UNIVERSITY OF CALIFORNIA

Los Angeles

On the Structure and Learning of Perceptual Representations in Deep Neural Networks

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Electrical and Computer Engineering

by

Michael Jonathan Kleinman

2023

ABSTRACT OF THE DISSERTATION

On the Structure and Learning of Perceptual Representations in Deep Neural Networks

by

Michael Jonathan Kleinman

Doctor of Philosophy in Electrical and Computer Engineering
University of California, Los Angeles, 2023
Jonathan Chau-Yan Kao, Chair

To act and survive in an environment, humans and other organisms need to form useful representations of it. Such representations are formed through co-ordinated transformations across areas (or layers), and often change through developmental experience. We find internal representations in trained deep neural networks capture the key features of multi-area neural recordings during a perceptual decision-making task, where minimal sufficient representations of sensory information emerge along a cortical hierarchy. Using our models, we show that these minimal sufficient representations emerge through preferential propagation of task-relevant information between areas.

To better understand how such representations emerge through learning, we introduce a notion of usable information, and use it to show that a noisy learning process (e.g. Stochastic Gradient Descent) plays an important role in forming these minimal sufficient representations. We find that the learning process is highly nonlinear: semantically meaningful information is initially encoded in the representation, even if it is not needed for the task. Additionally, we show that the ability of a neural network to integrate information from diverse sources hinges critically on being exposed to properly correlated signals during the early stages of learning. In particular we find, using analytical models and through simulations, that depth and competition between sources has a significant effect on critical learning periods observed in biological and artificial networks.

We further study how multisensory information can be decomposed, and develop novel approximations to compute the redundant information shared between a set of sources about a target, and show that the common information shared between a set of sources can be used to guide the learning of meaningful representations.

The dissertation of Michael Jonathan Kleinman is approved.

Dean Buonomano

Alyson Fletcher

Stefano Soatto

Jonathan Chau-Yan Kao, Committee Chair

University of California, Los Angeles

2023

Contents

List of Figures

Acknowledgements

First, I would like to thank my advisor, Jonathan Kao, for his unwavering support and guidance during my PhD. Even as one of his first PhD students, he gave me freedom and trust to explore a variety of new and interesting research directions, being helpful and supportive the entire way.

I owe an immense deal of gratitude to Alessandro Achille, who was a mentor, friend, and an incredible collaborator for a large part of the thesis. His curiosity and ability to think clearly and creatively, identifying critical questions and simple experiments amidst complexity, somehow has me leaving every research meeting energized and inspired.

I have also been fortunate to collaborate with and learn from many other great researchers, scientists, and people. Chand Chandrasekaran was particularly supportive and helpful throughout my PhD and helped ensure that our modelling work stayed grounded to real neural recordings. I was also really fortunate to work closely and explore a variety of interesting ideas with Stefano Soatto, whose depth and clarity in thought is something I aspire to.

I would like to thank my committee, Stefano Soatto, Dean Buonomano, and Alyson Fletcher for their time, guidance, discussions, and helpful feedback over the course of my PhD. I was fortunate to get to see first-hand their remarkable depth and clarity in thought and their science, as well as their contagious curiosity. I enjoyed and benefited from many discussions I had with Dean over the course of my PhD, and was especially fortunate to see his ability to ask incisive questions and stay directly involved in the scientific details (to the point of creating simulations and analyzing raw data to better understand a paper for a weekly lab meeting presentation.)

I would also like to thank all the members of the Kao Lab, the AWS AI Labs group in Pasadena, and all the members of the ECE Student Affairs office. Last, I'd like to thank my friends and family for their support over the course of my PhD. Special thanks to HV, whose kindness and support words cannot do it justice, for making my PhD years much more special.

Vita

Education

McGill University	2012 - 2016
Bachelor of Engineering	
University of California, Los Angeles	2017 - 2018
Master of Science	

Internships

AWS AI Labs (*Mentor: Alessandro Achille*)	2022

Awards

NSERC PGSD	2018 - 2021
Amazon Fellow	2021 - 2022
UCLA Dissertation Year Fellow	2022 - 2023

Publications

Kleinman, M., Wang, T., Xiao, D., Feghhi, E., Lee, K., Carr, N., Li, Y., Hadidi, N., Chandrasekaran, C., Kao, J. A Cortical Information Bottleneck During Decision-Making. *In Submission.*

Kleinman, M., Achille, A., Soatto, S. Critical Learning Periods for Multisensory Integration in Deep Networks. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023.*

Kleinman, M., Achille, A., Soatto, S., Kao, J. [2022]. Gacs-Korner Common Information Variational Autoencoder. *Preprint. arXiv:2205.12239. Under review.*

McMahan B., **Kleinman, M.**, Kao, J. Learning rule influences recurrent network representations but not attractor structure in decision-making tasks. *Advances in Neural Information*

Processing Systems 34 (2021).

Kleinman, M., Chandrasekaran, C., Kao, J. A mechanistic multi-area recurrent network model of decision-making. *Advances in Neural Information Processing Systems 34 (2021).*

Kleinman, M., Achille, A., Soatto, S., Kao, J. Redundant Information Neural Estimation. *Entropy 2021, 23, 922*

Kleinman, M., Achille, A., Idnani, D., Kao, J. Usable information and evolution of optimal representations during training. *International Conference on Learning Representations. 2021*

Dagher, M., **Kleinman, M.**, Ng, A., Juncker, D. [2018]. Ensemble multicolour FRET model enables barcoding at extreme FRET levels. *Nature nanotechnology, 2018, 13(10), 925-932*

Chapter 1

Introduction

To act and survive in an environment, animals need to develop useful representations of the environment. The field of *systems neuroscience* is largely concerned about understanding how neural representations – or the activity of populations of neurons – relates to behavior. In the past decade, there has been progress towards understanding how the activity of populations of neurons relates to behavior, which has been facilitated by the increasing ability to record activity from populations of neurons in animal experiments. At the same time, much of the understanding of the neural population activity relies on projections of the activity to a low-dimensional space and post-hoc interpretation of activity in the low-dimensional space. There are currently few guiding principles for describing the structure of the neural population activity during a task.

In parallel, the success of deep learning relies on learning useful representations of data for performing a task, such as classification. Such representations are not handcrafted, but are instead learned through optimization. In a similar manner to systems neuroscience, deep learning researchers are interested in understanding representations so that they can ensure desirable properties, such as generalization to unseen data. Guiding principles for the structure of optimal representations for a task are emerging, where neural networks are believed to optimize an information bottleneck [136], where networks learn minimal sufficient representations of the input through implicit regularization of Stochastic Gradient Descent (SGD) [3, 123]. Moreover, the minimality of the representations is believed to relate to the ability to generalize to unseen data.

In contrast to animal experiments, deep networks are particularly convenient model systems for studying representations because the task is explicitly defined, the connections (weights) are known, and it is much easier to perform perturbation experiments on the system. It is also much easier to study the learning dynamics of these networks. An improved understanding of deep network representations and how they are learned through training will thus be beneficial not only for ensuring desirable properties of the representation in deep networks (such as generalization, interpretability, fairness, compression, etc), but for establishing a set of guiding principles for reasoning about representations in biological systems.

This thesis consists of work towards these aims. In particular, the thesis focuses on better understanding multi-area neural representations, and how their emerge through learning, by studying analogous questions in artificial deep neural networks. To reason about such high-dimensional representations, we also develop tractable approximations for information theoretic quantities.

1.1 What is a representation?

We consider a **representation** $r(X)$ to be a function of input X that is *useful* with respect to a downstream task Y. This definition is general: the inputs can be multimodal and time-varying, the function can be both deterministic or stochastic, and the task could correspond to a set of tasks (such as survival for animals). We will primarily take an information-theoretic view for defining a "useful" representation for a task, and we will show that this approach is helpful for reconciling representations observed in both animals and deep neural networks.

Our definition of a representation gives rise to several natural questions that I investigate in this thesis, in particular:

1. What makes a useful or efficient representation of sensory stimuli (for a decision)?
2. How do such representations evolve over learning/development?
3. How can we quantify the information contained in a high dimensional representation?

These questions are important in neuroscience for understanding cortical representations and for those interested in understanding the behaviour and learning of (artificial) deep neural networks. We approach these questions primarily in the context of artificial networks, since it is "fully observable" in the sense that the task is explicitly defined, all the connections (weights) are known, and we can probe the representations in response to arbitrary inputs. However, we primarily leverage experiments used in neuroscience – which are very carefully designed to probe specific information processing phenomenon – to gain insight into learned representations, and how they change during development. Additionally, using experiments from neuroscience will allow us to make connections between representations in our artificial models and in recorded neural data.

1.2 Representations in brains and artificial neural networks

The brain is remarkable in its ability to learn to leverage computations of populations of neurons to give rise to flexible behavior. The brain is an incredibly complex network consisting of billions of neurons and trillions of synapses. With current experimental approaches however, neuroscientists can only record from a tiny fraction of it during simplistic behavior, making the development of an understanding of such a complex system and its internal representations even less tractable.

Rather than study the brain directly, I have been focusing on developing an understanding of *deep neural networks*, a more tractable but also incredibly complex network of billions of units and learned weights that connect them that also conspire to produce emergent behaviour. These networks are becoming increasingly relied upon for numerous critical decisions in medical, robotics, and industrial applications, especially with the public deployment of large (language) models at scale. And yet, we have a very limited understanding for how these networks work: how they process inputs and make decisions, and how such processing and decision-making depends on the implicit biases coming from how the network was trained.

1.3 Trained neural networks as cortical models

In addition to an intrinsic need for developing an understanding of artificial neural network representations, there is an emerging field that is using optimized feedforward and recurrent neural networks to model computations associated with visual [153, 154], cognitive [99], timing [89, 110], navigation [14], and motor tasks [58, 131]. Like the brain, artificial neural networks consist of billions of simple units and learned weights that connect them. Assuming the weights are learned appropriately, the network can combine the simple units to produce complex and emergent behavior, and can perform similar tasks to behaving animals. These neural networks are typically trained using stochastic gradient descent (or a related variant), which performs many steps of local parameter updates in order to minimize a loss function.

In contrast to animal experiments, trained deep networks can be useful model systems for studying representations because they are fully observable: we have complete access to all the weights and the the representations of all units in response to arbitrary inputs. Understanding how a trained artificial neural network is implementing a task can then potentially provide insight into how populations of cortical neurons implement such tasks. Biologically plausible constraints on the connectivity can also be included in the models, such as Dale's Law [127]. Typically, trained models are compared with neural data and are used to generate hypothesis regarding underlying neural circuits, and this approach has provided insight into decision making [99] and visual processing [153, 154].

1.4 Evolution of representations over learning and development

The local parameter updates coming from stochastic gradient descent (or a similar variant) lead to parameter updates that will depend directly on the knowledge previously embedded within the weights. Since these parameter updates occur in succession, the final representation can have a strong dependency on the trajectory the network took during learning.

In humans and other animals, developmental studies typically involve studying both normal and experimentally altered development [67, 146]. We use a similar approach for studying learning in deep networks. During normal development, we are interested in how relevant and irrelevant information from the input x becomes represented in the representation $z = r(x)$ during training. Additionally, we study how learning dynamics are affected by the implicit regularization coming from training with stochastic gradient descent from the use of a small batch size and small learning rate.

Critical periods in animals and deep networks: [146] showed that humans and animals are peculiarly sensitive to changes in the distribution of sensory information early in training, in a phenomenon known as *critical periods*. Critical periods have since been described in many different species and sensory organs. For example, barn owls originally exposed to misaligned auditory and visual information cannot properly localize prey [84]. Somewhat surprisingly, similar critical periods for learning have also been observed in deep networks. [2] found that early periods of training were critical for determining the asymptotic network behavior. Additionally, it was found that the timing of regularization was important for determining asymptotic performance [47], with regularization during the initial stages of training having the most influential effect.

Outline of the thesis, and summary of contributions

We find internal representations in trained deep neural networks capture the key features of multi-area neural recordings during a perceptual decision-making task, where minimal sufficient representations of sensory information emerge along a cortical hierarchy. Using our models, we show that these minimal sufficient representations emerge through preferential propagation of task-relevant information between areas (Chapter 2). Motivated by our model, our collaborators subsequently recorded from earlier cortical areas along the sensorimotor transformation, and the cortical data similarly displayed increased task information in earlier brain areas. We argue that the general principle of a *cortical information bottleneck* during decision-making explains both our model and multi-area cortical recordings (Chapter 3).

To better understand how such representations emerge through learning, we introduce a notion of usable information, and use it to show that a noisy learning process (e.g. Stochastic Gradient Descent) plays an important role in forming these minimal sufficient representations. We find that the learning process is highly nonlinear: semantically meaningful information is initially encoded in the representation, even if it is not needed for the task (Chapter 4). Additionally, we show that the ability of a neural network to integrate information from diverse sources hinges critically on being exposed to properly correlated signals during the early stages of learning. In particular we find, using analytical models and through simulations, that depth and competition between sources has a significant effect on critical learning periods observed in biological and artificial networks (Chapter 5).

We further study how multisensory information can be decomposed, and develop novel approximations to compute the redundant information shared between a set of sources about a target (Chapter 6), and show that the common information shared between a set of sources can be used to guide the learning of meaningful representations (Chapter 7).

Chapter. 2

These results have been published as [82]:

Kleinman, M., Chandrasekaran, C., Kao, J. A mechanistic multi-area recurrent network model of decision-making. *Advances in Neural Information Processing Systems 34 (2021).*

Chapter. 3

These results are in submission as:

Kleinman, M., Wang, T., Xiao, D., Feghhi, E., Lee, K., Carr, N., Li, Y., Hadidi, N., Chandrasekaran, C., Kao, J. A Cortical Information Bottleneck During Decision-Making. *In Submission.*

Chapter. 4

These results are published as [78]:

Kleinman, M., Achille, A., Idnani, D., Kao, J. Usable information and evolution of optimal representations during training. *International Conference on Learning Representations. 2021*

Chapter. 5

These results have been published as [79]:

Kleinman, M., Achille, A., Soatto, S. Critical Learning Periods for Multisensory Integration in Deep Networks. *To Appear at CVPR 2023.*

Chapter. 6

These results have been published as [81]:

Kleinman, M., Achille, A., Soatto, S., Kao, J. Redundant Information Neural Estimation. *Entropy 2021, 23, 922*

Chapter. 7

These results have been published as [80]:

Kleinman, M., Achille, A., Soatto, S., Kao, J. [2022]. Gacs-Korner Common Information Variational Autoencoder. *Preprint. arXiv:2205.12239. Under review.*

Chapter 2

A mechanistic multi-area recurrent network model of decision-making

Recurrent neural networks (RNNs) trained on neuroscience-based tasks have been widely used as models for cortical areas performing analogous tasks. However, very few tasks involve a single cortical area, and instead require the coordination of multiple brain areas. Despite the importance of multi-area computation, there is a limited understanding of the principles underlying such computation. We propose to use multi-area RNNs with neuroscience-inspired architecture constraints to derive key features of multi-area computation. In particular, we show that incorporating multiple areas and Dale's Law is critical for biasing the networks to learn biologically plausible solutions. Additionally, we leverage the full observability of the RNNs to show that output-relevant information is preferentially propagated between areas. These results suggest that cortex uses modular computation to generate minimal sufficient representations of task information. More broadly, our results suggest that constrained multi-area RNNs can produce experimentally testable hypotheses for computations that occur within and across multiple brain areas, enabling new insights into distributed computation in neural systems.

2.1 Introduction

Traditionally, RNNs have provided insight into local computations, and there has been limited insight into multi-area computation [107, 127]. To study multi-area computation, we explicitly constrained RNNs to have multiple recurrent areas, which we refer to as multi-area RNNs. We used these multi-area RNNs to study decision-making, a cognitive process known to involve multiple areas including the prefrontal, parietal, and premotor cortex [24,30,43,48,70,99,107,112]. Multi-area RNNs enable us to investigate several questions. Most broadly, what are the roles of within-area dynamics and inter-area connections in mediating distributed computations? How does the dimensionality and dynamics of neural computation differ across areas? What role do inter-area feedforward and feedback connections play in propagating information and rejecting noise? How do intra-area dynamics and inter-area connections coordinate to solve a task?

We use multi-area RNNs to study these questions in a decision-making task where premotor cortex and upstream areas are known to perform distinct computations. We trained multi-area RNNs to perform a perceptual decision-making task (Checkerboard Task) and

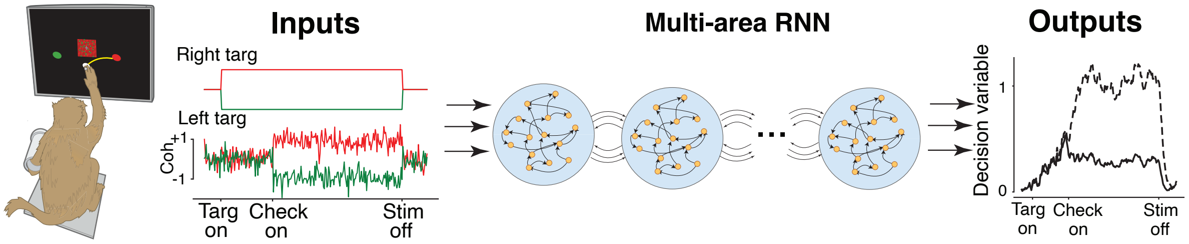

Figure 2.1: **Task.** RNN configuration. The RNN receives 4 inputs. Two inputs indicate the identity of the left and right targets, which can be red or green. These inputs are noiseless. The other two inputs indicate the value of the signed color coherence (proportional to amount of red in checkerboard) and negative signed color coherence (proportional to amount of green in checkerboard). We added independent Gaussian noise to these signals (see Appendix A.1.2). The network outputs two analog decision variables, each of which indicates evidence towards the right target (solid line) or left target (dashed line). A decision is made in the direction of whichever decision variable passes a preset threshold (0.6) first. The time at which the input passes the threshold is defined to be the reaction time.

compared their activity to monkey neuron recordings from the dorsal premotor cortex (PMd). We found that, when incorporating Dale's law and anatomically-informed levels of feedforward inhibition into training, PMd-resembling dynamics emerged in multi-area RNNs. Specifically, the multi-area RNN's output area (1) resembled PMd in single unit statistics and neural population activity, and (2) only retained the "output relevant" signals. Inter-area connections preferentially propagated these output relevant signals while attenuating output irrelevant signals. Our models and analyses provide a framework for studying distributed computations involving multiple areas in neural systems.

2.2 Motivation: Decision-making involves multiple brain areas

2.2.1 Checkerboard Task

In the "Checkerboard Task" [24,30], shown in Fig. 2.1, a monkey was first shown left and right targets whose color (red and green) was random on each trial. The monkey was subsequently shown a central static checkerboard composed of red and green squares. The monkey was trained to discriminate the dominant color of the static checkerboard and reach to the target matching the dominant color. Since the target colors were random on each trial, this task separates the reach direction decision from the color decision [42]. This task enables studying how information related to the selection of the color of the target and information related to the direction of the reach is represented.

2.2.2 PMd Data during Checkerboard Task

We analyzed the activity of neurons from the dorsal premotor cortex (PMd), an area associated with somatomotor decisions, in monkeys performing the Checkerboard Task [24]. Neural activity in PMd principally reflects the direction decision (left or right) and has

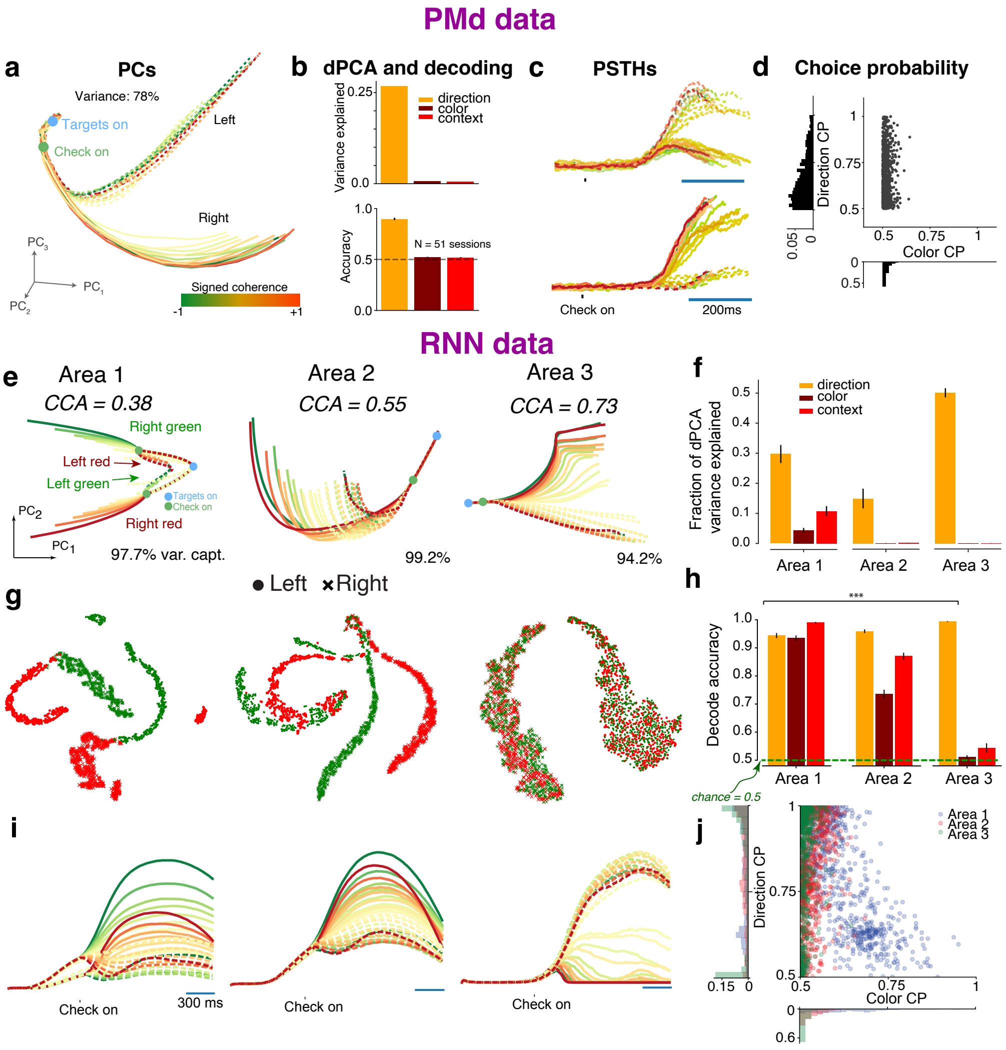

Figure 2.2: **PMd-resembling dynamics emerge in neuroscience constrained RNNs.** **(a)** PMd neural trajectories in the top 3 PCs. Color reflects signed color coherence, with darker shades of red (green) indicating more red (green) checkerboards. Right (left) reaches are denoted by solid (dotted) lines. **(b)** (Top) Variance captured by dPCA axes for the color decision, target configuration (context), and direction decision. (Bottom) Decode accuracy of the direction decision, color decision, and context in PMd sessions with U-probes and multiple neurons. **(c)** Representative PMd PSTHs aligned to checkerboard onset. **(d)** Direction and color choice probability (CP) for all recorded PMd units. **(e)** Neural trajectories in the top 2 principal components for each RNN area. **(f)** Variance captured by dPCA axes for color, context, and direction. **(g)** Non-linear tSNE embedding of peri-movement activity in each area. Each dot is a trial, with red or green denoting the color decision and '.' or 'x' denoting the direction decision. Unlike Areas 1 and 2, Area 3 only had two clusters separated based on the direction decision. **(h)** Decode accuracy of direction, color, and context in all three areas. **(i)** Example PSTHs in each area. **(j)** Choice probabilities for units in all areas (pooled over 8 RNNs).

minimal representations associated with the dominant color of the checkerboard (red or green) [24,30,141]. To summarize this phenomenon, we show the principal components (PCs) of the PMd neural population activity in Fig. 2.2a. These PC trajectories separate based on the eventual reach direction (right reaches in solid, left in dotted), but not the color (red and green largely overlapping). We identified principal axes via demixed PCA (dPCA [85]) that maximized variance related to the target configuration (context), color decision, and direction decision (see Appendix A.2.4). The direction axes captured significant variance (26.7%) while the color and context axes captured minimal variance (0.7%, 0.5%, respectively), as shown in Fig. 2.2e. It is possible, however, that there is direction-dependent color variance that is averaged away during marginalization when computing the dPCA variance [85]. Given simultaneously recorded data, a more appropriate measure of representation is the decode accuracy of direction, color and context. Across sessions where we analyzed multiple simultaneously recorded units from U-probes, the direction decision could be decoded from PMd activity significantly above chance (accuracy: 0.89, $p < 0.01$, bootstrap), but the color decision and context decode accuracy were not significantly above chance in any session (overall accuracies: 0.52 and 0.52, respectively, Fig. 2.2b, bottom).

Single neurons also had minimal color separation in individual PSTHs (e.g., Fig. 2.2c). To summarize this effect in single neurons, we computed the choice probabilities (CPs) reflecting how well the direction decision (direction CP) and color decision (color CP) could be decoded. PMd units generally had near chance color CP (0.5), but moderate to high direction CP, as shown in Fig. 2.2d. Together, these results demonstrate that PMd largely represents direction-related signals, but not the color decision or target configuration context. Since PMd activity minimally represents the color of the checkerboard or the target configuration, we reasoned that checkerboard and target inputs are transformed into a direction signal upstream of PMd and that multiple brain areas are necessary for solving this task. Brain areas, including the dorsolateral prefrontal cortex (DLPFC), and the ventrolateral prefrontal cortex (VLPFC), have been implicated in related sensorimotor transformations [10,43,61,148,152].

2.3 Multi-Area RNN Training Details

We trained RNNs of the form

$$\tau\dot{\mathbf{x}}(t) = -\mathbf{x}(t) + \mathbf{W}_{\text{rec}}\mathbf{r}(t) + \mathbf{W}_{\text{in}}\mathbf{u}(t) + \mathbf{b}_{\text{rec}} + \epsilon_t, \tag{2.1}$$

where $\mathbf{r}(t) = \text{relu}(\mathbf{x}(t))$, τ is a time-constant of the network, $\mathbf{W}_{\text{rec}} \in \mathbb{R}^{N\times N}$ defines how the artificial neurons are recurrently connected, $\mathbf{b}_{\text{rec}} \in \mathbb{R}^{N}$ defines a constant bias, $\mathbf{W}_{\text{in}} \in \mathbb{R}^{N\times N_{in}}$ maps the RNN's inputs onto each artificial neuron, and ϵ_t is the recurrent noise. The output of the network is given by a linear readout of the network rates, i.e.,

$$\mathbf{z}(t) = \mathbf{W}_{\text{out}}\mathbf{r}(t), \tag{2.2}$$

where $\mathbf{W}_{\text{out}} \in \mathbb{R}^{N_{\text{out}}\times N}$ maps the network rates onto the network outputs. For a 3-area RNN, $\mathbf{W}_{\text{rec}}$ is defined through the following block matrix

$$\mathbf{W}_{\text{rec}} = \begin{pmatrix} \mathbf{W}_{11} & \mathbf{W}_{12} & 0 \\ \mathbf{W}_{21} & \mathbf{W}_{22} & \mathbf{W}_{23} \\ 0 & \mathbf{W}_{32} & \mathbf{W}_{33} \end{pmatrix},$$

where $\mathbf{W}_{ii}$ refer to the recurrent connections of area i, and we use the convention that $\mathbf{W}_{i+1,i}$ refer to feedforward connections, and $\mathbf{W}_{1,i+1}$ refer to the feedback connections. Feedforward and feedback connections were only allowed between adjacent areas. Task inputs were defined to project onto the first area, and outputs were read out from the final area. In the rest of the text, we primarily focus on a 3-area RNN that had approximately 10% feedforward and 5% feedback connections between areas, based on projections between prefrontal and premotor cortex in a macaque atlas [100]. The network was also constrained to follow Dale's law, as in [127]. The RNN processed the target context and checkerboard inputs to output decision variables reflecting accumulated evidence for a left and right decision (Fig. 2.1). Further details are discussed in Appendix A.1.2.

2.4 Results

Because the Checkerboard Task involves multiple brain areas, we reasoned that a single-area RNN would not resemble PMd recordings. We first trained traditional single-area RNNs to perform the Checkerboard Task. We found that these RNN representations mixed color and direction information, as summarized in Appendix Fig. A.2, and therefore did not resemble PMd activity. This led us to study multi-area RNN models performing the Checkerboard Task, which turn out to accurately model PMd activity.

2.4.1 PMd-like representations emerge in optimized multi-area RNNs with neuroscience constraints

Given the anatomical and physiological evidence suggesting that multiple brain areas are implicated in the CB task, we hypothesized that the last area of an optimized multi-area RNN would more closely resemble PMd, receiving transformed direction signals computed using the checkerboard coherence and target configuration from upstream areas. We trained multi-area RNNs to perform the Checkerboard Task as described in Section 2.3.

The 3-area RNN had qualitatively different population trajectories across areas, shown in Fig. 2.2e. Area 1 had four distinct trajectory motifs corresponding to the four possible task outcomes (combinations of left vs right and red vs green decisions). PC_1 primarily varied with direction, while PC_2 varied with both the target context and red versus green checkerboards. In contrast, Area 2 and Area 3 population trajectories primarily separated on direction, not color, like in PMd. Area 3 trajectories most strongly resembled PMd trajectories (canonical correlations, $r = 0.38$, 0.55, 0.73 for Areas 1, 2, and 3; see Appendix A.2.7).

We quantified the variance captured by dPCA principal axes for the context, color, and direction axis. We found that color axis variance decreased in later areas (Area 1: 5.6%,

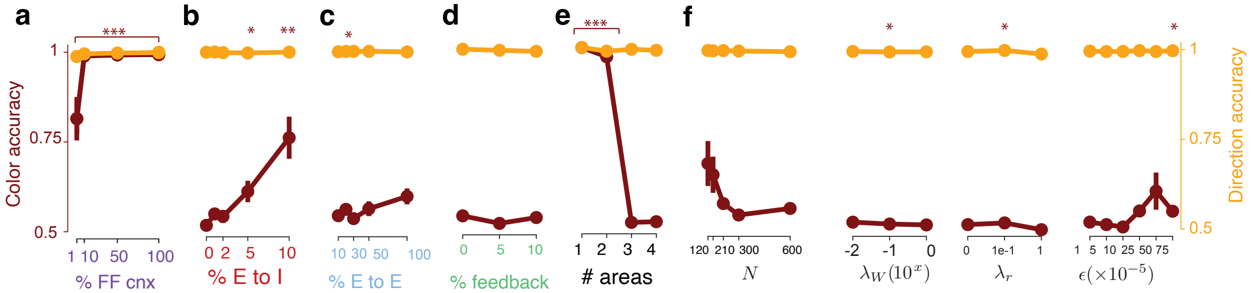

Figure 2.3: **PMd-resembling dynamics emerge in neuroscience constrained RNNs.** **(a)** We trained 3-area RNNs without explicit excitatory (E) or inhibitory (I) neurons. Inputs projected onto Area 1, and outputs were read out from Area 3. We varied the percentage of feedforward connections and computed the color and direction accuracy in Area 3. At 1% feedforward connections, color could still be significantly decoded above chance. Dots are the mean across networks and error bars are s.e.m. For significance, * is $p < 0.05$, ** is $p < 0.01$, and *** is $p < 0.001$ (with appropriate correction for multiple comparisons). We incorporated Dale's law with 80% E, 20% I neurons into subsequent sweeps, **(b)** We varied the percentage of feedforward E-to-I connections. Minimal representations with chance color decode accuracy emerged when the percentage of feedforward E to I connections was 2% or less (feedforward E to E was fixed at 10%). **(c-d)** Color information was relatively robust to feedforward E-E connections and feedback connections. **(e)** At least 3 areas were required for the RNN's last area to resemble PMd dynamics. **(f)** 3-area RNNs with neurophysiological constraints had minimal representations that were generally robust to machine learning hyperparameters. The only exceptions were when the number of units was relatively small, or the learning rate was relatively large.

Area 2: 0.13%, Area 3: 0.07%, Fig. 2.2f). In contrast, Area 3 had the largest direction axis variance (Area 1: 30.9%, Area 2: 18.2%, Area 3: 48.5%, Fig. 2.2f). An important assumption of dPCA is that the neural activity can be decomposed as a sum of terms that depend solely on particular task variables [90]. The color variance found by dPCA indicate that color, on its own, did not account for a large fraction of the overall neural variance. However, it is possible there is significant color variance within a reach direction that dPCA, a linear dimensionality reduction technique, does not capture.

As we are interested in whether the color information is contained in the representation, a more appealing measure is decode accuracy. If the color of the target can be decoded from the representation of neural activity, then color information is present in the representation. We performed nonlinear dimensionality reduction via t-distributed stochastic neighbor embedding (tSNE) [98], shown in Fig. 2.2g. These results suggest that Areas 1 and 2 contain color information, but Area 3 does not (color decisions overlap). We decoded the color decision and context (target configuration) from RNN units in each area (Fig. 2.2h, Area 1, 2, and 3 color accuracy: 0.93, 0.76, 0.51, and Area 1, 2, and 3 context accuracy: 0.99, 0.87, 0.54). Area 1 and 2 had above chance context and color decode accuracies ($p < 0.01/9$, 1-tailed t-test with Bonferroni correction), while Area 3 color and context decode accuracies were near chance, and most similar to PMd (Fig. 2.2h, color: $p = 0.05$, context: $p = 0.024$). The direction decision could be decoded significantly above chance in all areas (Fig. 2.2h, $p < 0.01/9$). We also observed that Area 3 unit PSTHs more closely resembled PMd neuron PSTHs (e.g., Fig. 2.2i), and color CP progressively decreased in later areas (Fig. 2.2j). Area 3, like PMd, had many neurons with moderate to high direction CP, but low color CP.

We tested how robust these results were to architecture and hyperparameter selection[1]. In particular, we quantified how well color could be decoded in the multi-area RNN's last area across several hyperparameter settings. We found that architecture impacted whether optimized multi-area RNNs had PMd-like minimal representations. In particular, we found that PMd-like representations emerged when we incorporated anatomical and neurophysiological constraints: Dale's law, empirical levels of feedforward inhibition, and at least 3 areas (Fig. 2.3a-e). When we varied machine learning hyperparameters, we found that our results were generally robust: multi-area RNNs had PMd-like representations in their last area over a wide range of hyperparameter settings (Fig. 2.3f). Together, this constellation of results shows that Area 3 of the multi-area RNN recapitulates key features of PMd activity, making this RNN a candidate model of multi-area decision-making in the Checkerboard Task.

In the next sections, we leverage the full observability of this biologically-plausible multi-area RNN to understand the mechanisms in different areas of the network and also how the network filters color information while propagating direction information.

[1] These sweeps over different random initializations and different parameter settings consisted of the most significant computational cost, roughly requiring 500 CPU hours on AWS, with each model training in approximately 1-2 hours.

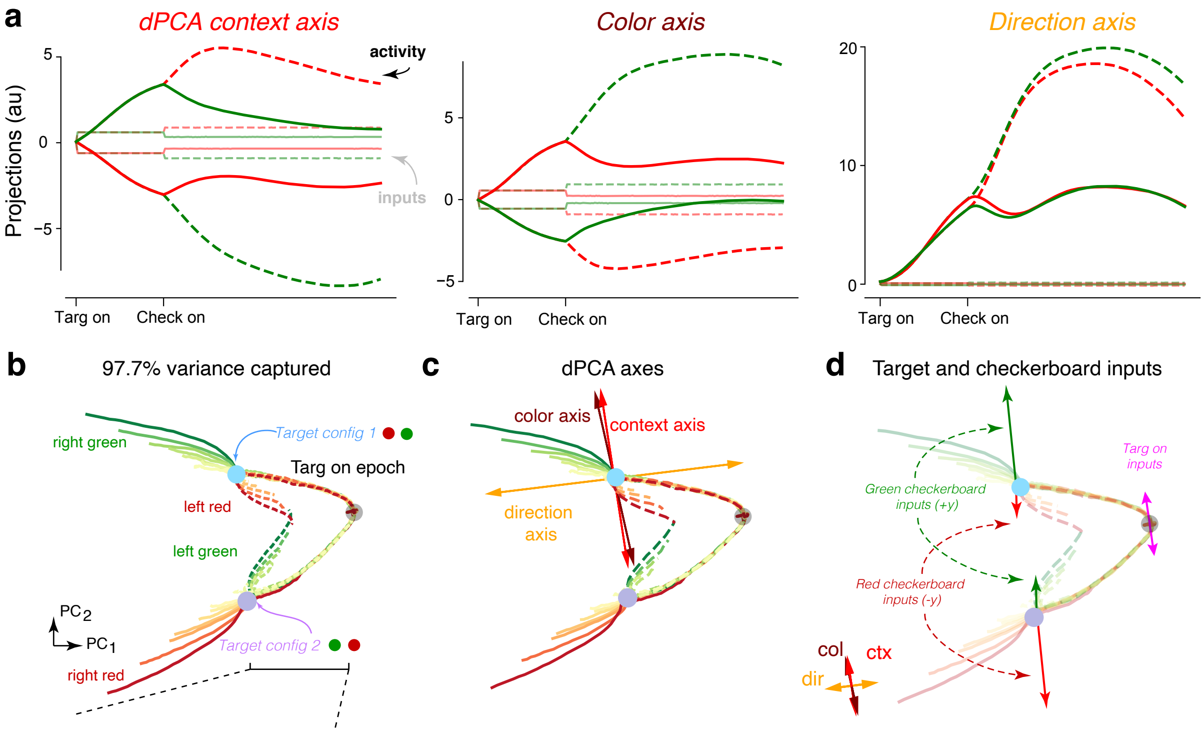

Figure 2.4: **Separation of direction and color in Area 1.** **(a)** The context, color, and direction axis correspond to the dPCA principal axes, which are not constrained to be orthogonal. Trajectories for different contexts and colors were separable on both the context and color axis. In contrast, the direction axis separated primarily on chosen direction. The RNN input representation had strong projections on the context and color axes, but not the direction axis. **(b)** Top 2 PCs of Area 1 activity, which capture 97.7% of the Area 1 variance. In the targets on epoch, the trajectories separate to two regions corresponding to the two potential target configurations (Target config 1 in blue, and Target config 2 in purple). The trajectories separate upon checkerboard color input, leading to four total trajectory motifs: right green, left red, right red, and left green. **(c)** Projection of the dPCA principal axes onto the PCs. **(d)** Projection of the context and color inputs onto the PCs. Context inputs are shown in pink, a green checkerboard input in green, and a red checkerboard input in red. Green (red) checkerboards lead to an increase (decrease) in PC_2 and the color axis, and differ in magnitude depending on the location of the trajectory in PC space. Trajectories are reduced in opacity to better visualize inputs.

2.4.2 Separation of the color and direction decision in Area 1

What are the key computational features of how the multi-area RNN represents color and direction information in the Checkerboard task? We first focused our analysis on Area 1, which uniquely has substantial variance for both color and direction decisions (Fig. 2.2h), implying a central role in computing the direction choice. We performed dPCA to identify demixed principal components related to the RNN inputs (coherence and context) and decisions (color and direction) [85]. We found demixed components that separated information related to coherence, context, the color choice, and the direction choice (Fig. 2.4b), consistent with these quantities being decodable from activity (Fig. 2.2h). We subsequently identified the context, color, and direction axes as the dPCA principal axes (unit norm, analogous to PCA eigenvectors), which combine the demixed components (analogous to PCA scores) to reconstruct neural activity [85].

We projected RNN activity and input representations onto the principal axes for context, color, and direction (Fig. 2.4a). We found that the context and color axis both responded to context and color inputs, and overall trajectories represented both context and color information. This suggests that color and context information are mixed in Area 1. In contrast, the direction axis strongly represented the direction choice, but did not strongly represent context or color (Fig. 2.4a, right). Strikingly, context and color inputs had nearly zero projection on the direction axis (Fig. 2.4a, right, opaque traces at 0). Consistent with these observations, we found the color and context axes were highly overlapping (dot product: 0.93), indicating that context and checkerboard variance are mixed in Area 1 activity. In contrast, the direction axes was closer to orthogonal to the context and color axes (overlap with color and context: 0.14 and 0.09, respectively).

These conclusions were upheld when we performed targeted dimensionality reduction (TDR), where we found (1) a direction axis separating left and right choices, with negligible input projections, and (2) that color and context representations were mixed (Appendix Fig. A.8). Further, this structure was unique for PMd-like 3-area RNNs. In single-area RNNs, dPCA identified nearly orthogonal context, color, and direction axes, with trajectories that separated almost exclusively based on context, color, and direction, respectively (Appendix Fig. A.7a).

This Area 1 representation has an important property: the direction choice is represented robustly on a nearly orthogonal axis that has close to zero context and color input projections,

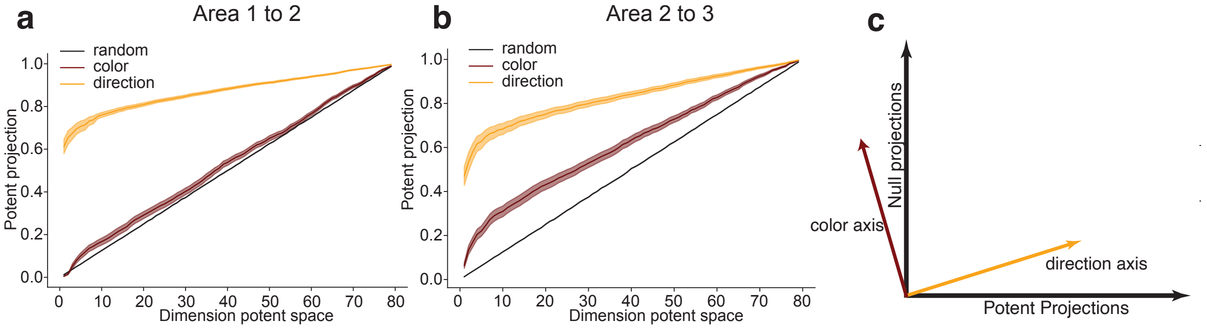

Figure 2.5: **(a)** Projections onto the potent space between Areas 1 and 2 for the color and direction axis, and a random vector as a function of effective rank for the input area to the middle area. Regardless of the dimension of the potent space, the direction axis is preferentially aligned with the potent space, indicating the information along this axis propagates, while the color axis is approximately randomly aligned. Shading indicates s.e.m. **(b)** Same as (a) but for projections between Areas 2 and 3. **(c)** Illustration depicting how the orientation of the axes affect the information that propagates.

(Fig. 2.4a). This is not trivial: as counter-examples, single-area RNNs use direction axes that have context and color input projections (Fig. A.7a), while the direction axis of an unconstrained 3-area RNN (without anatomical connectivity constraints that did not resemble PMd) has context and color information, and also receives context and color inputs (Fig. A.7b). We show the axes overlapped with the PCs in Fig.2.4, as well as the effect of the checkerboard and target inputs, which qualitatively shows that the inputs do not project onto the direction axis.

2.4.3 Inter-area connections preferentially propagate output-relevant direction information

The differentiating aspect of multi-area computation is that the different areas are separated. A natural question to ask is how then does information propagate between areas? As defined in Section 2.3, we denote the feedforward connections from Area 1 to 2 as $\mathbf{W}_{21}$, and from Area 2 to 3 as $\mathbf{W}_{32}$. We present results for feedforward connections from excitatory connections to excitatory units. Based on the hypothesis that the brain uses null and potent spaces to selectively filter information [69], we evaluated the effective potent and null spaces of $\mathbf{W}_{21}$ and $\mathbf{W}_{32}$. We defined the effective potent space to be the right singular vectors corresponding to the largest singular values (see Appendix A.2.9). The effective null space corresponded to the singular vectors with the smallest singular values.

We quantified how the color and direction axis were aligned with these potent and null spaces (see Appendix A.2.9). The projections onto the potent space are shown in Fig. 2.5a,b for $\mathbf{W}_{21}$ and $\mathbf{W}_{32}$, respectively. The null projection magnitudes are equal to one minus the potent projection. We found the direction axis was more aligned with the potent space and the color axis was more aligned with the null space. In fact, the direction axis (computed using the activity in Area 1) was consistently most aligned with the top singular vector (governed by the parameters of the feedforward matrix; which do not affect the activity in Area 1). In contrast, the color axis was similarly aligned to a random vector. This alignment was robust to the dimension of the effective potent space, and was consistent across networks with varying feedforward connectivity percentages (10%, 20%, 30%, 50%, 100%). This suggests that learning in the multi-area recurrent network involved aligning the relevant information (in the activations) with the top singular vector (governed by the learned parameters of the feedforward matrix). These results indicate that direction information is preferentially propagated to subsequent areas, while color information is not. This phenomena is schematized in Fig. 2.5c. To better understand the propagation and filtering of information in networks that had color information in the output area, we performed the same analyses on networks trained without Dale's law and 2 area networks, and found that these networks had significantly reduced alignment of the direction axis with the top singular vectors (Appendix Fig. A.11).

These results also have implications on how inter-area connections relay information between areas. Color activity has significant representation in Area 1 (see Fig. 2.2). Therefore, the inter-area connections must not merely propagate the highest variance dimensions of a preceding area [120]. Consistent with this reasoning, we found that while the top 2 PCs

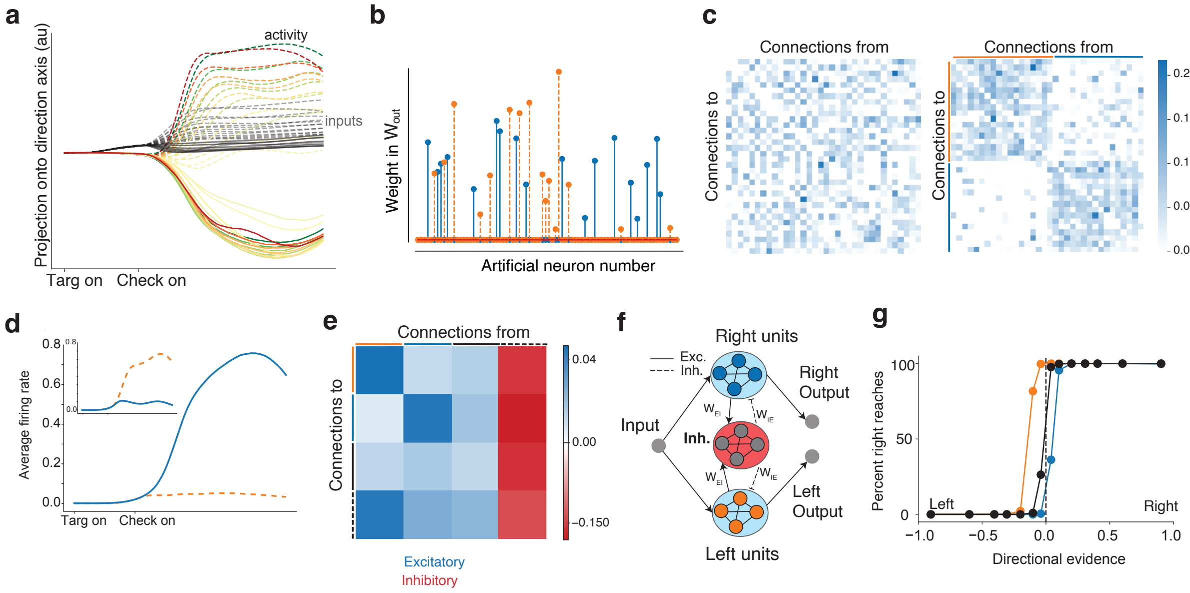

Figure 2.6: **Area 3 mechanism.** **(a)** Projection of input and overall activity onto the direction axis identified through dPCA. The conventions are the same as in Fig. 2.4. **(b)** Readout weights in $\mathbf{W}_{\text{out}}$ are sparse, with many zero entries, and selective weights for a left or right reach. **(c)** The unsorted connectivity matrix for the nonzero readout units (left panel), and the sorted connectivity matrix when the matrix was reordered based on the readout weight pools (right). **(d)** Average PSTHs from units for a leftward reach and (inset) rightwards reach. When one pool increases activity, the other pool decreases activity. **(e)** Averaged recurrent connectivity matrix. **(f)** Schematic of output area. **(g)** Psychometric curve after perturbation experiment, where 10% of inhibitory weights to the left pool (orange) and right pool (blue) were increased (doubled). Directional evidence is computed by using the signed coherence and using target configuration to identify the strength of evidence for a left reach and strength of evidence for a right reach. Increasing inhibition to the left excitatory pool leads to more right choices and vice versa.

capture 97.7% excitatory unit variance, the top 2 readout dimensions of $\mathbf{W}_{21}$ only captured 40.0% of Area 1's excitatory unit neural variance (Appendix Fig. A.10). Hence, inter-area connections are not aligned with the most variable dimensions, but are rather aligned to preferentially propagate certain types of information — a result consistent with a recent study analyzing links between activity in V1 and V2 [120].

2.4.4 Area 3, modeling PMd dynamics, is primarily input driven and implements bistable dynamics

We showed previously that Area 3 most closely resembled PMd's dynamics (Fig. 2.2). Our results suggest that a direction signal has been computed before Area 3 and is selectively propagated through the RNN's inter-area connections. We found that the input to Area 3 (through $\mathbf{W}_{32}$) is a graded direction signal that provides a directional evidence signal for left or right reaches (Fig. 2.6a). This activity must be transformed into eventual DV outputs, which are the accumulated evidence for a left or right reach. This is illustrated in Fig. 2.6a, where we plot $\mathbf{W}_{32}\mathbf{r}_t^2$ ($\mathbf{r}_t^2$ are the unit activations of Area 2), and $\mathbf{r}_t^3$.

To analyze Area 3's dynamics, we first observed that $\mathbf{W}_{\text{out}}$'s coefficients were sparse, with 44 out of 80 output weights being identically zero. We found that the readout led to two separate clusters of artificial units: units with non-zero coefficients for the left DV (orange) and those with non-zero coefficients for the right DV (blue). Artificial units projected either to the left or right DV outputs, but not both, suggesting that there are two clusters mediating left and right choices.

Based on this clustering, we sorted and visualized the connections of excitatory units of Area 3, which upon first glance generally has no discernible structure (Fig. 2.6c, left panel). After sorting, we found that two self-excitatory pools of units emerged in $\mathbf{W}_{\text{rec}}$, the first pool in Fig. 2.6c (right) corresponding to the left DV and the second pool corresponding to the right DV. In addition to these two pools, we identified a pool of randomly connected excitatory units and a pool of inhibitory units with strong projections from and to the two pools. The full Area 3 connectivity matrix is shown in Appendix Fig. A.12. This structure is consistent with a winner-take-all network, where increasing activity in one pool inhibits activity in the other pool through a separate inhibition pool (Fig. 2.6d). By taking the averaged connectivity matrix, similar to [155], we confirmed that there were two excitatory pools that received similar projections from the random excitatory pool and inhibitory pool (Fig. 2.6e). We summarize the behavior with a schematic of the area in Fig. 2.6f.

We subsequently applied selective perturbations to $\mathbf{W}_{\text{rec}}$ to determine how behavioral performance was biased. We increased inhibition to either the right or the left pool by doubling the weights of 10% of the inhibitory neurons associated with each pool. We found that this biased the network towards more left or right reaches, respectively, shown in Fig. 2.6g. When inhibition was increased to the right excitatory pool, the network was more likely to respond left. Conversely, when inhibition was increased to the left excitatory pool, the network was more likely to respond right.

Together, these results show that the output area, modeling PMd, robustly transforms separable direction inputs to a decision variable through a winner-take-all mechanism.

2.5 Discussion

Even though behavior and cognition arise from the coordinated computations of multiple brain areas, there is limited understanding of how interacting brain areas coordinate to produce cognitive behavior [86,120]. In this study, we used multi-area RNNs to gain mechanistic insight into how the brain computes a perceptual decision in the Checkerboard Task and transmits only the direction decision to PMd. These results propose hypotheses for computations that occur upstream of PMd, particularly how neural population activity representing context, color, and direction are structured, and what information is propagated between areas. We found that inter-area connections were preferentially aligned to the direction axis, not axes of maximal variance, leading to selective propagation of direction activity and attenuation of color activity. This role for inter-area connections is consistent with null and potent spaces for filtering and propagating information between areas [69, 130] and communication subspaces, which are aligned with lower variance dimensions [120].

Our results suggest that cortex and multi-area RNNs may share a more general principle of multi-area information processing: if information becomes irrelevant for later computations, it is reduced or discarded. In the Checkerboard Task, color information is necessary to compute the direction decision, but does not need to be represented after the direction decision is computed, as in PMd [24,29,30,141,152]. In deep neural networks, it is believed that minimal representations simplify the role of the output classifier [3,126]. This idea is consistent with (1) the multi-area RNN developing a minimal (little color information) but sufficient (robust direction information) representation of task inputs, and (2) Area 3, the output area, using a simple winner-take-all readout, forming two pools of neurons representing right and left decisions (Fig. 2.6).

Our analysis of the multi-area RNN leads to testable hypotheses for future experiments. First, we expect that neurons in cortical areas upstream of PMd should exhibit mixed selectivity for color and direction information, consistent with studies of dorsolateral prefrontal cortex (DLPFC) and ventrolateral prefrontal cortex (VLPFC) in cognitive tasks [44,99,111]. More specifically, our model predicts the following organization of population dynamics in these areas: neural population dynamics should diverge to two regions with slow dynamics based on target configuration, with largely overlapping context and color axes, but an orthogonal direction axis. Second, due to alignment of inter-area connections, direction axis activity in DLPFC/VLPFC should be more predictive of activity in downstream regions such as PMdr and PMd than activity in the top PCs.

Chapter 3

A Cortical Information Bottleneck During Decision-making

The brain uses multiple areas for cognition, decision-making, and action, but it is unclear why the brain distributes the computation and why cortical activity differs by brain area. Machine learning and information theory suggests that one benefit of multiple areas is that it provides an "information bottleneck" that compresses inputs into an optimal representation that is minimal and sufficient to solve the task. Combining experimental recordings from behaving animals and computational simulations, we show that later brain areas have a tendency to form such minimal sufficient representations of task inputs through preferential propagation of task-relevant information present in earlier areas. Our results thus provide insight into why the brain uses multiple brain areas for supporting decision-making and action.

3.1 Introduction

The brain uses multiple areas to perform cognitive functions and tasks, including decision-making, multisensory integration, attention, motor control, and timing [28, 43, 66, 107, 110, 120, 124, 133, 152, 156]. But why distribute computation across multiple areas? One reason is that distributed computation supports important fault tolerance in the brain which allows it to compensate when dynamics in relevant areas are altered [40,64,92,102]. But is there also a computational benefit for distributing computations across multiple areas? In deep learning for computer vision, deeper artificial neural networks (ANNs) generally perform better and exhibit hierarchical computation, a phenomenon also observed in the visual cortex, where early layers of the neural network form representations that contain low-level details (e.g., edges) and deeper layers represent higher-level concepts (e.g., object identity) [153,154]. This hierarchical computation is related to the idea of an information bottleneck: downstream areas should form representations that remove irrelevant information not necessary to do the task. For example, to know an image of a dog is a dog, we do not need to store every pixel of the image.

We unpack this more formally by first asking: what makes a representation optimal? Consider the binary classification task in Fig. 3.1a, where the task Y is to answer if an image, $\mathbf{x}$, is a dog. The data processing inequality (DPI) states that any *representation* of the original image, $\mathbf{z} = f(\mathbf{x})$, where f is a function or transformation, cannot contain more information than the image itself [31]. Rather, a representation $\mathbf{z}$ will often contain less information, so that the information between random variables $\mathbf{Z}$ and $\mathbf{X}$, denoted $I(\mathbf{Z};\mathbf{X})$, decreases through successive transformations. This is illustrated in Fig. 3.1a, where different transformations of the original image decrease the mutual information $I(\mathbf{Z};\mathbf{X})$ in the representation. But the task information contained in the representation, $I(\mathbf{Z};Y)$, is similar, since all images can be used to correctly answer "is this a dog?" (Fig. 3.1b).

The information bottleneck principle defines an optimal representation to be one that retains only the relevant or useful information for solving a task [136]. When a representation that is *sufficient* for solving the task (mathematically, $I(\mathbf{Z};Y) \approx I(\mathbf{X};Y)$) is also *minimal* (mathematically, $I(\mathbf{Z};\mathbf{X})$ as small as possible), machine learning theory proves these representations are the most robust to nuisances (such as the color of the dog or the background) [3]. In general, multi-area computation increases the minimality of a representation due to the data processing inequality: so long as these representations maintain the information to solve the task and are trainable, the *deep* multi-area computation produces more optimal representations for solving tasks. Although these representations contain less information than the input, they are often more useful and robust representations for solving the task [3, 78, 151]. Typically, these neural networks are not explicitly trained to optimize an information bottleneck objective [136]. However, recent studies [3, 78] showed that training ANNs with stochastic gradient descent implicitly minimizes an information bottleneck objective that results in minimal sufficient representations.

We unpack this more formally by first asking: what makes a representation optimal? Consider the binary classification task in Fig. 3.1a, where the task Y is to answer if an image, $\mathbf{x}$, is a dog. The data processing inequality (DPI) states that any *representation* of the original image, $\mathbf{z} = f(\mathbf{x})$, where f is a function or transformation, cannot contain more information than the image itself [31]. Rather, a representation $\mathbf{z}$ will often contain less information, so that the information between random variables $\mathbf{Z}$ and $\mathbf{X}$, denoted $I(\mathbf{Z}; \mathbf{X})$, decreases through successive transformations. This is illustrated in Fig. 3.1a, where different transformations of the original image decrease the mutual information $I(\mathbf{Z}; \mathbf{X})$ in the representation. But the task information contained in the representation, $I(\mathbf{Z}; Y)$, is similar, since all images can be used to correctly answer "is this a dog?" (Fig. 3.1b).

The information bottleneck principle defines an optimal representation to be one that retains only the relevant or useful information for solving a task [136]. When a representation that is *sufficient* for solving the task (mathematically, $I(\mathbf{Z}; Y) \approx I(\mathbf{X}; Y)$) is also *minimal* (mathematically, $I(\mathbf{Z}; \mathbf{X})$ as small as possible), machine learning theory proves these representations are the most robust to nuisances (such as the color of the dog or the background) [3]. In general, multi-area computation increases the minimality of a representation due to the data processing inequality: so long as these representations maintain the

information to solve the task and are trainable, the *deep* multi-area computation produces more optimal representations for solving tasks. Although these representations contain less information than the input, they are often more useful and robust representations for

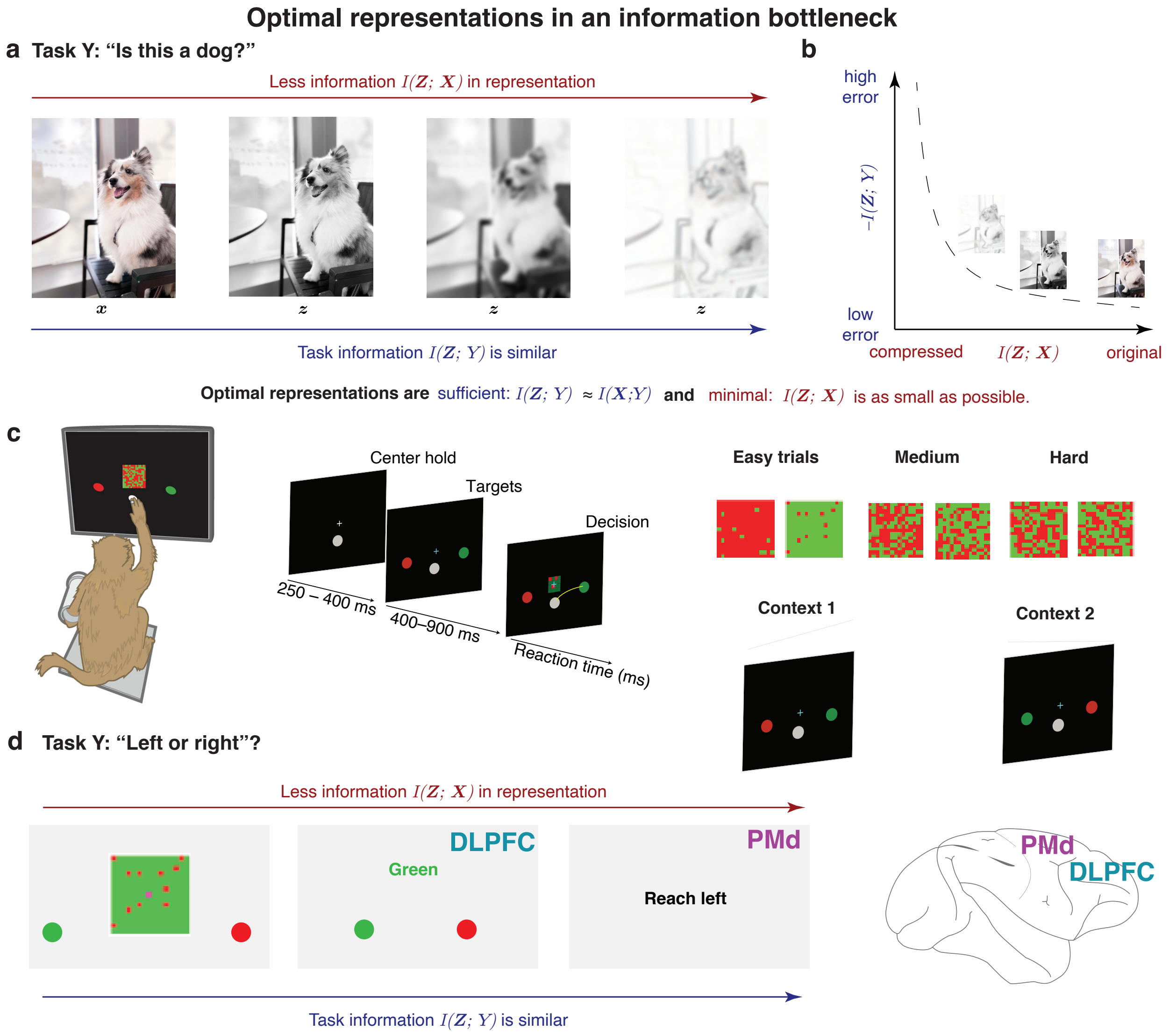
Optimal representations in an information bottleneck
a Task Y: "Is this a dog?"
Less information $I(Z;\, X)$ in representation
x
z
z
z
Task information $I(Z;\, Y)$ is similar
b
high error
$-I(Z;\, Y)$
low error
compressed
$I(Z;\, X)$
original
Optimal representations are sufficient: $I(Z;\, Y) \approx I(X;Y)$ and minimal: $I(Z;\, X)$ is as small as possible.
c
Center hold
Targets
Decision
250 – 400 ms
400–900 ms
Reaction time (ms)
Easy trials
Medium
Hard
Context 1
Context 2
d Task Y: "Left or right"?
Less information $I(Z;\, X)$ in representation
DLPFC
Green
PMd
Reach left
PMd
DLPFC
Task information $I(Z;\, Y)$ is similar

Figure 3.1: **Optimal representations are formed through an information bottleneck.** **(a)** Consider the task Y of discerning whether the image is of a dog. Images to the right have less information than the original image ($I(\mathbf{Z};\mathbf{X})$ is smaller) but still contain approximately the same amount of information, $I(\mathbf{Z};Y)$ to perform the task: "is this a dog?" **(b)** The information bottleneck trades off minimality, $I(\mathbf{Z};\mathbf{X})$ as small as possible, with sufficiency, $I(\mathbf{Z};Y) \approx I(\mathbf{X};Y)$. **(c)** Checkerboard task. The monkey reaches to the target whose color matches the checkerboard dominant color. Because there are two equally likely contexts where the color of the left and right targets are swapped, this task dissociates the color and direction choice. **(d)** A minimal sufficient representation of this task is to only retain the direction decision, in this case, reach left. A cortical information bottleneck should therefore only find direction information in motor output areas.

solving the task [3, 78, 151]. Typically, these neural networks are not explicitly trained to optimize an information bottleneck objective [136]. However, recent studies [3, 78] showed that training ANNs with stochastic gradient descent implicitly minimizes an information bottleneck objective that results in minimal sufficient representations.

We hypothesized multiple areas in the brain provides a similar *computational benefit* by forming minimal sufficient representations of the task inputs, thus implementing a cortical information bottleneck. We tested this hypothesis by combining electrophysiological recordings in behaving monkeys, and modeling using recurrent neural networks. We recorded from the DLPFC and PMd as monkeys performed a decision-making task called the Checkerboard Task (Fig. 3.1c). In this task, the monkey discriminated the dominant color of a checkerboard composed of red and green squares and reached to a target matching the dominant color. Because the red and green target locations were randomly assigned to be left or right on each trial ("target configuration", which we also term "context"), the direction decision is independent of the color decision. That is, a green color decision is equally likely to correspond to a left or right decision.

The animal's behavioral report was either a right or left reach, determined after combining the sensory evidence with the target configuration (Fig. 3.1d). While color is initially needed to solve the task, the minimal sufficient representation of the task to generate the correct output is a representation of only the direction decision without the color decision or the target configuration (aka "context"). A cortical information bottleneck would therefore predict that upstream areas should contain information about the task inputs and decision-making process, including the target configuration, perceived dominant color of the checkerboard, and direction choice, while downstream areas should only contain the direction choice (in Fig. 3.1d, "reach left").

Consistent with the predictions of the information bottleneck principle, we found that DLPFC has information about the color, target configuration, and direction. In contrast, PMd had little to no information about target configuration and color, but strongly represented the direction choice. PMd therefore had a *minimal and sufficient representation* of direction. We then trained a multi-area RNN to perform this task. We found that the RNN faithfully reproduced DLPFC and PMd activity, enabling us to propose a mechanism for how cortex uses multiple areas to compute a minimal sufficient representation.

3.2 Results

The checkerboard task involves multiple brain areas

We used linear multi-contact electrodes (U and V-probes) to record from the DLPFC (2819 single neurons and multiunits) and PMd (996 single neurons and multi units) as the monkeys performed the Checkerboard Task ((Fig. 3.1c, and also described in Sect. 2.2.1). Sample peri-stimulus time histograms (PSTHs) for neurons in DLPFC and PMd are shown in Fig. 3.2c, d, respectively, where solid (dotted) lines correspond to left (right) reaches and color (red or green) denotes the color decision. DLPFC PSTHs in Fig. 3.2c separate based on direction choice, target configuration (context), and color choice, whereas PMd PSTHs primarily separate based on the direction choice, and only very modestly with target configuration or color.

Together, these examples demonstrate that DLPFC and PMd single units exhibit activity reflecting the decision-making process, implicating multiple brain areas in decision-making. Further, DLPFC likely contains multiple task-relevant signals signals, whereas PMd contains only direction choice related signals necessary for the behavioral report in the task. In the next sections, we use dimensionality reduction, decoding, and information theory to quantify the extent of color, target configuration, and direction representations in DLPFC and PMd at the population level and show that these physiological observations are consistent with the information bottleneck principle. We then use recurrent neural network models to build a mechanistic hypothesis for how an information bottleneck could be implemented.

Evidence for a cortical information bottleneck between DLPFC and PMd

Our single neuron examples suggest that neuronal responses in DLPFC are modulated by color choice and target context, but PMd neurons generally are not. Our hypothesis is that these cortical representations in DLPFC and PMd are consistent with the information

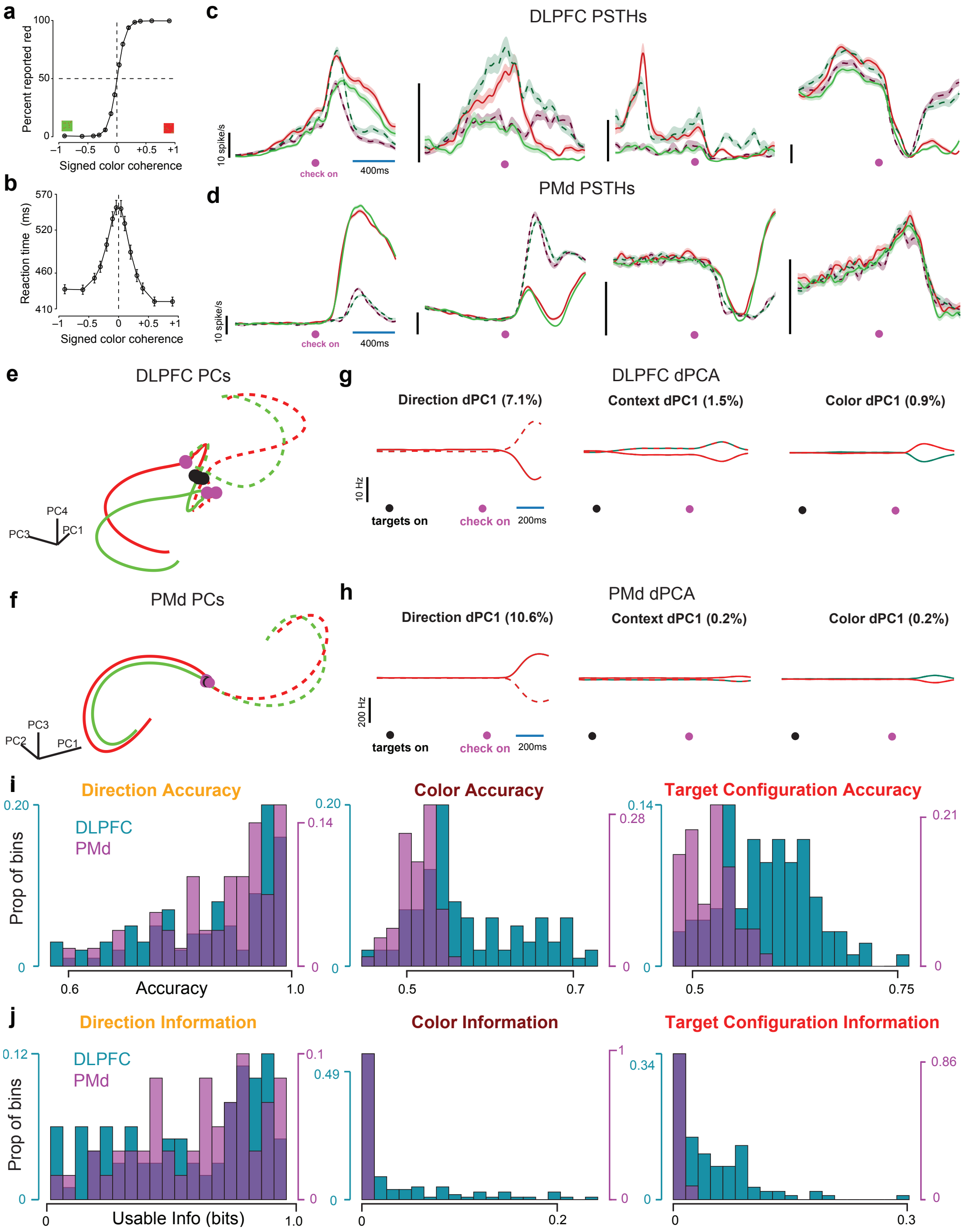

Figure 3.2: **DLPFC and PMd recordings during the checkerboard task.** **(a)** Psychometric and **(b)** reaction time curves. **(c)** Example DLPFC and **(d)** PMd PSTHs. Red and green traces correspond to red and green color choices, respectively. Dotted and solid traces correspond to right and left direction choices, respectively. **(e)** DLPFC and **(f)** PMd PCs. **(g)** DLPFC and **(h)** PMd dPCA for direction, context, and color. **(i)** Histogram (across sessions) of direction, color, and target configuration ("context") decode accuracy and **(j)** usable information for DLPFC and PMd.

bottleneck principle. The direct prediction of this hypothesis is that the PMd population activity should contain a minimal and sufficient representation of the behaviorally relevant output – the direction choice – while upstream DLPFC population activity should represent multiple task-relevant variables.

To study this at the population level, we performed principal components analysis (PCA) on DLPFC and PMd neural population activity. In these PCA trajectories, we subtracted the condition-independent component of the signal to better highlight representations of direction, target configuration, and color. DLPFC and PMd exhibited qualitatively different neural population trajectories (Fig. 3.2e,f). DLPFC population activity converged to two distinct locations in state space based on the two target configurations (green left and red right, or red left and green right). At the time of checkerboard onset (purple dots), DLPFC activity then separated into four distinct trajectories based on the four possible color × direction outcomes (green left, green right, red left, red right). In contrast, PMd trajectories in Fig. 3.2f did not exhibit target-configuration-specific steady state responses. Thus, at the point of checkerboard onset (purple dots), trajectories overlapped in the top 3 principal components, and only separated based on the direction, but not color, choice.

To quantify these these differences, we performed demixed principal component analysis (dPCA) on the DLPFC and PMd population activity (Fig. 3.2g,h). DLPFC and PMd activity both exhibited strong condition independent activity (82% and 86% variance, respectively). DLPFC activity represented the target configuration, but PMd did not (Fig. 3.2g,h, context dPC). dPCA also identified principal axes that maximized variance related to the direction choice, color choice, and target configuration. In DLPFC, the top direction choice, color choice, and target configuration axes captured 7.1%, 1.5%, and 0.9% of the population activity. In PMd, these values were 10.6%, 0.2%, and 0.2%. Across all direction, color, and target configuration axes, the dPCA variance captured for DLPFC was 11%, 3%, 5%, while for PMd it was 12%, 1%, 1%. This dPCA analysis provides further evidence that DLPFC represents direction, target configuration, and color while PMd has nearly minimal representations of color and target configuration, consistent with the infromation bottleneck principle.

Our dPCA results with trial-averaged firing rates suggest that axes associated with color and target configuration in PMd have very little variance associated with them. However, these results do not rule out the possibility that there is decodable information about these task-related variables on single trials. We performed two other analyses to assess if there is an information bottleneck and that PMd contains a minimal sufficient representation. We calculated the decode accuracy and an estimate of mutual information for direction, color, and target configuration in DLPFC and PMd population activity. We decoded direction, color, and target configuration from DLPFC and PMd sessions where we recorded a small population of neurons using a support vector machine (see Methods). To estimate mutual information, we quantified the Usable Information, a variational lower bound to mutual information that can be computed on high-dimensional data through estimating cross-entropy loss [78, 151] (see Methods).

Across 102 sessions in DLPFC, we found that direction, color, and target configuration could all be reliably decoded above chance (mean across sessions with above chance accuracy: direction 86%, target configuration 60%, color 59%, $p < 0.01$, bootstrap, for details on decoding see Methods). Histograms of decode accuracy are shown in Fig. 3.2i. These histograms reveal that most recording sessions could reliably decode direction, while target configuration(“context”), and to a lesser degree, color, could be reliably decoded in a subset of sessions. In contrast, PMd rarely exhibited sessions where target configuration and color could be reliably decoded above chance (mean accuracy across sessions: direction 88%, target configuration 53%, color 54%, only direction above shuffled decoding accuracy, $p < 0.01$), as shown in Fig. 3.2i. The differences in mean decoding accuracy in DLPFC and PMd were significant for only color and target configuration ($p < 0.001$, ranksum test), but not direction ($p = 0.208$, ranksum test).

We also quantified usable information for all DLPFC and PMd sessions, shown in Fig. 3.2j. DLPFC had sessions with non-zero usable information for direction, color, and target configuration (average direction information: 0.56 bits, context: 0.05 bits, color: 0.05 bits) while PMd only had non-zero usable information for direction (average direction information:

0.63 bits, context: 0.003 bits, color: 0.008-bits). The differences in usable information in DLPFC and PMd were also significant for only color and target configuration ($p < 0.001$, ranksum test), but not direction ($p = 0.158$, ranksum test). Together, these results indicate that PMd had a more minimal representation of task inputs, particularly color and target configuration, than DLPFC.

Our dPCA and decoding results are strongly consistent with the existence of a cortical information bottleneck between DLPFC and PMd that reduces the amount of target configuration and color information in PMd while preserving the direction choice information necessary to solve the task. We next sought to model this multi-area information bottleneck and develop a mechanistic hypothesis for how this cortical information bottleneck could be computationally implemented.

A multi-area recurrent neural network model of DLPFC and PMd

To develop a mechanistic hypothesis for this cortical information bottleneck, we studied our previously reported multi-area RNN to perform the Checkerboard task ([82], described in Chapter 2). We chose to use this multi-area RNN because prior work demonstrated this RNN, like our PMd data, has a minimal color representation in Area 3. The RNN had 3 areas, obeyed Dale's law [127], and had approximately 10% feedforward and 5% feedback connections between areas based on projections between prefrontal and premotor cortex in a macaque atlas [100]. RNN psychometric and RT curves for the multi-area RNN exhibited similar behavior to monkeys performing this task (Fig. 3.3b,c; across several RNNs, see Fig. B.1).

Although the multi-area RNN was not regularized to reproduce DLPFC and PMd activity, activity in Area 1 resembled neural responses in DLPFC and Area 3 resembled PMd (Fig. 3.3d,e). Like DLPFC, Area 1 had four distinct trajectories corresponding to the four possible task outcomes and represented context, direction choice, and color choice (Fig. 3.3d and see Fig. B.2). In contrast, Area 3 population trajectories primarily separated based on

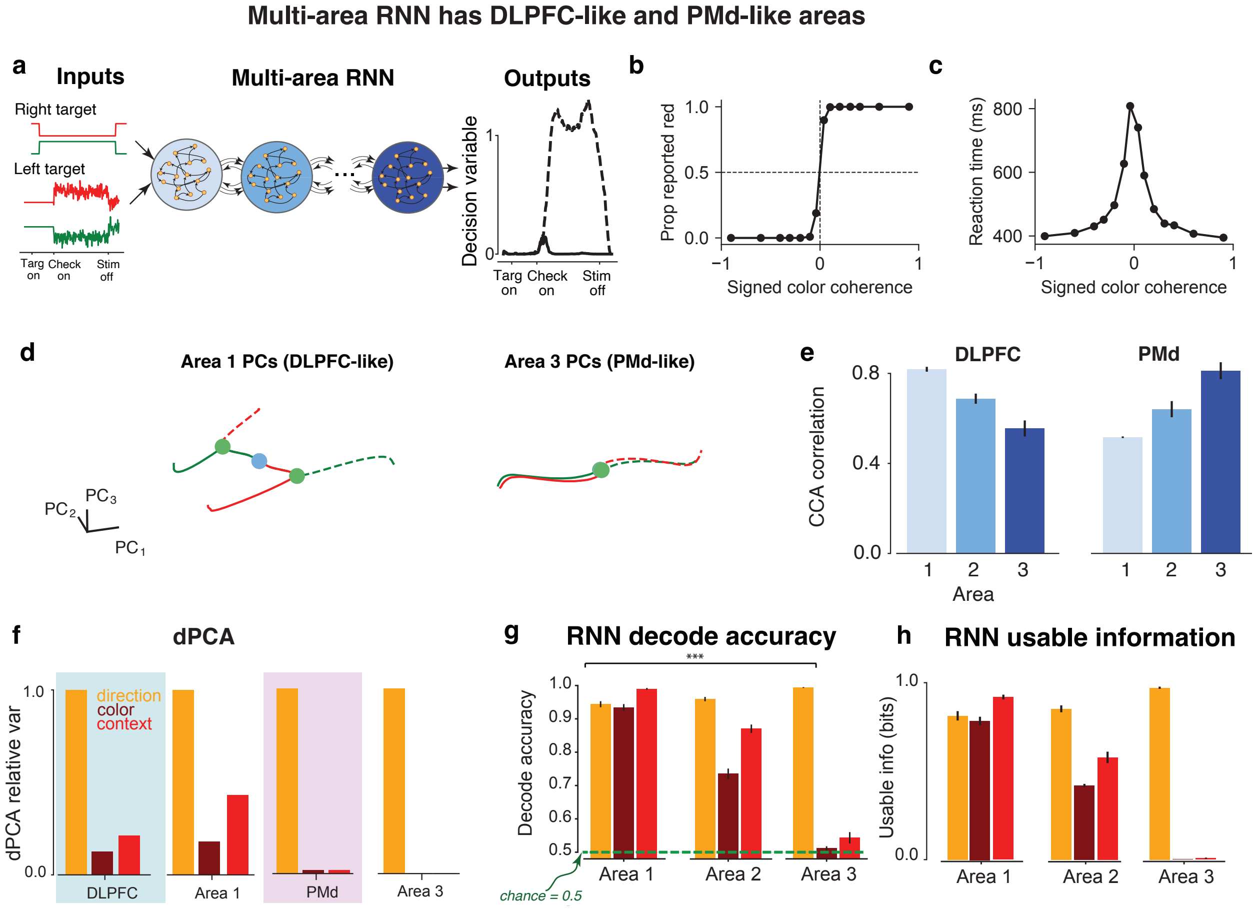

Figure 3.3: **RNN modeling of the CB task.** **(a)** Multi-area RNN configuration. The RNN received 4 inputs. The first two inputs indicated the identity of the left and right targets, which was red or green. These inputs were noiseless. The last two inputs indicated the value of the signed color coherence (proportional to amount of red in checkerboard) and negative signed color coherence (proportional to amount of green in checkerboard). We added independent Gaussian noise to these signals (see Methods). The network outputted two analog decision variables indicating evidence towards the right target (solid line) or left target (dashed line). A decision was made in the direction of whichever decision variable passed a preset threshold (0.6) first. The time at which the decision variable passed the threshold was defined to be the reaction time. **(b,c)** Psychometric and reaction time curves for exemplar multi-area RNN. **(d)** Area 1 and Area 3 principal components for exemplar RNN. **(e)** CCA correlation between each area and DLPFC principal components (left) and PMd principal components (right). DLPFC activity most strongly resembles Area 1, while PMd activity most strongly resembles Area 3. See also Fig. B.3where we computed CCA as a function of the number of dimensions. **(f)** Relative dPCA variance captured by the direction, color, and context axes. Normalization makes direction variance equal to 1. Area 1 (3) variances more closely resemble DLPFC (PMd). **(g)** Area 1 has significantly higher decoding accuracies and **(h)** usable information compared to Area 3, consistent with DLPFC and PMd. Large variance in recordings due to across-session variance.

direction and not by target configuration or color — remarkably similar to the trajectories observed in PMd.

We performed CCA to assess the similarity between the empirical neural trajectories to each RNN area's neural trajectories (see Methods). The CCA analysis suggested that Area 1 exhibited the strongest resemblance to DLPFC, while Area 3 most strongly resembled PMd activity (Fig. 3.3e). These results show that a multi-area RNN reproduced similar behavior to the monkey, and further that it did so with architecturally and qualitatively distinct areas that strongly resembled the physically distinct DLPFC and PMd cortical areas.

The RNN activity differs from cortical activity in two important ways. First, RNNs generally had a significantly smaller variance condition-independent signal (46.7% and 49.4% average variance in Area 1, and 3, respectively) than in DLPFC and PMd (82% and 86% variance, respectively). One possible explanation is that condition-independent variance in PMd is associated with a trigger signal, likely from the thalamus [71], and these RNNs do not output arm kinematics, forces, or electromyography (EMG). Similarly, in DLPFC, we did not explicitly model the target and checkerboard inputs to have large onset signals that are often associated with visual stimulation. This significant condition independent variance in the neurophysiological data may therefore make decoding more difficult since there is relatively lower variance representing direction, color, or target configuration. While our CCA analysis was performed with the condition-independent signal removed, this difference impacts both dPCA and decoding results. In general, we found that RNN exhibited trends observed in the neurophysiological data more strongly, including more variance captured for direction, color, and target configuration, as well as higher decoding accuracies. We therefore compared the relative, rather than absolute, trends in RNN activity and DLPFC for dPCA and decoding analyses for the purposes of identifying a RNN information bottleneck similar to the cortical information bottleneck.

We found that the 3-area RNN exhibited similar trends to DLPFC and PMd activity in dPCA variance and decoding accuracy. When comparing only the top axis for direction, color, and context, DLPFC activity had relatively large variance captured along the direction axis (7.1% variance captured), followed by relatively weaker representations for target configuration (1.5%) and color (0.9%). Area 1 activity had similar relative trends, with the direction axis explaining 30.9% variance, followed by target configuration (13.3%) and color (5.6%). In

Fig. 3.3f, we show the similarity in relative trends in DLPFC and Area 1 by normalizing these quantities by the variance captured by the direction axis. Meanwhile, PMd activity exhibited more direction-related variance than DLPFC (10.6% variance captured) and did not exhibit a significant target configuration and color axis representation (0.2% for both axes). Likewise, Area 3 had a stronger representation of direction (48.5% variance captured) than in Area 1, but negligible target configuration and color axes variance (0.1% for both axes), demonstrating the same relative trend (Fig. 3.3f). These results show that Area 1 more strongly resembles DLPFC and Area 3 more strongly resembles PMd in relative dPCA variance.

We next evaluated the decode accuracy and usable information in the multi-area RNN. We found Area 1, like DLPFC, had significant information for direction, target configuration, and color. Direction, color, and target configuration could be decoded from Area 1 population activity at accuracies of 94.4%, 93.4%, and 99.0%, respectively, corresponding to 0.81, 0.79, and 0.92 bits of usable information (Fig. 3.3g,h). In contrast, Area 3 direction decode accuracy was 99.4%, while color and target configuration accuracy were significantly lower (51.1% and 54.3%, respectively). This corresponded to 0.97, 0.0023, and 0.0078 bits of usable information for direction, target configuration, and color, respectively.

Together, these results show that our multi-area RNN exhibited distinct areas that resembled DLPFC and PMd activity, and also implemented an information bottleneck so that its output area only had primarily direction information and less color and target configuration information. This multi-area RNN therefore implements a candidate mechanism for how a cortical information bottleneck could be implemented between DLPFC and PMd.

3.2.1 Mechanistic features of the DLPFC and PMd bottleneck: partial orthogonalization and selective propagation

The multi-area RNN contains representations consistent with the information bottleneck principle — its input and output areas resemble DLPFC and PMd. The RNN therefore models many aspects of our physiological data and is therefore a candidate system to understand how multiple areas could lead to the empirically observed minimal sufficient representations. The unique advantage of the RNN is that we know the firing rates in each area as well as the within and inter-areal connections, which allows us to investigate how the multi-area RNN deemphasizes color information through an IB. We reasoned that such an information bottleneck could be implemented in three ways: Color information may be (1) primarily attenuated through *recurrent neural dynamics*, (2) primarily attenuated through *inter-areal connections*, or (3) attenuated through a combination of recurrent dynamics and inter-areal connections. This is illustrated in Fig. 3.4a.

To test these hypotheses, we first quantified how color, context, and direction information was represented in the animals and our network. We performed dPCA in the different areas to identify demixed principal components that represented the corresponding information. In DLPFC, we quantified the overlap of the dPCA principal axes for context, color, and direction. While the context and color axes were relatively aligned (dot product, DP: 0.52), the direction axis was closer to orthogonal to the color axis (DP: 0.18) and the context axis (DP: 0.35), as shown in Fig. 3.4b. These results suggest that DLPFC partially orthogonalizes information about the direction choice from the color choice and context. We also observed these trends in the RNN, albeit more strongly. In DLPFC-resembling Area 1, we observed the context and color were also highly aligned (DP: 0.95) but that the direction axis was more orthogonal to the color axis (DP: 0.13) and the context axis (DP: 0.12). In our simulations, the reported values reflect the mean across 8 networks trained with the same hyperparameters. A candidate mechanism for this orthogonalization, found by performing dynamical analyses on the RNN, is shown in Fig. B.5.

The advantage of our model is that both the intra-areal dynamics and inter-areal connectivity matrices are known. We analyzed how these axes were aligned with the intra-areal recurrent dynamics and inter-areal connectivity matrices to identify which hypothesis explained how the RNN implemented the information bottleneck. To do so, we performed singular value decomposition (SVD) on these matrices. We defined a k-dimensional "potent space" to be the right singular vectors corresponding to the k largest singular values of the matrix. The "null space" is the orthogonal complement of the potent space, which comprises the remaining $d - k$ smallest singular vectors, where d refers to number of columns of the matrix. The null projection magnitudes are equal to one minus the potent projection. We quantified how the color and direction axis were aligned with these potent and null spaces (see Methods). This enabled us to study if the emergence of minimal sufficient representations was due to: (1) relative amplification of the direction information with respect to a random vector, (2) relative suppression of the color/context information with respect to a random vector, or (3) a combination of both. Finally, we focused our analyses on Area 1 recurrent dynamics and the inter-areal connections between Areas 1 and 2 ($\mathbf{W}_{21}$) because color information is significantly attenuated by Area 2 (dPCA color variance in Area 2: 0.14%). The same analyses applied to downstream areas are shown in Fig. B.8.

We first tested the hypothesis that the RNN IB is implemented primarily by recurrent dynamics (left side of Fig. 3.4a). We quantified how the color and direction axis were aligned with these potent and null spaces of the intra-areal recurrent dynamics matrix of Area 1 ($\mathbf{W}^1_{rec}$). In Area 1, we found significant alignment of the color axis with the top singular vectors (potent space) of the recurrent dynamics matrix (Fig. 3.4c). This finding argues *against the hypothesis that recurrent dynamics preferentially attenuate color information by projecting it into a nullspace of the recurrent dynamics.* Rather, these data suggest that Area 1 has significant color information in its potent space, indicating that the recurrent computation amplifies color information. We also performed an alternative analysis where we compared input and activity representations of color discriminability and direction discriminability for our exemplar network. We observe an amplification, not a reduction, in color discriminability with respect to the inputs in Area 1 (Fig. B.6) consistent with the amplification observed in Fig. 3.4c. In Areas 2 and 3, the color axis (which had small variance of 0.14% and 0.07% in Areas 2 and 3, respectively) was again typically more strongly aligned with $\mathbf{W}^i_{\text{rec}}$ than a

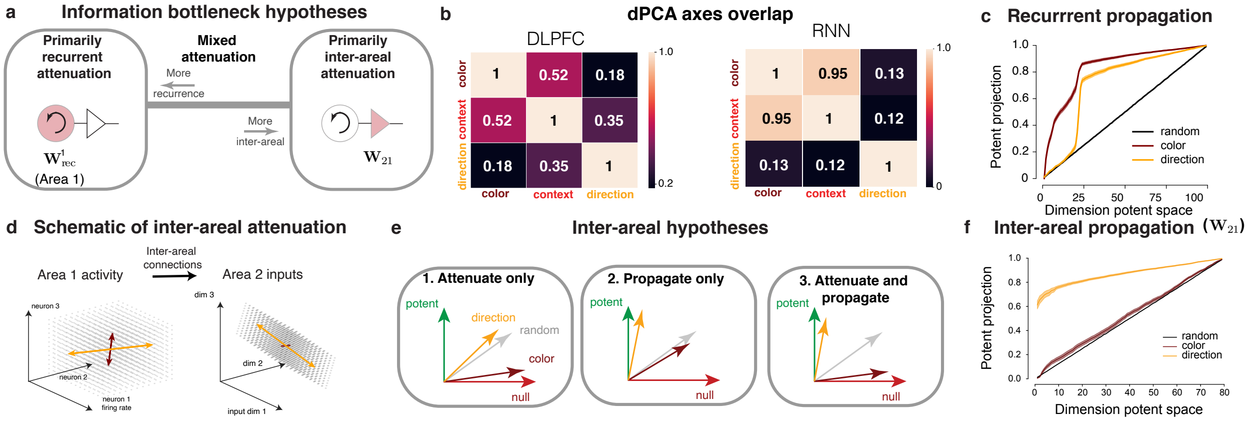

Figure 3.4: **IB hypotheses and mechanism. (a)** Candidate mechanisms for IB. **(b)** Axes overlap of the direction, color, and context axes for DLPFC and RNN data. The direction axis is more orthogonal to the color and context axes. **(c)** Projections onto the potent space of the intra-areal dynamics for each area. We computed the potent projection of the direction axis, color axis, and a random vector with each area's intra-areal dynamics matrix. We found intra-areal dynamics amplify color information in Area 1, and do not selectively attenuate color information in Areas 2 and 3. **(d)** Illustration depicting how the orientation of the axes affect information propagation. Information on the direction axis (orange) can be selectively propagated through inter-areal connections which information on the color axis (maroon) is not. **(e)** Inter-areal hypotheses. **(f)** Projections onto the potent space between areas for the color axis, direction axis, and random vector. Regardless of the dimension of the potent space, the direction axis is preferentially aligned with the potent space, indicating the information along this axis propagates, while the color axis is approximately randomly aligned. We emphasize the high alignment of the direction axis: the direction axis has a stronger alignment onto the first potent dimension of $\mathbf{W}_{21}$ than the remaining dimensions combined. Meanwhile, the color axis is aligned at nearly chance levels, and will therefore be propagated significantly less than the direction axis. Shading indicates s.e.m.

random vector, (Fig. B.8). In summary, the dPCA and discriminability analyses suggest that the network did not use recurrent dynamics to attenuate color information. We therefore reject the hypothesis that the IB is primarily implemented through intra-areal recurrent dynamics.

Our alternative hypothesis is that color information is primarily attenuated through inter-areal connections. This is schematized in Fig. 3.4d, where inter-areal connections propagate activity along the Area 1 direction axis (orange) to Area 2, but attenuate Area 1 color axis activity (maroon). To test this hypothesis, we quantified how the color and direction axis were aligned with these potent and null spaces of the inter-areal matrices. This enabled us to quantify the alignment of the direction and color axes with the inter-areal potent and null spaces and specifically determine how direction and color information were differentially propagated (Fig. 3.4e). Inter areal connections could attenuate color information by aligning the color axis with the null space of $\mathbf{W}_{21}$ (Hypothesis 1 in Fig. 3.4e), propagate information preferentially (Hypothesis 2 in Fig. 3.4e), or both attenuate and propagate information (Hypothesis 3 in Fig. 3.4e).

We calculated the projections for both the color and choice axes on to the potent space for the the connection matrix from area 1 to area 2 ($\mathbf{W}_{21}$) The projections onto the potent space are shown in Fig. 3.4f for the color and direction axis. We found the direction axis was more aligned with the potent space and the color axis was more aligned with the null space. In fact, the direction axis was consistently most aligned with the top singular vector of the $\mathbf{W}_{21}$ matrix, on average more than the remaining $d-1$ singular vectors. *In contrast, the color axis was aligned to a random vector.* This suggests that learning in the multi-area recurrent network involved aligning the relevant information (in the activations) with the top singular vector (governed by the learned parameters of the feedforward matrix). These results indicate that direction information is preferentially propagated to subsequent areas, while color information is aligned with a random vector thus maximally consistent with the "propagate" hypothesis shown in Fig. 3.4e.

Such alignment of the direction axis with the top singular vector of the connection matrix isn't trivial: the potent space depends on the parameter $\mathbf{W}_{21}$ learned during training, while the direction axis is not a parameter but a dPCA axis computed from Area 1 activity. This alignment was robust to the dimension of the effective potent space, and was consistent across networks with varying feedforward connectivity percentages (10%, 20%, 30%, 50%, 100%). Further, we found that $\mathbf{W}_{21}$ in unconstrained 3 area networks had significantly reduced alignment of the direction axis with the top singular vectors (Fig. B.8d).

In summary, the multi-area RNN information bottleneck is primarily implemented through preferential propagation of direction information through inter-areal connections. Recurrent dynamics play a role in processing color information and context to arrive at direction choice information. In Area 1, RNN dynamics actually amplify color information. Our results are therefore most consistent with the hypothesis that the IB is implemented primarily through

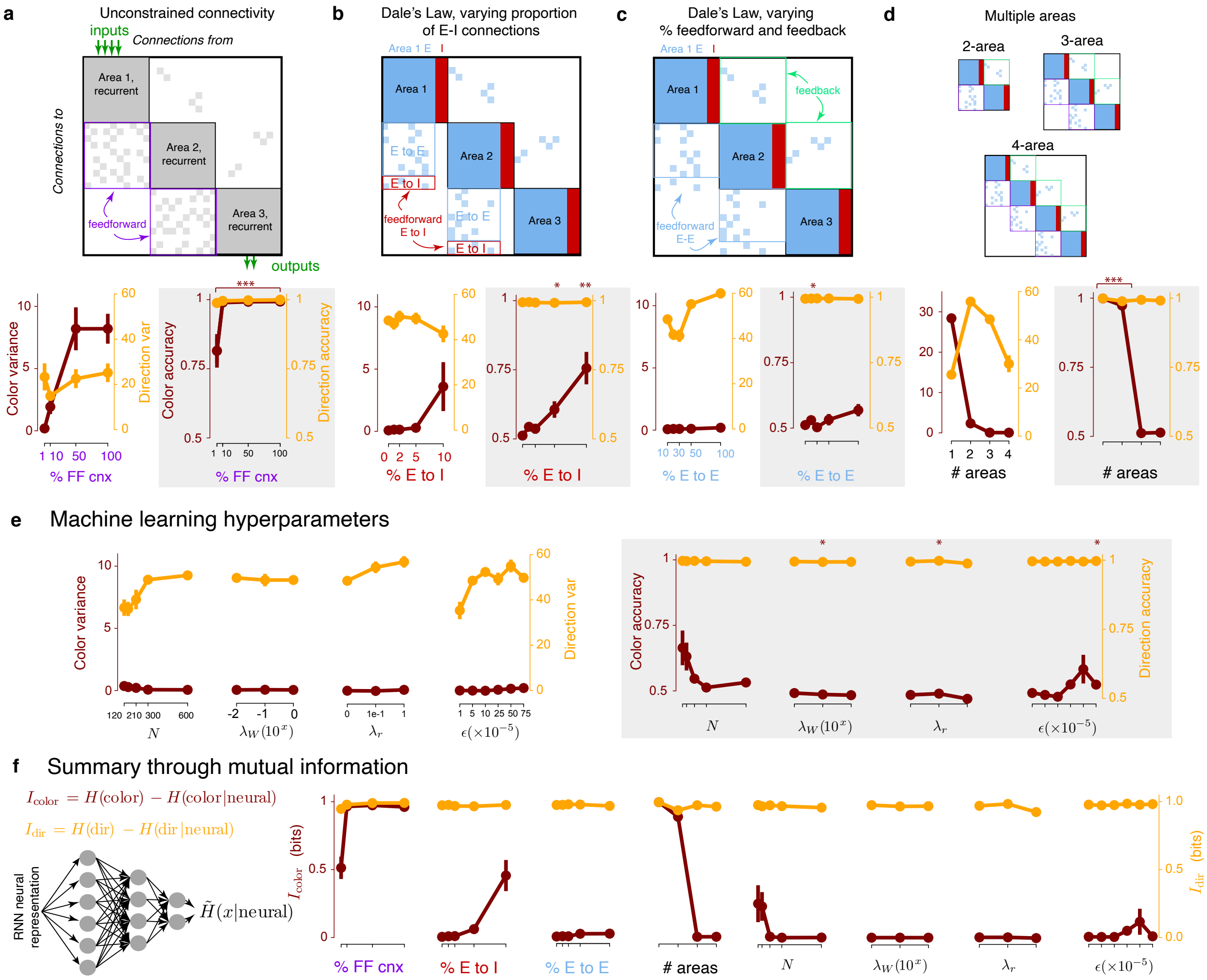

Figure 3.5: **Robustness of the information bottleneck across hyperparameters.** Varying **(a)** proportion of feedforward connections in an unconstrained network, **(b)** E-I connections in a Dale's law network, **(c)** the proportion of feedforward and feedback E-E connections in a Dale's law network, **(d)** the number of areas, and **(e)** the machine learning hyperparameters revealed that Area 3 color variance and color accuracy decrease as long as there is a connectivity bottleneck between areas. **(f)** Summary of these results quantifying usable information.

inter-areal connections, not recurrent dynamics, in Fig. 3.4a.

3.2.2 Effect of network architecture and training hyperparameters on the information bottleneck

We next assessed the network architectures and hyperparameters that influenced the formation of minimal sufficient representations during the Checkerboard task. We swept RNN architectural parameters and machine learning hyperparameters to assess what variables were important for learning minimal sufficient representations without color information. Specifically, we varied the connectivity type (unconstrained connectivity vs Dale's law and varying proportion of connections), the percentage of unconstrained feedforward connections, the percentage of feedforward E to I connections, the percentage of E to E connections, the number of areas (from 1 to 4), the number of artificial networks, L2 weight regularization, L2 rate regularization, and the learning rate. In our sweeps we quantified the color (and direction) variance and accuracy in the last area.

We generally observed minimal sufficient representations in the last area so long as there was a sufficient *connection* bottleneck between RNN areas. In unconstrained networks, shown in Fig. 3.5a, color variance and decode accuracy decreased as the percentage of feedforward connections between areas decreased, though the representations were not minimal. We incorporated Dale's law with 80% E and 20% I neurons following Song et al. [127] into subsequent sweeps (Fig. 3.5b-e). Minimal representations with chance color decode accuracy emerged when the percentage of feedforward E to I connections was $2-5\%$ or less (the overall percentage of feedforward E to E was fixed at 10% following a macaque atlas). We also found that when there was no feedforward inhibition, but when we varied the percentage of feedforward or feedback E-to-E connections RNNs generally had nearly minimal representations (Fig. 3.5c, and Supp. Fig. B.9). We observed that as long as there were 3 or 4 areas, there was a large decrease in color information in the last area (Fig. 3.5d) quantified by decoding, though note that there was a large drop in color variance for 2 area networks. These results suggest that multi-area networks, with a feedforward connection bottleneck tend to produce more minimal representations for the Checkerboard task.

We also varied machine learning hyperparameters (Fig. 3.5e) to assess the extent to which the information bottleneck was present. To prevent an exponential search space, we fixed the architecture to the exemplar network used in this study and tested one hyperparameter at a time. We varied the number of artificial units in the network, the L2 weight regularization, the L2 rate regularization, and the learning rate. At each hyperparameter setting, we trained a total of 8 multi-area RNNs. Our exemplar network consistently exhibited little to no Area 3 color information across every hyperparameter setting we chose, suggesting that the presence of the information bottleneck is not a result of a particular choice of machine learning hyperparameters.

We summarized all sweeps by calculating the "Usable Information" [78]) to quantify the direction and color information in RNNs, as shown in Fig. 3.5f and the results reaffirmed conclusions from the variance and decoding analyses. Together, these results suggest that a connection bottleneck in the form of neurophysiological architecture constraints was the key design choice leading to RNNs with minimal color representations and consistent with the information bottleneck principle.

3.3 Discussion

The goal of this study was to investigate if predictions from the information bottleneck principle in machine learning and information theory are also observed in cortical circuits. The information bottleneck principle defines an optimal representation to be one that retains only the relevant or useful information for solving a task [136]. This principle has been applied to explain the success of deep networks [3, 123], by forming minimal sufficient representations of task inputs, leading to better generalization bounds and invariance to noise [3]. We explored whether such a principle could explain cortical representations across different areas during a visual perceptual decision making task. We found that later areas of cortex along a sensorimotor transformation (in PMd) only represented the behavioral report, that is the action choice, while earlier areas had stronger input representations and performed relevant computations to define the behavioral report (in DLPFC). To better understand how such a phenomenon could be implemented in cortex, we trained many artificial multi-area RNNs to perform this task. Surprisingly, we also observed that RNNs formed minimal sufficient representations across a range of hyperparameter settings, suggesting the formation of minimal sufficient representations may be a more general feature of multi-area computation.

Given the "full-observability" of our multi-area models, we were able to analyze the learned weight matrices and understand how the network converged to transform task inputs and form minimal sufficient representations by the output area. In particular, we found that the output-relevant direction information was preferentially propagated between areas by having the largest overlap with the top singular vector of the learned feedforward matrices. In contrast, color information was almost randomly propagated through feedforward connections. This mechanism is related to prior work on output potent and output null subspaces [69] and communication subspaces [120], with the important difference that color information isn't preferentially projected to a nullspace, but is aligned similarly to any random vector. Preferential alignment with a cortical nullspace is therefore not *necessary* to achieve an IB — color information may be attenuated through random alignment to the communication subspace. This solution (random alignment) poses less constraints on inter-areal connectivity than a solution that preferentially propagates direction information while also preferentially projecting color information to a nullspace.

Our results are also consistent with recent work proposing that cortical areas convey information through communication subspaces. One observation in communication subspaces is that they do not merely propagate the directions of highest variance [120]. We also observed this for the $\mathbf{W}_{21}$ connectivity matrix, which communicates information from Area 1 to Area 2. Color activity had significant variance in Area 1 (see Fig. B.5). Inter-areal connections must therefore not merely propagate the highest variance dimensions of a preceding area, otherwise color information would be conveyed to Area 2. Consistent with this, we found that while the top 2 PCs capture 97.7% excitatory unit variance, the top 2 readout dimensions of $\mathbf{W}_{21}$ only captured 40.0% of Area 1's excitatory unit neural variance (Fig. B.7). Hence, inter-area connections are not aligned with the most variable dimensions, but are rather aligned to preferentially propagate certain types of information — a result consistent with a recent study analyzing links between activity in V1 and V2 [120].

We find minimal sufficient representations in PMd and in the later areas of our recurrent network models. Do such representations have any advantages? One possibility is that in cortex a minimal sufficient representation provides energetic benefits [8, 121]. Another possibility is that such a representation provides a computational advantage. This is an open question that is still somewhat unresolved in the machine learning community, with representation learning approaches that *maximize* mutual information between representations and inputs also leading to useful task representations [60], in addition to compressed representations [3, 78]. The information contained in the representation of a neural network is related to the "Information in the Weights" [1], which can be quantified using the Fisher Information [2, 41, 77], a measure of sensitivity to perturbations. This "Information in the Weights" view would predict that minimal sufficient representations have smaller Fisher information and are therefore less sensitive to (local) perturbations in the readout weights. In the context of deep networks, it has been proposed that minimal sufficient representations simplify the role of the output readout or classifier [3]. Further, a minimal sufficient representation with respect to a family of probabilistic decoders/classifiers will provably generalize better [36].

Although finding a resolution to this debate in machine learning is beyond the scope of this paper, we assessed if minimal RNNs exhibited any qualities consistent with machine learning predictions. We explored whether minimal sufficient representations would simplify the readout, which we quantified by measuring the model's performance in response to perturbations to the readout weights. We found that 3-area networks with minimal color information (particularly networks in Fig. 3.5b with no feedforward E-to-I connectivity) were less sensitive to perturbations than corresponding networks with significant color information (networks in Fig. 3.5a with 10% unconstrained feedforward connectivity, see Fig. B.10). We also found that these networks differed significantly in readout complexity, with 3-area networks with minimal color information exhibiting simpler and sparser readouts (Fig. B.10). However, we did not observe a clear trend between perturbation sensitivity and usable color information across random initializations (Fig. B.10) for a fixed parameter setting (networks with 10% feedforward inhibition in Fig. 3.5b). An interesting venue for future work is to further examine the potential advantages of a minimal sufficient representation. Such findings would be valuable to the machine learning and neuroscience community. In our study, several factors including recurrent connectivity, multiple areas, and E/I populations make theoretical study of this question difficult. It is likely that studying this question requires simplifying the setting. For example, it likely makes sense to first focus on feedforward networks with a variable amount of task input information, similar to the generalized checkerboard-task used in [78].

Our task could be solved with or without feedback connections with equivalent performance, indicating that feedback was not necessary to solve the task (Fig. B.9). Minimal sufficient representations were found in both purely feedforward RNNs or RNNs with feedback (Fig. B.9). When the model had feedback connections, we observed that feedback connections between Areas 2 and 1 preferentially conveyed direction information. Due to the presence of choice related signals in several cortical areas, these feedback connections may also play a role in computation of the direction choice. Another perspective on feedback signals is that they may related to error signals used for learning [94]. Multi-area networks may help understand and develop new hypotheses for physiological studies of feedforward and feedback computation [32, 103], and more generally distributed processing for decision-making and cognition [72, 107].Future research may use carefully designed tasks in conjunction with multi-area RNNs to better understand the role of feedback in computation.

Chapter 4

Usable Information and Evolution of Optimal Representations During Training

We introduce a notion of usable information contained in the representation learned by a deep network, and use it to study how optimal representations for the task emerge during training. We show that the implicit regularization coming from training with Stochastic Gradient Descent with a high learning-rate and small batch size plays an important role in learning minimal sufficient representations for the task. In the process of arriving at a minimal sufficient representation, we find that the content of the representation changes dynamically during training. In particular, we find that semantically meaningful but ultimately irrelevant information is encoded in the early transient dynamics of training, before being later discarded. In addition, we evaluate how perturbing the initial part of training impacts the learning dynamics and the resulting representations. We show these effects on both perceptual decision-making tasks inspired by neuroscience literature, as well as on standard image classification tasks.

4.1 Introduction

An important open question for the theory of deep learning is why highly over-parametrized neural networks learn solutions that generalize well even though the model can in principle memorize the entire training set. Some have speculated that neural networks learn minimal but sufficient representations of the input through implicit regularization of Stochastic Gradient Descent (SGD) [3, 123], and that the minimality of the representations relates to generalizability. Follow-up work has disputed the validity of some of these claims when using deterministic deep networks [119], leading to an ongoing debate on the notion of optimality of representations and how they are learned during training.

Part of the disagreement stems from the use of information-theoretic quantities: most previous studies in deep learning have analyzed the amount of information that the learned representation contains about the inputs using Shannon's mutual information. However, when the mapping from input to representation is deterministic, the mutual information between the representation and input is degenerate [49, 119]. Rather than study the mutual information in a neural network, here we instead define and study the "usable information" in the network, which measures the amount of information that can be extracted from the representation by a learned decoder, and is scalable to high dimensional realistic tasks. We use this notion to quantify how relevant and irrelevant information is represented across layers of the network throughout the training process, and how this is affected by the optimization algorithms and the network pretraining.

In particular, we propose to study a simple task inspired by decision-making tasks in neuroscience, where inputs and outputs are carefully designed to probe specific information processing phenomena. We then extend our findings to standard image classification tasks trained with state-of-the-art models. Our neuroscience-inspired task is the checkerboard (CB) task [24, 83]. In the CB task, one discerns the dominant color of a checkerboard filled with red and green squares. The subject then makes a reach to a left or right target whose color matches the dominant color in the checkerboard (Fig 4.1a). This task therefore involves making two binary choices: a color decision (i.e., reach to the red or green target) and a direction decision (i.e., reach to left or right). Critically, the color of the targets (red left, green right; or green left, red right) is random on every trial. The direction decision output is conditionally independent of the color decision, as detailed further in Fig 4.1b and Section C.2.6, even though the color information needs to be used to solve the task. This task allows us to evaluate how both of these components of information are represented through training and across layers.

We used this task and extensions to study the evolution of minimal representations during training. If a representation is sufficient and minimal, we refer to this representation as optimal [3]. Our contributions are the following. **(1)** We introduce a notion of usable information for studying representations and training dynamics in deep networks (Section 4.3). **(2)** We used this notion to characterize the transient training dynamics in deep networks by studying the amount of usable relevant and irrelevant information in deep network layers and across training epochs. We first use the CB task to gain intuition of the training dynamics in a simplified setting. We find that training with SGD is critical to bias the network toward learning minimal representations in intermediate layers (Section 4.4.1). This adds to the literature suggesting that SGD results in minimal representations of input information [3, 123] while avoiding some of the pitfalls. **(3)** We used the intuition gained from the simple task, evaluating our findings on CIFAR-10 and CIFAR-100 task using modern architectures. Remarkably, we find that the networks increased usable information about an irrelevant component of information early in training and discarded it later on in training to arrive at a minimal sufficient solution, consistent with a proposed [123] though controversial theory [119].

4.2 Related Work

Some efforts to understand why neural networks generalize focus on representation learning, that is, how deep networks learn optimal (i.e., minimal and sufficient) representations of inputs in order to solve a task. Typically, representation learning is focused on studying the properties of the asymptotic representations after training [3]. Recent work suggests that these asymptotic representations contain minimal but sufficient input information for performing a task [3, 123]. Implicit regularization coming from SGD, and in particular from the use of large learning rates and small batch sizes, is believed to play an important role in forming these minimal sufficient representations.

How does the training process lead to these minimal but sufficient asymptotic representations? [123] propose that there are two distinct phases of training: an empirical risk minimization phase where the network minimizes the loss on the training set, and a "compression" phase where the network discards information about the inputs that do not need to be represented to solve the task. Recently, [119] challenged this view, arguing that the observed compression was dependent on the activation function and the mutual information estimator used in [123]. These works highlight the challenges of estimating mutual information to study how representations emerge through training.

In general, estimating mutual information from samples is challenging for high-dimensional random variables [105]. The primary difficulty in estimating mutual information is estimating a high-dimensional probability distribution from the samples, since generally the number of samples required scales exponentially with the dimension. This is impractical for realistic deep learning tasks where the representations are high dimensional. To estimate the mutual information, [123] used a binning approach, discretizing the activations into a finite number of bins. While this approximation is exact in the limit of infinitesimally small bins, in practice, the size of the bin affects the estimator [49, 119]. In contrast to binning, other approaches to estimate mutual information include entropic-based estimators (e.g., [49]) and a nearest neighbours approach [88]. Although mutual information is difficult to estimate, it is an appealing quantity to summarily characterize key aspects of the transient neural network training behavior because of its invariance to smooth and invertible transformations. In this work, rather than estimate the mutual information directly, we instead define and study the “usable information” in the network, which corresponds to a variational approximation of the mutual information [15,108] (see Sections 4.3 and C.1.1). Recently, such variational approximations to mutual information have been viewed as a meaningful characterization of representations in deep networks, and the theoretical underpinnings of this approach are beginning to be investigated [36,151].

Research into the training dynamics of deep networks, and how they represent relevant and irrelevant task information, is nascent. A related study by [2] found that early periods of training were critical for determining the asymptotic network behavior. Additionally, it was found that the timing of regularization was important for determining asymptotic performance [47], with regularization during this “critical period" having the most influential effect. Notably, both of these studies found an initial increase in the amount of information that weights encode about the dataset (as measured by the Fisher information), that coincides with the critical period of learning. This phase is followed later in training by a “forgetting” phase where the network discards unnecessary information. This suggests that a similar dynamic to the one we study can be observed in weight space instead of representation space.

4.3 Usable information in a representation

A deep neural network consists of a set of ℓ layers, with each layer forming a successive representation of the input. A representation Z_ℓ may store information in a variety of ways. It may be that a complex transformation is required to read out the information, or it may be that a simple linear decoder could read out the information. In both cases, from an information-theoretic perspective, the same information is contained in the representation, however, there is an important distinction regarding how “usable” this information is. Information is usable if later layers, which comprise affine transformations and element-wise nonlinearities, can easily extract it to solve the task. Equivalently, usable information should be decodable by a separate neural network also employing affine transformations and element-wise nonlinearities.

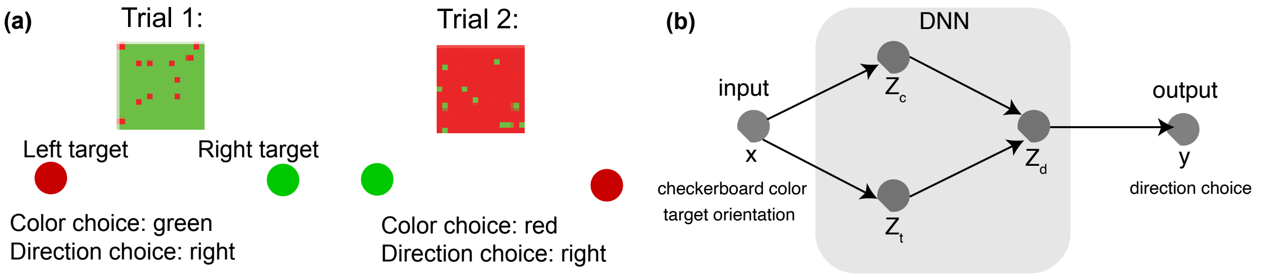

Figure 4.1: **(a)** Checkerboard task. Given two binary target locations (left or right) with randomly selected binary colors (red or green), one has to discern the dominant color in the checkerboard and reach to the target of the dominant color. On every trial, there is a correct color and direction choice. However, the identities of the left and right targets are random every trial, decoupling the direction and color decision. **(b)** We trained a deep neural network to perform the task by specifying the proportion of green and red squares on the checkerboard, as well as two scalars denoting the colors of the left and right target. The network was trained to output the correct direction choice. As only the direction, but not the color choice, was reported, given a representation of the correct direction choice Z_d, the network does not need to represent the color choice Z_c in deeper layers. Z_t is the representation of the target orientation.

Formally, we define the usable information that a representation Z contains about a quantity Y, which may refer to the output or a component of the input, as:

$$I_u(Z;Y) = H(Y) - L_{CE}(p(y|z), q(y|z)). \tag{4.1}$$

Here, $H(Y)$ is the entropy, or uncertainty, of Y, and L_{CE} is the cross-entropy loss on the test set of a discriminator network $q(y|z)$ trained to approximate the true distribution $p(y|z)$. Our definition is motivated in the following manner. The test set cross-entropy loss approximates how much uncertainty there is in the output Y given Z and the discriminator. A low loss implies that there is low uncertainty in Y given Z, or that the discriminator can extract a lot of "information" about Y from Z. If the logarithm in the cross-entropy loss is in base 2, it is measured in bits. If the value of Y were approximately the same for any Z, there would be little uncertainty in Y to begin with, so it is important to know the amount of uncertainty in Y given Z with respect to the initial uncertainty in Y. What is most relevant is the amount of remaining uncertainty in Y given Z. Thus we use the difference in uncertainty $H(Y) - L_{CE}$ as the amount of "usable information" that Z contains about Y, as shown in our definition in Equation 4.1.

This definition is appealing to study representations, in part, because it can be computed from samples of Z and Y, and is a quantity that is comparable across network training. We estimate L_{CE} using a small neural network that learns a distribution $q(y|z)$. To train the network, we sample activations Z and the quantity Y and learn $q(y|z)$ by minimizing the cross-entropy loss on a training set. We then evaluate the L_{CE} on the test set (Equation 4.1). We provide details about the neural network and the training we used for decoding in Appendix C.2.3 and C.3.2. We also show in the Appendix that the usable information is a lower bound on the mutual information (Appendix C.1.1). Importantly, usable information also is not constrained by the data processing inequality; that is, the information can be made more "usable" by transformation to later layers, consistent with the representation learning view that later layers are forming improved representations of the inputs [151].

4.4 Experiments

Our goal was to characterize how optimal representations are formed through SGD training. We trained multiple network architectures on tasks and assessed the usable information in representations across layers and training epochs. For a given architecture and task, all hyper-parameters were kept constant throughout experiments, unless explicitly stated.

To develop intuition, we initially investigate how small fully connected networks represent the relevant and irrelevant information in the CB Task. We trained two different network architectures, 'Small FC': 5 layers, with $10-7-5-4-3$ units in each layer, 'Medium FC': $100-20-20-20$. Small FC was a network used in prior literature [119, 123]. Our networks were fully-connected and used ReLU activation. We trained the networks using SGD with a constant learning rate to perform the CB task, described in detail in Appendix

C.2.4. The hyper-parameters used for the CB experiments are listed in Appendix C.2.5.

In our CB task experiments, we quantified the usable color and direction information in the hidden representation, Z_ℓ. In the $n = 2$ CB task, the color information represents half of the input information. We emphasize that, unless otherwise specified, the network was only trained to output the correct direction choice, so given a representation of the direction, representing the color choice is irrelevant. Therefore, a minimal representation should not include information about the color choice, since it is not necessary to represent given a representation of the direction decision. To make the task more complex, we also generalized the CB task to have $n = 10$ and $n = 20$ targets.

We then use this framework to examine how relevant and irrelevant information are represented in more realistic tasks and architectures, and how hyper-parameters affect the learning dynamics. We define a coarse labelling of task labels and study how the network represents the fine and coarse labelling through training, using a ResNet-18 [55] and All-CNN [129] on CIFAR-10 and CIFAR-100.

4.4.1 SGD with random initialization results in minimal sufficient representations in the CB Task

We first assessed the optimality of the network representations by training Small FC networks on the CB task using $n = 2$ colors (Fig 4.2a) using a random initialization for the weights. In particular, the initial weights do not contain information about the dataset. We computed the usable color and direction information across layers of the neural network and epochs of training. In our plots, later layers are denoted by darker shades. In deeper layers, there was a decrease in usable color information, corresponding to more minimal representations. After training, the asymptotic representation in the last layer contained zero usable color information and 1 bit of usable direction information. To visualize this minimal sufficient representation, we plotted the activations of the 3 units in the last layer of the Small FC network for different inputs. These visualizations are labeled by the correct color (red and

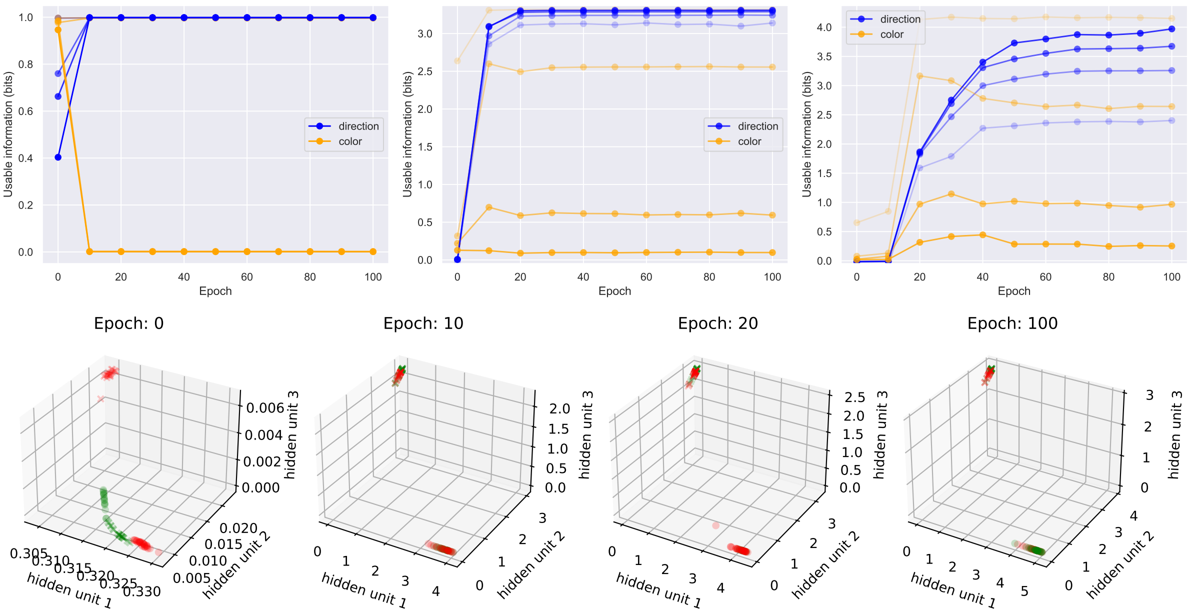

Figure 4.2: **SGD with random initialization leads to minimal representations. (a)** Small FC network trained on the $n = 2$ checkerboard task. Max usable direction and color information: 1 bit. This network was trained without regularization for 100 epochs using SGD with a learning rate of 0.05 and batch size of 32. Blue (orange) lines correspond to usable information about the direction (color) decision in the representation. Darker shades of color correspond to deeper layers in the network. In the asymptotic representations, we observed that direction information was high across layers, while color information decreased in the later layers.The usable color information was approximately zero in the last layer of the Small FC network. **(b)** Medium FC network trained with $n = 10$ checkerboard colors. Max usable direction and color information: 3.32 bits. In the last layer, there is nearly zero usable color information. Across layers, there is a decrease in usable color information, and an increase in usable direction information. **(c)** Medium FC network trained with $n = 20$ checkerboard colors, a batch size of 128 and a learning rate of 0.5. Max usable direction and color information: 4.32 bits. In the later layers (darker shades) there is small usable color information, but large usable direction information. **(d)** Visualization of the activations of the last layer of Small FC from (a) at epochs [0, 10, 20, 100], where the correct color choice is denoted by the marker color (red or green) and the correct direction choice is denoted by marker shape (crosses or dots). After training the crosses and dots are overlapping, corresponding to nearly zero usable color information and nearly 1 bit of direction information. This is a minimal and sufficient representation to solve the task.

green) and direction (cross or circle). In the asymptotic representation, representation of the input color is overlapping (red and green), while the representation of the direction output is separable (crosses and circles), forming a minimal sufficient representation.

To test if this observed minimality was a result of our simple task, we extended the CB task to a variant with n input checkerboard colors, with n corresponding output direction classes. We trained networks using a larger architecture (Medium FC). We show results for $n = 10$ and $n = 20$ classes in Fig 4.2b,c. We observed similar phenomena to the $n = 2$ case: there was decreasing usable color information in deeper layers, and nearly zero color information in the last layer's representation. In contrast, there was significant usable direction information across all layers in the asymptotic representation, with usable information about the direction increasing for deeper layers. We validated our results using different random initializations (Figures C.4, C.5, C.6).

These results show that, for a simple task with SGD and random initialization, minimal sufficient representations emerge through training. Asymptotic representations were sufficient to perform the task, but contained less usable color information in deeper layers, approaching zero color information in the last layer. In this simple task, we observed that it was possible for the network to solve the task with nearly zero usable color information in its last layer across training (Fig 4.2b,c).

We also examined how changing the initialization by pretraining the network to output the color choice affected the resulting representations. We found that the resulting representations were not minimal for the $n = 2$ checkerboard case (Fig C.1a), retaining some structure from the initialization (Fig C.1d). This result also held for the CB task with $n = 10$ and $n = 20$ (Fig C.1b,c). Furthermore, we found that pretraining on the color choice led to worse generalization performance (Fig C.2).

4.4.2 Acquisition and forgetting of usable information in modern deep networks

Using a similar approach as we did for the CB task to characterize relevant and irrelevant information, we next investigated how modern deep neural networks trained with SGD learned task representations. To study learning dynamics, we investigated (1) how networks learned and represented task information as well as information about a representative semantically

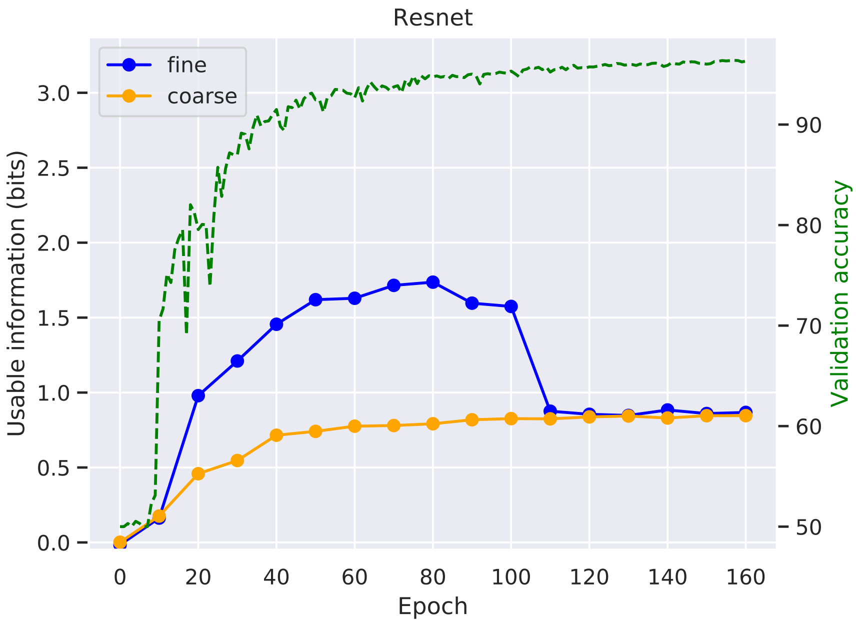

Figure 4.3: **Usable fine and coarse class information in a ResNet-18 on CIFAR-10.** The fine classes (show in blue) correspond to the 10 CIFAR-10 classes. The coarse classes correspond a superclass consisting of all the even and odd classes. We trained the network to output the correct coarse class, which corresponds to 1 bit of information. Through training epochs, while the validation accuracy (green dashed line) is increasing, the information about the coarse class also increases towards 1 bit. Early in training, the usable information about the fine label also increased, even though the network was not explicitly provided any information about the fine class. Around epoch 100, the network "forgets" this fine label information. The scale of the validation accuracy is shown on the right hand side of the plot.

meaningful variable, and (2) how this information was represented across training epochs. To this end, we defined coarse labels corresponding to groups of classes in the CIFAR-10 and CIFAR-100 datasets. The CIFAR-100 dataset defines fine labels corresponding to each of the 100 classes, as well as 20 coarse labels corresponding to meaningful groupings of 5 from the 100 classes. In the CIFAR-10 case, we defined two coarse labels arbitrarily, corresponding to even and odd class labels. Thus, when training the network to output the coarse label, we can investigate the network's representation of the semantically meaningful fine label description, which serves as a proxy for the computation and representations that the network is learning. We note that, when trained to output the coarse label, a minimal representation should contain no additional information about the fine label.

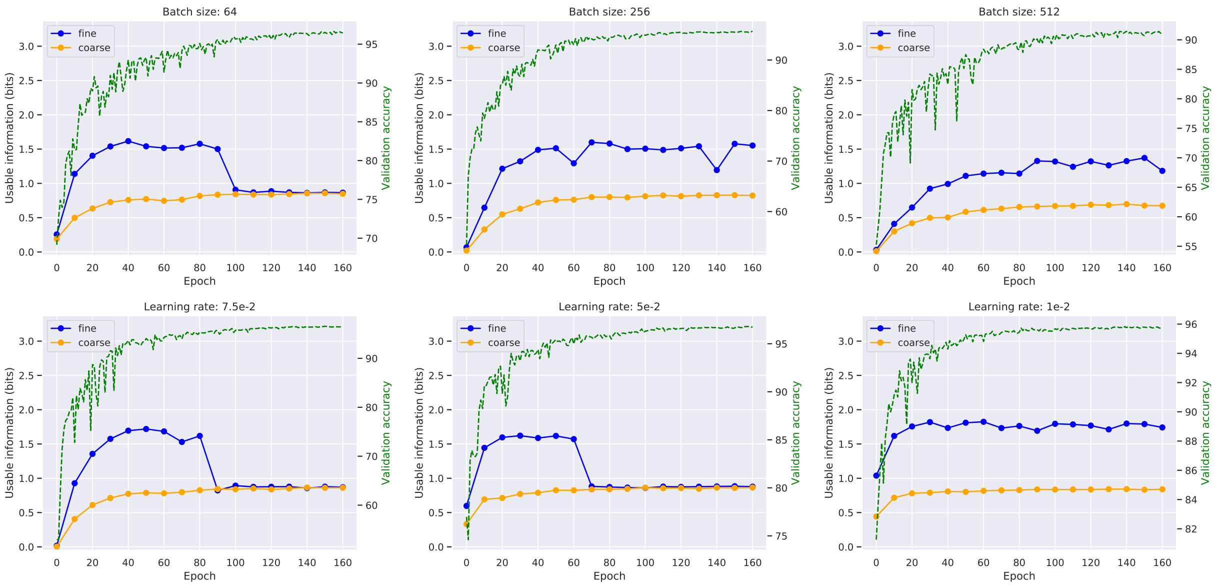

Figure 4.4: **Sensitivity to hyper-parameters. (a-c)** The usable coarse and fine label information through training with a batch size of 64, 256, and 512 (a batch size of 128 was used in Fig 4.3. The learning dynamics only undergo a compression at small batch sizes of 128 or less. The validation accuracy is higher for smaller batch sizes as well. The plot of a batch size of 1024 is in Fig C.3. **(d-f)** Usable coarse and fine label information using initial learning rates of 0.075, 0.05 and 0.01 (a learning rate of 0.1 was used in Fig 4.3. With larger learning rates, the network observed an increase and decrease in fine label information. With a smaller learning rate 0.01, the network exhibited an increase in fine label information, without a subsequent decrease. The final validation accuracies (green dashed lines) are approximately comparable (96.5%, 96.8% and 95.8% respectively) though lowest with initial learning rate of 0.01 when the network did not form a minimal representation.

We trained a ResNet-18 [55] to output the coarse label of CIFAR-10, using an initial learning rate of 0.1 with exponential annealing (0.97), momentum (0.9), and a batch size of 128. We investigated the usable information in the last layer of the ResNet-18, which has a dimension of 512. We found that while training the network to predict the coarse-grained class, the network acquired information about the coarse-grained class, evidenced by an increase in usable information during training (orange curve) while validation accuracy (green dashed line; scale on the right hand side of plot) was increasing (Fig 4.3). Strikingly, while the validation accuracy and usable coarse-grained class information increased, the information about the fine labels first increased and then decreased (around epoch 100). It then decreased to minimality, storing no additional usable information about the fine labels than was contained in the coarse labels. These learning dynamics were proposed [123], but due to controversies of their information estimation and experimental setup, have been widely debated [119]. We emphasize that even though we did not ask the network to acquire information about the fine labels, SGD naturally led the network to learn information about the fine label, and then decreased this information later in training.

Together, these results show that SGD tends to result in minimal representations, which may be guided by interesting learning dynamics. To achieve this minimality, the network displays a learning motif where it learns additional information early in training, then discards it later on. We next investigate how these findings depend on hyper-parameter choices, architecture, and task.

4.4.3 Sensitivity of usable information training dynamics to hyper-parameters, architecture, and task

Using this framework, we evaluated how hyper-parameter choices affected the learning dynamics in deep networks. We focus on the ResNet-18 trained on CIFAR-10 in Figure 4.3. We varied the batch sizes from 64 to 1024 and found that a small batch size led to dynamics similar to that of Fig 4.3, while a larger batch size did not lead to minimal representations

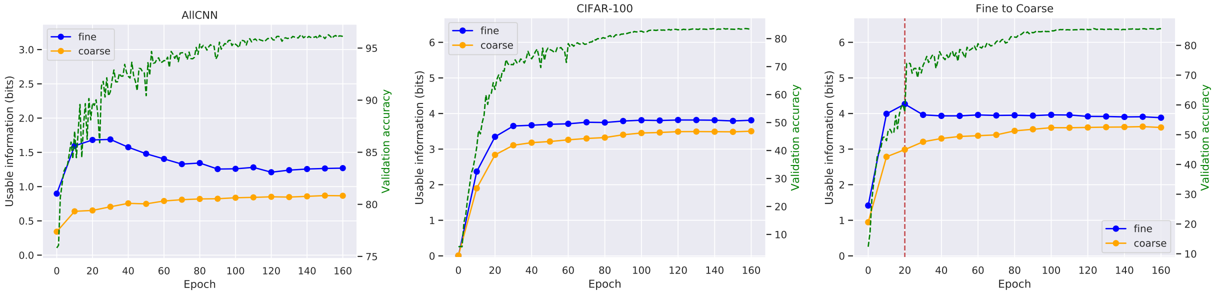

Figure 4.5: **Different Architecture, Task, and learning schedule (a)** Using an All-CNN architecture [129], we observe a similar trend in the learning dynamics of usable information, with a increase and decrease in the fine label information during the CIFAR-10 task. This decrease does not lead to a completely minimal representation, though it does become close to minimal. **(b)** We trained a ResNet-18 on the coarse labels in the CIFAR-100 task, and tracked the information the network had about the fine and coarse label through training. We find that the network converges to an approximately minimal representation, though it did not undergo a noticeable increase and decrease in the fine label information, suggesting that this learning motif depends on the structure of the task. **(c)** Pretraining the network to output the fine labels before epoch 20 led to improved final performance (85.6% vs 83.5%) in **(b)**. Note that the validation accuracy for the first 20 epochs was the validation accuracy on the 'fine' labels task, and was the validation accuracy on the 'coarse' task after epoch 20.

(Fig 4.4a-c). Results for a batch size of 1024 are shown in Fig C.3. The learning rate also affected the learning dynamics. We found that all networks increased the information about the fine labels during training. However, we found that only for large initial learning rates did the network "forget" the superfluous information. Results for a learning rate of 0.001 are also shown in the appendix in Fig C.3. We found that small learning rates (0.001) or large batch sizes (512 or larger) led to lower validation accuracy. Thus, the implicit regularization coming from the use of SGD with a small batch size and large learning rate, which is common in practical settings, is crucial for learning minimal sufficient representations. Here we have provided an underpinning for these choices by exposing their associated learning dynamics.

Additionally, we investigated whether the phenomenon of acquiring "superfluous" task information was common across different architectures and tasks. We used an All-CNN [129] trained on CIFAR-10 to output the binary coarse label, observing a similar trend with an increase and decrease in the usable information about the fine label (Fig 4.5a). In this case, the information about the fine label did not decrease to minimality, but nonetheless, there was a significant reduction in the fine label information, suggesting that SGD naturally compresses additional input information. Finally, we evaluated how a ResNet-18 represented task information using the CIFAR-100 dataset. This dataset is accompanied with 100 fine labels and 20 coarse labels, corresponding to groupings of the 100 classes. We used the same hyper-parameters as in Fig 4.3. We trained the network to output the coarse labels, observing an increase to approximately 3.5 bits of usable information. The network achieved a nearly minimal representation (Fig 4.5b).

It is important to note that for this setting of hyper-parameters in the CIFAR-100 task (the same as in the CIFAR-10 case), SGD did not show a visible increase followed by a decrease in usable information in the fine labels, a result different than what we observed in CIFAR-10. We conjectured this could be due to at least three potential reasons: (1) the hyper-parameter settings may be suboptimal, which we observed may result in learning dynamics that do not increase then decrease fine information (Fig 4.4c, f). (2) In CIFAR-100, coarse and fine labels are semantically similar, so there may not be not much more information to be naturally learned in the fine than the coarse labels, and further that it is possible that while the information about the fine labels remains approximately flat, the network is forgetting information about aspects of the fine labels while learning other parts of fine label information in the process of increasing coarse label information and arriving at a nearly minimal representation. (3) CIFAR-100 has relatively few examples, 500 per fine label, impacting the learning of fine label information. Despite these limitations, our results from CIFAR-10 suggest that SGD learning dynamics that increase then decrease information about the fine label should result in more optimal representations and higher validation accuracy. To test this, we performed an experiment where we pretrained the network to output fine label information until epoch 20, after which the network then was trained to output coarse information. This training process resulted in learning dynamics that resembled SGD learning in Fig 4.3. We observed that these learning dynamics resulted in networks with a 2.1% increase in validation accuracy (compare Fig 4.5b and c). These results support that learning dynamics that increase, and then decrease, information about inputs, may result in more optimal representations that achieve higher validation accuracy.

4.5 Discussion

We introduced a notion of the usable information in the representation, which reflects the amount of information that can be extracted by a learned decoder, for understanding the training dynamics in deep networks. This definition is appealing, in part, due to its flexibility. For instance, if it is important to understand how accessible the information is to a linear decoder, it suffices to apply our formulation of usable information using a linear decoder trained with cross-entropy loss. In contrast, if the goal is to extract all information present in a representation, regardless of how accessible this information is, one can train a high capacity nonlinear decoder. Since neural networks are powerful function approximators, as the function approximation improves, the decoder will approach the optimal decoder. In this case, the usable information approaches Shannon mutual information, as the lower bound becomes tight (Section C.1.1). Future theoretical and empirical work should investigate the tightness of this bound and its dependence on training parameters.

In our case, we used a relatively small nonlinear neural network as the decoder, which provided insight into the evolution of optimal representations through training on simple tasks inspired by neuroscience literature and on image classification tasks. These tasks allowed us to show that the implicit regularization of SGD plays an important role in learning minimal sufficient representations. In particular, in standard hyper-parameter settings, we observed learning dynamics where the network learns to encode semantically meaningful but ultimately irrelevant information early in training, before later discarding this information to arrive at a minimal sufficient representation.

Monkeys performing the checkerboard task, like our networks, also had minimal sufficient representations in an output (motor) area [24, 83]. Despite the obvious implementation differences of both information processing systems, we speculate that the general effects coming from a noisy learning process, which led to minimal sufficient representations in our artificial networks, may be an important factor leading to minimal sufficient representations in biological networks.

It is remarkable that in the CIFAR-10 task, SGD naturally exploited the semantically meaningful structure of the fine labels, in order to solve the coarse labels task. In general, it is difficult to identify the features that are being learned during training, and whether they correspond to something semantically meaningful. However by defining a coarse label, our task setup allowed us to study how semantically meaningful information was represented during training. During training, the network increased the information about the semantically important part of the input, even when only asked to output the coarse label. It then decreased the information later in training. We did not notice such a major increase in CIFAR-100, perhaps due to the nature of the dataset or hyper-parameter configuration. However, by inducing the network to follow similar learning dynamics to Fig 4.3 by pretraining the network to output the fine labels, we were able to improve the performance on the coarse labelling task. This suggests that a detailed understanding of the training dynamics and the features learned is important for learning optimal representations and successfully transferring representations between tasks.

Using usable information, we observed an increase and decrease in the information about an irrelevant variable, which has been proposed [123], but has been debated, largely due to controversies over the estimation of Shannon's mutual information [119]. Our observation is in accordance with the ideas of [123], and importantly we have observed these dynamics on modern architectures and realistic tasks. Our results are also consistent with a complementary view of information in the weights, where it has been observed that the Fisher Information increased and decreased during training [2], corresponding to a critical period in neural network training.

Chapter 5

Critical Learning Periods for Multisensory Integration in Deep Networks

We show that the ability of a neural network to integrate information from diverse sources hinges critically on being exposed to properly correlated signals during the early phases of training. Interfering with the learning process during this initial stage can permanently impair the development of a skill, both in artificial and biological systems where the phenomenon is known as *critical learning period.* We show that critical periods arise from the complex and unstable early transient dynamics, which are decisive of final performance of the trained system and their learned representations. This evidence challenges the view, engendered by analysis of wide and shallow networks, that early learning dynamics of neural networks are simple, akin to those of a linear model. Indeed, we show that even deep linear networks exhibit critical learning periods for multi-source integration, while shallow networks do not. To better understand how the internal representations change according to disturbances or sensory deficits, we introduce a new measure of source sensitivity, which allows us to track the inhibition and integration of sources during training. Our analysis of inhibition suggests cross-source reconstruction as a natural auxiliary training objective, and indeed we show that architectures trained with cross-sensor reconstruction objectives are remarkably more resilient to critical periods. Our findings suggest that the recent success in self-supervised multi-modal training compared to previous supervised efforts may be in part due to more robust learning dynamics and not solely due to better architectures and/or more data.

5.1 Introduction

Learning generally benefits from exposure to diverse sources of information, including different sensory modalities, views, or features. Multiple sources can be more informative than the sum of their parts. For instance, both views of a random-dot stereogram are needed to extract the *synergistic information*, which is absent in each individual view [65]. More generally, multiple sources can help identify latent common factors of variation relevant to the task, and separate them from source-specific nuisance variability, as done in contrastive learning.

Much information fusion work in Deep Learning focuses on the design of the architecture, as different sources may require different architectural biases to be efficiently encoded. We instead focus on the *learning dynamics*, since effective fusion of different sources relies on complex phenomena beginning during the early epochs of training. In fact, even slight interference with the learning process during this *critical period* can permanently damage a network's ability to harvest synergistic information. Even in animals, which excel at multi-sensor fusion, a temporary deficit in one source during early development can permanently impair the learning process: congenital strabismus in humans can cause permanent loss of stereopsis if not corrected sufficiently early; similarly, visual/auditory misalignment can impair the ability of barn owls to localize prey [67]. In artificial networks, the challenge of integrating different sources has been noted in visual question answering (VQA), where the model often resorts to encoding less rich but more readily accessible textual information [5,22], ignoring the visual modality, or in audio-visual processing, where acoustic information is

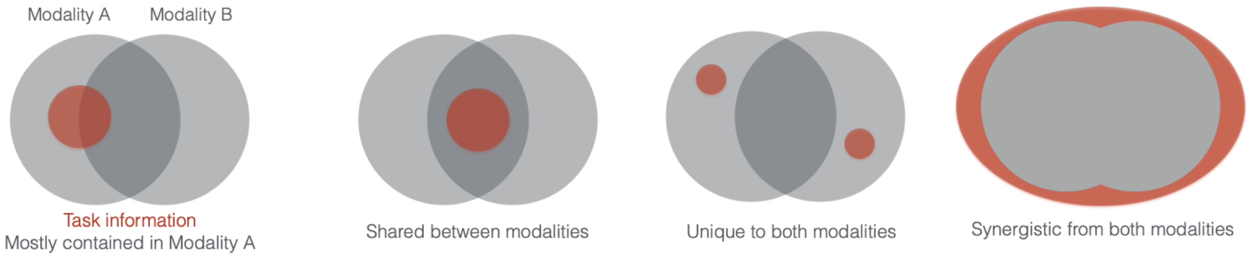
Modality A
Modality B
Task information
Mostly contained in Modality A
Shared between modalities
Unique to both modalities
Synergistic from both modalities

Figure 5.1: **Decomposition of information between different modalities.** Two modalities can have unique information, common information (denoted by the overlap in the venn-diagram), or synergistic information (denoted by the additional ellipse in the right panel). Task-relevant information (shown in red) can be distributed in a variety of ways across the different modalities. Task-relevant information can be mostly present in Modality A (left), shared between modalities (center-left), or could require unique (center-right) or synergistic information from both modalities (right).

often washed out by visual information [144].

Such failures are commonly attributed to the mismatch in learning speed between sources, or their "information asymmetry" for the task. It has also been suggested, based on limiting analysis for wide networks, that the initial dynamics of DNNs are very simple [62], seemingly in contrast with evidence from biology. In this paper, we instead argue that *the early learning dynamics of information fusion in deep networks are both highly complex and brittle, to the point of exhibiting critical learning periods similar to biological systems.*

In Sect. 5.2, we show that shallow networks do not exhibit critical periods when learning to fuse diverse sources of information, but *deep* networks do. Even though, unlike animals, artificial networks do not age, their learning success is still decided during the early phases of training. The existence of critical learning periods for information fusion is not an artifact of annealing the learning rate or other details of the optimizer and the architecture. In fact, we show that critical periods for fusing information are present even in a simple deep linear network. This refutes the idea that deep networks exhibit trivial early dynamics [62, 91]. We provide an interpretation for critical periods in linear networks in terms of mutual inhibition/reinforcement between sources, manifest through sharp transitions in the learning dynamics, which in turn are related to the intrinsic structure of the underlying data distribution.

In Sect. 5.3, we introduce a metric called "Relative Source Variance" to quantify the dependence of units in a representation to individual sources, allowing us to better understand inhibition and fusion between sources. Through it, in Sect. 5.4, we show that temporarily reducing the information in one source, or breaking the correlation between sources, can permanently change the overall amount of information in the learned representation. Moreover, even when downstream performance is not significantly affected, such temporarily changes result in units that are highly polarized and process only information from one source or the other. Surprisingly, we found that the final representations in our artificial networks that were exposed to a temporary deficit mirrored single-unit animal representations exposed to analogous deficits (Fig. 5.4, Fig. 5.6).

We hypothesize that features inhibit each other because they are competing to solve the task. But if the competitive effect is reduced, such as through an auxiliary cross-source reconstruction task, the different sources can interact synergistically. This supports cross-modal reconstruction as a practical self-supervision criterion. In Sect. 5.4.4, we show that indeed auxiliary cross-source reconstruction can stabilize the learning dynamics and prevent critical periods. This lends an alternate interpretation for the recent achievements in multi-modal learning as due to the improved stability of the early learning dynamics due to auxiliary cross-modal reconstruction tasks, rather than to the design of the architecture.

Empirically, we show the existence of critical learning periods for multi-source integration using state-of-the-art architectures (Sect. 5.4.3-5.4.4). To isolate different factors that may contribute to low-performance on multi-modal tasks (mismatched training dynamics, different informativeness), we focus on tasks where the sources of information are symmetric and homogeneous, in particular stereo and multi-view imagery. Even in this highly controlled setting, we observe the effect of critical periods both in downstream performance and/or in unit polarization. Our analysis suggests that pre-training on one modality, for instance text, and then adding additional pre-trained backbones, for instance visual and acoustic, as advocated in recent trends with Foundation Models, yields representations that fail to encode synergistic information. Instead, training should be performed across modalities at the outset. Our work also suggests that asymptotic analysis is irrelevant for deep network fusion, as their fate is sealed during the initial transient learning. Also, conclusions drawn from wide and shallow networks do not transfer to deep networks in use in practice.

5.1.1 Related Work

Multi-sensor learning. There is a large literature on sensor fusion in early development [125], including homogeneous sensors that are spatially dislocated (e.g., two eyes), or time-separated (e.g., motion), and heterogeneous sources (e.g., optical and acoustic, or visual and tactile). Indeed, given *normal learning*, humans and other animals have the remarkable ability to integrate multi-sensory data, such as incoming visual stimuli coming into two eyes, as well as corresponding haptic and audio stimuli. Monkeys have been shown to be adept at combining and leveraging arbitrary sensory feedback information [33].

In deep learning, multi-modal (or *multi-view* learning) learning typically falls into two broad categories: learning a joint representation (fusion of information) and learning an aligned representation (leveraging coordinated information in the multiple views) [11]. A fusion-based approach is beneficial if there is synergistic information available in the different views, while an alignment-based approach is helpful is there is shared information common to the different views (Fig. 5.1). Such a division of information typically affects architectural and model choices: synergistic information requires the information from the different modalities to be fused or combined, whereas shared information often serves as a self-supervised signal that can align information from the different modalities, as in contrastive learning [26,134,135] or correlation based approaches [7].

Critical periods in animals and deep networks: Such architectural considerations often neglect the impact coming from multisensory learning dynamics, where information can be learned at different speeds from each sensor [150]. Indeed, [146] showed that humans and animals are peculiarly sensitive to changes in the distribution of sensory information early in training, in a phenomenon known as *critical periods.* Critical periods have since been described in many different species and sensory organs. For example, barn owls originally exposed to misaligned auditory and visual information cannot properly localize prey [84]. Somewhat surprisingly, similar critical periods for learning have also been observed in deep networks. [2] found that early periods of training were critical for determining the asymptotic network behavior. Additionally, it was found that the timing of regularization was important for determining asymptotic performance [47], with regularization during the initial stages of training having the most influential effect.

Masked/de-noising Autoencoders: Reconstructing an input from a noisy or partial observation has been long used as a form of supervision. Recently, an in part due the successful usage of transformers in language [139] and vision tasks [35], such a pre-training strategy has been successfully applied to text [34] and vision tasks [56]. An extension of this has been recently applied to multi-modal data [9].

Models of learning dynamics We consider two approaches to gain analytic insight into the learning dynamics of deep networks. [117, 118] assume that the input-output mapping is done by a deep linear network. We show that under this model critical periods may exist. [62, 91] assume instead infinitely wide networks, resulting in a model linear with respect to the parameters. In this latter case, no critical period is predicted contradicting our empirical observations on finite networks.

5.2 A model for critical periods in sensor-fusion

We want to establish what is the difference, in terms of learning dynamics, between learning how to use two sources of information at the same time, or learning how to solve a task using each modality separately and then merging the results. In particular we consider the counterfactual question: if we disable sensor A during training, would this change how we learn to use sensor B? To start, let's consider the simple case of a linear regression model

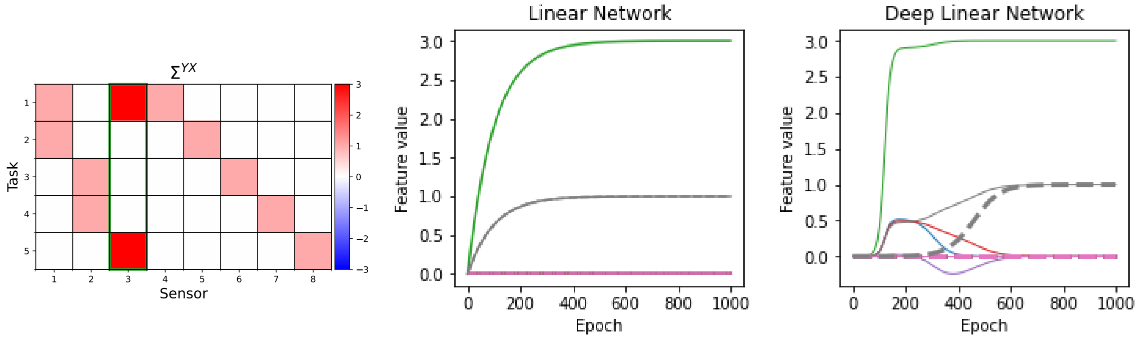

Figure 5.2: **(Left)** Σ^{yx}, with the highlighted green column representing the sensor that was dropped. **(Right)** We show total weights attributed to each feature (shown in different colors) during training in a deep linear network. The solid lines represent the dynamics when training with all features. The dashed lines represent the behavior when training with the green feature disabled. Note that disabling the green feature prevents the gray feature from being learned during the initial transient **(Center)** Same experiment with a shallow linear network. In this case the learning dynamics of the gray feature perfectly overlap in both cases.

$\mathbf{y} = \mathbf{W}\mathbf{x}$ trained with a mean square error loss

$$L = \frac{1}{N}\sum_{i=1}^{N} \frac{1}{2} ||\mathbf{y}^{(i)} - \mathbf{W}\mathbf{x}^{(i)}||^2$$

where $D = \{(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})\}_{i=1}^{N}$ is a training set of i.i.d. samples. In this simplified setting, we consider each component x_k of $\mathbf{x}$ as coming from a different sensor or source. To simplify even further, we assume that the inputs have been whitened, so that the input correlation matrix $\mathbf{\Sigma}^x = \frac{1}{N}\sum_i \mathbf{x}^{(i)}\mathbf{x}^{(i)T} = \mathbf{I}$.

In this case, the learning dynamics of any source is independent from the others. In fact, the gradient of the weight w_{jk} associated to x_k and y_j is given by

$$-\nabla_{w_{jk}} L(\mathbf{W}) = -\nabla_{w_{jk}} \frac{1}{N}\sum_{i=1}^{N} \frac{1}{2} ||\mathbf{y}^{(i)} - \mathbf{W}\mathbf{x}^{(i)}||^2 = \Sigma_{jk}^{yx} - w_{jk}$$

and does not depend on any w_{hl} with $w_{hl} \neq w_{jk}$. The answer to the counterfactual question is thus negative in this setting: adding or removing one source of information (or output) will not change how the model learns to extract information from the other sources. However, we now show that the addition of depth, even *without* taking introducing non-linearities, makes the situation radically different.

To this effect, consider a deep linear network with one hidden layer $\mathbf{y} = \mathbf{W}^2\mathbf{W}^1\mathbf{x}$. This network has the same expressive power (and the same global optimum) as the previous model. However, this introduces a mutual dependency between sensors (due to the shared layer) that can ultimately lead to critical periods in cross-sensor learning. To see this, we use an analytical expression of the learning dynamics for two-layer deep networks [117, 118]. Let $\mathbf{\Sigma}^{yx} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{y}^{(i)}\mathbf{x}^{(i)T}$ be the cross-correlation matrix between the inputs $\mathbf{x}$ and the target vector $\mathbf{y}$[1] and let $\mathbf{\Sigma}^{yx} = USV^T$ be its singular-value decomposition (SVD). [118] shows that the total weight $\mathbf{W}(t) = \mathbf{W}^2(t)\mathbf{W}^1(t)$ assigned to each source at time t during the training can be written as

$$\mathbf{W}(t) = \mathbf{W}^2(t)\mathbf{W}^1(t) = \mathbf{U}\mathbf{A}(t)\mathbf{V}^T \tag{5.1}$$

$$= \sum_{\alpha} a_\alpha(t)\mathbf{u}^\alpha\mathbf{v}^{\alpha T} \tag{5.2}$$

where

$$a_\alpha(t) = \frac{s_\alpha e^{2s_\alpha t/\tau}}{e^{2s_\alpha t/\tau} - 1 + s_\alpha/a_\alpha^0}. \tag{5.3}$$

This leads to non-linear learning dynamics where different features are learned at sharply distinct points in time [118]. Moreover, it leads to entanglement between the learning dynamics of different sources due to the eigenvectors $\mathbf{v}^\alpha$ mixing multiple sources.

Disabling (or adding) a source of information corresponds to removing (or adding) a column to the matrix $\mathbf{\Sigma}^{yx}$, which in turns affects its singular-value decomposition and the corresponding learning dynamics. To see how this change may affect the learning dynamics, in Fig. 5.2 we compare the weights associated to each sensor during training for one particular task. In solid we show the dynamics with all sensors active at the same time. In dashed line we show the dynamics when one of the sensor is disabled. We see that disabling a sensor

[1] Note that $\mathbf{W} = \mathbf{\Sigma}^{yx}$ is also the global minimum of the MSE loss $L = \frac{1}{N}\sum_i \frac{1}{2}||\mathbf{y}^{(i)} - \mathbf{W}\mathbf{x}^{(i)}||^2$.

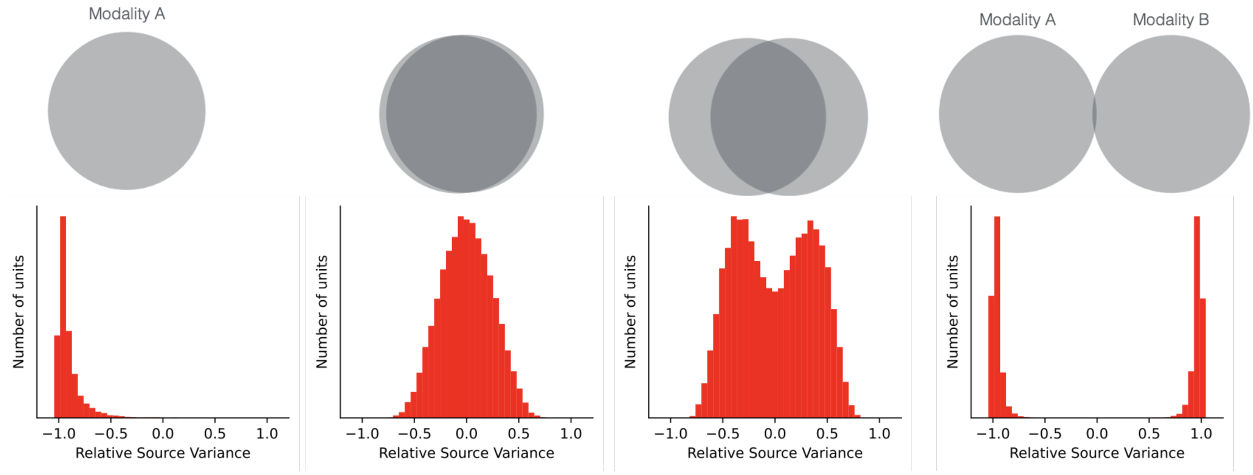

Figure 5.3: **Illustration of RSV distributions and relation to information diagrams. (Left)** Representations that vary predominantly due to one modality. **(Center-Left, Center-right)** All units in the representation vary nearly equally with both modalities. **(Right)** Units in the representation that vary uniquely with each sensor, which is reflected by a polarized RSV distribution.

(green in the figure) can completely inhibit learning of other task-relevant features (e.g., the gray feature) during the initial transient. This should be compared with the learning dynamics of a shallow one-layer network (Fig. 5.2, left) where all task-relevant features are learned at the same time, and where removal of a source does not affect the others.

In deep linear networks, the suboptimal configuration learned during the initial transient is eventually discarded, and the network reverts to the globally optimal solution. In the following we show this is not the case for standard non-linear deep networks. While the initial non-trivial interaction between sources of information remain, the non-linear networks are unable to unlearn the suboptimal configurations learned at the beginning (owing to the highly non-convex landscape). This can result in permanent impairments if a source of information is removed during the initial transient of learning, which reflects the trends observed in critical periods in animals.

5.3 Single Neuron Sensitivity Analysis

Before studying the empirical behavior of real networks on multi-sensor tasks, we should consider how to quantify the effect of a deficit on a down-stream task. One way is to look at the final performance of the model on the task. For example, animals reared with a monocular deprivation deficit have reduced accuracy on a visual acuity test and, similarly, deep networks may show reduced classification accuracy [2]. However, in some cases deficits may not drastically impair the accuracy but may still affect how the model is organized internally. Individuals with strabismus or ambliopia can perform just as well on most tasks, since the individual information coming from each sensor separately is enough to compensate. But the connectivity scheme of the synapses may change so that neurons eventually process only information from one sensor or the other, and not from both together, as observed in individuals without deficits [146].

To understand whether units in a representation of multisensory inputs depend on both sensors or only a particular sensor, we introduce a measure of *Relative Source Variance*. We first define the *Source Variance* (SV) for unit i of a representation due to sensor A, conditioned on an example b as

$$SV_i(A, b) = \mathrm{Var}(f(A, B)_i | B = b), \tag{5.4}$$

where f denotes the mapping from multisensory inputs to the representation and i indexes the unit of the representation. We note that the value of $SV_i(A, b)$ depends on the example b. We use an analogous formula for $SV_i(B, a)$.

Typically, we are interested in the distribution of the Source Variance of the units i in a representation, as a function of many examples a and b. To capture this, we define a notion of Relative Source Variance (RSV) for unit i as:

$$RSV_i(a, b) = \frac{SV_i(A, b) - SV_i(B, a)}{SV_i(A, b) + SV_i(B, a)} \tag{5.5}$$

If the RSV is 1, this means that the unit is only sensitive to sensor A, and if the RSV is -1, the unit is sensitive to sensor B. To compute $SV(A, b)$ (and analogously for $SV(B, a)$) from samples, we fix a sample b, and vary the inputs a, sampling from $a \sim p(a)$. We run this for multiple fixed samples from b, performing the computation over a batch. We perform analogous computations for $SV(B, a)$ We compute the $RSV_i(a, b)$ for all units i from a representation, and for many examples a and b. We then plot the distribution of RSVs, aggregating across all units (see, e.g., Fig. 5.4-5.6). In particular, we track how the distribution changes as a result of sensory deficits and perturbations, as well as how the distribution changes during normal training. Note that $-1 \leq RSV_i(a, b) \leq 1$. If $RSV_i(a, b) = 1$ (or -1) is 1, this means that the unit is only sensitive to sensor A (or B). If $RSV_i(a, b) = 0$ the unit is equally sensitive to both sensors. For controlled simulations (See Appendix D.1.1), we show the variety of distributions of units in a representation that the RSV can measure in Fig. 5.3.

5.4 Critical learning periods in deep multi-sensor networks

In this section, we investigate the learning dynamics of deep networks during the initial learning transient when multiple source of information are present. We evaluate how temporary perturbations of the relation between the two sensors during the training can change the final outcome. To exclude possible confounding factors, in all our experiments, the two input sources are perfectly symmetrical (same data distribution and same informativeness for the task) which ensures that any asymmetry observed in the final model is due to the perturbation.

5.4.1 Inhibition of a weak source

Uncorrected vision problems in an eye during early childhood can cause permanent visual impairment in humans, whereas even after correction the patient only sees through the

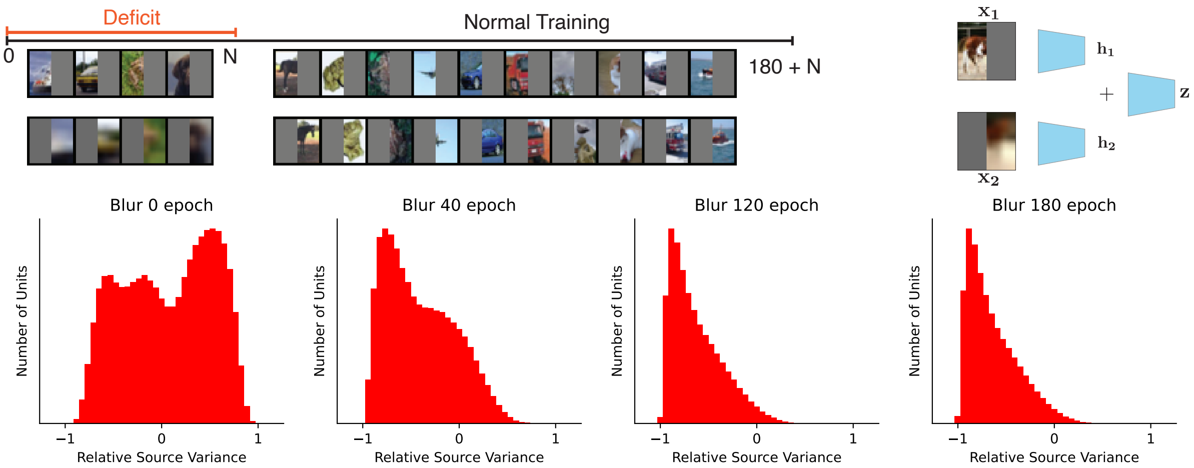

Figure 5.4: **Experimental setup and sensor selectivity as a function of a blurring deficit length. (Top)** In our experiments, we train the network with a deficit (blurred images to one pathway shown here) for the first N epochs, and then continue training with normal images for 180 more epochs. We feed each half of an image to the early stages of a ResNet-18, and then additively combine the representations from both pathways (followed by stages of common processing). We refer to this architecture as *Split-ResNet*. **(Bottom)** RSV distribution of units in last layer representation z for increasing duration of deficit (blur to one pathway) after resumption of normal training. With a sufficiently long deficit, the units in the representation remain only sensitive to the initially uncorrupted pathway, and do not vary with the initially corrupted pathway.

unaffected eye and does not recover vision in the affected eye (ambliopia, or lazy-eye). We explore whether such inhibition of a sensor can happen in DNNs following a similar experimental setup to [2]. To simulate binocular data from single images, we partition each image in a left and right crop and feed each to two separate pathways of the network, which are then fused in an additive manner at a later stage. For each initial pathway, we used the early stages of a ResNet-18 backbone. We then simulate the blurry vision of a weak eye by downsampling the input of the right pathway by 4×, and then resized the image to the original size. After training for t_0 initial epochs with the blur deficit, we remove it and train for further 180 epochs to ensure convergence (see Appendix for details). Here we focus on the simple CIFAR-10 classification dataset, and we later examine different architectures and datasets, and learning approaches.

At the end of the training, both sensors are working well and contain partially disjoint information about the task variable, so the network would benefit from using both of them. However, in Fig. 5.4 (top) we see from the RSV that weakening the right sensor by blurring it during the initial transient will permanently inhibit its use even after removing the deficit. More specifically, at the end of normal training units in the network attend equally to either sensor (leftmost panel). However, in the network trained with a short deficit the neurons only encode information about the "initially good" left sensor (the RSV of the units concentrates around -1, rightmost panel). This mirrors the occular dominance findings present in monkeys with a cataract [146, Fig. 7]. Similarly, the longer the deficit is present during the initial training, the more the downstream performance on the CIFAR-10 classification task is impaired (Fig. 5.5, left). However, the reduction of performance is not as drastic as the RSV change, since the network can compensate and achieve a good accuracy on the task using only the good sensor.

Dependency on depth. In Sect. 5.2 we note that depth is fundamental to make critical periods emerge in multi-sensor networks. We further claim that increasing the depth of the network makes critical periods more evident. Indeed, in Fig. 5.5 (right) we show that increasingly deeper network have increasingly more marked permanent impairment as a result of a temporary deficit.

5.4.2 Learning synergistic information

We have seen that temporary weakening of one sensor may completely inhibit its learning. We now consider an alternative deficit where the two sensors are both working well, but are initially trained on uncorrelated data and only later trained together. This situation is common in every day machine learning, for example when pre-training backbones on different modalities separately (e.g., a text and a vision backbone) and then fine-tuning them together on a downstream task.

Dissociation deficit. To keep the two modalities symmetrical, we consider a similar set up as before where we feed to each pathway of a network the left and a right crop of an image.

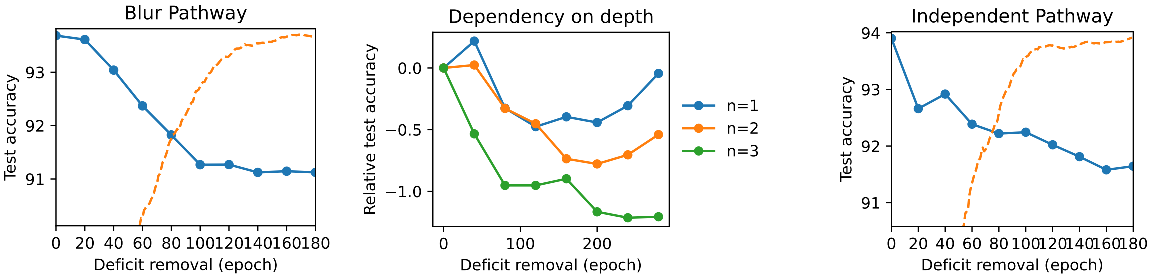

Figure 5.5: **Decrease in downstream performance as a function of the deficit length. (Left)** Final test accuracy when applying a blurring deficit to one pathway of Split-ResNet. Even though the network is exposed to a subsequent number of uncorrupted paired observations, the network cannot later learn to optimally fuse the information. **(Center)** The effect of a deficit is most pronounced when increasing the depth of the network (see Appendix for architecture detail). **(Right)** We also observe a degradation of test performance using a dissociation deficit (feeding uncorrelated views). We note that the effect is less marked than the blurring, due to better ability to compensate.

Both crops are now always full-resolution. However, we introduce a *dissociation* deficit, during which the right crop is sampled from a different image than the left one. During the dissociation, the task is to predict either the class of the left image or the right image with probability 0.5. This deficit removes any synergistic information between the two pathways, but still encourages the two pathways to extract any unique information from the inputs.

We observe that this setup too has a critical period: In Fig. 5.6, we see that, after normal training, the units are equally sensitive to both the left and right inputs (histogram clusters around zero). However, after training with an increasingly longer dissociation deficit, the histogram becomes increasingly polarized around ±1, suggesting that each unit is encoding information only about the right or the left image. This precludes the possibility that the network is extracting synergistic information from the two views (which would entail units that process information from both sensors). This mirrors the ocular dominance representations observed in strabismic monkeys [146, Fig. 10-12]. Similarly to the dissociation deficit, in strabismus, the eyes are not aligned, thus breaking the normal correlation between the views. The dissociation deficit also produces a permanent impairment in the downstream performance (Fig. 5.6, top) but again the effect is not as drastic as in the RSV plot since

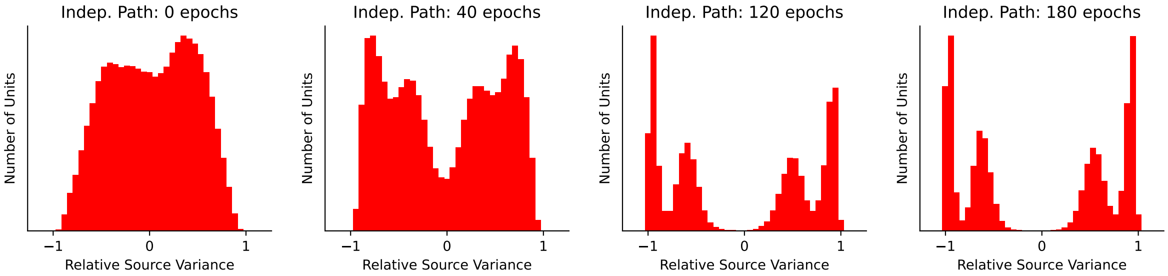

Figure 5.6: **Sensor selectivity as a function of a dissociation deficit length.** We also examined the asymptotic representations and found that, when exposed to a sufficiently long deficit of broken correlations between the views, the network could no longer learn a bimodal distribution that learned common features, but instead resulted in a polarized representation in which units are sensitive to either view (but none to both).

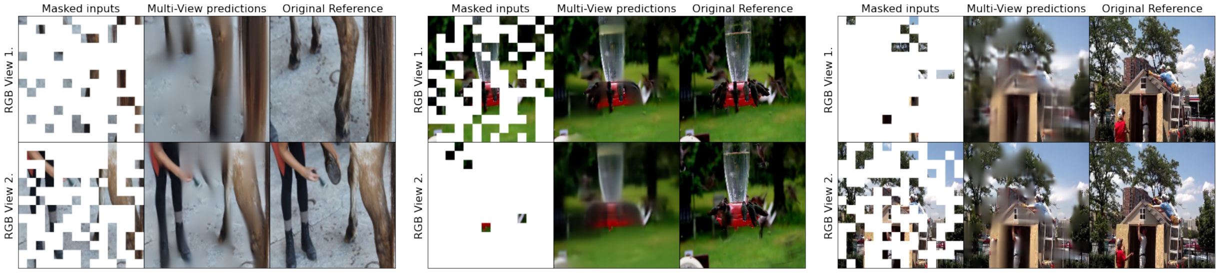

Figure 5.7: **Top.** Example inputs (left column), reconstructions (middle columns), and original targets (right columns) for the Multi-View Transformer, with random sampling of patches from the two views. Note that the model can reconstruct missing information from one view using the other.

the network compensates by using each pathway separately (albeit synergistic information is lost).

5.4.3 Synergistic information in videos

So far we have seen that supervised deep networks, similar to humans and animals, have critical periods for learning correspondences between multi-view data. We confirmed this both at the behavioural (measured in terms of performance and visual acuity for the deep networks and animals respectively) and at the representation level, quantified by the neuron sensitivity. We now investigate whether such phenomenon generalize across learning strategies, architecture, and tasks.

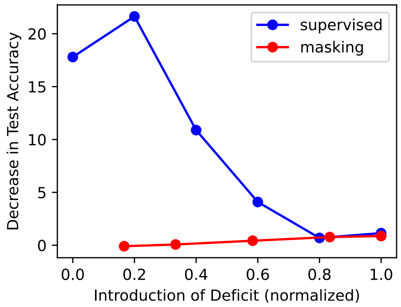

Figure 5.8: **Masking objective with cross-sensor reconstruction loss does not exhibit a critical learning period.** We found that the unsupervised network was much more robust to perturbations early in the training (red trace), whereas that supervised objective was not (blue trace).

Multi-View Transformer. Aside from integrating information from different sensors, animals and artificial networks need to be able to integrate information through time. We can think of frames of a video as being different views or sources of information that are correlated through time, and we can study how a network learns to integrate such information. We opted to use a more flexible transformer-based visual architecture, which has recently achieved state-of-the-art results in computer vision tasks [35, 56], and language tasks [34, 139]. Visual transformers are typically trained either with a supervised loss [35] or a masking-based objective, followed by fine-tuning [56]. We focus now on the first case, and analyze the second in the next section. In order to process multiple frames of a video, we use a modified Multi-Modal Masked Auto-Encoder [9], which we train in a fully supervised fashion. We refer to this as a *Multi-View Transformer*.

To capture multiple views of a scene, we opted to use the the Kinetics Action classification video dataset [23], which consist in classifying one of 400 possible actions given a video clip. To adapt the task to our setting, from each video we select two random frames that are a multiple of 0.33 seconds apart to comprise our two views, and feed them to the *Multi-View Transformer*. Due to their temporal correlation, the two frames together contain more information (the motion) than either frame individually. We use a similar dissociation deficit as in the previous section: During the dissociation deficit period, we sampled the two frames from independent videos in order to break their temporal correlation. In this case, the classification label coming from either view with $p = 0.5$ (see Appendix for training details).

Even on a largely different architectures (transformer instead of ResNet) and a more complex task (action classification on natural video instead of CIFAR-10), in Fig. 5.8 we observe the same trends as in the previous section. Training with a temporary dissociation deficit permanently prevents the network from extracting synergistic temporal information from the frames. Unlike in the previous experiment, since the synergistic information is fundamental for the action classification task, the network cannot compensate the deficit and perturbations during the critical period also results in an harsh decrease of up to 20% in the final test accuracy (Fig. 5.8, left).

5.4.4 Overcoming critical periods with cross-sensor reconstruction

Our previous experiments suggest that critical periods can be caused by competition between sensors which increases the selectivity of the units. If this is the case, we may hypothesize that training adding a cross-sensor reconstruction objective may help forcing the unit to learn how to encode cross-sensor information. To test this hypothesis, we train the Multi-View transformer of Sec. 5.4.3 using the cross-sensor masking-based reconstruction objective of [9] and compare it with the supervised case. The self-supervised masked-image reconstruction task could encourage correspondences to be learned (if un-occluded parts of one view are helpful for reconstructing the other view), and may force learning synergistic information irrespective of the initial transient. In Fig. 5.7, we show that indeed the masking-based pre-training is successful in using information from one source to predict masked patches of the other.

We train using the same protocol as Sec. 5.4.3 to pre-train the Multi-View Transformer using the cross-reconstruction objective. We then subsequently fine-tuned for 20 epochs on the downstream supervised classification tasks (see Appendix for details). In Fig. 5.8 we see that the unsupervised network was much more robust to perturbations early in the training, whereas that supervised objective was not. To understand whether such robustness was due to large changes to the representation when fine-tuning, we applied the RSV on the output of the encoder's representation and found that while the resulting distribution became slightly more symmetrically balanced, it retained a similar bimodal distribution to the pre-trained representation. (Fig. D.6).

5.5 Discussion

We have shown – in a variety of architectures and tasks – the existence of critical learning periods for multi-source integration: a temporary initial perturbations of an input source may permanently inhibit that source, or prevent the model from learning how to combine multiple sources. These trends replicate similar phenomena in animals, and point to the underlying complexity and brittleness of the learning dynamics that allow a network (or an animal) to fuse information. To simplify the analysis of the learning dynamics, we focused on tasks with homogeneous sources (stereo, video). We leave to future work to further study the role played by the asymmetry between sources (e.g., different informativeness or ease). Our theoretical and empirical analysis leads to several suggestions: Pre-training different backbones separately on each modality, as advocated in some foundational model, may yield representations that ultimately fail to encode synergistic information. Instead, training should be performed across modalities at the outset. On the theoretical side, our work suggests that analysis "at convergence" of the learning dynamics of a network are irrelevant for sensor fusion, as their fate is sealed during the initial transient learning. It also suggests that conclusions drawn from wide and shallow networks may not transfer to deep networks in current use.

Chapter 6

Redundant Information Neural Estimation

We introduce the Redundant Information Neural Estimator (RINE), a method that allows efficient estimation for the component of information about a target variable that is common to a set of sources, known as the "redundant information." We show that existing definitions of the redundant information can be recast in terms of an optimization over a family of functions. In contrast to previous information decompositions, which can only be evaluated for discrete variables over small alphabets, we show that optimizing over functions enables the approximation of the redundant information for high-dimensional and continuous predictors. We demonstrate this on high-dimensional image classification and motor-neuroscience tasks.

6.1 Introduction

Given a set of sources $X_1, \dots, X_n$ and a target variable Y, we study how information about the target Y is distributed among the sources: different sources may contain information that no other source has ("unique information"), contain information that is common to other sources ("redundant information"), or contain complementary information that is only accessible when considered jointly with other sources ("synergistic information"). Such a decomposition of the information across the sources can inform the design of multi-sensor systems (e.g., to reduce redundancy between sensors), or support research in neuroscience, where neural activity is recorded from two areas during a behavior. For example, a detailed understanding of the role and relationship between brain areas during a task requires understanding how much unique information about the behavior is provided by each area that is not available to the other area, how much information is redundant (or common) to both areas, and how much additional information is present when considering the brain areas jointly (i.e., information about the behavior that is not available when considering each area independently).

Standard information–theoretic quantities conflate these notions of information. [147] therefore proposed the Partial Information Decomposition (PID), which provides a principled framework for decomposing how the information about a target variable is distributed among a set of sources. For example, for two sources X_1 and X_2, the PID is given by

$$I(X_1, X_2; Y) = UI(X_1; Y) + SI + UI(X_2; Y) + I_\cap, \tag{6.1}$$

where UI represents the "unique" information, SI the "synergistic" information, and $I_\cap$ represents the redundant information, shown in Figure E.1. We provide details in Appendix E.1.1, describing how standard information–theoretic quantities, such as the mutual information $I(X_1; Y)$ and conditional mutual information $I(X_2; Y|X_1)$, are decomposed in terms of the PID constituents.

Despite efforts and proposals for defining the constituents [12, 18, 52–54, 87], existing definitions involve difficult optimization problems and remain only feasible in low-dimensional spaces, limiting their practical applications. One way to sidestep these difficult optimization problems is to assume a joint Gaussian distribution over the observations [16], and this approach has been applied to real-world problems [38]. To enable optimization for high-dimensional problems with arbitrary distributions, we reformulate the redundant information through a variational optimization problem over a restricted family of functions. We show that our formulation generalizes existing notions of redundant information. Additionally, we show that it correctly computes the redundant information on canonical low-dimensional examples and demonstrate that it can be used to compute the redundant information between different sources in a higher-dimensional image classification and motor-neuroscience task. Importantly, RINE is computed using samples from an underlying distribution, which does not need to be known.

Through RINE, we introduce a similarity metric between sources which is task dependent, applicable to continuous or discrete sources, invariant to reparametrizations, and invariant to addition of extraneous or noisy data.

6.2 Related Work

Central to the PID is the notion of redundant information $I_\cap$, and much of the work surrounding the PID has focused on specifying the desirable properties that a notion of redundancy should follow. Although there has been some disagreement as to which properties a notion of redundancy should follow [54, 87, 147], the following properties are widely accepted:

- Symmetry: $I_\cap(X_1; \ldots; X_n \to Y)$ is invariant to the permutation of $X_1, \ldots, X_n$.
- Self-redundancy: $I_\cap(X_1 \to Y) = I(X_1; Y)$.
- Monotonicity: $I_\cap(X_1; \ldots; X_n \to Y) \leq I_\cap(X_1; \ldots; X_{n-1} \to Y)$.

Several notions of redundancy have been proposed that satisfy these requirements, although we emphasize that these notions were generally not defined with efficient computation in mind.

[52] proposed a redundancy measure $I_\cap^\wedge$, defined through the optimization problem:

$$I_\cap^\wedge(X_1; \ldots; X_n \to Y) := \max_Q \; I(Y; Q) \quad \text{s.t.} \quad \forall i \, \exists f_i \; Q = f_i(X_i) \tag{6.2}$$

where Q is a random variable and f_i is a deterministic function. The redundant information is thus defined as the maximum information that a random variable Q, which is a deterministic function of all X_i, has about Y. This means that Q captures a component of information common to the sources X_i.

An alternative notion of redundant information $I_\cap^{\mathrm{GH}}$ [13,53] with a less restrictive constraint is defined in terms of the following optimization problem:

$$I_\cap^{\mathrm{GH}}(X_1; \ldots ; X_n \to Y) := \max_Q I(Y;Q) \quad \text{s.t.}\ \forall i \quad I(Y;Q|X_i) = 0. \tag{6.3}$$

$I_\cap^{\mathrm{GH}}$ reflects the maximum information between Y and a random variable Q such that $Y - X_i - Q$ forms a Markov chain for all X_i, relaxing the constraint that Q needs to be a deterministic function of X_i.

We show in Section 6.3 that our definition of redundant information is a generalization of $I_\cap^\wedge$ and can be extended to compute $I_\cap^{\mathrm{GH}}$.

The main hurdle in applying these notions of information to practical problems is the difficulty of optimizing over all possible random variables Q in a high-dimensional setting. Moreover, even if that was possible, such unconstrained optimization could recover degenerate forms of redundant information that may not be readily “accessible” to any realistic decoder. In the next section we address both concerns by moving from the notion of Shannon Information to the more general notion of Usable Information [36, 78, 151].

6.2.1 Usable Information in a Random Variable

An orthogonal line of recent work has looked at defining and computing the “usable” information $I_u(X;Y)$ that a random variable X has about Y [36, 78, 151]. This aims to capture the fact that not all information contained in a signal can be used for inference by a restricted family of functions. Given a family of decoders $\mathcal{V} \subseteq \mathcal{U} = \{f : \mathcal{X} \to \mathcal{P}(\mathcal{Y})\}$, the usable information that X has about Y is defined as

$$I_u(X;Y) = H(Y) - H_{\mathcal{V}}(Y|X), \tag{6.4}$$

where $H_{\mathcal{V}}(Y|X)$ is defined as

$$H_{\mathcal{V}}(Y|X) = \inf_{f \in \mathcal{V}} \mathbb{E}_{x,y \sim X,Y} \left[-\log f(y|x) \right]. \tag{6.5}$$

Thus, the "usable" information differs from Shannon's mutual information in that it involves learning a decoder function f in a model family $\mathcal{V}$, which is a subset of all possible decoders $\mathcal{U}$. When the "usable" information is defined such that the model family corresponds to the universal model family, the definition recovers Shannon's mutual information, $I(X;Y) = H(Y) - H_{\mathcal{U}}(Y|X)$. However, in many cases, the "usable information" is closer to our intuitive notion of information, reflecting the amount of information that a learned decoder, as opposed to the optimal decoder, can extract under computational constraints [151]. We extend these ideas to compute the "usable redundant information" in the next section.

6.3 Redundant Information Neural Estimator

We introduce the Redundant Information Neural Estimator (RINE), a method that enables the approximation of the redundant information that high-dimensional sources contain about a target variable. In addition to being central for the PID, the redundant information also has direct applicability in that it provides a task-dependent similarity metric that is robust to noise and extraneous input, as we later show in Section 6.4.4.

Our approximation leverages the insight that existing definitions of redundancy can be recast in terms of a more general optimization over a family of functions, similar to how the "usable information" was defined above. To this end, given two sources, we define a notion of redundancy, RINE, through the following optimization over models $f_1, f_2 \in \mathcal{V} \subseteq \mathcal{U} = \{f : \mathcal{X} \to \mathcal{P}(\mathcal{Y})\}$.

$$L_\cap^{\mathcal{V}}(X_1; X_2 \to Y) := \min_{f_1, f_2 \in \mathcal{V}} \frac{1}{2}\big[H_{f_1}(Y|X_1) + H_{f_2}(Y|X_2)\big] \tag{6.6}$$

$$\text{s.t.} \;\; D(f_1, f_2) = 0 \tag{6.7}$$

$$I_\cap^{\mathcal{V}}(X_1; X_2 \to Y) := H(Y) - L_\cap^{\mathcal{V}}, \tag{6.8}$$

where $H_{f_i}(Y|X_i)$ denotes the cross-entropy when predicting Y using the decoder $f_i(y|x)$ and $D(f_1, f_2) = \mathbb{E}_{x_1,x_2}\big[\|f_1(y|x_1) - f_2(y|x_2)\}\|_1\big]$ denotes the expected difference of the predictions of the two decoders. Importantly, the model family $\mathcal{V}$ can be parametrized by neural networks, enabling optimization over the two model families with backpropagation. In general, one can optimize over different model families $\mathcal{V}_1$ and $\mathcal{V}_2$, but for notational simplicity we assume we optimize over the same model family $\mathcal{V}$ in the paper. Note that here we constrained the predictions directly, as opposed to using an intermediate random variable Q. In contrast, direct optimization of Equations (6.2) and (6.3) is only feasible for discrete sources with small alphabets [87]. Our formulation can be naturally extended to n sources (Appendix E.1.8) and other divergence measures between decoders. Since our formulation involves learning decoders that map the sources to target predictions, the learned decoder can safely ignore task-irrelevant variability, such as noise, as we demonstrate in Section 6.4.4.

To solve the constrained minimization problem in Equation (6.6-6.7), we can minimize the corresponding Lagrangian:

$$L_\cap^{\mathcal{V}}(X_1; X_2 \to Y, \beta) := \min_{f_1, f_2 \in \mathcal{V}} \frac{1}{2}\big[H_{f_1}(Y|X_1) + H_{f_2}(Y|X_2)\big] + \beta D(f_1, f_2). \tag{6.9}$$

When $\beta \to \infty$ the solution to the Lagrangian is such that $D(f_1, f_2) \to 0$, thus satisfying the constraints of the original problem. In practice, when optimizing this problem with deep networks, we found it useful to start the optimization with a low value of β, and then increase it slowly during training to some sufficiently high value ($\beta = 50$ in most of our experiments).

Note that while $H(Y)$ does not appear in the Lagrangian, it is still used to compute $I_\cap^\mathcal{V}$, as in Equation (6.8). The Lagrangian is optimized, using *samples* from an underlying distribution $p(X_1, X_2, Y)$; importantly, the underlying distribution can be continuous or discrete.

Our definition of $\mathcal{V}$-redundant information (Equation (6.8)) is a generalization of $I_\cap^\wedge$ (Section 6.2) as shown by the following proposition:

Proposition 1 (Appendix 6.7)**.** *Let $\mathcal{V} = \{f : \mathcal{X} \to \mathcal{P}(\mathcal{Y})\}$ consist of the family of deterministic functions from X to distributions over $\mathcal{Y}$. Then $I_\cap^\mathcal{V} = I_\cap^\wedge$.*

Our formulation involving a constrained optimization over a family of functions is general: indeed, optimizing over stochastic functions or channels with an appropriate constraint can recover $I_\cap^{\text{GH}}$ or $I_\cap^K$ [87] (described in the Appendix) but the computation in practice becomes more difficult.

Our definition of redundant information is also invariant to reparametrization of the sources as shown by the following proposition:

Proposition 2 (Appendix 6.7)**.** *Let $t : \mathcal{X} \to \mathcal{X}$ be any invertible transformation in $\mathcal{V}$. Then,*

$$I_\cap^\mathcal{V}(X_1; X_2 \to Y) = I_\cap^\mathcal{V}(t_1(X_1); t_2(X_2) \to Y). \tag{6.10}$$

Note that when $\mathcal{V} = \mathcal{U}$, $I_\cap^\mathcal{V}$ is invariant to *any* invertible transformation. In practice, when optimizing over a subset $\mathcal{V} \subseteq \mathcal{U}$, our definition is invariant to transformations that preserve the usable information (this accounts for practical transformations, for example the reflection or rotation of images). As an example of transformations that lie in $\mathcal{V}$, consider the case in which $\mathcal{V}$ is a set of linear decoders. This model family is closed under any linear transformation $t(X)$ applied to the source, since the composition of linear functions is still a linear function.

As an additional example, the family of fully connected networks is closed to permutations of the pixels of an image since there exists a corresponding network $f \in \mathcal{V}$ that would behave the same on the transformed image. The family of convolutional networks, for a given architecture on the other hand, is not closed under arbitrary transformations of the pixels, but it is closed, e.g., under rotations/flips of the image.

In contrast, complex transformations such as encryption or decryption (which preserve Shannon's mutual information) can decrease or increase respectively the usable information content with respect to the model family $\mathcal{V}$. Arguably, such complex transformations do modify the "information content" or the "usable information" (in this case measured with respect to $\mathcal{V}$) even though they do not affect Shannon's mutual information (which assumes an optimal decoder in $\mathcal{U}$ that may not be in $\mathcal{V}$).

6.3.1 Implementation Details

In our experiments, we optimize over a model family $\mathcal{V}$ of deep neural networks, using gradient descent. In general, the model family to optimize over should be selected such that it is not so complicated that it overfits to spurious features of the finite training set, but has high enough capacity to learn the mapping from source to target.

We parametrize the distribution $f_i(y|x)$ in Equation (6.9), using a deep neural network. In particular, in the case that y is discrete (which is the case in all our experiments), the distribution $f_i(y|x) = \text{softmax}(h_{w_i}(x))$ is parametrized as the softmax of the output of a deep network with weights w_i. In this case, the distance $D(f_1, f_2)$ can be readily computed as the average L_1 distance between the softmax outputs of the two networks $h_{w_1}(x_1)$ and $h_{w_2}(x_2)$ for different inputs x_1 and x_2. If the task label y is continuous, for example in a regression problem, one can parametrize $f_i(y|x) = \mathcal{N}(h_{w_i}(x), \sigma^2 I)$ using a Normal distribution whose means is the output of a DNN. We optimize over the weights parametrizing all $f_i(y|x)$ jointly, and we show a schematic of our architecture in Figure 6.1.

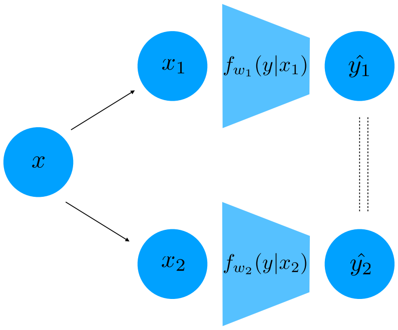

Figure 6.1: A schematic of our architecture for two sources X_1 and X_2. Note that the two networks do *not* share weights. The dashed lines indicate that the predictions are constrained to be similar.

Once we parametrize f_1 and f_2, we need to optimize the weights in order to minimize the Lagrangian in Equation (6.9). We do so using Adam [73] or stochastic gradient descent, depending on the experiment. For images we optimize over ResNet-18's [55], and for other tasks we optimize over fully-connected networks. The hyperparameter β needs to be high enough to ensure that the constraint is approximately satisfied. However, we found that starting the optimization with a very high value for β can destabilize the training and make the network converge to a trivial solution, where it outputs a constant function (which trivially satisfies the constraint). Instead, we use a reverse-annealing scheme, where we start with a low beta and then slowly increase it during training up to the designated value (Appendix E.1.3). A similar strategy is also used (albeit in a different context) in optimizing β-VAEs [21].

6.4 Results

We apply our method to estimate the redundant information on canonical examples that were previously used to study the PID, and then demonstrate the ability to compute the redundant information for problems where the predictors are high dimensional.

6.4.1 Canonical examples

We first describe the results of our method on standard canonical examples that have been previously used to study the PID. They are particularly appealing because for these examples it is possible to ascertain ground truth values for the decomposition. Additionally, the predictors are low dimensional and have been previously studied, allowing us to compare our variational approximation. We describe the tasks, the values of the sources X_1, X_2, and the target Y for in Section 6.6. Briefly, in the UNQ task, each input X_1 and X_2 contributes 1 bit of unique information about the output and there is no redundant information. In the AND task, the redundant information should be in the interval [0, 0.311] depending on the stringency of the notion of redundancy used [53]. When using deterministic decoders, as we do, we expect the redundant information to be 0 bits (not 0.311 bits). The RDNXOR task corresponds to a redundant XOR task, where there is 1 bit of redundant and 1 bit of synergistic information. Finally the IMPERFECTRDN task corresponds to the case where X_1 fully specifies the output, with X_2 having a small chance of flipping one of the bits. Hence, there should be 0.99 bits of redundant information. As we show in Table 6.1, RINE (optimizing with a deterministic family; Appendix E.1.4) recovers the desired values on all these canonical examples.

	True	$I_\cap^\wedge$	$I_\cap^{\text{GH}}$	$I_\cap^{\mathcal{V}}$ ($\beta = 15$)
UNQ [T6.2]	0	0	0	**0.006 (0.016)**
AND [T6.3]	[0, 0.311]	0	0	**0.007 (0.001)**
RDNXOR [T6.4]	1	1	1	**0.977 (9e-4)**
IMPERFECTRDN [T6.5]	0.99	0	0.99	**0.984 (0.002)**

Table 6.1: Comparison of redundancy measures on canonical examples. Quantities are in bits, and $I_\cap^{\mathcal{V}}$ denotes our variational approximation (for $\beta = 15$). The mean and standard deviation (inside parentheses) are reported over 5 different initializations. $I_\cap^\wedge$ denotes the redundant information in [52] and $I_\cap^{\text{GH}}$ denotes the redundant information in [53]. Note that [87] computed $I_\cap^{\text{GH}}$ for the AND operation and got 0.123 bits, as opposed to the 0 bits reported in [53]. We do this computation for different values of β in Table E.1.

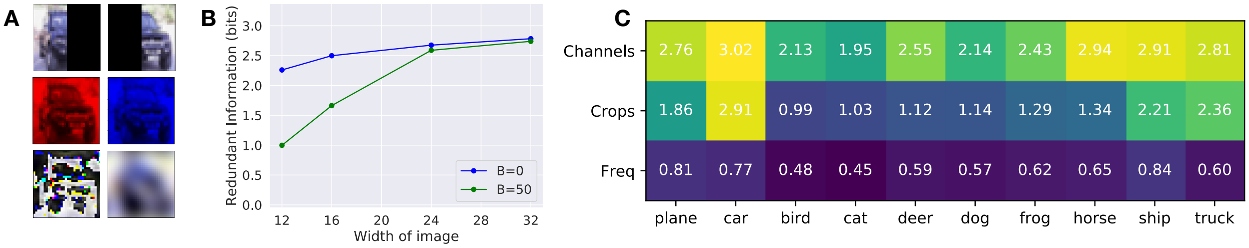

Figure 6.2: **(A)** Examples of the different views of the image used in the experiment. **(B)** Redundant information of different crops of CIFAR-10 images. Redundant information as a function of the width of each partition, for different values of β. A width of 16 means that both X_1 and X_2 is a 16 x 32 image. The images begin from opposing sides, so in the case of the 16 x 32 image, there is no overlap between X_1 and X_2. As the amount of overlap increases, the redundant information increases. The distance function used was the L_1 norm of the difference. **(C)** Per class redundant information for different channels, crops, and frequency decompositions, with $\beta = 50$ used in the optimization.

6.4.2 Redundant information in different views of high-dimensional images

To the best of our knowledge, computations of redundant information have been limited to predictors that were one-dimensional [12, 52, 53, 87]. We now show the ability to compute the redundant information when the predictors are high dimensional. We focus on the ability to predict discrete target classes, corresponding to a standard classification setting. In particular, we analyze redundant information between left and right crops of an images (to simulate a system with two stereo cameras), between different color channels of an images (sensors with different frequency bands), and finally between high and low spatial frequency components of an images.

We analyze the redundant information between different views of the same CIFAR-10 image (Figure 6.2), by optimizing over a model family of ResNet-18's [55], described in Appendix E.1.6. In particular, we split the image in two crops, a left crop X_1 containing all pixels in the first w columns, and a right crop X_2 containing all pixels in the last w columns (Fig E.3). Intuitively, we expect that as the width of the crop w increases, the two views will overlap more, and the redundant information that they have about the task will increase.

Indeed, this is what we observe in Figure 6.2 (left).

We next study the redundant information between different sensor modalities. In particular, we decompose the images into different color channels (X_1 = red channel and X_2 = blue channel), and frequencies (X_1 = low-pass filter and X_2 = high-pass filter). We show example images in Fig E.3. As expected, different color channels have highly redundant information about the task (Figure 6.2, right) except when discriminating classes (like dogs and cats) where precise color information (coming from using jointly the two channels synergistically) may prove useful. On the contrary, the high-frequency and low-frequency spectrum of the image has a lower amount of redundant information, which is also expected, since the high and low-frequencies carry complementary information. We also observe that left and right crop of the image are more redundant for pictures of cars than other classes. This is consistent with the fact that many images of cars in CIFAR-10 are symmetric frontal pictures of cars, and can easily be classified using just half of the image. Overall, there is more redundant information between channels, then crops, then frequencies. Together, these results show we can compute the redundant information of high dimensional sources, providing an empirical validation for our approximation and a scalable approach to apply in other domains.

6.4.3 Neural Data Decoding

We next applied our framework to analyze how information is encoded in motor-related cortical regions of monkeys during the preparatory period of a center-out reaching task [115]. Our goal was to confirm prior hypotheses known about motor cortical encoding from the literature. In the center-out reaching task, there are 8 target locations and the monkey needs to make a reach to one of the targets depending on a cue (Fig 6.3, left). Our dataset consists of a population recording of spike trains from 97 neurons in the dorsal premotor cortex (PMd) during trials that were 700ms long. Each trial comprises a 200ms baseline period (before the reach target turned on) and a 500ms preparatory (planning) period after the reach target

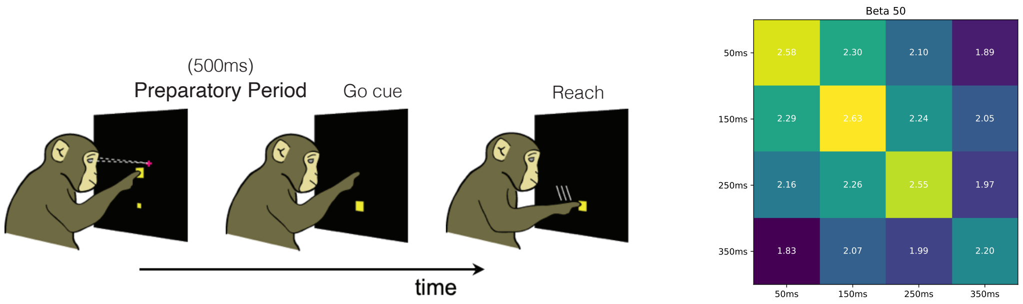

Figure 6.3: (Left) Schematic of delayed-center-out reaching task. There are 8 possible target locations (equally spaced), one of which is shown. Neural data is recorded from the premotor cortex of a monkey using 97 electrodes. (Right) Redundant information between short disjoint time windows during the preparatory period, before a reach can be initiated. Even before the reach is initiated, the target location can be decoded from the premotor cortex using neural data averaged in a short 100ms time window. In the confusion matrix, adjacent time bins have higher redundant information about the target location during the preparatory period, reflecting that the encoding of the target location is more similar in adjacent time windows.

turned on but before the monkey can initiate a reach. Both our training and testing dataset consist of 91 reaches to each target. During the 500 ms preparatory period, the monkey prepared to reach towards a target but did not initiate the reach, enabling us to study the PMd neural representation of the planned reach to the target.

First, we used RINE to compute redundant information of PMd activity over time during the delay period. PMd activity is known to be relatively static during the delay period, approaching a stable attractor state [122]. We therefore expect the redundant information between adjacent time windows to be high. To quantify this, we evaluated the redundant information between different time segments of length 100 ms, beginning 50 ms after the beginning of the preparatory period. For our feature vector, we counted the total number of spikes for each neuron during the time segment. We note that even in the relatively short window of 100 ms, there is a significant amount of usable information about the target in the recorded population of neurons, since the diagonal elements of Fig. 6.3 are close to 3 bits. This is consistent with prior studies that show small windows of preparatory activity can be used to decode target identity [115, 116]. We also found that adjacent time windows contain higher redundant information (closer to the 3 bits), consistent with the idea that the encoding of the target between adjacent time windows are more similar [46]. Together, these results show that RINE computes redundant information values consistent with results reported in the literature showing that PMd representations stably encode a planned target.

Second, we used RINE to study the redundant information between the neural activity recorded on different days and between subjects. We analyzed data from another delayed-center-out task with 8 targets and a variable $400 - 800$ms delay period, during which the monkey could prepare to reach to the target, but was not allowed to initiate the reach (Appendix E.1.7). We examined the redundant information about the target location in the premotor cortex on different sessions and between the different monkeys, Monkey J and Monkey R. When data came from different sessions, we generated a surrogate dataset by conditioning on the desired target reach, ensuring that X_1 and X_2 corresponded to the same target Y. At an extreme, if we could only decode 4 of the 8 targets from Monkey J's PMd activity and the other 4 of the 8 targets from Monkey R's PMd activity, there would be no redundant information in the recorded PMd activity. Our results are shown in Fig. 6.4 (left). Since the PMd electrodes randomly sample tens of neurons out of hundreds of millions in motor cortex, we expect the redundant information between Monkey J and Monkey R PMd recordings to be relatively low. We also expect the redundant information across sessions for the same monkey to be higher, since the electrodes are relatively stable across days [104]. RINE calculations are consistent with these prior expectations. We find the redundant information is higher between sessions recorded from the same monkey than between sessions recorded from different monkeys.

Finally, we quantified redundant information between PMd and the primary motor cortex (M1) during the delay period (Fig 6.4, center). We expect redundant information to be relatively low; whereas PMd strongly represents the motor plan through an attractor state, activity in M1 is more strongly implicated in generating movements with dynamic activity [27]. We find that the values of the redundant information between PMd and M1 are low (0.4 to 0.7 bits), indicating that there is little redundant encoding of target information during

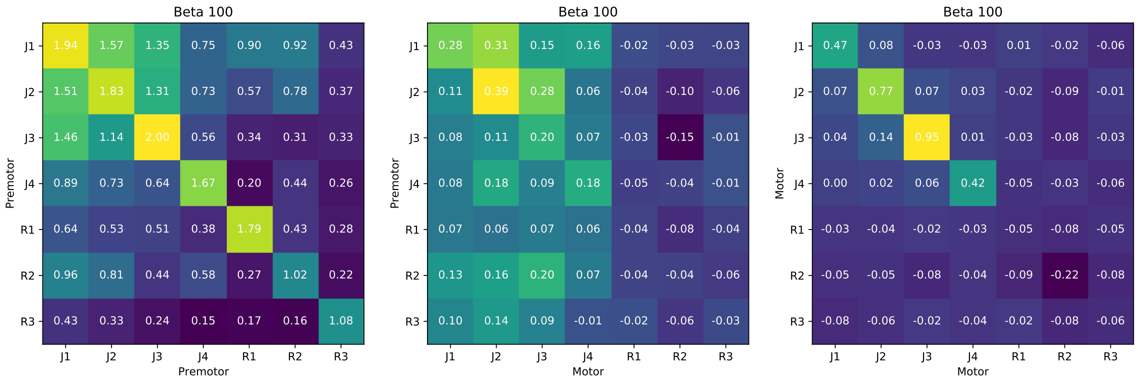

Figure 6.4: Neural decoding confusion matrix for different monkeys and different sessions (left), motor and premotor cortex (middle) and between motor cortex across different monkeys and sessions (right).

the delay period between premotor and motor cortex, even for the same monkey. This is consistent with these two regions having distinct roles related to the initiation and execution of movement [122]. One explanation for having low redundant information between the motor and the premotor cortex during the preparatory period is that there is little encoding of the target location in the motor cortex during the preparatory period, and that the motor cortex serves a role more related to producing appropriate muscle activity. Similar to how we analyzed the redundant information between the premotor cortex, we analyzed the redundant information between the motor cortex across sessions (Fig 6.4, right). We find that there is little information about the planned target in M1 activity for both monkeys (far from 3 bits). Monkey R's M1 information is particularly low due to M1 electrodes recording from very few neurons. The lower values of redundant information between motor cortices compared to premotor cortices implies there is less information in M1 than PMd about the target during the preparatory, consistent with prior literature.

6.4.4 Advantage of redundant information as a task-related similarity measure

How does the notion of redundancy compare to other similarity metrics such as $I(X_1; X_2)$ or the cosine similarity between X_1 and X_2? Critically, both measures are agnostic to a target

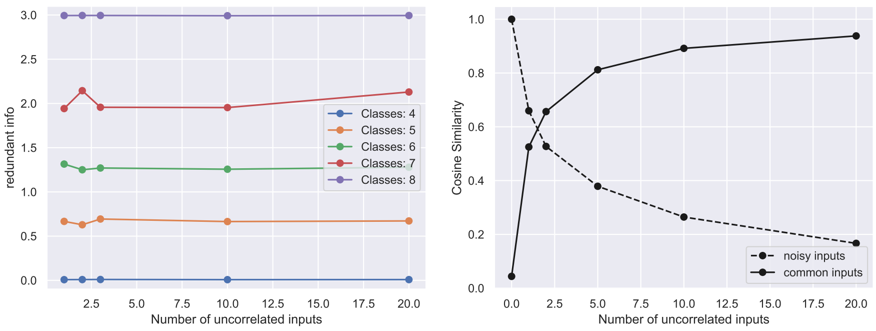

Figure 6.5: Comparison of redundant information against cosine similarity metric. **(Left)** The redundant information is invariant to the number of uncorrelated inputs, and we validate empirically that our approximation of redundant information remains approximately constant with increasing number of uncorrelated inputs. **(Right)** In contrast, alternative similarity metrics like the cosine similarity decreases with increasing number of random noisy units (dashed lines) or increases with correlated non-task units, (solid line).

Y, whereas the redundant information reflects the common information about the target Y. Hence, the redundant information is unaffected by factors of variation that are either pure noise, or caused by target-independent factors, but these factors of variation affect other similarity metrics. This may be particularly important in neuroscience, since recordings from different areas or neurons contain significant noise or non-task variability that can affect similarity metrics. We design a synthetic task to showcase these effects. The task is similar to the neural center-out reaching task, with 8 classes. The task was designed so that each input X_1 and X_2 contain information about n classes, with the minimum overlap between the classes specified: when each input specifies $n = 4$ classes, there are no classes that are encoded by both X_1 and X_2 (hence 0 bits of redundant information), and with $n = 5$ classes it means that 2 common classes are encoded by the the two inputs. Full details are provided in Appendix E.1.5. We swept the number of classes n that each input specified from 4 to 8.

In Fig 6.5 (left), we show that the redundant information increases with increasing overlap between the classes specified by the input, but the redundant information is unaffected by adding units that are uncorrelated with the target, evidenced by approximately flat lines for each value of n. In contrast, the cosine similarity is affected by the addition of such units (Fig 6.5, right). Adding noisy inputs decreases the cosine similarity, whereas the addition of shared non-task-related inputs increase the cosine similarity (Appendix E.1.5). Thus, the important distinction of the redundant information in comparison to direct similarity metrics applied on the inputs is that the redundant captures information in sources about a *target* Y, whereas direct similarity metrics applied on the sources are agnostic to the target or task Y.

6.5 Discussion

Central to the Partial Information Decomposition, the notion of redundant information offers promise for characterizing the component of task-related information present across a set of sources. Despite its appeal for providing a more fine-grained depiction of the information content of multiple sources, it has proven difficult to compute in high-dimensions, limiting widespread adoption. Here, we show that existing definitions of redundancy can be recast in terms of optimization over a family of deterministic or stochastic functions. By optimizing over a subset of these functions, we show empirically that we can recover the redundant information on simple benchmark tasks and that we can indeed approximate the redundant information for high-dimensional predictors.

Although our approach correctly computes the redundant information on canonical examples as well as provides intuitive values on higher-dimensional examples when ground-truth values are unavailable, with all optimization of overparametrized networks on a finite training set, there is the possibility of overfitting to features in the training set and having poor generalization on a test set. This is not just a problem for our method but is a general feature of many deep learning systems, and it is common to use regularization to help mitigate this. PAC-style bounds on the test set risk that factor in the finite nature of the training set exist [138] and it would be interesting to derive similar bounds that could be applied on the distance term to bound the deviation on the test set. Additionally, future work should investigate the properties arising from the choice of distance term, since other distance terms could have preferable optimization properties or desirable information-theoretic interpretations, especially when it is non-zero. Last, the choice of beta-schedule beginning with a small value and increasing during training was important (Fig E.2), and may need to be tuned to a particular task.

Our approach only provides a value summarizing how much of the information in a set of sources is redundant, and it does not detail what aspects of the sources are redundant. For instance, when computing the redundant information in the image classification tasks, we optimized over a high-dimensional parameter space, learning a complicated nonlinear function. Although we know the exact function mapping the input sources to prediction, it is difficult to identify the "features" or aspects of the input that contributed most to the prediction. Future work could try to extend our work to not only describe how much information is redundant but what parts of the sources are redundant.

6.6 Canonical tasks

The probabilities on the right hand side of the table denote the probability $p(x_1, x_2, y)$.

X_1	X_2	Y	
a	b	ab	1/4
a	B	aB	1/4
A	b	Ab	1/4
A	B	AB	1/4

Table 6.2: UNQ. X_1 and X_2 contribute uniquely 1 bit of Y. Hence, there is no redundant and synergistic information.

X_1	X_2	Y	
0	0	0	1/4
0	1	0	1/4
1	0	0	1/4
1	1	1	1/4

Table 6.3: AND. X_1 and X_2 combine nonlinearly to produce the output Y. It is generally accepted that the redundant information is between [0,0.311] bits [53], where $I(X_1; Y) = I(X_2; Y) = 0.311$ bits.

X_1	X_2	Y	
r0	r0	r0	1/8
r0	r1	r1	1/8
r1	r0	r1	1/8
r1	r1	r0	1/8
R0	R0	R0	1/8
R0	R1	R1	1/8
R1	R0	R1	1/8
R1	R1	R0	1/8

Table 6.4: RDNXOR. A combination of redundant a synergistic information where X_1 and X_2 contributes 1 bit of redundant information, and 1 bit of synergistic information.

X_1	X_2	Y	
0	0	0	0.499
0	1	0	0.001
1	1	1	0.500

Table 6.5: IMPERFECTRDN. X_1 fully specifies the output, with X_2 having a small chance of flipping one of the bits. There should be 0.99 bits of redundant information.

6.7 Proofs

Proposition 1: Let $\mathcal{V} = \{f : \mathcal{X} \to \mathcal{P}(\mathcal{Y})\}$ consist of the family of **deterministic** functions from X to distributions over $\mathcal{Y}$. Then $I_\cap^{\mathcal{V}} = I_\cap^{\wedge}$.

Proof. We show that $I_\cap^{\mathcal{V}} = I_\cap^{\wedge}$ by proving both inequalities $I_\cap^{\mathcal{V}} \geq I_\cap^{\wedge}$ and $I_\cap^{\mathcal{V}} \leq I_\cap^{\wedge}$.

To show that $I_\cap^{\mathcal{V}} \geq I_\cap^{\wedge}$. Let $f_i : \mathcal{X} \to \mathcal{Q}$ be the functions that minimize eq. (2), and let $Q = f_i(X_i)$. Let $p(y|q)$ be the corresponding optimal decoder. Define $\hat{f}_i : \mathcal{X} \to \mathcal{P}(\mathcal{Y})$ as $\hat{f}_i(x) = p(y|f_i(x))$. Note that

$$\begin{aligned}
H_{\hat{f}_i}(Y|X_i) &= -\int p(y,x) \log p(y|f_i(x)) dx\, dy \\
&= -\int p(y,x) \Big(\int \delta_{q,f_i(x)} \log p(y|q) dq \Big) dx\, dy \\
&= -\int p(y,x) \Big(\int p(q|x,y) \log p(y|q) dq \Big) dx\, dy \\
&= -\int p(q,x,y) \log p(y|q) dq\, dx\, dy \\
&= -\int p(q,y) \log p(y|q) dq\, dy \\
&= H(Y|Q)
\end{aligned}$$

where between the first and second line we used the definition of dirac delta, between the second and third used the definition of $p(q|x) = \delta_{q,f_i(x)}$, and between the fourth and fifth line we marginalized over x. Using this result in eq. (6) and eq. (8), we obtain:

$$I_\cap^{\mathcal{V}} \geq H(Y) - H(Y|Q) = I(Y;Q) = I_\cap^{\wedge}.$$

The above inequality is obtained because $\hat{f}_i \in \mathcal{V} = \mathcal{U}$ but is not necessarily the function corresponding to the infimum.

To show that $I_\cap^{\mathcal{V}} \leq I_\cap^{\wedge}$, let $f_i : \mathcal{X} \to \mathcal{Q}$ and let $Q = f_i(X_i)$. Define $\hat{f}_i : \mathcal{X} \to \mathcal{P}(\mathcal{Y})$ as $\hat{f}_i(x) = \hat{p}(y|f_i(x))$ where $\hat{f}_i$ satisfies eq. (6) and (7). Note that

$$\begin{aligned} I_\cap^{\mathcal{V}} &= H(Y) - H_{\hat{f}_i}(Y|X) \\ &= H(Y) - H(Y|Q) \\ &= I(Y;Q) \\ &\leq I_\cap^{\wedge}. \end{aligned}$$

The second equality comes since we showed above $H(Y|Q) = H_{\hat{f}_i}(Y|X)$. The inequality comes since Q satisfies the constraint of Eq. 2 but does not necessarily maximize the objective.

□

Proposition 2: Let $t : \mathcal{X} \to \mathcal{X}$ be any invertible transformation in $\mathcal{V}$. Then:

$$I_\cap^{\mathcal{V}}(X_1; X_2 \to Y) = I_\cap^{\mathcal{V}}(t_1(X_1); t_2(X_2) \to Y) \tag{6.11}$$

Proof. We define an invertible transformation in $\mathcal{V}$ to be one such that $f \circ t \in \mathcal{V}$ for all $f \in \mathcal{V}$, which implies that $f \circ t^{-1} \in \mathcal{V}$. Recall that $I_\cap^{\mathcal{V}} := H(Y) - L_\cap^{\mathcal{V}}$ (Eq. 6.8), and note that $H(Y)$ is not affected by transformations on the sources. Let L^* correspond to the minimum of

$$\min_{f_1, f_2 \in \mathcal{V}} \frac{1}{2}\big[H_{f_1}(Y|X_1) + H_{f_2}(Y|X_2)\big] \quad s.t\ D(f_1, f_2) = 0. \tag{6.12}$$

And let L_t^* correspond to the minimum of

$$\min_{\tilde{f}_1, \tilde{f}_2 \in \mathcal{V}} \frac{1}{2}\big[H_{\tilde{f}_1}(Y|t_1(X_1)) + H_{\tilde{f}_2}(Y|t_2(X_2))\big] \quad s.t \; D(\tilde{f}_1, \tilde{f}_2) = 0. \tag{6.13}$$

We will show that $L^* = L_t^*$. Let

$$\tilde{f}_1 = f_1 \circ t_1^{-1} \in \mathcal{V},$$

$$\tilde{f}_2 = f_2 \circ t_2^{-1} \in \mathcal{V},$$

where $\tilde{f}_1, \tilde{f}_2, f_1, f_2 \in \mathcal{V} \subseteq \mathcal{U} = \{f : \mathcal{X} \to \mathcal{P}(\mathcal{Y})\}$. We can rewrite Eq. 6.13 by canceling out $t^{-1} \circ t$ as shown below so that:

$$\begin{aligned}
L_t^* &= \min_{\tilde{f}_1, \tilde{f}_2 \in \mathcal{V}} \frac{1}{2}\big[H_{\tilde{f}_1}(Y|t_1(X_1)) + H_{\tilde{f}_2}(Y|t_2(X_2))\big] \quad s.t \; D(\tilde{f}_1, \tilde{f}_2) = 0 \\
&= \min_{f_1^{\circ} t_1^{-1}, f_2^{\circ} t_2^{-1} \in \mathcal{V}} \frac{1}{2}\big[H_{f_1 \circ t_1^{-1}}(Y|t_1(X_1)) + H_{f_2 \circ t_2^{-1}}(Y|t_2(X_2))\big] \quad s.t \; D(f_1 \circ t_1^{-1}, f_2 \circ t_2^{-1}) = 0 \\
&= \min_{f_1, f_2 \in \mathcal{V}} \frac{1}{2}\big[H_{f_1}(Y|X_1) + H_{f_2}(Y|X_2)\big] \quad s.t \; D(f_1, f_2) = 0 \\
&= L^*.
\end{aligned}$$

□

Chapter 7

Gács-Körner Common Information Variational Autoencoder

We propose a notion of common information that allows one to quantify and separate the information that is shared between two random variables from the information that is unique to each. Our notion of common information is a variational relaxation of the Gács-Körner common information, which we recover as a special case, but is more amenable to optimization and can be approximated empirically using samples from the underlying distribution. We then provide a method to partition and quantify the common and unique information using a simple modification of a traditional variational auto-encoder. Empirically, we demonstrate that our formulation allows us to learn semantically meaningful common and unique factors of variation even on high-dimensional data such as images and videos. Moreover, on datasets where ground-truth latent factors are known, we show that we can accurately quantify the common information between the random variables.

7.1 Introduction

Data coming from different sensors often capture information related to common latent factors. For example, many animals have two eyes that capture different but highly-correlated

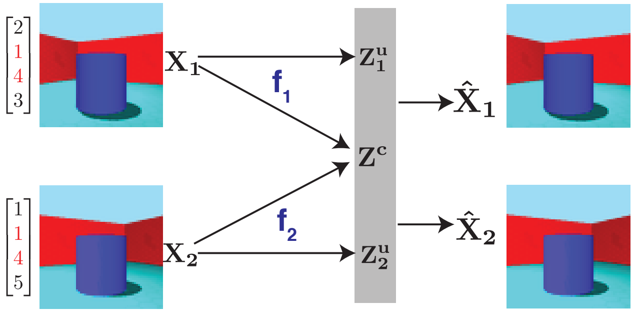

Figure 7.1: **High level schematic.** Red denotes shared latent factors (size, shape, floor, background and object color) and black denotes unique latent (viewpoint). The aim is to extract $\mathbf{z}_c$, which is a random variable that is a function of both inputs $\mathbf{x}_i$. We also allow for unique latent variables $\mathbf{z}_u$ to capture information that unique to each view. The latent representations are used to reconstruct the inputs.

views of the same objects in the scene. Similarly, sensors of different modalities, such as eyes and ears, capture correlated information about the underlying scene, as do videos and other time series, where the sensors are separated in time rather than in modality. Learning how information of one sensor maps to information of another is an appealing task, since it provides a self-supervised signal to disentangle the variability that is intrinsic in a sensor from the latent causes (e.g., objects) that are shared between multiple sensors. Indeed, there is evidence that infants spend a long time during development purposefully experiencing objects through different senses at the same time [125].

Motivated by this, we propose to learn meaningful representations of multi-view data by quantifying and exploiting such correlations in an unsupervised fashion, by using a information theoretic notion of *common information* as the guiding signal to disentangle common shared information present in high dimensional sensors (Fig. 7.1).

However, defining a notion of common information is itself not trivial. The most natural and typical way to quantify the "common part" between random variables would be by quantifying their mutual information. But mutual information has no clear interpretation in terms of a decomposition of random variables in unique and common components. In particular, [45] note that there is generally no way to write two variables X and Y using a three part code (A, B, C) such that $X = f(A, C)$, $Y = g(B, C)$ and where C encodes all and only the mutual information $I(X;Y)$. Discovering the largest common factor C, which encodes what is known as the Gács-Körner common information, from high dimensional data is then a distinct problem on its own [113, 149].

To the best of our knowledge, there are currently no approaches to compute or approximate the Gács-Körner common information from high-dimensional samples. In this work, we seek to learn common representations that satisfy the constraint that they are (approximately) a function of each input. An important contribution in this paper is that we relax the constraint that the representation needs to be a deterministic function and allow it to be a stochastic map. As we later show, this is helpful for quantifying and interpreting the latent representation, and allows us to parametrize the optimization with deep networks.

We show that our objective can be optimized using a multi-view Variational Auto-Encoder (VAE). Since in general each view can contain individual factors of variation that are not shared between the views, we augment our model with a set of unique latent variables that can capture unexplained latent factors of variation, and show that the common and unique component can be efficiently inferred from data through standard training. While training the multi-view VAE, we simultaneously develop a scalable approximation for the Gács-Körner common information, as we describe in Sect. 7.3.

To empirically evaluate the ability to separate the common and unique latent factors we introduce two new datasets, which extend commonly used datasets for evaluating disentangled representations learning: dSprites [101] and 3dShapes [20]. For each dataset, we generate a set of paired views ($\mathbf{x_1}$, $\mathbf{x_2}$) such that they share a set of common factors. We also compare our method to multi-view contrastive learning [134] and show that thanks to our definition we avoid learning degenerate representations when the views share little information. We also extend the disentanglement metric proposed by [37] to quantitatively evaluate the ability to separate the common and unique factors, as well as the ability to learn disentangled representations.

Surprisingly, we find that the addition of separate viewpoints, without any explicit supervision, enables superior disentanglement. We hypothesize that a key reason precluding the identification of latent generating factors from observed data is that receiving a single sample of a scene is quite limiting. Indeed, classical neuroscience experiments has shown that the ability to interact with an environment, as opposed to passively observing sensory inputs, is critical for learning meaningful representations of the environment [57].

7.2 Preliminaries and Related Work

We use uppercase letters to denote random variables (RVs) and lower case letters to denote their realizations. The entropy $H(X)$ of a random variable X is $\mathbb{E}_{p(x)}[\log \frac{1}{p(x)}]$. The mutual information $I(X;Z) = H(Z) - H(Z|X)$. Another useful identity for mutual information that we use is $I(X;Z) = \mathbb{E}_x[KL(p(z|x)||p(z))]$ where KL denotes the Kullback-Leibler divergence.

Gács-Körner Common Information. The Gács-Körner common information [45] is defined as

$$C_{GK}(X_1; X_2) := \max_Z H(Z) \quad \text{s.t } Z = f(X_1) = g(X_2), \tag{7.1}$$

where f and g are *deterministic* functions. The Gács-Körner common information is thus defined through a random variable Z that is a deterministic function of both inputs X_1 and X_2. Among all such random variables, Z is the random variable with maximum entropy. This has also been referred to as the "zero error information" in applications to cryptography [149]. It is an attempt to formalize and operationalize the idea of the common part between sources, which mutual information lacks. It is also a lower bound to the mutual information [45, 149]. To the best of our knowledge, there are no efficient techniques for computing the GK common information for high-dimensional X_1, X_2.

Variational Autoencoders Variational Autoencdoers (VAEs) [75] are latent variable generative models that are trained to maximize the likelihood of the data by maximizing the *evidence lower bound*:

$$\mathcal{L}_{\text{VAE}} = \mathbb{E}_{p(\mathbf{x})}[\ \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[-\log p_\theta(\mathbf{x}|\mathbf{z})] + KL(q_\phi(\mathbf{z}|\mathbf{x})\ ||\ p(\mathbf{z}))]. \tag{7.2}$$

[59] introduced the β-VAE, which modifies the traditional VAE by changing how the KL regularization is penalized (it corresponds to the traditional VAE loss when $\beta = 1$):

$$\mathcal{L}_{\beta-\text{VAE}} = \mathbb{E}_{p(\mathbf{x})}[\ \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[-\log p_\theta(\mathbf{x}|\mathbf{z})] + \beta KL(q_\phi(\mathbf{z}|\mathbf{x})\ ||\ p(\mathbf{z}))]. \tag{7.3}$$

For larger values of β the representations become more disentangled, although reconstructions become worse [59]. The modified VAE loss can also be motivated in an information theoretic manner as optimizing an information bottleneck [136], where the reconstruction term encourages a *sufficient* representation and the regularization term encourages a *minimal* representation [3, 6].

Disentangled representations A guiding assumption for representation learning is that the observed data $\mathbf{x}$ (i.e an image) can be generated from a (simpler) set of latent generating factors $\mathbf{z}$. Assuming the latent factors are independent, the idea of learning *disentangled* representations involves learning these latent factors of variation in an unsupervised manner [17]. However, despite apparent empirical progress in learning disentangled representations [21, 25,59], there remains inherent issues in both learning and defining disentangled representations [95]. In many cases, different independent latent factors may lead to equivalent observed data, and without an inductive bias, disentanglement remains ill-defined. For example, color can be decomposed into an RGB decomposition, or an equivalent HSV decomposition.

In [95, Theorem 1] it is shown that without any inductive bias, one cannot uniquely identify the underlying independent latent factors in a purely unsupervised manner from observed data. Empirically, they also found that there was no clear correlation between training statistics and disentanglement scores without supervision. Later, and related to our work, the authors examined the setting where there is paired data and no explicit supervision (weak supervision), and found that such a setup was helpful for learning disentangled representations [96]. The authors examined the setting in which the set of shared latent factors changed for each example, which was necessary for their identifiability proof. This also required using the same encoder for each view, and thus is a restricted setting that does not easily scale to multi-modal data.

Here, we study the scenario where the set of generating factors is the same across examples, as in the case of a pair of fixed sensors receiving correlated data. Additionally, our objective is motivated in an information theoretic way and our method generalizes to the case where we have different sensory modalities, which is relevant to neuroscience and multi-modal learning. Finally, our variational objective is flexible and allows estimation of the *common information* in a principled way.

Approximating Mutual Information Estimating mutual information from samples is challenging for high-dimensional random vectors [105]. The primary difficulty in estimating mutual information is constructing high-dimensional probability distribution from samples, as the number of samples required scales exponentially with dimensionality. This is impractical for realistic deep learning tasks where the representations are high dimensional. To estimate mutual information, [123] used a binning approach, discretizing the activations into a finite number of bins. While this approximation is exact in the limit of infinitesimally small bins, in practice, the size of the bin affects the estimator [49, 119]. In contrast to binning, other approaches to estimate mutual information include entropic-based estimators (e.g., [49]) and a nearest neighbours approach [88]. Although mutual information is difficult to estimate, it is an appealing quantity to summarily characterize neural network behavior because of its invariance to smooth and invertible transformations. In this work, rather than estimate the mutual information directly, we study the "usable information" in the network [78, 151], which corresponds to a variational approximation of the mutual information [15, 108].

Contrastive and Multi-View Approaches While (multi-view) contrastive learning aims to learn a representation of *only* the common information between views [26, 134, 135], we aim to learn a decomposition of the information in the views into common and unique components. Our work naturally extends to multi-sensor data that have different amounts of common/unique information (e.g., touch and vision). Moreover, contrastive approaches assume that the unique information is nuisance variability, and discard this information. Similarly, [39] also seeks to identify common information in both views, but also does not provide an objective to retain the unique information. While the multi-view literature is broad, we are not aware of previous attempts to quantify the common and unique information. Most related to our approach, [143] aim to find shared and private representations using VAEs, but it differs in how the alignment of shared information is specified and the resulting objective, and they do not provide a way to quantify the information content of the private and shared components.

7.3 Method: Gács-Körner Variational Auto-Encoder

Our formulation involves generalizing the Gács-Körner common information in eq. (7.1) to the case where f and g are stochastic functions so that the optimization problem becomes:

$$\tilde{C}_{GK}(X_1; X_2) := \max_Z I(X_i; Z) \tag{7.4}$$

$$\text{s.t. } Z = f_s(X_1) = g_s(X_2), \tag{7.5}$$

where f_s and g_s are *stochastic* functions. By the equality in eq. (7.5), we mean that $p(z|x_1) = p(z|x_2)$ for all $(x_1, x_2) \sim p(x_1, x_2)$. Note that when f and g are deterministic functions (which is a subset of stochastic functions), then $H(Z|X_i) = 0$ and we recover the original definition since

$$I(X_i; Z) = H(Z) - H(Z|X_i) = H(Z). \tag{7.6}$$

Our latter generalization (eq. 7.4-7.5) is more amenable to optimization and interpretable, as we will later demonstrate. In eq. (7.4), we used X_i as a placeholder since when $p(z|x_1) = p(z|x_2)$ for all $(x_1, x_2) \sim p(x_1, x_2, z)$ then $I(Z; X_1) = I(Z; X_2)$ since

$$\begin{aligned} I(Z; X_1) &= \mathbb{E}_{x_1}[KL(p(z|x_1)||p(z))] \\ &= \mathbb{E}_{(x_1,x_2)\sim p(x_1,x_2)}[KL(p(z|x_1)||p(z))] \\ &= \mathbb{E}_{(x_1,x_2)\sim p(x_1,x_2)}[KL(p(z|x_2)||p(z))] \\ &= \mathbb{E}_{x_2}[KL(p(z|x_2)||p(z))] \\ &= I(Z; X_2). \end{aligned}$$

This means that another equivalent formulation to maximize is $\max_z \frac{1}{2} \sum_i I(X_i; Z) = \max_z I(X_i; Z)$, for any i. To optimize the objective in eq. 7.4-7.5, we need to learn a set of latent factors Z that maximize $I(X_i; Z)$, while satisying the constraint in eq. 7.5. We propose an optimization reminiscent of the VAE objective. Define $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2)$ as the concatenation of both views, and $\mathbf{z} = (\mathbf{z}_{u_1}, \mathbf{z}_c, \mathbf{z}_{u_2})$ as a decomposition of the representation into common and unique components, and $\mathbf{z}_i = (\mathbf{z}_{u_i}, \mathbf{z}_c)$. In particular, we seek to learn latent encodings through an encoder $q_\phi(\mathbf{z}|\mathbf{x})$, which maps $\mathbf{x}$ to $\mathbf{z}$. To optimize the objective, the representation $\mathbf{z}$ should maximize $I(X_i; Z)$, and so we should also learn a decoder $p_\theta(\mathbf{x}|\mathbf{z})$ that minimizes $H(X_i|Z)$. This corresponds to the reconstruction term in a traditional VAE, though note here we reconstruct both views.

$$\mathcal{L}^1_{\text{CVAE}} = \mathbb{E}_{p(\mathbf{x})}[\ \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[-\log p_\theta(\mathbf{x}|\mathbf{z})]] \tag{7.7}$$

Without any constraints, this could be achieved trivially by using an identity mapping. To ensure that the latents encode only common information between the different views, we decompose the encodings to ensure the following constraint corresponding to eq. (7.5):

$$D(q_{\phi_{c_1}}, q_{\phi_{c_2}}) = KL(q_{\phi_{c_1}}(\mathbf{z}_c|\mathbf{x}_1) \ || \ q_{\phi_{c_2}}(\mathbf{z}_c|\mathbf{x}_2)) \ = 0. \tag{7.8}$$

Here $q_{\phi_{c_i}}(\mathbf{z}_c|x_i)$ maps $\mathbf{x}_i$ to $\mathbf{z}_{c_i}$ Rather than enforcing a hard constraint, in practice it is easier to optimize the corresponding Lagrangian relaxation:

$$\mathcal{L}^2_{\text{CVAE}} = \mathbb{E}_{p(\mathbf{x})}[\ \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[-\log p_\theta(\mathbf{x}|\mathbf{z})] + \lambda_c D(q_{\phi_{c_1}}, q_{\phi_{c_2}})]. \tag{7.9}$$

After optimizing this objective, for a sufficiently large λ so that $D(q_1, q_2) \approx 0$, the common information would be:

$$C_{GK}(X_1; X_2) = \mathbb{E}_{p(\mathbf{x})}[\ KL(q_{\phi_{c_i}}(\mathbf{z}_c|\mathbf{x}_i) \ || \ q^*(\mathbf{z}))\ ], \tag{7.10}$$

where $q^*(\mathbf{z})$ is the marginal distribution induced by the encoder. However, estimating the true marginal $q^*(\mathbf{z})$ is difficult for high-dimensional problems. In practice, we follow [75] and learn an approximate prior $p(z) \approx q^*(\mathbf{z})$, where both $q_\phi(\mathbf{z}|\mathbf{x})$ and $p(\mathbf{z})$ are taken from a given family of distributions (such as multivariate Gaussians with diagonal covariance matrix). This will additionally enable us to sample from the distribution, and interpret the latent factors. To learn $p(\mathbf{z})$ we also add the following regularization to our training objective:

$$\mathbb{E}_{p(\mathbf{x})}[\ KL(q_\phi(\mathbf{z}|\mathbf{x}) \ || \ p(\mathbf{z}))\ ]. \tag{7.11}$$

Alternatively, we can also exploit the degree of freedom in learning $q_\phi(\mathbf{z}|\mathbf{x})$ and fix $p(\mathbf{z})$ to be $\mathcal{N}(0, I)$. In both cases, our overall objective becomes:

$$\mathcal{L}_{\text{CVAE}} = \mathbb{E}_{p(\mathbf{x})}[\ \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[-\log p_\theta(\mathbf{x}|\mathbf{z})] + \lambda_c D(q_{\phi_{c_1}}, q_{\phi_{c_2}}) + \beta KL(q_\phi(\mathbf{z}|\mathbf{x}) \ || \ p(\mathbf{z}))]. \tag{7.12}$$

Optimizing this objective alone could lead to unexplained components of information, for example the unique components. Alternatively, unique information present in the individual views may be encoded in the "common" latent variable if the reconstruction benefits outweighed the cost of the divergence between the posteriors of the encoders (the term corresponding to the β).

In addition to these common latent components, we can learn unique latent components by optimizing a traditional VAE objective (i.e. with $\lambda_c = 0$) for a subset of the latent variables. Importantly we also need to ensure that the KL penalty for the unique component subset is greater than for the common subset (so that it is beneficial to encode common information in the common latent components). Our final objective becomes

$$\begin{aligned}\mathcal{L}_{\text{CVAE}} = \mathbb{E}_{p(\mathbf{x})}[\, \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[-\log p_\theta(\mathbf{x}|\mathbf{z})] + \lambda_c D(q_{\phi_{c_1}}, q_{\phi_{c_2}}) \\ + \beta_c KL(q_{\phi_c}(\mathbf{z_c}|\mathbf{x}) \mid\mid p(\mathbf{z}_c)) + \beta_u KL(q_{\phi_u}(\mathbf{z_u}|\mathbf{x}) \mid\mid p(\mathbf{z}_u))],\end{aligned} \quad (7.13)$$

where β_c and β_u correspond to a multiplier enforcing the cost of encoding common and unique information respectively. Importantly $\beta_u > \beta_c > 0$, resulting in a larger penalty on the unique latent variables (otherwise all the information would be encoded in the "unique" components). $p(\mathbf{z}_u)$ and $p(\mathbf{z}_c)$ are both sampled from $\mathcal{N}(0, I)$ of appropriate dimensionality.

We now show that, if the network architecture used for the VAE implements a generic enough class of encoder/decoders our method will recover the GK common information.

Theorem 1 (GK VAE recovers the common information)**.** *Suppose our observations* $(\mathbf{x_1}, \mathbf{x_2})$ *have GK common information defined through the random variable* $\mathbf{z}_c$ *satisfying eq. 7.4-7.5 and that our parametric function class* $q(\mathbf{z}|\mathbf{x})$ *optimized over can express any function. Then, our optimization (with* $\beta_c = 0$ *and* $\beta_u < 1$*) will recover latents* $\hat{\mathbf{z}} = (\hat{\mathbf{z}}_u^1, \hat{\mathbf{z}}_u^2, \hat{\mathbf{z}}_c)$ *where* $\hat{\mathbf{z}}_c$ *is the common random variable that maximizes the "stochastic" GK common information in eq. 7.4-7.5, while* $\hat{\mathbf{z}}_u^i$ *is the unique information of the i-th view, which maximizes* $I(\mathbf{x}_i; \mathbf{z}_u^i, \hat{\mathbf{z}}_c)$*.*

We provide the proof in Appendix F.1. Note that while the previous theorem guarantees that we will be able to separate the common and unique factors at the block level, we might not be able to disentangle the individual common factors.

7.3.1 Quantifying the common information

Suppose $D(q_{\phi_1}, q_{\phi_2}) = 0$. The term corresponding to the rate R_c of the VAE

$$R_c = I_q(Z_c; X) = \mathbb{E}_{\mathbf{x}} KL(q_{\phi_c}(\mathbf{z_c}|\mathbf{x}) \mid\mid p(\mathbf{z})) \tag{7.14}$$

is neither an upper nor lower bound on the true common information. It represents an upper bound to the information encoded in the representation specified by $q_{\phi_c}(\mathbf{z_c}|\mathbf{x})$, but does not bound the true common information in the data, since $q_\phi(\mathbf{z}|\mathbf{x})$ itself is a variational approximation.

To find a lower bound on the common information encoded in the dataset, we can use any mutual information estimator $\hat{I}$ that is a lower bound (see [108] for several). The approximate common information can then be quantified by $\hat{I}(Z_q, X)$, where $Z_q \sim q_\phi(\mathbf{z}|\mathbf{x})$. We report both the rate R_c and $\hat{I}$ in the paper. We emphasize that $\hat{I}$ can be any mutual information estimator. When the data generating distribution is known, as in our synthetic examples, we employ the "Usable Information" estimator, described in Sect. 7.4.2, which is a variational approximation [15].

7.3.2 Identifiability of the common and unique components

We now show that our optimization will in general identify the common and unique latent components. This is an important question for ICA [63], as well as for disentanglement [95,96].

Usually we do not directly observe the latent factors z, but rather an observation generated from them. We may then ask whether the common latent factors can still be reconstructed from this observation. The following proposition show that this is indeed the case, as long as the function generating f the observation is invertible, i.e., we can recover the latent factors from the observation itself.

Proposition 3. *([149], Ex. 1): Define*

$$\mathbf{z}_1 = (\mathbf{z}_c, \mathbf{z}_u^1), \quad \mathbf{z}_2 = (\mathbf{z}_c, \mathbf{z}_u^2)$$

where $\mathbf{z}_c, \mathbf{z}_u^1$, and $\mathbf{z}_u^2$ are mutually independent. Then for any invertible transformation t_i the random variable $\mathbf{z}_c$ encodes all the common information:

$$\mathbf{z}_c = \arg\max_{\hat{\mathbf{z}}} C_{GK}(t_1(\mathbf{z}_1), t_2(\mathbf{z}_2))$$

We provide the proof in Appendix F.1. The above proposition shows that when a set of factors is shared between views and when the unique factors are sampled independently, then the GK common random variable corresponds to shared latent factors. In particular, if the observations $\mathbf{x}_i$ are generated through an invertible function $\mathbf{x}_i = f(\mathbf{z}_c, \mathbf{z}_u^i)$ where $\mathbf{z}_c \sim p(\mathbf{z}_c)$ corresponds to the shared factors, the proposition shows that such factors can be recovered from the observations by maximizing the GK common information. In our GK VAE optimization, we optimize the "stochastic" GK common information and we also find in our experiments that we can (approximately) recover the latent factors from observations $\mathbf{x}_i$ generated from this process.

7.4 Experiments

We train our GK-VAE models with Adam using a learning rate of 0.001, unless otherwise stated. When the number of ground truth latent factors is known, we set the size of the latent vector of the VAE equal to the number of ground truth factors. To improve optimization, we use the idea of free bits [76] and we set $\lambda_{\text{free-bits}} = 0.1$. This was easier than using β scheduling [21], since it only involved tuning one parameter. We set β_u to be 10, β_c to be 0.1 and $\lambda_c = 0.1$. We trained networks for 70 epochs, except for the MNIST experiments, where we trained for 50 (details in the Appendix).

To ensure that the latents are shared to both encoders, during training we randomly

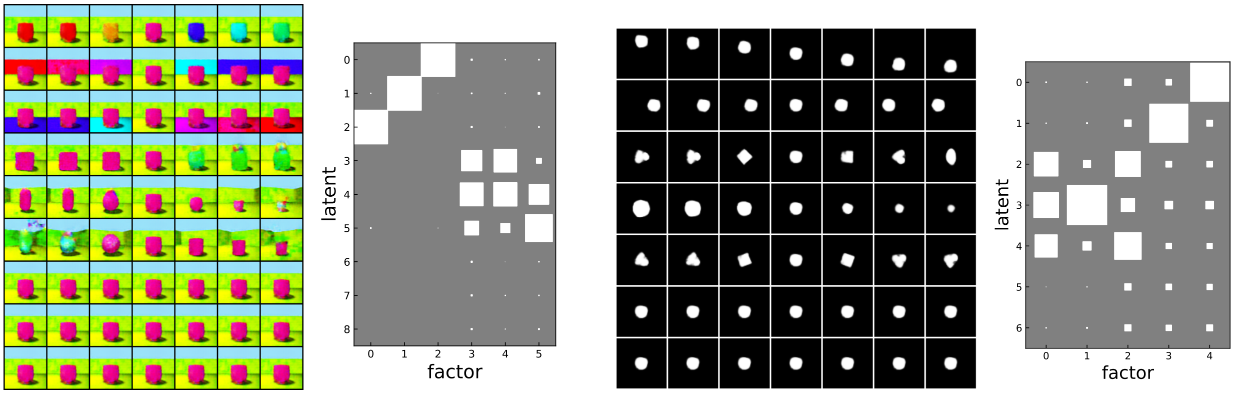

Figure 7.2: **Latent traversals and DCI plots show optimization results in separation of common and unique information. (Left) 3dshapes:** The top 3 rows shows the unique factors, the middle 3 the common (and the bottom 3 are the unique factors for the second view). Ground truth generative model: factors 0,1,2 are unique; latent unique variables are specified a priori to be latents: 0,1,2. **(Right) dsprites:** Top 2 rows; Unique: Middle 3; Common. Ground truth generative model: factors 3,4 are unique. Unique latent variables are specified a priori to be latents: 0,1.

sample $\mathbf{z}$ from either encoder $q_{\phi_i}(\mathbf{z}_c|\mathbf{x}_i)$ with $p = 0.5$. We opted to randomly sample the latents from each encoder, as opposed to performing averaging, to ensure that the latent will always be a function of an individual view $\mathbf{x}_i$. This is in addition to the soft constraint governed by λ_c in the loss.[1]

7.4.1 Evaluation Datasets

We primarily focus on the setting where the ground truth latent factors and generative model are known, in order to quantitatively benchmark our approach. To do so, we constructed datasets with ground truth latent factors so that some of the latent factors are shared between each views. That is, the generative model for the data $(\mathbf{x_1}, \mathbf{x_2})$ is

$$\mathbf{x_1} = f(\mathbf{z_u^1}, \mathbf{z_c}), \quad \mathbf{x_2} = f(\mathbf{z_u^2}, \mathbf{z_c}), \tag{7.15}$$

where $\mathbf{z_c}$ is shared between the views and $\mathbf{z_u^i}$ is the unique information encoded in the i^{th} view and f corresponds to a rendering function.

[1] Our experiments can be reproduced in approximately 3 days on a single GPU (g4dn instance).

To construct such datasets, we modified the *3dshapes* [20] and the *dsprites* dataset [101]. We select a subset of the latent factors to be shared between the views, while the remaining factors are sampled independently for each view. The *3dshapes* dataset [20] contains six independent generating factors: floor color, background color, shape color, size, shape, and viewpoint. Each latent factor can only take one of a *discrete* number of values. The *dsprites* dataset [101] contains six independent generating factors: color, shape, scale, rotation, x and y position. Each latent factor can only take one of a *discrete* number of values. When we generate multi-view data following the generative model in eq. (7.15), we refer to these datasets as *Common-3dshapes* and *Common-dsprites* respectively.

We also examine the *Rotated Mnist Dataset.* where the two views are two random digits of the same class to which a random rotation is applied. In particular, the class of the digit is common information between the views whereas the rotation is unique. We also examine the synthetic video dataset *Sprites* (not to be confused with *dsprites*) described in [93] and evaluate the common information in frames separated t frames apart. When possible, we report the average results across 3 random initializations (additional runs are included in the Appendix).

7.4.2 Metrics

DCI Disentanglement [37]. Let d be the dimension of the latent space and let $\mathbf{t}$ be the true generating factors. The idea is to train a regressor $f_j(\mathbf{z}) : \mathbb{R}^d \to \mathbb{R}$ to predict the ground truth factors $\mathbf{t}_j$ for each j. This results in a matrix of coefficients that describe the importance of each latent for predicting each ground truth factors. This can then be visualized as a matrix where the size of the square reflects the coefficient. We use this metric to evaluate disentangled representations. We used the random forest regressor, similar to [95] to predict a *discrete* number of latent classes.

	floor hue (10)	wall hue (10)	bg. hue (10)	scale (8)	shape (4)	angle (15)	KL Total
Common	-0.01	-0.02	-0.03	2.73	1.98	3.83	15.0
Unique	3.31	3.31	3.31	0.19	0.37	0.19	12.7
Total	3.31	3.31	3.31	2.69	1.98	3.82	27.7

Table 7.1: Usable Information (in bits) in representation for *3dShapes*. The common information is separated from the unique information. The ground truth factors were almost perfectly encoded in the latents. The numbers in parenthesis represents the number of discrete factors for each latent variable.

Usable Information [36, 78, 151]. We use this to approximate the mutual information when $H(X)$ is known, as it is in the datasets previously described. It is a lower bound to mutual information. We use this to lower bound the information contained in the representation Z in the next section.

7.4.3 Results

Separation of Common and Unique Latent Variables: We first examine whether our formulation can correctly separate the common and unique latent factors. After optimizing a network on our *Common-3dshapes* dataset we examined how much information about the ground-truth latent factors were encoded in the common latents $\mathbf{z}_c$ and the unique latents $\mathbf{z}_u$ (Table 1).

Given the encoded representation specified by $q_\phi(\mathbf{z}|\mathbf{x})$, we evaluated the usable information for the two latent components ($\mathbf{z}_c$ and $\mathbf{z}_u$), as well as by using the complete latent variable $\mathbf{z}$. As done in previous work [95], we directly use the mean of $q_\phi(\mathbf{z}|\mathbf{x})$ as our representation $\mathbf{z}$ rather than sampling. In Table 1, we see that the common and unique information was perfectly separated. Note, that information values reported are a lower bound to the true information, as our variational approximation is a lower bound to $I_q(Z; X)$ (which is itself a variational approximation). Our method accurately encodes all common information between views (ground truth: 3.32 bits for floor, wall, and background hue; 3 bits for scale; 2 bits for shape; 3.91 bits for orientation).

We also performed these analyses on the *Common-dsprites* dataset and found similar

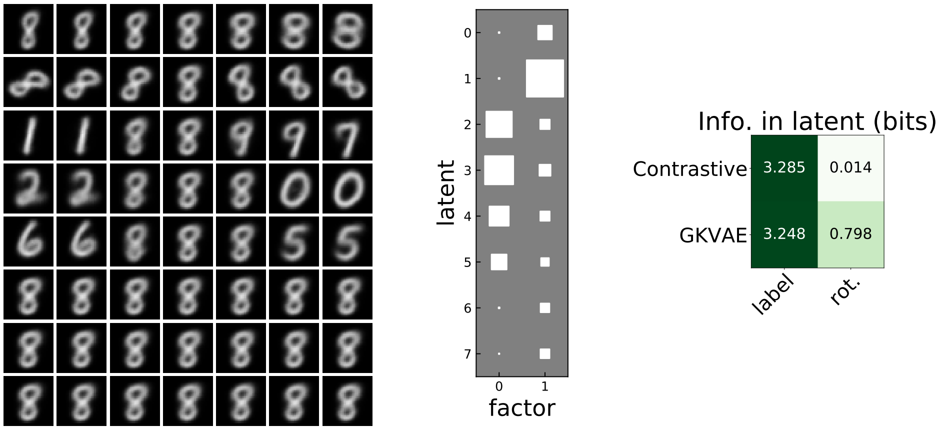

Figure 7.3: **Left.** Traversals for *Rotated Mnist*. The unique components of the latent (rows 1,2) appear to encode the “thickness" and rotation of the digit, whereas the common components appear to represent the overall digit (rows 3-6); and also the output of view 1 does not depend on the latents in rows 7,8 (these correspond to the unique components for view 2.) **Center.** Corresponding DCI matrix, where factor 0 corresponds to the label, while factor 1 corresponds to the rotation (discretized into 10 bins). **Right.** Comparison against contrastive implementation from [134], where the contrastive approach does not encode any usable information about the unique factor (the rotation).

results (Table 2, Appendix). In particular, the unique latent factors corresponding to position are encoded in the unique components of the latent representation, while the other factors are encoded in the common latent representation. We emphasize that the generative model was not used at all during training, and was only used for quantitative evaluation after training. Additional runs are in Appendix F.5.

Rotated Mnist and Comparison with Contrastive Learning: As described before, we generate a dataset of paired views of digits of the same class, each rotated by an independent random amount. In this manner, the unique information is about the rotation, whereas the common information is about the class. In Fig. 7.3 we see that the unique components of the latent (rows 1, 2) appear to encode the rotation and “thickness" of the digit, whereas the common components seem to represent the class of the digit (rows 3-6). Also, as expected, the output of view 1 does not depend on the latents in rows 7, 8 which by construction correspond to the unique components of view 2.

This setup is reminiscent of contrastive learning, where the goal is to learn a representation which is invariant to a random data augmentation of the input (such as a random rotation). By construction, contrastive learning aims to encode the common information before and after data augmentation, but may not encode any other information. This can lead to degraded performance on downstream tasks, as the discarded unique information may still be important for the task [135, 140]. On the other hand, our GK-VAE separates the unique and common information without discarding information.

To highlight this difference between approaches, we trained using a contrastive objective[2] [134], and found that indeed while we can decode the shared class label, we cannot decode the unique rotation angle of view 1 (discretized into 10 bins; Fig. 7.3, right). On the other hand, using our method we recover the common and unique information.

Video Experiment: The existence of common information though time is another important learning signal. To study it, we perform an experiment on the *Sprites* dataset described in [93]. This dataset consists of synthetic sequences all with 8 frames. We optimized using the same architecture and hyperparameters except we set $\lambda_c = 0.5$. We examine the common information between frames t frames apart, approximated using the KL divergence term. In particular, the two views are two frames $(\mathbf{X}_1, \mathbf{X}_t)$, where each pair belongs to a different video sequences. In Fig. 7.4 we see that in general, as t increases the common information between the frames decreases evidencing the fact that, due to the random temporal evolution of the video, common information is lost as time progresses. We also note that the common information appears to increase in the last frame; this could be that in many of the sequences the sprite returns close to the initial state (see Fig. 3 in [93]).

Emergence of Disentangled Representations: While not the main focus of our work, we qualitatively observed that our optimization led not only to separation of unique and common information, but also in the emergence of interpretable/disentangled latent factors. We quantify disentanglement using DCI score [37], which we show in Fig. 7.2. For the *dSprites*, the disentanglement score was 0.54 (std 0.015) (averaged over 3 initializations),

[2] We used the code from: https://github.com/HobbitLong/CMC (BSD 2-Clause License)

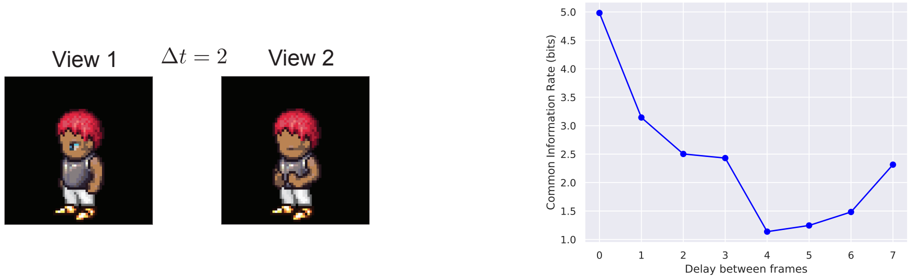

Figure 7.4: Sprites [93] video experiment. **Left.** Example views separated 2 frames apart. **Right** Common information as a function of delay between frames. In general common information is decreasing as the delay gets longer.

and for the *3dshapes* the DCI score was 0.83 (std: 0.04 (averaged over 3 initializations). In contrast a standard β-VAE with the same hyper-parameters ($\beta = \beta_u = 10$) and $z = 8$ obtained a DCI score of 0.44 and 0.76 over 3 random initializations for *dsprites* and *3dshapes*, respectively. Additionally, these plots visually reaffirm that the common and unique factors are identified at the block-level. We also include traversals of the prior shown in Fig. 7.2 to show qualitatively that the learned factors of variation are meaningful.

Our Common VAE therefore disentangled latents better than a β-VAE in these tasks. [3] argue that deep networks have an implicit bias toward recovering disentangled factor of variations. While this is not enough on its own to uniquely identify the ground-truth latent variables (Sect. 7.2), our results suggest that having the additional "weak supervision" coming from the sepatation of common and unique information was also helpful for recovering disentangled and interpretable factors, even when not explicitly optimizing for it.

7.5 Discussion

We show formally and empirically that we can partition the latent representation of multi-view data into a common and unique component, and also provide a tractable approximation for the Gács-Körner common information between high dimensional random variables. In many practical scenarios where high dimensional data comes from multiple sensors, such as neuroscience and robotics, it is desirable to understand and quantify what is common and what is unique between the observations. Motivated by the definition of common information proposed by Gács and Körner [45], we propose a variational relaxation and show that it can be efficiently learned from data by training a slighly modified VAE. Empirically, we demonstrate that our formulation allows us to learn semantically meaningful common and unique factors of variation. Moreover, our formulation allows us to approximate the Gács-Körner common information for realistic high-dimensional data, which has been a difficult problem [113]. Our formulation is also a generative multi-view model that allows sampling and manipulation of the common and unique factors.

As the common information was motivated by an information theoretic coding problem [45], our work naturally relates to compression schemes. Indeed, approximate forms of the common information, discussed further in Appendix F.3, are scenarios for distributed compression, since the common information needs to only be transmitted once [113, 114]. It may be interesting to combine our approach with recent advances in practical compression algorithms that leverage VAEs [137].

Chapter 8

Conclusion

In animals, the perceptual representations they form of the environment underlies their remarkable and flexible behavior. The success of Deep Neural Networks has been driven by a paradigmatic shift towards *learning* task specific representations through optimization, as opposed to using hand-engineered features. This dissertation consists of "artificial neuroscience" experiments on these artificial networks to better understand how they learn to process and represent inputs.

We find internal representations in trained deep neural networks capture the key features of multi-area neural recordings during a perceptual decision-making task, where minimal sufficient representations of sensory information emerge along a cortical hierarchy (Chapter 2 and Chapter 3). We then show that these minimal sufficient representations emerge through complex learning dynamics beginning during the early phases of training where additional information not relevant to the task is acquired, but later discarded (Chapter 4). This initial stage of training is critical for the network: sensory deficits during this initial period permanently affect performance and learned representations, in a remarkably similar fashion to *critical learning periods* observed in humans and other animals (Chapter 5).

We also study how multisensory information can be decomposed, and develop novel approximations to compute the redundant information shared between a set of sources about a target (Chapter 6), and show that the common information shared between a set of sources can be used to guide the learning of meaningful representations (Chapter 7).

8.1 Implications for the Brain

This dissertation began with using trained artificial networks to generate hypotheses for cortical computation. This approach of using trained artificial networks as cortical models needs to be done carefully to generate neuroscientific impact, as these models often lack many biological details and constraints; for example they do not model neural spiking. Further, these deep neural networks themselves are challenging to understand.

In the subsequent work in this dissertation, I made the conscious decision to focus on understanding the deep networks themselves independent of their use as cortical models, however there are intriguing connections. In particular, we found that a noisy learning process (coming from training with SGD with a small batch size and large learning) was important for learning optimal (minimal sufficient) representations. This suggests one potential benefit of seemingly noisy neural data is that learning to process information with such noise can lead to more minimal sufficient (or optimal) representations. Such representations were consistent with neural representations of monkeys during a perceptual decision-making task (Chap. 2 and Chap. 3).

Next, we observed critical learning periods for multisensory integration in artificial deep networks, which mirrored many of the phenomenon observed in animal experiments, including altered behavior (quantified by altered generalization ability) and learned representations. Critically, to replicate these observations, we only needed to consider temporary perturbations to the data distribution early during learning, and did not need to include any plasticity or biological factors often used for explaining critical learning periods. An interpretation of our results is that critical learning periods may be a general consequence of a network or agent that needs to learn from experience through many local parameter (or synaptic) updates with a non-convex loss landscape.

8.2 Potential Directions for Future Research

There are some natural directions for future research, that builds upon some of the results presented in this dissertation. In particular, it may now be possible to develop a mathematical theory explaining critical learning periods in terms of temporary changes to the data distribution. Additionally, our information theoretic approximations can be directly applied to neural data. For example, our notion of Usable Information can be helpful for formalizing the information that is accessible to biological decoders, and may be helpful for disentangling information usage from the presence of information in an area. Additionally, our approximation of the Redundant Information (Chapter 6) and common information (Chapter 7) can now be applied for real-world high dimensional inputs (such as high dimensional neural data from multiple brain areas), and may be helpful for providing insight into multi-area neural processing.

Appendices

Appendix A

Supplementary Material for Chapter 2

A.1 Task and training details

A.1.1 Somatomotor reaction time visual discrimination task and recordings from PMd:

The task, training and electrophysiological methods used to collect the data used here have been described previously [24] and are reviewed briefly below. All surgical and animal care procedures were performed in accordance with National Institutes of Health guidelines and were approved by the Stanford University Institutional Animal Care and Use Committee. Two trained monkeys (Ti and Ol) performed a visual reaction time discrimination task. The monkeys were trained to discriminate the dominant color in a central static checkerboard composed of red and green squares and report their decision with an arm movement. If the monkey correctly reached to and touched the target that matched the dominant color in the checkerboard, they were rewarded with a drop of juice. This task is a reaction time task, so that monkeys initiated their action as soon as they felt they had sufficient evidence to make a decision. On a trial-by-trial basis, we varied the signed color coherence of the checkerboard, defined as $(R - G)/(R + G)$, where R is the number of red squares and G the number of green squares. The color coherence value for each trial was chosen uniformly at random from 14 different values arranged symmetrically from 90% red to 90% green. Reach targets were located to the left and right of the checkerboard. The target configuration (left red, right green; or left green, right red) was randomly selected on each trial. Both monkeys demonstrated qualitatively similar psychometric and reaction-time behavior. 996 units were recorded from Ti (n=546) and Ol (n=450) while they performed the task [24]. Monkey Ol and Ti's PMd units both had low choice color probability. Reported analyses from PMd data use units pooled across Monkey Ol and Ti.

A.1.2 RNN description and training

We trained a continuous-time RNN to perform the checkerboard task. The RNN is composed of N artificial neurons (or units) that receive input from N_{in} time-varying inputs $\mathbf{u}(t)$ and produce N_{out} time-varying outputs $\mathbf{z}(t)$. The RNN defines a network state, denoted by $\mathbf{x}(t) \in \mathbb{R}^N$; the ith element of $\mathbf{x}(t)$ is a scalar describing the "currents" of the ith artificial neuron. The network state is transformed into the artificial neuron firing rates (or network rates) through the transformation:

$$\mathbf{r}(t) = f(\mathbf{x}(t)), \tag{A.1}$$

where $f(\cdot)$ is an activation function applied elementwise to $\mathbf{x}(t)$. The activation function is typically nonlinear, endowing the RNN with nonlinear dynamics and expressive modeling capacity [50]. In this work, we use $f(x) = \max(x, 0)$, also known as the rectified linear unit, i.e., $f(x) = \text{relu}(x)$. In the absence of noise, the continuous time RNN is described by the equation

$$\tau\dot{\mathbf{x}}(t) = -\mathbf{x}(t) + \mathbf{W}_{\text{rec}}\mathbf{r}(t) + \mathbf{W}_{\text{in}}\mathbf{u}(t) + \mathbf{b}_{\text{rec}} + \epsilon_t, \tag{A.2}$$

where τ is a time-constant of the network, $\mathbf{W}_{\text{rec}} \in \mathbb{R}^{N\times N}$ defines how the artificial neurons are recurrently connected, $\mathbf{b}_{\text{rec}} \in \mathbb{R}^N$ defines a constant bias, $\mathbf{W}_{\text{in}} \in \mathbb{R}^{N\times N_{in}}$ maps the RNN's inputs onto each artificial neuron, and ϵ_t is the recurrent noise. The output of the network is given by a linear readout of the network rates, i.e.,

$$\mathbf{z}(t) = \mathbf{W}_{\text{out}}\mathbf{r}(t), \tag{A.3}$$

where $\mathbf{W}_{\text{out}} \in \mathbb{R}^{N_{\text{out}} \times N}$ maps the network rates onto the network outputs.

We trained RNNs to perform the checkerboard task as follows. For all networks, unless we explicitly varied the amount of units, we used $N_{\text{in}} = 4$, $N = 300$, and $N_{\text{out}} = 2$.

The four inputs were defined as:

1. Whether the left target is red (-1) or green (+1).

2. Whether the right target is red (-1) or green (+1).

3. Signed coherence of red (ranging from -1 to 1), $(R - G)/(R + G)$.

4. Signed coherence of green (ranging from -1 to 1), $(G - R)/(R + G)$. Note that, prior to the addition of noise, the sum of the signed coherence of red and green is zero.

The inputs, $\mathbf{u}(t) \in \mathbb{R}^4$, were defined at each time step, t, in distinct epochs. In the 'Center Hold' epoch, which lasted for a time drawn from distribution $\mathcal{N}(200\text{ ms}, 50^2\text{ ms}^2)$, all inputs were set to zero. Subsequently, during the 'Targets' epoch, which lasted for a time drawn from distribution $\mathcal{U}[600\text{ ms}, 1000\text{ ms}]$, the colors of the left and right target were input to the network. These inputs were noiseless, as illustrated in Fig. 2.1, to reflect that target information is typically unambiguous in our experiment. Following the 'Targets' epoch, the signed red and green coherences were input into the network during the 'Decision' epoch. This epoch lasted for 1500 ms. We added zero mean independent Gaussian noise to these inputs, with standard deviation equal to 5% of the range of the input, i.e., the noise was drawn from $\mathcal{N}(0, 0.1^2)$. At every time point, we drew independent noise samples and added the noise to the signed red and green coherence inputs. We added recurrent noise ϵ_t, adding noise to each recurrent unit at every time point, from a distribution $\mathcal{N}(0, 0.05^2)$. Following the 'Decision' epoch, there was a 'Stimulus Off' epoch, where the inputs were all turned to 0.

The two outputs, $\mathbf{z}(t) \in \mathbb{R}^2$ were defined as:

1. Decision variable for a left reach.

2. Decision variable for a right reach.

We defined a desired output, $\mathbf{z}_{\text{des}}(t)$, which was 0 in the 'Center Hold' and 'Targets' epochs. During the 'Decision' epoch, $\mathbf{z}_{\text{des}}(t) = 1$. In the 'Stimulus Off' epoch, $\mathbf{z}_{\text{des}}(t) = 0$. In RNN training, we penalized output reconstruction using a mean-squared error loss,

$$\mathcal{L}_{\text{mse}} = \frac{1}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} |\mathbf{z}(t) - \mathbf{z}_{\text{des}}(t)|^2 . \tag{A.4}$$

The set $\mathcal{T}$ included all times from all epochs except for the first 200 ms of the 'Decision' epoch from the loss. We excluded this time to avoid penalizing the output for not immediately changing its value (i.e., stepping from 0 to 1) in the 'Decision' epoch. Decision variables are believed to reflect a gradual process consistent with non-instantaneous integration of evidence, e.g., as in drift-diffusion style models, rather than one that steps immediately to a given output.

To train the RNN, we minimized the loss function:

$$\mathcal{L} = \mathcal{L}_{\text{mse}} + \frac{\lambda_{\text{in}}}{NN_{\text{in}}} \|\mathbf{W}_{\text{in}}\|_F^2 + \frac{\lambda_{\text{rec}}}{N^2} \|\mathbf{W}_{\text{rec}}\|_F^2 + \frac{\lambda_{\text{out}}}{NN_{\text{out}}} \|\mathbf{W}_{\text{out}}\|_F^2 + \frac{\lambda_r}{T} \sum_t \|\mathbf{r}(t)\|^2 + \lambda_\Omega \mathcal{L}_\Omega \tag{A.5}$$

where

- $\|\mathbf{A}\|_F$ denotes the Frobenius norm of matrix $\mathbf{A}$

- $\lambda_{\text{in}} = \lambda_{\text{rec}} = \lambda_{\text{out}} = 1, \lambda_r = 0$ to penalize larger weights.

- $\lambda_\Omega = 2$

- $\mathcal{L}_\Omega$ is a regularization term that ameliorates vanishing gradients proposed and is described in prior literature [106, 127].

During the training process, we also incorporated gradient clipping to prevent exploding gradients [106]. Training was performed using stochastic gradient descent, with gradients calculated using backpropagation through time. For gradient descent, we used the Adam optimizer, which is a first order optimizer incorporating adaptive gradients and momentum [74].

Every 200 or 500 training epochs, we generated 2800 cross-validation trials, 100 for each of the 28 possible conditions (14 coherences × 2 target configurations). For each trial, there was a correct response (left or right) based on the target configuration and checkerboard coherence. When training, we defined a "correct decision" to be when the RNNs DV for the correct response was greater than the other DV and the larger DV was greater than a pre-set threshold of 0.6. We evaluated the network 500ms before the checkerboard was turned off (the end of the trial). We required this criteria to be satisfied for at least 65% of both leftward and rightward trials. We note that this only affected how we terminated training. It had no effect on the backpropagated gradients, which depended on the mean-squared-error loss function. Note that a trial that outputted the correct target but did not reach the 0.6 threshold would not be counted towards the 65% criteria.

When testing, we defined the RNNs decision to be either: (1) whichever DV output (for left or right) first crossed a pre-set threshold of 0.6, or (2) if no DV output crossed the pre-set threshold of 0.6 by the end of the 'Decision epoch,' then the decision was for whichever DV had a higher value at the end of this epoch — an approach that is well established in models of decision-making [19, 109]. If the RNN's decision on a single trial was the same as the correct response, we labeled this trial 'correct.' Otherwise, it was incorrect. The proportion of decisions determined under criterion (2) was negligible (0.5% across 100 trials for each of 28 conditions). An interpretation for criterion (2) is that if the RNN's DV has not achieved the threshold certainty level by the end of a trial, we assign the RNN's decision to be the

Hyperparameter	**Value**
Number of units	300
Number of areas	3
Learning rate	5e-5
Time Constant	50ms
Discretization bin width	10ms
Rate regularization	0
Weight regularization	1
Activation function	Relu
Feedforward connection	10%
Feedback connections	5%
Dale law	Yes

Table A.1: Hyperparameters of exemplar RNN.

direction for which its DV had the largest value. Finally, in training only, we introduced 'catch' trials 10% of the time. On 50% of catch trials, no inputs were shown to the RNN and $\mathbf{z}_{\text{des}}(t) = 0$ for all t. On the remaining 50% of catch trials, the targets were shown to the RNN, but no coherence information was shown; likewise, $\mathbf{z}_{\text{des}}(t) = 0$ for all t on these catch trials.

We trained the three-area RNNs by constraining the recurrent weight matrix $\mathbf{W}_{\text{rec}}$ to have connections between the first and second areas and the second and third areas. In a multi-area network with N neurons and m areas, each area had N/m neurons. In our 3-area networks, each area had 100 units. Of these 100 units, 80 were excitatory and 20 were inhibitory. Excitatory units were constrained to have only positive outgoing weights, while inhibitory units were constrained to have only negative outgoing weights. We used the `pycog` repository [127] to implement these architecture constraints. The parameters for the exemplar RNN used in the paper are shown in Table A.1. In our hyperparameter sweeps, we varied the hyperparameters of the exemplar RNN. For each parameter configuration, we trained 8 different networks with different random number generator seeds.

A.2 Additional description of analyses

A.2.1 Decoding analysis for PMd data

For PMd data, we calculated decoding accuracy using 400 ms bins. We report numbers in a window [-300ms, +100 ms] aligned to movement onset. We used the MATLAB *classify* command with 75% training and 25 % test sets. Decoding analyses were performed using 5-31 simultaneously recorded units from Plexon U-probes and the averages reported are across 51 sessions. To assess whether decoding accuracies were significant on a session by session basis, we shuffled the labels 200 times and estimated the 1^{st} and 99^{th} percentiles for this surrogate distribution. The decode accuracy for direction, color, and context variables for a session was judged to be significant if it lay outside this shuffled accuracy. Every session had significant direction decode, while no session had significant color and context decode accuracy.

A.2.2 Decoding and Mutual information for RNNs

We used a decoder and mutual information approximation to quantify the amount of information (color, context, direction) present in the network. We trained a neural network to predict a relevant choice (for example, color) on a test set from the activity of a population of units. We used 700 trials for training, and 2100 independent trials for testing. To generate the trials for training and testing, we increased the recurrent noise to be drawn from the distribution ($\mathcal{N}(0, 0.1^2)$) to prevent overfitting. For each trial, we averaged data in a window [-300ms, +100ms] around reaction time.

We trained a neural network with 3 layers, 64 units per layer, leakyRelu activation (α=0.2), and dropout (p=0.5), using SGD, to predict the choice given the activity of the population. We removed the leakyRelu activation for the linear network, and increased dropout (p=0.8). For both the nonlinear and linear network, we trained the neural network to minimize the cross-entropy loss. We used the same neural network from the decode to compute an approximation to mutual information, described in Supplementary Note 2.

A.2.3 RNN behavior

To evaluate the RNN's psychometric curve and reaction-time behavior, we generated 200 trials for each of the 28 conditions, producing 400 trials for each signed coherence. For these trials, we calculated the proportion of red decisions by the RNN. This corresponds to all trials where the DV output for the red target first crossed the preset threshold of 0.6; or, if no DV output crossed the threshold of 0.6, if the DV corresponding to the red target exceeded that corresponding to the green target. The reaction time was defined to be the time between checkerboard onset to the first time a DV output exceeded the preset threshold of 0.6. If the DV output never exceeded a threshold of 0.6, in the reported results, we did not calculate a RT for this trial.

A.2.4 dPCA

Demixed principal components analysis (dPCA) is a dimensionality reduction technique that provides a projection of the data onto task related dimensions while preserving overall variance [85]. dPCA achieves these aims by minimizing a loss function:

$$L_{dpca} = \sum_{c} \|\mathbf{X}_c - \mathbf{P}_c \mathbf{D}_c \mathbf{X}\|^2. \tag{A.6}$$

Here, $\mathbf{X}_c$ refers to data averaged over a "dPCA condition" (such as time, coherence, context, color, or direction), having the same shape as $\mathbf{X} \in \mathbb{R}^{N \times cT}$, but with the entries replaced with the condition-averaged response. The aim is to recover (per dPCA condition c) a $\mathbf{P}_c$ and $\mathbf{D}_c$ matrix. $\mathbf{P}_c$ is constrained to have orthonormal columns, while $\mathbf{D}_c$ is unconstrained. The number of columns of $\mathbf{P}_c$ and rows of $\mathbf{D}_c$ reflects the number of components one seeks to find per condition. We project the data onto the principal components $\mathbf{D}_c\mathbf{X}$ to observe the demixed components (Fig. 2.4b). The column of $\mathbf{P}_c$ reflects how much the demixed data contributes to each neuron. We use the principal axes from $\mathbf{P}_c$ to compute the axis overlap, as in Kobak et al [85]. We used axes of dimension 1 for RNNs, which were sufficient to capture most color, context, or direction variance. For the neural data, we used five components for direction, color and context since the PMd data was higher dimensional than the RNNs.

Our results were consistent if we used dPCA or TDR (Fig. A.8). The top principal axis from each $\mathbf{P}_c$ are analogous to the axes found from TDR. Both methods seek to reconstruct neural activity from demixed components. To apply TDR, one explicitly parametrizes task variables (See Targeted Dimensionality Reduction (Appendix A.2.5)), while $\mathbf{D}_c\mathbf{X}$ serves the purpose of finding demixed components in dPCA. Overall, the choice of using dPCA or TDR to find the axes did not affect our conclusions.

For multi-area analyses, we separated the units for each area and found the task-relevant axes for this subset of units. For the inter-area analyses, we used RNNs with only excitatory connections, and therefore found the color and direction axis using only the excitatory units (Fig. A.9). In all other analyses, all units were used to identify the axes. For RNN activity, we performed dPCA using activity over the entire trial. For PMd activity, we used a window of (0ms, 800ms) relative to checkerboard onset. We restricted time windows for the PMd activity because we wanted to minimize movement related variance.

A.2.5 Targeted Dimensionality Reduction

Targeted dimensionality reduction (TDR) is a dimensionality reduction technique that finds low dimensional projections that have meaningful task interpretations. We applied TDR as described by the study by [99]. We first z-scored the firing rates of each of the 300 units across time and trials, so that the firing rates had zero mean and unit standard deviation. We then expressed this z-scored firing rate as a function of task parameters using linear regression,

$$r_{i,t}(k) = \beta^1_{i,t}\text{color}(k) + \beta^2_{i,t}\text{direction}(k) + \beta^3_{i,t}\text{context}(k). \tag{A.7}$$

Here, $r_{i,t}(k)$ refers to the firing rate of unit i at time t on trial k. The total number of trials is N_{trials}. This regression identifies coefficients $\beta^m_{i,t}$ that multiply the m$^{\text{th}}$ task parameter to explain $r_{i,t}(k)$. We defined the task parameters as follows:

- color(k) was the signed coherence of the checkerboard on trial k, given by $(R-G)/(R+G)$.
- direction(k) was -1 for a left decision and $+1$ for a right decision.
- context(k) was the target orientation, taking on -1 if the green (red) target was on the left (right) and $+1$ if the green (red) target was on the right (left).

We did not fit a bias term since the rates were z-scored and therefore zero mean. For each unit, i, we formed a matrix $\mathbf{F}_i$ having dimensions $N_{\text{trials}} \times 3$, where each row consisted of [color(k), direction(k), context(k)]. We define $\mathbf{r}_{i,t}$ to be the rate of unit i at time t across all trials. We then solved for the coefficients, denoted by $\boldsymbol{\beta}_{i,t} = [\beta^1_{i,t},\ \beta^2_{i,t},\ \beta^3_{i,t}]^T$, using least squares,

$$\boldsymbol{\beta}_{i,t} = (\mathbf{F}_i^T \mathbf{F}_i)^{-1} \mathbf{F}_i^T \mathbf{r}_{i,t}. \tag{A.8}$$

Each $\boldsymbol{\beta}_{i,t}$ is therefore a 3×1 vector, and concatenating $\boldsymbol{\beta}_{i,t}$ across t results in $\boldsymbol{\beta}_i$, a $3 \times T$ matrix, of which there are N. We then formed a tensor where each β_i is stacked, leading to a tensor with dimensions $3 \times T \times N$. For each of the 3 task variables, we found the time T where the norm of the regression coefficients, across all units, was largest. For the mth task variable, we denote the vector $\beta^m_{\text{max}} \in \mathbb{R}^N$ to be a vector of coefficients that define a 1-dimensional projection of the neural population activity related to the mth task variable. These vectors are what we refer to as the task related axes. To orthogonalize these vectors, we performed QR decomposition on the stacked $\boldsymbol{\beta}_{\text{max}}$ matrix $[\boldsymbol{\beta}^1_{\text{max}}, \boldsymbol{\beta}^2_{\text{max}}, \boldsymbol{\beta}^3_{\text{max}}]$, which is an $N \times 3$ matrix. This decomposition finds orthogonal axes so that the axes would capture independent variance.

A.2.6 Choice probability

To calculate the choice probability for a single unit, we first computed the average firing rate in a window from [−300 ms, +100 ms] around the reaction time for each trial. We used the average firing rates calculated across many trials to create a firing rate distribution based on either the color decision (trials corresponding to a red or green choice) or the direction decision (trials corresponding to a left or right choice).

To compute the color choice probability, we constructed the firing rate distributions corresponding to a green choice or red choice. If these two distributions are non-overlapping, then the neuron has a color choice probability of 1; the average firing rate will either overlap with the red or green firing rate distributions, but not both. On the other hand, if the two distributions are completely overlapping, then the neuron has a color choice probability of 0.5; knowing the firing rate of the neuron provides no information on whether it arose from the red or green firing rate distribution. When there is partial overlap between these two distributions, then firing rates where the distributions overlap are ambiguous. We computed choice probability as the area under the probability density function at locations when the two distributions did not overlap, divided by 2 (to normalize the probability). To calculate the direction choice probability, we repeated the same calculation using firing rate distributions corresponding to a left choice or right choice.

A.2.7 Canonical correlation

We applied CCA to assess the similarity between neural activity and the artificial unit activity [132]. Before applying CCA, we performed principal component analysis to reduce the dimensionality of the artificial and neural activity to remove noise [132]. We reduced the dimensionality to 3 and 8 for RNNs and PMd, respectively. These dimensionalities were chosen as they captured over 88% of the variance for each dataset when aligned to checkerboard. We report the average CCA correlation coefficients in Fig. 2.2 using times in a window of [0, 400ms] aligned to checkerboard onset for the PMd and RNN activity. The data was binned in 10ms bins.

A.2.8 Analyses of inputs and activity

In order to disentangle the effects of external inputs and recurrence, in Fig. 2.4a, we evaluated the input contribution and overall activity. For Area 1, we defined the input contribution as $\mathbf{W}_{\text{in}}\mathbf{u}_t$, and for areas 2 and 3, we defined the input contribution as $\mathbf{W}_{21}\mathbf{r}_t^1$, and $\mathbf{W}_{32}\mathbf{r}_t^2$ respectively, where $\mathbf{r}_t^m$ denotes the activity of the units in area m. The activity $\mathbf{r}_t^m$ corresponds to the firing rate that experimentalists could measure, reflecting a combination of input and recurrent interactions. For constant inputs, a stable value of the activity implies there is little recurrent processing.

A.2.9 Inter-Area Projection Analyses

To calculate the overlap between the color and direction axes with the potent and null spaces, we performed singular value decomposition on the inter-area connections, $\mathbf{W}_{21}$ and $\mathbf{W}_{32}$. $\mathbf{W}_{21}$ and $\mathbf{W}_{32}$ were 80×80 matrices, and were full rank. Nevertheless, they had near some zero singular values, indicating that the effective rank of the matrix was less than 80. We defined the potent dimensions to be the top m right singular vectors, while the null dimensions were the remaining $80 - m$ right singular vectors.

We performed the analyses of Fig. 2.5a,b by varying the potent and null dimensions, sweeping m from 1 to 80. For each defined potent and null space, we calculated the axis overlap between the direction (or color) axis and the potent (or null) space by computing the L2-norm of the orthogonal projection (squared). We report the squared quantity because the expectation of the norm of a projection of a random vector onto an m-dimensional subspace of an n-dimensional space is m/n. We include an approximation of the expectation of the projection of a random vector in Fig. 2.5a,b by averaging the projection of 100 random vectors. Our results show that the direction axis was always more aligned with potent dimensions than the color axis, irrespective of the choice of m, and that the direction axis was preferentially aligned with the top singular vector.

A.2.10 Visualization of neural activity in a low dimensional space

The activity of multiple units on a single trial is high dimensional, with dimension equal to the number of units. To visualize the activity in a lower dimensional space, dimensionality reduction techniques can be used. In addition to TDR, we also utilized Principal Components Analysis (PCA) and t-distributed Stochastic Neighbor Embedding (tSNE) to visualize neural activity in low-dimensional spaces.

PCA finds a linear low-dimensional projection of the high dimensional data that maximizes captured variance. We performed PCA on both the experimental data and RNN rates. PCA is an eigenvalue decomposition on the data covariance matrix. To calculate the covariance matrix of the data, we averaged responses across conditions. This reduces single trial variance and emphasizes variance across conditions. Firing rates were conditioned on reach direction and signed coherence. In both the experimental data and RNN rates, we had 28 conditions (14 signed coherences each for left and right reaches).

tSNE embeds high dimensional data in a low dimensional manifold that is nonlinear, enabling visualization of activity on a nonlinear manifold. The tSNE embedding maintains relative distances between data points when reducing dimensionality, meaning that points closer in high dimensional space remain closer when viewed in a low dimensional manifold. We projected our data into a two dimensional manifold. We used the default parameters from the sickit-learn implementation (https://scikit-learn.org/stable/modules/generated/sklearn.manifold.TSNE.html). The data visualized under tSNE was averaged in a window [-300ms, +100ms] around reaction time.

A.3 Supplementary Figures

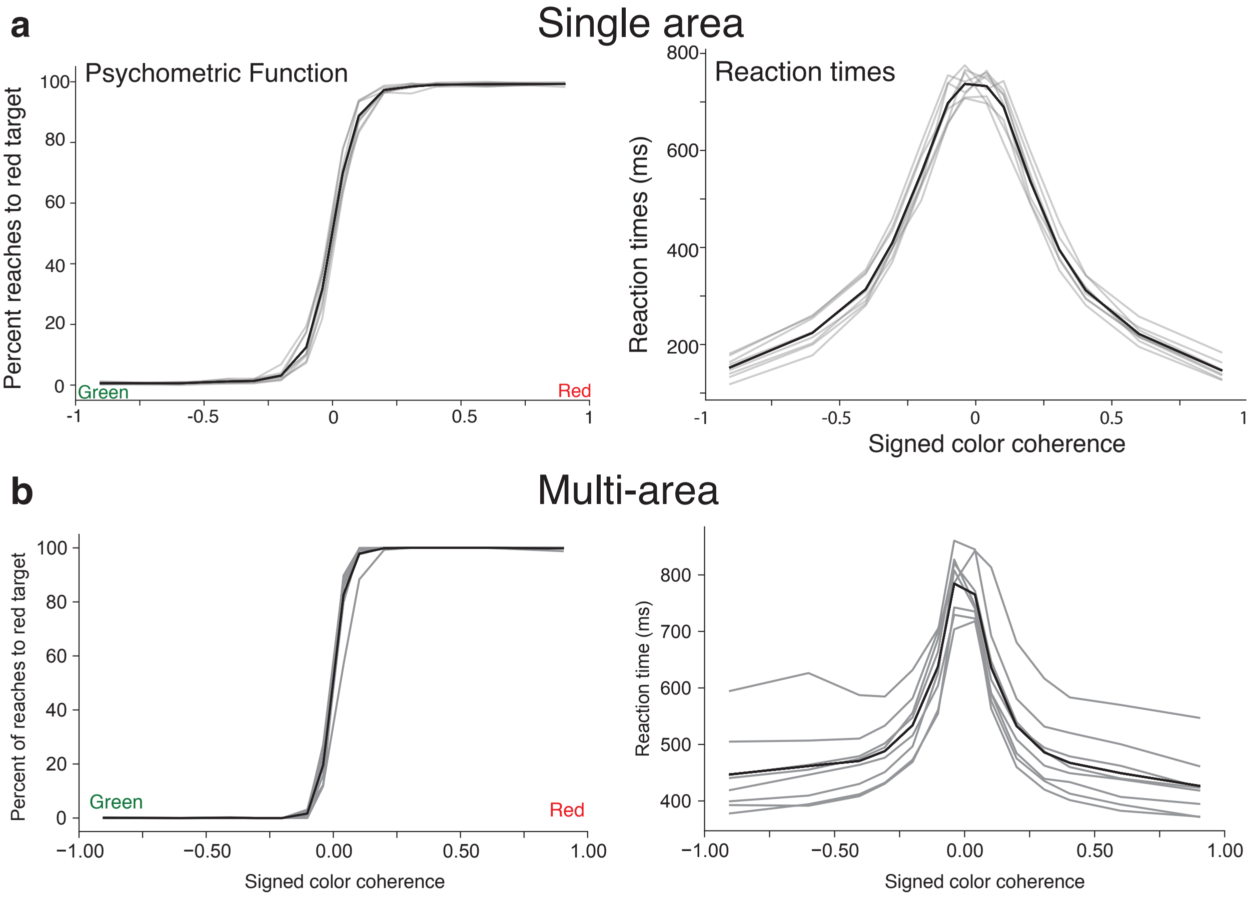

Figure A.1: Psychometric and reaction time curves for single-area **(a)** and multi-area RNNs **(b)** with Dale's law trained for this study. The hyperparameters used for these RNNs are described in Table A.1. Gray lines represent individual RNNs and the black solid line is the average across all RNNs.

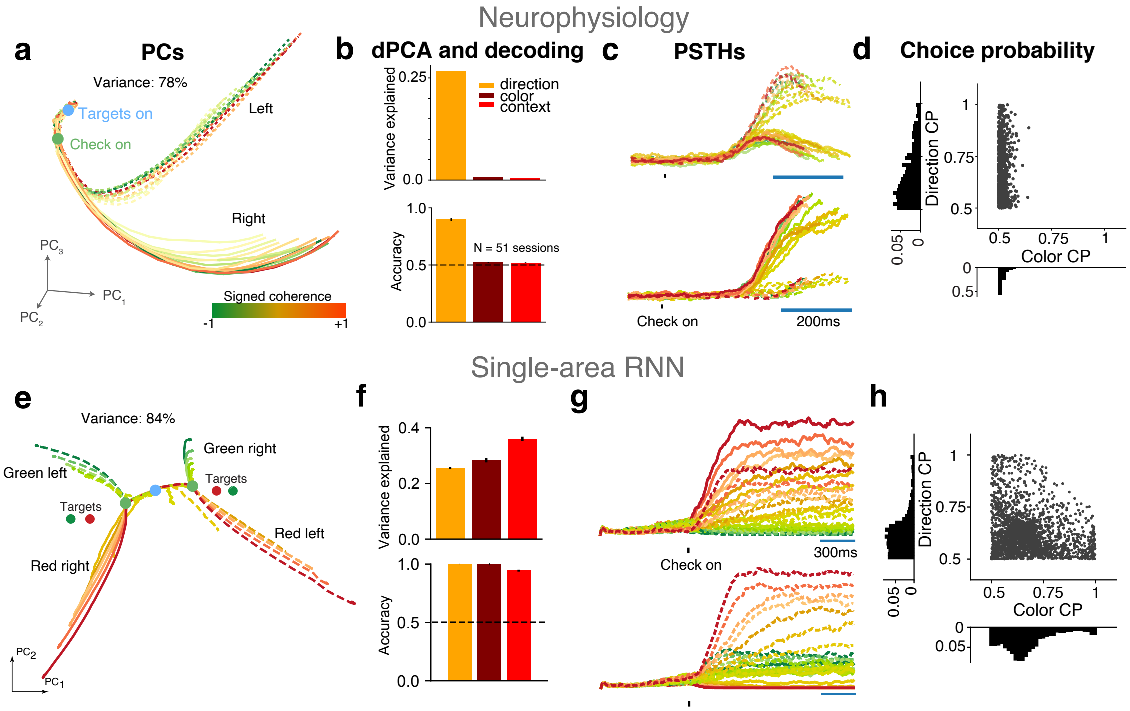

Figure A.2: **Single-area RNNs do not naturally reproduce PMd dynamics. (a-d)** Reproduced from Fig. 2.2 for comparison. **(e)** Single-area RNN neural trajectories in the top 2 PCs. Single-area RNNs had four trajectory motifs for each combination of (left vs right) and (red vs green). In the Targets epoch, the RNN's activity approached one of two locations in state space (light green dots), corresponding to the two target configurations. In the checkerboard epoch, trajectories separate based on the coherence of the checkerboard, causing 4 total distinct trajectory motifs. Although the direction decision is not separable in the principal components, the direction decision is separable in higher dimensions (see the direction axis found using dPCA in Fig. A.7a). **(f)** dPCA variance captured for the color (28%), context (26%), and direction (36%) axes for the RNN. The color and direction decisions, as well as the target configuration context, could be decoded from the RNN population activity well above chance. **(g)** Example RNN PSTHs, demonstrating coherence selectivity (top) and mixed selectivity (bottom). **(h)** Choice probability for simulated single-area RNN units. Many units have high color choice probabilities.

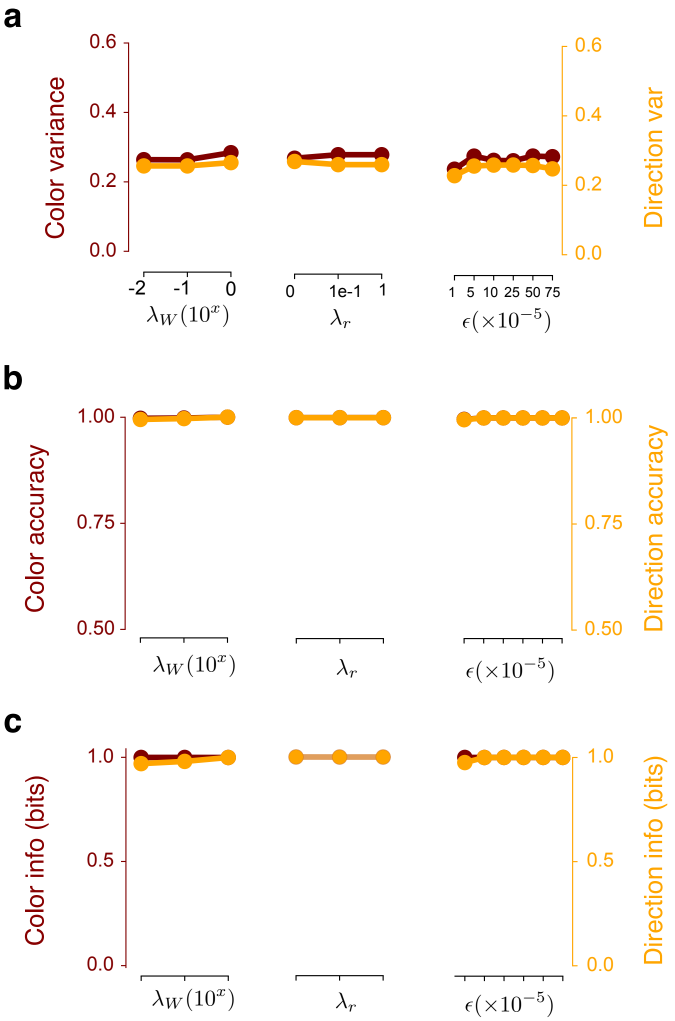

Figure A.3: **Hyperparameter sweeps for single-area RNNs**. **(a)** dPCA color and direction variance captured for three different regularization parameters (weight regularization: λ_w, rate regularization: λ_r, and learning rate: ϵ). There is a significant color representation in all optimized single-area RNNs. **(b)** Decode accuracy of the color and direction decision; color accuracy is at 1 (hidden behind direction accuracy) for the three different hyper parameters. The color decode accuracy (maroon) is at nearly 1 across all tested hyperparameters. These points are behind the direction decode accuracy (orange). **(c)** Mutual information estimate. The color mutual information (maroon) is nearly 1 across all tested hyperparameters.

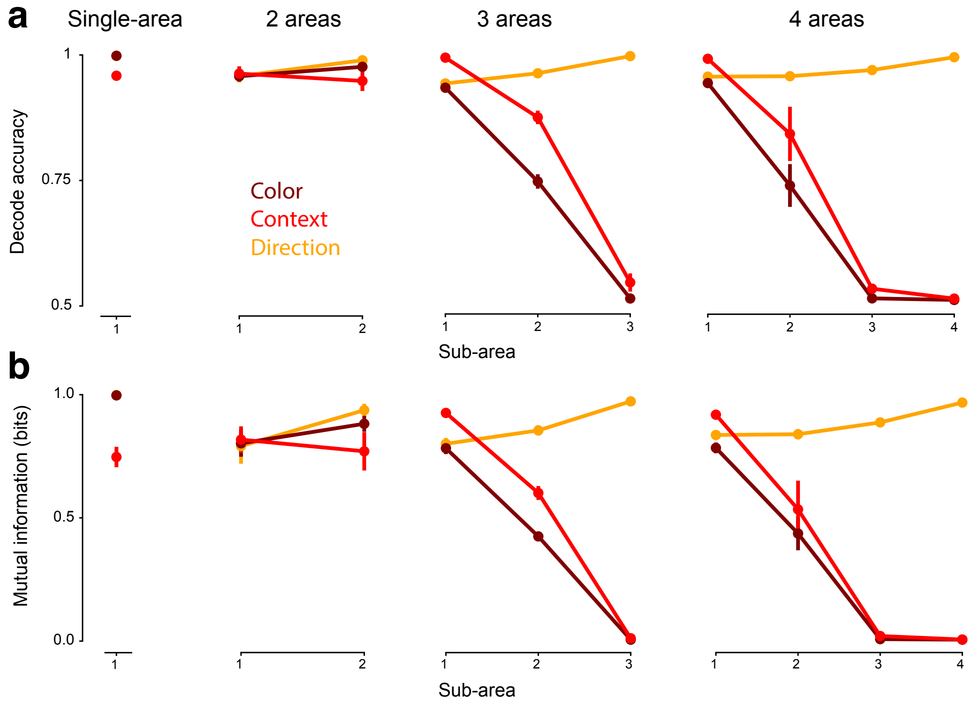

Figure A.4: **Decode accuracy and mutual information per area in multi-area RNNs**. **(a)** Decode accuracy in each area for 1- to 4-area RNNs for color, context, and direction corresponding to Fig. 2.3d. The 3- and 4-area RNNs had minimal color representations in their last area. Note that the 4-area RNN also has a minimal color representation in Area 3. **(b)** Mutual information in each area for 1- to 4-area RNNs. Color conventions as in Fig. 3. Red is context, dark brown is color, and orange is direction.

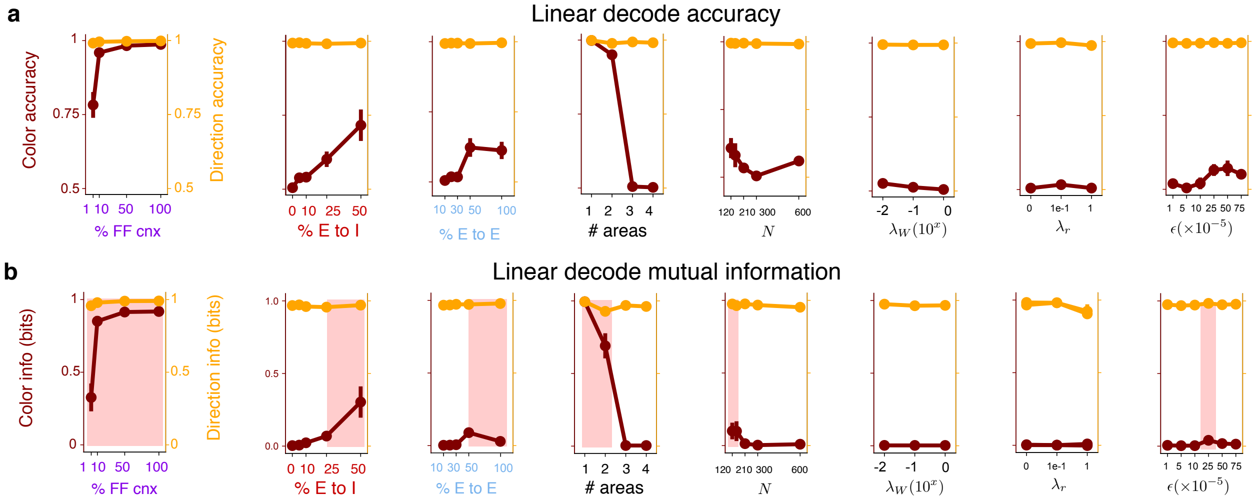

Figure A.5: **Results of Fig. 2.3 reproduced with a linear classifier**. This figure reproduces the simulations in Fig 3, but with a linear classifier. The main conclusions are upheld. **(a)** Linear decode accuracy for all hyperparameter sweeps shown in Fig. 2.3. **(a)** Mutual information estimated by using the linear network trained with cross entropy loss.

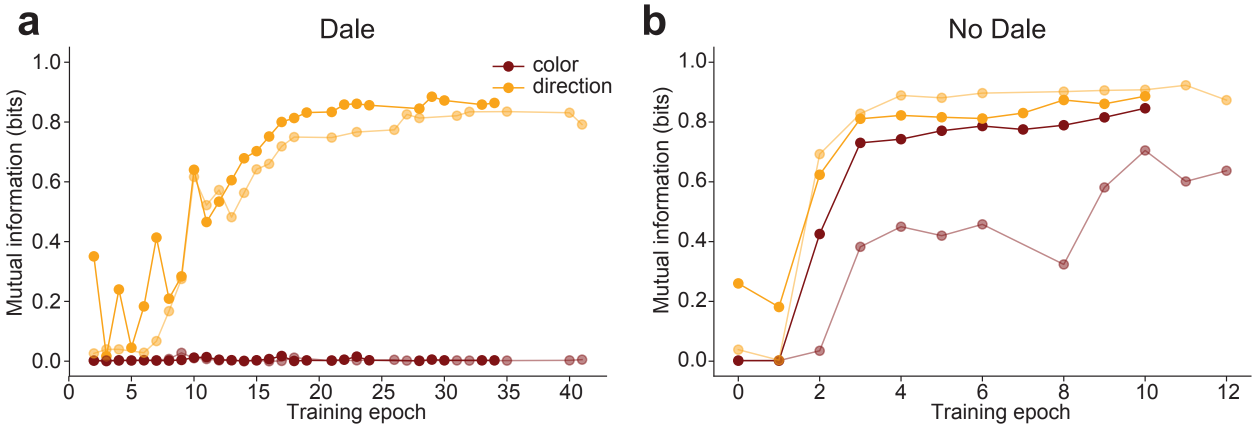

Figure A.6: **Color and direction information through training in Area 3.** Each "training epoch" represents 500 iterations of gradient descent. **(a)** In the PMd-like 3-area RNNs that were trained with Dale's law, color information in Area 3 remained near zero throughout training (two different representative networks, light and dark shade). **(b)** In the unconstrained 3-area RNNs, color information in Area 3 increased early in training and appeared to plateau (two different networks, light and dark shade). Networks were only saved if the loss function decreased, so certain training epochs are not present.

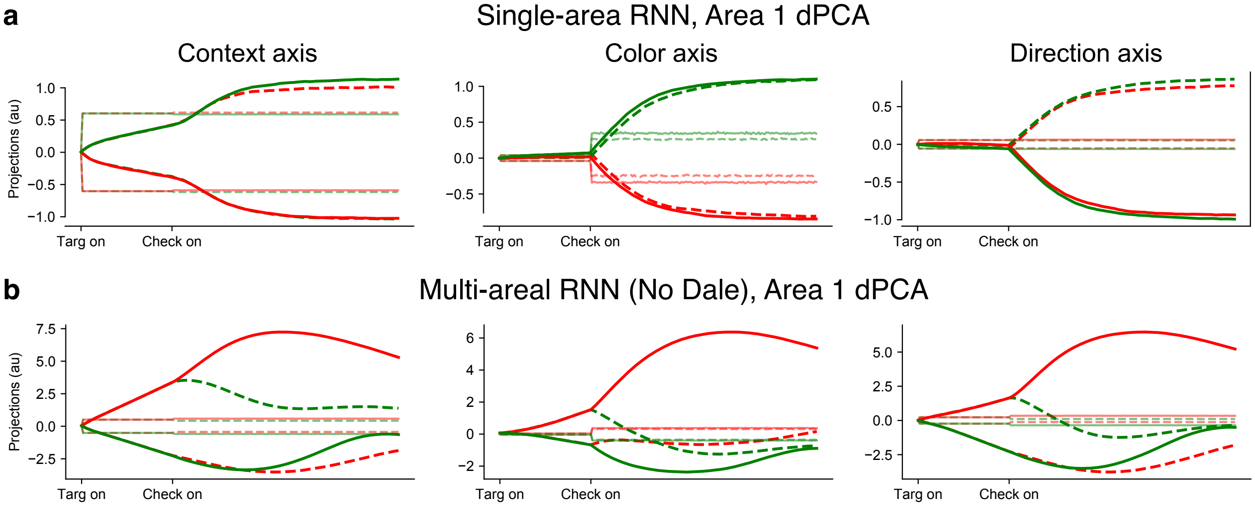

Figure A.7: **dPCA trajectories for single-area and 3-area RNNs with No Dale's law.** **(a)** Projections onto the dPCA context, color, and direction axes for a single-area RNN. dPCA was able to find axes that separate the context input, color decision, and direction decision. Importantly, in these networks, Inputs were non-zero on the direction axis. **(b)** dPCA projections for the unconstrained 3-area RNNs with color representation in Area 3. Inputs were similarly non-zero on the direction axis. The context inputs, color decision, and direction decision, had similar projection motifs.

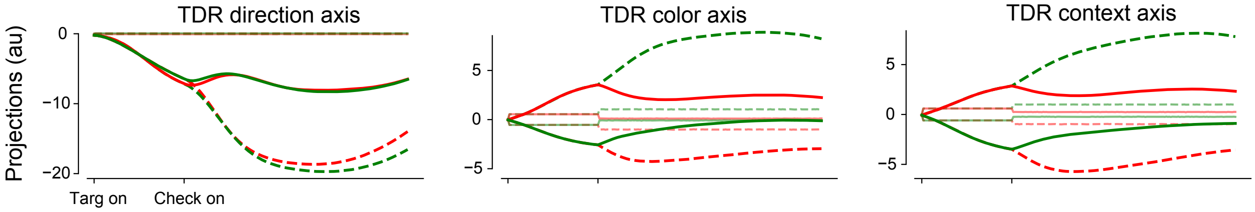

Figure A.8: **TDR results closely match dPCA results, and identifies mixed color and context axes.** The direction axis separated trajectories based on the direction choice. The color and context axes had trajectory separation depending on both color and context. We did not show the orthonormalized bases, because we found that the QR decomposition was susceptible to the order in which orthonormalization was performed. This is further evidence that the color and context axes are closely aligned.

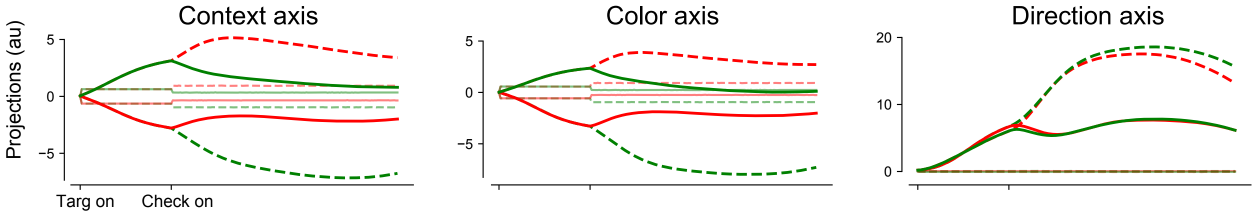

Figure A.9: **dPCA projections when only considering excitatory units.** We identified the dPCA principal axes for context, color, and direction using only excitatory units. Results are consistent with the results of Fig. 2.4d.

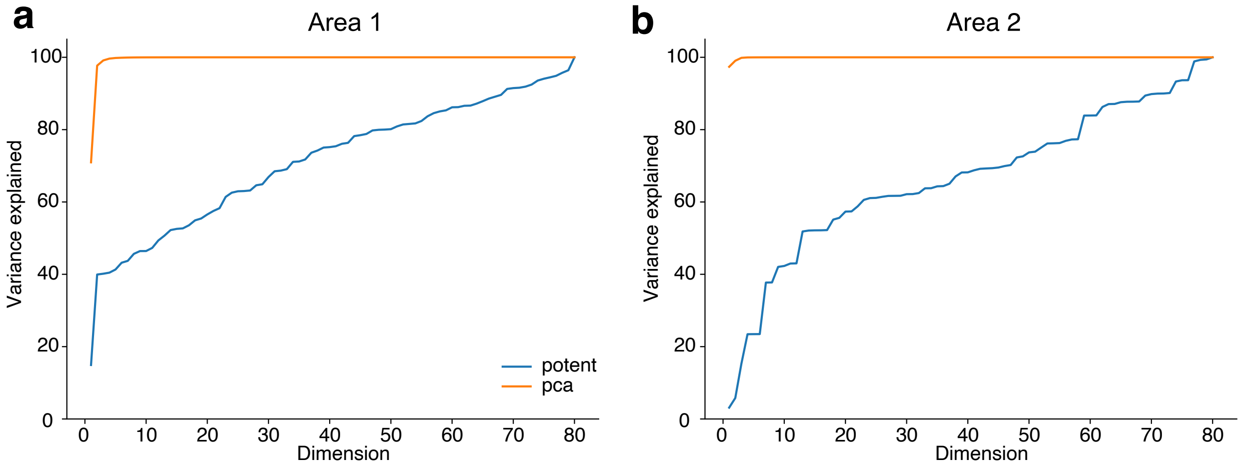

Figure A.10: **Relationship between PCs and inter-area potent space. (a)** Variance explained of the excitatory units in Area 1 by the top principal components and top dimensions of potent space of $\mathbf{W}_{21}$, swept across all dimensions. **(b)** Variance explained of the excitatory units in Area 2 by the top principal components and top dimensions of potent space of $\mathbf{W}_{32}$, swept across all dimensions. These plots show that the connections between areas do not necessarily propagate the most dominant axes of variability in the source area to the downstream area. Excitatory units were used for the comparison because only excitatory units are read out by subsequent areas. These results were upheld when comparing to the variance explained by the top principal components obtained from all units.

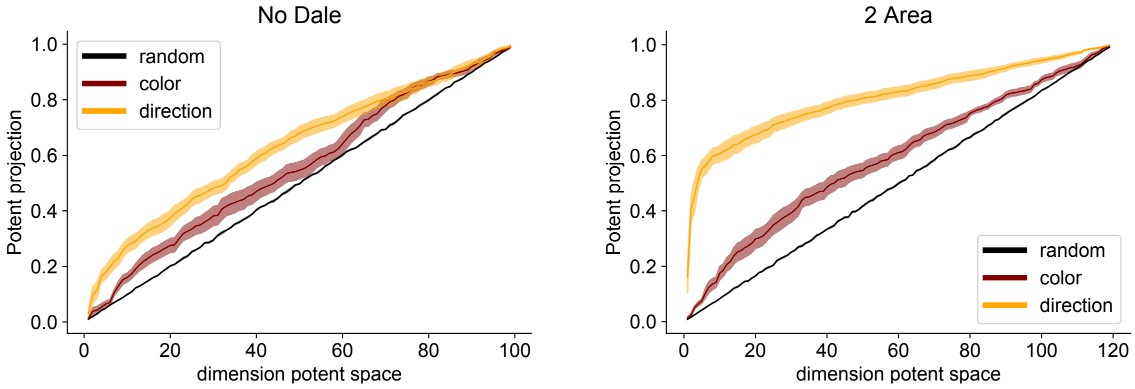

Figure A.11: Projections between Area 1 and Area 2 for a network without Dale's law **(left)** and a 2 area network **(right)**, averaged across 8 trained networks. The conventions are the same as in Fig. 2.5. The alignment of the direction axis with the top singular vectors is reduced (compare to Fig. 2.5).

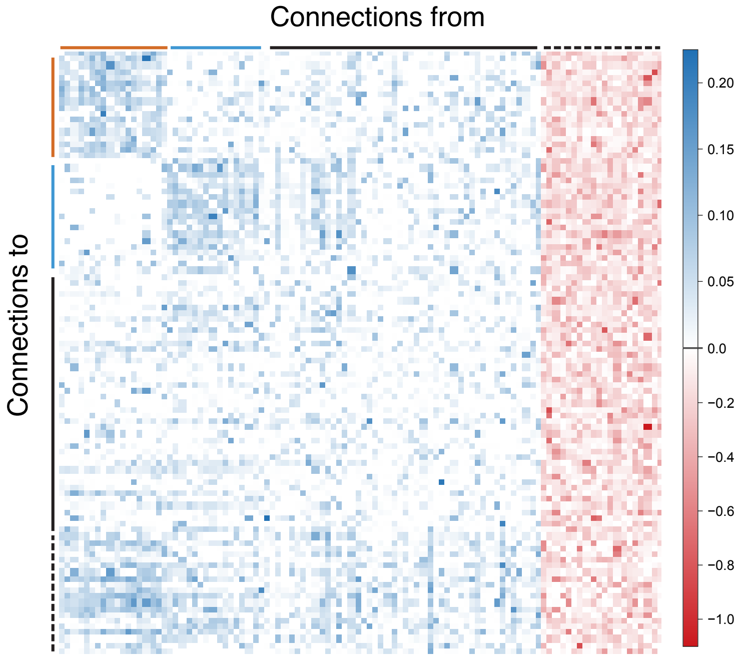

Figure A.12: **Structure of $\mathbf{W}_{33}$ of $\mathbf{W}_{\text{rec}}$.** Full connectivity matrix of $\mathbf{W}_{33}$, reordered so that the structured excitatory components lie at the top left. The matrix is composed of a structured excitatory component (orange and blue), a set of random excitatory units (black), and a set of inhibitory units (dashed black), with non-obvious structure. The averaged connectivity matrix is shown in Fig. 2.6e.

A.4 Supplementary Notes

Supplementary Note 1: Viewing the CB task as an XOR task

Here we show that a nonlinearity is necessary to solve the task, proving that the task cannot be solved by the linear layer $\mathbf{W}_{\text{in}}$. First, we note that the Checkerboard task corresponds to an exclusive-or (XOR) problem. If we identify the two target configurations as 0 or 1 (corresponding to green on left, or green on right respectively, with the red target on the complement side), and the dominant checkerboard color as 0 or 1 (for green or red, respectively), then the output direction d (identified as 0: left, 1: right) can be seen be in Table S1.

If the representation $\mathbf{r}$ was purely input driven, then:

$$\mathbf{r} = \mathbf{W}_{\text{in}}\mathbf{u}, \tag{A.9}$$

Our readout was a linear readout of the rates, i.e:

$$d = \mathbf{W}_{\text{out}}\mathbf{r} \tag{A.10}$$

The inputs $\mathbf{u}$ are the four dimensional input we trained with. But $\mathbf{u}$ is a linear transformation of two variables: the target orientation θ, and checkerboard color c, which each can take two values. That is, if we let $\mathbf{q} = [\theta, c]$, then, the inputs could be written as a linear transformation of $\mathbf{q}$:

$$\mathbf{u} = \mathbf{W}\mathbf{q}, \tag{A.11}$$

where $\mathbf{W}$ is a linear transformation. Since the mappings from $\mathbf{q}$ to d are all linear, they can be combined into a single linear transformation $\tilde{\mathbf{W}}$, i.e.,

$$d = \mathbf{W}_{\text{out}}\mathbf{W}_{\text{in}}\mathbf{W}\mathbf{q} = \tilde{\mathbf{W}}\mathbf{q}. \tag{A.12}$$

It is not possible for a linear classifier to solve the XOR problem by classifying correct outputs [50]. Hence, the trained RNNs cannot purely be input driven, and requires nonlinearity from the recurrent interactions to solve the task. After nonlinear processing, the left or right decision could be achieved by a linear readout of the units.

target configuration	**color**	**direction**
0	0	0
0	1	1
1	0	1
1	1	0

Table A.2: Checkerboard task truth table

context	**signed color**	**signed motion**	**direction**
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

Table A.3: [99] model truth table

Supplementary Note 2: Mutual Information Estimation

The entropy of a distribution is defined as

$$H(x) = \mathbb{E}_{x \sim p(x)} \left[\log \frac{1}{p(x)} \right]. \tag{A.13}$$

The mutual information, $I(X;Y)$, can be written in terms on an entropy term and as conditional entropy term:

$$I(Z;Y) = H(Y) - H(Y|Z). \tag{A.14}$$

We want to show that the usable information lower bounds the mutual information:

$$I(Z;Y) \geq I_u(Z;Y) := H(Y) - L_{CE}(p(y|z), q(y|z)) \tag{A.15}$$

It suffices to show that:

$$H(Y|Z) \leq L_{CE} \tag{A.16}$$

where L_{CE} is the cross-entropy loss on the test set. For our study, $H(Y)$ represented the known distribution of output classes, which in our case were equiprobable.

$$\begin{aligned} H(Y|Z) &:= \mathbb{E}_{(z,y) \sim p(z,y)} \left[\log \frac{1}{p(y|z)} \right] & \text{(A.17)} \\ &= \underbrace{\mathbb{E}_{(z,y) \sim p(z,y)} \left[\log \frac{1}{q(y|z)} \right]}_{\text{cross-entropy loss}} - \underbrace{\mathbb{E}_{z \sim p(z)} \left[\text{KL}(p(y|z) || q(y|z) \right]}_{\geq 0}, & \text{(A.18)} \\ &\leq \mathbb{E}_{(z,y) \sim p(z,y)} \left[\log \frac{1}{q(y|z)} \right] := L_{CE} & \text{(A.19)} \end{aligned}$$

To approximate $H(Y|Z)$, we first trained a neural network with cross-entropy loss to predict the output, Y, given the hidden activations, Z, learning a distribution $q(y|z)$. The KL denotes the Kullback-Liebler divergence. We multiplied (and divided) by an arbitrary variational distribution, $q(y|z)$, in the logarithm of equation A.17, leading to equation A.18. The first term in equation A.18 is the cross-entropy loss commonly used for training neural networks. The second term is a KL divergence, and is therefore non-negative. In our approximator, the distribution, $q(y|x)$, is parametrized by a neural network. When the distribution $q(y|z) = p(y|z)$, our variational approximation of $H(Y|Z)$, and hence approximation of $I(Z;Y)$ is exact [15, 78, 108].

In the paper, we additionally report the accuracy of the neural network on the test set. This differs from the cross-entropy in that the cross-entropy incorporates a weighted measure of the accuracy based on how “certain” the network is, while the accuracy does not.

Appendix B

Supplementary Material for Chapter 3

B.1 Materials and Methods

B.2 Additional description of analyses

Decoding analysis for DLPFC and PMd data

For DLPFC and PMd data, we calculated decoding accuracy using 400 ms bins. We report numbers in a window [-300ms, +100 ms] aligned to movement onset. We used the Python *sklearn.svm.SVC* command with 80% training and 20 % test sets. Decoding analyses were performed using 2-49 simultaneously recorded units from Plexon U-probes and the averages reported are across PMd and DLPFC sessions, respectively. To assess whether decoding accuracies were significant, we choose confidence interval $CI = 0.5 + SEM * 2.58$ (99 percentile). The decoding accuracy for direction, color and context variables of a session was judged to be significant if it lies above the CI. For DLPFC, direction, color and context decoding accuracy of 100%, 80.4%, 90.2% sessions were judged to be significantly above chance; for PMd, 100%, 58.6%, 55.2% sessions demonstrated significant decoding accuracy to direction, color and context.

Mutual information was calculated by computing $H(Y) - L_{CE}$ where $H(Y)$ was 1 and

L_{CE} denotes the cross entropy loss (in bits). We computed the decoding and information only for sessions with decoding accuracy significantly above chance. Negative mutual information was set to zero.

B.2.1 Decoding and Mutual information for RNNs

We used a decoder and mutual information approximation to quantify the amount of information (color, context, direction) present in the network. We trained a neural network to predict a relevant choice (for example, color) on a test set from the activity of a population of units. We used 700 trials for training, and 2100 independent trials for testing. To generate the trials for training and testing, we increased the recurrent noise to be drawn from the distribution ($\mathcal{N}(0, 0.1^2)$) to prevent overfitting. For each trial, we averaged data in a window [-300ms, +100ms] around reaction time.

We trained a neural network with 3 layers, 64 units per layer, leakyRelu activation (α=0.2), and dropout (p=0.5), using stochastic gradient descent, to predict the choice given the activity of the population. We removed the leakyRelu activation for the linear network, and increased dropout (p=0.8). For both the nonlinear and linear network, we trained the neural network to minimize the cross-entropy loss. We used the same neural network from the decode to compute an approximation to mutual information, described in Supplementary Note 2.

B.2.2 RNN behavior

To evaluate the RNN's psychometric curve and reaction-time behavior, we generated 200 trials for each of the 28 conditions, producing 400 trials for each signed coherence. For these trials, we calculated the proportion of red decisions by the RNN. This corresponds to all trials where the DV output for the red target first crossed the preset threshold of 0.6; or, if no DV output crossed the threshold of 0.6, if the DV corresponding to the red target exceeded that corresponding to the green target. The reaction time was defined to be the time between checkerboard onset to the first time a DV output exceeded the preset threshold of 0.6. If the DV output never exceeded a threshold of 0.6, in the reported results, we did not calculate a RT for this trial.

B.2.3 dPCA

Demixed principal components analysis (dPCA) is a dimensionality reduction technique that provides a projection of the data onto task related dimensions while preserving overall variance [85]. dPCA achieves these aims by minimizing a loss function:

$$L_{dpca} = \sum_{c} \|\mathbf{X}_c - \mathbf{P}_c\mathbf{D}_c\mathbf{X}\|^2. \tag{B.1}$$

Here, $\mathbf{X}_c$ refers to data averaged over a "dPCA condition" (such as time, coherence, context, color, or direction), having the same shape as $\mathbf{X} \in \mathbb{R}^{N \times cT}$, but with the entries replaced with the condition-averaged response. The aim is to recover (per dPCA condition c) a $\mathbf{P}_c$ and $\mathbf{D}_c$ matrix. $\mathbf{P}_c$ is constrained to have orthonormal columns, while $\mathbf{D}_c$ is unconstrained. The number of columns of $\mathbf{P}_c$ and rows of $\mathbf{D}_c$ reflects the number of components one seeks to find per condition. The column of $\mathbf{P}_c$ reflects how much the demixed data contributes to each neuron. We use the principal axes from $\mathbf{P}_c$ to compute the axis overlap, as in Kobak et al [85]. We used axes of dimension 1 for RNNs, which were sufficient to capture most color, context, or direction variance. For the neural data, we used five components for direction, color and context since the PMd data was higher dimensional than the RNNs.

For multi-area analyses, we separated the units for each area and found the task-relevant axes for this subset of units. For the inter-area analyses, we used RNNs with only excitatory connections, and therefore found the color and direction axis using only the excitatory units. In all other analyses, all units were used to identify the axes. For RNN activity, we performed dPCA using activity over the entire trial.

For neural data, due to the stochasticity in task design, there is a trial-by-trial difference in interval between target and checkerboard onset (TC interval). The reaction time (from the checkerboard onset to monkey's hand movement initiation) also varies for each trial. To align time events across trials, we restretched the firing rates in each trial. For DLPFC units, each trial was aligned to targets onset first. Median reaction time (527ms) and TC interval (735ms) were calculated by combining every trial in the database. For each trial, TC interval and reaction time was either compressed or stretched to the median values through linear interpolation. After the data restretching, we choose the data window **T** as 1300ms, from -100ms to 1200ms around target onset with sample size of 1ms. For every unit **n** in total units number **N**, we averaged the single-trial firing rate by stimulus **S** (checkerboard dominant color, green or red) and decision of choice **D** (left or right). As a result, a 4D firing-rate matrix $\mathbf{X}^{\mathbf{N}\times\mathbf{S}\times\mathbf{D}\times\mathbf{T}}$ was created as input to demixed principal component analysis algorithm.

For PMd units, activities before checkerboard onset were minimal. As a result, each trial was aligned to target onset and a segment with time window of $[-100ms, 367ms]$ was chosen first. Then the same trial was aligned to checkerboard first and a segment with a window of $[-368ms, 465ms]$ was chosen. The final restretched data was the concatenation of these two data segments.

When computing the overlap in Fig. 3.4, we averaged across 8 initializations, and computed the PSTHs over 700 trials. In our dpca variance sweeps (Fig. 3.5), we computed the PSTHs over 280 trials.

B.2.4 PCA

Principal components analysis (PCA) is a dimensionality reduction technique that projects high-dimension data into low-dimensional axis which maximize the variance in the data. PCA provides low-dimensional projections by minimizing the loss function:

$$L_{pca} = \sum \|\mathbf{X} - \mathbf{D}^T\mathbf{D}\mathbf{X}\|^2. \tag{B.2}$$

$\mathbf{X^{N \times T}}$ is high-dimension raw data and $\mathbf{D^{N \times N}}$ is the decoding matrix. The low-dimension trajectories $\mathbf{x^{M \times T}}$ $(M < N)$ calculated by multiplying first M rows of $\mathbf{D}$ by $\mathbf{X}$.

Before applying PCA on the data, the raw data was preprocessed by data normalization and average firing rate removal:

Condition independent signal removal for PCA

The condition independent signal is another source that explains substantial amount of population variance other than task-related signal. Before conducting principal component analysis (PCA), we calculated the average firing rate of each single unit $\mathbf{X^{1 \times 1 \times 1 \times T}}$ over all stimulus and decision conditions and subtracted this condition independent signal from the time-restretched data $\mathbf{X^{1 \times S \times D \times T}}$.

Canonical correlation

We applied CCA to assess the similarity between neural activity and the artificial unit activity [132]. Before applying CCA, we performed principal component analysis to reduce the dimensionality of the artificial and neural activity to remove noise, which can be arbitrarily reshaped to increase the canonical correlation [132]. We reduced the dimensionality of PMd data to 2 (which captures over 80% of the PMd variance). For DLPFC, 18 dimensions were required to capture over 80% of the variance, but at such a high dimensionality, noise can be reshaped to significantly increase the canonical correlations. For DLPFC, we therefore show a comparison to the top 4 PCs in Fig. 3.3e. However, the trends held irrespective of the DLPFC dimensionality we chose, as shown in Fig. B.3. In all cases, we compared to CCA with the number of dimensions equal to 2, looking at the We report the average CCA correlation coefficients in Fig. 3.3e using times in a window of [-400ms, 400ms] aligned to checkerboard onset. The data was binned in 10ms bins.

B.2.5 Analyses of inputs and activity

In order to disentangle the effects of external inputs and recurrence, in Fig. B.6, we evaluated the input contribution and overall activity. For Area 1, we defined the input contribution as $\mathbf{W}_{\text{in}}\mathbf{u}_t$, and for areas 2 and 3, we defined the input contribution as $\mathbf{W}_{21}\mathbf{r}_t^1$, and $\mathbf{W}_{32}\mathbf{r}_t^2$ respectively, where $\mathbf{r}_t^m$ denotes the activity of the units in area m. The activity $\mathbf{r}_t^m$ corresponds to the firing rate that experimentalists could measure, reflecting a combination of input and recurrent interactions. For constant inputs, a stable value of the activity implies there is little recurrent processing.

B.2.6 Inter-Area Projection Analyses

To calculate the overlap between the color and direction axes with the potent and null spaces, we performed singular value decomposition on the inter-area connections, $\mathbf{W}_{21}$ and $\mathbf{W}_{32}$. $\mathbf{W}_{21}$ and $\mathbf{W}_{32}$ were 80×80 matrices, and were full rank. Nevertheless, they had near some zero singular values, indicating that the effective rank of the matrix was less than 80. We defined the potent dimensions to be the top m right singular vectors, while the null dimensions were the remaining $80 - m$ right singular vectors.

We performed the analyses of Fig. 3.4f by varying the potent and null dimensions, sweeping m from 1 to 80. For each defined potent and null space, we calculated the axis overlap between the direction (or color) axis and the potent (or null) space by computing the L2-norm of the orthogonal projection (squared). We report the squared quantity because the expectation of the norm of a projection of a random vector onto an m-dimensional subspace of an n-dimensional space is m/n. We include an approximation of the expectation of the projection of a random vector in Fig. 3.4 by averaging the projection of 100 random vectors. Our results show that the direction axis was always more aligned with potent dimensions than the color axis, irrespective of the choice of m, and that the direction axis was preferentially aligned with the top singular vector.

Supplementary Information for: "A cortical information bottleneck during decision-making"

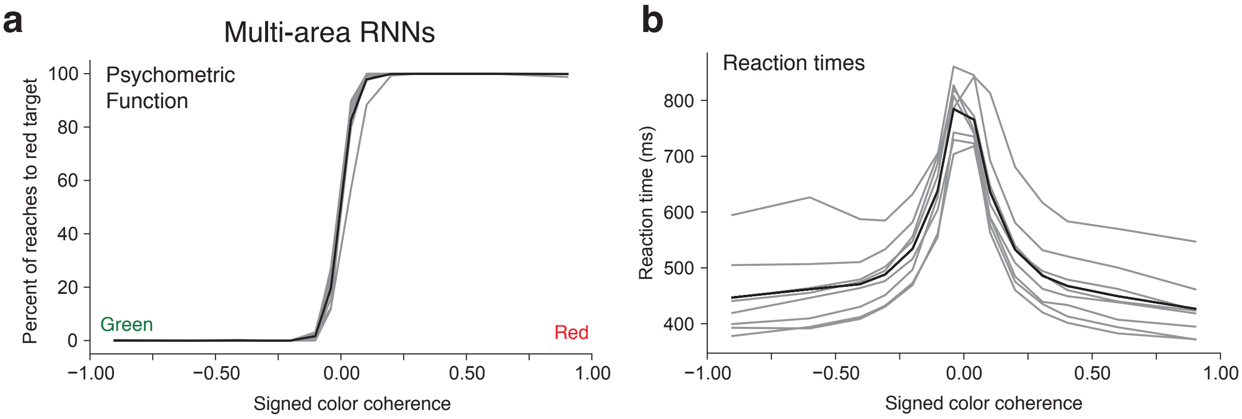

Figure B.1: **(a)** Psychometric and **(b)** reaction time curves for multi-area RNNs. The hyperparameters used for these RNNs are described in Table A.1. Gray lines represent individual RNNs and the black solid line is the average across all RNNs.

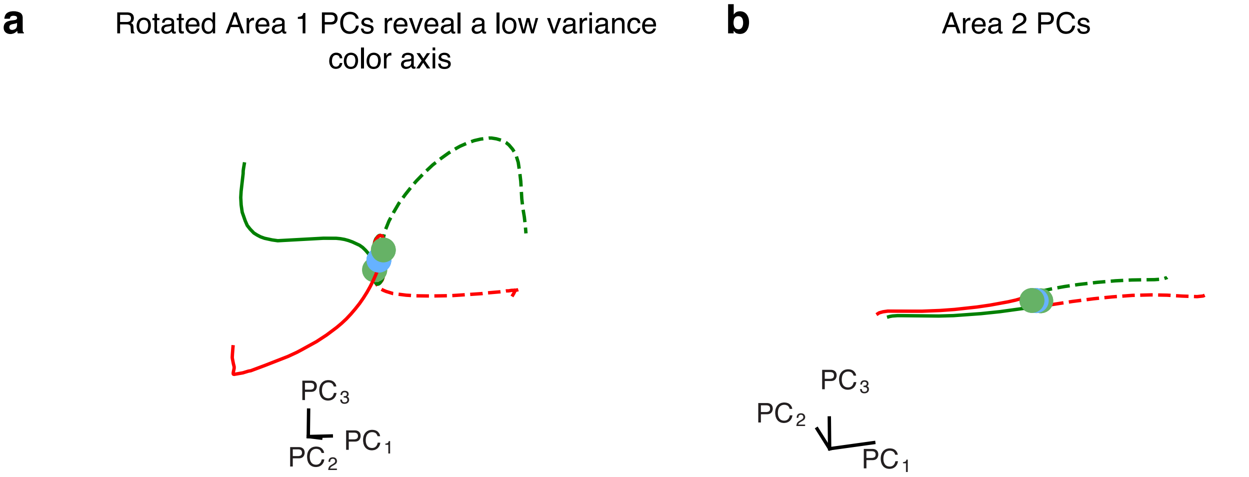

Figure B.2: **(a)** Another rotation of the first three PCs for Area 1 RNN, with PC_3 amplified to show that there is a low variance color axis. **(b)** Area 2 PCs in the same projection as used in Figure 3. While these PCs qualitatively appear to represent the direction decision, they are distinct from Area 3, with Area 3 demonstrating a stronger resemblance to PMd activity.

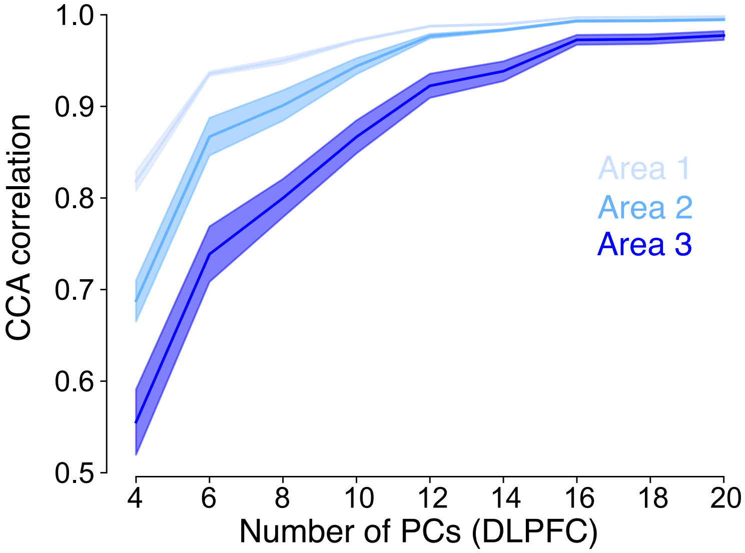

Figure B.3: Because DLPFC is higher-dimensional than PMd, we performed the CCA correlation coefficient comparison to Areas 1-3 of the RNN varying the number of dimensions used for the DLPFC PCs. Note that as dimensionality increases, CCA correlation coefficient increases because additional dimensions, which are low variance, can be weighted to better reproduce the RNN PCs. We nevertheless observe that Area 1 has the highest CCA correlation to DLPFC, while Area 3 has the least.

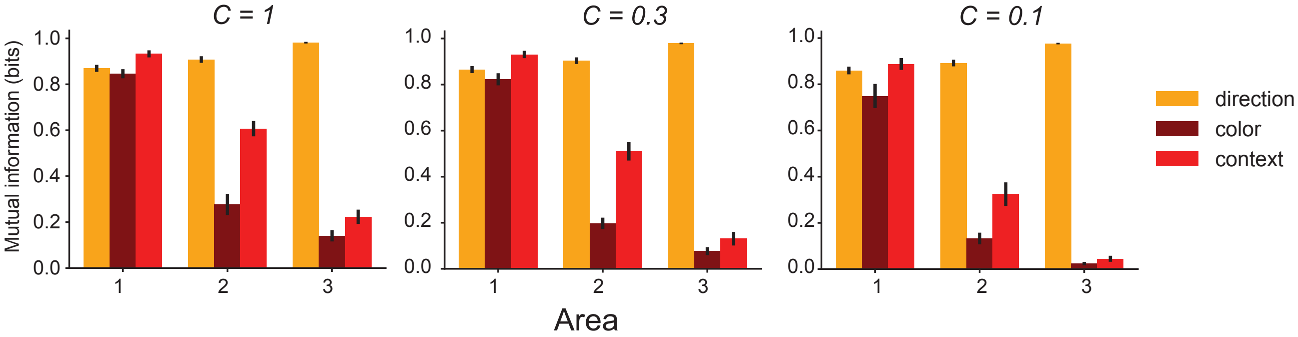

Figure B.4: SVM Mutual Information (approximated using the Usable Information) for each RNN area as a function of increasing decoder regularization C. A lower C implies more regularization.

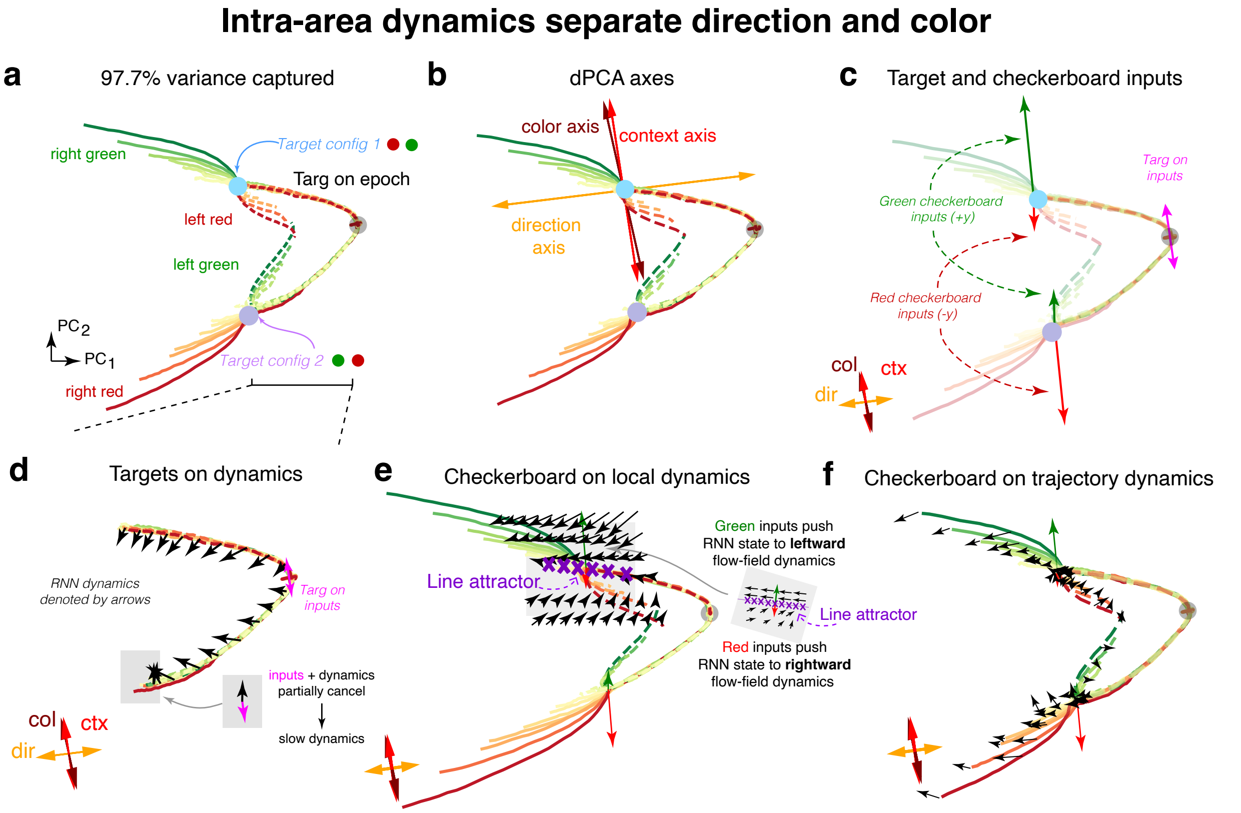

Figure B.5: **Candidate mechanism for axis orthogonalization.** **(a)** Top 2 PCs of RNN Area 1 activity. Trajectories are now colored based on the coherence of the checkerboard, and the condition-independent signal is not removed. We did not remove the condition-independent signal so we could directly study the high-dimensional dynamics of the RNN and its equilibrium states. The trajectories separate to two regions corresponding to the two potential target configurations (Target config 1 in blue, Target config 2 in purple). The trajectories then separate upon checkerboard color input, leading to four trajectory motifs. **(b)** Projection of the dPCA principal axes onto the PCs. **(c)** Projection of the context and color inputs onto the PCs. Context inputs are shown in pink, a strongly green checkerboard in green, and a strongly red checkerboard in red. Irrespective of the target configuration, green checkerboards cause the RNN state to increase along PC$_2$ while red checkerboards cause the RNN state to decrease along PC$_2$. The strength of the input representation is state-dependent: checkerboards corresponding to left reaches, whether they are green or red, cause smaller movements of the RNN state along the color axis. **(d)** Visualization of RNN dynamics and inputs during the target presentation. In the Targets On epoch, context inputs cause movement along the vertical context axis. The RNN dynamics implemented a leftward flow-field that pushed the RNN state into an attractor region of slow dynamics. **(e)** At the Target config 1 attractor, we plot the local dynamics using a previously described technique [68]. The RNN implements approximately opposing flow fields above and below a line attractor. Above the attractor, a leftward flow-field increases direction axis activity, while below the attractor, a rightward flow-field decreases direction axis activity. A green checkerboard input therefore pushes the RNN state into the leftward flow-field (solid green trajectories) while a red checkerboard input pushes the RNN state into a rightward flow-field (dotted red trajectories). This computes the direction choice in a given context, while allowing the direction axis to be orthogonal to color inputs. Arrows are not to scale; checkerboard inputs have been amplified to be visible. **(f)** Visualized dynamics across multiple trajectory motifs. These dynamics hold in both target configurations leading to separation of right and left decisions on the direction axis. Arrows are not to scale, for visualization purposes.

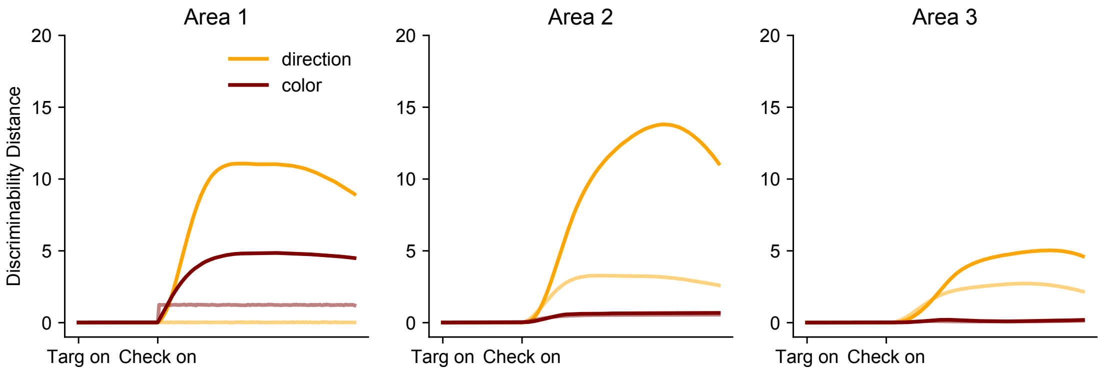

Figure B.6: The norm of the direction discriminability (left red - right red + left green - right green)/2 and color discriminability (left green - left red + right green - right red)/2 as a function of the processing area. The inputs are shown in lighter transparency and the overall activity is shown in solid lines. Area 1 has significant recurrence evidenced by a large separation between the input and overall activity. For our exemplar network, there is very little evidence of recurrent filtering of color information (i.e recurrent activity is never below inputs).

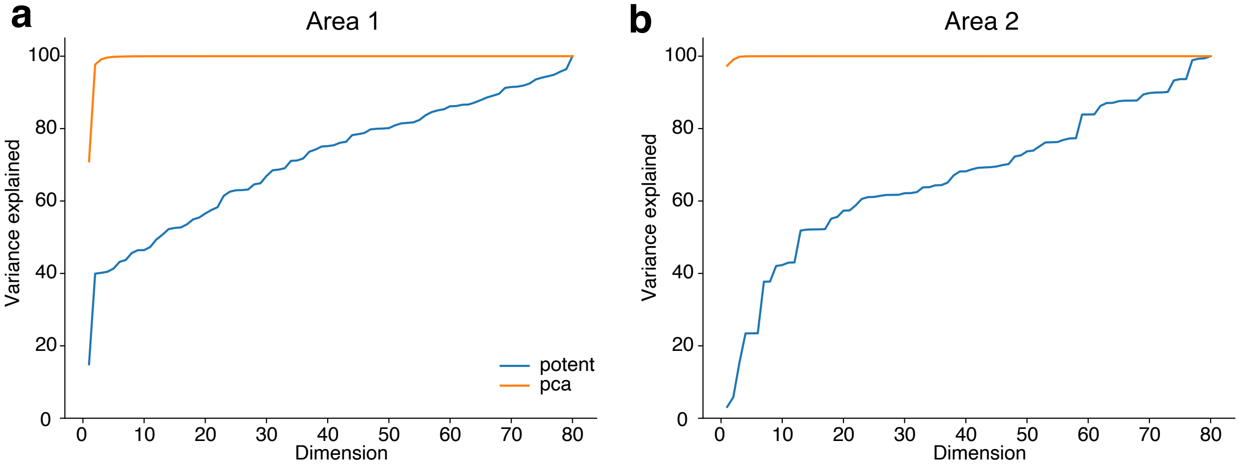

Figure B.7: **Relationship between PCs and inter-area potent space. (a)** Variance explained of the excitatory units in Area 1 by the top principal components and top dimensions of potent space of $\mathbf{W}_{21}$, swept across all dimensions. **(b)** Variance explained of the excitatory units in Area 2 by the top principal components and top dimensions of potent space of $\mathbf{W}_{32}$, swept across all dimensions. These plots show that the connections between areas do not necessarily propagate the most dominant axes of variability in the source area to the downstream area. Excitatory units were used for the comparison because only excitatory units are read out by subsequent areas. These results were upheld when comparing to the variance explained by the top principal components obtained from all units.

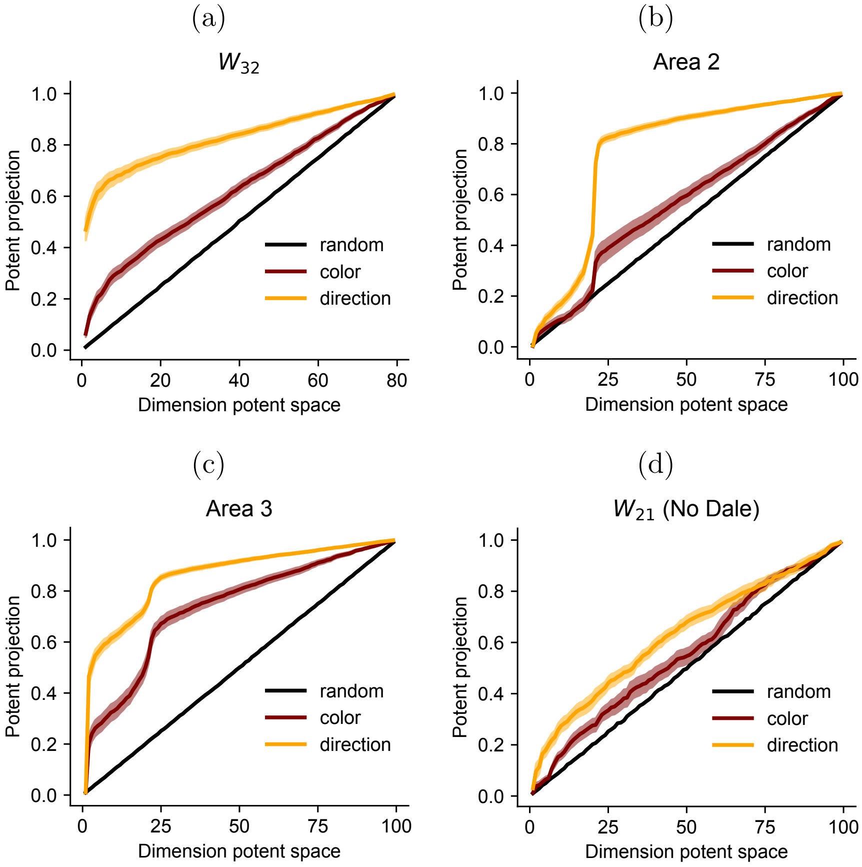

Figure B.8: **(a)** Alignment of dpca color and context axes from area 2 with inter-areal connections $\mathbf{W}_{32}$. **((b,c)** Alignment of dpca axes with intra-areal recurrent matrices for 3 area dale networks (Area 2 and Area 3). **(d).** Alignment of dpca axes in area 1 with $\mathbf{W}_{21}$ for networks without Dale's law. In contrast to Fig. 3.4f, direction information is not preferentially propagated. Same conventions as Fig. 3.4c,f.

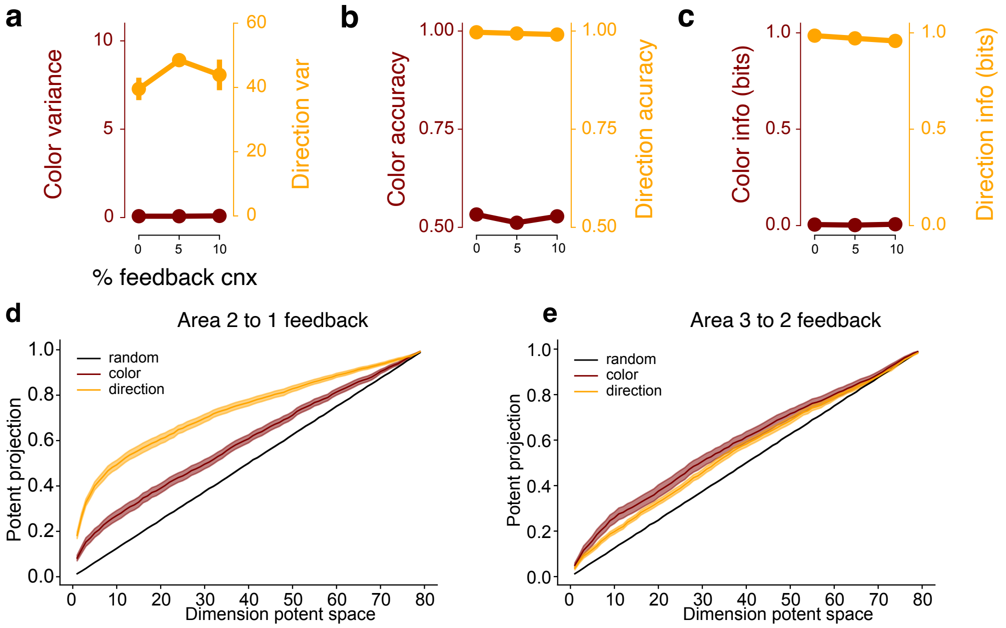

Figure B.9: **Effect of feedback connections (a)** dPCA variance in area 3 of RNNs where we varied the amount of feedback connectivity. RNNs exhibited nearly zero dPCA color variance in Area 3 across networks with 0%, 5%, and 10% feedback connections. **(b, c)** RNNs also exhibited minimal color representations, achieving nearly chance levels of decode accuracy and nearly zero mutual information. **(d, e)** Feedback projections of the color and direction axis on the feedback inter-area matrix between **(d)** area 2 and area 1, and **(e)** area 3 and area 2 (for networks trained with 5% feedback connections, across variable feedforward connectivity percentages).

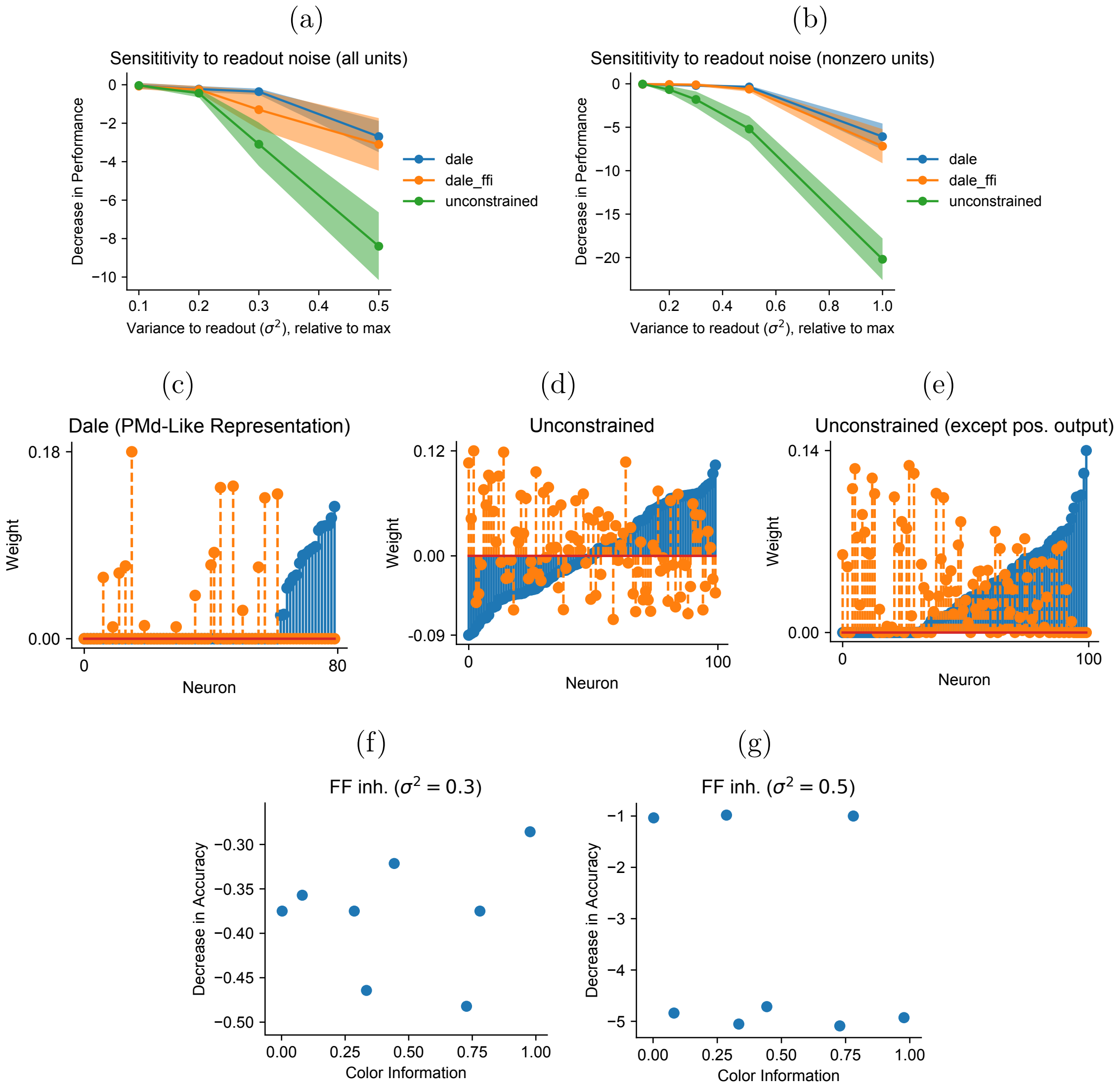

Figure B.10: **Potential multi-area computational advantage. (Top Row)** Sensitivity to isotropic readout noise added to the output weights. **(a)** Noise added to all units in output (even the zero weights). **(b)** Noise only added to nonzero units. **(Middle Row)** Readout weights for left (dashed orange) and right (blue) reaches. **(c)** Readout weight with Dale's Law enforced, (**d**) Readout weights in unconstrained networks. (**e**) Readout weights in unconstrained but ensuring positive outputs. **(Bottom Row)** No correlation between robustness to noise and usable color information across random initializations for networks with 10% feedforward inhibition, where after training some networks had color information (Fig. 3.5b). We used a noise perturbation to each unit of variance **(f)** $\sigma^2 = 0.3$ and **(g)** $\sigma^2 = 0.5$.

Appendix C

Supplementary Material for Chapter 4

C.1 Proofs

C.1.1 Usable information lower bounds the mutual information

The entropy of a distribution is defined as

$$H(x) = \mathbb{E}_{x \sim p(x)} \left[\log \frac{1}{p(x)} \right]. \tag{C.1}$$

The mutual information, $I(X;Y)$, can be written in terms of an entropy term and a conditional entropy term:

$$I(Z;Y) = H(Y) - H(Y|Z). \tag{C.2}$$

We want to show that:

$$I(Z;Y) \geq I_u(Z;Y) := H(Y) - L_{CE}(p(y|z), q(y|z)) \tag{C.3}$$

It suffices to show that:

$$H(Y|Z) \leq L_{CE} \tag{C.4}$$

where L_{CE} is the cross-entropy loss on the test set. For our study, $H(Y)$ represented the known distribution of output classes, which in our case were equiprobable.

$$H(Y|Z) := \mathbb{E}_{(z,y)\sim p(z,y)}\left[\log \frac{1}{p(y|z)}\right] \tag{C.5}$$

$$= \underbrace{\mathbb{E}_{(z,y)\sim p(z,y)}\left[\log \frac{1}{q(y|z)}\right]}_{\text{cross-entropy loss}} - \underbrace{\mathbb{E}_{z\sim p(z)}\left[\mathrm{KL}(p(y|z)||q(y|z)\right]}_{\geq 0}, \tag{C.6}$$

$$\leq \mathbb{E}_{(z,y)\sim p(z,y)}\left[\log \frac{1}{q(y|z)}\right] := L_{CE} \tag{C.7}$$

To approximate $H(Y|Z)$, we first trained a neural network with cross-entropy loss to predict the output, Y, given the hidden activations, Z, learning a distribution $q(y|z)$. The KL denotes the Kullback-Liebler divergence. We multiplied (and divided) by an arbitrary variational distribution $q(y|z)$ in the logarithm of equation C.5, leading to equation C.6. The first term in equation C.6 is the cross-entropy loss commonly used for training neural networks. The second term is a KL divergence and is therefore non-negative. In our approximator, the distribution $q(y|x)$ is parametrized by a neural network. When the distribution $q(y|z) = p(y|z)$, our variational approximation of $H(Y|Z)$, and hence approximation of $I(Z;Y)$ is exact [15, 108].

C.2 Additional results and details in the Checkerboard Task

C.2.1 SGD with non-random initialization may not form minimal representations in the CB task

Implicit regularization in SGD is hypothesized to result in a minimal representation through compression of irrelevant input information, also called a "forgetting" phase [2, 3, 123]. We tested this hypothesis by initializing networks with significant color information, and subsequently performing SGD on the CB task. We then evaluated whether SGD resulted in networks with minimal color representations. We initialized the weights by pretraining the network to output the color decision for 20 epochs, which required the network to represent color information. After 20 epochs, we reverted to training on the CB task, where only the direction decision was reported. Since the learning rate was kept constant, the pretrained weights can be viewed as a different initialization in parameter space for the modified task.

Strikingly, we found that the resulting representations were not minimal for the $n = 2$ checkerboard case (Fig C.1a). This result also held for the CB task with $n = 10$ and $n = 20$ (Fig C.2b,c). While we observed some compression of usable color information through training, the asymptotic representations had significantly greater than zero color information. In Fig C.2b, we observed all layers had more usable color information than the direction information in the first layer. The network therefore solved the task using an alternative representation that was not minimal. We visualized the activations corresponding to the asymptotic non-minimal representations of Small FC in Fig C.1d. In the early epochs the red and green points converge (both crosses and dots) as a result of successful pretraining. However, when we trained the CB task starting at epoch 20, the representations changed. While the dot clusters for red and green checkerboards are overlapping, the cross clusters are not. This representation is not minimal as color information can be decoded above chance.

These results show that the initialization affects the asymptotic representation of neural networks. SGD, under particular initializations, may not lead to minimal representations of the task inputs. This suggests there is a trade-off between learning a minimal representation and simply reusing the existing representations present in the initial weights. Initial structure in the network representations from pretraining, such as the separation of the red and green crosses in the last layer representation, was maintained even when performing SGD to train a different task. Together, these results suggest that while SGD compresses representations towards minimality, it finds a solution that is functionally related to the initial representation. This may correspond to a optima in the neighborhood of the initialization.

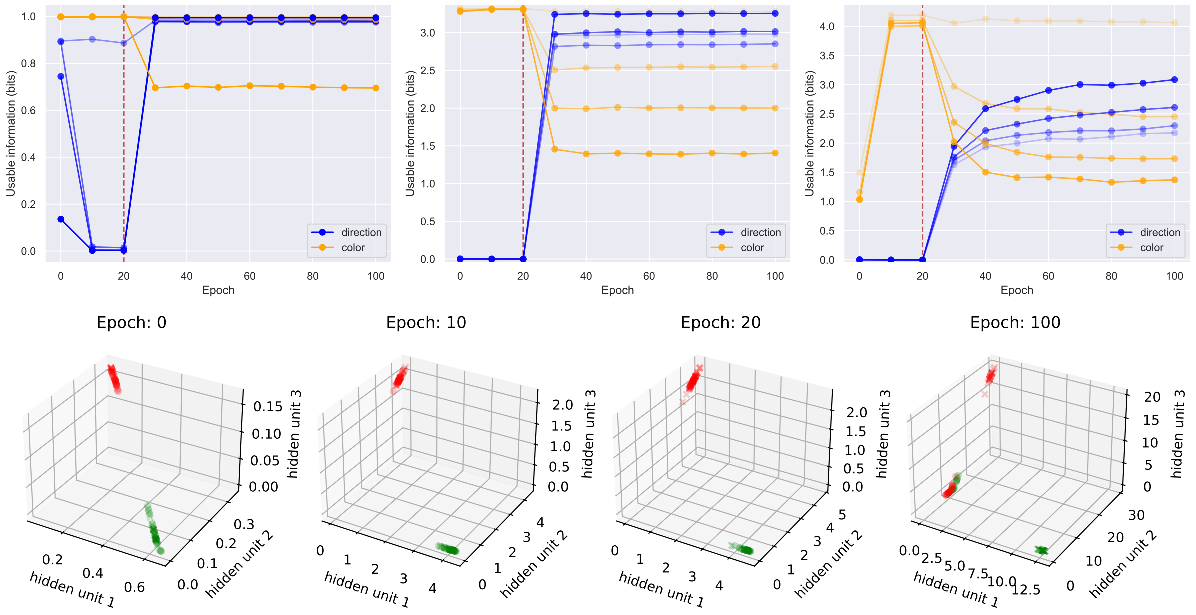

Figure C.1: Usable color and direction information in a network through training following pretraining the network to output color, not direction. Pretraining occurred for the first 20 epochs, indicated by the dashed red line. Subsequently, the network was trained to output direction, as in Fig 4.2. **(a)** Usable information for Small FC trained on the $N = 2$ CB task. Usable color information increased in training, and decreased when the loss function changed. However, the asymptotic representation is not minimal. **(b)** Medium FC trained on $N = 10$ CB task. Similarly, the network formed a representation of color during pretraining, but the asymptotic representation is not minimal. **(c)** Medium FC trained on $N = 20$ checkerboard task. **(d)** Visualization of the Small FC network in (a) showing that an optimal representation is not formed. The asymptotic representation in the last area has separate representations for red and green crosses. These should be overlapping in a minimal representation.

C.2.2 Relationship between pretraining, minimality, and generalization in the CB task

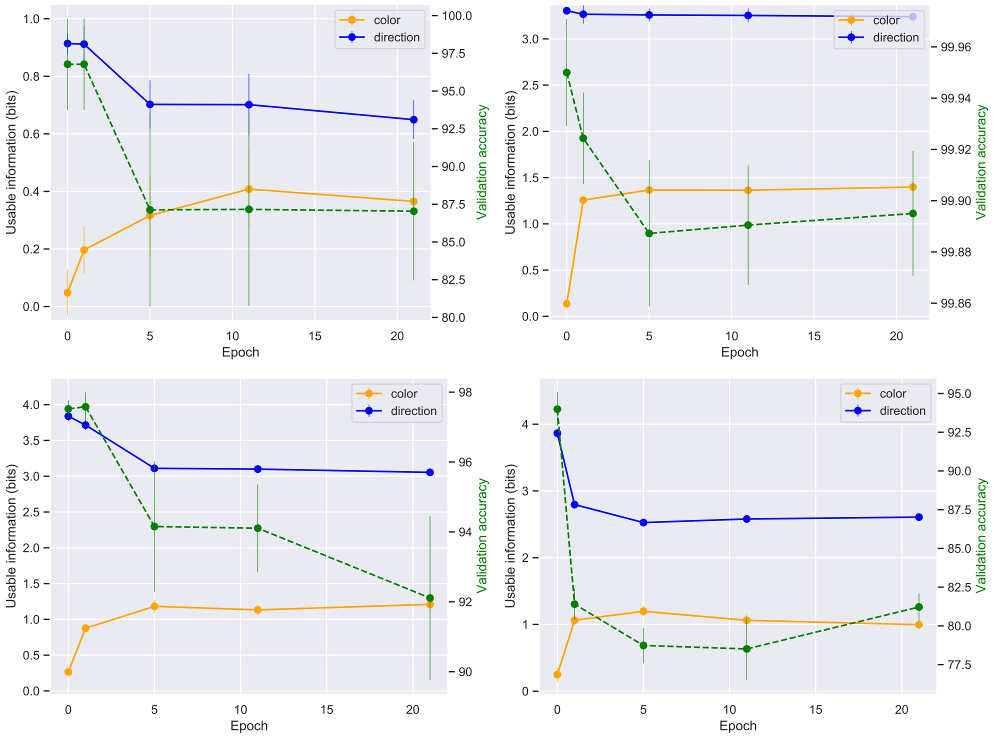

Figure C.2: **(a)** Final usable information and validation accuracy (green dashed line) as a function of pretraining epoch for the CB task ($n = 2$) averaged over 8 random initializations. **(b)** Final usable information and accuracy as a function of pretraining epoch for the CB task ($n = 10$) averaged over 8 random initializations. **(c)** Final usable information and accuracy as a function of pretraining epoch for the CB task ($n = 20$) averaged over 8 random initializations. **(d)** Final usable information and accuracy as a function of pretraining epoch for the CB task ($n = 25$) averaged over 8 random initializations. Error bars show the S.E.M.

Our results show that the minimality of network representations, and therefore solutions, depends on initialization. All trained networks (for n larger than 2), however, achieved zero training error. A natural question to ask is how does the pretraining affect the resulting representation and generalization performance?

To answer this, we varied the number of epochs that we pretrained the CB tasks of $n = 2$, $n = 10$, and $n = 20$ classes, and quantified the usable color and direction information, as well as the trained network's test accuracy to understand how the network generalizes (Fig. C.2).

We found that networks trained with longer pretraining had less minimal representations and worse generalization performance. This was true regardless of the number of classes, but the effect was more pronounced (in terms of absolute difference in accuracy) when the network did not solve the task perfectly without pretraining. We note that regardless of how long the networks were pretrained for, the networks were subsequently trained for the same number of epochs (80), with the same learning rate throughout training. One interpretation is that when using existing structure to solve the task, the network learned a suboptimal solution to solving the task, increasing the chance of overfitting. Another interpretation is that the pretraining changed the distribution of the weights, affecting the minimality and generalization.

C.2.3 Details of neural network for usable information in the CB Task

To estimate usable information, we computed the cross-entropy loss of a decoder $q(y|z)$ that predicts Y from Z. The decoder was a three-layer neural network, with 128, 64, and 32 units per layer, with Leaky-ReLU activations (slope = 0.2), batch-norm and dropout ($p = 0.7$). At each epoch, 1250 training samples were generated and supplied to the decoder, along with either the corresponding correct direction or color choice. We evaluated the cross-entropy loss on 3750 test samples to minimize overfitting. We trained the network for 100 epochs using a learning rate of 0.5 for 'Medium FC' and 0.05 for 'Small FC.'

C.2.4 Checkerboard Task description

Following the conventions of [83], we modeled the CB task (Fig 4.1a), inputting the checkerboard color and target configuration to a neural network that outputted the direction choice (Fig 4.1b). We minimized the cross-entropy loss of the network output and the ground truth output. We extended the checkerboard task to the n checkerboard task by increasing the number of checkerboards. Each target was 1 out of the n colors, with the targets forming an 'n-polygon'. The correct direction corresponds to the direction of the target having the same color of the checkerboard. We specified the color of each target using a one-hot encoding, and the color of the checkerboard as a one-hot encoding. Noise with mean 0 and standard deviation of 0.1 was added to the checkerboard inputs. The target and checkerboard color inputs were concatenated to form an input vector. The correct direction of the target was the output.

C.2.5 Details of CB experiments

The following are the hyper-parameters used in our experiments. We trained two different network architectures, 'Small FC': 5 layers, with $10 - 7 - 5 - 4 - 3$ units in each layer, 'Medium FC': $100 - 20 - 20 - 20$. We trained networks using SGD with a constant learning rate throughout training.

FC Small, $n = 2$:

- batch size: 32, learning rate: 0.05, number of data samples: 10000 (90% train, 10% validation)

Medium FC, $n = 10$**:**

- batch size: 64, learning rate: 0.5, number of data samples: 25000 (90% train, 10% validation)

Medium FC, $n = 20$**:**

- batch size: 128, learning rate: 0.5, number of data samples: 50000 (90% train, 10% validation)

Medium FC, $n = 25$**:**

- batch size: 128, learning rate: 0.5, number of data samples: 75000 (90% train, 10% validation)

C.2.6 Definition of relevant and irrelevant information in the CB Task

In the CB task, the color of the checkerboard and target configuration (inputs) are necessary to determine the correct direction to reach (output). While both a color and direction decision are made, after the direction is determined, the color decision no longer needs to be represented: the network can generate the correct output with only the direction representation. Formally, the output y is conditionally independent of the color representation, Z_c, given the direction representation Z_d (i.e., $y \perp\!\!\!\perp (Z_c, Z_t)|Z_d$, as illustrated by the graph in Fig 4.1b). Hence, given a representation of the direction choice, the color choice (and target configuration) no longer needs to be represented. We emphasize that, in general, the output is not independent of the color representation and target configuration representation Z_t, i.e., $y \not\perp\!\!\!\perp (Z_c, Z_t)$, hence information about the dominant color of the checkerboard is necessary to compute y. When this conditional independence holds, we call the conditionally independent variable "irrelevant." We therefore refer to the color choice as "irrelevant" and the direction choice as "relevant." We study how these components evolve together throughout training.

C.3 Additional results and details in the CIFAR-10 and CIFAR-100 task

C.3.1 CIFAR-10 and CIFAR-100 task description

We trained a ResNet-18 and an All-CNN architecture to output a superclass corresponding to the twenty coarse-grained classes in CIFAR-100 and, in CIFAR-10, to an arbitrary superclass corresponding to the even and odd classes. Accordingly, a minimal representation should only encode the superclass.

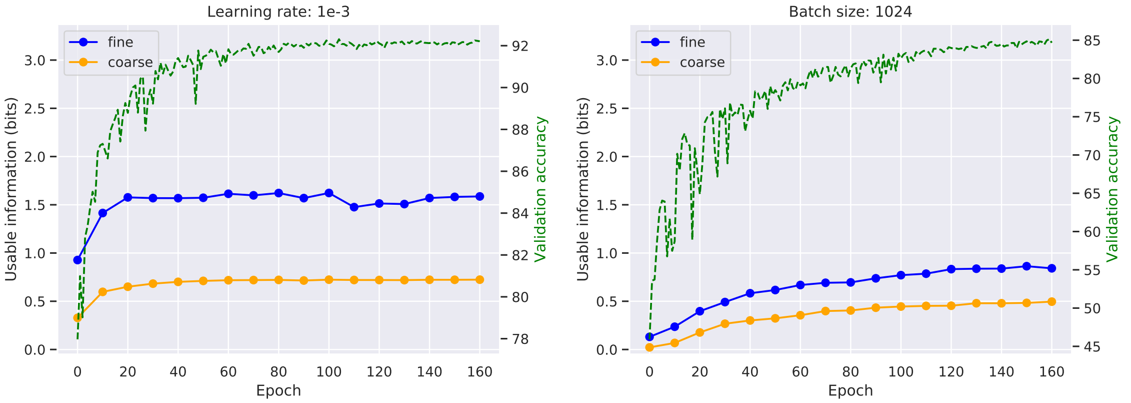

Figure C.3: **(Left)** Usable information for initial learning rate of 0.001 in CIFAR-10. The information about the fine labels does not decrease, and the validation accuracy only reaches 92%, in co ntrast to Fig 4.3 where the validation accuracy reached 96%. **(Right)** Usable information for batch size of 1024 in CIFAR-10.

C.3.2 Details of neural network for usable information

To estimate usable information, we computed the cross-entropy loss of a decoder $q(y|z)$ that predicts Y from Z. We used a two-layer neural network, with 200 and 100 with Leaky-ReLU activations (slope $= 0.2$), batch-norm and dropout ($p = 0.7$). At each epoch, 7500 samples were supplied to the decoder, along with either the corresponding correct direction or color choice. We evaluated the cross-entropy loss on 2500 test samples. We trained the network for 50 epochs using Adam with a learning rate of 0.01 and weight decay of 0.001.

C.3.3 Details of neural network training

In our experiments, unless otherwise stated, we trained a ResNet-18 [55] with an initial learning rate of 0.1 decaying smoothly with a factor of 0.97 at each epoch, batch size of 128, momentum of 0.9 and weight decay with coefficient 0.0005. For the All-CNN [129] we used a batch size of 128, initial learning rate of 0.05 decaying smoothly by a factor of 0.97 at each epoch, momentum of 0.9, and weight decay with coefficient 0.001. We used standard data augmentation with random translations up to 4 pixels and random horizontal flipping. These parameter configurations were taken directly from prior work [2].

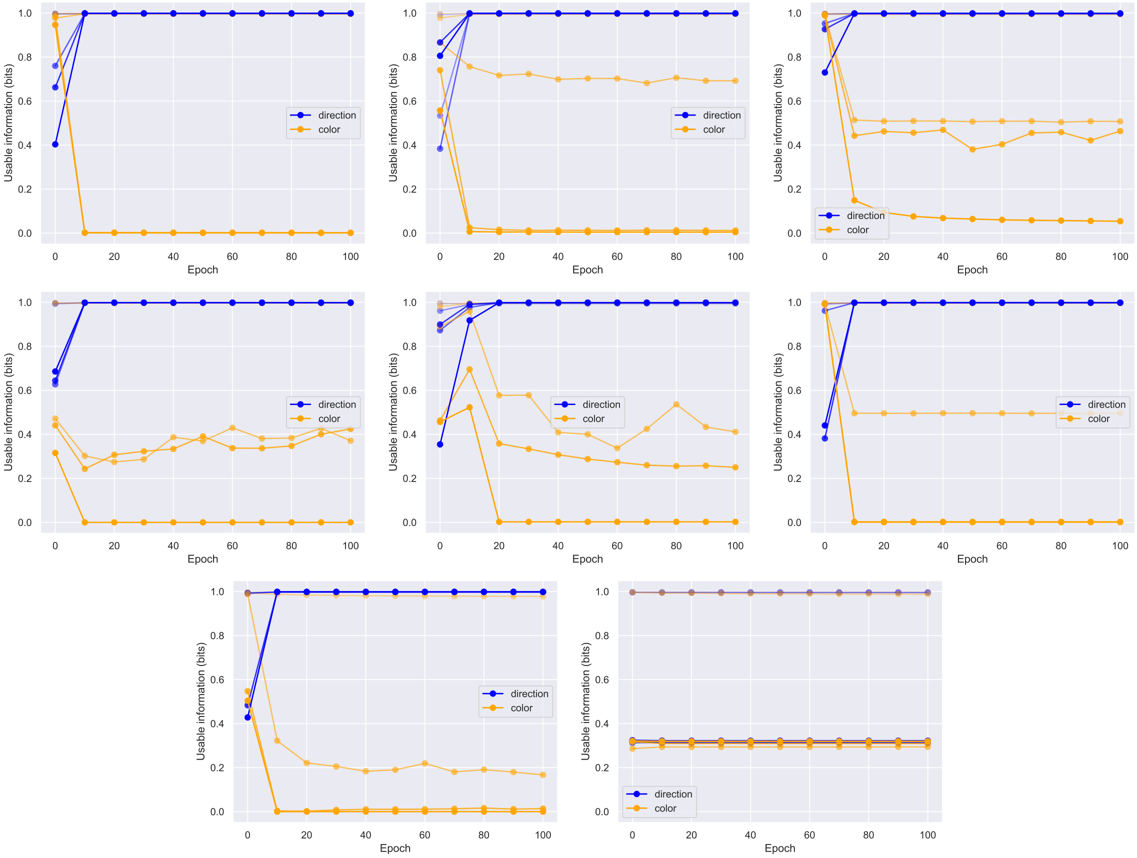

Figure C.4: Evolution of usable information for eight random initializations for the $n = 2$ CB task.

C.4 Additional plots

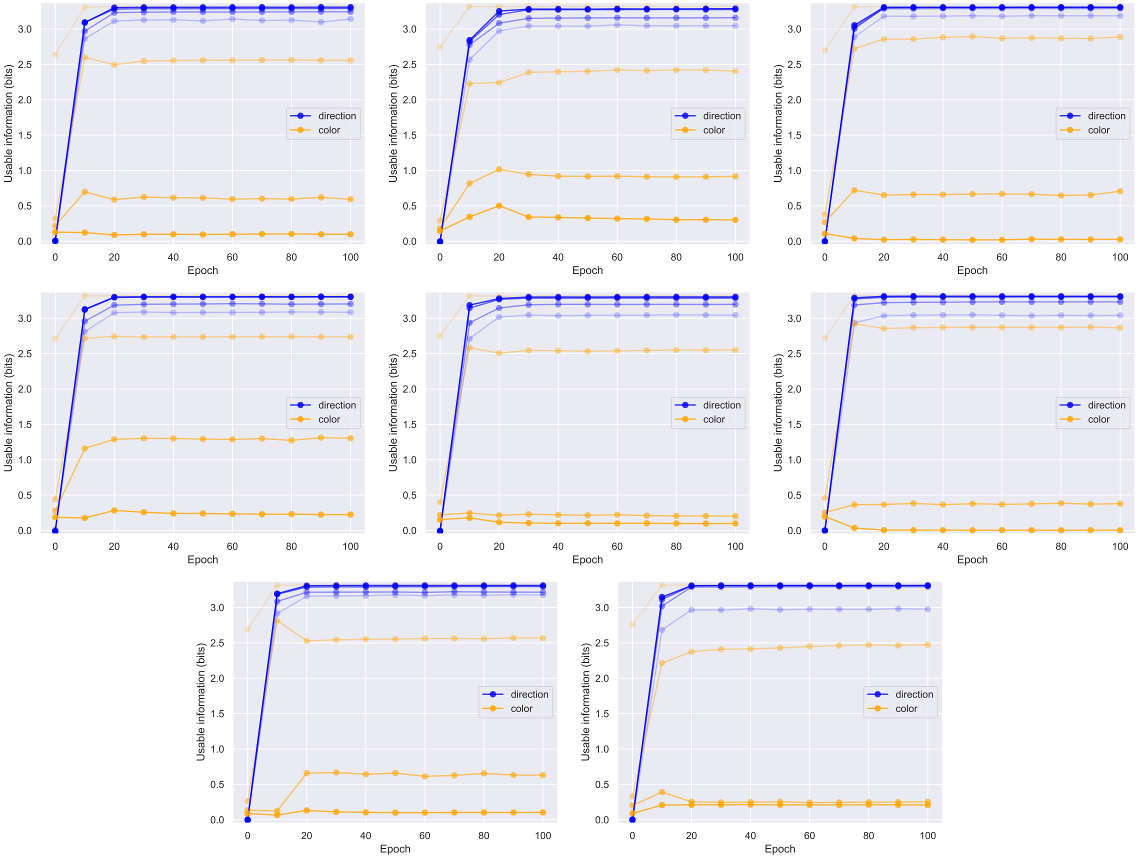

Figure C.5: Evolution of usable information for eight random initializations for the $n = 10$ CB task.

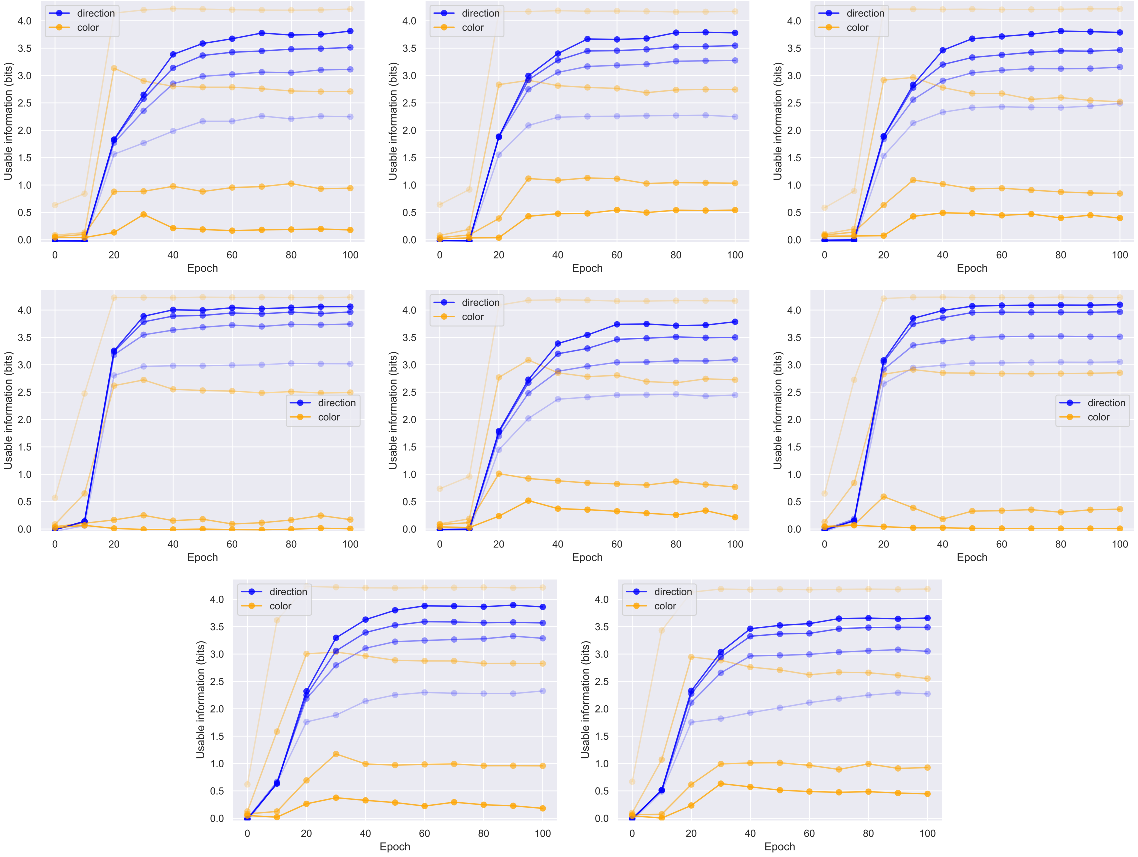

Figure C.6: Evolution of usable information for eight random initializations for the $n = 20$ CB task.

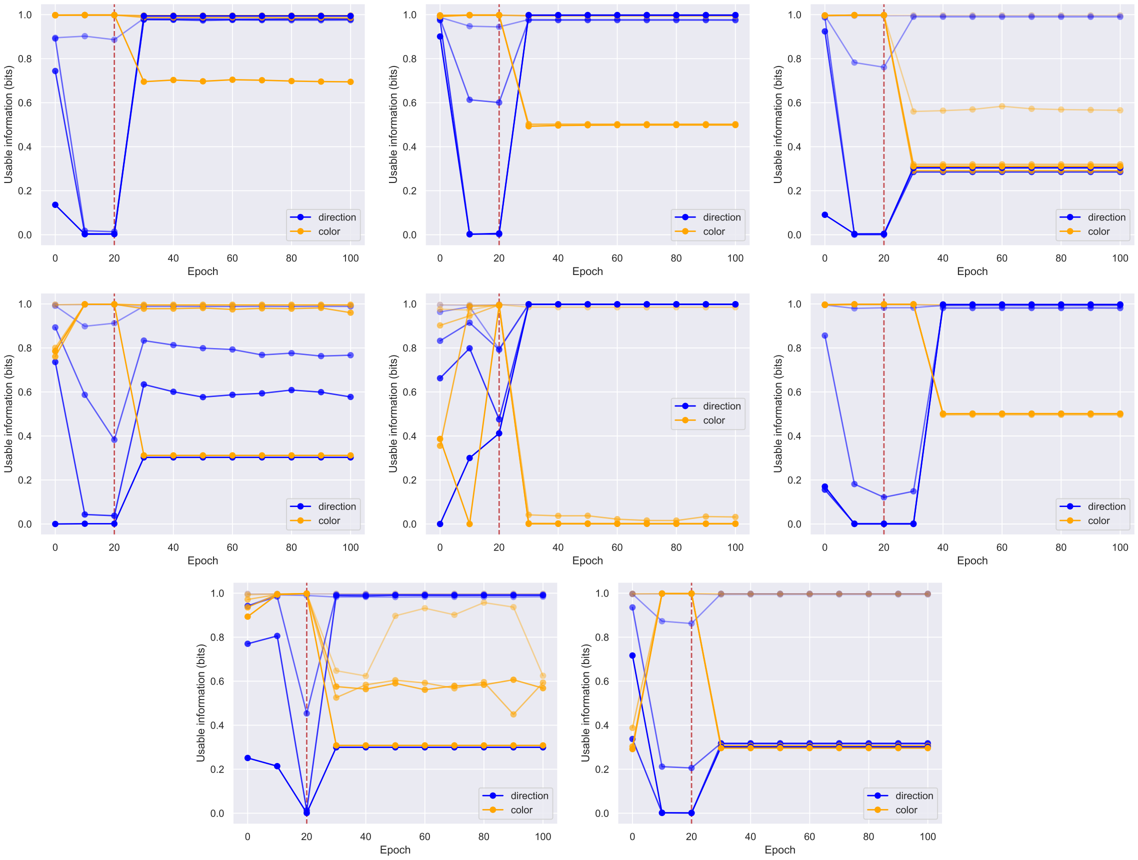

Figure C.7: Evolution of usable information for eight random initializations for the $n = 2$ CB task with 20 epochs of pretraining. If the the usable information was negative, indicating that the decoder overfit, we set the usable information to 0. Note that this occurred for a very small number of points.

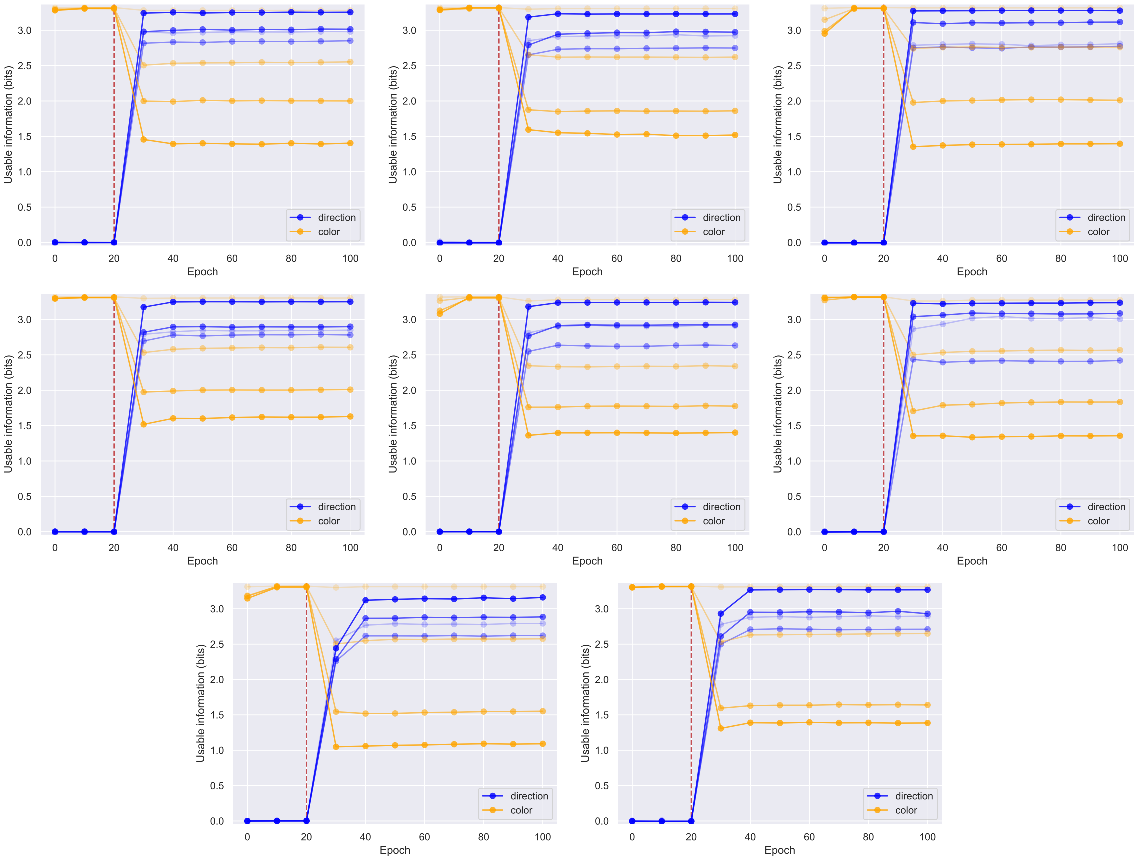

Figure C.8: Evolution of usable information for eight random initializations for the $n = 10$ CB task with 20 epochs of pretraining.

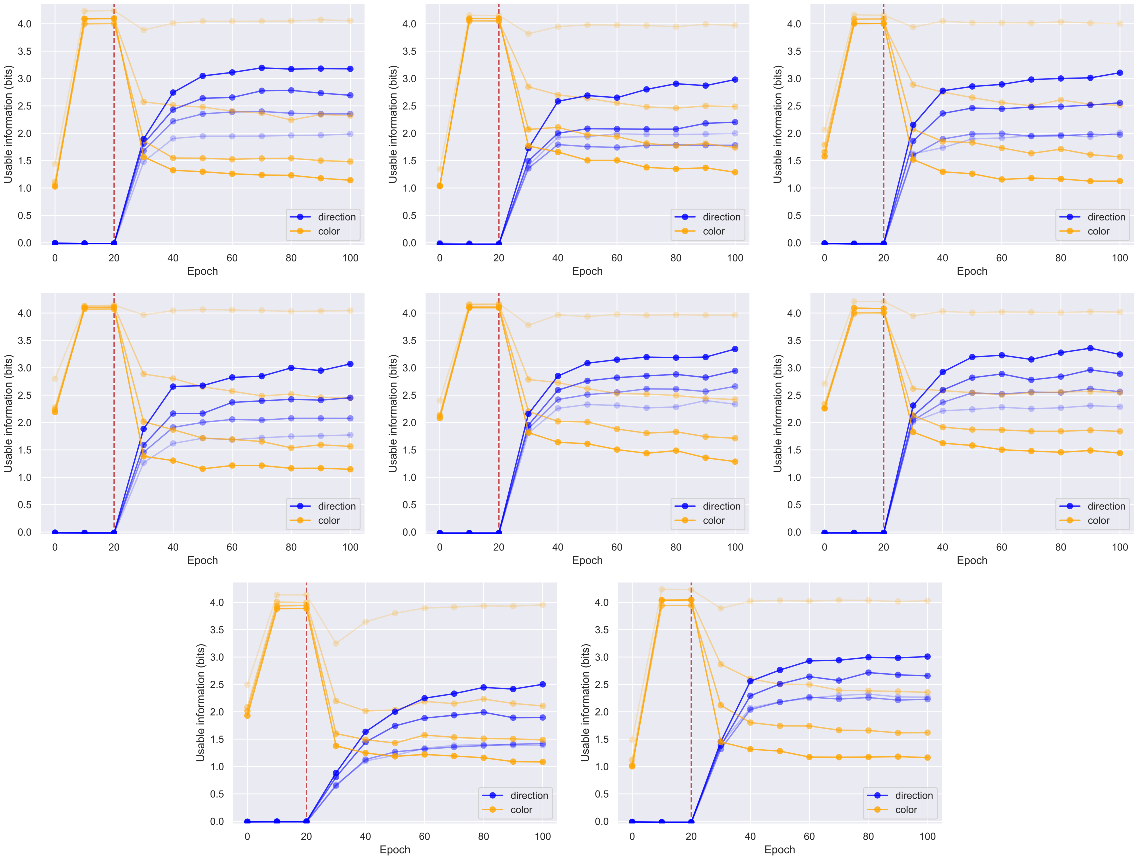

Figure C.9: Evolution of usable information for eight random initializations for the $n = 20$ CB task with 20 epochs of pretraining.

Appendix D

Supplementary Material for Chapter 5

D.1 Supplementary Material

D.1.1 Description of simulated RSV distributions

When evaluating the RSV on a synthetic distribution, we considered the following generative model that consists of a common component x_0 with additive noise:

$$\begin{gathered} x_a = x_0 + n_a, \quad x_b = x_0 + n_b, \\ z_i = w_i x_a + (1 - w_i) x_b, \\ w_i \sim \mathrm{Beta}(\alpha, \beta), \quad x_0 \sim \mathcal{N}(0,\ 1), \quad n_a \sim \mathcal{N}(0,\ 1), \quad n_b \sim \mathcal{N}(0,\ 1). \end{gathered} \tag{D.1}$$

Depending on the values of α and β, the Beta distribution that the weights w_i are drawn from will take different shapes, changing how units in the representation z vary with inputs x_a and x_b. We find that the distribution of RSVs in Fig. 5.3 reflect the full spectrum of these various distributions, where the resulting RSVs can vary from an approximately Gaussian distribution where units vary equally with both modalities, to polarized representations where units vary uniquely with one modality. For this synthetic simulation, we can derive a closed form expression for the RSV. In particular (and dropping the subscript i for clarity),

$$z = x_0 + wn_a + (1-w)n_b \tag{D.2}$$

and note that z will be distributed as a normal distribution. Then,

$$SV_i = Var(Z|X_a = x_a) \tag{D.3}$$

$$= \sigma_z^2(1-p^2) \tag{D.4}$$

$$= \sigma_z^2(1 - \frac{Cov(z, x_a)^2}{\sigma_z^2 \sigma_{x_a}^2}) \tag{D.5}$$

We know that

$$\sigma_z^2 = \sigma_{x_0}^2 + w^2\sigma_a^2 + (1-w)^2\sigma_b^2 \tag{D.6}$$

since x_0, n_a, and n_b are independent. Finally,

$$Cov(z, x_a) = \mathbb{E}[(Z - \mathbb{E}[Z])(X_a - \mathbb{E}[X_a]] \tag{D.7}$$

$$= \mathbb{E}[ZX_a] \tag{D.8}$$

$$= \mathbb{E}[(wX_a + (1-w)X_b)X_a] \tag{D.9}$$

$$= \mathbb{E}[(w(X_0 + N_a) + (1-w)(X_0 + N_b))(X_0 + N_a)] \tag{D.10}$$

$$= \mathbb{E}[(X_0 + wN_a + (1-w)N_b)(X_0 + N_a)] \tag{D.11}$$

$$= \mathbb{E}[X_0^2] + w\mathbb{E}[N_a^2] \tag{D.12}$$

$$= \sigma_{x_0}^2 + w\sigma_a^2 \tag{D.13}$$

We also know that

$$\sigma_{x_a}^2 = \sigma_{x_0}^2 + \sigma_a^2. \tag{D.14}$$

We can then solve for SV_i by plugging Eq 9, 16, 17 into Eq 8 and obtain:

$$
\begin{aligned}
SV_i &= \sigma_z^2(1 - \frac{Cov(z, x_a)^2}{\sigma_z^2 \sigma_{x_a}^2}) && \text{(D.15)} \\
&= (\sigma_{x_0}^2 + w^2\sigma_a^2 + (1-w)^2\sigma_b^2)(1 - \frac{\sigma_{x_0}^2 + w\sigma_a^2}{(\sigma_{x_0}^2 + \sigma_a^2)(\sigma_{x_0}^2 + w^2\sigma_a^2 + (1-w)^2\sigma_b^2)}) && \text{(D.16)}
\end{aligned}
$$

We assumed that the representation z_i for half of the units were sampled from above generative model, while the other half the representation z_i were sampled from the reverse convex combination of inputs, i.e, $z_i = w_i x_b + (1 - w_i)x_a$.

For simulations 2-4, we set $\beta = 20$ and varied α in $[1, 20, 30]$ respectively. We considered a representation on $N = 20000$ units. For the first simulation we only considered the half of units in the generative model above, with $\alpha = 1$ and $\beta = 10$.

D.1.2 Generalization of RSV to arbitrary number of sensors

We can naturally generalize the RSV to an arbitrary number n of sources. To do so, define:

$$SV_i(X_j, x_1, ..., x_{j-1}, x_{j+1}, ..., x_n) = Var(f(\mathbf{X})_i | X_1 = x_1, ..., X_{j-1} = x_{j-1}, X_{j+1} = x_{j+1}, ..., X_n = x_n),$$

and then collect the individual source variances into a vector $\mathbf{SV}_i$ of size n. Then normalized sensor variance would be

$$RSV_i = \text{softmax}(\mathbf{SV_i}),$$

which provides a normalized quantification (between 0 and 1) of how much an individual unit varies with each sensor modality j.

D.1.3 Description of deep linear network experiment

We considered the original input-output correlation (before dropping a sensor) to be

$$\mathbf{\Sigma}^{yx}_{pre} = \begin{bmatrix} 1 & 0 & 3 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 3 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{D.17}$$

Our perturbation involved dropping a sensor, in this case the third column, leading to

$$\mathbf{\Sigma}^{yx}_{post} = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{D.18}$$

Using the analytical equations for the learning dynamics given by [118] for the shallow and deep network, we investigated how learning the task (row 5) was affected (Fig. 5.2), finding that such a perturbation had a significant on the dynamics of sensor learning in the deep, but not shallow, network.

D.1.4 Description of architectures and training

Most of our experiments are based on the ResNet-18 architecture [55]. We modified the architecture to process multi-sensor input with what we call a SResNet-18. We separately process two initial pathways which we combine in an additive manner. In particular, the initial pathway followed the architecture of [55] directly up to (and including) `conv3_x` (See Table 1 of [55]). After combining the pathways, the remaining layers followed the ResNet-18 architecture directly.

To examine the effect of depth, we modified the All-CNN architecture [128], following [2]. In particular we processed each pathway with the following architecture:

$$\texttt{conv 96 - [conv } 96 \cdot 2^{i-1} \texttt{ - conv } 96 \cdot 2^{i} \texttt{ s2]}_{i=1}^{n} \texttt{ - conv } 96 \cdot 2^{n} \texttt{ - conv1 } 96 \cdot 2^{n} \texttt{ - conv1 10}$$

where s refers to the stride. We then merged the final representation from each pathway in an additive manner. We examined the setting when $n = 1, 2, 3$. We used a fixed learning rate of 0.001 in these experiments.

D.1.5 Description of Blurring Experiments (Fig. 5.4)

We attempted to simulate a cataract-like deficit by blurring the image to one pathway. We reduced the resolution of the image being passed to one pathway by first resizing the Cifar images to 8×8, and then resizing to its original size (32×32 pixels, decreasing the available information.

While training, we applied standard data augmentation on the uncorrupted pathway (random translation of up to 4 pixels, and random horizontal flipping. We then retained a width w of the leftmost and rightmost pixels from uncorrupted and corrupted pathway respectively, setting $w = 16$ unless otherwise stated. At inference time, no data augmentation was applied and the leftmost w pixels and rightmost w pixels was supplied to each pathway respectively. We used an initial learning rate of 0.075, decaying smoothly at each epoch with a scale factor of 0.97.

To quantify the information contained in the representation, we randomly masked out each pathway with $p = 0.1$ during training, and computed the usable information I_u contained in the representation Z abbout the task Y following [78,151] by computing $I_u(Z;Y) = H(Y) - L_{CE}$, with $H(Y)$ being known and equal to $\log_2 10$ since the distribution of targets is uniform, and L_{CE} being the cross-entropy loss on the test set. We reported the corresponding RSV plots, and network performance in Appendix Fig. D.1, which reveal similar performance trends and polarization of units, when pre-training with the random masking as in Fig. 5.4.

D.1.6 Description of Independent Pathways Experiment (Fig. 5.6)

We followed the same setup as above, but instead randomly permuted the images fed to the 'right' pathway across the batch, breaking the correlation between the views. We trained using an initial learning rate of 0.05, decaying smoothly with a scale factor of 0.97. When training with the deficit we randomly sampled the target from the different views with $p = 0.5$. We also modified the architecture to produce multiple classification outputs, corresponding to a classification based on both views, or each pathway respectively. This modification was helpful for interpreting the polarization plots. While training, the loss function was applied on the head that contained the proper input-target correspondence. After the deficit, and during inference, only the head corresponding to both views was used.

D.1.7 Description of Masking + Supervised MultiViT training

These experiments were based on the MultiMAE architechture [9], using their implementation and closely following their default settings. We adapted their implementation to process two separate RGB views coming from Kinetics-400 dataset [23]. We used a patch size of 16 in all experiments, and the AdamW optimizer [97]. All inputs were first resized to 224×224 pixels. Our learning rate followed the linear scaling rule [51].

For the masking sensitivity experiments in Fig. 5.8, we used a fixed delay of 1.33 seconds (4 frames) between frames, and trained with an initial base learning rate of 0.0001, with 40 epochs of warmup for the learning rate. We trained for 800 epochs, with a 200 epoch deficit of independent frames during the pre-training starting at different epochs during training. We used a masking ratio of 0.75. We pre-trained with a batch size of 256 per GPU on 8 GPUs. After the pre-training, we fine-tuned for 20 epochs with all the tokens and the corresponding action classification label. We fine-tuned on 8 GPUs with a batch size of 32. We fine-tuned with a learning rate of 0.0005, with 5 epochs of warmup.

For the supervised experiments, we trained our networks with an initial base learning rate of 0.01 for 120 epochs using all the tokens, with 20 epochs of warmup. We applied a

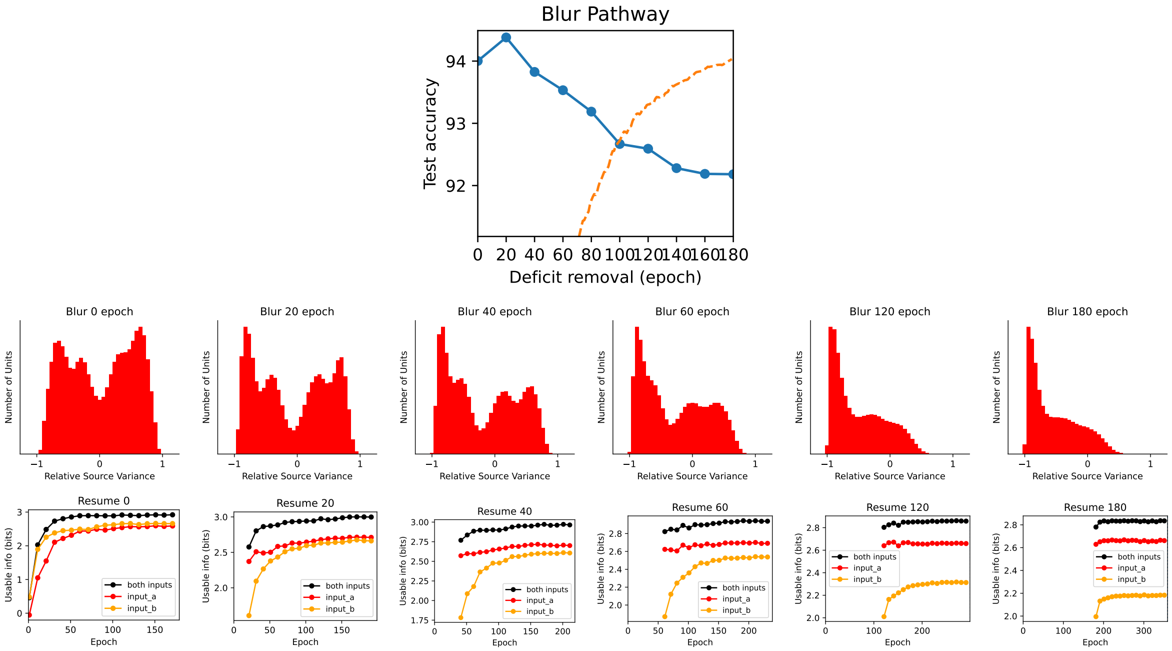

Figure D.1: Same blurring experiment as Fig. 5.5 with corresponding Relative Source Sensitivity, Fig. 5.4, but with the addition of random masking on each view with $p = 0.1$, allowing the decoding of the usable information [78] (bottom row). Note that the polarization (second row) is similar to Fig. 5.4, which is also reflected by the inability to decode the inhibited pathway, after exposure to a sufficiently long deficit (orange trace in bottom row).

temporary deficit of independent frames for 20 epochs, starting at various epochs during the training. We used in cutmix (1.0) and mixup (0.8) applied to each view) while training and we used a random baseline between frames. For the supervised experiments, we used a batch size of 64 per GPU.

In both the masking and supervised experiments in Fig. 5.8, we reported the difference of networks trained with a deficit starting at different epochs of training against a corresponding model trained without any deficit. In Fig. 5.7, we show example reconstructions from our Multi-View transformer pre-trained without a deficit for 800 epochs with a random baseline between frames.

D.2 Additional Plots

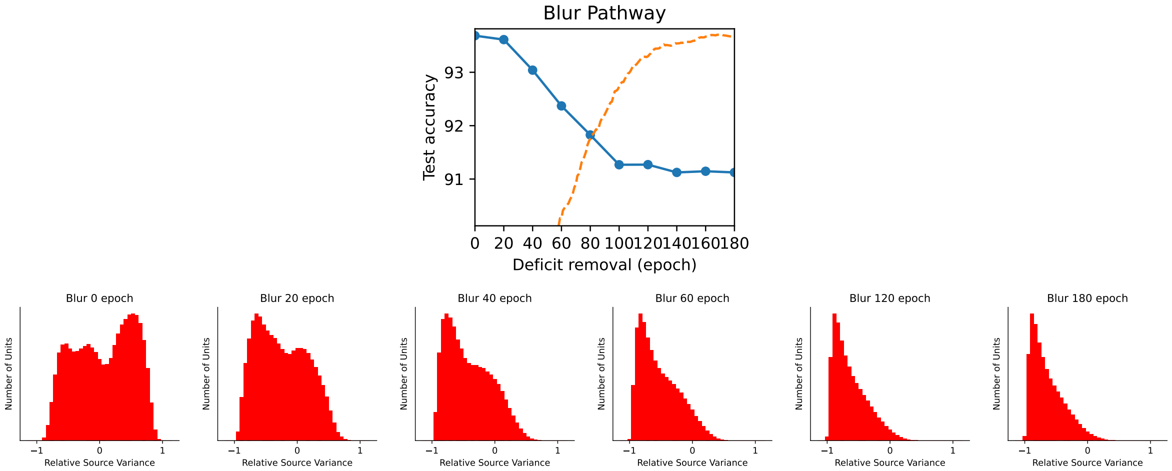

Figure D.2: Same blurring experiment as Fig. 5.5 with corresponding Relative Source Sensitivity, Fig. 5.4 for crop width of 16 (used in the main text) for easier comparison against different crop widths in Fig. D.3 and Fig. D.4.

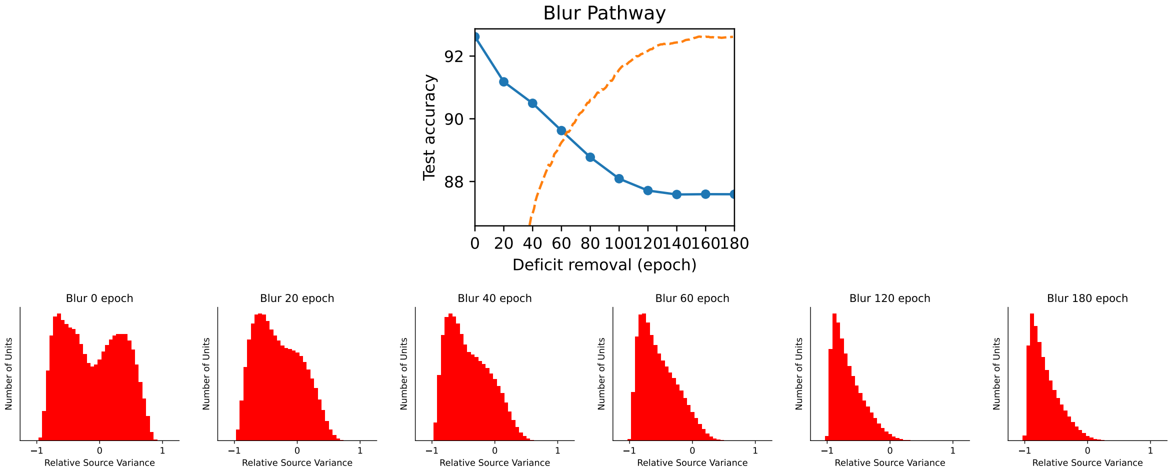

Figure D.3: Same blurring experiment as Fig. 5.5 with corresponding Relative Source Sensitivity, Fig. 5.4 for crop width of 14.

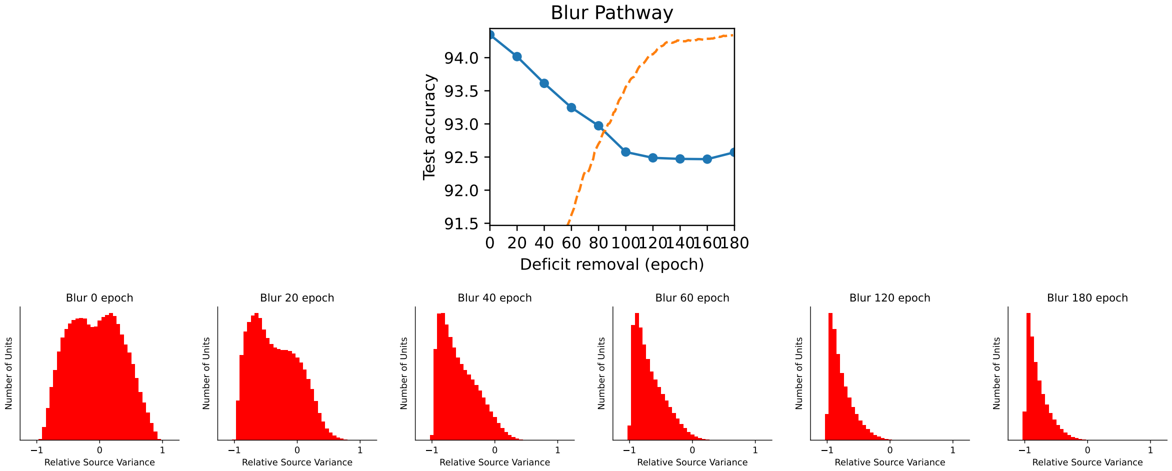

Figure D.4: Same blurring experiment as Fig. 5.5 with corresponding Relative Source Sensitivity, Fig. 5.4 for crop width of 18.

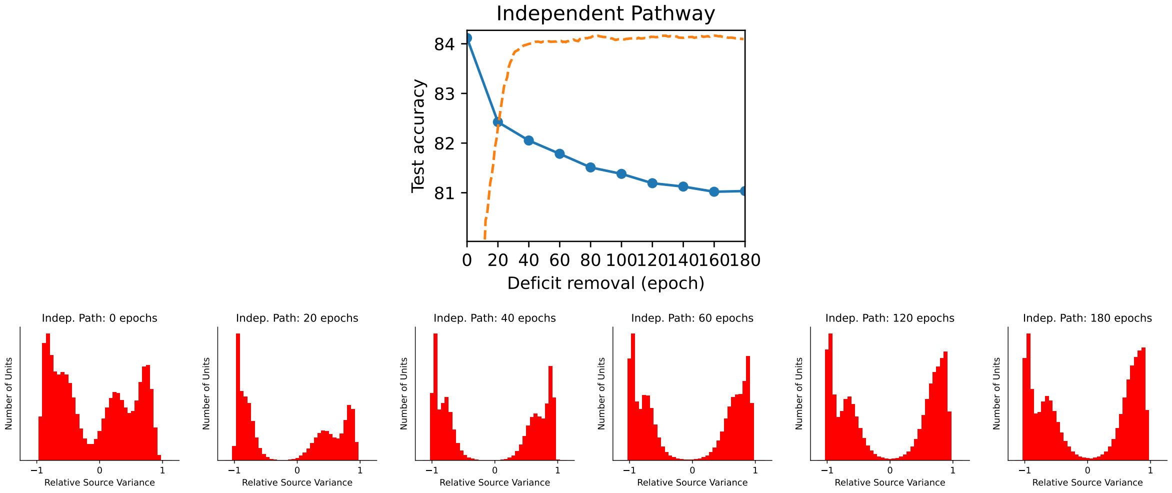

Figure D.5: Strabismus-Like Deficit for ablation of no weight decay (wd = 0), no data augmentation and initial lr = 0.05. We also observe a polarized representation. Note the performance is reduced in comparison to Fig. 5.6, due to the lack of data augmentation and weight decay.

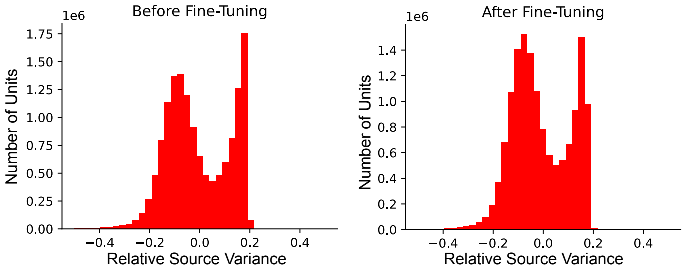

Figure D.6: **Relative Source Variance for Multi-View Transformer. (Left)** We show the distribution of RSV evaluated on the units at output of the encoder before fine-tuning, revealing a bimodal distribution. Here, training was performed without any deficits. **(Right)** During fine-tuning, the representations appear to adapt to become slightly more balanced, depending more evenly on each view, while retaining the initial bimodal structure learned during pre-training.

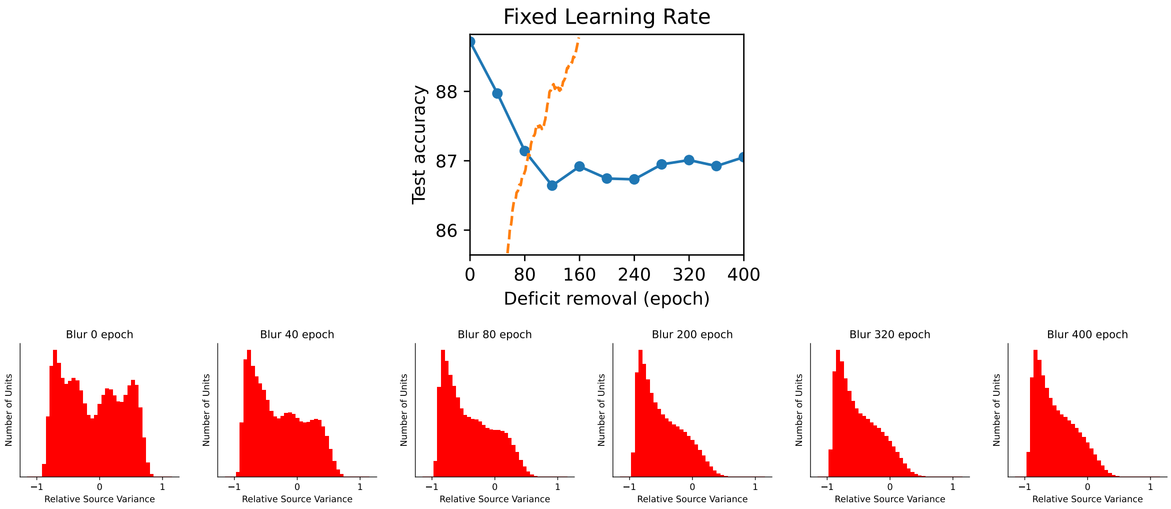

Figure D.7: Fixed learning rate of 0.0005 during training have similarly shaped critical periods to those in paper, and similar RSV distributions as a result of the deficit.

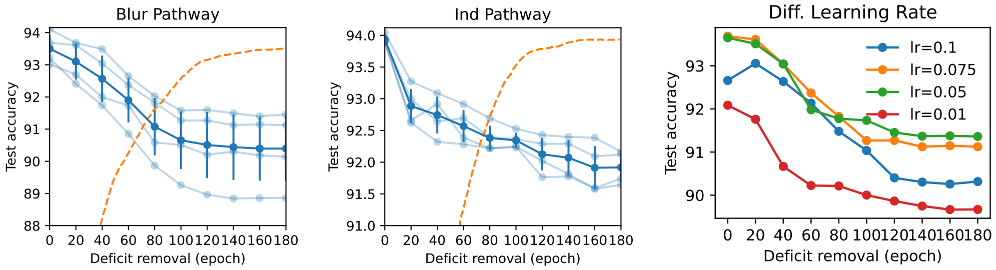

Figure D.8: Results of multiple runs (light blue), their average (dark blue), and std (bars) for **(Left)** blurring and **(Center)** dissociation deficit. **(Right)** Different initial learning rates (for blur deficit) have have similarly shaped critical periods to those in paper.

Appendix E

Supplementary Material for Chapter 6

E.1 Additional details

E.1.1 Partial Information Decomposition

Information theory provides a powerful framework for understanding the dependencies of random variables through the notion of mutual information [31]. However, information theory does not naturally describe how the information about a target Y is distributed among a set of sources $X_1, ...X_n$. For example, ideally, we could decompose the mutual information $I(X_1, X_2; Y)$ into a set of constituents describing how much information that X_1 contained about Y was also contained in X_2, how much information about Y was unique to X_1 (or X_2), as well as how much information about Y was only present when knowing both X_1 and X_2 together. These ideas were presented in [147] in the Partial Information Decomposition (PID).

Standard information-theoretic quantities $I(X_1; Y)$, $I(X_1; Y|X_2)$, and $I(X_1, X_2; Y)$ can

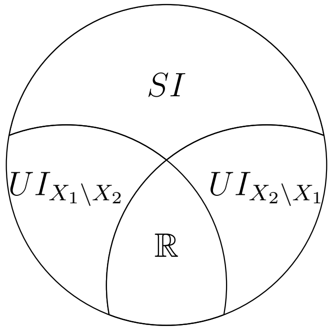

Figure E.1: Decomposition of the mutual information of sources X_1, X_2 and target Y into the synergistic information SI, the unique information UI of X_1 with respect to Y and X_2 with respect to Y, and the redundant information $\mathbb{R}$. Figure adapted from [12].

be formed with components of the decomposition:

$$I(X_1; Y) = UI(X_1; Y) + \mathbb{R} \tag{E.1}$$

$$I(X_2; Y|X_1) = UI(X_2; Y) + SI \tag{E.2}$$

$$I(X_1, X_2; Y) = UI(X_1; Y) + SI + UI(X_2; Y) + \mathbb{R} \tag{E.3}$$

Here UI represents the “unique” information and SI represents the “synergistic” information. Equation E.3 comes form the chain rule of mutual information, and by combining equation E.1 and equation E.2. These quantities are shown in the PID diagram shown in Figure E.1. Computing any of these quantities allows us to compute all of them [18]. In [12], they described an approach to compute the unique information, which was only feasible in low dimensions. In our paper, we instead focus on computing the “redundant” information.

E.1.2 Alternative notion of redundancy

Recently [87] proposed to quantify redundancy through the following optimization problem:

$$\mathbb{R}^K(X_1; \ldots; X_n \to Y) := \max_{s_{Q|Y}} I(Q; Y) \quad \text{s.t.} \quad \forall i \; s_{Q|Y} \preceq p_{X_i|Y} \tag{E.4}$$

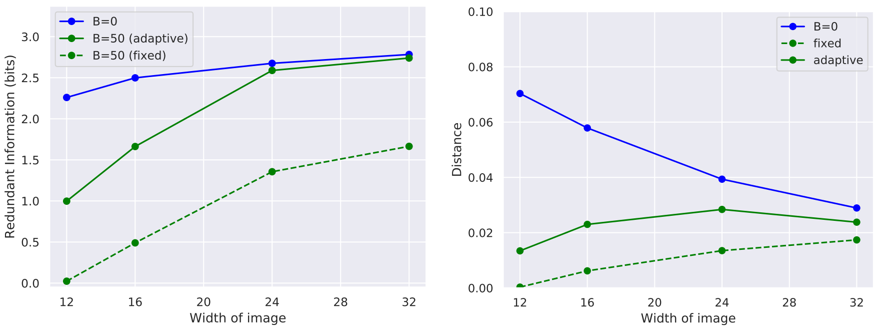

Figure E.2: **(Left)** If $B = 50$ for all epochs of training, the networks is stuck in a trivial solution in learning. Setting β adaptively leads to an improved solution. **(Right)** The final distance terms are comparable.

The notation $s_{Q|Y} \preceq p_{X_i|Y}$ indicates that there exists a channel $p_{Q|X_i}$ such that Equation E.5 holds for all q and y.

$$s(q|y) = \sum_{x_i} p(q|x_i)p(x_i|y). \tag{E.5}$$

In a sense, Equation E.5 indicates that Q is a "statistic" of X_i.

E.1.3 Setting value of β

When optimizing the equation in practice, it is more difficult to optimize initially using very large values of β, since the network could easily learn a trivial solution. We therefore adaptively set β depending on the epoch of training. In this manner, we find that the network settles in a redundant solution that performs well on the task, as opposed to a solution that is trivial. We smoothly increase β_i during training following the formula, so that the value of β at epoch i ($\gamma = 0.97$):

$$\beta_i = \beta(1 - \gamma^i). \tag{E.6}$$

We also perform an ablation study where we fix $\beta_i = \beta$, and find that the network settles at a more trivial solution (Fig E.2).

E.1.4 Training details for canonical examples

We trained a small fully-connected network with hidden layers of size $[25-15-10]$, using batch normalization and ReLU activations, with an initial learning rate of 0.01 decaying smoothly by 0.97 per epoch, for 30 epochs. We generated a dataset consisting of $10,000$ samples, of which 80% corresponded to training data, and the remaining 20% corresponded to test data. We trained with different values of β. $\beta=0$ corresponds to the the average usable information of $I_u(X_1;Y)$ and $I_u(X_2;Y)$. As β increases, the quantity $\mathbb{R}^{\mathcal{V}}$ more strongly reflects redundant information. RINE produces values close to the ground truth for these canonical examples. The tasks, with their corresponding inputs, outputs and associated probabilities are shown in Appendix 6.6. Our comparison is shown in Table 6.1. Note, that there is some randomness that occurs due to different initialization optimizing the neural networks, hence the values may differ slightly.

E.1.5 Comparison with cosine similarity

To highlight the difference between the redundant information that two inputs X_1 and X_2 have about a task Y and a direct similarity that could be applied on X_1 and X_2, we designed a synthetic task. In this task, there are 8 classes. We designed the inputs so that each input X_1 and X_2 would contain information about n classes, with the minimal overlap. For instance, if $n=4$, each input would contain information about 4 distinct classes, so there would be no redundant information. We swept the value of n ranging from 4 to 8 (Fig 6.5 (left), with increasing redundant information for increasing values of n). We optimized over a two-hidden-layer deterministic neural network with hidden layer dimensions of 25 and 15, using Adam with a learning rate of 0.01 for 50 epoch, with $\beta=50$. We added noisy inputs with each input coming from $\mathcal{N}(0,2^2)$ These inputs did not affect the value of redundant information, however adding noisy inputs decreases the cosine similarity (shown for the case of $n=8$), whereas the addition of non-task related common inputs increases the cosine similarity (shown for the case of $n=4$).

E.1.6 Training details for CIFAR-10

To compute the redundant information for CIFAR-10, we optimized over the weights in Equation 6.6 using ResNet-18's [55]. We trained the network for 40 epochs, with an initial learning rate of 0.075, decreasing smoothly by 0.97 per epoch, with weight decay of 0.005. We show example images that represent the inputs x_1 and x_2 in Fig E.3. We jointly train two networks that process inputs x_1 and x_2 respectively, constrained to have similar predictions through including $D(f_1, f_2)$ in the loss. To compute $D(f_1, f_2)$, we quantified the L_1 norm of the distance between the softmax scores of the predictions. We evaluated the cross-entropy loss on the test set.

E.1.7 Training details for Neural Decoding

Fixed Delay Center Out Task

In this task, there are 8 target locations. After a target is shown, the monkey makes a plan to reach towards the target. The monkey then reaches to the target after a go cue (Fig 6.3, left). Our dataset consisted of a population recording of spike trains from 97 neurons in the premotor cortex during trials that were 700ms long. Each trial comprises a 200ms baseline period (before the reach target turned on) and a 500ms preparatory (planning) period after the reach target turned on but before the monkey can initiate a reach. Both our training and testing dataset consisted of 91 reaches to each target.

Variable Delay Center Out Task

We analyzed data from another delayed-center-out task with 8 targets with a variable $400 - 800$ms delay period, during which the monkey could prepare to reach to the target, but was not allowed to initiate the reach until the go cue. In these datasets, there were significantly fewer total trials per session (220 total reaches across 8 targets) in comparison to the dataset with a fixed delay period. Data from two motor-related regions, the premotor and primary motor cortex, was recorded from 2 monkeys (J and R). There were 4 sessions associated with monkey J and 3 sessions associated with monkey R. We used 90% of the trials to train and 10% of the trials to test, and the plots reflect the redundant information on the test set.

E.1.8 Generalization to n sources

Our formulation naturally generalizes to n sources $X_1, ..., X_n$. In particular, Equation 6.9 can be generalized as:

$$L_{\cap}^{\mathcal{V}}(X_1; ...; X_N \to Y, \beta) := \min_{f_1,...,f_n \in \mathcal{V}} \frac{1}{n} \Big[\sum_{i=1}^{n} H_{f_i}(Y|X_i) \Big] + \beta D(f_1, ..., f_n). \tag{E.7}$$

We note that when computing the redundant information, we compute the loss without the distance term $D(f_1, ..., f_n)$. A naive extension of the distance term to n sources is computing the sum of all the pairwise distance terms. If the number of sources is large, however, it may be beneficial to consider efficient approximations of this distance term.

E.1.9 Details on canonical examples

	True	$\mathbb{R}^{\wedge}$	$\mathbb{R}^{\text{GH}}$	$\mathbb{R}^{\mathcal{V}}$ ($\beta = 0$)	$\mathbb{R}^{\mathcal{V}}$ ($\beta = 5$)	$\mathbb{R}^{\mathcal{V}}$ ($\beta = 15$)
UNQ [T6.2]	0	0	0	0.981	0.809	0.006
AND [T6.3]	[0, 0.311]	0	0	0.318	0.008	0.007
RDNXOR [T6.4]	1	1	1	0.981	0.983	0.977
IMPERFECTRDN [T6.5]	0.99	0	0.99	0.983	0.978	0.984

Table E.1: Comparison of redundancy measures on canonical examples for additional values of β than Table 6.1. Quantities are in bits. $\mathbb{R}^{\mathcal{V}}$ denotes our variational approximation, for different values of β. $\mathbb{R}^{\wedge}$ denotes the redundant information in [52] and $\mathbb{R}^{\text{GH}}$ corresponds to the redundant information in [53].

E.1.10 Example decomposition of images

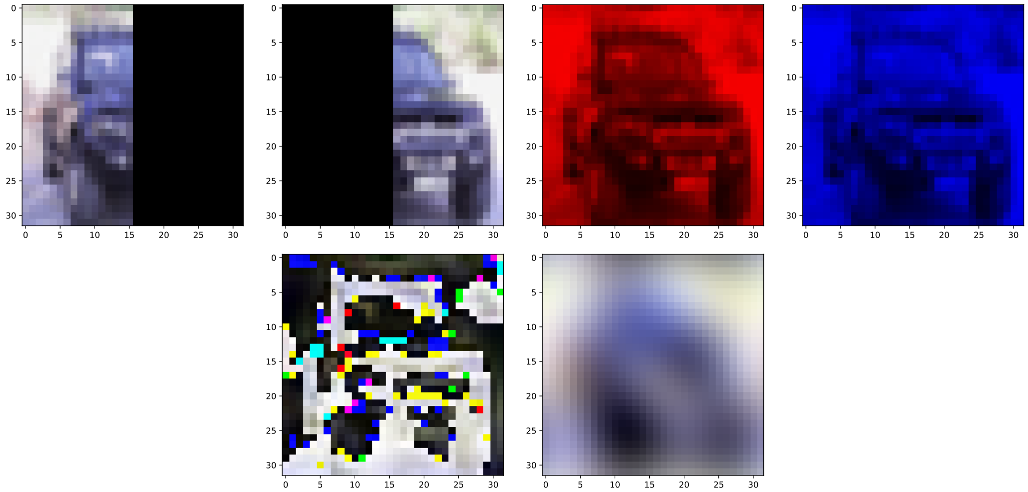

Figure E.3: Example decompositions of an image (car) from CIFAR-10. This is an example of x_1 and x_2 in our CIFAR experiments. **(Top left)**: different crops, **(top right)** colors of channels, and **(bottom)**: frequencies.

Appendix F

Supplementary Material for Chapter 7

F.1 Proofs

Theorem 2 (GK VAE recovers the common information). *Suppose our observations* $(\mathbf{x_1}, \mathbf{x_2})$ *have GK common information defined through the random variable* $\mathbf{z}_c$ *satisfying eq. 7.4-7.5 and that our parametric function class* $q(\mathbf{z}|\mathbf{x})$ *optimized over can express any function. Then, our optimization (with* $\beta_c = 0$ *and* $\beta_u < 1$*) is minimized by recovering latents* $\hat{\mathbf{z}} = (\hat{\mathbf{z}}_u^1, \hat{\mathbf{z}}_u^2, \hat{\mathbf{z}}_c)$ *where* $\hat{\mathbf{z}}_c$ *is the common random variable that maximizes the "stochastic" GK common information in eq. 7.4-7.5, while* $\hat{\mathbf{z}}_u^i$ *is the unique information of the i-th view, which maximizes* $I(\mathbf{x}_i; \mathbf{z}_u^i, \hat{\mathbf{z}}_c)$*.*

Proof. Let's consider the hard-constrained problem with an infinitely expressive function class (i.e. so that the cross-entropy loss corresponds to the conditional entropy). Our VAE objective corresponds to

$$\begin{aligned} \min \quad & H(\mathbf{x}|\mathbf{z}_u, \mathbf{z}_c) + \beta_u I(\mathbf{z}_u, \mathbf{x}) + \beta_c I(\mathbf{z}_c, \mathbf{x}) + \\ & H(\mathbf{x}'|\mathbf{z}_u', \mathbf{z}_c) + \beta_u I(\mathbf{z}_u', \mathbf{x}) + \beta_c I(\mathbf{z}_c, \mathbf{x}) \\ \text{s.t.} \quad & D(q_{\phi_1}, q_{\phi_2}) = 0 \end{aligned}$$

We consider a sequential optimization of finding $\hat{\mathbf{z}}_c$ and $\hat{\mathbf{z}}_u^i$, and then show that this solution minimizes the joint objective above. We first consider the hard-constrained version of eq. (7.12).

$$\begin{aligned}\min \quad & H(\mathbf{x}_1|\mathbf{z}_c) + \beta_c I(\mathbf{z}_c;\mathbf{x}_1) + \\ & H(\mathbf{x}_2|\mathbf{z}_c) + \beta_c I(\mathbf{z}_c;\mathbf{x}_2) \\ \text{s.t.} \quad & D(q_{\phi_1}, q_{\phi_2}) = 0\end{aligned}$$

Note that $H(\mathbf{x}_i) = H(\mathbf{x}_i|\mathbf{z}_c) + I(\mathbf{z}_c;\mathbf{x}_i)$. We can rewrite the loss as:

$$\begin{aligned}L &= H(\mathbf{x}_1|\hat{\mathbf{z}}_c) + H(\mathbf{x}_2|\hat{\mathbf{z}}_c) + \beta_c(I(\hat{\mathbf{z}}_c, \mathbf{x}_1) + I(\hat{\mathbf{z}}_c, \mathbf{x}_2)) \\ &= H(\mathbf{x}_1) + H(\mathbf{x}_2) + (\beta_c - 1)(I(\hat{\mathbf{z}}_c, \mathbf{x}_1) + I(\hat{\mathbf{z}}_c, \mathbf{x}_2))\end{aligned}$$

This tells us that the optimal $\mathbf{z}_c$ maximizes $I(\hat{\mathbf{z}}_c, x_1) + I(\hat{\mathbf{z}}_c, x_2)$. This is exactly the definition that we give of "stochastic" GK common information. Note we have previously shown $I(\mathbf{z};\mathbf{x}_1) = I(\mathbf{z};\mathbf{x}_2)$. Given $\hat{\mathbf{z}}_c$ found above, the remaining objective (eq. (7.13)) becomes:

$$\begin{aligned}\min \quad & H(\mathbf{x}_1|\mathbf{z}_u^1, \hat{\mathbf{z}}_c) + \beta_u I(\mathbf{z}_u^1;\mathbf{x}_1) + \\ & H(\mathbf{x}_2|\mathbf{z}_u^2, \hat{\mathbf{z}}_c) + \beta_u I(\mathbf{z}_u^2;\mathbf{x}_2).\end{aligned}$$

For $\beta_u < 1$, the objective maximizes $I(\mathbf{x}_1;\mathbf{z}_u^1, \hat{\mathbf{z}}_c)$, which was the definition of the unique information. (For $\beta_u > 1$, this corresponds to a β-VAE, and will have the corresponding trade-off between rate and reconstruction [3, 6, 59].)

Finally, suppose $\tilde{\mathbf{z}}_c$ did **not** contain all the common information as $\hat{\mathbf{z}}_c$; i.e. $I(\tilde{\mathbf{z}}_c;\mathbf{x}_i) < I(\hat{\mathbf{z}}_c;\mathbf{x}_i)$. Write the final equation as a maximization by noting that

$$H(\mathbf{x}_i|\mathbf{z}_u^i, \hat{\mathbf{z}}_c) = -I(\mathbf{x}_i;\mathbf{z}_u^i, \hat{\mathbf{z}}_c) + H(\mathbf{x}_i)$$

Then the final optimization (for any i) is equivalent to

$$\max \quad I(\mathbf{x}_i; \mathbf{z}_u^i, \hat{\mathbf{z}}_c) - \beta_u I(\mathbf{z}_u^i; \mathbf{x}_i) - H(\mathbf{x}_i) = \max \quad I(\mathbf{x}_i; \hat{\mathbf{z}}_c) + I(\mathbf{x}_i; \mathbf{z}_u^i | \hat{\mathbf{z}}_c) - \beta_u I(\mathbf{z}_u^i; \mathbf{x}_i) - H(\mathbf{x}_i) \tag{F.1}$$

$$> \max \quad I(\mathbf{x}_i; \tilde{\mathbf{z}}_c) + I(\mathbf{x}_i; \mathbf{z}_u^i | \tilde{\mathbf{z}}_c) - \beta_u I(\mathbf{z}_u^i; \mathbf{x}_i) - H(\mathbf{x}_i) \tag{F.2}$$

$$= \max \quad I(\mathbf{x}_i; \mathbf{z}_u^i, \tilde{\mathbf{z}}_c) - \beta_u I(\mathbf{z}_u^i; \mathbf{x}_i) - H(\mathbf{x}_i) \tag{F.3}$$

For any $\mathbf{z}_u^i$, Eq. F.1 is maximized with $\hat{\mathbf{z}}_c$ that encodes all the common information. Thus the GK VAE optimization is minimized with $\hat{\mathbf{z}} = (\hat{\mathbf{z}}_u^1, \hat{\mathbf{z}}_u^2, \hat{\mathbf{z}}_c)$. □

Proposition 4. *([149], Ex. 1): Define*

$$\mathbf{z}_1 = (\mathbf{z}_c, \mathbf{z}_u^1), \quad \mathbf{z}_2 = (\mathbf{z}_c, \mathbf{z}_u^2)$$

where $\mathbf{z}_c, \mathbf{z}_u^1$, and $\mathbf{z}_u^2$ are mutually independent. Then for any invertible transformation t_i the random variable $\mathbf{z}^*$ that satisfies

$$arg \max_{\hat{\mathbf{z}}} C_{GK}(t_1(\mathbf{z}_1), t_2(\mathbf{z}_2))$$

is $\mathbf{z}_c$.

Proof. Note that if t is the identity transformation $t(z) = z$, then

$$\arg\max_{\hat{\mathbf{z}}} C_{GK}(\mathbf{z}_1, \mathbf{z}_2)$$

is $\mathbf{z}_c$. As an aside, in this case $I(\mathbf{z}_1; \mathbf{z}_2) = C_{GK}(\mathbf{z}_1, \mathbf{z}_2)$

If t is an invertible transformations, suppose that f_1 and f_2 are the functions satisfying $\hat{Z} = f_1(Z_1) = f_2(Z_2)$ corresponding to $C_{GK}(\mathbf{z}_1, \mathbf{z}_2)$. Then the functions corresponding to $C_{GK}(t_1(\mathbf{z}_1), t_2(\mathbf{z}_2))$ will be $\hat{Z} = f_1 \circ t_1^{-1}(t_1(Z_1)) = f_2 \circ t_2^{-1}(t_2(Z_2))$ and the random variable $\hat{\mathbf{z}}$

is equivalent. □

F.2 Experimental Details

We trained networks with Adam with a learning rate of 0.001, unless otherwise stated. When the number of ground truth latent factors is known, we set the number of latents equal to the number of ground truth factors. To improve optimization, we use the idea of free bits [76] and we set $\lambda_{free-bits} = 0.1$. This was easier than using β scheduling, since it only involved one parameter $\lambda_{free-bits}$. We set β_u to be 10 and β_c to be 0.1. We trained networks for 70 epochs, except for the Mnist experiments, where we trained for 50 epochs. We used a batch size of 128 and we set $\lambda_c = 0.1$. For all our experiments we used the same encoders and decoders as [21], which has been also used in recent work [95]. Our architecture is schematized in Fig. 7.1. Note that we optimized encoders and decoders separately between the views (i.e weights were not shared).

To ensure that the latents are shared to both encoders, during training we randomly sample $\mathbf{z}$ from either encoder $q_{\phi_i}(\mathbf{z}_c|\mathbf{x}_i)$ with $p = 0.5$. We opted to randomly sample the latents from each encoder, as opposed to performing averaging, to ensure that the latent will always be a function of an individual view $\mathbf{x}_i$. This is in addition to the soft constraint governed by λ_c in the loss.

To quantify the information contained in the representation, we calculate the usable information (in bits), which is a lower bound to the information contained in the representation [78, 151]. To train our decoder, we used the `GradientBoostingClassifier` from `sklearn` with default parameters. We trained on 8000 samples and tested on 2000. We evaluated the information on a held-out test set, and hence the negative values correspond to overfitting on the training set. In Table 7.1, the numbers in parentheses correspond to the number of ground truth factors. We used the same setup for the *rotated Mnist* experiments (Fig. 7.3). For the rotation angle, we discretized the angle of rotation ($-45°$, $45°$) into 10 bins of equal size, and predicted the discrete bin. We predicted the rotation angle applied to the first view. When comparing with a constrastive learning approach (Fig. 7.3, right), we used the same encoder backbone as our GK-VAE[1]. We pre-trained with a batch size of 256 for 60 epochs with a learning rate of 0.001, and then trained the linear classifier for 10 epochs with an initial learning rate of 0.03. We used a latent dimension of 20.

F.2.1 DCI Plots and Disentanglement Score:

Let d be the dimension of the representation $\mathbf{z}$ and let $\mathbf{t}$ be the true generating factors. The idea is to train a regressor $f_j(\mathbf{z}) : \mathbb{R}^d \to \mathbb{R}$ to predict the ground truth factors t_j for each j from the representation $\mathbf{z}$. This results in a matrix of coefficients that describe the importance of each component of the representation for predicting each ground truth factors. This matrix R is the *importance matrix* that we visualize in the paper, where R_{ij} reflects the relative importance of of z_i for predicting t_j. We used the `GradientBoostingClassifier` from `sklearn` with default parameters, similar to [95] to predict the ground truth factors.

Following [37] we compute the DCI disentanglement score as $\sum_i \rho_i(1 - H(P_i))$, where $P_{ij} = R_{ij}/\sum_j R_{ij}$, and $\rho_i = \sum_j R_{ij}/\sum_{ij} R_{ij}$. In other words, P represents a normalized importance matrix, and ρ_i scales the contribution of the i^{th} row to the overall disentanglement score. The DCI disentanglement score is highest when each latent variable only encodes one ground-truth factor (and thus depends on the rows of the DCI plots).

F.3 Other Related Work

A similar formulation has been used for multi-view learning [142], with two separate autoencoders, with an constraint that each latent representation be similar, with the similarity measured by the canonical correlation of the latent representations. They did not motivate it from an information-theoretic perspective; and rather empirically found that such an

[1] We used the code from: https://github.com/HobbitLong/CMC

optimization lead to good representations in the multi-view setting.

Also related to our work is [113]. [113] defined the approximate Gács-Körner information in the following manner:

$$\begin{aligned} \max_{Z} \quad & I(X_1; Z) \\ \text{s.t.} \quad & H(Z|X_2) < \delta \\ & Z \leftrightarrow X_1 \leftrightarrow X_2 \end{aligned} \tag{F.4}$$

By showing that they could perform the optimization over deterministic functions f such that $Z = f(X_1)$, they formed a Lagrangian corresponding to:

$$\max_{f} \quad H(f(X_1)) - \lambda H(f(X_1)|X_2) \tag{F.5}$$

They noted that the above optimization is difficult to perform and that future work should look into avenues for computing this quantity; indeed it looks difficult to learn the function f from the above optimization problem. They also suggested that this approximate form of the Gács-Körner common information had potential applications in terms of compression, since the (approximate) common information only needs to be represented once.

F.3.1 Learning Disentangled Representations with VAEs

[59] To better understand why the β-VAE leads to more disentangled representations, it is helpful to decompose the second term in the following β-VAE loss

$$\mathcal{L}_{\beta-VAE} = \mathbb{E}_{p(\mathbf{x})}[\ \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[-\log p_\theta(\mathbf{x}|\mathbf{z})] + \beta KL(q_\phi(\mathbf{z}|\mathbf{x}) \ || \ p(\mathbf{z}))]. \tag{F.6}$$

The second term can be decomposed (for example [4, 25])

$$KL(q_\phi(\mathbf{z}|\mathbf{x}) \ || \ p(\mathbf{z})) = I_q(X; Z) + KL[q(\mathbf{z}) \ || \ \Pi_j q(z_j)] + \Sigma_j KL[q(z_j) \ || \ p(z_j)] \tag{F.7}$$

The first term is the mutual information (with respect to the encoder q_ϕ), i.e. $KL(q_\phi(\mathbf{z}|\mathbf{x}) \,||\, q_\phi(\mathbf{z}))$, the second term is the total correlation [145], and the third term is a dimension-wise KL divergence. [25] found that that the important term to minimize was the total correlation term, and that this was the key factor leading to disentanglement in the β-VAE.

F.3.2 Relationship to redundant information in the Partial Information Decomposition

Our approach also relates to approaches that aim understand how the information that a set of sources contain about a target variable is distributed among the sources. In particular, [147] proposed the Partial Information Decomposition (PID), which decomposes the information that two sources X_1 and X_2 contain about a target variable Y into a the components that are *unique*, *redundant*, and *complementary*. A central quantity in this decomposition, the *redundant information*, reflects the shared information about a *target* variable. The Gacs-Korner common information is equivalent to existing definitions redundant information if the target is reconstructing the sources (i.e, $Y = (X_1, X_2)$) [87]. We note that computing the redundant information from high dimensional samples has been challenging. Recently [81] proposed an approach that could be applied on high dimensional *sources* but where the *target* was low dimensional. Here, our approximation of the common information reflects a further step which can be applied on high dimensional samples (and targets).

F.4 Limitations of our approach

To validate our approach, we focused on the simpler setting where we have paired data, however, we could extend our formulation to find common information between $n > 2$ sources, as well as finding common information between subsets of sources. While our approach can be naturally extended to find the common information between n sources, future work could investigate a scalable approach to identify common and unique information between arbitrary subsets of the sources. Additionally, to validate our approach empirically, we focused on using a convolutional encoder on relatively small images and video frames, but our formulation is general and the encoder could be interchanged depending on the complexity and inductive biases of the task and data.

F.5 Additional Experiments

In Table F.1, we compute the information encoded in the common and unique latent components for the *common-dsprites* experiment described in the main text, with corresponding DCI matrix and latent traversals in Fig. 7.2. We also report additional runs for the *common-dsprites* and *common-3dshapes* experiments in Fig. F.1 and Fig. F.2 respectively. These additional runs are consistent with what was reported in the text, separating the common and unique factors.

In addition to the experiments described in the main text, we report variants of *common-dsprites* and *common-3dshapes*. In particular, we change the set of ground-truth common and latent factors.

For the *common-3dshapes*, we specified that the viewpoint was the unique latent variable $\mathbf{z}_u$, whereas the other latent variables (backgroud color, floor color, object color, shape, size) were common to both views. We show the DCI matrix and the traversals in Fig F.3. For the *common-dsprites* variant, we set the unique components to be the size, scale, and orientation, and the common latent factors to be the x and y position. We show the DCI matrix and latent traversals in Fig. F.4. The common and unique latent variables from our optimization separated these ground truth factors.

Table F.1: Usable information approximation for the information contained in the common and unique latents of a *dsprite* experiment. Note that the KL components are not ordered but are close to the “common” and ‘unique’ usable information. Unique factors: position; Common Factors: shape, scale, angle. Units in bits.

	Shape (3)	Scale (6)	Angle (40)	X-Pos (32)	Y-Pos(32)	KL Total
Common	1.54	2.45	2.88	-0.33	-0.29	9.57
Unique	0.08	0.03	-0.5	3.58	3.63	9
Total	1.54	2.45	2.76	3.68	3.69	18.57

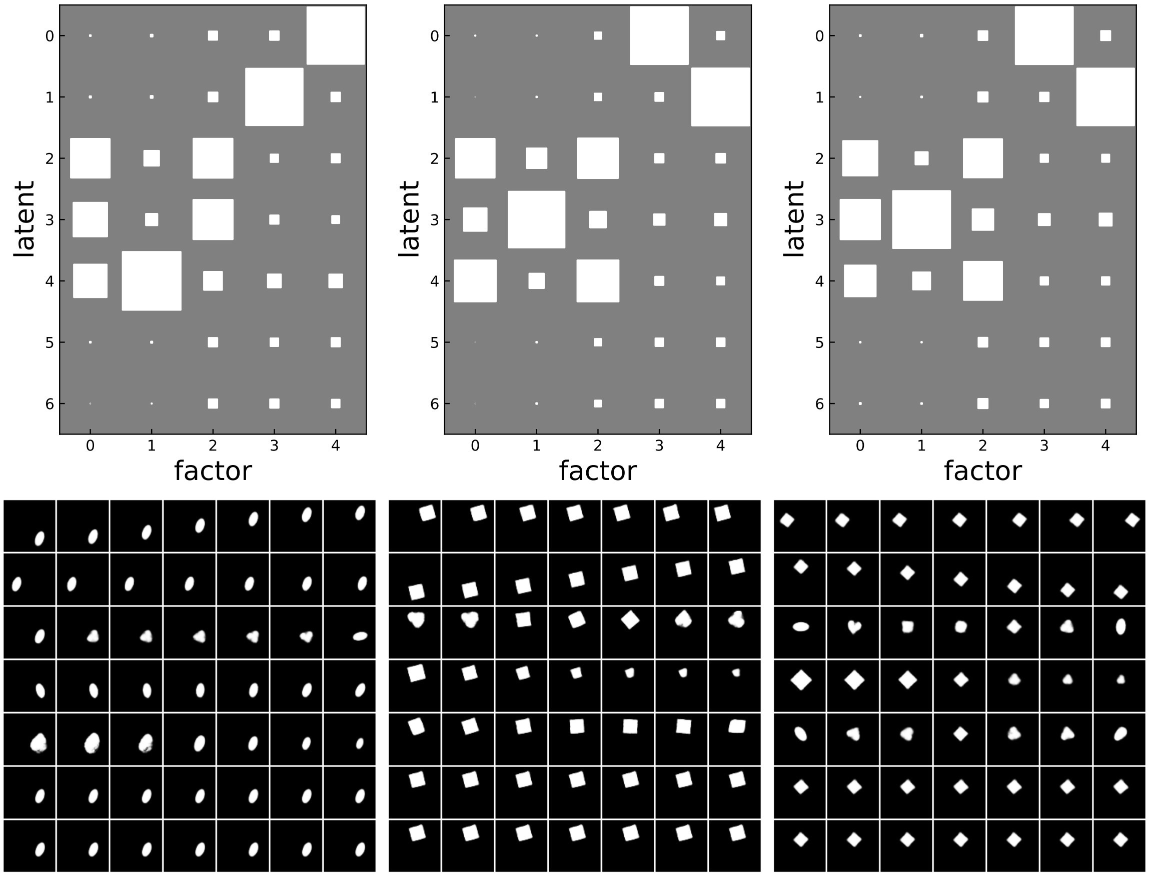

Figure F.1: Additional runs for the same experiment as Fig. 7.2 (right) for *common-dsprites*, with the same conventions as Fig 7.2.

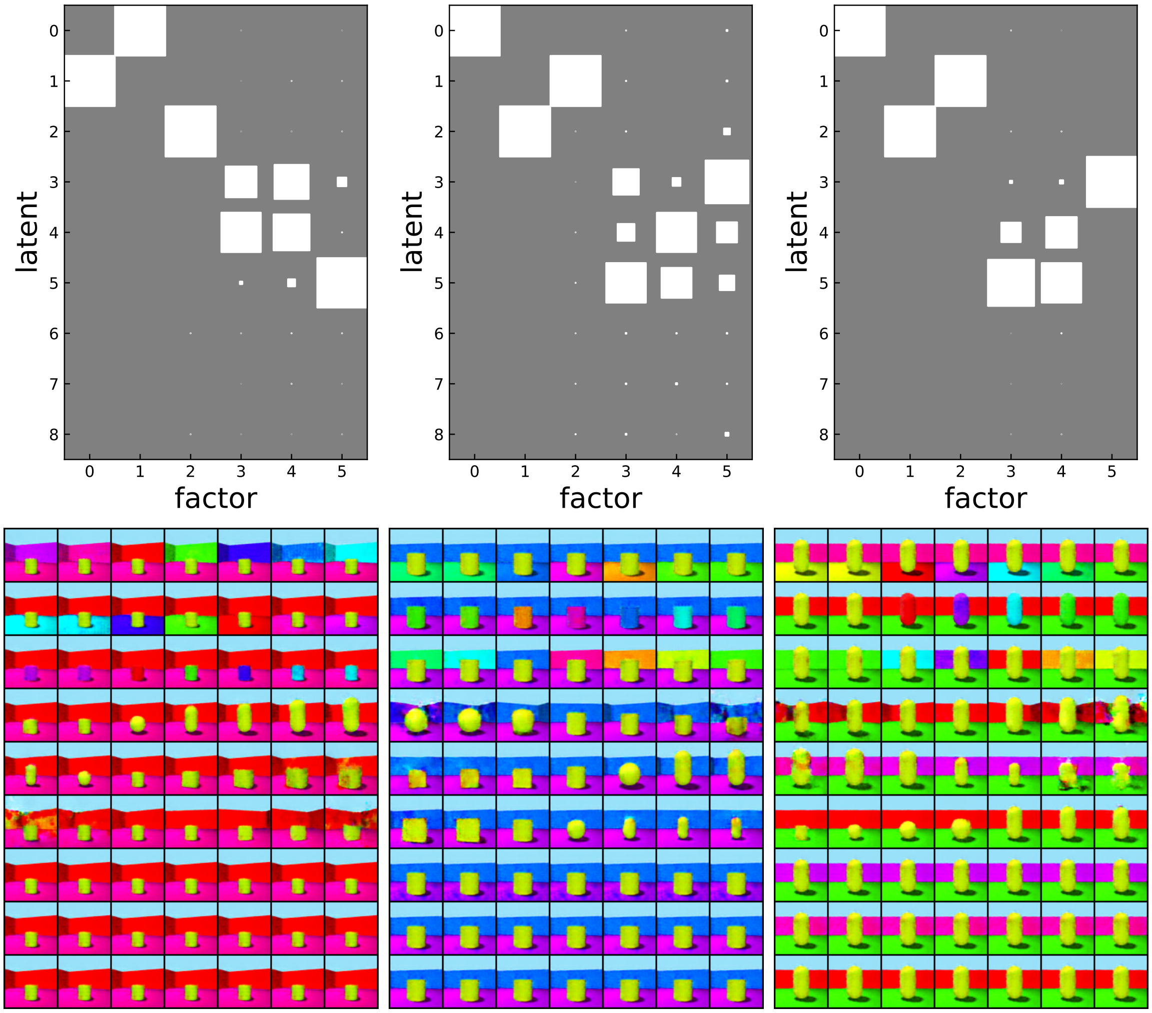

Figure F.2: Additional runs for the same experiment as Fig. 7.2 (right) for *common-3dshapes*, with the same conventions.

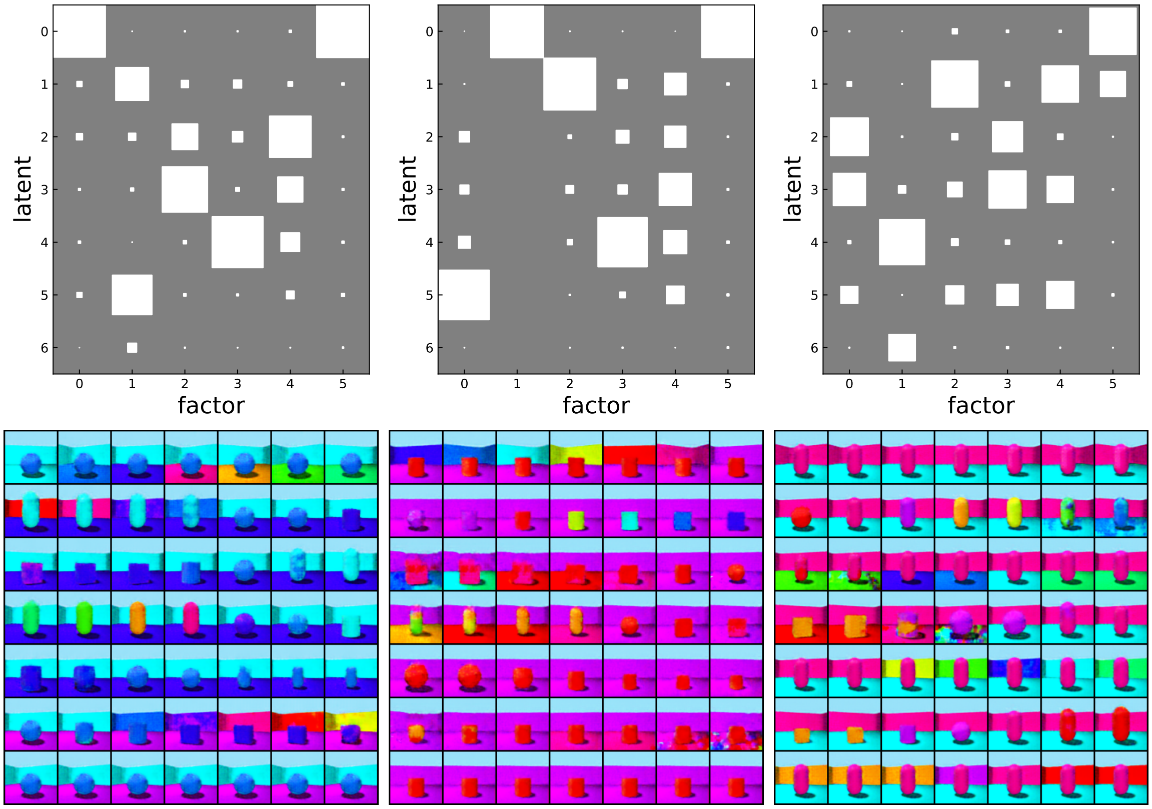

Figure F.3: DCI matrix of *common-3dshapes-2* (different viewpoints, latent factor 5 unique) for different random seeds. (Works better for smaller batch size. Ground truth generative model: factors 5 are unique. Unique latent variables are specified a priori to be latents: 0,

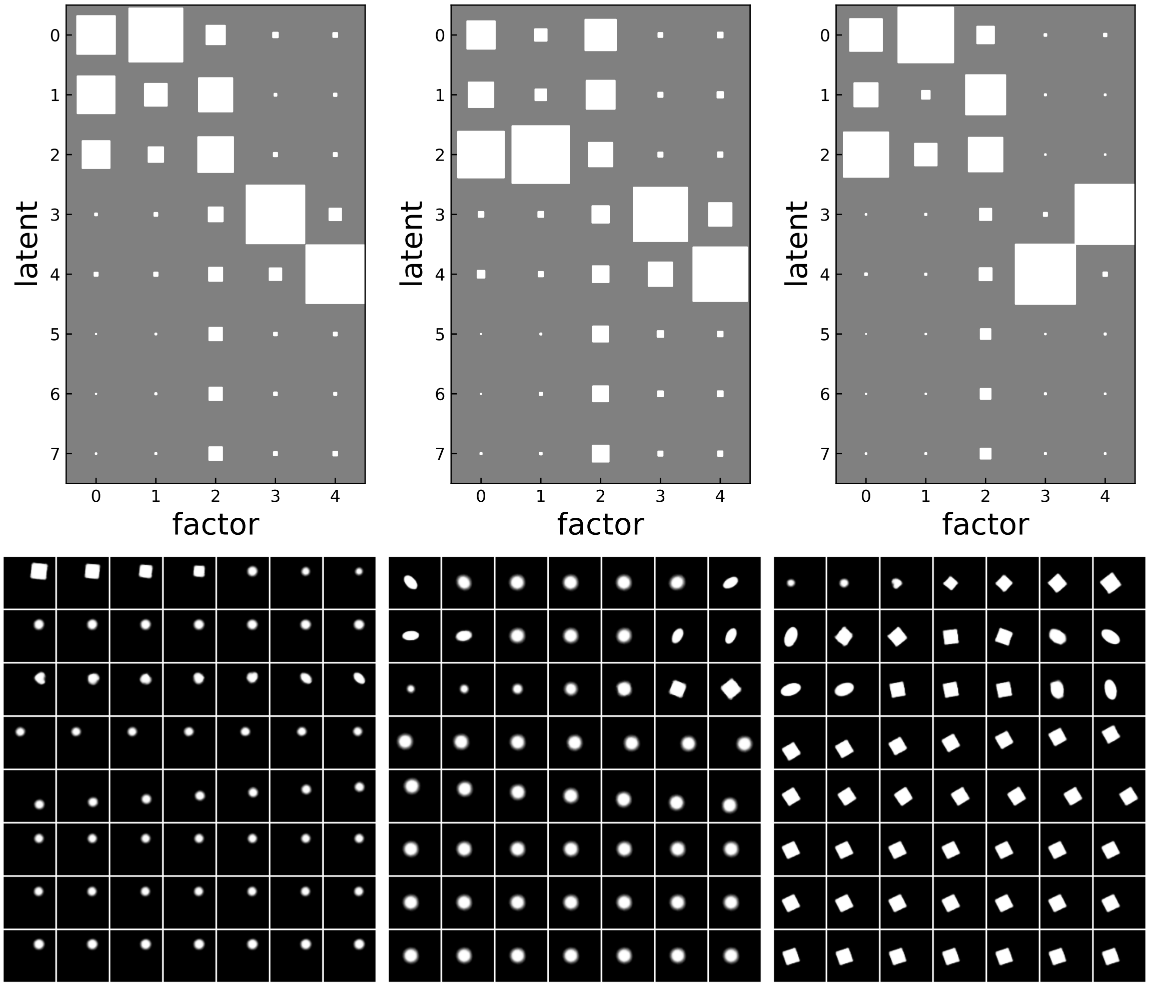

Figure F.4: DCI Matrix of dsprites for different random seeds. Common: Rows 4-5: Positions. Ground truth generative model: factors 0,1,2 are unique. Unique latent variables are specified a priori to be latents: 0,1,2

Bibliography

[1] Alessandro Achille, Giovanni Paolini, and Stefano Soatto. Where is the information in a deep neural network? *arXiv preprint arXiv:1905.12213*, 2019.

[2] Alessandro Achille, Matteo Rovere, and Stefano Soatto. Critical learning periods in deep networks. In *International Conference on Learning Representations*, 2019.

[3] Alessandro Achille and Stefano Soatto. Emergence of invariance and disentanglement in deep representations. *Journal of Machine Learning Research*, 19(50):1–34, 2018.

[4] Alessandro Achille and Stefano Soatto. Information dropout: Learning optimal representations through noisy computation. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2897–2905, 2018.

[5] Aishwarya Agrawal, Dhruv Batra, and Devi Parikh. Analyzing the behavior of visual question answering models. *arXiv preprint arXiv:1606.07356*, 2016.

[6] Alexander Alemi, Ben Poole, Ian Fischer, Joshua Dillon, Rif A Saurous, and Kevin Murphy. Fixing a broken elbo. In *International Conference on Machine Learning*, pages 159–168. PMLR, 2018.

[7] Galen Andrew, Raman Arora, Jeff Bilmes, and Karen Livescu. Deep canonical correlation analysis. In *International conference on machine learning*, pages 1247–1255. PMLR, 2013.

[8] David Attwell and Simon B Laughlin. An energy budget for signaling in the grey matter of the brain. *Journal of Cerebral Blood Flow & Metabolism*, 21(10):1133–1145, 2001.

[9] Roman Bachmann, David Mizrahi, Andrei Atanov, and Amir Zamir. Multimae: Multi-modal multi-task masked autoencoders. *arXiv preprint arXiv:2204.01678*, 2022.

[10] David Badre, Joshua Hoffman, Jeffrey W Cooney, and Mark D'esposito. Hierarchical cognitive control deficits following damage to the human frontal lobe. *Nature neuroscience*, 12(4):515, 2009.

[11] Tadas Baltru**v**saitis, Chaitanya Ahuja, and Louis-Philippe Morency. Multimodal machine learning: A survey and taxonomy. *IEEE transactions on pattern analysis and machine intelligence*, 41(2):423–443, 2018.

[12] P. K. Banerjee, J. Rauh, and G. Montúfar. Computing the unique information. In *2018 IEEE International Symposium on Information Theory (ISIT)*, pages 141–145, 2018.

[13] Pradeep Kr Banerjee and Virgil Griffith. Synergy, redundancy and common information. *arXiv preprint arXiv:1509.03706*, 2015.

[14] Andrea Banino, Caswell Barry, Benigno Uria, Charles Blundell, Timothy Lillicrap, Piotr Mirowski, Alexander Pritzel, Martin J Chadwick, Thomas Degris, Joseph Modayil, et al. Vector-based navigation using grid-like representations in artificial agents. *Nature*, 557(7705):429–433, 2018.

[15] David Barber and Felix Agakov. The im algorithm: A variational approach to information maximization. In *Proceedings of the 16th International Conference on Neural Information Processing Systems*, NIPS'03, page 201–208, Cambridge, MA, USA, 2003. MIT Press.

[16] Adam B Barrett. Exploration of synergistic and redundant information sharing in static and dynamical gaussian systems. *Physical Review E*, 91(5):052802, 2015.

[17] 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.

[18] Nils Bertschinger, Johannes Rauh, Eckehard Olbrich, Jürgen Jost, and Nihat Ay. Quantifying unique information. *Entropy*, 16(4):2161–2183, Apr 2014.

[19] Bingni W Brunton, Matthew M Botvinick, and Carlos D Brody. Rats and humans can optimally accumulate evidence for decision-making. *Science*, 340(6128):95–98, 2013.

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

[21] 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.

[22] Remi Cadene, Corentin Dancette, Matthieu Cord, Devi Parikh, et al. Rubi: Reducing unimodal biases for visual question answering. *Advances in neural information processing systems*, 32, 2019.

[23] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308, 2017.

[24] Chandramouli Chandrasekaran, Diogo Peixoto, William T. Newsome, and Krishna V. Shenoy. Laminar differences in decision-related neural activity in dorsal premotor cortex. *Nature Communications*, 8(1):614, 2017.

[25] Ricky T. Q. Chen, Xuechen Li, Roger Grosse, and David Duvenaud. Isolating sources of disentanglement in variational autoencoders. *Advances in Neural Information Processing Systems*, 2018.

[26] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[27] Mark M Churchland, John P Cunningham, Matthew T Kaufman, Justin D Foster, Paul Nuyujukian, Stephen I Ryu, and Krishna V Shenoy. Neural population dynamics during reaching. *Nature*, 487(7405):51–56, 2012.

[28] Paul Cisek. Making decisions through a distributed consensus. *Current opinion in neurobiology*, 22(6):927–936, 2012.

[29] Paul Cisek and John F Kalaska. Neural correlates of reaching decisions in dorsal premotor cortex: Specification of multiple direction choices and final selection of action. *Neuron*, 45(5):801–814, 2005.

[30] Émilie Coallier, Thomas Michelet, and John F Kalaska. Dorsal premotor cortex: neural correlates of reach target decisions based on a color-location matching rule and conflicting sensory evidence. *Journal of neurophysiology*, 113(10):3543–3573, 2015.

[31] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, USA, 2006.

[32] Bruce G Cumming and Hendrikje Nienborg. Feedforward and feedback sources of choice probability in neural population responses. *Current opinion in neurobiology*, 37:126–132, 2016.

[33] Maria C Dadarlat, Joseph E O'doherty, and Philip N Sabes. A learning-based approach to artificial sensory feedback leads to optimal integration. *Nature neuroscience*, 18(1):138–144, 2015.

[34] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[35] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[36] Yann Dubois, Douwe Kiela, David J. Schwab, and Ramakrishna Vedantam. Learning optimal representations with the decodable information bottleneck, 2020.

[37] Cian Eastwood and Christopher K. I. Williams. A framework for the quantitative evaluation of disentangled representations. 2018.

[38] Luca Faes, Daniele Marinazzo, and Sebastiano Stramaglia. Multiscale information decomposition: Exact computation for multivariate gaussian processes. *Entropy*, 19(8), 2017.

[39] Marco Federici, Anjan Dutta, Patrick Forré, Nate Kushman, and Zeynep Akata. Learning robust representations via multi-view information bottleneck. *arXiv preprint arXiv:2002.07017*, 2020.

[40] Christopher R Fetsch, Naomi N Odean, Danique Jeurissen, Yasmine El-Shamayleh, Gregory D Horwitz, and Michael N Shadlen. Focal optogenetic suppression in macaque area mt biases direction discrimination and decision confidence, but only transiently. *Elife*, 7:e36523, 2018.

[41] Ronald Aylmer Fisher. Theory of statistical estimation. In *Mathematical proceedings of the Cambridge philosophical society*, volume 22, pages 700–725. Cambridge University Press, 1925.

[42] David J Freedman and John A Assad. A proposed common neural mechanism for categorization and perceptual decisions. *Nature neuroscience*, 14(2):143, 2011.

[43] David J Freedman and John A Assad. Neuronal mechanisms of visual categorization: an abstract view on decision making. *Annual review of neuroscience*, 39:129–147, 2016.

[44] Stefano Fusi, Earl K Miller, and Mattia Rigotti. Why neurons mix: high dimensionality for higher cognition. *Current opinion in neurobiology*, 37:66–74, 2016.

[45] Peter Gács and János Körner. Common information is far less than mutual information. *Problems of Control and Information Theory*, 2(2):149–162, 1973.

[46] Peiran Gao, Eric Trautmann, Byron Yu, Gopal Santhanam, Stephen Ryu, Krishna Shenoy, and Surya Ganguli. A theory of multineuronal dimensionality, dynamics and measurement. *BioRxiv*, page 214262, 2017.

[47] Aditya Sharad Golatkar, Alessandro Achille, and Stefano Soatto. Time matters in regularizing deep networks: Weight decay and data augmentation affect early learning dynamics, matter little near convergence. In *Advances in Neural Information Processing Systems 32*, pages 10677–10687. Curran Associates, Inc., 2019.

[48] Joshua I Gold and Michael N Shadlen. The neural basis of decision making. *Annual review of neuroscience*, 30, 2007.

[49] Ziv Goldfeld, Ewout van den Berg, Kristjan H. Greenewald, Igor Melnyk, Nam Nguyen, Brian Kingsbury, and Yury Polyanskiy. Estimating information flow in neural networks. *CoRR*, abs/1810.05728, 2018.

[50] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning.* MIT Press, November 2016.

[51] Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017.

[52] Virgil Griffith, Edwin Chong, Ryan James, Christopher Ellison, and James Crutchfield. Intersection information based on common randomness. *Entropy*, 16(4):1985–2000, Apr 2014.

[53] Virgil Griffith and Tracey Ho. Quantifying redundant information in predicting a target random variable. *Entropy*, 17(12):4644–4653, Jul 2015.

[54] Malte Harder, Christoph Salge, and Daniel Polani. Bivariate measure of redundant information. *Phys. Rev. E*, 87:012130, Jan 2013.

[55] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[56] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. *arXiv preprint arXiv:2111.06377*, 2021.

[57] Richard Held and Alan Hein. Movement-produced stimulation in the development of visually guided behavior. *Journal of comparative and physiological psychology*, 56(5):872, 1963.

[58] Guillaume Hennequin, Tim P Vogels, and Wulfram Gerstner. Optimal control of transient dynamics in balanced networks supports generation of complex movements. *Neuron*, 82(6):1394–1406, June 2014.

[59] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic vi-

sual concepts with a constrained variational framework. *International Conference on Learning Representations*, 2017.

[60] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. *arXiv preprint arXiv:1808.06670*, 2018.

[61] Eiji Hoshi. Cortico-basal ganglia networks subserving goal-directed behavior mediated by conditional visuo-goal association. *Frontiers in neural circuits*, 7:158, 2013.

[62] Wei Hu, Lechao Xiao, Ben Adlam, and Jeffrey Pennington. The surprising simplicity of the early-time learning dynamics of neural networks. *Advances in Neural Information Processing Systems*, 33:17116–17128, 2020.

[63] Aapo Hyvärinen and Erkki Oja. Independent component analysis: algorithms and applications. *Neural networks*, 13(4-5):411–430, 2000.

[64] Danique Jeurissen, S Shushruth, Yasmine El-Shamayleh, Gregory D Horwitz, and Michael N Shadlen. Deficits in decision-making induced by parietal cortex inactivation are compensated at two timescales. *Neuron*, 110(12):1924–1931, 2022.

[65] Bela Julesz. Binocular depth perception of computer-generated patterns. *Bell System Technical Journal*, 39(5):1125–1162, 1960.

[66] JF Kalaska and DJ Crammond. Cerebral cortical mechanisms of reaching movements. *Science*, 255(5051):1517–1523, 1992.

[67] Eric R Kandel, James H Schwartz, Thomas M Jessell, Steven A Siegelbaum, and A J Hudspeth. *Principles of neural science.* McGraw-Hill, New York, fifth edition, 2013.

[68] Jonathan C Kao. Considerations in using recurrent neural networks to probe neural dynamics. *Journal of neurophysiology*, 122(6):2504–2521, 2019.

[69] Matthew T Kaufman, Mark M Churchland, Stephen I Ryu, and Krishna V Shenoy. Cortical activity in the null space: permitting preparation without movement. *Nat. Neurosci.*, 17(3):440–448, March 2014.

[70] Matthew T Kaufman, Mark M Churchland, Stephen I Ryu, and Krishna V Shenoy. Vacillation, indecision and hesitation in moment-by-moment decoding of monkey motor cortex. *Elife*, 4:1–21, 2015.

[71] Matthew T Kaufman, Jeffrey S Seely, David Sussillo, Stephen I Ryu, Krishna V Shenoy, and Mark M Churchland. The largest response component in the motor cortex reflects movement timing but not movement type. *eneuro*, 3(4), 2016.

[72] Isaac V. Kauvar, Timothy A. Machado, Elle Yuen, John Kochalka, Minseung Choi, William E. Allen, Gordon Wetzstein, and Karl Deisseroth. Cortical observation by synchronous multifocal optical sampling reveals widespread population encoding of actions. *Neuron*, 107(2):351 – 367.e19, 2020.

[73] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[74] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, December 2014.

[75] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2014.

[76] Durk P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved variational inference with inverse autoregressive flow. *Advances in neural information processing systems*, 29:4743–4751, 2016.

[77] 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.

[78] Michael Kleinman, Alessandro Achille, Daksh Idnani, and Jonathan Kao. Usable information and evolution of optimal representations during training. In *International Conference on Learning Representations*, 2021.

[79] Michael Kleinman, Alessandro Achille, and Stefano Soatto. Critical learning periods for multisensory integration in deep networks. *arXiv preprint arXiv:2210.04643*, 2022.

[80] Michael Kleinman, Alessandro Achille, Stefano Soatto, and Jonathan Kao. Gacs-korner common information variational autoencoder. *arXiv preprint arXiv:2205.12239*, 2022.

[81] Michael Kleinman, Alessandro Achille, Stefano Soatto, and Jonathan C. Kao. Redundant information neural estimation. *Entropy*, 23(7):922, 2021.

[82] Michael Kleinman, Chandramouli Chandrasekaran, and Jonathan Kao. A mechanistic multi-area recurrent network model of decision-making. In *Conference on Neural Information Processing Systems*, 2021.

[83] Michael Kleinman, Chandramouli Chandrasekaran, and Jonathan C. Kao. Recurrent neural network models of multi-area computation underlying decision-making. *bioRxiv*, 2019.

[84] Eric I Knudsen and Phyllis F Knudsen. Sensitive and critical periods for visual calibration of sound localization by barn owls. *Journal of Neuroscience*, 10(1):222–232, 1990.

[85] Dmitry Kobak, Wieland Brendel, Christos Constantinidis, Claudia E Feierstein, Adam Kepecs, Zachary F Mainen, Xue-Lian Qi, Ranulfo Romo, Naoshige Uchida, and Christian K Machens. Demixed principal component analysis of neural population data. *Elife*, 5, April 2016.

[86] Adam Kohn, Anna I. Jasper, João D. Semedo, Evren Gokcen, Christian K. Machens, and Byron M. Yu. Principles of corticocortical communication: Proposed schemes and design considerations. *Trends in Neurosciences*, 43(9):725–737, 2020/10/12 2020.

[87] Artemy Kolchinsky. A novel approach to multivariate redundancy and synergy. *arXiv preprint arXiv:1908.08642*, 2019.

[88] Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. Estimating mutual information. *Physical review E*, 69(6):066138, 2004.

[89] Rodrigo Laje and Dean V Buonomano. Robust timing and motor patterns by taming chaos in recurrent neural networks. *Nature neuroscience*, 16(7):925, 2013.

[90] Kenneth W. Latimer. Nonlinear demixed component analysis for neural population data as a low-rank kernel regression problem. *Neurons, Behavior, Data Analysis, & Theory*, pages 1–24, 2019.

[91] Jaehoon Lee, Lechao Xiao, Samuel Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, and Jeffrey Pennington. Wide neural networks of any depth evolve as linear models under gradient descent. *Advances in neural information processing systems*, 32, 2019.

[92] Nuo Li, Kayvon Daie, Karel Svoboda, and Shaul Druckmann. Robust neuronal dynamics in premotor cortex during motor planning. *Nature*, 532(7600):459–464, 2016.

[93] Yingzhen Li and Stephan Mandt. Disentangled sequential autoencoder. *arXiv preprint arXiv:1803.02991*, 2018.

[94] Timothy P. Lillicrap, Adam Santoro, Luke Marris, Colin J. Akerman, and Geoffrey Hinton. Backpropagation and the brain. *Nature Reviews Neuroscience*, 21(6):335–346, 2020.

[95] 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.

[96] Francesco Locatello, Ben Poole, Gunnar Rätsch, Bernhard Schölkopf, Olivier Bachem, and Michael Tschannen. Weakly-supervised disentanglement without compromises. In *International Conference on Machine Learning*, pages 6348–6359. PMLR, 2020.

[97] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

[98] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[99] Valerio Mante, David Sussillo, Krishna V Shenoy, and William T Newsome. Context-dependent computation by recurrent dynamics in prefrontal cortex. *Nature*, 503(7474):78–84, November 2013.

[100] N T Markov, M M Ercsey-Ravasz, A R Ribeiro Gomes, C Lamy, L Magrou, J Vezoli, P Misery, A Falchier, R Quilodran, M A Gariel, J Sallet, R Gamanut, C Huissoud, S Clavagnier, P Giroud, D Sappey-Marinier, P Barone, C Dehay, Z Toroczkai, K Knoblauch, D C Van Essen, and H Kennedy. A weighted and directed interareal connectivity matrix for macaque cerebral cortex. *Cereb. Cortex*, 24(1):17–36, January 2014.

[101] Loic Matthey, Irina Higgins, Demis Hassabis, and Alexander Lerchner. dsprites: Disentanglement testing sprites dataset. https://github.com/deepmind/dsprites-dataset/, 2017.

[102] Beatriz EP Mizusaki and Cian O'Donnell. Neural circuit function redundancy in brain disorders. *Current opinion in neurobiology*, 70:74–80, 2021.

[103] Hendrikje Nienborg and Bruce G Cumming. Decision-related activity in sensory neurons reflects more than a neuron's causal effect. *Nature*, 459(7243):89–92, 2009.

[104] Paul Nuyujukian, Jonathan C Kao, Joline M Fan, Sergey D Stavisky, Stephen I Ryu, and Krishna V Shenoy. Performance sustaining intracortical neural prostheses. *Journal of neural engineering*, 11(6):066003, 2014.

[105] Liam Paninski. Estimation of entropy and mutual information. *Neural Comput.*, 15(6):1191–1253, June 2003.

[106] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning*, pages 1310–1318, February 2013.

[107] Lucas Pinto, Kanaka Rajan, Brian DePasquale, Stephan Y Thiberge, David W Tank, and Carlos D Brody. Task-dependent changes in the large-scale dynamics and necessity of cortical regions. *Neuron*, 104(4):810–824, 2019.

[108] Ben Poole, Sherjil Ozair, Aaron Van Den Oord, Alex Alemi, and George Tucker. On variational bounds of mutual information. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5171–5180, Long Beach, California, USA, 09–15 Jun 2019. PMLR.

[109] Roger Ratcliff. A theory of memory retrieval. *Psychological review*, 85(2):59, 1978.

[110] Evan D Remington, Devika Narain, Eghbal A Hosseini, and Mehrdad Jazayeri. Flexible sensorimotor computations through rapid reconfiguration of cortical dynamics. *Neuron*, 98(5):1005–1019.e5, June 2018.

[111] Mattia Rigotti, Omri Barak, Melissa R Warden, Xiao-Jing Wang, Nathaniel D Daw, Earl K Miller, and Stefano Fusi. The importance of mixed selectivity in complex cognitive tasks. *Nature*, 497(7451):585–590, May 2013.

[112] Ranulfo Romo and Victor de Lafuente. Conversion of sensory signals into perceptual decisions. *Progress in Neurobiology*, 103:41–75, 2013. Conversion of Sensory Signals into Perceptions, Memories and Decisions.

[113] S. Salamatian, A. Cohen, and M. Médard. Approximate gács-körner common information. In *2020 IEEE International Symposium on Information Theory (ISIT)*, pages 2234–2239, 2020.

[114] Salman Salamatian, Asaf Cohen, and Muriel Médard. Maximum entropy functions: Approximate gacs-korner for distributed compression. *arXiv preprint arXiv:1604.03877*, 2016.

[115] Gopal Santhanam, Stephen I Ryu, M Yu Byron, Afsheen Afshar, and Krishna V Shenoy. A high-performance brain–computer interface. *nature*, 442(7099):195–198, 2006.

[116] Gopal Santhanam, Byron M Yu, Vikash Gilja, Stephen I Ryu, Afsheen Afshar, Maneesh Sahani, and Krishna V Shenoy. Factor-analysis methods for higher-performance neural prostheses. *Journal of neurophysiology*, 102(2):1315–1330, 2009.

[117] 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.

[118] 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.

[119] Andrew Michael Saxe, Yamini Bansal, Joel Dapello, Madhu Advani, Artemy Kolchinsky, Brendan Daniel Tracey, and David Daniel Cox. On the information bottleneck theory of deep learning. 2018.

[120] João D Semedo, Amin Zandvakili, Christian K Machens, Byron M Yu, and Adam Kohn. Cortical Areas Interact through a Communication Subspace. *Neuron*, 102(1):249–259.e4, apr 2019.

[121] Biswa Sengupta, Martin Stemmler, Simon B Laughlin, and Jeremy E Niven. Action potential energy efficiency varies among neuron types in vertebrates and invertebrates. *PLoS computational biology*, 6(7):e1000840, 2010.

[122] Krishna V Shenoy, Maneesh Sahani, and Mark M Churchland. Cortical control of arm movements: a dynamical systems perspective. *Annual review of neuroscience*, 36:337–359, 2013.

[123] Ravid Shwartz-Ziv and Naftali Tishby. Opening the black box of deep neural networks via information. *CoRR*, abs/1703.00810, 2017.

[124] Markus Siegel, Timothy J Buschman, and Earl K Miller. Cortical information flow during flexible sensorimotor decisions. *Science*, 348(6241):1352–1355, 2015.

[125] Linda Smith and Michael Gasser. The development of embodied cognition: Six lessons from babies. *Artificial life*, 11(1-2):13–29, 2005.

[126] S. Soatto and A. Chiuso. Visual representations: Defining properties and deep approximation. In *Proceedings of the International Conference on Learning Representations (ICLR)*, June 2016.

[127] H Francis Song, Guangyu R Yang, and Xiao Jing Wang. Training excitatory-inhibitory recurrent neural networks for cognitive tasks: a simple and flexible framework. *PLoS Comput. Biol.*, 12(2):1–30, 2016.

[128] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014.

[129] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net, 2015.

[130] Sergey D Stavisky, Jonathan C Kao, Stephen I Ryu, and Krishna V Shenoy. Motor cortical visuomotor feedback activity is initially isolated from downstream targets in Output-Null neural state space dimensions. *Neuron*, 95(1):195–208.e9, July 2017.

[131] Jake P Stroud, Mason A Porter, Guillaume Hennequin, and Tim P Vogels. Motor primitives in space and time via targeted gain modulation in cortical networks. *Nature neuroscience*, 21(12):1774, 2018.

[132] David Sussillo, Mark M Churchland, Matthew T Kaufman, and Krishna V Shenoy. A neural network that finds a naturalistic solution for the production of muscle activity. *Nat. Neurosci.*, 18(7):1025–1033, 2015.

[133] Sara M. Szczepanski, Christina S. Konen, and Sabine Kastner. Mechanisms of spatial attention control in frontal and parietal cortex. *Journal of Neuroscience*, 30(1):148–160, 2010.

[134] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. In *European conference on computer vision*, pages 776–794. Springer, 2020.

[135] Yonglong Tian, Chen Sun, Ben Poole, Dilip Krishnan, Cordelia Schmid, and Phillip Isola. What makes for good views for contrastive learning? *Advances in Neural Information Processing Systems*, 33:6827–6839, 2020.

[136] Naftali Tishby, Fernando C. Pereira, and William Bialek. The information bottleneck method. In *Proc. of the 37-th Annual Allerton Conference on Communication, Control and Computing*, pages 368–377, 1999.

[137] James Townsend, Tom Bird, and David Barber. Practical lossless compression with latent variables using bits back coding. *arXiv preprint arXiv:1901.04866*, 2019.

[138] Leslie G Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.

[139] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[140] Haoqing Wang, Xun Guo, Zhi-Hong Deng, and Yan Lu. Rethinking minimal sufficient representation in contrastive learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16041–16050, 2022.

[141] Megan Wang, Christéva Montanède, Chandramouli Chandrasekaran, Diogo Peixoto, Krishna V Shenoy, and John F Kalaska. Macaque dorsal premotor cortex exhibits decision-related activity only when specific stimulus-response associations are known. *Nature Communications*, 10(1):1793, 2019.

[142] Weiran Wang, Raman Arora, Karen Livescu, and Jeff Bilmes. On deep multi-view representation learning. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, page 1083–1092. JMLR.org, 2015.

[143] Weiran Wang, Xinchen Yan, Honglak Lee, and Karen Livescu. Deep variational canonical correlation analysis. *arXiv preprint arXiv:1610.03454*, 2016.

[144] Weiyao Wang, Du Tran, and Matt Feiszli. What makes training multi-modal classification networks hard? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12695–12705, 2020.

[145] Satosi Watanabe. Information theoretical analysis of multivariate correlation. *IBM Journal of research and development*, 4(1):66–82, 1960.

[146] Torsten N Wiesel. Postnatal development of the visual cortex and the influence of environment. *Nature*, 299(5884):583–591, 1982.

[147] Paul L Williams and Randall D Beer. Nonnegative decomposition of multivariate information. *arXiv preprint arXiv:1004.2515*, 2010.

[148] Steven P Wise and Elisabeth A Murray. Arbitrary associations between antecedents and actions. *Trends in Neurosciences*, 23(6):271–276, 2000.

[149] Stefan Wolf and J Wultschleger. Zero-error information and applications in cryptography. In *Information Theory Workshop*, pages 1–6. IEEE, 2004.

[150] Nan Wu, Stanislaw Jastrzebski, Kyunghyun Cho, and Krzysztof J Geras. Characterizing and overcoming the greedy nature of learning in multi-modal deep neural networks. In *International Conference on Machine Learning*, pages 24043–24055. PMLR, 2022.

[151] Yilun Xu, Shengjia Zhao, Jiaming Song, Russell Stewart, and Stefano Ermon. A theory of usable information under computational constraints. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[152] Tomoko Yamagata, Yoshihisa Nakayama, Jun Tanji, and Eiji Hoshi. Distinct information representation and processing for goal-directed behavior in the dorsolateral and ventrolateral prefrontal cortex and the dorsal premotor cortex. *Journal of Neuroscience*, 32(37):12934–12949, 2012.

[153] Daniel L K Yamins and James J DiCarlo. Using goal-driven deep learning models to understand sensory cortex. *Nat. Neurosci.*, 19(3):356–365, March 2016.

[154] Daniel LK Yamins, Ha Hong, Charles F Cadieu, Ethan A Solomon, Darren Seibert, and James J DiCarlo. Performance-optimized hierarchical models predict neural responses in higher visual cortex. *Proceedings of the national academy of sciences*, 111(23):8619–8624, 2014.

[155] 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.

[156] Jeffrey M. Yau, Gregory C. DeAngelis, and Dora E. Angelaki. Dissecting neural circuits for multisensory integration and crossmodal processing. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 370(1677):20140203, 2015.