Emergent Fibration Symmetry Reveals the Geometry of Learning in Deep Neural Networks

Osvaldo M. Velarde[1], Alireza Hashemi[1], Lucas C. Parra[2], Hernán A. Makse[1*]

[1]Levich Institute and Physics Department, City College of New York, New York, NY, 10031, USA.

[2]Biomedical Engineering Department, City College of New York, New York, NY, 10031.

[*]Corresponding author. Email: hmakse@ccny.cuny.edu

Deep neural networks are often regarded as powerful yet opaque black boxes. In the absence of theoretical insights, performance has been driven by empirical scaling laws. Here, we demonstrate that learning generates local symmetries in the structure of deep neural networks, characterized by isomorphic input and output trees, called fibrations and coverings. These graph symmetries are more flexible than global symmetries used in theoretical physics and geometric deep learning. We prove that covers are stable attractors of stochastic gradient descent. Balanced coloring algorithms readily identify these emergent symmetries, and fibrations enable us to drastically compress the size of networks for most modern architectures without compromising performance. In Transformers, we improve the prevailing scaling law in the number of parameters. Breaking local symmetries also yields state-of-the-art performance in continual learning. These results suggest a new foundation for AI, not based on brute-force scaling up, but on emergent geometry that converts black boxes into interpretable colored graphs.

Despite the development of increasingly powerful neural network architectures, our understanding of how these networks learn and make inferences remains limited. They are often regarded as black boxes, unable to provide explanations for their decision-making processes. In the absence of a theoretically grounded understanding of learning, the recent surge in artificial intelligence (AI) has been driven predominantly by empirical scaling laws identified by Kaplan *et al.* (*1*, *2*). This paradigm promotes the notion that “bigger is better": more parameters, more layers, more data, and more computation.

The reliance on scaling as a substitute for understanding creates significant challenges. Training with single large runs has huge cost and energy demands, and when trained sequentially, networks often suffer from a loss of plasticity over time (*3*, *4*, *5*). Additionally, training large models from scratch does not make efficient use of the data, which requires exceedingly large datasets. The processing in these large models has been notoriously difficult to interpret, making it difficult to build trust in AI for critical applications, such as in the medical space. Without principled guidelines, architecture design becomes a costly trial-and-error process. Finally, large models are excessively over-parameterized, yet paradoxically, they perform well in practice (*6*), a subject of ongoing theoretical debate (*7*, *8*).

To address these challenges, we propose a theoretical formalism based on symmetries in the internal structure of the computational graph of deep neural networks. We find that learning via stochastic gradient descent (SGD) leads to the emergence of local symmetries in the computational graph, which are less rigid than better-known global symmetry groups, such as shift equivariance in convolutional neural networks (CNNs) ([9, 10]) or permutation equivariance ([11, 12, 13]) in graph neural networks (GNNs). Although symmetry groups are fundamental to geometric deep learning (GDL) ([14]), and our understanding of theoretical physics ([14, 15, 16]), they are too restrictive to capture the diversity observed not only in artificial, but also in biological neural systems ([17, 18, 19, 20]). We instead identify *local* symmetries that naturally emerge from the symmetries in the input trees of network nodes, relevant during inference, and in the output trees, relevant during learning.

These local symmetries are represented using mathematical maps known as fibrations, opfibrations, and coverings, and are found with "balance coloring" algorithms from graph theory ([17, 21]). These maps were originally developed by Grothendieck in category theory ([22]) and later adapted for graphs ([23, 24]). Here, we prove mathematically that covering symmetries emerge in stochastic gradient descent due to "synchronized learning" of the weight updates. This, in turn, induces fibration symmetries, which explains the emergence of neural activity synchronization often reported in deep networks ([20, 25, 26]), and allows substantial compression of over-parameterized models without compromising performance. We validate this phenomenon across a wide range of architectures, including multilayer perceptrons (MLP), CNNs, recurrent networks (e.g., long short-term memory, LSTM), and Transformers in both supervised and reinforcement learning (RL) settings. This geometric compression outperforms existing scaling laws ([1]) to achieve the same performance with fewer parameters. Furthermore, by carefully breaking these symmetries, we expand the network capacity to overcome loss of plasticity ([4, 5]), demonstrating state-of-the-art performance in continual learning ([3]). This principled approach, grounded in systematic symmetry identification, mitigates the inefficiencies of trial-and-error methods.

1 Hierarchy of symmetries

The symmetries we will discuss form a hierarchy and emerge in a variety of network structures. For an easy visualization of this hierarchy, consider a layered feedforward network with binary weights (where the connections are 1 or 0), as shown in Fig. 1. In the figure, nodes are colored to indicate their class according to various symmetries. We will generalize these concepts to continuous-valued weights in the next section.

Nodes have a *fibration symmetry* and are said to belong to the same *fiber* if they have isomorphic *input trees* ([17, 23, 24]). This means that the entire structure of the connections from the graph's input to the nodes is isomorphic (see Methods 6.1). In practice, fibers are identified by balanced coloring algorithms from graph theory ([17, 21]). This algorithm partitions the network into balanced coloring classes of nodes (the fiber partition) by iteratively assigning the same color to nodes that receive the same set of input colors ([27]), hence the term *balanced coloring*. An example is shown in Fig. 1a, where nodes in the same fiber are colored the same. The input tree for a red node is shown on the left.

Fibration symmetries allow the graph G to be compressed into a smaller *base* graph B_{fib}. This compression $\varphi_{\text{fib}} : G \to B_{\text{fib}}$ works by merging all nodes that share the same color (i.e., belong to the same fiber) while conserving the input trees (Fig. 1a). We will show that the resulting base

graph B_{fib} performs the same "forward" computation (see Eq. 1) as the original graph G. In deep neural networks (DNNs), it means that we can compress the network without losing performance. The inverse of this compression is called *lifting* (*23*), explained in more detail in Methods 6.1.1.

A similar principle applies when learning the parameters of the network. For the canonical learning algorithm, backpropagation, the error at the output is propagated backward through the *output tree* (see Eq. 2). An *opfiber symmetry* occurs when nodes share isomorphic output trees — the structure of connections leading from them to the output layer (Fig. 1b right). This allows for a similar compression $\varphi_{\text{op}} : G \to B_{\text{op}}$ of *opfibers* which are out-balanced colorings of the graph as shown in Fig. 1b. We will show that nodes with this symmetry receive the same error signal.

When multiple nodes have isomorphic input and output trees, they form a *covering symmetry* and belong to a *cover*(Fig. 1c). The corresponding coloring is the intersection of the fibration and the opfibration coloring partitions; i.e., a finer partition of the graph. Clearly, a covering has a more stringent symmetry than fibers or opfibers.

The most strict symmetry is the *automorphism*, which forms symmetry groups. This is a global permutation of the network's nodes that leaves the entire graph's connectivity unchanged, i.e., it is a global symmetry, which contrasts with the local preservation of inputs (outputs) in (op)fibration. An example of nodes that are in the same automorphism symmetry is shown in Fig. 1d along with the permutation of node labels. Although automorphisms are the cornerstone of GDL (*14*) and theoretical physics (*15*, *16*, *28*), this symmetry is so restrictive that we have never observed it in our trained networks.

In summary, these symmetries form a hierarchy of increasing strictness: from fibrations and opfibrations, to coverings, and finally to automorphisms. As conditions become more stringent, there are more distinct color classes and fewer nodes within each class (meaning fewer symmetries). Less strict local symmetries, such as fibrations, are more common and allow greater compression into a more compact base. This compression results in a model with fewer effective degrees of freedom, which corresponds to a stronger inductive bias. For more theoretical details about symmetries of a graph, see Methods 6.1.

2 Theoretical results

2.1 Computational graphs in deep learning

The hierarchical local symmetries explained above emerge in DNN structures with trainable parameters, like MLPs, CNNs, RNNs and Transformers (see Methods 6.2), but they are most easily understood in the canonical MLP. In this architecture, nodes are organized into multiple layers ($\ell = 1, ..., N$), with weight $W_{ik}^{(\ell)}$ connecting node k to i from layer $\ell - 1$ to ℓ. The weighted edges $W_{ik}^{(\ell)}$ create the computational graph that is the basis of our fibration analysis (see Fig. S6). Each node i in the network performs a weighted sum of its inputs k, followed by a point-wise nonlinear activation function σ,

$$h_i^{(\ell)} = \sigma\left(\sum_k W_{ik}^{(\ell)} h_k^{(\ell-1)}\right), \tag{1}$$

where $h^{(0)} = x \in \mathbb{R}^{d_0}$ is the input of the network and $h^{(N)} = y \in \mathbb{R}^{d_N}$ is the output of the network.

The activity propagates "forwards" across layers from the input x to the output y. During learning, the gradient of a loss function $\mathcal{L}$ with respect to weights $W_{ik}^{(\ell)}$ can be calculated with the error backpropagation algorithm, with the error signal $\delta_i^{(\ell)}$ flowing "backwards" through the computational graph (*29*),

$$\delta_i^{(\ell)} = \sigma'\left(h_i^{(\ell)}\right) \sum_k W_{ki}^{(\ell+1)} \delta_k^{(\ell+1)} , \tag{2}$$

where $\delta_i^{(N)} = \frac{\partial \mathcal{L}}{\partial h_i^{(N)}}$ is the error evaluated from the output layer. Note that the error signal depends on the activity of the node via the derivative of the activation function.

2.2 Fibration symmetry implies activity synchronization

We can now formally generalize the concept of fibration symmetry from binary weights to continuous weights. Two nodes i and j in layer ℓ belong to the same fiber (denoted $i \underset{\text{fib}}{\sim} j$) if and only if (iff):

- **Fibration symmetry:**

$$i \underset{\text{fib}}{\sim} j \iff \forall c \in C_{\ell-1}^{\text{fib}} : \quad \sum_{k\in c} W_{ik}^{(\ell)} = \sum_{k\in c} W_{jk}^{(\ell)} \tag{3}$$

This criterion partitions the nodes in the layer ℓ into a coloring C_ℓ^{fib} based on the input weights $W_{ik}^{(\ell)}$ and the color partitioning of the previous layer $C_{\ell-1}^{\text{fib}}$. In words, this definition states that two nodes are in the same fiber iff for all colors from the previous layer the sums of weights from these colors are the same. Or, more simply, the total input from each color is the same, which also ensures that their activity is the same. This recursive definition starts in the input layer, $\ell = 0$, with all nodes (d_0 nodes) having distinct colors $C_0^{\text{fib}} = \{1, 2, ..., d_0\}$, i.e., trivial fibers. The definition of fibers in Eq. (3) based on the sum of weights is the correct generalization from binary graphs (Fig. 1) to weighted graphs, as we show in Methods 6.3.

Similarly to symmetry-induced invariance theorems in physics (*15*), fibration symmetry induces activity synchronization during forward inference. This is the subject of the following theorem:

- **Activity synchronization:** Given two nodes i and j in layer ℓ, the following hold in networks with inputs x:

$$i \underset{\text{fib}}{\sim} j \implies h_i^{(\ell)} = h_j^{(\ell)} \qquad \text{for any network input } x \tag{4}$$

The proof of Eq. (4) is in Methods 6.4. The relation emerges naturally from the recursive dependence of neuronal activity $h_i^{(\ell)}$ on activity in the previous layer through input weights $W_{ik}^{(\ell)}$. Similar proofs exist for dynamical systems in (*17*, *30*, *31*).

Equation (1) establishes the relationship between the network structure captured by its weights and the network function captured by its activity. Therefore, the input tree structure governs the synchronization patterns.

2.3 Flavored opfibration implies error synchronization

The backpropagation of errors in Eq. (2) exhibits greater complexity due to the intrinsic coupling between error gradients and forward activations mediated by the slope of the nonlinear activation $\sigma'(h_i^\ell)$. Specifically, the error $\delta_i^{(\ell)}$ for node i depends not only on the error signals from the next layer (mediated by the output weights $W_{ik}^{(\ell)}$) but also on the activation value $h_i^{(\ell)}$ at that node, which introduces a nonlinear coupling between $\delta_i^{(\ell)}$ and $h_i^{(\ell)}$, absent in forward propagation.

This coupling can be systematically absorbed into an output tree that governs error propagation analogously to how the input tree governs activity propagation. The key observation is that the slope — which is a function of $h_i^{(\ell)}$ — couples the in-balanced color assignments from the forward fiber structure (see Eq. (3)) to the edges that define the error flow in Eq. (2).

We formalize this by introducing *edge flavors* into the output tree, where the flavors are inherited from the node colors of the forward computation. We take advantage of the concept of *flavor-preserving* opfibration symmetry (*32*), which preserves flavor assignments of edges. Therefore, two nodes belong to the same flavored opfiber if:

- **Flavor-preserving opfibration symmetry:**

$$i \underset{\text{op}}{\sim} j \implies \forall \hat{c} \in C_{\ell+1}^{\text{op}} : \quad \sum_{k\in\hat{c}} W_{ki}^{(\ell+1)} = \sum_{k\in\hat{c}} W_{kj}^{(\ell+1)} \quad \& \quad q(i) = q(j) \tag{5}$$

with flavors given by

$$q(i) = \begin{cases} 1 & \text{linear activation} \\ c(i) \in C_\ell^{\text{fib}} & \text{non-linear activation} \end{cases}$$

Equation (5) defines the opfiber partition C_ℓ^{op}. In the linear activation case ($\sigma' = 1$), all weights have the same flavor ($q = 1$), and the forward and backward coloring decouple and can be treated independently. In the non-linear case, the flavors must first be calculated using the fiber coloring C_ℓ^{fib} computed forward through the entire network as above. Then, the sum-of-weights condition of Eq. (5) is evaluated for each flavor separately, recursively starting at the output of the network, with all nodes in different opfibers $C_N^{\text{op}} = \{1, 2, ..., d_N\}$.

Using this definition, we arrive at the following theorem. The proof is given in Methods 6.4.

- **Errors synchronization:** Given two nodes i and j in layer ℓ, the following hold in a network with inputs x and targets y:

$$i \underset{\text{op}}{\sim} j \implies \delta_i^{(\ell)} = \delta_j^{(\ell)} \qquad \text{for any network input } x, \text{ target } y\,. \tag{6}$$

We note that for linear activations ($\sigma' = 1$), the relationship between synchronization and (op)fibration becomes bidirectional ($\iff$) in both Eqs. (4) and (6, as we prove in Methods 6.5. This means that synchronized nodes necessarily form (op)fibers, and conversely, nodes in a (op)fiber will synchronize. This represents a strict one-to-one structure-function equivalence that is difficult to demonstrate for more general non-linear dynamical systems (*33*).

Alternative definitions for opfibration that capture the coupling with the forward pass are treated in Supplementary text 7.1. In the theory of gases, the linear case is analogous to non-interacting particles in an ideal gas, whereas the nonlinear case corresponds to particle interactions in a real gas, and flavor plays the role of a compression factor.

2.4 Fibers and opfibers synchronize learning

The weight update of the gradient descent algorithm locally depends on node activity and error signals (29):

$$\Delta W_{mi}^{(\ell)} = -\alpha \frac{\partial \mathcal{L}}{\partial W_{mi}^{(\ell)}} = -\alpha \delta_m^{(\ell)} \cdot h_i^{(\ell-1)} , \tag{7}$$

where α is a learning rate (see the derivation in Methods 6.6).

According to Eqs. (4) and (6) nodes i, j of layer $\ell - 1$ have the same activity, $h_i^{(\ell-1)} = h_j^{(\ell-1)}$ if they are in a fiber; and nodes m, n of layer ℓ have the same error signal $\delta_n^{(\ell)} = \delta_m^{(\ell)}$ if they are in an opfiber. From this and Eq. (7) follows:

- **Learning synchronization:** Given nodes $i \underset{\text{fib}}{\sim} j$ in layer $\ell - 1$ and $m \underset{\text{op}}{\sim} n$ in layer ℓ, the weight updates according to gradient descent are synchronized:

$$\Delta W_{mi}^{(\ell)} = \Delta W_{mj}^{(\ell)} = \Delta W_{ni}^{(\ell)} = \Delta W_{nj}^{(\ell)} . \tag{8}$$

The proof is given in Methods 6.7. Note that, unlike the synchronization of activations and errors in Eqs. (4) and (6), this synchronization occurs in the parameter space. Fig. 2a shows an example of this synchronization.

2.5 Covering symmetries emerge during stochastic gradient descent

The final symmetry that we introduce in DNNs is the covering symmetry. Two nodes belong to the same cover iff they are in the same fiber and the same flavored opfiber:

- **Covering symmetry:**

$$i \underset{\text{cov}}{\sim} j \iff i \underset{\text{fib}}{\sim} j \quad \& \quad i \underset{\text{op}}{\sim} j . \tag{9}$$

This condition defines a covering partition of the network C_ℓ^{cov}, which, as we shall see, is preserved under GD.

We now turn to the question of how these symmetries emerge during learning. When a neural network is initialized, its weights are set to random values. Large random graphs of this kind rarely have significant global automorphism symmetries as they are forbidden by Erdős-Rényi asymmetry theorem (34). However, the training process itself creates local redundancies. For example, recent studies have observed that SGD causes different nodes to develop similar input and output weights (35). We have also previously observed the emergence of *synchronized* nodes – nodes that have similar activity across all inputs during training (20).

Our results provide a theoretical proof linking these empirical observations to the emergence of covering symmetry. Namely, as a consequence of *synchronized learning*, once two nodes are in a single covering at learning time step t, the weight update with the gradient descent algorithm keeps the nodes in the covering at time $t + 1$. This is formalized in the following theorem:

Theorem 2.1 ***Cover Coarse-Graining Theorem.*** *Under the dynamic of GD, the covering partition* $C_\ell^{\text{cov}}(t+1)$ *is a coarsening of* $C_\ell^{\text{cov}}(t)$.

We proof this theoretical result in Methods 6.8. The theorem implies that once multiple nodes are in the same cover, they cannot exit that cover, and thus constitute an invariant set. However, distinct covers can merge to form larger (coarser) covers. The proof of this Cover Coarse-Graining Theorem is based on stability rules (see Methods 6.8). These provide an intuition for how fibration and opfibration symmetries develop during learning. The stability rules state that the fibers in the layer ℓ preserve the fibers in the layer $\ell + 1$, while the opfibers preserve the opfibers in the layer $\ell - 1$, as shown in Fig. 2b. Thus, opfiber formation tends to propagate backward from the output layer, while fiber formation tends to propagate forward from the input as learning progresses. Note that for the non-linear case, the opfibers have a flavoring of the edges to capture their dependence on the forward pass. This introduces an asymmetry in the stability rules, reflected in Fig 2b, with edge colors defined at the output, but not at the input.

Chen *et al.* (*35*) have shown that any invariant parameter set of GD is a stable attractor in the SGD algorithm, provided that there is a sufficiently large learning constant. Another interpretation of Theorem 2.1 is: *If network parameters θ_t at time t are such that $i \underset{\text{cov}}{\sim} j$; then $i \underset{\text{cov}}{\sim} j$ still holds for θ_{t+1}.* That is, $i \underset{\text{cov}}{\sim} j$ is invariant under the weight update in GD. The intuition is that stochastic fluctuations during learning, akin to noise-induced stabilization in dynamical systems (*36*), occasionally bring nodes into an invariant set; once there, our theorem says the gradient descent algorithm can no longer break the symmetry due to synchronized learning. Using the same result, we conclude that cover formation is a stable attractor of the SGD dynamics. In Section 3.1, we provide empirical evidence for this claim.

The theoretical results presented here readily generalize to other computational graphs, including hypergraphs, as we discuss in Methods 6.2, which includes operations such as convolutions (in CNN), multiplicative gating (in most RNN and LSTM in particular), residual connections, and attention layers, which is the core innovation of Transformers.

2.6 Fibration compression preserves activity and loss

Once fibers are identified, one can produce a more compact base graph by merging all nodes in a fiber. The forward computation of the graph G is preserved in this base graph B_{fib} if the new weights $\hat{W}$ are specified as follows (see the proof in Methods 6.9.1):

$$\hat{W}^{(\ell)}_{c'c} = \frac{1}{|c'|} \sum_{i \in c', k \in c} W^{(\ell)}_{ik}. \tag{10}$$

where $c \in C^{\text{fib}}_{\ell-1}$, $c' \in C^{\text{fib}}_{\ell}$ and $|c'|$ is the number of nodes in the fiber c'. In other words, we sum up all output weights with the same colors and average over the input weights. This equation defines fiber compression in the weighted graph $\varphi_{\text{fib}} : G \to B_{\text{fib}}$, and is exemplified in Fig. 2c.

For convolutional networks, the same compression rule applies to feature channels rather than nodes (Fig. 2d, Methods 6.9.2). For gates in recurrent networks (e.g. LSTM) and attention layers (Transformers), there are no weighted interactions and compression only needs to connect nodes in the base as exemplified in Fig. 2e-f (see Methods 6.9.3).

We will demonstrate that these compression methods allow substantial compression of networks without changing their performance, and that this targeted fibration compression outperforms existing pruning methods (Section 3.2). We then show that more aggressive compression followed by retuning results in a new scaling law that outperforms the existing parameter scaling law for

Transformers (*1*) (Section 3.3). Finally, we argue that the emergence of covering symmetry results in the loss of plasticity in deep networks (*5*) and show empirically that breaking these symmetries achieves a new state-of-the-art in continual learning (Section 3.4). But first, we empirically confirm how fiber symmetries emerge during learning (Section 3.1).

3 Empirical results

3.1 Emergence of fibers and opfibers during learning

We empirically validate the emergence of covers during SGD for an MLP trained in MNIST and a CNN trained in ImageNet (see Fig. 3a, c). At the start of learning, all nodes have unique random connectivity, constituting trivial (single-node) fibers and opfibers. As learning progresses, the nodes start to group into covers, forming fibers and opfibers. Hence, the number of unique fibers decreases. When the fibers stabilize and the accuracy no longer increases, the network has learned the training set. Details about the MLP and CNN used here are provided in Methods 6.10.1 and 6.10.2. The formation of fibers across layers is summarized in Fig. S7 - Methods 6.11. The emergence of symmetries is illustrated in Supplementary Material Movie S1.

3.2 Fibration compression outperforms existing pruning methods

To identify fibers, we have to check if weights satisfy Eq. (3). In practice, this can only be determined with a precision ε: $|\sum_{k\in c}[W_{ik}^{(\ell)} - W_{jk}^{(\ell)}]| \leq \varepsilon$. The coloring algorithm that identifies isomorphic input trees (detailed in Methods 6.12) therefore has this fiber precision ε as a parameter. With increasing ϵ, a larger fraction of nodes can be compressed into (quasi-) fibers (*37*), but the forward computation is no longer exactly preserved and we observe a loss of performance.

For the MLP trained on MNIST, the fibration compression reduces the model to 17% of its original parameter count without an appreciable loss of classification accuracy (black point in Fig. 3b). We observe comparable compression for a CNN trained on ImageNet, where fibration compression achieves reductions of similar magnitude without degradation in test performance (Fig. 3d). For both MLP and CNN, we find that alternative compression methods are less effective. In Fig. 3b-d, fibration compression (red curve) outperforms random pruning nodes (blue curve) and pruning nodes with small weights (L2-norm, pink curve). Details about pruning are shown in Methods 6.10.3.

Next, we evaluate lossless compression in a reinforcement learning context using an LSTM-based agent trained to play the Atari game Beam Rider (details in Methods 6.10.4, see Fig. 3e, green curve). At training epoch 50, we compress the network using $\varepsilon = 0.3$ (black), achieving a network with 40% of the original parameters. The compressed network did not exhibit degraded performance after continued training. The latter continues to create fibers, allowing for further compression to 20% of the original model size in episode 300.

3.3 Fibration compression with retuning reveals efficient scaling law for Transformers

For larger compression factors (achieved with larger ε) the loss in performance can be compensated for by retuning the network after compression, a standard approach in existing pruning methods (*38*, *39*). We explore this in the context of a sequence-to-sequence Transformer trained for German-English translation (Methods 6.13). For this purpose, we develop a fibration compression and retuning method with adaptive fiber precision (Methods 6.14 - Fig. S8) resulting in retuning curves (Fig. 3g, orange and purple curves). For a model with 78M parameters, we compress the transformer by a large factor of 31x of its original size with 4.1% loss in performance (black arrow on the orange retuning curve in Fig. 3g). In Fig. 3g, we compare this method with other approaches: standard pruning methods (blue and pink curves), and fibration compression without retuning while searching for the optimal ε value for each layer individually (red curve).

Next, we train different Transformers with varying network sizes (e.g. 50M and 78M parameters - purple and orange curves in Fig.3g and orange retuning curves in Fig.3h) and then compress each model size with fibration retuning. The baseline model (without compression) follows the established scaling law of Kaplan *et al.* (*1*) between the loss function and the number of parameters (black dashed line) with a power-law behavior:

$$\mathcal{L} = L_0^k \; P^{-\alpha_k} \,, \tag{11}$$

where P is the number of parameters of the baseline model, L_0^k is a constant and $\alpha_k = 0.028$ is the Kaplan exponent (Methods 6.15). By Kaplan scaling, we mean increasing the model's parameter count by increasing all layers' dimensions by the same amount. For each baseline model, we then apply the fibration compression-retuning method and find a more efficient scaling law for the compressed network (orange dashed line):

$$\mathcal{L} = L_0^f \; (P_{min|L})^{-\alpha_f} \,. \tag{12}$$

Here, $P_{min|L}$ is the minimum number of parameters of the compressed model at a given loss value and baseline model (see Methods 6.15). The power-law exponent obtained with the fibration compression protocol ($\alpha_f = 0.049$) is larger than the value obtained without compression implying a faster-decaying scaling curve.

The model at $P_{min|L}$ represents a minimal fibration base of the transformer computational graph with baseline P, effectively capturing the data's symmetries using a minimal representation. The faster decay of the fibration scaling suggests the existence of an alternative, more efficient approach to modeling scaling than the brute-force method, in which all layers are uniformly scaled up, as exemplified by the Kaplan scaling laws.

3.4 Fibration symmetry breaking for continual learning

Cover symmetries that emerge during SGD reduce the network's degrees of freedom and manifest as redundant, synchronized activity that limits the repertoire of learnable features. Several heuristics have been proposed to promote diversity and avoid redundancy. For instance, "dropout" (*40*) breaks symmetries by updating only a fraction of nodes selected at random. Residual connections (*41*) break symmetries that have emerged in the skipped layers, as we show in Methods 6.2. Techniques

that explicitly avoid redundancy also prevent fiber formation. For example, (*42*) proposes to identify and remove redundancy in convolutional layers (see Supplementary Text 7.2). SlimConv recombines features in convolutional layers (*43*) and Barlow Twins (*44*), aiming to reduce correlations between learned features (see Supplementary Text 7.3). However, these ad hoc mechanisms operate indiscriminately by preventing or breaking the fibers that emerge from the data with SGD.

In contrast, we propose a principled surgical approach to symmetry breaking (*45*) by first compression covers and then randomizing the remaining weights as follows (Fig. 1e):

1. Identify covers and reduce the graph to the covering base by applying fibration symmetry compression via Eq. (10) to guarantee that the function of the network and its performance are preserved.

2. Add nodes to restore the original size. Each new node is initialized with random input weights and zero outgoing weights. This restores degrees of freedom for further learning.

Deep networks have been found to lose their ability to learn when trained sequentially on multiple new tasks (*4*, *5*). Current heuristics for training new tasks in continual learning, such as resetting the final layers with random weights or resetting specific nodes, do not fully overcome this loss of plasticity (*5*).

We propose using the symmetry-breaking approach above to overcome this loss of plasticity in continual learning. We evaluate the breaking process on the continuous ImageNet task (*46*), which comprises 5,000 sequential binary classification tasks drawn from the 1,000-class data set (details in Methods 6.16). First, we observe that conventional SGD stops learning after approximately 2,000 tasks (Fig. 3f, blue curve), as shown by (*3*). We find that this is accompanied by cover formations (Methods 6.16 - Fig. S9). The emergence of symmetry induces parameter tying, which reduces the degrees of freedom in the parameter space. This constraint directly explains the observed loss of network plasticity: with fewer independent directions for optimization, learning stagnates. We compared performance to the state-of-the-art "Continual Backpropagation" from (*3*), which outperforms heuristics like shrink-and-perturb (*47*). Continual backpropagation resets nodes that have become inactive. In Supplementary text 7.4, we show that this technique is a special case of our symmetry-breaking mechanism. Up to task 1,500, the performance of continual backpropagation and our method "Fibration Symmetry Breaking" is comparable. However, beyond that point, our method outperforms continual backpropagation, cutting the remaining deficit in accuracy by half (from 90% to 95% by task 5,000). This indicates that the network continues to capture useful features that generalize between classes.

3.5 Emergence of synchronization clusters

The activity synchronization theorem, Eq. (4), establishes that the nodes of a fiber have the same activity for all inputs. We ask how node synchronization changes when we limit the input space and, in particular, if we draw inputs from individual classes in a classification task. We find that the correlation matrices for individual classes form large clusters of synchronized nodes (Fig. 4a). These class-conditional clusters are a coarsening of the fibers (see Methods 6.17 - Fig. S11). Thus, each class-conditional synchrony cluster can be decomposed into multiple fibers, forming hierarchical clusters of nodes. One can interpret the class-conditional synchronization clusters in Fig. 4a in terms of features of the input: It is well accepted that node activity in hidden layers

captures a hierarchy of features in the stimulus, that are shared across classes; e.g. simple examples for 2D CNN are edges, junctions, and corners (*48*). The intersection of all class-conditional clusters is a refinement that is captured by the fibers. Therefore, fibers capture the hierarchical regularity present in all training data.

4 Hierarchical organization of the loss landscape under SGD dynamics

Cover formation is a hierarchical coarse-graining under the SGD dynamic in the parameter and node space. This is exemplified in Fig. 4b-c, where we show the state of the network in three different training stages (T_1, T_2, T_3), their transitions $T_{1\to2}$ and $T_{2\to3}$, and the evolution of three nodes (i, j, k) as covers merge.

Covers are a refinement of fibers. Fibers are a refinement of synchronization clusters, and these in turn are a refinement of class-conditional synchrony clusters. This is exemplified in Fig. 4b, along with the development of covers and class-conditional clusters during learning. The red box groups the class-conditional clusters for the red class (and similarly for the blue class) at time T_1. Covers in green are a refinement of both red and blue clusters. As learning progresses, the covers merge at transitions $T_{1\to2}$ and $T_{2\to3}$ according to the coarse-graining theorem.

Figure 4c is a symbolic representation of a parameter space (weights). The potential learning trajectories (black curves) merge the covers containing nodes i, j and k. During the learning phase T_1, the three nodes i, j, and k belong to distinct covers. At a later point in the training, i and j merge into the same cover. According to the coarse-graining theorem, once this occurs, they cannot separate again. In other words, the first time i and j share a cover defines an irreversible transition (green dashed line $T_{1\to2}$). A similar transition occurs when the cover containing node k merges into the cover containing i and j, defining a second transition line $T_{2\to3}$. The colored zones in this space are regions where the network has the same cover base B (with identical nodes and weights).

The loss landscape $\mathcal{L}(w)$, with $w \in \mathbb{R}^d$, is characterized by a hierarchy that can be formally represented as a tree (Fig. 4d). All network configurations in this d-dimensional parameter space that have the same minimal base B'' will have the same loss value. The root of the tree is this base in $\mathbb{R}^{d''}$; while the leaf nodes are all the possible lifting of the base in $\mathbb{R}^d$. The tree contains multiple leaves because φ_{fib} can reduce different graphs G into the same minimal base B'' (This is evident in Eq. (10), as a linear subset of all possible $W^{(\ell)}$ can generate the same $\hat{W}^{(\ell)}$). The inverse lifting operation can transform a representation from $\mathbb{R}^{d''}$ (minimal base B'') to $\mathbb{R}^d$ (original network G) through an intermediate lifting to $\mathbb{R}^{d'}$ where $d'' < d' < d$. These intermediate steps create the hierarchy of the trees (see Methods 6.18 - Fig. S12).

The learning trajectory navigates downhill in this loss landscape (solid black line, in Figs. 4c-d) following the hierarchical coarsening of the covers under the SGD learning dynamics where the system progressively explores increasingly coarse-grained symmetry quotients during optimization. This structure has analogies in the organization of associative memory in neural networks (*49*) and the energy landscape in spin glasses (*50*).

5 From black box to colored graphs through geometric interpretability

Our geometric interpretation transforms the interpretability problem in deep learning. Rather than treating neural networks as opaque black boxes, the fibration framework reveals them as colored geometric structures where each color represents a distinct fiber. The training process becomes interpretable as a geometric flow through hierarchical parameter space, where SGD converges on regions of constant-loss manifolds and crosses boundaries at irreversible symmetry transitions. The interpretability emerges directly from the geometry: we can trace which nodes belong to which fibers (colors), understand when and why they merge during training (geometric transitions) and predict their collective behavior (synchronization within fibers).

This geometric lens transforms the fundamental question from "what is the network computing?" to "what geometric structure has the network discovered?" The colored fiber decomposition provides an atlas of the network's internal organization, where the geometry itself constitutes the interpretable representation. Unlike post-hoc interpretability methods that attempt to explain black boxes after training, our framework reveals that networks are inherently geometric objects whose training dynamics naturally expose their interpretable colored structure. The black box was never truly opaque; it simply waited to be viewed through the right geometric coordinates, which are the colored fibers that capture its fundamental symmetries.

We have shown how learning aligns the network's internal structure with the structure of the data, finding a compact, minimal set of features sufficient to describe the dataset, and thus controlling for overtraining. This perspective resolves the paradox of over-parameterization in deep networks: gradient descent finds structured symmetric solutions, implying that the emergent model is much simpler than the raw parameter count suggests. It also explains why deep networks outperform shallow networks despite the Universal Approximation Theorem; depth facilitates the gradual layer-by-layer formation of coverings, a process that is statistically improbable in a wide, shallow network.

Our framework also provides a unified theoretical foundation for many ad-hoc heuristics. Hard-coded inductive biases such as weight-tying in CNNs are simply a predefined symmetry; our work shows how such symmetries can emerge from learning. Our symmetry-driven pruning provides an exact, data-independent theory for network compression, validated across diverse architectures including MLPs, CNNs, transformers, and in reinforcement learning settings. The fibration allows for drastic compression of symmetric nodes without loss of performance. Furthermore, we introduce a novel symmetry-breaking protocol based on the lifting property that outperforms state-of-the-art continual learning. This provides a principled mechanism to "stay young" and generalizes practices such as continual backpropagation.

In essence, our work frames learning not as parameter-tuning, but as structure formation. The process transforms the opaque black box into an interpretable colored graph, where emergent symmetries reveal the data's learned regularities. This theory-driven perspective offers a path to designing future AIs in which mathematical principles drive performance, generalization, and interpretability.

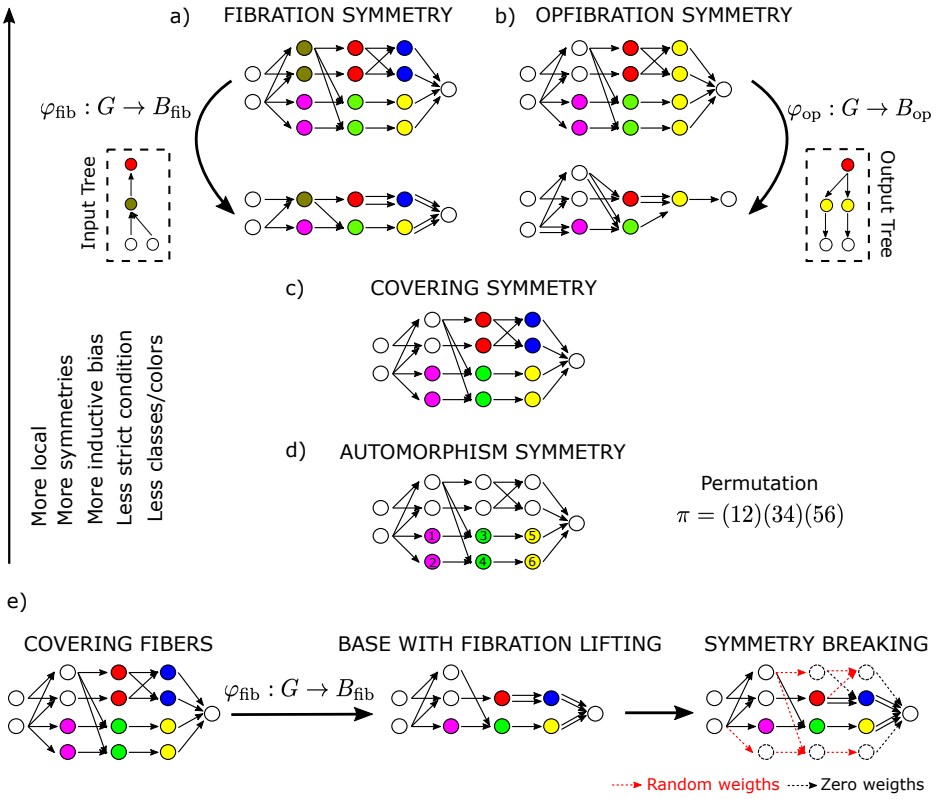

Figure 1: **Hierarchy of symmetries in a feedforward network.** For simplicity, weights are binary, and edges with zero weight are not plotted. Conventional automorphism permutation is exemplified in the case of a graph in panel **(d)**. It identifies three pairs of symmetric nodes by a permutation π. The proposed framework is based on the generalization to local symmetries, such as (op)fibrations and coverings (**a-c**); see text for detail. These symmetries have less strict conditions that can be satisfied by a larger number of nodes resulting in fewer symmetry classes (colors) (blank nodes should be interpreted as distinct colors). The reduced degrees of freedom increases the inductive bias. **(e)** Breaking of symmetry for continual learning consists of two steps. Compress to the covering base via fibration-lifting (middle) to preserve the learned task. Randomize or zero out redundant weights (right) to provide new degrees of freedom to continue learning.

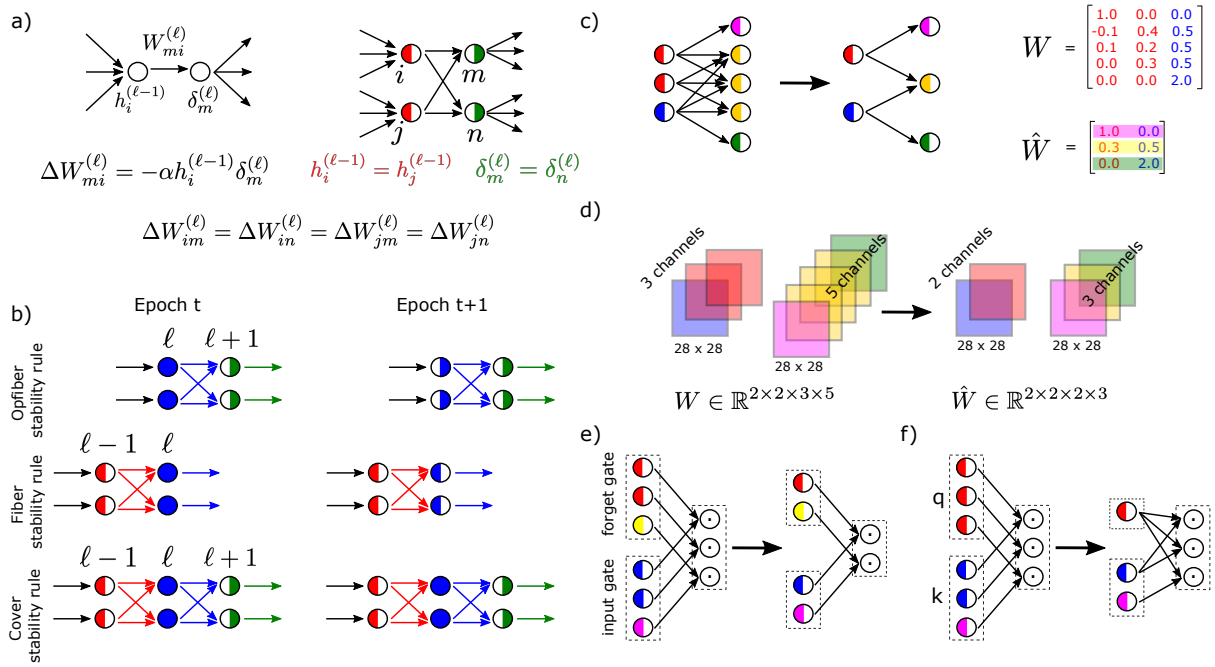

Figure 2: **Theoretical results**. **(a)** The gradient descent learning rule updates a connection's weight $W_{mi}^{(\ell)}$ based on the product of the activity $h_i^{(\ell-1)}$ and the error signal $\delta_m^{(\ell)}$. Nodes in the same fiber (red left sides) synchronize their activities - Eq. (4). Nodes in the same opfibers (green right sides) synchronize their error signals - Eq. (6). Weights connecting nodes in a fiber with nodes in an opfiber will have the same updates $\Delta W_{mi}^{(\ell)}$ - Eq. (8). **(b)** Stability rules used to proof the cover coarse-graining theorem (see Methods 6.8). Connection colors indicate "flavor" for the opfibers. Black connections for fibrations are not flavored. **(c)** Example of fibration compression of a dense layer. Edges with zero weight are not plotted. $\hat{W}$ is calculated based on W using Eq. (10). **(d)** Example of fibration compression of a convolution layer. It reduces the number of channels (3 to 2 and 5 to 3). **(e)** Example of fibration compression of gates in LSTM. The base has an identical number of nodes across all gates (here 2 nodes per gate). **(f)** The fibration compression for attention modules preserves the count of interactions (3 multiplications) even when the number of fibers of the k and q differ.

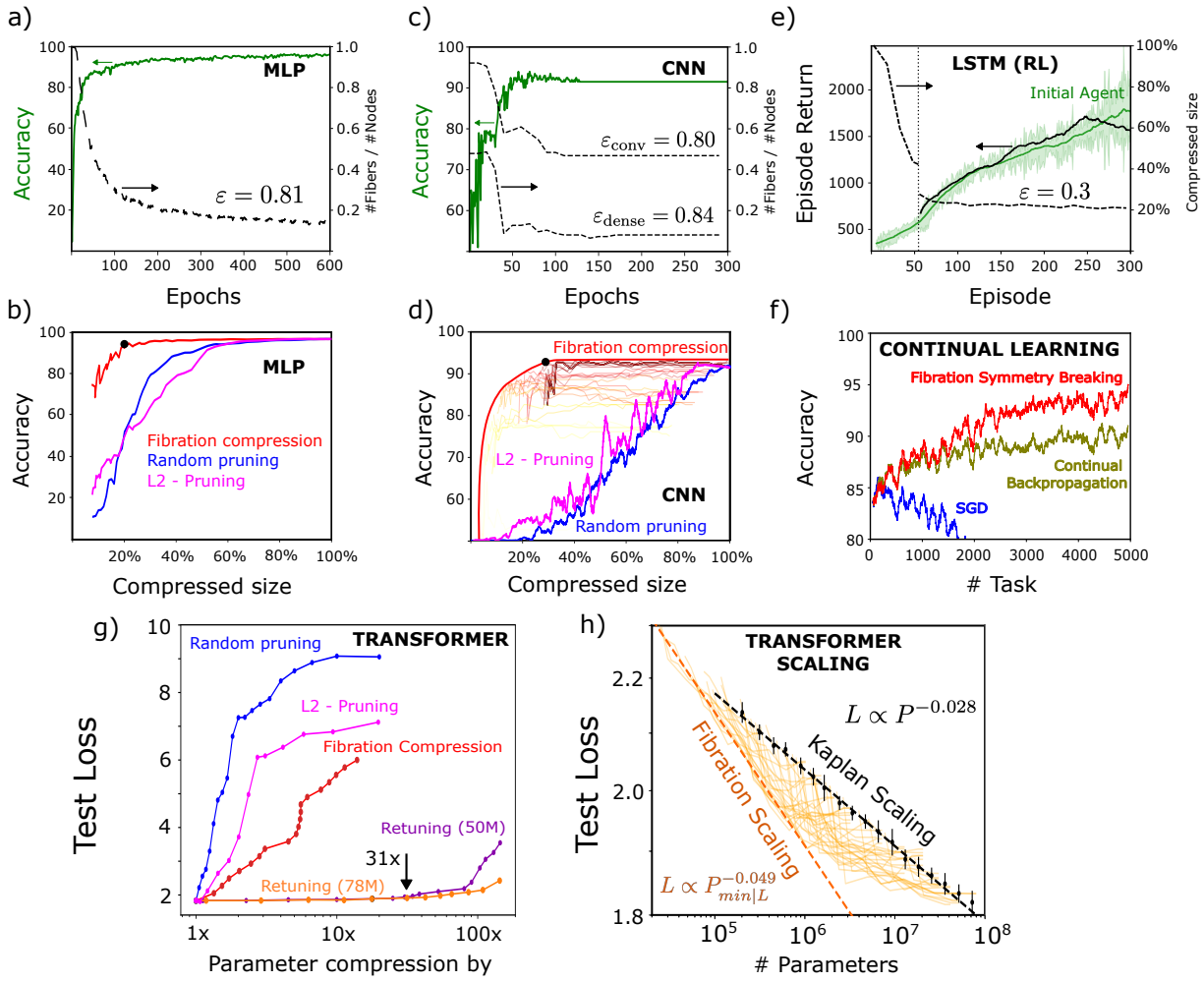

Figure 3: **Empirical results. (a)** MLP trained on MNIST dataset. Classification accuracy increases across learning epochs (green), while the number of trivial (single-node) fibers decreases in favor of larger multi-node fibers (black dashed curves). **(b)** Accuracy of the compressed network depends on the level of compression (red). Alternative pruning methods at different levels of compression (blue and pink curves). **(c)** Same as (a) but for CNN trained on ImageNet. **(d)** Same as (b) for CNN. Yellow to red curves correspond to different values of $\varepsilon_{\text{conv}}$. Bold red curve represents maximum performance for different combinations of ($\varepsilon_{\text{conv}}$,$\varepsilon_{\text{dense}}$). **(e)** Agent with LSTM trained via reinforcement learning to play the Atari game Beam Rider. Returns increase with learning episode (solid curves). Learning with no compression (green). At episode 50 (vertical line), the network is compressed (black) and training continues. Size of fibration base (dashed curves). **(f)** Continual learning with sequential ImageNet binary classification tasks, using conventional SGD (blue), Continual Backpropagation (brown), and the proposed Fibration Symmetry Breaking (red). **(g)** Transformers trained on the Multi30k dataset. Fibration compression using layer-specific optimal ε (red). Fibration compression followed by retuning for a model with 50M parameters (purple) and 78M (orange). Blue and pink as in (b). **(h)** Power-law scaling of the loss value as a function of the number of parameters in the original transformer (black) and after fibration compression and retuning (orange curves). Parameter count excludes the embedding layers. Dashed scaling lines are fit to the original networks and to the most compressed network at a fixed loss. More details in the text.

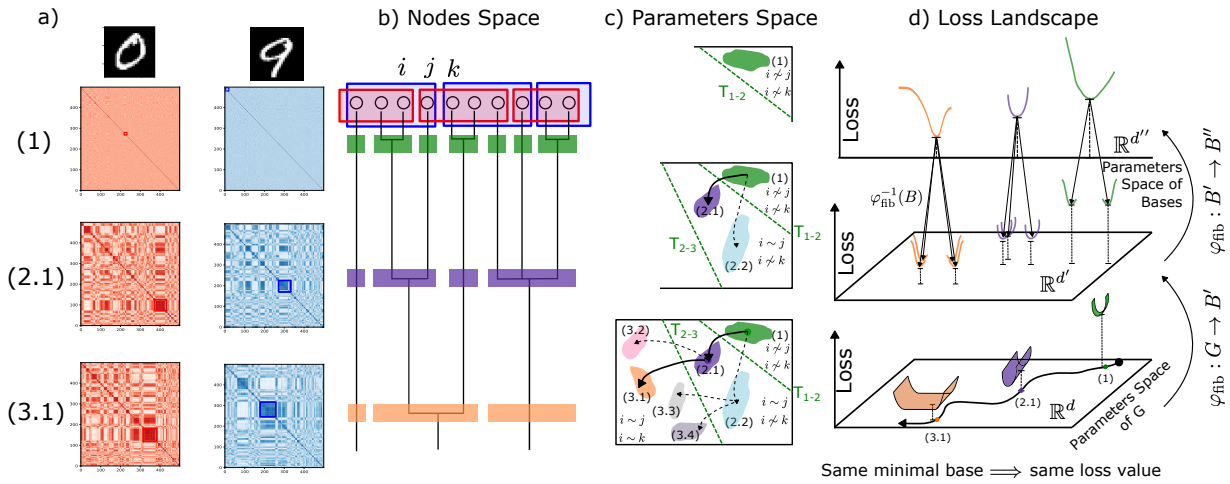

Figure 4: **Synchronization, hierarchical clustering and loss landscape. (a)** Class-conditional correlation matrix of activity with network inputs drawn from individual classes. Here they are shown for the last layer of an MLP trained on MNIST in early epochs (1), in middle of the training (2.1) and after training (3.1). Clusters in this matrix are referred to as synchrony clusters. **(b)** Temporal evolution of covers in one layer. In one of the early epochs, (1), we identify synchrony clusters (red and blue) and covers (green). Covers are located within cluster intersections. In later epochs, (2.1) and (3.1), covers merged to form coarser partitions (purple and orange). Nodes i, j and k start in different covers (green), but merge into the same cover (purple and orange). **(c)** A network is a point in parameter space, e.g. at time point (1). Multiple networks can have the same basis B (green region). Learning trajectories (black curves) merge nodes i and j into the same cover at an irreversible transition boundary T_{1-2}. Multiple B may have i and j in the same cover (e.g., network (2.1) and (2.2)). SGD selects one of these possibilities. Once k merges into the same cover, another irreversible transition occurs T_{2-3}. **(d)** Hierarchical loss landscape (see text for details).

References and Notes

1. J. Kaplan, et al., Scaling laws for neural language models. arXiv (2020), https://arxiv.org/abs/2001.08361.

2. J. Hoffmann, et al., Training compute-optimal large language models. arXiv (2022), https://arxiv.org/abs/2203.15556.

3. S. Dohare, et al., Loss of plasticity in deep continual learning. Nature **632**, 768–774 (2024).

4. C. Lyle, et al., Understanding plasticity in neural networks, in International Conference on Machine Learning (PMLR) (2023), pp. 23190–23211.

5. C. Lyle, et al., Disentangling the causes of plasticity loss in neural networks. arXiv (2024), https://arxiv.org/abs/2402.18762.

6. T. J. Sejnowski, The unreasonable effectiveness of deep learning in artificial intelligence. Proc. Nat. Acad. Sci. USA **117**, 30033–30038 (2020).

7. M. Belkin, D. Hsu, S. Ma, S. Mandal, Reconciling modern machine-learning practice and the classical bias–variance trade-off. Proc. Nat. Acad. Sci. USA **116**, 15849–15854 (2019).

8. P. Nakkiran, et al., Deep double descent: Where bigger models and more data hurt, in International Conference on Learning Representations (ICLR) (2020).

9. Y. Lecun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition. Proceedings of the IEEE **86** (11), 2278–2324 (1998).

10. T. Cohen, M. Welling, Group equivariant convolutional networks, in Proceedings of The 33rd International Conference on Machine Learning, M. F. Balcan, K. Q. Weinberger, Eds. (PMLR, New York, New York, USA), vol. 48 of Proceedings of Machine Learning Research (2016), pp. 2990–2999.

11. T. N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks. arXiv (2016), https://arxiv.org/abs/1609.02907.

12. V. G. Satorras, E. Hoogeboom, M. Welling, E(n) Equivariant Graph Neural Networks, in International Conference on Machine Learning (2021).

13. H. A. Makse, R. P. J. Perazzo, The thermodynamics of dyslexic learning. Inter. J. Neural Sys. **3**, 351–360 (1992).

14. M. M. Bronstein, J. Bruna, T. Cohen, P. Veličković, Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. arXiv (2021), https://arxiv.org/abs/2104.13478.

15. F. Wilczek, A Beautiful Question: Finding Nature's Deep Design (Penguin Press) (2016).

16. S. Weinberg, The Quantum Theory of Fields, vol. 2 (Cambridge University Press) (1995).

17. H. A. Makse, P. Boldi, F. Sorrentino, I. Stewart, Symmetries of Living and Artificial Intelligence Systems: Fibrations and Synchronization in Networks (Cambridge University Press, forthcoming) (2026), https://arxiv.org/pdf/2502.18713.

18. T. Gili, et al., Fibration symmetry-breaking supports functional transitions in a brain network engaged in language. arXiv (2025), https://arxiv.org/abs/2409.02674.

19. B. Avila, et al., Symmetries and synchronization from whole-neural activity in *C. elegans* connectome: Integration of functional and structural networks. Proc. Natl. Acad. Sci. USA **122**, e2417850122 (2025).

20. O. Velarde, L. C. Parra, P. Boldi, H. A. Makse, The role of fibration symmetries in Geometric Deep Learning. Proc. Nat. Acad. Sci. USA, in press (2026), https://arxiv.org/abs/2408.15894.

21. H. Kamei, P. J. A. Cock, Computation of balanced equivalence relations and their lattice for a coupled cell network. SIAM J. Appl. Dyn. Syst. **12**, 352–382 (2013).

22. A. Grothendieck, Technique de descente et théorémes d'existence en géométrie algébrique, I. Généralités. Descente par morphismes fidélement plats. Seminaire Bourbaki **190** (1959-1960).

23. P. Boldi, S. Vigna, Fibrations of graphs. Discrete Math. **243**, 21–66 (2002).

24. F. Morone, I. Leifer, H. A. Makse, Fibration symmetries uncover the building blocks of biological networks. Proc. Nat. Acad. Sci. USA **117**, 8306–8314 (2020).

25. V. Papyan, X. Han, D. L. Donoho, Prevalence of neural collapse during the terminal phase of deep learning training. Proc. Natl. Acad. Sci. USA **117**, 24652–24663 (2020).

26. D. Doimo, A. Glielmo, S. Goldt, A. Laio, Redundant representations help generalization in wide neural networks, in Advances in Neural Information Processing Systems, S. Koyejo, et al., Eds. (Curran Associates, Inc.), vol. 35 (2022), pp. 19659–19672.

27. M. Golubitsky, I. Stewart, Nonlinear dynamics of networks: the groupoid formalism. Bulletin of the American Meteorological Society **43**, 305–364 (2006).

28. H. Georgi, Lie Algebras in Particle Physics: From Isospin to Unified Theories (Taylor & Francis) (2000).

29. D. E. Rumelhart, G. E. Hinton, R. J. Williams, Learning representations by back-propagating errors. Nature **323**, 533–536 (1986).

30. L. DeVille, E. Lerman, Modular dynamical systems on networks. J. Eur. Math. Soc. **17**, 2977–3013 (2015).

31. E. Nijholt, B. Rink, J. Sanders, Graph fibrations and symmetries of network dynamics. Diff. Equ. **261**, 4861–4896 (2016).

32. P. Boldi, V. Lonati, M. Santini, S. Vigna, Graph fibrations, graph isomorphism, and PageRank. RAIRO - Theoretical Informatics and Applications **40**, 227–253 (2006).

33. H. Park, K. Friston, Structural and functional brain networks: from connections to cognition. Science **342**, 1238411 (2013).

34. P. Erdős, A. Rényi, Asymmetric graphs. Acta Mathematica Academiae Scientiarum Hungaricae **14**, 295–315 (1963).

35. F. Chen, D. Kunin, A. Yamamura, S. Ganguli, Stochastic collapse: How gradient noise attracts SGD dynamics towards simpler subnetworks. Advances in Neural Information Processing Systems **36**, 35027–35063 (2023).

36. D. Herzog, J. Mattingly, Noise-induced stabilization of planar flows I. Electronic Journal of Probability **20**, 1–43 (2015).

37. P. Boldi, I. Leifer, H. A. Makse, Quasifibrations of graphs to find symmetries and reconstruct biological networks. J. Stat. Mech.: Theor. Exp. **2022**, 113401 (2022).

38. F. Meng, et al., Pruning filter in filter, in Advances in Neural Information Processing Systems (Curran Associates, Inc.), vol. 33 (2020), pp. 17629–17640.

39. S. Vadera, S. Ameen, Methods for pruning deep neural networks. IEEE Access (2020).

40. N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: A simple way to prevent neural Networks from Overfitting. Journal of Machine Learning Research **15**, 1929–1958 (2014).

41. K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in Proceedings of the IEEE conference on computer vision and pattern recognition (2016).

42. Z. Wang, C. Li, X. Wang, Convolutional neural network pruning with structural redundancy reduction, in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (2021), pp. 14913–14922.

43. J. Qiu, C. Chen, S. Liu, H. Y. Zhang, B. Zeng, Slimconv: Reducing channel redundancy in convolutional neural networks by features recombining. IEEE Transactions on Image Processing **30**, 6434–6445 (2021).

44. J. Zbontar, L. Jing, I. Misra, Y. LeCun, S. Deny, Barlow twins: Self-supervised learning via redundancy reduction, in International conference on machine learning (PMLR) (2021), pp. 12310–12320.

45. P. W. Anderson, More is different: Broken symmetry and the nature of the hierarchical structure of science. Science **177**, 393–396 (1972).

46. G. M. van de Ven, T. Tuytelaars, A. S. Tolias, Three types of incremental learning. Nature Machine Intelligence **4**, 1185–1197 (2022).

47. A. Chebykin, A. Dushatskiy, T. Alderliesten, P. A. N. Bosman, Shrink-Perturb improves architecture mixing during population based training for neural Architecture Search. arXiv (2023), https://arxiv.org/abs/2307.15621.

48. M. D. Zeiler, R. Fergus, Visualizing and understanding convolutional networks, in European conference on computer vision (Springer) (2014), pp. 818–833.

49. N. Parga, M. A. Virasoro, The ultrametric organization of memories in a neural network. J. Phys. (Paris) **47**, 1857–1864 (1986).

50. M. Mézard, G. Parisi, M. A. Virasoro, Spin Glass Theory and Beyond, vol. 9 (World Scientific Lecture Notes in Physics) (1987).

51. M. A. Aguiar, A. P. S. Dias, F. Ferreira, Patterns of synchrony for feed-forward and auto-regulation feed-forward neural networks. Chaos: An Interdisciplinary Journal of Nonlinear Science **27** (2017).

52. I. Leifer, D. Phillips, F. Sorrentino, H. A. Makse, Symmetry-driven network reconstruction through pseudobalanced coloring optimization. J. Stat. Mech.: Theor. Exp. **2022**, 073403 (2022).

53. J. Schulman, F. Wolski, P. Dhariwal, A. Radford, O. Klimov, Proximal policy optimization algorithms. arXiv (2017), https://arxiv.org/abs/1707.06347.

54. N. Norris, Universal covers of graphs: Isomorphism to depth n-1 implies isomorphism to all depths. Discrete Applied Mathematics **56**, 61–74 (1995).

55. D. Elliott, S. Frank, K. Simaan, L. Specia, Multi30K: Multilingual English-German image descriptions, in Proceedings of the 5th Workshop on Vision and Language (Association for Computational Linguistics) (2016), pp. 70–74.

56. A. Vaswani, et al., Attention is all you need. arXiv (2023), https://arxiv.org/abs/1706.03762.

57. D. P. Kingma, J. Ba, Adam: A method for stochastic optimization. arXiv (2017), https://arxiv.org/abs/1412.6980.

58. B. O. Ayinde, T. Inanc, J. M. Zurada, Redundant feature pruning for accelerated inference in deep neural networks. Neural Networks **118**, 148–158 (2019).

59. K. Liu, M. Suganuma, T. Okatani, Bridging the gap from asymmetry tricks to decorrelation principles in non-contrastive self-supervised learning. Advances in neural information processing systems **35**, 19824–19835 (2022).

Acknowledgments

Funding and acknowledgments: Partial support for this work was provided by the National Institutes of Health through grants R01CA247910 (OMV, LCP, HAM) and R01EB028157 (HAM), as well as the Army Research Office with grant WF911-NF-24-1-0031 (OMV, LCP). We thank Matteo Serafino for help and discussions.

Competing interests: There are no competing interests to declare.