Uncovering How Neural Network Representations Vary with Width and Depth

Thao Nguyen[*]
Google Research
thaotn@google.com

Maithra Raghu
Google Research
maithra@google.com

Simon Kornblith
Google Research
skornblith@google.com

Abstract

A key factor in the success of deep neural networks is the ability to scale models to improve performance by varying the architecture depth and width. This simple property of neural network design has resulted in highly effective architectures for a variety of tasks. Nevertheless, there is limited understanding of effects of depth and width on the *learned representations*. In this paper, we study this fundamental question. We begin by investigating how varying depth and width affects model internal representations, finding a characteristic *block structure* in the hidden representations of larger capacity (wider or deeper) models. We demonstrate that this block structure arises when model capacity is large relative to the size of the training set, and is indicative of the underlying layers preserving and propagating the dominant principal component of their representations.[1]

1 Introduction

Deep neural network architectures are typically tailored to available computational resources by scaling their width and/or depth. Remarkably, this simple approach to model scaling can result in state-of-the-art networks for both high- and low-resource regimes [20]. Nevertheless, there is limited understanding of how varying these properties affects the final model *beyond* its performance. Investigating this fundamental question is critical, especially with the continually increasing compute resources devoted to designing and training new network architectures. More concretely, we can ask, how do depth and width affect the final learned representations? Do different model architectures also learn different intermediate (hidden layer) features? In this paper, we study these core questions, through detailed analysis of ResNet models with varying depths and widths trained on CIFAR-10 [10], CIFAR-100 and ImageNet [4]. We show that depth/width variations result in distinctive characteristics in the model internal representations. Specifically, our contributions are as follows:

- We apply CKA (centered kernel alignment) to measure the similarity of the hidden representations of different neural network architectures, finding that representations in wide or deep models exhibit a characteristic structure, which we term the *block structure*. We study how the block structure varies across different training runs, and uncover a connection between block structure and model overparametrization — block structure primarily appears in overparameterized models.
- Through further analysis, we find that the block structure corresponds to hidden representations having a single principal component that explains the majority of the variance in the representation, which is preserved and propagated through the corresponding layers. We show that some hidden layers exhibiting the block structure can be pruned with minimal impact on performance.
- Finally, with this insight on the hidden representational structures within a single network, we turn to comparing representations *across* different architectures, finding that models without the block structure show reasonable representation similarity in corresponding layers, but block structure representations are unique to each model.

[*]Work done as a member of the Google AI Residency program.

[1]The full version of this work can be found at https://arxiv.org/pdf/2010.15327.pdf

1st NeurIPS workshop on Interpretable Inductive Biases and Physically Structured Learning (2020), virtual.

Related Work. Neural network models of different depths and widths have been studied through the lens of universal approximation theorems [3, 7, 16, 13, 5, 12] and functional expressivity [21, 18]. However, this line of work only shows that such networks can be constructed, and provides neither a guarantee of learnability nor a characterization of their performance when trained on finite datasets. Other work has studied the behavior of neural networks in the infinite width limit by relating architectures to their corresponding kernels [14, 11, 8]. In contrast to this theoretical work, we attempt to develop empirical understanding of the behavior of practical neural network architectures after training. Previous empirical work has studied the effects of width and depth upon model accuracy in the context of convolutional neural network architecture design, finding that optimal accuracy is typically achieved by balancing width and depth [23, 20]. We instead seek to study the impact of width and depth on network internal representations by applying techniques for measuring similarity of neural network hidden representations [9, 17, 15].

2 Experimental Setup and Background

Our goal is to understand the effects of depth and width on the function learned by the underlying neural network, in a setting representative of models used in practice. We thus train ResNets [6, 23] on standard image classification datasets CIFAR-10, CIFAR-100 and ImageNet. For CIFAR ResNets, we scale networks' depths by increasing the number of blocks in each stage, and scale width by multiplying the number of neurons in each layer by some width multiplier. More details on network design, training parameters, and accuracies of all investigated models, can be found in Appendix B.

We use linear centered kernel alignment [9, 2] to measure similarity between neural network hidden representations. We compute CKA by averaging HSIC scores over k minibatches:

$$\mathrm{CKA} = \frac{\frac{1}{k}\sum_{i=1}^{k}\mathrm{HSIC}_1(\mathbf{X}_i\mathbf{X}_i^\mathsf{T}, \mathbf{Y}_i\mathbf{Y}_i^\mathsf{T})}{\sqrt{\frac{1}{k}\sum_{i=1}^{k}\mathrm{HSIC}_1(\mathbf{X}_i\mathbf{X}_i^\mathsf{T}, \mathbf{X}_i\mathbf{X}_i^\mathsf{T})}\sqrt{\frac{1}{k}\sum_{i=1}^{k}\mathrm{HSIC}_1(\mathbf{Y}_i\mathbf{Y}_i^\mathsf{T}, \mathbf{Y}_i\mathbf{Y}_i^\mathsf{T})}}, \tag{1}$$

where $\mathbf{X}_i \in \mathbb{R}^{n \times p_1}$ and $\mathbf{Y}_i \in \mathbb{R}^{n \times p_2}$ are matrices containing the activations of two layers, one with p_1 neurons and another p_2 neurons, to the same minibatch of n examples sampled without replacement. We compute the HSIC values for CKA in Eq. 1 using an unbiased estimator of CKA [19]:

$$\mathrm{HSIC}_1(\mathbf{K}, \mathbf{L}) = \frac{1}{n(n-3)}\left(\mathrm{tr}(\tilde{\mathbf{K}}\tilde{\mathbf{L}}) + \frac{\mathbf{1}^\mathsf{T}\tilde{\mathbf{K}}\mathbf{1}\mathbf{1}^\mathsf{T}\tilde{\mathbf{L}}\mathbf{1}}{(n-1)(n-2)} - \frac{2}{n-2}\mathbf{1}^\mathsf{T}\tilde{\mathbf{K}}\tilde{\mathbf{L}}\mathbf{1}\right), \tag{2}$$

where $\tilde{\mathbf{K}}$ and $\tilde{\mathbf{L}}$ are obtained by setting the diagonal entries of $\mathbf{K}$ and $\mathbf{L}$ to zero. This minibatch estimator gives the same result as if the entire dataset were used to compute HSIC_1 (see Appendix A).

3 Depth, Width and Model Internal Representations

We begin our study by investigating how the depth and width of a model architecture affects its internal representation structure. How do representations evolve through the hidden layers in different architectures? How similar are different hidden layer representations to each other?

3.1 Internal Representations and the Block Structure

In Figure 1, we show the results of training ResNets of varying depths (top row) and widths (bottom row) on CIFAR-10. For each ResNet, we use CKA to compute the representation similarity of all pairs of layers within the same model. Note that the total number of layers is much greater than the stated depth of the ResNet, as the latter only accounts for the convolutional layers in the network but we include *all* intermediate representations. We can visualize the result as a heatmap, with the x and y axes representing the layers of the network, going from the input layer to the output layer.

The heatmaps start off as showing a checkerboard-like representation similarity structure, which arises because representations after residual connections are more similar to other post-residual representations than representations inside ResNet blocks. As the model gets wider or deeper, we see the emergence of a distinctive *block structure* — a considerable range of hidden layers that have very high representation similarity (seen as a yellow square on the heatmap). This block structure mostly appears in the later layers (the last two stages) of the network. As shown in Appendix Figure D.1, while the exact size and position of the block structure can vary, it is present across all training runs. We observe similar results in networks without residual connections (Appendix Figure C.1).

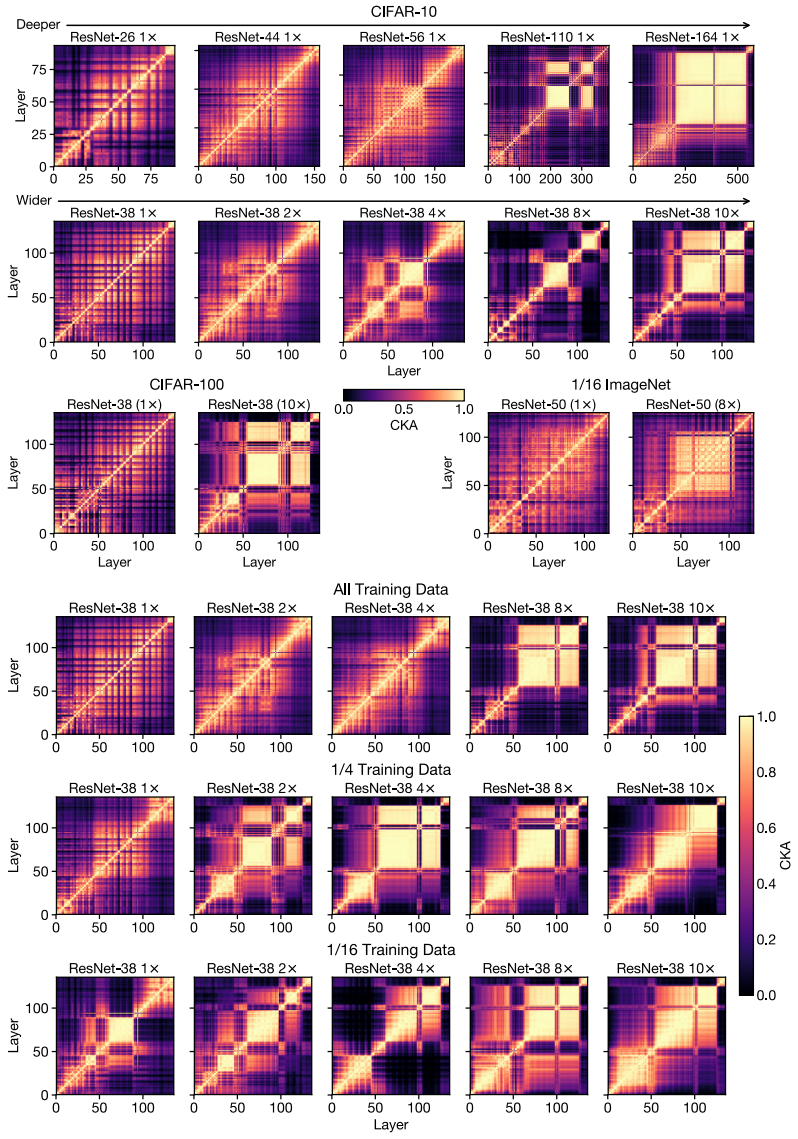

Figure 1: ***Block structure*** **emerges with increasing width or depth.** As we increase the depth or width of neural networks, we see the emergence of a large, contiguous set of layers with very similar representations — the block structure. Each of the panes of the figure computes the CKA similarity between all pairs of layers in a single neural network and plots this as a heatmap, with x and y axes indexing layers. See Appendix Figure C.1 for block structure in wide networks without residual connections.

Figure 2: **Block structure emerges in narrower networks when trained on less data.** We plot CKA similarity heatmaps as we increase network width (going right along each row) and also decrease the dataset size (down each column). As a result of the increased model capacity (with respect to the task) from smaller dataset size, smaller (narrower) models now also exhibit the block structure.

3.2 The Block Structure and Model Overparametrization

Having observed that the block structure emerges as models get deeper and/or wider (Figure 1), we next study whether it is connected to the *absolute* model size, or to the size of the model *relative* to the size of the training data.

The results of this experiment with varying network widths are shown in Figure 2, while the corresponding plot with varying network depths (which supports the same conclusions) can be found in Appendix Figure D.2. Each column of Figure 2 shows the internal representation structure of a fixed architecture as the amount of training data is reduced, and we can clearly see the emergence of the block structure in narrower (lower capacity) networks as less training data is used. Refer to Figures D.3 and D.4 in the Appendix for a similar set of experiments on CIFAR-100. Together, these observations indicate that the block structure in the internal representations arises in models that are heavily overparameterized relative to the training dataset.

4 Probing the Block Structure

In the previous section, we show that wide and/or deep neural networks exhibit a block structure in the CKA heatmaps of their internal representations, and that this block structure arises from the large capacity of the models in relation to the learned task. Nonetheless, there remains a key open question, which this section seeks to answer: what is happening to the neural network representations as they propagate through the block structure?

4.1 The Block Structure and The First Principal Component

For centered matrices of activations $\boldsymbol{X} \in \mathbb{R}^{n \times p_1}$, $\boldsymbol{Y} \in \mathbb{R}^{n \times p_2}$, linear CKA may be written as $\mathrm{CKA}(XX^{\mathrm{T}}, YY^{\mathrm{T}}) = \sum_{i=1}^{p_1} \sum_{j=1}^{p_2} \lambda_X^i \lambda_Y^j \langle \mathbf{u}_X^i, \mathbf{u}_Y^j \rangle^2 / \left(\|\boldsymbol{\lambda}_X\|_2 \|\boldsymbol{\lambda}_Y\|_2 \right)$ where $\boldsymbol{u}_X^i \in \mathbb{R}^n$ and $\boldsymbol{u}_Y^i \in \mathbb{R}^n$ are the i^{th} normalized principal components of $\boldsymbol{X}$ and $\boldsymbol{Y}$ and λ_X^i and λ_Y^i are the corresponding squared singular values [9]. As the fraction of the variance explained by the first principal components approaches 1, CKA reflects the squared alignment between these components $\langle \mathbf{u}_X^1, \mathbf{u}_Y^1 \rangle^2$. We find that, in networks with a visible block structure, the first principal component explains a large fraction of the variance, whereas in networks with no block structure, it does not (Appendix Figure D.5), suggesting that the block structure reflects behavior of the first principal component of the representations.

Figure 3 explores this relationship between the block structure and the first principal components of the corresponding layer representations. In layers that are part of the block structure, the first principal component explains a large proportion of the variance (bottom left), and the cosine similarities between first principal components of pairs of layers (top left) closely resembles CKA (top right). Removing this first principal component nearly eliminates the block structure (bottom right). Together these results demonstrate that the block structure arises from preserving and propagating the first principal component across its constituent layers.

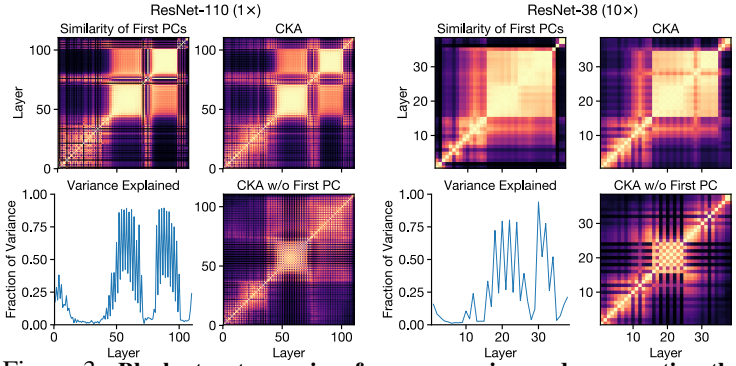

Figure 3: **Block structure arises from preserving and propagating the (dominant) first principal component (PC) of the layer representations.** In each group of plots, CKA of the representations (top right) shows block structure in a deep (left group) or wide (right group) network. Layers in the block structure have a dominant first PC (bottom left). This PC is also preserved throughout the block structure, seen by comparing the squared cosine similarity of the first principal component between layers (top left) to CKA (top right). Removing the first PC from the representations (bottom right), reduces the block structure.

Although layers inside the block structure have representations with high CKA and similar first principal components, each layer nonetheless computes a nonlinear transformation of its input. Appendix Figure D.6 shows that the sparsity of ReLU activations inside and outside of the block structure is similar. Thus, ReLU activations in the block structure are sometimes in the linear regime and sometimes in the saturating regime, just like activations elsewhere in the network.

Additional experiments with linear probes [1] further show that some layers that make up the block structure can be removed with minimal performance loss. Refer to Appendix E for more details.

5 Depth and Width Effects on Representations Across Models

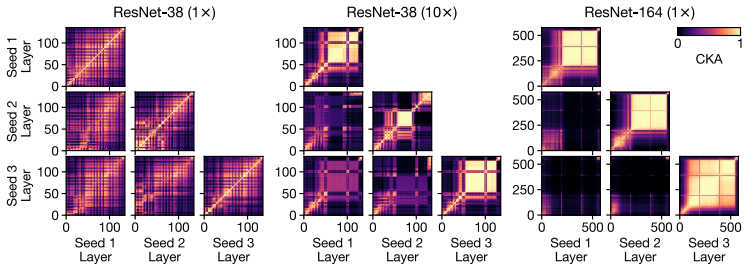

Figure 4: **Representations within "block structure" differ across initializations.** Groups of plots shows CKA between layers of models with the same architecture but different initializations (off the diagonal) or within a single model (on the diagonal). For narrow, shallow models such as ResNet-38 (1×), there is no block structure, and CKA across initializations closely resembles CKA within a single model. For wider (middle) and deeper (right) models, representations within the block structure are highly dissimilar.

We next look at how depth and width affect the hidden representations *across* models. Concretely, are learned representations similar across models of different random initializations? How is this affected as model capacity is changed? Figure 4 illustrates CKA heatmaps for a smaller model (left), wide model (middle) and deep model (right), trained from random initializations. The smaller model does not have the block structure, and repre-

sentations across seeds (off diagonal plots) exhibit the same grid-like similarity structure as within a single model. The wide and deep models show block structure in all their seeds (as seen in plots along the diagonal), and comparisons across seeds (off-diagonal plots) show that while layers not in the block structure exhibit some similarity, the block structure representations are highly dissimilar across models.

6 Conclusion

In this work, we study the effects of width and depth on neural network representations. Through experiments on CIFAR-10, CIFAR-100 and ImageNet, we have demonstrated that as either width or depth increases relative to the size of the dataset, analysis of hidden representations reveals the emergence of a characteristic *block structure* that reflects the similarity of a dominant first principal component, propagated across many hidden layers in the network. Further analysis finds that while the block structure is unique to each model, other learned features are shared across different initializations and architectures, particularly across relative depths of the network. There remain interesting open questions on how the block structure arises through training, and using the insights on network depth and width to inform optimal task-specific model design.

References

[1] G. Alain and Y. Bengio. Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644*, 2016.

[2] C. Cortes, M. Mohri, and A. Rostamizadeh. Algorithms for learning kernels based on centered alignment. *The Journal of Machine Learning Research*, 13(1):795–828, 2012.

[3] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314, 1989.

[4] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[5] B. Hanin and M. Sellke. Approximating continuous functions by relu nets of minimal width. *arXiv preprint arXiv:1710.11278*, 2017.

[6] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[7] K. Hornik. Approximation capabilities of multilayer feedforward networks. *Neural networks*, 4 (2):251–257, 1991.

[8] A. Jacot, F. Gabriel, and C. Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in neural information processing systems*, pages 8571–8580, 2018.

[9] S. Kornblith, M. Norouzi, H. Lee, and G. Hinton. Similarity of neural network representations revisited. In *ICML*, 2019.

[10] A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[11] J. Lee, J. Sohl-dickstein, J. Pennington, R. Novak, S. Schoenholz, and Y. Bahri. Deep neural networks as gaussian processes. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=B1EA-M-0Z`.

[12] H. Lin and S. Jegelka. Resnet with one-neuron hidden layers is a universal approximator. In *Advances in neural information processing systems*, pages 6169–6178, 2018.

[13] Z. Lu, H. Pu, F. Wang, Z. Hu, and L. Wang. The expressive power of neural networks: A view from the width. In *Advances in neural information processing systems*, pages 6231–6239, 2017.

[14] A. G. d. G. Matthews, M. Rowland, J. Hron, R. E. Turner, and Z. Ghahramani. Gaussian process behaviour in wide deep neural networks. *arXiv preprint arXiv:1804.11271*, 2018.

[15] A. Morcos, M. Raghu, and S. Bengio. Insights on representational similarity in neural networks with canonical correlation. In *Advances in Neural Information Processing Systems*, pages 5727–5736, 2018.

[16] A. Pinkus. Approximation theory of the mlp model in neural networks. *Acta numerica*, 8(1): 143–195, 1999.

[17] M. Raghu, J. Gilmer, J. Yosinski, and J. Sohl-Dickstein. Svcca: Singular vector canonical correlation analysis for deep learning dynamics and interpretability. In *Advances in Neural Information Processing Systems*, pages 6076–6085, 2017.

[18] M. Raghu, B. Poole, J. Kleinberg, S. Ganguli, and J. Sohl-Dickstein. On the expressive power of deep neural networks. In *international conference on machine learning*, pages 2847–2854. PMLR, 2017.

[19] L. Song, A. Smola, A. Gretton, J. Bedo, and K. Borgwardt. Feature selection via dependence maximization. *The Journal of Machine Learning Research*, 13(1):1393–1434, 2012.

[20] M. Tan and Q. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pages 6105–6114, 2019.

[21] M. Telgarsky. Representation benefits of deep feedforward networks. *arXiv preprint arXiv:1509.08101*, 2015.

[22] M. Yamada, Y. Umezu, K. Fukumizu, and I. Takeuchi. Post selection inference with kernels. In *International Conference on Artificial Intelligence and Statistics*, pages 152–160. PMLR, 2018.

[23] S. Zagoruyko and N. Komodakis. Wide residual networks. In *British Machine Vision Conference (BMVC)*, pages 87.1–87.12, September 2016.

Appendix

A Minibatch CKA

Our approach of estimating HSIC on minibatches is equivalent to the bagging block HSIC approach of [22]. Below, we prove that the unbiased estimator of HSIC averaged over minibatches converges to the same value as the same estimator applied to the full dataset.

Proposition 1. *Let $\boldsymbol{K} \in \mathbb{R}^{m \times m}$ and $\boldsymbol{L} \in \mathbb{R}^{m \times m}$ be two kernel matrices constructed by applying kernel functions k and l respectively to all pairs of examples in a dataset $\mathcal{D}$. Form c random partitionings p of $\mathcal{D}$ into m/n minibatches b of size n, and let $\tilde{\boldsymbol{K}}^{b,p} \in \mathbb{R}^{n \times n}$ and $\tilde{\boldsymbol{L}}^{b,p} \in \mathbb{R}^{n \times n}$ be kernel matrices constructed by applying kernels k and l to all pairs of examples within each minibatch. Define $U_0 = \mathrm{HSIC}_1(\boldsymbol{K}, \boldsymbol{L})$, the value of HSIC_1 applied to the full dataset, and $\tilde{U}_p = \frac{n}{m} \sum_{b=1}^{m/n} \mathrm{HSIC}_1(\boldsymbol{K}^{b,p}, \boldsymbol{L}^{b,p})$, the average value of HSIC_1 over the minibatches in partitioning (epoch) p. Then $\frac{1}{c} \sum_{p=1}^{c} \tilde{U}_p \xrightarrow{P} U_0$ as $c \to \infty$.*

Proof. Let $\boldsymbol{i}_4^m$ be the set of all 4-tuples of indices between 1 and m where each index occurs exactly once. As proven in Theorem 3 of [19], U_0 is a U-statistic:

$$U_0 = \mathrm{HSIC}_1(\boldsymbol{K}, \boldsymbol{L}) = \frac{(m-4)!}{m!} \sum_{S \in \boldsymbol{i}_4^m} h(K_S, L_S), \tag{3}$$

where $K_{(i,j,q,r)} = (K_{i,j}, K_{i,q}, K_{i,r}, K_{j,q}, K_{j,r}, K_{q,r})$ and the kernel of the U-statistic h is defined in [19]. Let δ_S^b be 1 if the 4-tuple of dataset indices S is selected in minibatch b and 0 otherwise. Then:

$$\tilde{U}_p = \frac{(n-4)!}{n!} \frac{n}{m} \sum_{b=1}^{m/n} \sum_{S \in \boldsymbol{i}_4^m} \delta_S^b h(K_S, L_S). \tag{4}$$

Taking the expectation with respect to $\boldsymbol{\delta}$, and noting that $\boldsymbol{\delta}$ is independent of $h(K_S, L_S)$,

$$\mathbb{E}_{\boldsymbol{\delta}}[\tilde{U}_p] = \frac{(n-4)!}{n!} \frac{n}{m} \sum_{b=1}^{m/n} \sum_{S \in \boldsymbol{i}_4^m} \mathbb{E}_{\boldsymbol{\delta}} \left[\delta_S^b h(K_S, L_S)\right] \tag{5}$$

$$= \frac{(n-4)!}{n!} \frac{n}{m} \sum_{b=1}^{m/n} \sum_{S \in \boldsymbol{i}_4^m} \mathbb{E}_{\boldsymbol{\delta}} \left[\delta_S^b\right] h(K_S, L_S). \tag{6}$$

By symmetry, $\mathbb{E}_{\boldsymbol{\delta}} \left[\delta_S^b\right]$ is the same for all example and batch indices. Specifically, there are $n!/(n-4)!$ 4-tuples that can be formed from each batch and $m!/(m-4)!$ 4-tuples that can be formed from the entire dataset, so the probability that a given 4-tuple is in a given batch is $\mathbb{E}_{\boldsymbol{\delta}} \left[\delta_S^b\right] = (n!/(n-4)!)/(m!/(m-4)!)$. Thus:

$$\mathbb{E}_{\boldsymbol{\delta}}[\tilde{U}_p] = \frac{(m-4)!}{m!} \sum_{S \in \boldsymbol{i}_4^m} h(K_S, L_S) = U_0. \tag{7}$$

The minibatch indicators δ_S^b are either 0 or 1, so their variances and covariances are bounded, and the weighted sum in Eq. 4 has finite variance. Thus, by the law of large numbers, $\frac{1}{c} \sum_{p=1}^{c} \tilde{U}_p \xrightarrow{P} U_0$ as $p \to \infty$. □

B Training Details

Our CIFAR-10 and CIFAR-100 ResNets follow the same architecture as [6, 23]. The network's layers are evenly divided between three stages (feature map sizes), with numbers of channels increasing by a factor of two from one stage to the next. We adjust the network's width and depth by increasing the

number of channels and layers respectively in each stage, following [23]. We train a set of models where we fix the width multiplier of deep networks to 1 and experiment with models of depths 32, 44, 56, 110, 164. On CIFAR-100, the block structure only appears at a greater depth so we also include depths 218 and 224 in our investigation. For wide networks, we examine width multipliers of 1, 2, 4, 8 and 10 and depths of 14, 20, 26, and 38. We use SGD with momentum of 0.9, together with a cosine decay learning rate schedule and batch size of 128, to train each model for 300 epochs. Models are trained with standard CIFAR-10 data augmentation comprising random flips and translations of up to 4 pixels. Each depth and width configuration is trained with 10 different seeds for CKA analysis.

For ImageNet ResNets, ResNet-50 and ResNet-101 architectures differ only by the number of layers in the third (14×14) stage. Thus, for experiments on ImageNet, we scale only the width or depth of layers in this stage. We train for 120 epochs using SGD with momentum of 0.9 and a cosine decay learning rate schedule at a batch size of 256.

For experiments with reduced dataset size, we subsample the training data from the original CIFAR training set by the corresponding proportion, keeping the number of samples for each class the same. All CKA heatmaps are then computed based on the full CIFAR test set.

Table B.1: **Accuracy of examined neural networks on CIFAR-10 and CIFAR-100.**

Depth	Width	CIFAR-10 Test Accuracy (%)	CIFAR-100 Test Accuracy (%)
32	1	93.5	71.2
44	1	94.0	72.0
56	1	94.2	73.3
110	1	94.3	74.0
164	1	94.4	73.9
14	1	92.0	67.8
14	2	94.1	72.9
14	4	95.4	77.0
14	8	95.9	80.0
14	10	96.0	80.2
20	1	92.8	69.4
20	2	94.6	74.4
20	4	95.4	77.6
20	8	96.0	80.2
20	10	95.8	80.8
26	1	93.3	70.5
26	2	94.9	75.8
26	4	95.6	79.3
26	8	95.9	80.9
26	10	95.8	81.0
38	1	93.8	72.3
38	2	95.1	75.9
38	4	95.5	78.6
38	8	95.7	79.8
38	10	95.7	80.5

C Block Structure in a Different Architecture

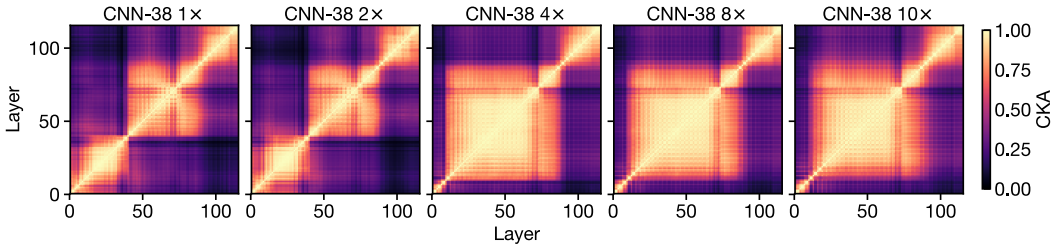

Figure C.1: **Block structure also appears in models without residual connections.** We remove residual connections from existing CIFAR-10 ResNets and plot CKA heatmaps for layers in the resulting architecture after training. Since the lack of residual connections prevents deep networks from performing well on the task, here we only show the representational similarity for models of increasing width. As previously observed in Figure 1, the block structure emerges in higher capacity models.

D Probing the Block Structure

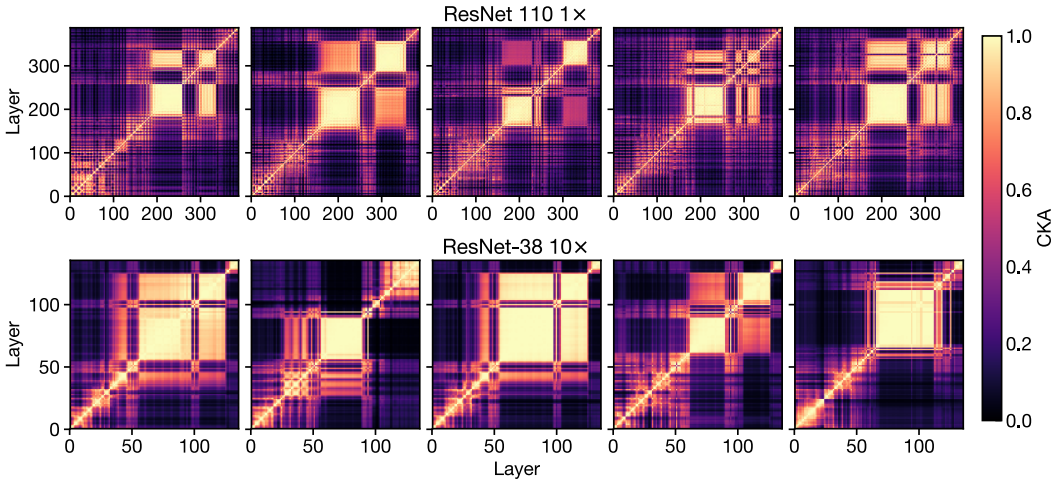

Figure D.1: **Block structure varies across random initializations.** We plot CKA heatmaps as in Figure 1 for 5 random seeds of a deep model (top row) and a wide model (bottom row) trained on CIFAR-10. While the size and position vary, the block structure is clearly visible in all seeds.

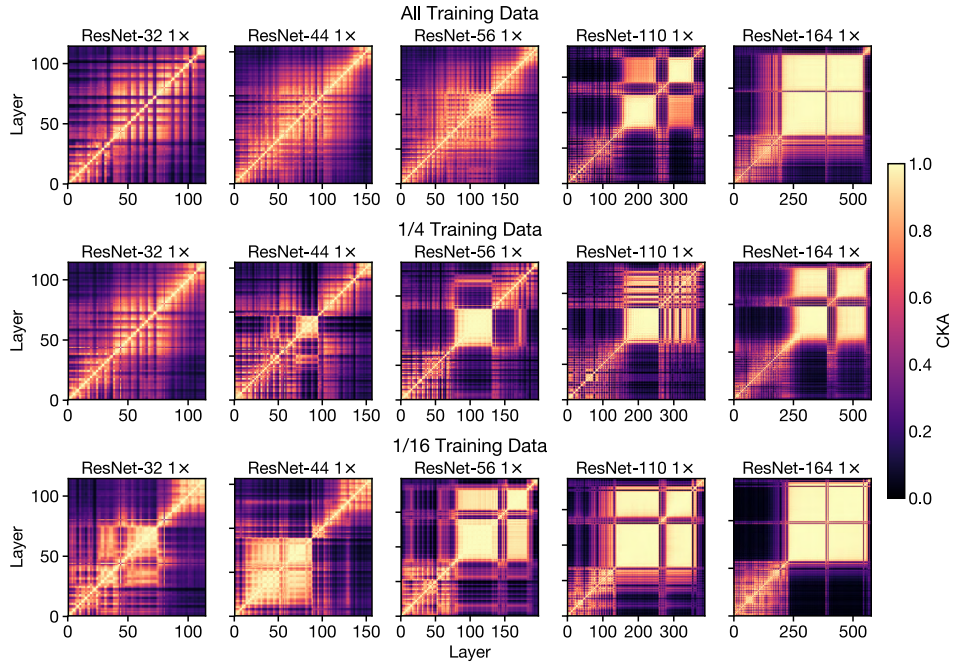

Figure D.2: **Block structure emerges in shallower networks when trained on less data (CIFAR-10).** We plot CKA similarity heatmaps as we increase network depth (going right along each row) and also decrease the size (down each column) of training data. Similar to the observation made in Figure 2, as a result of the increased model capacity (with respect to the task) from smaller dataset size, smaller (shallower) models now also exhibit the block structure.

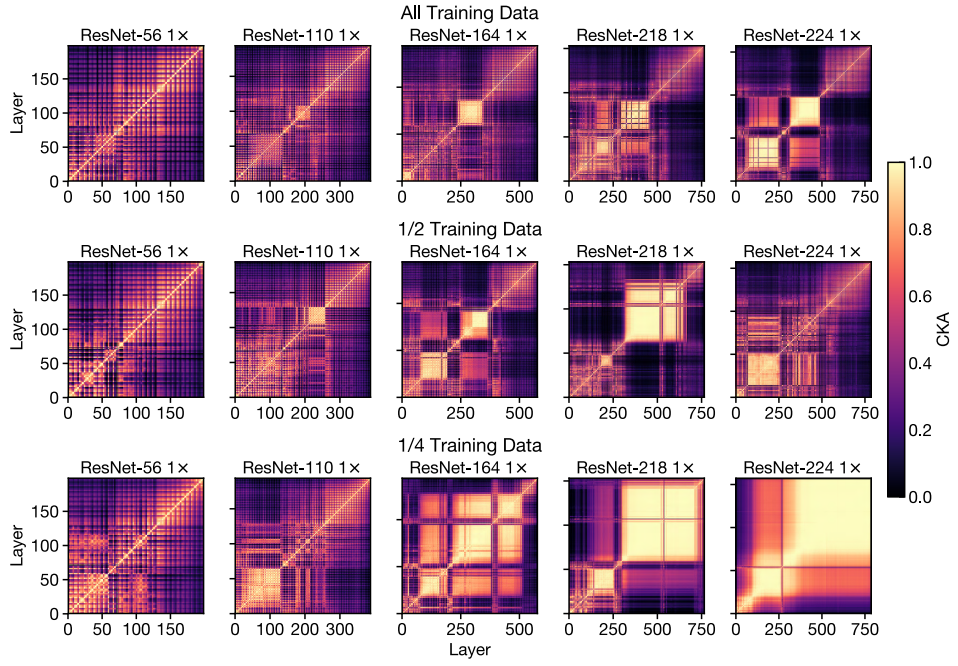

Figure D.3: **Block structure emerges in shallower networks when trained on less data (CIFAR-100).** We plot CKA similarity heatmaps as we increase network depth (going right along each row) and also decrease the size (down each column) of training data. Similar to the observation made in Figure 2, as a result of the increased model capacity (with respect to the task) from smaller dataset size, smaller (shallower) models now also exhibit the block structure.

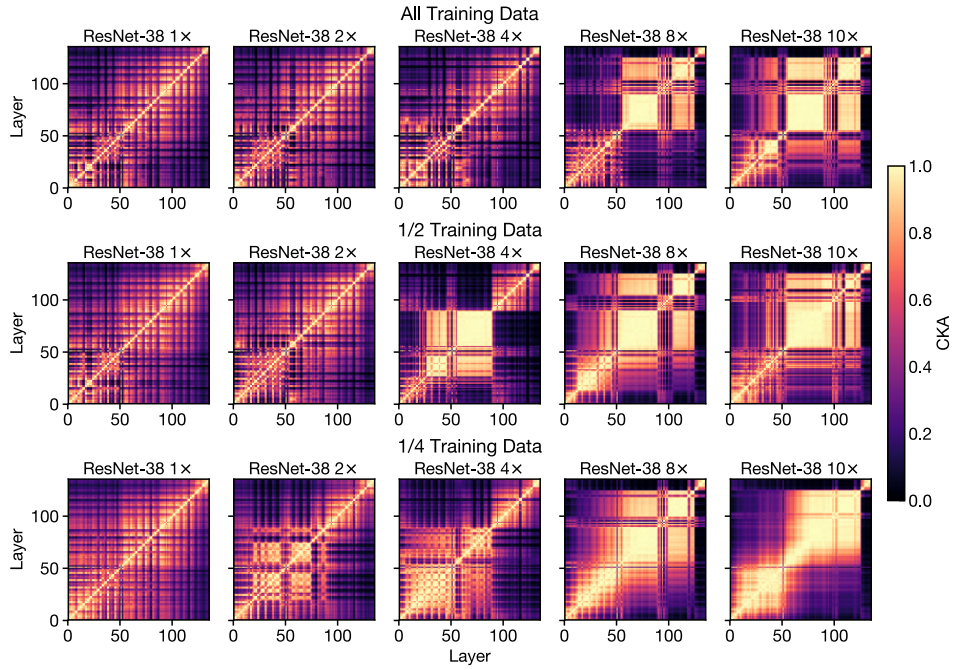

Figure D.4: **Block structure emerges in narrower networks when trained on less data (CIFAR-100).** We plot CKA similarity heatmaps as we increase network width (going right along each row) and also decrease the size (down each column) of training data. Similar to the observation made in Figure 2, as a result of the increased model capacity (with respect to the task) from smaller dataset size, smaller (narrower) models now also exhibit the block structure.

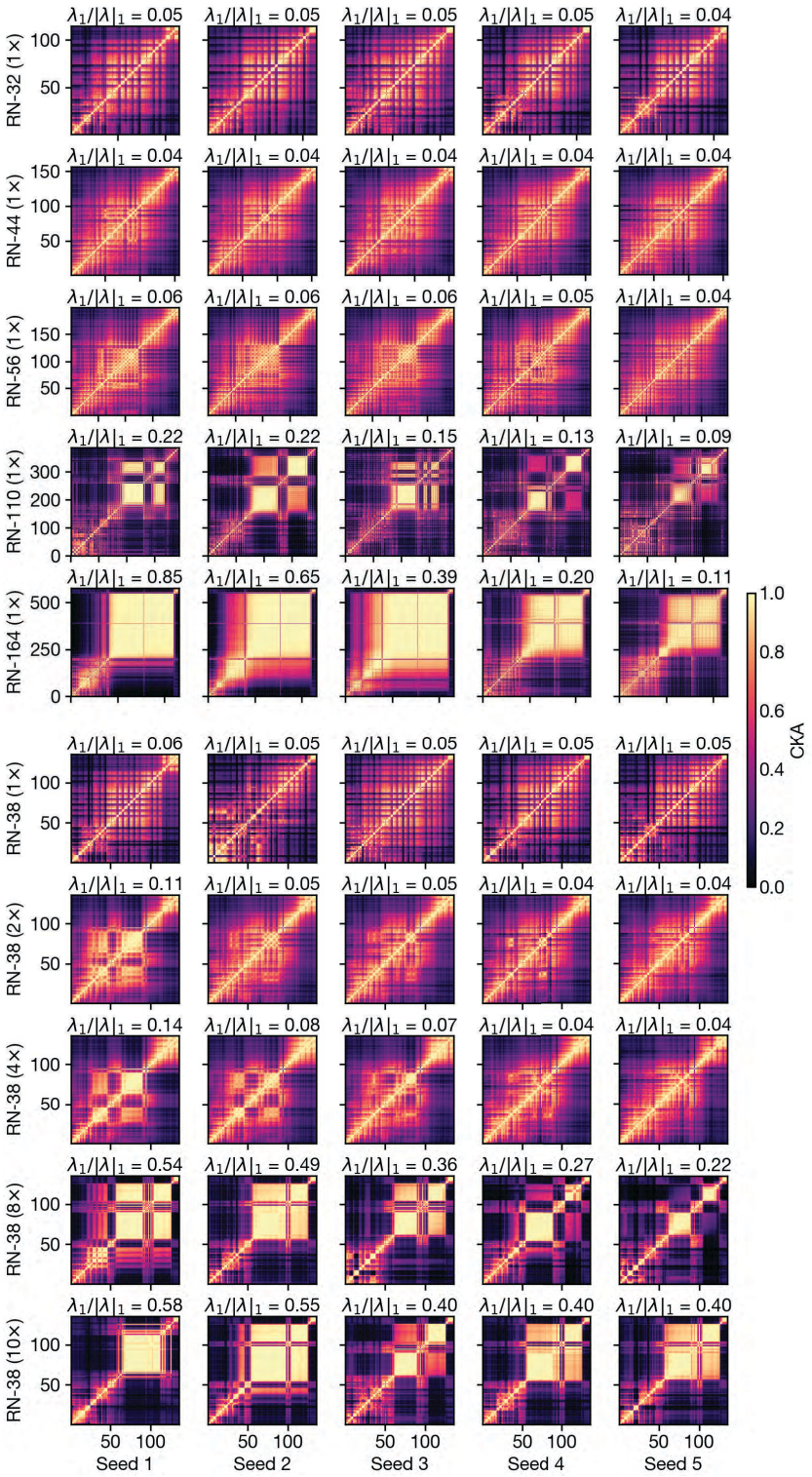

Figure D.5: **Top principal component explains a large fraction of variance in the activations of models with block structure.** Each row shows a different model configuration that is trained on CIFAR-10, with the first 5 rows showing models of increasing depth, and the last 5 rows models of increasing width. Columns correspond to different seeds. Each heatmap is labeled with the fraction of variance explained by the top principal component of activations combined from the last 2 stages of the model (where block structure is often found). Rows (seeds belonging to the same architecture) are sorted by decreasing value of the proportion of variance explained. We observe that this variance measure is significantly higher in model seeds where the block structure is present.

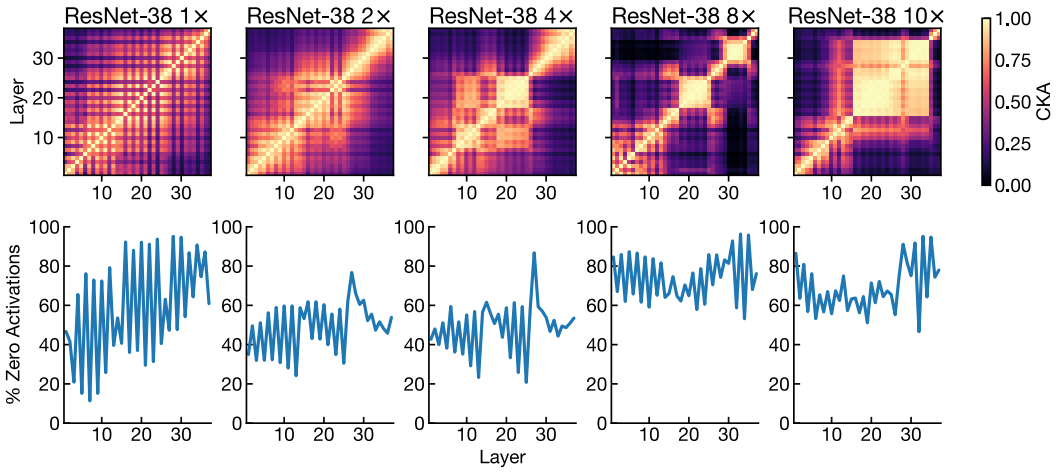

Figure D.6: **ReLU activations inside and outside the block structure are similarly sparse.** To rule out the possibility that the block structure arises because layers inside it behave linearly, we measured the sparsity of the ReLU activations. We observe that a significant proportion of activations are always non-zero, and the proportion is similar inside and outside the block structure. Thus, although layers inside the block structure have similar representations, each layer still applies a nonlinear transformation to its input.

E Linear Probes and Collapsing the Block Structure

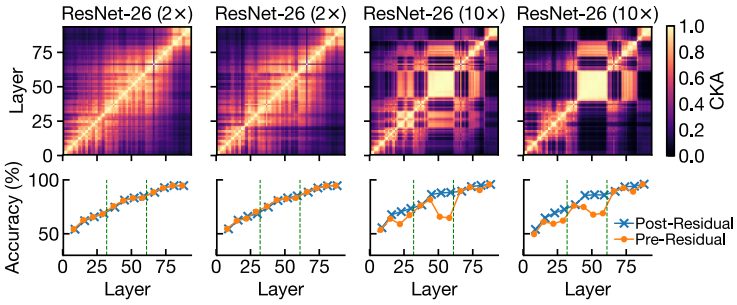

Figure E.1: **Linear probe accuracy.** Top: CKA between layers of individual ResNet models, for different architectures and initializations. Bottom: Accuracy of linear probes for each of the layers before (orange) and after (blue) the residual connections.

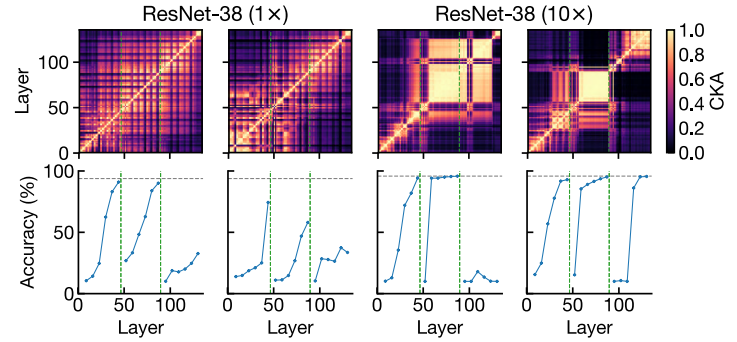

Figure E.2: **Effect of deleting blocks on accuracy for models with and without block structure.** Blue lines show the effect of deleting blocks backwards one-by-one within each ResNet stage. (Note the plateau at the block structure.) Vertical green lines reflect boundaries between ResNet stages. Horizontal gray line reflects accuracy of the full model.

With the insight that the block structure is preserving key components of the representations, we investigate how these preserved representations impact task performance throughout the network, and whether the block structure can be collapsed in a way that minimally affects performance.

In Figure E.1, we train a linear probe [1] for each layer of the network, which maps from the layer representation to the output classes. In models without the block structure (first 2 panes), we see a monotonic increase in accuracy throughout the network, but in models with the block structure (last 2 panes), linear probe accuracy shows little improvement inside the block structure. Comparing the accuracies of the probes for layers pre- and post-residual connections, we find that these connections also play an important role in preserving representations in the block structure.

Informed by these results, we proceed to pruning blocks one-by-one from the end of each residual stage, while keeping the residual connections intact, and find that there is little impact on test accuracy when blocks are dropped from the middle stage (Figure E.2), unlike what happens in models without block structure. When compared across different seeds, the magnitude of the drop in accuracy appears to be connected to the size and the clarity of the block structure present. This result suggests that block structure could be an indication of redundant modules in model design, and that the similarity of its constituent layer representations could be leveraged for model compression.