Sigma^x)\\
\tau\frac{d}{dt}(R\overline{W}V^T) =& R\overline{h}U^T(USV^T - U\overline{h}R^T R\overline{W}V^T VDV^T)\\
\tau\frac{d}{dt}(R\overline{W}V^T) =& R\overline{h}(SV^T - \overline{hW}DV^T)\\
\tau\frac{d}{dt}\overline{W} =& \overline{h}(S - \overline{hW}D)
\end{aligned}$$

and

$$\begin{aligned}
\tau\frac{d}{dt}h =& (\Sigma^{yx} - hW\Sigma^x)W^T\\
\tau\frac{d}{dt}(U\overline{h}R^T) =& (USV^T - U\overline{h}R^T R\overline{W}V^T VDV^T)V\overline{W}R^T\\
\tau\frac{d}{dt}(U\overline{h}R^T) =& (US - U\overline{hW}D)\overline{W}R^T\\
\tau\frac{d}{dt}\overline{h} =& \overline{W}(S - \overline{hW}D)
\end{aligned}$$

Here we have used the orthogonality of the singular vectors such that $V^T V = I$ and $U^T U = I$. Importantly, all matrices in the dynamics are now diagonal and represent the decoupling of the network into the modes transmitted from input to the hidden neurons and from hidden to output neurons. In practice we do not initialize the network weights to adhere to this diagonalisation and so it is not guaranteed that the matrices will be diagonal at initialisation. However, empirically it has been found that the network singular values rapidly align to this required configuration [37, 47].

The derivative then for the full-network input-output mapping can be obtain by using the product rule:

$$\begin{aligned}
\tau\frac{d}{dt}\overline{hW} =& (\tau\frac{d}{dt}h)W + h(\tau\frac{d}{dt}W)\\
=& \left(\overline{W}(S - \overline{hW}D)\right)W + \overline{h}\left(\overline{h}(S - \overline{hW}D)\right)\\
=& \overline{W}^2(S - \overline{hW}D) + \overline{h}^2(S - \overline{hW}D)\\
=& \left(\overline{W}^2 + \overline{h}^2\right)(S - \overline{hW}D)
\end{aligned}$$

This means that at a minimum: $S - \overline{hW}D = 0$ or $\frac{S}{D\overline{W}} = \overline{h}$. This defines a hyperbolic space between $\overline{W}$ and $\overline{h}$. As a result we can use the change of variables: $\overline{W} = \sqrt{\lambda}\sinh\frac{\theta}{2}$ and $\overline{h} = \sqrt{\lambda}\cosh\frac{\theta}{2}$ parametrized by θ.

We note that there is a conserved quantity between the singular values of the weight matrices:

$$\begin{aligned}
\overline{W}^2 - \overline{h}^2 &= (\sqrt{\lambda}\sinh\frac{\theta}{2})^2 - (\sqrt{\lambda}\cosh\frac{\theta}{2})^2 \\
&= \lambda\sinh^2\frac{\theta}{2} - \lambda\cosh^2\frac{\theta}{2} \\
&= \lambda\left(\frac{cosh(\theta)+1}{2}\right) - \lambda\left(\frac{cosh(\theta)-1}{2}\right) \\
&= \frac{\lambda}{2}\cosh\theta + \frac{\lambda}{2} - \frac{\lambda}{2}\cosh\theta + \frac{\lambda}{2} \\
&= \lambda
\end{aligned}$$

This is known as λ-Balanced weights [33] and for a given initial value for λ this quantity will be conserved for all times during training. Aiming to write the network dynamics in terms of this quantity to understand its effect on learning speed and initialisation and with the change of variables to hyperbolic coordinates we begin with:

$$\begin{aligned}
\left(\overline{W}^2 + \overline{h}^2\right)^2 =&(\overline{W}^2)^2 + (\overline{h}^2)^2 \\
=&(\overline{W}^2)^2 + (\overline{h}^2)^2 + 4\overline{W}^2\overline{h}^2 - 4\overline{W}^2\overline{h}^2 \\
=&\left(\overline{W}^2 - \overline{h}^2\right)^2 + 4\overline{W}^2\overline{h}^2
\end{aligned}$$

Substituting this into the network dynamics equation and defining the network singular value as $\omega = \overline{hW}$ we obtain:

$$\begin{aligned}
\tau\frac{d}{dt}\omega &= \left(\overline{W}^2 + \overline{h}^2\right)(S - \omega D) \\
\tau\frac{d}{dt}\omega &=\sqrt{\left((\overline{W}^2 - \overline{h}^2)^2 + 4\overline{W}^2\overline{h}^2\right)}(S - \omega D)
\end{aligned}$$

Now applying the change of variables to hyperbolic coordinates with $\overline{W} = \sqrt{\lambda}\sinh\frac{\theta}{2}$ and $\overline{h} = \sqrt{\lambda}\cosh\frac{\theta}{2}$ parametrized by θ:

$$\begin{aligned}
&\tau\frac{d}{dt}(\sqrt{\lambda}\cosh\frac{\theta}{2})(\sqrt{\lambda}\sinh\frac{\theta}{2}) = \\
&\qquad\sqrt{\left((\lambda\sinh^2\frac{\theta}{2}) - (\lambda\cosh^2\frac{\theta}{2})\right)^2 + 4(\lambda\sinh^2\frac{\theta}{2})(\lambda\cosh^2\frac{\theta}{2})}(S - (\sqrt{\lambda}\cosh\frac{\theta}{2})(\sqrt{\lambda}\sinh\frac{\theta}{2})D) \\
&\tau\frac{d}{dt}\lambda\cosh\frac{\theta}{2}\sinh\frac{\theta}{2} = \sqrt{\left((\lambda\sinh^2\frac{\theta}{2}) - (\lambda\cosh^2\frac{\theta}{2})\right)^2 + 4\lambda^2(\cosh\frac{\theta}{2}\sinh\frac{\theta}{2})^2}(S - \lambda\cosh\frac{\theta}{2}\sinh\frac{\theta}{2}D)
\end{aligned}$$

We can then apply the identities: $\cosh\frac{\theta}{2}\sinh\frac{\theta}{2} = \frac{1}{2}\sinh\theta$ and $\lambda\sinh^2\frac{\theta}{2} - \lambda\cosh^2\frac{\theta}{2} = \lambda$:

$$\begin{aligned}
\tau\frac{d}{dt}\frac{\lambda}{2}\sinh(\theta) &= \sqrt{\lambda^2 + 4\lambda^2(\frac{1}{2}\sinh(\theta))^2}(S - \frac{\lambda}{2}\sinh(\theta)D) \\
\tau\frac{d}{dt}\frac{\lambda}{2}\sinh(\theta) &= \sqrt{\lambda^2 + \lambda^2\sinh^2(\theta)}(S - \frac{\lambda}{2}\sinh(\theta)D) \\
\tau\frac{d}{dt}\frac{\lambda}{2}\sinh(\theta) &= |\lambda|\sqrt{1 + \sinh^2(\theta)}(S - \frac{\lambda}{2}\sinh(\theta)D)
\end{aligned}$$

$$\tau \frac{d}{dt}\frac{\lambda}{2}\sinh(\theta) = |\lambda|\sqrt{\cosh^2(\theta)}(S - \frac{\lambda}{2}\sinh(\theta)D)$$
$$\tau \frac{d}{dt}\frac{\lambda}{2}\sinh(\theta) = |\lambda|\cosh(\theta)(S - \frac{\lambda}{2}\sinh(\theta)D)$$

Now applying the derivative on the left:

$$\tau \frac{\lambda}{2}\cosh(\theta)\frac{d}{dt}\theta = |\lambda|\cosh(\theta)(S - \frac{\lambda}{2}\sinh(\theta)D)$$
$$\frac{d}{dt}\theta = \frac{1}{\tau} sgn(\lambda)(2S - \lambda D\sinh(\theta))$$

This is a separable differential equation in θ:

$$\int_{\theta_0}^{\theta_f} \frac{1}{(2S - \lambda D\sinh(\theta))} d\theta = \int_0^t \frac{sgn(\lambda)}{\tau} dt$$

$$\left[\frac{\log\left(\left|2S\tanh\left(\frac{\theta}{2}\right) + \sqrt{4S^2+\lambda^2D^2} + \lambda D\right|\right) - \log\left(\left|2S\tanh\left(\frac{\theta}{2}\right) - \sqrt{4S^2+\lambda^2D^2} + \lambda D\right|\right)}{\sqrt{4S^2+\lambda^2D^2}}\right]_{\theta_0}^{\theta_f} = \frac{sgn(\lambda)}{\tau}t$$

$$\frac{1}{\sqrt{4S^2+\lambda^2D^2}}\left[\log\left(\frac{\left|2S\tanh\left(\frac{\theta}{2}\right) + \sqrt{4S^2+\lambda^2D^2} + \lambda D\right|}{\left|2S\tanh\left(\frac{\theta}{2}\right) - \sqrt{4S^2+\lambda^2D^2} + \lambda D\right|}\right)\right]_{\theta_0}^{\theta_f} = \frac{sgn(\lambda)}{\tau}t$$

$$\frac{1}{\sqrt{4S^2+\lambda^2D^2}}\left[\log\left(\frac{\left|2S\tanh\left(\frac{\theta_f}{2}\right) + \sqrt{4S^2+\lambda^2D^2} + \lambda D\right|}{\left|2S\tanh\left(\frac{\theta_f}{2}\right) - \sqrt{4S^2+\lambda^2D^2} + \lambda D\right|}\right) - \log\left(\frac{\left|2S\tanh\left(\frac{\theta_0}{2}\right) + \sqrt{4S^2+\lambda^2D^2} + \lambda D\right|}{\left|2S\tanh\left(\frac{\theta_0}{2}\right) - \sqrt{4S^2+\lambda^2D^2} + \lambda D\right|}\right)\right] = \frac{sgn(\lambda)}{\tau}t$$

If we let:

$$C = \frac{\left|2S\tanh\left(\frac{\theta_0}{2}\right) + \sqrt{4S^2+\lambda^2D^2} + \lambda D\right|}{\left|2S\tanh\left(\frac{\theta_0}{2}\right) - \sqrt{4S^2+\lambda^2D^2} + \lambda D\right|}; K = \sqrt{4S^2+\lambda^2D^2}$$

then:

$$\frac{1}{K}\left[\log\left(\frac{\left|2S\tanh\left(\frac{\theta_f}{2}\right) + K + \lambda D\right|}{\left|2S\tanh\left(\frac{\theta_f}{2}\right) - K + \lambda D\right|}\right) - \log(C)\right] = \frac{sgn(\lambda)}{\tau}t$$

Writing θ_f in terms of t:

$$\frac{1}{K}\left[\log\left(\frac{\left|2S\tanh\left(\frac{\theta_f}{2}\right) + K + \lambda D\right|}{\left|2S\tanh\left(\frac{\theta_f}{2}\right) - K + \lambda D\right|}\right) - \log(C)\right] = \frac{sgn(\lambda)}{\tau}t$$

$$\log\left(\frac{\left|2S\tanh\left(\frac{\theta_f}{2}\right) + K + \lambda D\right|}{\left|2S\tanh\left(\frac{\theta_f}{2}\right) - K + \lambda D\right|}\right) = \frac{sgn(\lambda)K}{\tau}t + \log(C)$$

$$\frac{\left|2S\tanh\left(\frac{\theta_f}{2}\right) + K + \lambda D\right|}{\left|2S\tanh\left(\frac{\theta_f}{2}\right) - K + \lambda D\right|} = C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)$$

$$2S\tanh\left(\frac{\theta_f}{2}\right) + K + \lambda D = C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)(K - 2S\tanh\left(\frac{\theta_f}{2}\right) - \lambda D)$$

$$2S\tanh\left(\frac{\theta_f}{2}\right) + C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)2S\tanh\left(\frac{\theta_f}{2}\right) = C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)(K - \lambda D) - K - \lambda D$$

$$2S\tanh\left(\frac{\theta_f}{2}\right)\left(1+C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)\right)=-K\left(1-C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)\right)-\lambda D\left(1+C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)\right)$$

$$\tanh\left(\frac{\theta_f}{2}\right)=\frac{-K\left(1-C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)\right)-\lambda D\left(1+C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)\right)}{2S\left(1+C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)\right)}$$

$$\theta_f=2\tanh^{-1}\left(\frac{K\left(C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)-1\right)-\lambda D\left(C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)+1\right)}{2S\left(C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)+1\right)}\right)$$

To obtain the dynamics for the singular value of a mode of the network we use:

$$\begin{aligned}\omega =&\lambda\sinh\frac{\theta}{2}\cosh\frac{\theta}{2}\\ =&\frac{\lambda}{2}\sinh\theta\\ =&\frac{\lambda}{2}\sinh\left(2\tanh^{-1}\left(\frac{K\left(C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)-1\right)-\lambda D\left(C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)+1\right)}{2S\left(C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)+1\right)}\right)\right)\end{aligned}$$

With the linear network dynamics we can now derive a network's hitting time (t^*). Let υ^* be a sufficiently small value:

$$\frac{S}{D}-\omega=\upsilon^*$$

$$\frac{S}{D}-\frac{\lambda}{2}\sinh\left(2\tanh^{-1}\left(\frac{K\left(C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)-1\right)-\lambda D\left(C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)+1\right)}{2S\left(C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)+1\right)}\right)\right)=\upsilon^*$$

$$\frac{1}{2}\sinh^{-1}\left(\frac{2S-2D\upsilon^*}{\lambda D}\right)=\tanh^{-1}\left(\frac{K\left(C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)-1\right)-\lambda D\left(C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)+1\right)}{2S\left(C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)+1\right)}\right)$$

$$\tanh\left(\frac{1}{2}\sinh^{-1}\left(\frac{2S-2D\upsilon^*}{\lambda D}\right)\right)=\frac{K\left(C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)-1\right)-\lambda D\left(C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)+1\right)}{2S\left(C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)+1\right)}$$

Let $T^*=\tanh\left(\frac{1}{2}\sinh^{-1}\left(\frac{2S-2D\upsilon^*}{\lambda D}\right)\right)$ then

$$T^*=\frac{K\left(C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)-1\right)-\lambda D\left(C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)+1\right)}{2S\left(C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)+1\right)}$$

$$2ST^*\left(C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)+1\right)=K\left(C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)-1\right)-\lambda D\left(C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)+1\right)$$

$$2ST^*C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)+2ST^*=KC\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)-K-\lambda DC\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)-\lambda D$$

$$2ST^*C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)-KC\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)+\lambda DC\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)=-2ST^*-K-\lambda D$$

$$\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)(2ST^*C-KC+\lambda DC)=-2ST^*-K-\lambda D$$

$$\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)=\frac{-2ST^*-K-\lambda D}{2ST^*C-KC+\lambda DC}$$

$$\frac{sgn(\lambda)K}{\tau}t = \log\left(\frac{-2ST^* - K - \lambda D}{2ST^*C - KC + \lambda DC}\right)$$
$$t^* = \frac{\tau}{sgn(\lambda)K}\log\left(\frac{-2ST^* - K - \lambda D}{2ST^*C - KC + \lambda DC}\right)$$
$$t^* = \frac{\tau}{sgn(\lambda)K}\log\left(\frac{K + 2ST^* + \lambda D}{KC - 2ST^*C - \lambda DC}\right)$$

578 Similarly we derive the escaping time for a mode with sufficiently small $\hat{\upsilon}$ as:

$$\omega = \hat{\upsilon}$$
$$\frac{\lambda}{2}\sinh\left(2\tanh^{-1}\left(\frac{K\left(C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right) - 1\right) - \lambda D\left(C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right) + 1\right)}{2S\left(C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right) + 1\right)}\right)\right) = \hat{\upsilon}$$
$$\frac{K\left(C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right) - 1\right) - \lambda D\left(C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right) + 1\right)}{2S\left(C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right) + 1\right)} = \tanh\left(\frac{1}{2}\sinh^{-1}\left(\frac{2\hat{\upsilon}}{\lambda}\right)\right)$$

579 Let $\hat{T} = \tanh\left(\frac{1}{2}\sinh^{-1}\left(\frac{2\hat{\upsilon}}{\lambda}\right)\right)$ then

$$\hat{T} = \frac{K\left(C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right) - 1\right) - \lambda D\left(C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right) + 1\right)}{2S\left(C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right) + 1\right)}$$
$$2S\hat{T}\left(C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right) + 1\right) = K\left(C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right) - 1\right) - \lambda D\left(C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right) + 1\right)$$
$$2S\hat{T}C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right) + 2S\hat{T} = KC\exp\left(\frac{sgn(\lambda)K}{\tau}t\right) - K - \lambda DC\exp\left(\frac{sgn(\lambda)K}{\tau}t\right) - \lambda D$$
$$2S\hat{T}C\exp\left(\frac{sgn(\lambda)K}{\tau}t\right) - KC\exp\left(\frac{sgn(\lambda)K}{\tau}t\right) + \lambda DC\exp\left(\frac{sgn(\lambda)K}{\tau}t\right) = -2S\hat{T} - K - \lambda D$$
$$\exp\left(\frac{sgn(\lambda)K}{\tau}t\right)\left(2S\hat{T}C - KC + \lambda DC\right) = -2S\hat{T} - K - \lambda D$$
$$\exp\left(\frac{sgn(\lambda)K}{\tau}t\right) = \frac{-2S\hat{T} - K - \lambda D}{2S\hat{T}C - KC + \lambda DC}$$
$$\frac{sgn(\lambda)K}{\tau}t = \log\left(\frac{-2S\hat{T} - K - \lambda D}{2S\hat{T}C - KC + \lambda DC}\right)$$
$$\hat{t} = \frac{\tau}{sgn(\lambda)K}\log\left(\frac{-2S\hat{T} - K - \lambda D}{2S\hat{T}C - KC + \lambda DC}\right)$$
$$\hat{t} = \frac{\tau}{sgn(\lambda)K}\log\left(\frac{K + 2S\hat{T} + \lambda D}{KC - 2S\hat{T}C - \lambda DC}\right)$$

580 Thus, the escaping time can be summarised as:

$$\hat{t} = \frac{\tau}{sgn(\lambda)K}\log\left(\frac{K + 2S\hat{T} + \lambda D}{KC - 2S\hat{T}C - \lambda DC}\right) \tag{13}$$

581 with the escaping time constant:

$$\hat{T} = \tanh\left(\frac{1}{2}\sinh^{-1}\left(\frac{2\hat{\upsilon}}{\lambda}\right)\right) \tag{14}$$

Similarly the hitting time is summarised as:

$$t^* = \frac{\tau}{sgn(\lambda)K} \log\left(\frac{K + 2ST^* + \lambda D}{KC - 2ST^*C - \lambda DC}\right) \tag{15}$$

with the hitting time constant:

$$T^* = \tanh\left(\frac{1}{2}\sinh^{-1}\left(\frac{2S - 2Dv^*}{\lambda D}\right)\right) \tag{16}$$

B Mean-field theory of the dynamics

As outlined in Sec.4, the key observation for the mean-field analysis is that the main properties of the learning dynamics can be expressed as functions of the order parameters–Eqs. 8. By combining these definitions with the update rules–Eqs. (6, 7)–we can derive closed-form expressions for the evolution of the order parameters, enabling us to track the key observables throughout the training process. In the high-dimensional limit ($d \to \infty$), these discrete update equations converge to ordinary differential equations (ODEs), which can be integrated either numerically or analytically in certain cases [77]. As is often the case in the statistical physics of disordered systems, this approach was first derived non-rigorously by [38] and [40], with later works laying down a mathematical foundation showing concentration of the ODEs [73, 78].

Following these prescriptions, we obtain the update equations as in [42]. Let us define the pre-activations of the student and task-t teacher given an input $\boldsymbol{x}$ from task t as

$$\lambda_i = \frac{1}{\sqrt{d}}\boldsymbol{W}_i \cdot \boldsymbol{x}, \qquad \rho_i^{(t)} = \frac{1}{\sqrt{d}}\boldsymbol{W}_{T,i}^{(t)} \cdot \boldsymbol{x}, \tag{17}$$

and denote the difference between the teacher and student predictions by $\Delta^{(t)} = \boldsymbol{h}^{(t)} \cdot \phi(\boldsymbol{\lambda}) - \boldsymbol{h}_T^{(t)} \cdot \phi(\boldsymbol{\rho})$. The corresponding ODEs for the order parameters in the limit $d \to \infty$ are given by:

$$\frac{dQ_{ik}}{d\tau} = -\eta h_i^{(t)} \langle \phi'(\lambda_i)\Delta^{(t)}\lambda_k\rangle - \eta h_k^{(t)} \langle \phi'(\lambda_k)\Delta^{(t)}\lambda_i\rangle + \eta^2 h_i^{(t)} h_k^{(t)} \langle \phi'(\lambda_i)\phi'(\lambda_k)(\Delta^{(t)})^2\rangle, \tag{18}$$

$$\frac{dR_{in}^{(t')}}{d\tau} = -\eta h_i^{(t)} \langle \phi'(\lambda_i)\Delta^{(t)}\rho_n^{(t')}\rangle, \tag{19}$$

$$\frac{dh_i^{(t)}}{d\tau} = -\eta \langle \Delta^{(t)}\phi(\lambda_i)\rangle, \tag{20}$$

where $\tau = \text{epoch}/d$ represents continuous time in the high-dimensional limit, and $t, t' \in 1, 2$ denote the task indices. The angular brackets indicate an average over the pre-activations. The pre-activations themselves are centered Gaussian random variables with covariances determined by the order parameters $\boldsymbol{Q}$, $\boldsymbol{R}^{(t)}$, and $\boldsymbol{T}$.

These averages can be computed analytically for certain activation functions. For instance, in the case of a rescaled error function introduced in the main text [38, 40], the relevant averages are given by:

$$\langle \phi(\beta)\phi(\gamma)\rangle = \frac{1}{\pi}\arcsin\left(\frac{\Sigma_{12}}{\sqrt{(1+\Sigma_{11})(1+\Sigma_{22})}}\right), \tag{21}$$

$$\langle \phi'(\zeta)\beta\phi(\gamma)\rangle = \frac{2\Sigma_{23}(1+\Sigma_{11}) - 2\Sigma_{12}\Sigma_{13}}{\sqrt{\Lambda_3}(1+\Sigma_{11})}, \tag{22}$$

$$\langle \phi'(\zeta)\phi'(\iota)\phi(\beta)\phi(\gamma)\rangle = \frac{4}{\pi^2\sqrt{\Lambda_4}}\arcsin\left(\frac{\Lambda_0}{\sqrt{\Lambda_1\Lambda_2}}\right), \tag{23}$$

where the Greek letters represent arbitrary pre-activations with covariance matrix $\boldsymbol{\Sigma}$, and the auxiliary quantities Λ_i are given by:

$$\Lambda_0 = \Lambda_4\Sigma_{34} - \Sigma_{23}\Sigma_{24}(1+\Sigma_{11}) - \Sigma_{13}\Sigma_{14}(1+\Sigma_{22}) + \Sigma_{12}\Sigma_{13}\Sigma_{24} + \Sigma_{12}\Sigma_{14}\Sigma_{23}, \tag{24}$$

$$\Lambda_1 = \Lambda_4(1+\Sigma_{33}) - \Sigma_{23}^2(1+\Sigma_{11}) - \Sigma_{13}^2(1+\Sigma_{22}) + 2\Sigma_{12}\Sigma_{13}\Sigma_{23}, \tag{25}$$

$$\Lambda_2 = \Lambda_4(1+\Sigma_{44}) - \Sigma_{24}^2(1+\Sigma_{11}) - \Sigma_{14}^2(1+\Sigma_{22}) + 2\Sigma_{12}\Sigma_{14}\Sigma_{24}, \tag{26}$$

$$\Lambda_3 = (1+\Sigma_{11})(1+\Sigma_{33}) - \Sigma_{13}^2. \tag{27}$$

These expressions provide a comprehensive analytical framework for tracking the dynamics of the student network and the evolution of specialisation across training.

C Method for Linear Network Phase Transition

D Disentenglement

We conduct our experiments using open-source frameworks [24, 79]. Specifically, we implement a beta-VAE with the "DeepGaussianLinear" architecture for the decoder and "DeepLinear" for the encoder. We modify the Xavier initialisation where the weights of the linear layers will have values sampled from $U(-a, a)$ with

$$a = \text{gain} \times \sqrt{\frac{6}{\text{fan\_in} + \text{fan\_out}}}$$

We vary the gain between 0.3 and 3 and run each experiment over 4 seeds. All network parameters are set to their default values as provided by the respective open-source frameworks. We run the experiments for 20 Epochs and 157499 iterations.

These experiment illustrate the impact of initialisation on network specialisation. Although the scope of these experiments is limited, they provide preliminary validation of our theoretical framework in more realistic contexts. We advocate for further investigation into alternative initialisation schemes with varying levels of balance. Moreover, we highlight the need for future research to extend these experiments by considering a wider variety of datasets (Car3D [80], dSprites [76],), network architectures (Conv,Linear), initialisation strategies (Gaussian Xavier Initalisation) and different metric (SAP [68, 63],) to fully explore the implications of our findings. In practice, linear networks in PyTorch are initialized using a uniform distribution, specifically:

$$\mathbf{W} \sim \mathcal{U}\left(-\sqrt{k}, \sqrt{k}\right), \quad \text{where} \quad k = \frac{1}{\text{in\_features}}$$

This initialization is equivalent to applying a small gain in our experimental setting, aligning with the weight scaling typically seen in neural network training setups. **DCI Disentanglement** [72] define three key properties of learned representations: Disentanglement, Completeness, and Informativeness. To assess these, they calculate the importance of each dimension of the representation in predicting a factor of variation. This can be done using models like Lasso or Random Forest classifiers. Disentanglement is computed by subtracting the entropy of the probability that a representation dimension predicts a factor, weighted by its relative importance. Completeness is similarly measured, focusing on how well a factor is captured by the dimensions. Informativeness is evaluated as the prediction error of the factors. We use the implementation in [24]. In this implementation, we sample 10,000 training and 5,000 test points, then use gradient-boosted trees from Scikit-learn to obtain feature importance weights. These weights form an importance matrix, with rows representing factors and columns representing dimensions. Disentanglement is calculated by normalizing the columns of this matrix, subtracting the entropy from 1 for each column, and then weighting by each dimension's relative importance.

E Additional Entropy Phase Diagrams

In Fig. 5 we showed phase diagrams of the aggregate entropy as a function of initialisation parameters, for both ReLU and sigmoidal networks. Below we show additional plots with the individual entropy terms (H_u defined over the unit activations, and H_h defined over the head weights).

F Diversity of Forgetting Curves

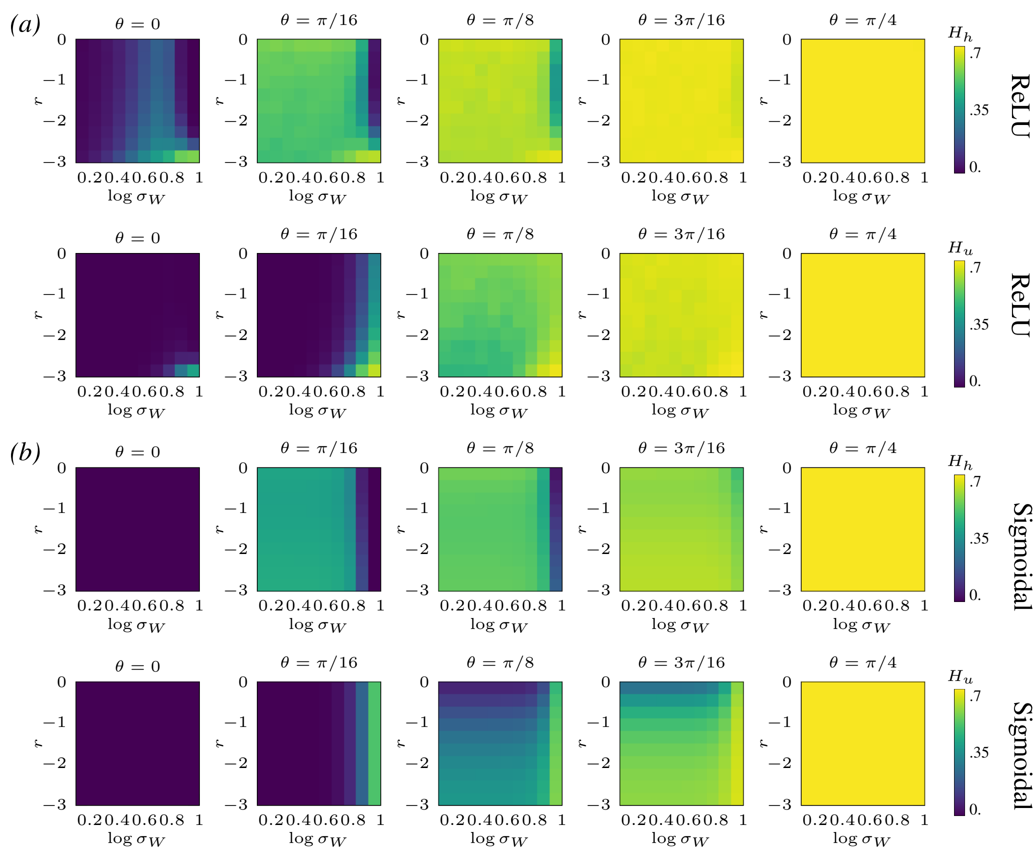

Figure 8: **Additional Phase Diagrams.** Here we show the equivalent phase diagrams from Fig. 5 for entropy measures over the unit activations and head weights.

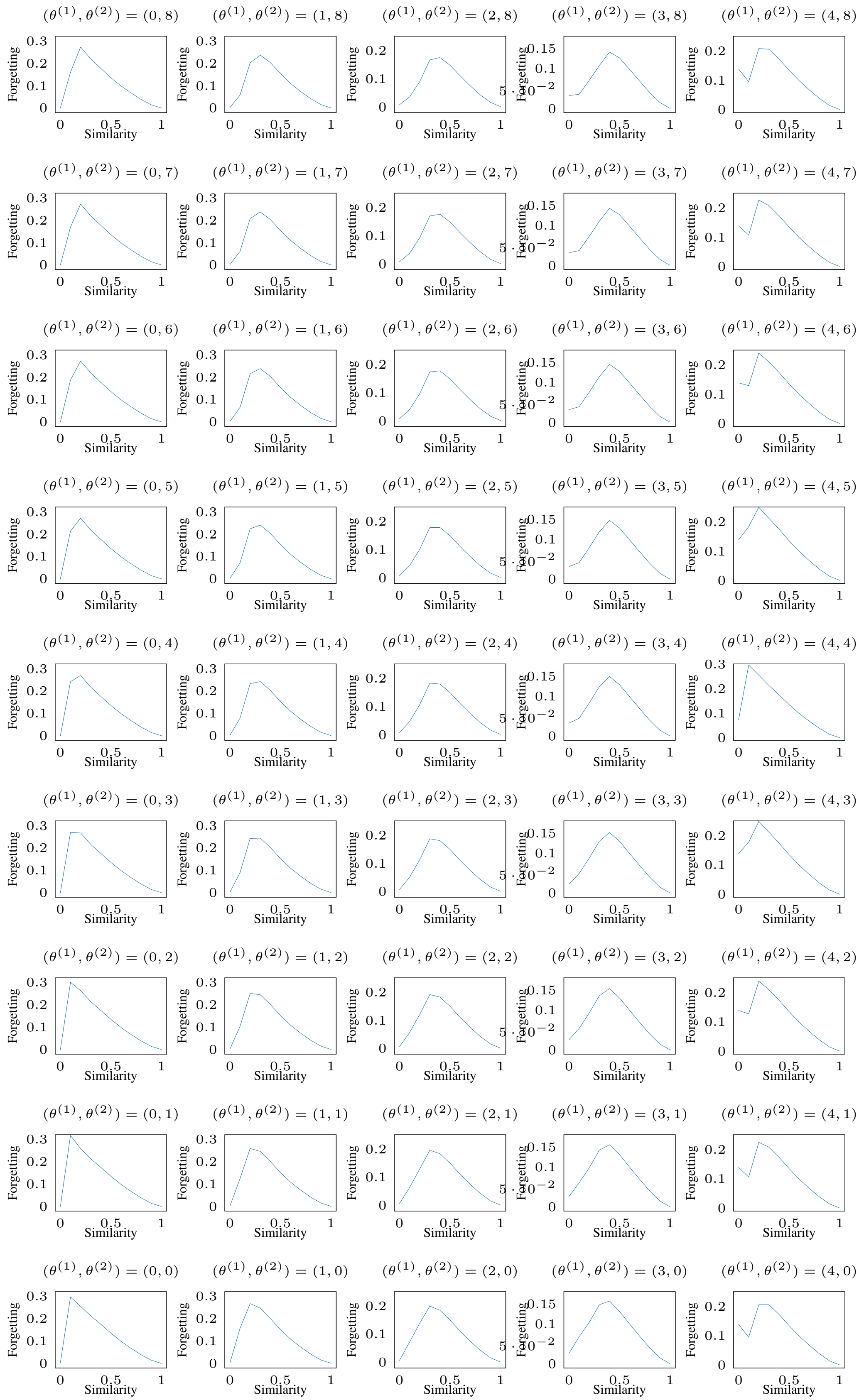

Figure 9: **Initialisation can lead to a diversity of specialisation dynamics and a diversity of relationships between forgetting and task similarity.** R, σ_W fixed, $\theta^{(1)}, \theta^{(2)}$ measured in increments of $\pi/16$. Scaled error function, $P^* = 1, P = 1$.