On the Symmetries of Deep Learning Models and their Internal Representations

Anonymous Author(s)
Affiliation
Address
email

Abstract

Symmetry has been a fundamental tool in the exploration of a broad range of complex systems. In machine learning, symmetry has been explored in both models and data. In this paper we seek to connect the symmetries arising from the architecture of a family of models with the symmetries of that family's internal representation of data. We do this by calculating a set of fundamental symmetry groups, which we call the *intertwiner groups* of the model. Each of these arises from a particular nonlinear layer of the model and different nonlinearities result in different symmetry groups. These groups change the weights of a model in such a way that the underlying function that the model represents remains constant but the internal representations of data inside the model may change. We connect intertwiner groups to a model's internal representations of data through a range of experiments that probe similarities between hidden states across models with the same architecture. Our work suggests that the symmetries of a network are propagated into the symmetries in that network's representation of data, providing us with a better understanding of how architecture affects the learning and prediction process. Finally, we speculate that for ReLU networks, the intertwiner groups may provide a justification for the common practice of concentrating model interpretability exploration on the activation basis in hidden layers rather than arbitrary linear combinations thereof.

1 Introduction

Symmetry provides an important path to understanding across a range of disciplines. This principle is well-established in mathematics and physics, where it has been a fundamental tool (e.g., Noether's Theorem [Noe18]). Symmetry has also been brought to bear on deep learning problems from a number of directions. There is, for example, a rich research thread that studies symmetries in data types that can be used to inform model architectures. The most famous examples of this are standard convolutional neural networks which encode the translation invariance of many types of semantic content in natural images into a network's architecture. In this paper, we focus on connections between two other types of symmetry associated with deep learning models: the symmetries in the learnable parameters of the model and the symmetries across different models' internal representation of the same data.

The first of these directions of research starts with the observation that in modern neural networks there exist models with different weights that behave identically on all possible input. We show that at least some of these equivalent models arise because of symmetries intrinsic to the nonlinearities of the network. We call these groups of symmetries, each of which is attached to a particular type of nonlinear layer σ of dimension n, the *intertwiner groups* G_{σ_n} of the model. These intertwiner groups come with a natural action on network weights for which the realization map to function

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

space of [JGH18] is invariant (proposition 3.4). As such they provide a unifying framework in which to discuss well-known weight space symmetries such as permutation of neurons [Bre+19] and scale invariance properties of ReLU and batch-norm [IS15; NH10].

Next, we tie our intertwiner groups to the symmetries between different model's internal representations of the same data. We do this through a range of experiments that we describe below; each builds on a significant recent advance in the field.

Neural stitching with intertwiner groups: The work of [BNB21; Csi+21; LV15] demonstrated that one can take two trained neural networks, say A and B, with the same architecture but trained from different randomly initialized weights, and connect the early layers of network A to the later layers of network B with a simple "stitching" layer and achieve minimal loss in prediction accuracy. This was taken as evidence of the similarity of strong model's representations of data. Though the original experiments use a fully connected linear layer to stitch, we provide theoretical evidence in theorem 4.2 that much less is needed. Indeed, we show that the intertwiner group (which has far fewer parameters in general) is the minimal viable stitching layer to preserve accuracy. We conduct experiments stitching networks at ReLU activation layers with the stitching layer restricted to elements of the group G_{ReLU}. We show that one can stitch CNNs on CIFAR10 [Kri09] with only elements of G_{ReLU} incurring less than $\approx 10\%$ accuracy penalty at most activation layers. This is surprisingly close to the losses found when one allows for a much more expressive linear layer to be used to stitch two networks together.

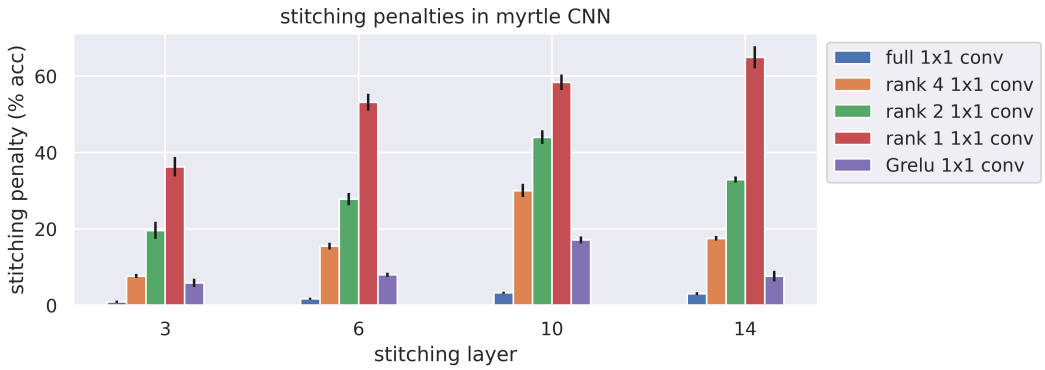
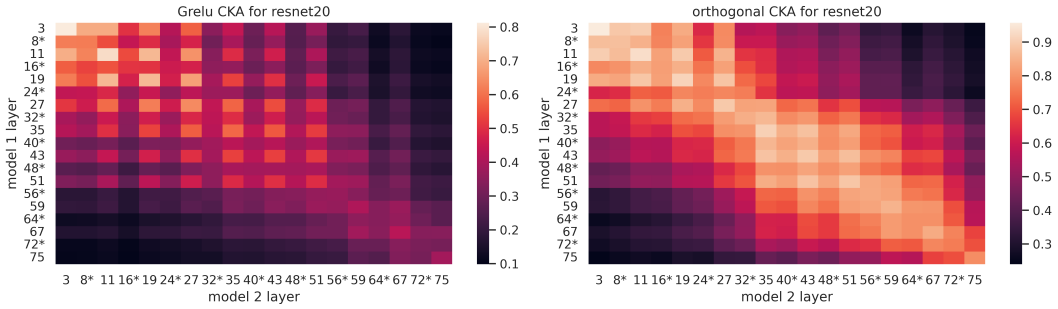


Figure 1: Full, reduced rank, and G_{ReLU} 1-by-1 convolution stitching penalties (4.3) for two independently initialized Myrtle CNNs [Pag18] on CIFAR10 . Confidence intervals were taken over 5 independent runs. The accuracy of the independently initialized models was 90.5 ± 0.2 %.

Representation dissimilarity metrics for G_{ReLU}: In section 5 we present two statistical dissimilarity measures for ReLU-activated hidden features in different networks, say A and B, that are maximized when the hidden features differ by a transformation from G_{ReLU} (that is, permutations and independent scaling of variables). These metrics are analogous to the metrics CKA [Kor+19] and SVCCA [Rag+17], which are invariant to orthogonal transformations, and are maximized when both hidden features differ only up to orthogonal rotation and isotropic scaling. We compare and contrast these metrics with their orthogonal counterparts, as well as with stitching experiment results.

Impact of activation functions on interpretability: An intriguing finding from [Bau+17, §3.2] was that individual neurons are more interpretable than random linear combinations of neurons. Our results on intertwiner groups (theorem 3.3) predict that this is a particular feature of ReLU networks. Indeed, we show in section 6 that in the absence of ReLU activations interpretability does not decrease when one moves from individual neurons to linear combinations of neurons. This result suggests that intertwiner groups provide a theoretical justification for the explainable AI community's focus on individual neuron activations, rather than linear combinations thereof [Erh+09; Na+19; Yos+15; ZF14; Zho+14], but that this justification is only valid for layers with certain activation functions.

Taken together, our experiments provide evidence that a network's symmetries (realized through intertwiner groups) propagate down to symmetries of a model's internal representation of data. Since understanding how different models process the same data is a fundamental goal in fields such as



(a) (b)

Figure 2: G_{ReLU}-CKA and orthogonal CKA for two independently initialized ResNet20s trained on CIFAR10. Layers marked with '*' occur inside residual blocks (remark 3.6). Results averaged over 5 independent runs of the experiment.

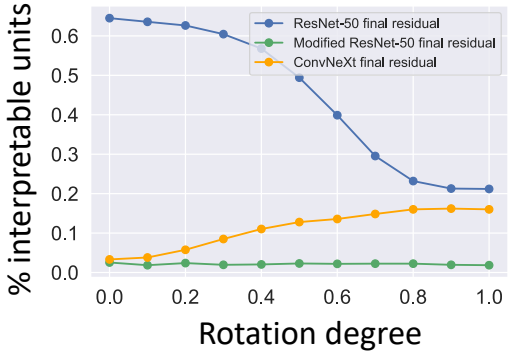


Figure 3: Fraction of network dissection interpretable units under rotations of the representation basis for a ResNet-50, as well as a modified ResNet-50 and ConvNeXt model (both without an activation function on the residual output). We note that the models without residual activation functions also have far fewer concept covering units for a given basis.

explainable AI and the safety of deep learning systems, we hope that our results will provide an additional lens with which to grapple with these problems.

2 Related Work

The research on symmetries of neural networks is extensive, especially when one considers that symmetry considerations are a frequently used tool even when they are not the primary focus of a given paper, hence we aim to provide a representative sample knowing it will be incomplete. [Bre+19; FB17; GBC16; Yi+19] study the effect of weight space symmetries on the loss landscape. On the other hand, [BMC15; GBC16; Kun+21; Men+19] study the effect of weight space symmetries on training dynamics, while [GBC16; RK20] show that weight space symmetries pose an obstruction to model identifiability.

Neural stitching was introduced as a means of comparing learned representations between networks in [LV15]. In [BNB21] it was shown to have intriguing connections with the "Anna Kareninna " (high performance models share similar internal representations of data) and "more is better" (stitching later layers of a weak model to early layers of a model trained with more data/parameters/epochs can improve performance) phenomena. [Csi+21] considered constrained stitching layers by restricting the rank of the stitching matrix or by introducing an ℓ_1 sparsity penalty — our methods are distinct in that we explicitly optimize over the intertwiner group for ReLU nonlinearities (permutations and

scalings). Both [BNB21; Csi+21] compare their stitching results with statistical dissimilarity metrics such as CKA. One of our dissimilarity metrics in section 5 is a close relative of the permutation Procrustes distance introduced in [Wil+22], and also closely related to the orthogonal Procrustes distance considered in [DDS21]. The other is a variant of CKA [Kor+19] in which a certain dot product $\sum_{i=1}^{d} x_i y_i$ is replaced by the operation $\max\{x_1 \cdot y_1, \ldots, x_d \cdot y_d\}$.

Approaches to deep learning interpretability often assume that the activation basis is special [Erh+09; Na+19; Yos+15; ZF14; Zho+14]. Studying individual neurons rather than linear combinations of neurons significantly reduces the complexity of low-level interpretability approaches to the understanding of neural networks [Elh+21]. In tension with this, many different projections of hidden layer activations appear to be semantically coherent [Sze+13]. However, Bau et al. found evidence that the representations closer to the coordinate basis align more with human concepts than random unitary transformations.

3 The Symmetries of Nonlinearities

Let $\mathrm{Mat}_{n_1,n_0}(\mathbb{R})$ be the algebra of all $n_0 \times n_1$ real matrices and $GL_n(\mathbb{R})$ be the group of all invertible $n \times n$ matrices. Let $\sigma : \mathbb{R} \to \mathbb{R}$ be a continuous function. For any $n \in \mathbb{N}$, we can build a *nonlinearity* σ_n from $\mathbb{R}^n$ to $\mathbb{R}^n$ by applying σ coordinatewise, i.e., $\sigma_n(x_1, \ldots, x_n) = (\sigma(x_1), \ldots, \sigma(x_n))$. Let $\ell_i : \mathbb{R}^{n_{i-1}} \to \mathbb{R}^{n_i}$ be the composition of an affine layer and a nonlinear layer, so that $\ell_i(x) := \sigma_{n_i}(W_i x + b_i)$ if $1 \leq i < k$, and $\ell_k(x) := W_k x + b_k$. Here $W_i \in \mathrm{Mat}_{n_i,n_{i-1}}(\mathbb{R})$ and $b_i \in \mathbb{R}^{n_i}$ are the weights and bias of the layer respectively. We define $f : \mathbb{R}^{n_0} \to \mathbb{R}^{n_k}$ to be the neural network $f = \ell_k \circ \cdots \circ \ell_1$. For each $1 \leq i < k-1$ we can then decompose f as $f = f_{>i} \circ f_{\leq i}$ where

$$f_{\leq i} = \ell_i \circ \cdots \circ \ell_1 \quad \text{and} \quad f_{>i} = \ell_k \circ \cdots \circ \ell_{i+1}.$$

We define

$$W := (W_i, b_i \,|\, i = 1, \ldots, k) \quad \text{and} \quad \mathcal{W} := \prod_{i=1}^{k} (\mathrm{Mat}_{n_i,n_{i-1}}(\mathbb{R}) \times \mathbb{R}^{n_i})$$

where the former is the collection of all weights of f and the latter is the space of all possible weights for a given architecture. When we want to emphasize the dependence of f on weights W, we write $f(-, W)$ (and similarly $f_{\leq i}(-, W), f_{>i}(-, W)$).

One of the topics this work will consider is vector space bases for f's hidden spaces $\mathbb{R}^{n_i}$, for $1 \leq i \leq k-1$. We will investigate the legitimacy of analyzing features $f_{\leq i}(D)$ for dataset $D \subset \mathbb{R}^{n_0}$ with respect to the *activation basis* for $\mathbb{R}^{n_i}$ which is simply the usual coordinate basis, $e_1, \ldots, e_{n_i}$ where $e_j = [\delta_{j\ell}]_{\ell=1}^{n_i}$ is naturally parameterized by individual neuron activations. Note that $\mathbb{R}^{n_i}$ has an infinite number of other possible bases that could be chosen.

3.1 Intertwiner Groups

For any $0 \leq i < k$, elements of $GL_{n_i}(\mathbb{R})$ can be applied to the hidden activation space $\mathbb{R}^{n_i}$ both before and after the nonlinear layer σ_{n_i}. We define

$$G_{\sigma_{n_i}} := \{A \in GL_{n_i}(\mathbb{R}) \,|\, \text{ there exists a } B \in GL_{n_i}(\mathbb{R}) \text{ such that } \sigma_{n_i} \circ A = B \circ \sigma_{n_i}\}.$$

Informally, we can understand $G_{\sigma_{n_i}}$ to be the set of all invertible linear transformations whose action on $\mathbb{R}^{n_i}$ prior to the nonlinear layer σ_{n_i} has an equivalent invertible transformation after σ_{n_i}. This is a generalization of the common procedure of understanding a function by understanding those operators that commute with it. For any $A \in GL_{n_i}(\mathbb{R})$, we can write $\sigma(A)$ for the $n_i \times n_i$ matrix formed by applying σ to all entries in A.

Lemma 3.1. *Suppose $\sigma(I_n)$ is invertible and for each $A \in GL_n(\mathbb{R})$ define $\phi_\sigma(A) = \sigma(A)\sigma(I_n)^{-1}$. Then G_{σ_n} is a group, $\phi_\sigma : G_{\sigma_n} \to GL_n(\mathbb{R})$ is a homomorphism and $\sigma_n \circ A = \phi_\sigma(A) \circ \sigma_n$.*

We defer all proofs to appendix E. We include concrete examples of σ for small n_i there as well.

Definition 3.2. When the hypotheses of lemma 3.1 are satisfied (namely, $\sigma(I_n)$ is invertible) we call G_{σ_n} the **intertwiner group of the activation** σ_n. We denote the image of the homomorphism ϕ_σ as $\phi_\sigma(G_{\sigma_n})$.

Activation	G_{σ_n}	$\phi_\sigma(A)$
$\sigma(x) = x$ (identity)	$GL_n(\mathbb{R})$	A
$\sigma(x) = \frac{e^x}{1+e^x}$	Σ_n	A
$\sigma(x) = \mathrm{ReLU}(x)$	Matrices PD, where D has positive entries	A
$\sigma(x) = \mathrm{LeakyReLU}(x)$	Same as ReLU as long as negative slope $\neq 1$	A
$\sigma(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}$ (RBF)	Matrices PD, where D has entries in $\{\pm 1\}$	$\mathrm{abs}(A)$
$\sigma(x) = x^d$ (polynomial)	Matrices PD, where D has non-zero entries	$A^{\odot d}$

Figure 4: Explicit descriptions of G_{σ_n} and ϕ_σ for six different activations. Here $P \in \Sigma_n$ is a permutation matrix, D is a diagonal matrix, abs denotes the operation of taking the absolute value of every entry of A, and $A^{\odot d}$ denotes the process of taking the dth power of every entry of A.

The intertwiner group G_{σ_n} and ϕ_σ are concretely described for a range of activations in fig. 4 — the last two examples motivate the generality of definition 3.2. Note also that in both of those cases $A \mapsto \phi_\sigma(A)$ is *not* a homomorphism on all of $GL_n(\mathbb{R})$, but *is* a homomorphism when restricted to the appropriate subgroup G_{σ_n}. Details related to the calculations can be found in appendix E.1.

The following theorem shows that the activation basis is intimately related to the intertwiner group of ReLU: G_{ReLU} admits a natural group-theoretic characterization in terms of the rays spanned by the activation basis, and dually the rays spanned by the activation basis can be recovered from G_{ReLU}. While both its statement and proof are elementary, our interest in this theorem lies in the question of whether it could *potentially* provide theoretical justification for focusing model interpretation studies on individual activations. We investigate this question further in section 6.

Theorem 3.3. *The group G_{ReLU_n} is precisely the stabilizer of the set of rays $\{\mathbb{R}_{\geq 0} e_i \subset \mathbb{R}^n | i = 1, \ldots, n\}$. Moreover if $\mathbb{R}_{\geq 0} v_1, \ldots, \mathbb{R}_{\geq 0} v_N \subseteq \mathbb{R}^n$ is a finite set of rays stabilized by G_{ReLU}, then for each $v_i = [v_{i1}, \ldots, v_{in}]^T$, it must be that $v_{ij} = 0$ for all but one $j \in \{1, \ldots, n\}$. Equivalently up to multiplication by a positive scalar every v_i is of the form $\pm e_j$ for some j.*

3.2 Weight space symmetries

The intertwiner group is also a natural way to describe the weight space symmetries of a neural network. We denote by $\mathcal{F} \subseteq C(\mathbb{R}^{n_0}, \mathbb{R}^{n_k})$ the space of continuous functions that can be described by a network with the same architecture as f. As described in [JGH18] there is a **realization map** $\Phi : \mathcal{W} \to \mathcal{F}$ mapping weights $W \in \mathcal{W}$ to the associated function $f \in \mathcal{F}$. Φ arises because there are generally multiple sets of weights that yield the same function. We will show that Φ is invariant with respect to an action of the intertwiner groups on $\mathcal{W}$ so that intertwiner groups form a set of "built-in" weight space symmetries of f. This result, which encompasses phenomena including permutation symmetries of hidden neurons, is well known in many particular cases (e.g., [GBC16, §8.2.2], [Bre+19, §3], [FB17, §2], [Men+19, §3], [RK20, §3, A]). From Proposition 3.4 we can also derive corollaries regarding symmetries of the loss landscape — these are included in appendix E.4.

Proposition 3.4. *Suppose $A_i \in G_{\sigma_{n_i}}$ for $1 \leq i \leq k-1$, and let*

$$W' = (A_1 W_1, A_1 b_1, A_2 W_2 \phi_\sigma(A_1^{-1}), A_2 b_2, \ldots, W_k \phi_\sigma(A_{k-1}^{-1}), b_k)$$

Then, as functions, for each m

$$f_{\leq m}(x, W) = A \circ f_{\leq m}(x, W') \text{ and } f_{>m}(x, W') = f_{>m}(x, W) \circ \phi_\sigma(A^{-1}) \tag{3.5}$$

In particular, $f(x, W') = f(x, W)$ for all $x \in \mathbb{R}^{n_0}$. Equivalently, we have $\Phi(W') = \Phi(W) \in \mathcal{F}$.

Remark 3.6. The conclusion of this theorem no longer holds if the architecture of f contains residual connections and the layer m occurs inside a residual block. We see empirical evidence consistent with this failure in figs. 2, 7 and 8 below.

4 Intertwining group symmetries and model stitching

In this section we provide evidence that some of the differences between distinct model's internal representations can be explained in terms of symmetries encoded by intertwiner groups. We do this

using the stitching framework from [BNB21; Csi+21]. We start by reviewing the concept of network stitching.

Suppose $f, \tilde{f}$ are two networks as in section 3, with weights $W, \tilde{W}$ respectively. For any $1 \leq l \leq k-1$ we may form a **stitched network** $S(f, \tilde{f}, l, \varphi) : \mathbb{R}^{n_0} \to \mathbb{R}^{n_k}$ defined in the notation of section 3 by $S(f, \tilde{f}, l, \varphi) = \tilde{f}_{>l} \circ \varphi \circ f_{\leq i}$ – here $\varphi : \mathbb{R}^{n_l} \to \mathbb{R}^{n_l}$ is a **stitching layer**. In a typical stitching experiment one freezes the weights of networks f and $\tilde{f}$ respectively where f and $\tilde{f}$ were trained from different initializations, constrains φ to some simple function class $\mathcal{S}$ (e.g., affine maps in [BNB21]), and trains $S(f, \tilde{f}, l, \varphi)$ to optimize φ alone. The final validation accuracy $\operatorname{Acc} S(f, \tilde{f}, l, \varphi)$ of $S(f, \tilde{f}, l, \varphi)$ is then considered a measure of similarity (or lack therof) of the internal representations of f and $\tilde{f}$ in $\mathbb{R}^{n_l}$ — in this framework the situation

$$\operatorname{Acc} S(f, \tilde{f}, l, \varphi) \approx \operatorname{Acc} f, \operatorname{Acc} \tilde{f} \tag{4.1}$$

corresponds to high similarity since the hidden representations of model f and $\tilde{f}$ could be related by a transformation $\mathcal{S}$.

Recall that even though the networks $f(W)$ and $f(W')$ may be *equal as functions*, their hidden representations need not be the same (an example of this is given in appendix C). Our next result shows that in the case where f and $\tilde{f}$ do only differ up to an element of $G_{\sigma_{n_l}}$, eq. (4.1) is achievable even when the stitching function class $\mathcal{S}$ is restricted down to elements of $G_{\sigma_{n_l}}$ or $\phi_\sigma(G_{\sigma_{n_l}})$ (see definition 3.2).

Theorem 4.2. *Suppose* $\tilde{W} = (A_1 W_1, A_1 b_1, A_2 W_2 \phi_\sigma(A_1^{-1}), A_2 b_2, \ldots, W_k \phi_\sigma(A_k^{-1}), b_k)$ *where* $A_i \in G_{\sigma_{n_i}}$ *for all* i*. Then eq. (4.1) is achievable with equality* if *the stitching function class* $\mathcal{S}$ *containing* φ *contains* $G_{\sigma_{n_l}}$ *or* $\phi_\sigma(G_{\sigma_{n_l}})$*.*

Motivated by theorem 4.2, we attempt to stitch various networks at ReLU activation layers using the group G_{ReLU} described in Figure 4. Every matrix $A \in G_{\mathrm{ReLU}}$ can be written as PD, where P is a permutation matrix and D is diagonal with positive diagonal entries — hence optimization over G_{ReLU} requires optimizing over permutation matrices. We use the well-known convex relaxation of permutation matrices to doubly stochastic matrices. See Section D.1 in the Appendix for details related to our optimization procedure.

Figure 1 gives the difference between the average test error of Myrtle CNN [Pag18] networks [1] f and $\tilde{f}$ and the network $S(f, \tilde{f}, l, \varphi)$,

$$\operatorname{Avg}(\operatorname{Acc}(f), \operatorname{Acc}(\tilde{f})) - \operatorname{Acc}(S(f, \tilde{f}, l, \varphi)) \tag{4.3}$$

where $S(f, \tilde{f}, \varphi)$ was stitched together at layer l via a stitching transformations φ that was either optimized over all affine transformations, reduced rank affine transformations as in [Csi+21] or transformations restricted to G_{ReLU}. We consider only the ReLU activation layers, as these are the only layers where the theory of section 3 applies, and we only act on the channel tensor dimension — in practice, this is accomplished by means of 1-by-1 convolution operations. In particular, with G_{ReLU} we are *only permuting and scaling channels*. Lower values indicate that the stitching layer was sufficient to translate between the internal representation of f at layer l and the internal representation of $\tilde{f}$.

We find that when we are allowed to learn a stitching layer over arbitrary affine transformations of channels, we can nearly achieve the accuracy of the original models. When we are only allow to optimize over G_{ReLU} there is an appreciable increase in test error difference (consistent with findings in [Csi+21; Li+15; Wan+18] discussed in section 2, and also consistent with observations that hidden features of neural networks exhibit distributed representations and polysemanticism [Ola+20]). Nonetheless, that $S(f, \tilde{f}, l, \varphi)$ is able get within less than 10% of the accuracy of f and $\tilde{f}$ in all but one layer suggests that elements of G_{ReLU} can account for a substantial amount of the variation in the internal representations of independently trained networks. We include the reduced rank transformations as the dimension of their parameter spaces is greater than that of G_{ReLU}, and yet they incur significantly higher stitching penalties. If n_l is the number of channels, we have

[1] This is a simple 5-layer CNN, with no residual connections; we describe its architecture and training hyperparameters in further detail in appendix D.

$\dim G_{\mathrm{ReLU}_{n_l}} = n_l$ whereas the dimension of rank r transformations is $2n_l \cdot r - r^2$ (hence greater than $\dim G_{\mathrm{ReLU}_{n_l}}$ even for $r = 1$). Finally, the stitching penalties incurred when using any layer other than 1-by-1 convolution with a rank 1 matrix all follow similar general patterns: they increase up to the third activation layer, then decrease at the final activation layer.

Further stitching results on the ResNet20 architecture can be found in appendix D.2. This includes results where we vary the negative slop of the LeakyReLU and find similar stitching penalties up to but not including a slope of 1, providing further evidence of the importance of the intertwiner group.

5 Dissimilarity metrics for the intertwiner group of ReLU

Stitching penalties can be viewed as task oriented measures of hidden feature dissimilarity. From a different perspective, we can consider raw statistical measures of hidden feature dissimilarity. In the design of measures of dissimilarity, a crucial choice is the group of transformations under which the dissimilarity measure is invariant. For example, Centered Kernel Alignment (CKA) [Kor+19] is invariant with respect to orthogonal transformation and isotropic scaling. We ask for a statistical dissimilarity metric μ on datasets $X, Y \in \mathbb{R}^{N \times d}$ with the properties that (0) $0 \leq \mu(X, Y) \leq 1$, (i) (G_{ReLU}-*Invariance*) If $A, B \in G_{\mathrm{ReLU}_d}$ and $v, w \in \mathbb{R}^d$ then $\mu(XA + \mathbf{1}v^T, YB + \mathbf{1}w^T) = \mu(X, Y)$, and (ii) (*Alignment Property*) $\mu(X, Y) = 1$ if $(*)$ $Y = XA + \mathbf{1}v^T$ for some $A \in G_{\mathrm{ReLU}_d}, v \in \mathbb{R}^d$. To motivate this question, we note that given such a metric μ, one can detect if X and Y do *not* differ by an element of G_{ReLU} by checking if $\mu(X, Y) < 1$. Our basic tool for ensuring (i) is the next lemma.

Lemma 5.1. *Suppose $\mu(XA, YB) = \mu(X, Y)$ if A, B are* either *positive diagonal matrices or permutation matrices. Then, (i) holds.*

In effect, this allows us to divide the columns of X and Y by their norms to achieve invariance to the action of positive diagonal matrices and then apply dissimilarity metrics for the permutation group such as those presented in [Wil+22]. Ensuring (ii) seems to require case-by-case analysis to determine an appropriate normalization constant.

Definition 5.2 (G_{ReLU}-Procrustes)**.** Let $D_X = \mathrm{diag}(|X_{[:,i]}|)$ and $D_Y = \mathrm{diag}(|Y_{[:,i]}|)$. Assuming these are invertible, let $\tilde{X} = XD_X^{-1}$ and $\tilde{Y} = YD_Y^{-1}$. Let δ be the permutation Procrustes distance between $\tilde{X}, \tilde{Y}$, defined by

$$\delta := \min_{P \in \Sigma_d} |\tilde{X} - \tilde{Y}P|$$

(as pointed out in [Wil+22] this can be computed via the linear sum assignment problem). Then the G_{ReLU}**-Procrustes distance** is

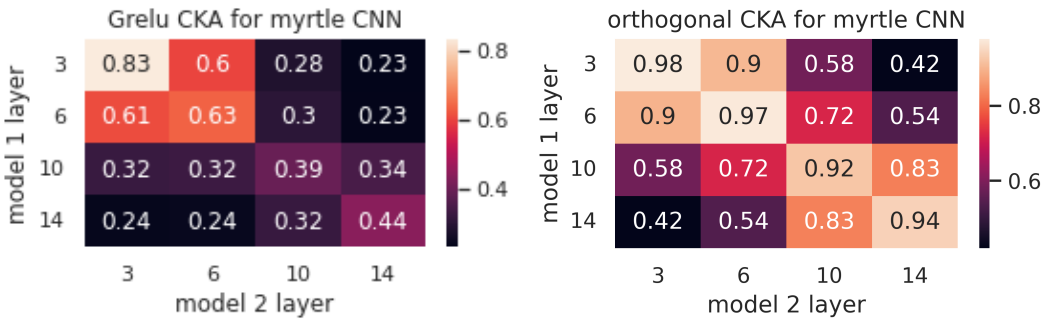
$$\mu_{\text{Procrustes}}(X, Y) := 1 - \frac{\delta}{2\sqrt{d}}.$$

The factor of $2\sqrt{d}$ ensures this lies in $[0, 1]$, and equals 1 if (and only if) the condition $*$ of (ii) holds.

	layer 3	layer 6	layer 10	layer 14
G_{ReLU}	0.645 ± 0.017	0.521 ± 0.008	0.447 ± 0.003	0.483 ± 0.002
Orthogonal	0.810 ± 0.006	0.662 ± 0.004	0.530 ± 0.003	0.601 ± 0.001

Figure 5: $\mu_{\text{Procrustes}}$ and orthogonal Procrustes similarities for two independently initialized Myrtle CNNs trained on CIFAR10. Confidence intervals correspond to 5 independent runs of the experiment.

We apply G_{ReLU}-Procrustes and orthogonal Procrustes similarities to 4 different hidden representations from Myrtle CNNs in fig. 5 and ResNet20s in fig. 8, all trained on CIFAR10 [Kri09]. In keeping with the discussion of section 4, we only consider permutations or orthogonal transformations of channels (for details on how this is implemented we refer to appendix D.3). We see that distinct representations register less similarity in terms of G_{ReLU}-Procrustes than they do in terms of orthogonal Procrustes. This makes sense as similarity up to G_{ReLU}-transformation requires a greater degree of absolute similarity between representations than is required of similarity up to orthogonal transformation (the latter being a higher-dimensional group containing all of the permutations in G_{ReLU}).



(a) (b)

Figure 6: G_{ReLU}-CKA and orthogonal CKA for two independently initialized Myrtle CNNs trained on CIFAR10 (results averaged over 5 independent runs of the experiment).

Otherwise patterns in G_{ReLU}-Procrustes similaritiy largely follow those of G_{ReLU}-orthogonal, with similarity between representations decreasing as one progresses through the network, only to increase again in the last layer. This correlates with the stitching penalties of fig. 1, which increase with depth only to decrease in the last layer.

Definition 5.3 (G_{ReLU}-CKA)**.** Assume that X and Y are data matrices that have been centered by subtracting means of rows: $X \leftarrow X - \mathbf{1}\mathbf{1}^T X$ and $Y \leftarrow Y - \mathbf{1}\mathbf{1}^T Y$. Let $\tilde{X} = X D_X^{-1}$ and $\tilde{Y} = Y D_Y^{-1}$. Let $\tilde{x}_1, \ldots, \tilde{x}_N$ be the rows of X, and similarly for Y. Form the matrices $K, L \in \mathbb{R}_{\geq 0}^{N \times N}$ defined by $K_{ij} = \max(\tilde{x}_i \odot \tilde{x}_j)$ and $L_{ij} = \max(\tilde{y}_i \odot \tilde{y}_j)$ where $\odot$ is the Hadamard product. Then the G_{ReLU}-CKA for X and Y is defined as:

$$\mu_{\mathrm{CKA}}(X, Y) := \frac{\mathrm{HSIC}_1(K, L)}{\sqrt{\mathrm{HSIC}_1(K, K)}\sqrt{\mathrm{HSIC}_1(L, L)}}. \tag{5.4}$$

where HSIC_1 is the unbiased form of the Hilbert-Schmidt independence criterion of [NRK21, eq. 3].

Symmetry of the $\max$ function ensures (i), the Cauchy-Schwarz inequality ensures $\mu_{\mathrm{CKA}}(X, Y) \in [0, 1]$, and we claim that $\mu_{\mathrm{CKA}}(X, Y) = 1$ if the condition $*$ of (ii) is met. We do not claim 'if and only if', however we point out the following in lemma E.13: if A is a matrix such that $\max(Ax_1 \odot Ax_2) = \max(x_1 \odot x_2)$ for all $x_1, x_2 \in \mathbb{R}^d$, then A is of the form PD where P is a permutation matrix and D is diagonal with diagonal entries in $\{\pm 1\}$.

As with CKA [Kor+19], this metric makes sense even if X, Y are datasets in $\mathbb{R}^d, \mathbb{R}^{d'}$ respectively with $d \neq d'$. Results for a pair of Myrtle CNNs independently trained on CIFAR10, as well as standard orthogonal CKA for comparison, are shown in fig. 6. Analogous results for ResNet20s are shown in fig. 2. We find that G_{ReLU}-CKA respects basic trends found in their orthogonal counterparts: model layers at the same depth are more similar, early layers are highly similar, and the metric surfaces the block structure of the ResNet in fig. 2: layers inside residual blocks are less similar than those at residual connections. One notable difference for G_{ReLU}-CKA in fig. 2 is that the similarity difference between early and later layers in the orthogonal CKA (discussed for ResNets in [Rag+21]) shown in (b) is less pronounced in (a), and in fact later layers are found to be less similar between runs. We found similar results for stitching in fig. 1.

6 Interpretability of the coordinate basis

In this section we explore the confluence of model interpretability and intertwiner symmetries using *network dissection* from [Bau+17]. Network dissection measures alignment between the individual neurons of a hidden layer and single, pre-defined concepts. The Broden concept dataset, compiled by Bau et al., contains pixel-level annotations for hierarchical concepts including colors, textures, objects, and scenes. For every channel activation, network dissection assesses the binary segmentation performance with every visual concept from Broden. The method first computes the channel activation for every Broden image. The distribution of the activations for the channel is used to binarize the activation (where we threshold by the top 0.5% of all activations for the channel) to

define a segmentation mask for the channel activation which is interpolated to the size of the input image. If the Intersection over Union (IoU) of the activation segmentation mask and a concept mask is high enough (namely where IoU > 0.04), network dissection labels the activation an interpretable detector for the concept.

We use an experiment from [Bau+17] to compare the axis-aligned interpretability of hidden activation layers with and without an activation function. Bau et al. compares the interpretability of individual neurons, measured via network dissection, with that of random orthogonal rotations of neurons. We likewise rotate the hidden layer representations and then measure their interpretability. Using the methodology from [Dia05], we define a random orthogonal transform Q drawn uniformly from $SO(n)$ by using Gram-Schmidt to orthonormalize the normally-distributed $QR = A \in \mathbb{R}^{n^2}$. Like in [Bau+17], we also consider smaller rotations $Q^\alpha \in SO(n)$ where $0 \leq \alpha \leq 1$, where α is chosen to form a minimal geodesic gradually rotating from I to Q. [Bau+17] found that the number of interpretable units decreased away from the activation basis as α increased for layer5 of an AlexNet.

We compare three models trained on ImageNet: a normal ResNet-50, a modified ResNet-50 where we remove the ReLU on the residual outputs (training details in appendix F.1), and a ConvNeXt [Liu+22] analog of the ResNet-50, which also does not have an activation function before the final residual output. We give results in 3, and provide sample unit detection outputs and full concept labels for the figures in appendix F. As was shown in [Bau+17], interpretability decreases as we rotate away from the axis for the normal ResNet-50. On the other hand, with no activation function and therefore no canonical basis, interpretability does not drop with rotation for the modified ResNet-50 and the ConvNeXt. Interestingly, while the number of interpretable units remains constant for the residual output of the modified ResNet-50, for the ConvNeXt model it actually *increases*. We find similar results, where the number of interpretable units increase with rotation, for the convolutional layer inside the residual block for the modified ResNet-50 in fig. 9.

7 Limitations

Our theoretical analysis does not account for standard regularization techniques that are known to have symmetry-breaking effects (for example weight decay reduces scaling symmetry). More generally, we do not account for any implicit regularization of our training algorithms. As illustrated in figs. 1 and 7, stitching with intertwiner groups appears to have significantly more architecture-dependent behaviour than stitching with arbitrary affine transformations (however, since different architectures have different symmetries this is to be expected). Our empirical tests of the dissimilarity metrics in section 5 are limited to what [Kor+19] terms "sanity tests"; in particular we did not perform the specificity, sensitivity and quality tests of [DDS21].

8 Conclusion

In this paper we describe groups of symmetries that arise from the nonlinear layers of a neural network. We calculate these symmetry groups for a number of different types of nonlinearities, and explore their fundamental properties and connection to weight space symmetries. Next, we provide evidence that these symmetries induce symmetries in a network's internal representation of the data that it processes, showing that previous work on the internal representations of neural networks can be naturally adapted to incorporate awareness of the intertwiner groups that we identify. Finally, in the special case where the network in question has ReLU nonlinearities, we find experimental evidence that intertwiner groups justify the special place of the activation basis within interpretable AI research.

References

[Bau+17] David Bau et al. "Network Dissection: Quantifying Interpretability of Deep Visual Representations". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017), pp. 3319–3327.

[BMC15] Vijay Badrinarayanan, Bamdev Mishra, and R. Cipolla. "Understanding Symmetries in Deep Networks". In: *ArXiv* (2015).

[BNB21] Yamini Bansal, Preetum Nakkiran, and Boaz Barak. "Revisiting Model Stitching to Compare Neural Representations". In: *NeurIPS*. 2021.

[Bre+19] Johanni Brea et al. "Weight-Space Symmetry in Deep Networks Gives Rise to Permutation Saddles, Connected by Equal-Loss Valleys across the Loss Landscape". In: (July 5, 2019). arXiv: 1907.02911 [cs, stat]. URL: http://arxiv.org/abs/1907.02911 (visited on 01/12/2022).

[Csi+21] Adrián Csiszárik et al. "Similarity and Matching of Neural Network Representations". In: *Advances in Neural Information Processing Systems*. Ed. by A. Beygelzimer et al. 2021. URL: https://openreview.net/forum?id=aedFIIRRfXr.

[DDS21] Frances Ding, Jean-Stanislas Denain, and J. Steinhardt. "Grounding Representation Similarity with Statistical Testing". In: *ArXiv* (2021).

[Dia05] Persi Diaconis. "What is a random matrix". In: *Notices of the AMS* 52.11 (2005), pp. 1348–1349.

[Elh+21] N Elhage et al. *A mathematical framework for transformer circuits*. 2021.

[Erh+09] Dumitru Erhan et al. "Visualizing higher-layer features of a deep network". In: *University of Montreal* 1341.3 (2009), p. 1.

[FB17] C. Daniel Freeman and Joan Bruna. "Topology and Geometry of Half-Rectified Network Optimization". In: *ArXiv* abs/1611.01540 (2017).

[GBC16] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016.

[IS15] S. Ioffe and Christian Szegedy. "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift". In: *ICML* (2015).

[JGH18] Arthur Jacot, Franck Gabriel, and Clément Hongler. "Neural tangent kernel: convergence and generalization in neural networks (invited paper)". In: *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing* (2018).

[Kor+19] Simon Kornblith et al. "Similarity of Neural Network Representations Revisited". In: *ICML* (2019).

[Kri09] Alex Krizhevsky. *Learning multiple layers of features from tiny images*. Tech. rep. 2009.

[Kun+21] Daniel Kunin et al. *Neural Mechanics: Symmetry and Broken Conservation Laws in Deep Learning Dynamics*. Mar. 29, 2021. arXiv: 2012.04728 [cond-mat, q-bio, stat]. URL: http://arxiv.org/abs/2012.04728 (visited on 01/12/2022).

[Lec+22] Guillaume Leclerc et al. *ffcv*. https://github.com/libffcv/ffcv/. commit 849. 2022.

[Li+15] Yixuan Li et al. "Convergent Learning: Do Different Neural Networks Learn the Same Representations?" In: *FE@NIPS*. 2015.

[Liu+22] Zhuang Liu et al. "A ConvNet for the 2020s". In: *arXiv preprint arXiv:2201.03545* (2022).

[LV15] Karel Lenc and A. Vedaldi. "Understanding Image Representations by Measuring Their Equivariance and Equivalence". In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015). DOI: 10.1109/CVPR.2015.7298701.

[Men+19] Qi Meng et al. "G-SGD: Optimizing ReLU Neural Networks in Its Positively Scale-Invariant Space". In: *ICLR*. 2019.

[Na+19] Seil Na et al. "Discovery of natural language concepts in individual units of cnns". In: *arXiv preprint arXiv:1902.07249* (2019).

[NH10] Vinod Nair and Geoffrey E. Hinton. "Rectified Linear Units Improve Restricted Boltzmann Machines". In: *ICML*. 2010.

[Noe18] E Noether. "Invariante Variationsprobleme". In: 1918, pp. 235–257.

[NRK21] Thao Nguyen, Maithra Raghu, and Simon Kornblith. "Do Wide and Deep Networks Learn the Same Things? Uncovering How Neural Network Representations Vary with Width and Depth". In: *ArXiv* abs/2010.15327 (2021).

[Ola+20] Chris Olah et al. "Zoom in: An introduction to circuits". In: *Distill* 5.3 (2020), e00024–001.

[Pag18] David Page. *How to Train Your ResNet*. Myrtle. Sept. 24, 2018. URL: https://myrtle.ai/learn/how-to-train-your-resnet/ (visited on 05/09/2022).

[PE20] Sebastian Prillo and Julian Martin Eisenschlos. "SoftSort: A Continuous Relaxation for the argsort Operator". In: *ICML*. 2020.

[Rag+17] M. Raghu et al. "SVCCA: Singular Vector Canonical Correlation Analysis for Deep Learning Dynamics and Interpretability". In: *NIPS*. 2017.

[Rag+21] Maithra Raghu et al. "Do vision transformers see like convolutional neural networks?" In: *Advances in Neural Information Processing Systems* 34 (2021).

[RK20] D. Rolnick and Konrad Paul Kording. "Reverse-Engineering Deep ReLU Networks". In: *ICML*. 2020.

[Sze+13] Christian Szegedy et al. "Intriguing properties of neural networks". In: *arXiv preprint arXiv:1312.6199* (2013).

[Vir+20] Pauli Virtanen et al. "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python". In: *Nature Methods* 17 (2020), pp. 261–272. DOI: 10.1038/s41592-019-0686-2.

[Wan+18] Liwei Wang et al. "Towards Understanding Learning Representations: To What Extent Do Different Neural Networks Learn the Same Representation". In: *NeurIPS*. 2018.

[Wil+22] Alex H. Williams et al. "Generalized Shape Metrics on Neural Representations". Jan. 12, 2022. arXiv: 2110.14739 [cs, stat]. URL: http://arxiv.org/abs/2110.14739 (visited on 05/03/2022).

[Yi+19] Mingyang Yi et al. "Positively Scale-Invariant Flatness of ReLU Neural Networks". In: *ArXiv* (2019).

[Yos+15] Jason Yosinski et al. "Understanding neural networks through deep visualization". In: *arXiv preprint arXiv:1506.06579* (2015).

[ZF14] Matthew D Zeiler and Rob Fergus. "Visualizing and understanding convolutional networks". In: *European conference on computer vision*. Springer. 2014, pp. 818–833.

[Zho+14] Bolei Zhou et al. "Object detectors emerge in deep scene cnns". In: *arXiv preprint arXiv:1412.6856* (2014).

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Section 7.
   (c) Did you discuss any potential negative societal impacts of your work? [N/A] This paper is largely focused on the mathematical aspects of deep learning so we do not think there are any immediate negative societal impact to the methods described. From a broader perspective though, we see this work helping to create a more principled groundwork for many explainable AI techniques. We explain why this could have positive societal impacts in Section B.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes] All proofs can be found in appendix E.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? **[TODO]** We are in the process of making code publicly available.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See appendix D.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See appendix D — while we did not keep precise track of CPU/GPU hours, we do specify the hardware used.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes] See Section G.
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]