MULTITASKING NETWORKS USE MULTIAFFINE REPRESENTATIONS TO DIRECT FLOW OF FEATURE DATA

Gregory Henselman-Petrusek, Tyler Giallanza, Sebastian Musslick, & Jonathan D. Cohen
Princeton Neuroscience Institute
Princeton University
Princeton, NJ 08540, USA
{gh10, tylerg, musslick, jdc}@princeton.edu

ABSTRACT

Artificial systems currently outperform humans in diverse computational domains, but none has achieved parity in speed and overall versatility of mastering novel tasks. A critical component to human success, in this regard, is the ability to redeploy and redirect data passed between cognitive subsystems (via abstract feature representations) in response to changing task demands. The present work formalizes this coordination procedure in terms of multiaffine functions of stimulus and control states. In experiments, the resulting model robustly predicts behavior and performance of multitasking networks on natural language data (MNIST) using common deep network architectures. Consistent with existing theory in cognitive control, representation structure varies in response to (a) environmental pressures for representation sharing, (b) demands for parallel processing capacity, and (c) tolerance for crosstalk. Implications for geometric (dimension, curvature), functional (automaticity, generalizability, modularity), and applied aspects of representation learning are discussed.

The separation of cognitive or sensory data into distinct dimensions, or *features*, is a central theme of intelligent systems (Kriegeskorte and Kievit, 2013; LeCun et al., 2015; Yamins et al., 2014). Determining how agents divide and recombine distinct channels of feature information to pursue higher goals is famously complex and challenging, and has been studied in many forms, e.g. distributed representation (McClelland et al., 1986) and the binding problem (Treisman, 1996). A common example asks how an agent with separate processing pathways for color and size distinguishes scenes with a blue circle and red square from scenes with a red circle and blue square.

Understanding pairwise interactions between the mechanisms that encode or represent individual feature channels is a basic step toward addressing this and other questions. While our basic knowledge such interactions remains highly incomplete, recent progress in basic understanding how these mechanisms operate, both in both humans and in machines, suggests tractable new lines of inquiry.

The present work offers proof of concept for one such direction, posing mild assumptions on underlying mechanisms (linear separability of feature data) which have been experimentally supported in contexts of interest for both human and machine studies (Rigotti et al., 2013; Chung et al., 2018; Bernardi et al., 2020).

Our test problem consists of constructing a formally rigorous model to robustly predict parallel processing capacity in networks trained to perform many tasks. This problem is motivated by scientific and computational problems in *cognitive control*, a field centrally concerned with systems that flexibly redirect and redeploy channels of feature data in response to changing environmental demands (abstraction, generalization) (Cohen et al., 2020).

Specifically, based on limited assumptions of linearity, we present a mathematically rigorous, geometric approach to analyzing networks. We demonstrate its utility by showing that it reveals network characteristics consistent with numerical evidence in support of the hypothesis that the same mechanisms that enable sharing of representations across tasks (and reap the concomitant benefits of transfer learning, abstraction, and generalization) make a system susceptible to channel-irrelevant information leaking into systems that otherwise perform correctly, thus inducing cross talk and signal degradation. Such local interactions have been shown to have global implications for a system

as a whole, both for parallel processing capacity (as suggested, for example, by existing bodies of literature linking crosstalk to limitations on parallel processing capacity in humans (Musslick et al., 2016; 2017; Logan and Gordon, 2001; Townsend and Wenger, 2004; Pashler, 1994)) and for the macro-scale organization of learning agents in general, and thus the ability to characterize them formally has potential broad-ranging significance.

In §1 we describe the test domain and its relation to feature representations. In §2 we describe a class of network architectures designed to operate in this domain, as well as two specific implementations. In §3 we use a simple test statistic to show that local, geometric comparisons between feature representations can predict global computational properties of a mutltitask-trained system as a whole. Consistent with predictions, this tie becomes stronger in deeper layers of network where, presumably, linear separation of feature values plays a stronger role and exposes the underlying relationships with greater clarity.

1 BACKGROUND

An attractive approach to the binding problem is to analyze patterns in which the systems that coordinate representations consistently fail. This approach has been explored from a number of angles, including the domain of parallel processing in human cognition (commonly referred to in cognitive and neuroscience literature as *multitasking* (Logan and Gordon, 2001; Musslick et al., 2016), the performance of multiple tasks simultaneously; this should be contrasted with *multitask learning*, a prominent branch of computer science devoted to the study of systems which learn to perform several different tasks in sequence) (Caruana, 1997; Maurer et al., 2016). The subject of parallel processing, and the broader field of cognitive control, examines phenomena such as the famous Stroop effect (subjects struggle to say "red" when presented with the word "green" typed in red ink) in terms underlying representations (Cohen et al., 1990).

One proposal is that Stroop-like effects (and their adverse consequences) result from a strategic sacrifice in computational abilities. More specifically, it is theorized that the representational structures which conduce to two desirable aspects of cognition – abstraction and automaticity/parallel processing capacity – are functionally incompatible in certain circumstances, and that Stroop-like pheomena result where abstraction has been favored over automaticity.

The key points of this assertion are threefold:

1. If a cognitive process P relies on some feature representation F to perform a specific function, then information will flow from F to P whenever both F and P are active (unless external forces intervene to disrupt this transfer; a subject can *can* choose to speak "red" when presented with the word "green" typed in red ink, but conscious effort is generally required to achieve this).
2. If P relies on input from two different sources, say F and G, to perform specific functions in different contexts, then at any time when F, G, and P are simultaneously active, process P will receive input from both sources. In some cases these dual inputs will mutually interfere and thus degrade. In others they will compete, hence the Stroop phenomenon.
3. Consequently, if a putative system architect wishes P to be able to process information from F while another process Q receives the data stored in G, it will be necessary to route this data to Q not through G but some other, separate representation G', which encodes the same data as G, but which transmits no information to P.

These three points set up the basic tradeoff: if one wishes to process data of the types stored in F and G simultaneously, then one must build redundancy into the system with G'. This redundancy may come at a cost, since there are appreciable benefits to sharing a single representation across tasks, in general. In Ravi et al. (in preparation), for example, networks trained to perform multiple tasks learned faster when exposed to conditions which favor shared representations, and slower when required to perform multiple tasks simultaneously during training. Similar effects were observed in Alon et al. (2017); Musslick et al. (2016; 2017). Such a phenomenon is interesting both in itself, and more broadly as a an example of the (many) diverse constraints placed on a global learning system by its local structure.

While the evidence gathered around this theory is compelling, direct experimental support remains a challenge. An important reason for this is that the notions of process and representation are difficult to quantify in many domains of interest. However, work by Rigotti et al. (2013), and more recently Bernardi et al. (2020), among others, suggests that standard methods may be effective in several domains of interest. The model proposed in these approaches expresses the activity of a computational unit, e.g. a neuron, as a function

$$m(c, f) = \sum_{d \in D} \sum_{i \leq n} \chi_d(c) \beta_d^i(f_i) \tag{1}$$

where D is a set of contexts, $f_1, \ldots, f_n$ are features (such as size, shape, color), χ_d is the characteristic function such that $\chi_d(d) = 1$ and $\chi_d(c) = 0$ when $c \neq d$, and β_d^i is a map from feature features to activity levels in the unit. To wit, each β_d^i represents a encoding of feature f_i specific to context d.

Such a model has advantages, from a computational standpoint, due to its linear structure; in particular, m is *multiaffine* in variables c and f. In the context of parallel performance, it likewise suggests a simple mechanism for signal interference. Suppose, for example, that F and G correspond to feature dimensions f_i and f_j, and that P gathers information about these features from a population of neurons whose vector of activity patterns $m(f, c)$ is faithfully captured by the regression model. If P receives data from this population in the form of an affine transformation $Tm(f, c)$ of $m(f, c)$, and if the regression functions β_c^i and β_d^i are the same for both c, the context where P must read from G, and d, the context where P must read from F and Q from G, then linearity implies that $Tm(f, c)$ will vary with G, consistent with the high-level intuition.

This model is attractive both for its simplicity and for its potential applications, but substantial work is required to examine its validity across a broader range of learning systems, both human and machine. In §2 we present an example framework to support such investigation.

2 METHODS

To test the basic tenets of the framework outlined in §1, we train families of neural networks in two environments which, while drawn from natural data, help to expose the underlying representational structure. Specifically, we seek to verify that (a) networks which share representations, in the form of similar encoding functions β_d^i estimated through linear regression, consistently induce crosstalk, (b) this crosstalk, in turn, degrades signal quality, making it functionally impossible to for downstream processes to extract needed input data.

We first describe the environments, then the networks, and finally, return to the proposed regression model. Treatment of certain details concerning formal foundations are abbreviated by necessity; the reader is referred to Lesnick et al. (2020) for a complete exposition.

2.1 TASK ENVIRONMENTS

Consider a hypothetical participant in a study on digit reading. For the purposes of the experiment, each digit has two meaningful feature dimensions: a *parity*, f_P, valued in the set $\mathbf{F}_P = \{\text{even, odd}\}$, and *magnitude*, f_M, valued in the set $\mathbf{F}_M = \{\text{big, small}\}$. The agent likewise has two available action types, or *response dimensions*: a left-handed press of the D or F key on a keyboard, coded as a variable $r_L \in \mathbf{R}_L = \{\text{D , F }\}$, or a right-handed button press coded as $r_R \in \mathbf{R}_R = \{\text{J, K}\}$. If we assign to D and J the meaning of "true" and to F and K the values of "false," then the statement "digit is even" determines mappings $R_{P,L} : \mathbf{F}_P \to \mathbf{R}_L$ and $R_{P,R} : \mathbf{F}_P \to \mathbf{R}_R$. Maps $R_{M,L} : \mathbf{F}_M \to \mathbf{R}_L$ and $R_{M,R} : \mathbf{F}_M \to \mathbf{R}_R$ arise similarly for the statement "digit is big." We refer to these as *action mappings*.

It would be physically possible for the participant to execute certain pairs of these action mappings simultaneously, at least in principle: presented with a single digit, they could indicate parity with the left hand and, at the same time, indicate magnitude with the right. On the other hand, some combinations of task mappings are either ill-posed (for example, $R_{P,L}$ and $R_{M,L}$ impose mutually exclusive demands for the action of the left-hand button press, for certain values of parity and magnitude), or fail to capture the basic phenomena we wish to study (for example, simultaneous execution of each action mapping in the set $\{R_{P,L}, R_{P,R}\}$ only requires the subject to process one feature dimension (parity) simultaneously, rather one distinct feature for each of the action mappings).

Since we have only one action mapping $R_{x,y}$ for each ordered pair $(x, y) \in \{P, M\} \times \{L, R\}$, we can characterize the set of all feasible combinations, excluding those that are ill posed or redundant, by the family of (sets of) input-output pairs

$$\mathbf{C} = \{\{(x_1, y_1), \ldots, (x_n, y_n)\} : x_i = x_j \iff y_i = y_j \iff i = j\} \tag{2}$$

The action mapping associated to $c \in \mathbf{C}$ is formally characterized as the function $\mathcal{M}_c$ such that, for each j,

$$\mathcal{M}_c(f_P, f_M)_j = \begin{cases} R_j\left(f_i\right) & (i,j) \in c \\ \text{null} & (i,j) \notin c \text{ for any } i \end{cases} \tag{3}$$

where "null" represents inaction. We call $\mathcal{M}_c$ a *multitask mapping of arity* n, where n is the set cardinality of c. Where context leaves no room for confusion, we shorten this to n*-task* or simply *multitask*, if we do not wish to specify cardinality.

These definitions naturally generalize across a broad class of task domains: one may add arbitrarily many new feature dimensions $(\mathbf{F}_i)_{i\in I}$ and response dimensions $(\mathbf{R})_{j\in J}$; for each pair (i, j), one may declare an action mapping $\mathbf{F}_i \to \mathbf{R}_j$. The set of feasible input-output combinations and associated task mappings are then specified by equation 2 and equation 3, respectively. Letting $\prod_i \mathbf{F}_i$ denote the Cartesian product of the sets $\mathbf{F}_i$, we refer to the function

$$\mathcal{M} : (\textstyle\prod_i \mathbf{F}_i) \times \mathbf{C} \to \prod_j \mathbf{R}_j,\ (f, c) \mapsto \mathcal{M}_c(f)$$

as the agent's *task environment*. Since this function comes equipped with an explicit domain and codomain, it represents an exhaustive specification of all multitasks which the participant might be asked to perform.

A *sample* from a task environment is an indexed family $((s^k, c^k), f^k, r^k)_{k=1}^N$, where

1. $r^k = (r^k_j)_{j\in J} = \mathcal{M}_{c^k}(f^k)$ is a tuple of responses
2. s^k denotes a *stimulus*, and
3. $f^k = (f^k_i)_{i\in I}$ denotes the tuple of feature values associated to that stimulus

For example, in the example experiment one might have $((s^k, c^k), f^k, r^k) = ((7, \{(\text{P}, \text{L})\}), (\text{odd}, \text{big}), (\text{F}\,, \text{null}))$. A deterministic *agent* is a function $\mathcal{A} : \mathbf{S} \times \mathbf{C} \to \prod_j \mathbf{R}_j$, where $\mathbf{S}$ denotes the space of stimuli.

Finally, to translate this model into concrete operational terms, it will be convenient to fix an injective embedding $\iota_j : \mathbf{R}_j \to \bar{\mathbf{R}}_j$ from each response dimension $\mathbf{R}_j$ into a real-linear vector space $\bar{\mathbf{R}}_j$. We will tacitly identify $\mathcal{A}$ with a map $\mathbf{S} \times \mathbf{C} \to \prod_j \bar{\mathbf{R}}_j$, under this embedding.

2.2 Network models

To test qualitative predictions posed by the theoretical framework outlined in §1, we examined the hidden layer activity patterns in two classes of neural networks trained as (deterministic) agents in the parallel-multitasking paradigm described in §2.1.

Each agent operates in a task environment with two feature (respectively, response) dimensions, and may be completely specified by the following data:

1. A sequence of *processing* layers $L_0, \ldots, L_N$ with projection maps $p_\alpha : \mathbb{R}^{L_{\alpha-1}} \to \mathbb{R}^{L_\alpha}$
2. A *control* layer $L_{\mathbf{C}}$ with projection maps $q_\alpha : \mathbb{R}^{L_{\mathbf{C}}} \to \mathbb{R}^{L_\alpha}$ for $\alpha = 1, \ldots, N$.
3. A rectification function $\sigma_\alpha : \mathbb{R}^{L_\alpha} \to \mathbb{R}^{L_\alpha}$ for each $\alpha = 1, \ldots, N$

We call L_0 the *input* or *stimulus* layer and L_N the *response* layer. The response layer subdivides two equal-sized sublayers $L_{N,0}$ and $L_{N,1}$, corresponding to the two response dimensions.

In each case, the control layer $L_{\mathbf{C}}$ contains four units labeled by the four elements of $I \times J$, where I and J are index sets of cardinality 2. We can identify each $c \in \mathbf{C}$ with its indicator function $\chi_c : I \times J \to \{0, 1\}$, and this indicator function with an activity pattern in $L_{\mathbf{C}}$.

For $\alpha \geq 1$, the unit activity pattern across L_α for user-specified $\boldsymbol{s} \in \mathbb{R}^{L_0}$ and $\boldsymbol{c} \in \mathbb{R}^{L_{\mathbf{C}}}$ is given by $\boldsymbol{m}_\alpha(\boldsymbol{s}, \boldsymbol{c}) = \sigma_\alpha(p_\alpha(\boldsymbol{m}_{\alpha-1}(\boldsymbol{s}, \boldsymbol{c})) + q_\alpha(\boldsymbol{c}))$. Thus each network is completely specified by the shapes of the layers L_α, the rectifications σ_α, and the projections p_α and q_α.

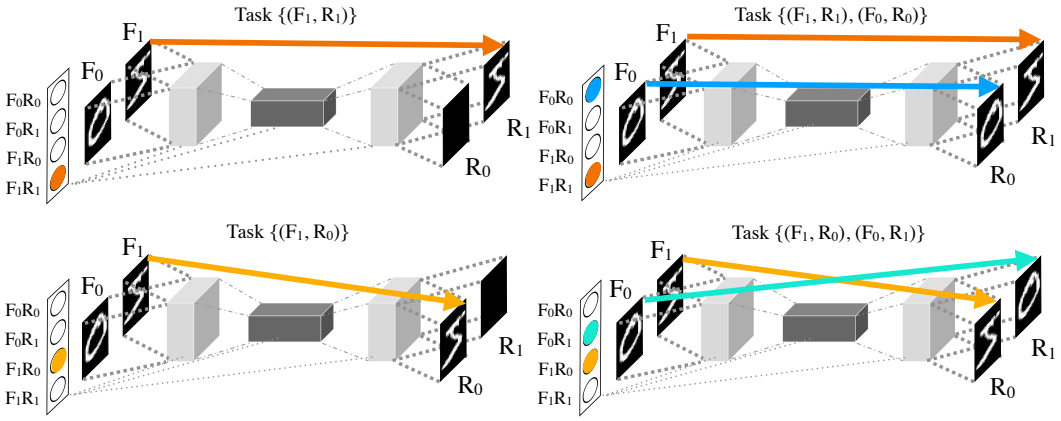

Figure 1: Schematic of multi-input, multi-output MNIST convolutional autoencoder.

2.2.1 CLASSIFICATION: PARITY AND MAGNITUDE

Description This family of models represents a minimal modification to those presented in Bernardi et al. (2020) for the purpose of exploring abstract representations. The task environment for these networks is the parity/magnitude function $\mathcal{M}$ described in §2. The embedding functions $\iota_j : \mathbf{R}_j \to \bar{\mathbf{R}}_j := \mathbb{R}^2$ are one-hot encodings. Stimuli take the form of 28x28 images from the MNIST data. Digits 1-4 are small, 5-8 big. We exclude 0's and 9's so that parity and magnitude have correlation 0.

Architecture Here $N = 5$, and layers $L_0, \ldots, L_N$ have shapes $28 \times 28, N_{\text{hddn}}, N_{\text{hddn}}, N_{\text{hddn}}, 4$ respectively where $N_{\text{hddn}} \in \{10, 100\}$ is an experimentally controlled parameter. As in Bernardi et al. (2020), each of projection p_α and q_α takes form $x \mapsto x[:]A + b$, where A is a weight matrix, b is a bias term, and $x[:]$ is the row vector obtained by flattening x. The rectification function σ_α is ELU (not ReLU) for $\alpha < N$ and logistic for $\alpha = N$.

2.2.2 IMAGE AUTOENCODING

Description These networks implement an image autoencoder with a twist. Input layer L_0 splits as the disjoint union of two sublayers, $L_{0,0}$ and $L_{0,1}$ each of size 28x28. Output layer L_N splits similarly into 28x28 layers $L_{N,0}$ and $L_{N,1}$. The inputs to each $L_{0,i}$ consist of a pair of 28x28 MNIST images; concretely, feature dimension $\mathbf{F}_i$ is the space of gray-scale images that can be passed to $L_{0,i}$. Thus the feature, stimulus, and response sets can all be identified with the same underlying space of images. Each embedding $\iota_j : \mathbf{R}_j \to \bar{\mathbf{R}}_j$ is an identity function, as is every action mapping (hence to "perform" task (i, j) means to copy the image from input i to output j).

Architecture Here $N = 4$. Projection p_0 sends an image I to a sum $p_{1,0}(I) + p_{1,1}(I)$, where $p_{1,0}$ and $p_{1,1}$ are convolutional projections with 3x3 kernels, followed by pooling. The final projection $p_4 : L_3 \to L_4$ splits similarly as a pair of transpose convolutional layers $p_{4,0}$ and $p_{4,1}$ with stride 2. Map $p_2 : L_1 \to L_2$ consists of a convolutional projection followed by pooling, and map $p_3 : L_2 \to L_3$ is transpose convolutional. Layers L_1, L_2, and L_3 have N_{chnl}, 4, and N_{chnl} channels, respectively, where $N_{\text{chnl}} \in \{5, 30\}$ is an experimentally controlled parameter. The rectification function σ_α is ReLU for $\alpha < N$ and logistic for $\alpha = N$.

2.2.3 TRAINING AND TESTING

Networks were implemented in Pytorch and trained with the Adam optimizer using smooth L_1 loss. Training conditions varied along three experimentally controlled parameters: (1) *size*: N_{hddn} for multiclassifiers, and N_{chnl} for multiautoencoders (2) *regularization* via L_2 penalty with scaled coefficient $\eta \in \{0, 0.0005\}$, (3) *maximum arity*, meaning the maximum n such that the network receives training samples of arity n. This parameter, denoted A_{max}, takes values in $\{1, 2\}$. Thus, for example, networks trained with $A_{\text{max}} = 2$ receive training data for all four 1-tasks and both 2-tasks. Each minibatch contains 96 samples, and these samples are divided evenly between tasks (interleaved

training). Each network was trained for one epoch on the MNIST training set (Deng, 2012). Networks trained with maximum arity 1 are called *serial trained*, and networks with maximum arity 2 are called *parallel trained*.

For each N_{hddn} (respectively, N_{chnl}) we generate a set of 20 random initial weights for the corresponding classifier (respectively, autoencoder) network architecture. These sets provided the initial conditions for a set of 20 networks each to be trained for the the 2^3 possible combinations of training parameters. Networks trained with $A_{\text{max}} = 2$ which failed to achieve a mean testing loss below 0.05 are excluded from the study, as are all corresponding networks with the same initial conditions (thus, fixing a size parameter N_{hddn} or N_{chnl}, each network in the 4 training conditions specified by penalty and training can be directly compared to another network with identical initial conditions in the other 3 conditions). This procedure produced a total of 14 and 18 networks per condition, respectively, for classifiers with $N_{\text{hddn}} = 10$ and $N_{\text{hddn}} = 100$, and 18 and 13 networks per condition, respectively, with $N_{\text{chnl}} = 5$ and $N_{\text{chnl}} = 30$.

For each network we evaluated each of the following diagnostic statistics. All statistics were generated with holdout data from the MNIST test set.

Neural encoding We model the activity pattern in layer L_α after the regression equation 1, where

1. $D = \mathbf{C}$, the set of feasible task combinations defined in equation 2.
2. $f = (f_0, f_1)$ is the tuple of feature values in input dimensions $\mathbf{F}_0$ and $\mathbf{F}_1$.

In particular, the contribution of feature dimension $\mathbf{F}_i$ to this pattern in context $c \in \mathbf{C}$ is modeled by $\beta_c(f_i)$, where β_c is obtained by least-squares estimate. We define the linear transformation β_c to be the (least squares estimated) *neural encoding function*, *neural representation*, or simply the *representation* of feature dimension $\mathbf{F}_i$ in context c.

Crosstalk (CT_1, CT_2) quantifies the proportion of information that "leaks" into a response dimension from an unintended source. Concretely, it is the coefficient of determination in the linear input to layer $L_{N,j}$ regressed against the feature values in feature dimension $\mathbf{F}_{\neg i}$ while the network performs the 2-task $\{(i, j), (\neg i, \neg j)\}$ (here $\neg i \in \{0, 1\}$ denotes the index such that $\neg i \neq i$). This quantity computed separately for test samples where the network is asked to 1-task, CT_1 versus 2-task, CT_2. Results are averaged within task condition.

Signal degradation (SD) measures the degree to which "true" signal remains discernable in each response dimension, regardless of degradation or interference. This quantity must be measured independently of CT, since equal levels of crosstalk can effect different changes on decobability of a signal under different conditions (just as an equal number of extra credit points may or may not influence a student's a final grade, depending on how near they stand to a margin). Specifically, we seek to determine not how how well or poorly an individual unit u in output layer L_N actually does, but how *any classifier could* do, given the information provided to u from hidden layer L_{N-1} in the form of linear input. Since our higher objective is to identify structural characteristics of hidden representations that preclude signal decoding at the output layer, we apply a metric which, if anything, overestimates the decodability of signal at this level. Concretely, we train an SVM classifier to optimally separate trials where output unit u ought to take 0 versus 1 values, given the linear input passed to that unit via projection map p_N. Measure SD is the mean squared error of this classifier. This coefficient is evaluated separately for each output unit and each of the two 2-tasking conditions in the test data. Due to the computational cost of fitting one classifier for each output unit in the autoenoders, we randomly subsample 100 units each for these networks.

Subsampled magnitude of interaction effect (MIE) approximates the overall difference between representations of a feature dimension f_i in two different contexts, c and c'. It is formalized as the Frobenius norm of the matrix difference $||[\beta^i_c] - [\beta^i_{c'}]||$, where $[\beta^i_c]$ denotes the array of regression coefficients for linear transform β^i_c. In the special case where f_i has two levels, this quantity coincides with the Euclidean norm of the interaction effect[1] between predictor variables f_i and c. When the number of levels in a feature dimension is very large, as is the case, for example, with the MNIST autoencoder task environment, such coefficient matrices can become difficult and computationally expensive to estimate. We therefore adopt the heuristic strategy of applying the metric to subsamples of two feature values each per input dimension, and averaging over subsamples. Data for the

[1]This alternate interpretation has also remarked in (Rigotti et al., 2013) and elsewhere.

regression is drawn from test samples where the network must 2-task (trials where it 1-tasks are excluded). Results reported in Table 1 are for layer L_{N-1}, for each network.

Hidden variance (V_H) refers to the sum of the mean squared deviation in the in layer L_{N-1} sampled during 2-tasking. This statistic functions as a rough estimate of the effective "size" of the point cloud, which can then be used as a basis for comparison with other measures of size, such as MIE.

2.3 PREDICTIONS

The analytic framework in §1 predicts two distinct representation strategies for networks trained with and without arity-2 training samples. On the one hand, attractive carrots (such as learning speed-ups and improved generalization) are thought to encourage network agents to use a single representation of feature dimension $\mathbf{F}_i$ ($i = 0, 1$) for both action mappings $R_{i,0}$ and $R_{i,1}$. On the other hand, a formidable stick (performance loss due to cross talk), is thought to discourage such practice when executing multitasks of arity ≥ 2, namely $\mathcal{M}_{\{(i,0),(\neg i,1)\}}$ and $\mathcal{M}_{\{(i,1),(\neg i,0)\}}$.

As a result, we predicted that networks exposed only to 1-tasks during training would favor the "Swiss army kniff" strategy of sharing a single representation of $\mathbf{F}_i$ across action mappings $R_{i,0}$ and $R_{i,1}$, while networks exposed to arity-2 training samples would develop separate, dedicated representations for each. These qualitative predictions lead to several concrete hypotheses concerning the test statistics in §2.2.3.

Each of the following effects was predicted to hold *controlling for extraneous independent variables*.

- **H.1** In parallel-trained networks, crosstalk will account for a relatively low proportion of hidden-layer variance in both arity-1 and arity-2 test samples ($\text{CT}_1, \text{CT}_2 \leq 0.1$). *Rationale:* Such networks must limit or eliminate crosstalk in order to reduce loss during training.
- **H.2** In serial-trained networks, crosstalk will account for a relatively high proportion of variance in arity-2 test samples ($\text{CT}_2 \geq 0.3$), and a relatively low proportion of variance in arity-1 test samples ($\text{CT}_1 \leq 0.1$). *Rationale:* Even though representations may be shared across action mappings, a serial-trained network can (and indeed must) suppress the activity of task-irrelevant feature representations to avoid cross-talk while executing tasks of arity 1. Such suppression cannot take place while 2-tasking, however, so crosstalk should be higher in this case.
- **H.3** For serial-trained networks, regularization (realized either by L_2 penalty or by reducing the number of hidden units or channels) will increase the difference CT_2. *Rationale:* Regularization fosters low-dimensional representations and greater generalizability, the hallmarks of representation sharing. We therefore predict that regularization will reinforce sharing, hence crosstalk.
- **H.4** Networks with higher CT_2 will have higher signal degradation SD. *Rationale:* Since feature variables are uncorrelated, crosstalk in the linear input to units in L_N should erode the ability of the network to decode true signal. The relationship between CT_2 and SD is far from direct, however, as noted in §2.2.3, and we predict no precise numeric relationship between the two, beyond this coarse pairwise comparison.
- **H.5** Arity-2 crosstalk will be lower in parallel-trained networks ($\text{CT}_2^{\text{prll}}$) than in serial-trained networks (CT_2^{srl}). Concretely, $\text{CT}_2^{\text{prll}}/\text{CT}_2^{\text{srl}} \leq 1/2$. *Rationale:* Hypotheses **H.1** and **H.2** make relatively conservative absolute predictions about the proportion of variance explained by crosstalk since CT_2 exerts only an indirect influence on training loss, mediated by SD and other factors. However, in relative terms $\text{CT}_2^{\text{prll}}$ should be significantly larger than CT_2^{srl}.
- **H.6** Networks with high arity-2 crosstalk ($\text{CT}_2 \geq 0.3$) will have relatively high objective loss on arity-2 test samples. However, networks with low CT_2 may have high objective loss, also. *Rationale:* By design, signal degradation SD imposes a hard upper bound on 2-tasking performance. However, exogenous factors may drive loss higher. This was a common phenomenon in pilot studies modeled off the networks of (Musslick et al., 2016) and (Musslick et al., 2017).
- **H.7** Networks with higher MIE will have lower CT_2 and lower SD. *Rationale:* This hypothesis rests on three ideas: (i) MIE measures dissimilarity between feature representations in

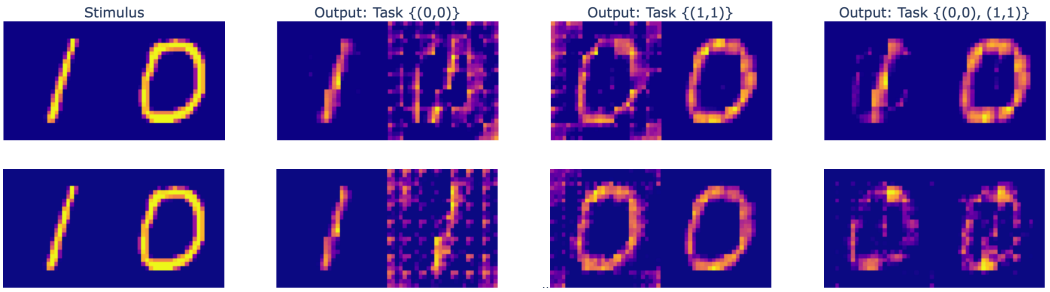

Figure 2: Values projected from hidden layer L_N to the output layer (prior to combination with values projected from the control layer, and logistic rectification) in two of the trained convolutional autoencoders. *Top row*: a parallel-trained network. *Bottom row*: a serial-trained network. *Left*: Input to the network. *Center left*: projection for task $\{(0,0)\}$, *Center right*: projection for task $\{(1,1)\}$, *Right*: projection for task $\{(0,0),(1,1)\}$. Each 28x28 projection is normalized by shifting and rounding-up extreme negative values; in particular, the left and right sides of each 28x56 image may have separate scaling coefficients. In general, task-irrelevant output dimensions took much lower values than task-relevant output dimensions.

different contexts, (ii) networks are "motivated" to deploy similar representations across different tasks when possible, and (iii) one of few motivating factors to depart from this paradigm is to reduce CT_2, and with it SD.

The numerical upper and lower bounds specified above are highly approximate; their purpose is primarily to convey coarse, order-of magnitude estimates based on the experimenters experience with similar models.

3 Results and Discussion

Experimental results appear in Table 1. Due to the large number of variables concerned, individual p-values are not reported, only mean differences. However, in most cases significance can be determined by cursory inspection of Table 1. The salient trends are as follows:

Signal degradation in serial-trained networks is higher than in parallel-trained networks. This holds true across training conditions. In the case of multiclassifiers, best-case decoding error in serial-trained networks is more than twice that of parallel-trained networks.

Crosstalk is highest in serial-trained networks tested on parallel processing tasks. In all but two cases (specifically, the two unregularized training conditions for serial-trained networks), variation in the task-irrelevant feature dimension accounts for over 45% of variance in the task-relevant response dimension. Neither serial-training nor parallel-testing is independently sufficient to produce this effect; one must combine the two. This is consistent with the idea that networks inhibit representations which are not currently of use (in part because activation of these representations is liable to produce crosstalk). See Figure efig:MNISTimages.

Representations overlap more in serial-trained networks than in parallel-trained networks. The ratio of MIE to V_H indicates that the multiaffine model fits these data well, in general.

Regularization accentuates the distinction between serial and parallel trained networks. This is consistent with the view of regularization as a tool which discourages overfitting and favor generalization, abstraction, and transferability. Under such pressures one expects a network to adopt shared representations where possible. It is particularly pronounced in the classifier networks, where regularization produces very small interaction effects (MIE), especially when compared with the variance of the ambient point cloud (V_H).

Exogenous inputs mask the effects of signal degradation. This point is most pronounced in the time course of training for classification networks shown in Figure 3. The salient feature of this plot is that serial-trained networks without regularization achieve a relatively low degree of cross talk; it

Model	Size	N_{smpl}	Reg	Arity	MIE	Loss 2-task	SD	CT_2	CT_1	V_H
Par./Mag.	10	15	1	serial	0.535± 0.19	0.167± 0.008	0.325± 0.028	0.457± 0.029	0.122± 0.14	4.24± 2.04
Par./Mag.	10	15	1	parallel	3.93± 1.27	0.065± 0.014	0.142± 0.047	0.144± 0.113	0.077± 0.072	5.49± 1.82
Par./Mag.	10	15	0	serial	4.13± 2.68	0.175± 0.021	0.252± 0.066	0.325± 0.116	0.069± 0.032	26.3± 8.19
Par./Mag.	10	15	0	parallel	9.26± 1.73	0.052± 0.007	0.117± 0.018	0.11± 0.041	0.065± 0.037	24.4± 7.6
Par./Mag.	100	20	1	serial	0.681± 0.318	0.17± 0.009	0.322± 0.022	0.465± 0.027	0.104± 0.1	5.72± 2.73
Par./Mag.	100	20	1	parallel	5.77± 0.538	0.056± 0.006	0.119± 0.013	0.151± 0.046	0.062± 0.033	8.47± 2.89
Par./Mag.	100	20	0	serial	21.2± 7.11	0.136± 0.042	0.203± 0.087	0.207± 0.087	0.157± 0.095	265.0± 102.0
Par./Mag.	100	20	0	parallel	32.2± 5.14	0.048± 0.007	0.111± 0.025	0.153± 0.07	0.095± 0.054	292.0± 81.5
Autoenc.	5	18	1	serial	8.03± 3.83	0.083± 0.009	0.192± 0.015	0.49± 0.033	0.046± 0.096	443.0± 174.0
Autoenc.	5	18	1	parallel	15.2± 5.04	0.037± 0.005	0.134± 0.019	0.195± 0.122	0.051± 0.018	347.0± 93.4
Autoenc.	5	18	0	serial	17.1± 4.9	0.083± 0.013	0.176± 0.015	0.49± 0.008	0.062± 0.154	1680.0± 521.0
Autoenc.	5	18	0	parallel	25.6± 6.3	0.034± 0.005	0.129± 0.02	0.239± 0.093	0.029± 0.009	935.0± 217.0
Autoenc.	30	13	1	serial	5.08± 3.28	0.079± 0.008	0.194± 0.017	0.48± 0.028	0.129± 0.167	291.0± 120.0
Autoenc.	30	13	1	parallel	22.7± 3.07	0.025± 0.004	0.098± 0.012	0.144± 0.021	0.056± 0.008	452.0± 138.0
Autoenc.	30	13	0	serial	11.9± 4.92	0.105± 0.014	0.201± 0.019	0.484± 0.014	0.327± 0.219	3950.0± 3150.0
Autoenc.	30	13	0	parallel	42.3± 6.98	0.022± 0.001	0.085± 0.009	0.152± 0.013	0.037± 0.007	1780.0± 359.0

Table 1: Network statistics: mean and standard deviation. Size refers to N_{chnl} in autoencoder networks, and to N_{hddn} in parity/magnitude classifiers. For each network size, 20 sets of initial weights were randomly generated; this set was then used to initialize 20 networks in each of four different training groups (corresponding to four blocks of rows in the table) in order to examine effect of training condition while controlling for initial conditions. Networks which failed to achieve testing loss below a set threshold were excluded (this criterion was applied only to 1-tasks for serial-trained networks, and to both 1- and 2-tasks for parallel-trained networks, consistent with training), leaving a sample size of $13 \leq N_{smpl} \leq 20$ networks in each condition. *Arity* indicates maximum arity of training samples; serial-trained networks were trained only on 1-tasks, while parallel-trained networks were trained on of cardinality 1 and 2 (interleaved). A value of 1 under *Reg* indicates use of L_2 penalty, while 0 indicates no penalty. *Loss 2-task* refers to average loss on held-out test data. Variables MIE, SD, CT_1, CT_2, and V_H are described in §2.2.3. All measures of hidden-layer activity refer to the penultimate network layer, L_{N-1}.

remains relatively closer to to the curve for parallel-trained networks, both with and without regularization, that to regularized serial-trained networks. This fact is reflected in relatively low levels of signal degradation, consistent with the proposed connection between the two. However, the test loss experience by the network for both regularized and unregularized serial-trained networks is similar. The explanation for the divergence in these two trends lies with exogenous input from the control layer to the output layer. In serial-task training scenarios, there is no penalty to inhibiting task-irrelevant response dimensions when any single task in undertaken. When two tasks are attempted simultaneously, the automated inhibitory effects from each can suppress the other.

MIE detects difference in crosstalk, signal degradation. This is true both as seen in Table 1 and in Figures 3 and 4. In particular, ratio of MIE in single versus parallel trained conditions remains approximately proportionate to that between parallel-trained and serial-trained crosstalk, even in the the case of unregularized serial-trained classifiers which deviate from the norm and achieve low levels of crosstalk and signal degradation. The relation admits a constant offset:, in particular, the red curves in the lefthand plot of Figure 4 lie vertically above the blue, while in all cases to the right corresponding curves lie directly atop one another.

Results overall conform to the qualitative and numerical predictions **H.1**-**H.7**, with two significant exceptions in feed-forward multiclassifier networks. First, the effects in these networks, while sizeable, were often smaller than numerically predicted (**H.1**, **H.2**, **H.5**). In most cases this, this numerical inaccuracy can be explained by underestimation of the ability of networks to "cope" with CT_2 signal interference. Second, serial-trained feedforward multiclassifiers trained without L_2-regularization failed to realize the tell-tail characteristics of shared representation; in particular, they had significantly lower CT_2 and SD. However, this was the exception that proved the rule, as these networks also had uncharacteristically low SD. These exceptional cases also illustrate the importance of distinguishing SD from objective loss in assessing the performance implications of crosstalk, since objective loss in these networks in fact conforms to overall predictions (presumably due signal passed from the control units directly to the output layer).

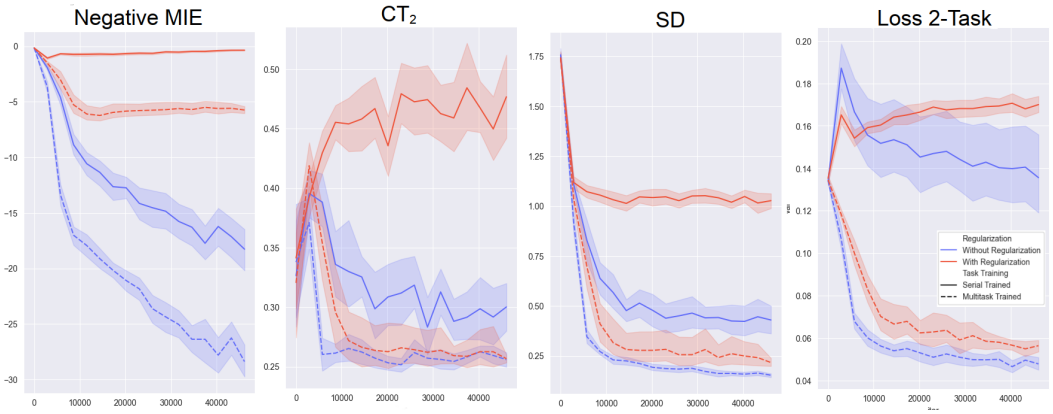

Figure 3: Network statistics (MIE, CT_2, SD, objective loss) over the time course of training in multiclassifier networks. Units of x-axis are number of training samples. Objective loss is measured on held-out test data with samples of arity 2.

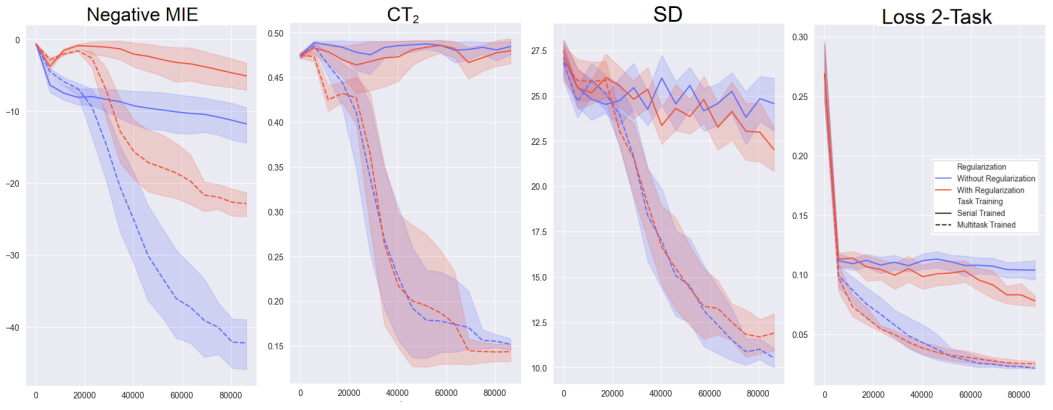

Figure 4: Network statistics (MIE, CT_2, SD, objective loss) over the time course of training in autoencoder networks. Units of x-axis are number of training samples. Objective loss is measured on held-out test data with samples of arity 2.

4 CONCLUSION

Formal accounts of neural encoding of feature representations are basic to explainable models in artificial intelligence and cognitive science generally. In the present work, we applied an elementary model based on limited assumptions of linearity (Rigotti et al., 2013) to evaluate theoretical accounts of parallel processing constraints in learning agents. To do so, we trained neural network architectures inspired by (Musslick et al., 2017; Bernardi et al., 2020) to perform multiple tasks with natural language (MNIST) data. Some networks were trained to perform multiple tasks simultaneously/in parallel, while others were allowed to switch tasks in a purely serial fashion. The parallel processing capacity of these models, measure in terms of the accuracy with which they were able to perform multiple tasks simultaneously, successfully reproduced trends seen both in human and in prior network models. In this controlled setting, the proposed model succeeded in establishing a direct numerical link between structure of neural feature representations and crosstalk on the level of individual unit activations. This link is noteworthy both from machine learning and neuroscience perspectives, as it relies exclusively on properties of the system that can be observed from activity patterns, without reference to connection weights. One consequence of this relationship is the formalization of quantitative local-to-global principle between functional connectivity of feature representations, on the one hand, and macro-scale system properties on the other. Implications for more advanced multitasking agents, both human and machine, are investigated in ongoing work.

REFERENCES

Noga Alon, Daniel Reichman, Igor Shinkar, Tal Wagner, Sebastian Musslick, Jonathan D Cohen, Tom Griffiths, Kayhan Ozcimder, et al. A graph-theoretic approach to multitasking. In *NIPS Proceedings*, pages 2097–2106, 2017.

Silvia Bernardi, Marcus K Benna, Mattia Rigotti, Jérôme Munuera, Stefano Fusi, and C Daniel Salzman. The geometry of abstraction in the hippocampus and prefrontal cortex. *Cell*, 2020.

Rich Caruana. Multitask learning. *Machine learning*, 28(1):41–75, 1997.

Sueyeon Chung, Daniel D. Lee, and Haim Sompolinsky. Classification and Geometry of General Perceptual Manifolds. *Physical Review X*, 8(3):031003, jul 2018. ISSN 21603308. doi: 10.1103/PhysRevX.8.031003.

Jonathan D Cohen, Kevin Dunbar, and James L McClelland. On the control of automatic processes: a parallel distributed processing account of the stroop effect. *Psychol. Rev.*, 97(3):332–361, 1990.

Uri Cohen, Sue Yeon Chung, Daniel D. Lee, and Haim Sompolinsky. Separability and geometry of object manifolds in deep neural networks. *Nature Communications*, 11(1):1–13, dec 2020. ISSN 20411723. doi: 10.1038/s41467-020-14578-5.

Li Deng. The MNIST database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012. ISSN 10535888. doi: 10.1109/MSP.2012.2211477.

Nikolaus Kriegeskorte and Rogier A Kievit. Representational geometry: integrating cognition, computation, and the brain. *Trends in cognitive sciences*, 17(8):401–412, 2013.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

Michael Lesnick, Sebastian Musslick, Biswadip Dey, and Jonathan D Cohen. A formal framework for cognitive models of multitasking. 2020.

Gordon Logan and Robert Gordon. Executive control of attention in dual-task situations. *Psychological review*, 108:393–434, 05 2001. doi: 10.1037//0033-295X.108.2.393.

Andreas Maurer, Massimiliano Pontil, and Bernardino Romera-Paredes. The benefit of multitask representation learning. *The Journal of Machine Learning Research*, 17(1):2853–2884, 2016.

James L McClelland, David E Rumelhart, PDP Research Group, et al. Parallel distributed processing. *Explorations in the Microstructure of Cognition*, 2:216–271, 1986.

Sebastian Musslick, Biswadip Dey, Kayhan Ozcimder, Mostafa Patwary, Theodore Wilkie, and Jonathan D Cohen. Controlled vs. Automatic Processing: A Graph-Theoretic Approach to the Analysis of Serial vs. Parallel Processing in Neural Network Architectures, 2016.

Sebastian Musslick, Andrew M Saxe, Kayhan Ozcimder, Biswadip Dey, Greg Henselman, and J.D. Cohen. Multitasking capability versus learning efficiency in neural network architectures. In *Cognitive Science Society*, 2017.

Harold Pashler. Dual-task interference in simple tasks: Data and theory. *Psychological Bulletin*, 116(2):220–244, 1994. ISSN 0033-2909. doi: 10.1037/0033-2909.116.2.220. URL `/record/1994-43838-001`.

S. Ravi, S. Musslick, M. Hamin, T. Willke, and J. D. Cohen. Navigating the tradeoff between multi-task learning and learning to multitask in deep neural networks. in preparation.

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, 2013.

J. T. Townsend and M. J. Wenger. A theory of interactive parallel processing: new capacity measures and predictions for a response time inequality series. *Psychological review*, 111(4):1003, 2004.

A. M. Treisman. The binding problem. *Current opinion in neurobiology*, 6(2):171–178, 1996.

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.

ACKNOWLEDGMENTS

This work is supported by the Swartz Foundation and the Templeton Foundation.