From Black Boxes to Colored Boxes: Fibration Symmetry Reveals the Geometry of Learning in Artificial Neural Networks

Osvaldo M. Velarde[1], Alireza Hashemi[1], Matteo Serafino[1], Lucas C. Parra[2], Hernán A. Makse[1,3,4*]

1 Levich Institute and Physics Department, City College of New York, New York, NY 10031
2 Biomedical Engineering Department, City College of New York, New York, NY 10031
3 Memorial Sloan Kettering Cancer Center, New York, NY
4 Neuroscience Program, Biology Department, Graduate Center, City University of New York, New York, NY

* hmakse@ccny.cuny.edu

Abstract

Deep neural networks are often regarded as powerful yet opaque black boxes. Here we show that learning in these networks creates symmetries in their computational graph. Unlike the global symmetries used in theoretical physics, these are local symmetries, characterized by identical input trees, referred to as "fibrations" and "coverings" which also have identical output tree. We show that covers are attractors of stochastic gradient descent and therefore emerge during learning. Graph "coloring algorithms" readily identify these emergent symmetries, which allows us to drastically compress the size of networks for most modern model architectures. Breaking local symmetries yields improvements in continual learning that outperform current state-of-the-art. The formation of fibers during learning uncovers shared features in the data. In this light, the process of learning transforms the black box network into a meaningfully "colored box". Our work suggests a new foundation for AI development, not rooted in brute-force scaling but in symmetries of the network structure.

1 Symmetries in learning

Despite the development of increasingly powerful neural network architectures, our understanding of how these networks learn and make inferences remains limited. 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 which argue that "bigger is better" [1, 2]: more parameters, more layers, more data and more compute.

The reliance on brute-force scaling as a substitute for understanding creates significant challenges. Training with single large runs has huge energy demands, and when trained sequentially, networks often suffer from a loss of plasticity over time [3]. The processing in these large models has been notoriously difficult to interpret, making it difficult to build trust in AI for critical applications. 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 [4], a subject of ongoing theoretical debate [5, 6].

To address these challenges, we propose a theoretical formalism based on symmetries in the internal structure of deep networks. We find that the computational graph promotes the emergence of local symmetries, which are less rigid than the well-known global symmetries, such as shift invariance in convolutional neural networks (CNNs) or permutation invariance in graph neural networks (GNNs). Although these global symmetries are fundamental to geometric deep learning and our understanding of physics [7–9], they are too restrictive to capture the diversity observed in biological and artificial neural systems [10–13]. We instead identify local symmetries that naturally emerge in the input trees of network nodes, which determine inference, and in the output trees, which determine the backpropagation of learning signals.

These local symmetries can be rigorously described using mathematical structures known as fibrations, opfibrations, and coverings, and are found with "balance coloring" algorithms from graph theory []. These structures were originally developed by Grothendieck in category theory [14] and later adapted for graphs [15, 16]. We prove mathematically that covering symmetries emerge from "synchronized learning" within stochastic gradient descent. This, in turn, induces fibration symmetries, which explains the emergence of redundant representations often reported in deep networks [17–19], and allows substantial compression of overparameterized models without compromising performance. We validate this phenomenon across a wide range of architectures, including multilayer perceptrons (MLP), recurrent networks (RNN), and Transformers in both supervised and reinforcement learning (RL) settings. Furthermore, by carefully breaking these symmetries, we can expand the network capacity to overcome loss of plasticity, demonstrating state-of-the-art performance in continuous learning [3].

Rather than treating neural networks as impenetrable black boxes, this symmetry analysis transforms them into a 'colored box' whose internal structure becomes interpretable through its invariant properties. Reducing the parameter space through symmetry reveals that over-parameterized models trained by gradient descent create structured redundancy, minimizing the model to its most efficient computational base.

Our results demonstrate that fibration symmetry provides a unifying geometric framework for understanding and improving deep learning architectures. Identifying and breaking these symmetries offers a novel mechanism for controlling inductive bias and a principled approach to designing efficient neural networks. This opens a path toward scalable, structure-aware, and lifelong-learning AI systems guided by first principles rather than brute-force computation.

2 Computational Graphs in Deep Learning

The symmetries we will discuss emerge in a variety of network structures like MLPs, CNNs, RNNs and Transformers, but they are most easily understood in the canonical MLP. In this architecture, nodes are organized into multiple dense layers ($l = 1, ..., N$), with weight $W_{ik}^{(l)}$ connecting node k to i from layer $l-1$ to l. The weighted edges $W_{ik}^{(l)}$ create the computational graph that is the basis of our fibration analysis (Fig. 7a). Each node i in the network performs a weighted sum of its inputs k, followed by a point-wise nonlinearity σ,

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

where $h_i^{(0)} = x$ is the input of the network.

The activity propagates "forward" across layers from the input x to the output. During learning, the gradient of a loss function $\mathcal{L}$ with respect to weights $W_{ik}^{(l)}$ can be

calculated with the error backpropagation algorithm, with the error signal $\delta_i^{(l)}$ flowing "backward" through the computational graph [20],

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

$$\delta_i^{(N)} = \frac{\partial \mathcal{L}}{\partial h_i^{(N)}} \qquad \text{Output layer}\,. \tag{3}$$

where $\delta_i^{(N)} = \frac{\partial \mathcal{L}}{\partial h_i^{(N)}}$ is the error evaluated of the output layer.

Next, we will describe the local symmetries that emerge in this MLP network. For an easy visualization of this hierarchy, consider an MLP with binary weights (where connections are 1 or 0), as shown in Fig. 1. In the figure, nodes are colored to indicate their symmetry class according to various symmetries. We will generalize these concepts to continuous-valued weights in Eq. (**??**).

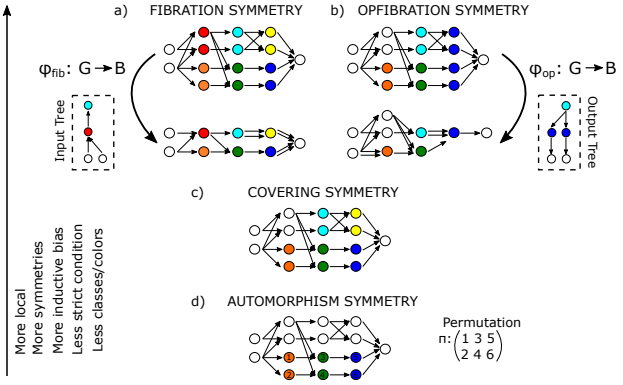

Figure 1. Hierarchy of symmetries in a sample layered feedforward network with binary edges for simplicity. Weights are binary, and edges with zero weight are not plotted. GDL is based on the most restrictive types of global symmetry groups or automorphisms. An automorphism permutation is exemplified in the case of a graph at panel (d). It identifies three pairs of symmetric nodes by a permutation (they are called orbits). The proposed framework is based on the generalization to local symmetries, such as (op)fibrations and coverings (see panels a-c). These symmetries contain the orbits but also identify other symmetric nodes. This increases the inductive bias with less strict conditions and more symmetries, resulting in more compression with less balanced colors (in these examples, blank nodes should be interpreted as distinct colors).

A *fibration symmetry* between nodes exists when the nodes have isomorphic "input trees." This means that the entire structure of the connections and colors from the input to the nodes is the same. An example is shown in Fig. 1a). For instance, the two cyan nodes have the identical input tree, as shown to the left. When two nodes have isomorphic input trees, then they belong to the same *fiber* (represented by the same color). We identify fibration symmetries using a balanced coloring algorithm from graph theory (see Ch. 13 in [**?**]). This algorithm partitions the network into color classes C_{fib} by assigning the same color to nodes that receive the exact same set of input colors [**?**], hence called 'balanced'.

Once identified, these fibration symmetries allow the full network graph G to be compressed into a smaller "base" graph B_{fib}. This compression $\varphi_{fib} : G \to B_{fib}$ works by merging all nodes that share the same color (i.e. belong to the same fiber) while adding the corresponding weights. Crucially, this fibration compression conserves the

inputs (this is called the *lifting property* [21]). Namely, the resulting base graph B_{fib} performs the exact same "forward" computation as the original graph G, which means that we can compress the network without changing its dynamic.

A similar principle applies to the backward propagation of the error signal during training. An *opfiber symmetry* occurs when nodes share an identical "output tree" — the structure of connections leading from them to the output layer (Fig. 1b). Nodes with this symmetry receive the same error signal. This allows for a similar compression $\varphi_{op} : G \to B_{opt}$ that preserves the output trees as shown on the upper right in the figure.

When nodes have both, isomorphic input and output trees, they form a *covering symmetry* (Fig. 1c). The corresponding coloring C_{cov} is a refinement (or intersection) of C_{fib} and C_{op}, i.e. a finer partition of the graph. Clearly, a covering has a more stringent symmetry than fibers or opfibers.

The most stringent symmetry is the *automorphism*. This is a global permutation of the network's nodes that leaves the entire graph's connectivity unchanged, ie, it is a global symmetry in contrast to 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 a cornerstone of geometric deep learning (GDL) [7] and theoretical physics [**?**, **?**], this symmetry is so restrictive that we have never observed it in our trained networks.

In contrast, we will show here that covering symmetries emerge naturally from stochastic gradient descent and demonstrate empirically that fibration symmetries are ubiquitous in trained networks, allowing for substantial model compression and computational savings.

In summary, these symmetries form a hierarchy of increasing strictness: from fibrations and opfibrations, to coverings, and finally to automorphisms. As the conditions become more stringent, there are more distinct color classes and fewer nodes within each class (meaning fewer symmetries). Less strict, local symmetries like fibrations are more common and allow for 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.

3 Fibers in a weighted graph

Generalizing to an MLP with continuous weights, and more formally, two nodes i and j in layer l belong to the same fiber (denoted $i \underset{fib}{\sim} j$) if and only if (iff):

- **Fibration symmetry:**

$$i \underset{fib}{\sim} j \iff \forall c \in \mathcal{C}^{\text{fib}}_{\ell-1} : \quad \sum_{k \in c} W^{(\ell)}_{ik} = \sum_{k \in c} W^{(\ell)}_{jk}$$

This criterion partitions the nodes in the layer ℓ into a coloring $\mathcal{C}^{\text{fib}}_{\ell}$ based on input weights $W^{(\ell)}_{ik}$ and the color partitioning of the previous layer $\mathcal{C}^{\text{fib}}_{\ell-1}$. 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. This recursive definition starts in the input layer, $\ell = 0$ with all nodes with distinct colors $\mathcal{C}^{fib}_0 = \{1, 2, ..., d_0\}$, i.e. trivial fibers. This definition of equivalence based on the sum of weights appears in the literature in the context of node synchronization [22, 23].

Analogously, two nodes belong to the same opfiber iff:

- **Opfibration symmetry:**

$$i \underset{op}{\sim} j \iff \forall \hat{c} \in \mathcal{C}^{\text{opf}}_{\ell+1} : \quad \sum_{k \in \hat{c}} W^{(\ell+1)}_{ki} = \sum_{k \in \hat{c}} W^{(\ell+1)}_{kj}$$

The starting condition is the last layer with all nodes in different opfibers $\mathcal{C}^{op}_N = \{1, 2, ..., d_N\}$.

Finally, two nodes belong to the same covering iff they are in the same fiber and same opfiber:

- **Covering symmetry:**

$$i \underset{cov}{\sim} j \iff i \underset{fib}{\sim} j \quad \& \quad i \underset{op}{\sim} j$$

If G is the MLP computational graph, then the map $\varphi_{fib} : G \to B_{fib}$ generates a smaller MLP with the MLP computational graph B_{fib}. The weight matrices $\hat{W}^{(l)}$ of B_{fib} can be calculated as a function of the weight matrices $W^{(l)}$ of G, using the following equation:

$$\hat{W}^{(l)}_{c'c} = \frac{1}{|c'|} \sum_{i \in c', k \in c} W^{(l)}_{ik}. \tag{4}$$

where $c \in \mathcal{C}^{\text{fib}}_{l-1}$, $c' \in \mathcal{C}^{\text{fib}}_{l}$ and $|c'|$ is the number of nodes in the fiber c'. This transformation of the weights is a representation of the map φ_{fib} that preserves the forward computation (see the proof in Section 7.4). Similarly, there is a transformation of the weights that represents φ_{op}. Moreover, this procedure for finding parameter transformations that represent φ_{fib} and φ_{op} can be extended to other architectures (see Section 7.4).

With these definitions, we demonstrate in the next section that fibration symmetry causes nodes to have the same activity, a phenomenon we call *activity synchronization*, while opfibration symmetry causes *error synchronization* in error backpropagation, the standard gradient descent algorithm used to train neural networks. With this we can then show that stochastic gradient descent serves as a mechanism for covering symmetry formation, where the nodes converge to have identical activity and error, leading to *synchronized learning*.

4 Theoretical results

When a neural network is initialized, its weights are typically set to random values. Large random graphs of this kind rarely have significant global automorphism symmetries [24].

However, the training process itself appears to create symmetry. For example, recent studies have observed that stochastic gradient descent (SGD) causes different nodes to develop similar input and output weights [25]. We have also previously observed the emergence of *synchronized* nodes – nodes that have identical activity across all inputs during training [13].

Here, we provide a theoretical proof that links these empirical observations. We prove that stochastic gradient descent naturally produces covering symmetries from any random starting condition. Our strongest result is the discovery of the covering invariant set: once two nodes become part of the same covering symmetry at a specific training epoch t, they remain in that symmetric relationship for all subsequent training times $t' > t$ (see Theorem 4.3). In other words, once a covering symmetry is formed, it's permanent.

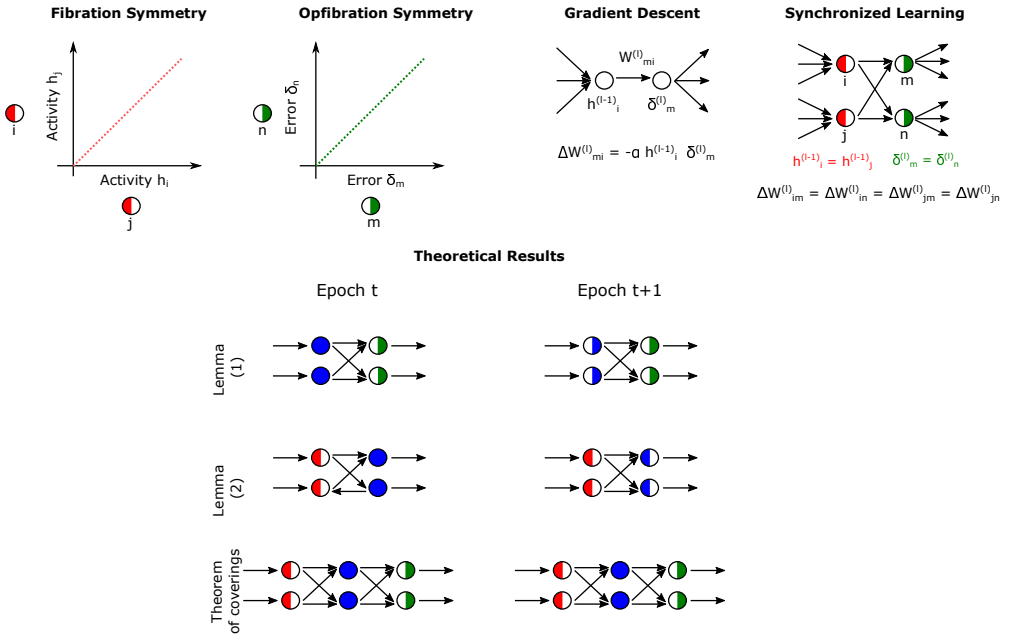

Figure 2. Theoretical results. Nodes in the same fiber (nodes with red left sides) synchronize their activities (i.e. $h_i = h_j$, see Lemma 4.1). Nodes in the same opfibers (nodes with green right sides) synchronize their error signals (i.e. $\delta_n = \delta_m$, see Lemma 4.1). The weights of connections between nodes in a fiber and nodes in an opfiber update with the same weight updates ΔW. In Lemma 4.2, we present this phenomenon called *synchronized learning* for Gradient Descent. Our Lemma 7.1 says: (1) Nodes in the same covering fiber (blue nodes) at epoch t will belong to the same opfiber at the next epoch $t+1$, if their output nodes are on the same opfiber. (2) Nodes in the same covering fiber at epoch t will belong to the same fiber at the next epoch $t+1$, if their input nodes are on the same fiber. Both lemmas are useful for proving Theorem 4.3, which ensures that nodes in the same covering at epoch t remain in the same covering in the subsequent epoch $t+1$.

4.1 Synchronization by fibration and opfibration symmetries

Similarly to conservation theorems in physics, these symmetries imply the synchronization of a specific network variable. Fibration symmetry induces activity synchronization during forward inference, and opfibration symmetry induces error signals synchronization during error backpropagation. This is the subject of the following theorem:

Theorem 4.1 ***Synchronization***. *Given two nodes i and j in layer l, the following holds:*

1. *Fibration symmetry implies synchronization of activities:* $i \underset{fib}{\sim} j \implies h_i^{(l)} = h_j^{(l)}$
2. *Opfibration symmetry implies synchronization of errors:* $i \underset{op}{\sim} j \implies \delta_i^{(l)} = \delta_j^{(l)}$

The proof of point (1) of Theorem 4.1 is in Methods 7.5 based on [26, 27]. The proof of point (2) is analogous to that of point (1), but uses the output trees and the error signal in the final layer instead (see Methods 7.6).

Conceptually, nodes in the same fiber are redundant, they synchronize and can be combined by fibration compression φ_{fib} so that their summed input weights preserve

their contribution to the next layer during the forward pass. Similarly, nodes in the same opfiber can be combined by opfibration compression φ_{op}, with their output weights summed to maintain their contribution in the previous layer during the backward pass.

4.2 Stochastic gradient descent induces covering symmetries

First, we analyze how the weight update of the gradient descent algorithm locally depends on node activity and error signals [20].

$$\Delta W_{mi}^{(l)} = -\alpha \frac{\partial L}{\partial W_{mi}^{(l)}} = -\alpha \delta_m^{(l)} \cdot h_i^{(l-1)} , \tag{5}$$

where α is a learning rate (see the derivation in Methods 7.7).

According to Theorem 4.1 nodes i, j of the layer $l-1$ 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. Then, we can prove the following.

Theorem 4.2 ***Synchronized learning***. *Given nodes $i \underset{fib}{\sim} j$ in layer $l-1$ and $m \underset{op}{\sim} n$ in layer l, the weight updates according to gradient descent are synchronized:*

$$\Delta W_{mi}^{(\ell)} = \Delta W_{ni}^{(\ell)} = \Delta W_{mj}^{(\ell)} = \Delta W_{nj}^{(\ell)} . \tag{6}$$

We term this phenomenon *synchronized learning* (see proof in Methods 7.9). Note that, unlike the synchronization of activations and errors in Theorem 4.1, this synchronization occurs in the parameter space.

As a consequence of *synchronized learning*, once two nodes are in a single covering, the weight update with the gradient descent algorithm keeps the nodes in the covering. This is formalized in the following Theorem:

Theorem 4.3 ***Conservation of coverings.*** *If the set of network parameters at epoch t is θ_t and it is such that $p \underset{cov}{\sim} q$; then θ_{t+1} is such that $p \underset{cov}{\sim} q$ still holds. That is, $p \underset{cov}{\sim} q$ is invariant under GD dynamics.*

We proof this theoretical result in Methods 7.10 and it implies that once multiple nodes are in the same covering fiber, they cannot exit that covering fiber ever. A summary of all our theoretical results is presented in Fig. 2.

Chen *et al.* have shown that any invariant set is an attractor in the stochastic gradient descent algorithm, provided a sufficiently large learning constant [25]. The intuition is that stochastic fluctuations during learning occasionally bring nodes into an invariant set; once there, the gradient descent algorithm can no longer break the symmetry due to synchronized learning. In Section 5.1 we provide empirical evidence that our *Covering Invariant Set* exhibits the same attracting behavior. This is the fundamental reason for "node collapse" that has been previously reported empirically, in particular in the last layers of deep networks [17–19], and has been attributed to a permutation invariant set [25]. We will empirically validate this claim for the much less restrictive covering symmetry, which will be found much more abundantly than the more restrictive permutation symmetry. This will be true in different network architectures and learning tasks, as is expected from the generic derivation of our theorem as discussed in Section 7.12.

We will also demonstrate empirically that the collapse of nodes into these symmetries limits the network's ability to learn new data. This will serve as the foundation for our new proposed learning rules based on symmetry breaking in Section 5.2.

On the flip-side, for a fixed task, at the end of training, nodes in a fiber are redundant and can be pruned from the network leaving a single node behind that serves the function for the whole fiber, i.e. the fibration base. We will demonstrate in Section 5.1 that this allows for substantial compression of the network without substantially changing its function. Finally, pruning fibers during learning will allow us to achieve high performance on a fixed task with much smaller networks. We will demonstrate that this targeted fiber pruning outperforms existing pruning methods in the literature.

5 Empirical results

In the proof of Theorem 7.5, a Lemma is used which also provides an intuition for how fibration and opfibration symmetries develop during learning. Lemma 7.1 states that fibers in layer ℓ preserve fibers in layer $\ell + 1$, while opfibers preserve opfibers in layer $\ell - 1$, as shown in Fig. 2b). Thus, opfibers tend to propagate backward during learning starting from the output layer, while fibers tend to propagate forward from the input. This empirical finding is demonstrated in Fig. **??**.

5.1 Fibration collapse preserves the network's performance

Supplementary video S1 and Fig. 7 show how activity and error synchronization arises from the fibers and opfibers, respectively, in a neural network. During training, coverings are created that act as attractors for stochastic gradient descent. When the fibers stabilize, the network has learned the training set.

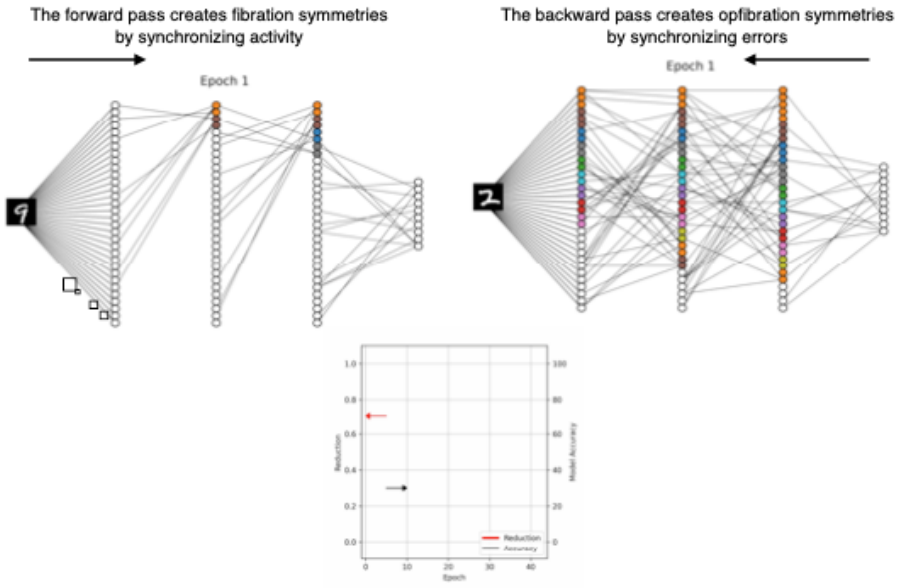

Figure 3.

Figure 4 shows a fibration-based analysis in a linear multilayer perceptron (MLP) trained on MNIST. Applying the collapse procedure, i.e., merging parameters along equivalence classes identified by the fibration, reduces the model to 17% of its original size (measured by parameter count) with no appreciable loss of classification accuracy. We observe comparable compressibility in Transformer architectures, indicating that the

emergence of fibers is not specific to feedforward networks. Consistent behavior is also obtained for CNN and MLP models trained on ImageNet and CIFAR, where fibration-guided compression achieves reductions of similar magnitude without degradation in test performance (Fig. 4).

We further assess the approach in a control setting with sequential decision-making. In the Beam Rider reinforcement-learning benchmark, a network comprising two convolutional layers and max-pooling, three MLP blocks, and one LSTM attains a final compression of approximately 30% of the original model size while maintaining performance as quantified by the game score (Fig. 4). Over the course of 300 training epochs, the fraction of parameters organized into fibers (blue curve) increases monotonically to roughly 60% of the original size. This growth closely tracks the rise in episodic return, and both quantities plateau concurrently, indicating that fiber/cover formation is coupled to the stabilization of the learned policy. At episode 50, we trigger an intermediate collapse step that yields an overall ∼70% reduction relative to the initial model (i.e., retaining ∼30% of parameters; green curve) with no meaningful decline in score is observed following the collapse.

These reductions in the number of parameters imply proportional savings in memory footprint and operations (FLOPs), which we observe empirically to translate into faster inference without compromising task metrics. Together, these results indicate that fibration-induced symmetries emerge during training across diverse architectures and tasks, and that exploiting them via collapse yields substantial compression at essentially unchanged performance.

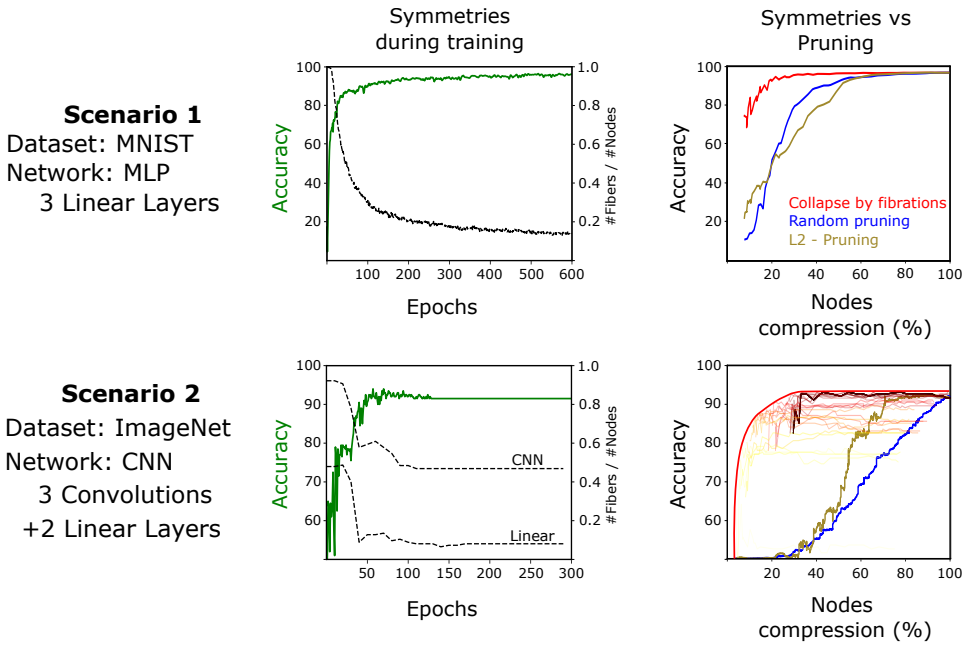

Figure 4. Empirical results in DNNs. Compression collapse and symmetry breaking in different architectures. (a) Collapsing MLP. Compression analysis in a MLP. The architecture can be compressed to 17% without major loss of performance. The study is performed as a function of the threshold ε used to define the balanced coloring. (b) Collapsing transformer. (c) Collapsing CNN, LSTM and MLP in RL.

In addition, we will use ResNet-18 that adds residual connections between convolutions to investigate their relevance to fibration formation (we expect residuals to act as a symmetry breaking mechanism).

5.2 Mechanisms of symmetry breaking

Strictly speaking a *fibration symmetry* is the map $\varphi_{fib} : G \to B$ that collapses the in-balanced color fibers in the original graph G into a unique node in the base B while preserving the inputs of every node. This is known as the *lifting property by fibration* (see Fig. 1a and Methods Section 7.4). The origin of the name "lifting" is the inverse operation of compression in Fig.1, where the base is "lifted" to a full graph (see Methods 7.4). Lifting is not unique, and we will use one such lifting operation as the basis for targeted symmetry breaking.

A *covering symmetry* is a map that collapses each covering fiber in G into a single node in the base B. However, there are two ways to collapse the edges in this map, either by fibration lifting or by opfibration lifting. We will use the former since it preserves the memory of the tasks already learned.

Randomly initialized networks admit nontrivial local symmetries across several architectures which naturally arise during training under SGD. These symmetries lead to redundancies (i.e., multiple neurons or pathways learn nearly identical representations), which eventually reduces the network's effective dimensionality, constraining its capacity for a given model size. However, traditional regularization mechanisms (i.e., L1, L2 regulations, dropout [28], weight decay, random perturbation [29], alternative initializations (e.g. orthogonal), skip connections) tend to promote symmetry breaking that breaks fibers. However, these mechanisms operate in an indiscriminate manner: they break symmetries randomly, often destroying not only redundant structures but also critical fibers that SGD has already consolidated into the learned representation.

Instead, the fibration offers a principled alternative to break the symmetries by enabling a surgical intervention directly to the fibers that avoids eliminating structure indiscriminately. We selectively preserve the fibers essential to the representation by collapsing the system to its base while pruning and randomizing only the redundant parameters in the fibers. In this way, the learned symmetries remain intact while excess parameterization is systematically reduced.

Building on the theoretical insights presented in the previous section, we develop a symmetry-breaking protocol designed specifically to preserve current input-output map while allowing for the optimal randomization of present weights to learn new data.

This breaking process occurs in two steps (Fig. 5):

1. According to Theorem 4.1, collapsing a fiber using the fibration symmetry map does not alter the input-output map (inference process) as illustrated in Fig. 5a. This step represents a reduction in the dimensionality of the ANN that maintains intact the previous learning capabilities and liberate the rest of the neurons in the fiber to participate in learning new data.

2. The nodes that remain in the fiber are considered redundant and can be utilized for new data. Consequently, training continues by first lifting the base of the covering by fibration and randomizing the input weights of these redundant neurons as shown in Fig. 5b. The out-going edges of the collapsed neurons (except the one at the base) are initialized to zero to ensure that the existing input-output map at the base is not disrupted.

This symmetry breaking theoretically guarantees that the network does not forget what it has learned while being allowed to explore the learning of new data. By managing this formation–breaking cycle, new capacities can be learned continuously realized by controlled symmetry breaking.

5.3 Loss of plasticity in continual learning

The reduced degrees of freedom of covering symmetries can be beneficial in learning a single task by reducing overfitting. However, they become a bottleneck when the network needs to adapt to a new task. Recent work in deep neural networks has documented the inability of a trained network to learn new tasks [3]. The source of this loss of plasticity is not well understood, and there is not yet a clear solution [30]. Our results provide a theoretical explanation for this loss of plasticity, and suggest a solution: breaking symmetries to promote functional diversity in the network. In particular, we propose to disrupt fibration symmetries formed during stochastic gradient descent explicitly. The theoretical framework behind these methods unifies ideas from previous work, including Deep Continual Learning [3], Random Perturbation [29], and Dropout [28], where empirical studies demonstrated the preservation of plasticity. This novel approach establishes a new research direction for studying symmetry in learning rules, particularly for problems requiring adaptive decision making, mirroring human cognitive processes.

The loss of plasticity can be analyzed through the lens of fibration symmetries in parameter space. Once gradient descent steers the network into a configuration exhibiting such symmetries, the induced covers persist and constrain subsequent updates. The system then behaves as if it were optimized on the base, reducing the accessible degrees of freedom and impeding the formation of new representations. By analogy with condensed-matter systems, where qualitatively new phases and response properties emerge through symmetry breaking [31], we show that controlled breaking of fibers restores the missing degrees of freedom and thereby reactivates plasticity. This perspective aligns with recent neurocognitive evidence suggesting that language processing in the human brain can be understood as a broken fibration of the resting-state architecture [32], enabling flexible, task-dependent reconfiguration without loss of prior capabilities.

Empirically, we observe that the fibration symmetry-breaking mechanism delivers consistent gains in continual-learning benchmarks (see Methods Section 7.13 for details). Figure 5c show that our approach achieves significant improvements surpassing state-of-the-art baselines reported in [33]. These benefits are obtained without compromising stability: once new representations are established, coverings can be selectively reconstituted to recover compression and inductive bias, preserving performance on both past and newly learned tasks.

The proposed symmetry-breaking interventions unify and systematize several strands of deep continual-learning practice, including stochastic regularization (e.g., dropout [28]), redundancy reduction, noise- or perturbation-based exploration (shrink–perturb [29]), and methods that reduce representational redundancy.

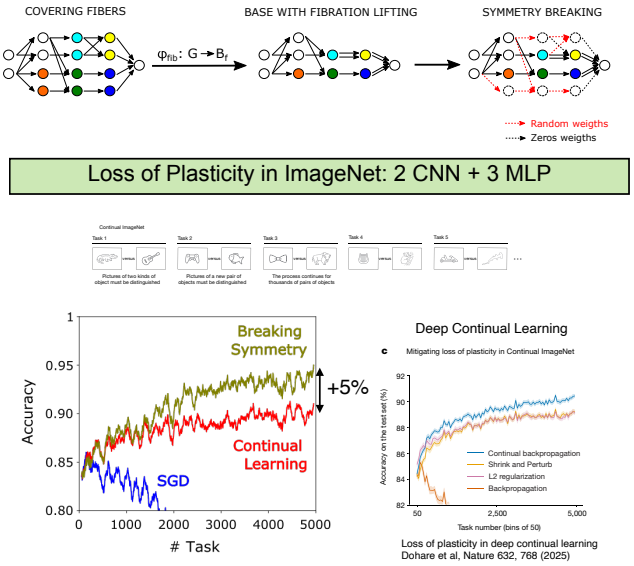

Figure 5. Breaking of symmetry. Two optimized steps **(a)** Collapse to the covering base via fiber-lifting to preserve the old task. **(b)** Randomizing redundant nodes. Notice that nodes in the base are the same since we collapse the same network, but the edges are different according to preservation of inputs or outputs. (c) Continual learning results.

6 Conclusion

Building upon the fibration framework, we propose a modern reinterpretation of Hebbian learning for artificial neural networks. The classical formulation for brain plasticity, "neurons that fire together wire together", captures the essence of biological synaptic strengthening through co-activation. However, in artificial systems trained via backpropagation, error signals play a crucial role that classical Hebbian theory does not need to address.

Synchronized learning leads to the formation of coverings which are functional units where neurons with shared representational and error roles are grouped together via symmetry. This leads to a new artificial Hebbian rule: "Neurons that fire together wire to neurons that err together".

This formulation captures the dual directionality of information flow in deep learning— forward activations and backward error propagation— and highlights that learning is not merely about correlation of activity, but about coordinated alignment of activation and error. By integrating both, the network aligns representational and optimization geometry, promoting efficient learning and avoiding redundancy.

Fibration theory provides a principled explanation for several long-standing empirical results in deep learning. It accounts for representation collapse by suppressing redundant neurons, elucidates the success of transfer learning by showing how randomized outputs propagate structural regularities backward through the network, and explains the effectiveness of weight tying—as seen in CNNs and transformers—as a bias toward symmetry formation and generalization.

Osvaldo: conclusions: here we need to address sync=fibers Figure 6

Discussion: our results are for random inputs, random colors in the input layer. Single class sync results in this figure are different, but they are related. Explain this difference and the meaning of Theorem 4.1.

Also, we need a nice explanation of the coarsening phenomena of covers/sync clusters we discussed in Fig. 6 and the energy-loss landscape. This will be a good conclusion for the paper.

Osvaldo. Here, we can display our picture of the energy landscape and the coarse-graining process under SGD. It is somehow related to the discussion of overparametrization since this is the way the network reduced the parameter size. So, we could mention here the figure of the energy landscape in W, which could be put into Fig. 3, since it is a nice theoretical way to understand the process of coarse-graining in the space of parameters.

Overparametrized ANNs admit rich covering symmetries that synchronize redundant neurons. These symmetries compress the representation after training, meaning the effective model that emerges is simpler than the raw parameter count suggests. Thus, once the network is sufficiently large, gradient descent finds structured symmetric solutions that generalize well, resolving the paradox of improved performance in overparametrized models. We also shed light on why deep networks outperform shallow ones despite their greater parameter count: depth facilitates the gradual formation of coverings layer by layer, a process that is statistically improbable in shallow networks.

Lucas: We need to address other pruning methods here in teh conclusions. I think this might be the place for discussing our results in terms of pruning.

We need to address the other methods of pruning, but with a twist. Our main differentiator is that we have a theory. We are pruning by theory, not randomly. So, the other methods, when they are introduced, they typically need to show that they are superior in some form to previous methods, with a comparison with everything else. So, we need to address the other methods, but not by performing a comparison with everything else. This is because our method is based on theory. Their methods are based on randomness. That is, we pick targets to compress, they go at random. In our case, even if our compression is not that much in comparison, everyone will need to go through our compression first. In a sense, the best compression method could be a composite our fibration + something else random. But in our case, we always guarantee to work since it is based on theory not trial and error. We need to address here the Lottery ticket paper.

We should emphasize that our method is independent on the dataset used to train. For instance, the Lottery ticket method depend on teh dataset, so if you try it with another class of samples it may not work. Our method works at the level of weight architecture so it is more general.

6.1 Osvaldo explanation

The main theorem of this paper establishes that Stochastic Gradient Descent (SGD) tends to form coverings in deep neural networks. Furthermore, we identify a relationship between the definition of fibration symmetries (see Eq) and clusters of synchronized nodes. The emergence of such synchronized nodes is a well-known phenomenon in deep neural networks.

In panels (a) and (b), we show the state of the network in three different training stages $(\mathrm{T}_1, T_2, T_3), and the evolution of the three nodes (p, q, z)$.

(a) shows the training trajectory in parameter space. Colored zones in this space are regions where the network has the same covering base. Networks sharing base will work identically and, consequently, have the same loss value. In other words, the loss function is constant across these regions. At time (1), the three nodes p, q, and z belong to distinct covers. Suppose that at a later moment, p and q converge into the same cover. According to the theorem, once this occurs, they can never separate again (as shown at 2 and 3). In other words, the first moment p and q share a cover defines an irreversible

transition (see green line T1-2). Similarly, an equivalent transition occurs when node z merges into the cover containing p and q, defining a second transition line T23.

(b) In the node space, we mark different covers in yellow. For a binary classification problem (red class and blue class), the red lines (resp, blue) delineate the synchronization clusters when the input belongs to the red class (resp, blue). By definition, a cover must be contained within a fiber. Furthermore, we know that two nodes belong to the same fiber if and only if they synchronize for any input. Therefore, they must, at a minimum, be synchronized when the input belongs to a particular class (red and blue). This implies that the nodes within a cover cannot lie in different class-specific synchronization clusters. In our plot, this means the yellow areas (covers) cannot cross the blue or red lines (class-specific clusters). Our theorem establishes that training progressively merges covers (the yellow zones) and that these mergers are irreversible. An example of a layer with 10 nodes is shown on the right. The merging of covers is represented as a tree, depicted by the yellow blocks.

(c) Intuitively, one expects the network (e.g MLP) to perform better on in-distribution data, such as MNIST digits, than on random pixel combinations. This suggests the existence of local minima in the loss landscape around the training inputs.

For each class and for random inputs, we can identify clusters of synchronized nodes based on their correlation matrices. Defining these clusters requires setting a correlation threshold (ranging from 0 to 1, where 0 represents the strictest synchronization condition). In contrast, identifying fibers based on the weights of the network also requires a threshold ϵ (ranging from 0 to 2, where 0 represents the strictest fibration symmetry).

We then investigated the alignment between these two structures. In the left panel, we compare the matching between fibers and clusters derived from random inputs. In the right panel, we assess whether the fibers constitute a refinement of the class-specific clusters.

Our work frames the learning process of deep neural networks as a transition from disorder to structural symmetry, a transformation from a black box into a colored box filled with balanced coloring symmetries. What was once seen as opaque and unstructured becomes, through fibration theory, a system endowed with inherent structured symmetry, interpretability and elegance.

Ultimately, our framework is theory-driven: fibration symmetry and breaking give explicit rules for designing inductive biases and architectures that generalize, rather than relying on ad-hoc heuristics like dropout, transfer learning, or fine-tuning. This perspective enables artificial systems where theory and structure, not brute-force scaling, drive performance.

7 Methods.

7.1 Deep Neural Networks

Here we define the major network architectures considered in this study. All theorems explained apply to them.

7.1.1 Multilayer Perceptron (MLP)

Let an MLP with $N-1$ hidden layers be described by the following equations.

$$\begin{aligned} z^{(l)} &= W^{(l)} h^{(l-1)} + b^{(l)}, \\ h^{(l)} &= \sigma(z^{(l-1)}), \\ \hat{y} &= h^{(N)}, \end{aligned}$$

where $z_i^{(l)}$ and $h_i^{(l)}$ are the input and activity of the node $i = 1, ..., d_l$ in the layer $l = 1, ..., N$, respectively. The vector $b^{(l)} \in \mathbb{R}^{d_l}$ is called bias of the layer l and the matrix $W^{(l)} \in \mathbb{R}^{d_l \times d_{l-1}}$ is the weight matrix from layer $l-1$ to layer l. σ is a non-linear activation and $\hat{y}$ is the prediction of the network.

7.1.2 Convolutional Neural Network (CNN)

Let a deep convolutional network with N convolutions and an average pooling,

$$\begin{aligned} \mathbf{H}^{(l)} &= \mathbf{H}^{(l-1)} * \mathbf{W}^{(l)} + \mathbf{b}^{(l)}, \\ \hat{y} &= \text{AvgPool}(\mathbf{H}^{(N)}) \in \mathbb{R}^{d_N} \end{aligned}$$

where $\mathbf{W}^{(l)} \in \mathbb{R}^{k \times k \times d_{l-1} \times d_l}$, $\mathbf{b}^{(l)} \in \mathbb{R}^{d_l}$, and $\mathbf{H}^{(l)} \in \mathbb{R}^{L \times L \times d_l}$ for all $l = 1, ..., N$.

7.1.3 Recurrent Neural Networks (RNN) - LSTM

$$\begin{aligned} \text{Input gate:} \quad & i_t = \sigma(W_{hi} h_{t-1} + W_{ii} x_t + b_i) \\ \text{Forget gate:} \quad & f_t = \sigma(W_{hf} h_{t-1} + W_{if} x_t + b_f) \\ \text{Output gate:} \quad & o_t = \sigma(W_{ho} h_{t-1} + W_{io} x_t + b_o) \\ \text{Cell gate:} \quad & g_t = \tanh(W_{hg} h_{t-1} + W_{ig} x_t + b_g) \\ \text{Cell state:} \quad & C_t = f_t \odot C_{t-1} + i_t \odot g_t \\ \text{Hidden state:} \quad & h_t = o_t \odot \tanh(C_t) \end{aligned}$$

where:

- x_t input at time t,
- $\odot$ element-wise multiplication.

7.1.4 Attention modules and Transformers

Transformers are architectures that uses the attention mechanism to process sequential data. The attention mechanism consists of

$$\begin{aligned}
\text{Query, Key, Value:} \quad & \mathbf{Q} = \mathbf{X}\mathbf{W}^{Q}, \quad \mathbf{K} = \mathbf{X}\mathbf{W}^{K}, \quad \mathbf{V} = \mathbf{X}\mathbf{W}^{V} \\
\text{Score:} \quad & \alpha = \frac{\mathbf{Q}\mathbf{K}^{\top}}{\sqrt{d_k}} \\
\text{Context vector:} \quad & \mathbf{z} = \text{softmax}(\alpha)\mathbf{V}
\end{aligned}$$

where:

- $\mathbf{X} \in \mathbf{R}^{T \times N}$: Embedding of tokens.
- $\mathbf{W}^{Q}, \mathbf{W}^{K}, \mathbf{W}^{V}$: Weights
- d_k: Key dimensions

Typically, transformers are a concatenation of encoders and decoders consisting of a focus module, feedforward networks, and residual connections. The transformer design will depend on the task at hand.

7.1.5 Reinforcement Learning (RL)

In Deep Reinforcement Learning, the Actor-Critic method features two components that often share a common initial structure. The base network processes the environment's state s to extract representative features, which then branch into two specialized heads: the *Actor head*, which outputs a probability distribution $p(a)$ over actions a (deciding "what to do"), and the *Critic head*, which estimates the state value V (judging "how good it is to be here").

This full architecture called *Agent* interacts with the environment in a continuous cycle: the Actor samples and executes an action based on its learned policy $s \mapsto a$; the environment responds with a reward and a new state; the Critic then assesses the outcome using the value $s \mapsto V$ to calculate a temporal difference error; this error, in turn, guides the update of both networks, pushing the Actor to refine its policy toward more rewarding actions and helping the Critic improve the accuracy of its value predictions.

In our case, the agent utilizes a network composed of a sequential stack of convolutional layers, dense layers, and an LSTM. The Actor then transforms the LSTM's output into a probability distribution over possible actions using a linear projection followed by a Softmax activation. Simultaneously, the Critic estimates the state value by applying a simple linear projection directly to the same LSTM output.

7.2 Global and Local symmetries in graphs: coverings, fibrations, and opfibrations

Understanding how symmetries emerge and how they are broken provides a deep understanding of the underlying principles that govern the dynamics of networks [34, 35]. Usually, the formation of symmetries arises from optimization processes [36, 37]; conversely, breaking symmetries leads to diversity, complexity, and functionality in systems [38, 39]. The framework for analyzing these symmetries is provided by graph theory.

In [13], we defined the symmetries of a directed graph as transformations that preserve the particular properties of a graph. These transformations can include merging, splitting, or permutation of nodes, and the properties to be preserved can be more or less strict. For example, a *fibration symmetry* is a transformation that preserves the input trees. A *covering symmetries* preserves both the input and output

tree; finally, the strictest level of symmetry is the *automorphism symmetry* which preserves all connections between nodes.

A directed graph G consists of a set N_G of nodes and a set A_G of edges. Each edge is associated with a source and a target node, and one way to represent the full structure of the graph is its adjacency matrix. For every node $u \in N_G$, there is a corresponding input tree T_u that represents the set of all paths of G ending in u. Similarly, there exists a corresponding output tree $\hat{T}_u$ that represents the set of all paths of G starting from u. We say that two input trees T_u and T_v are isomorphic ($T_u \sim T_v$) when there is a bijective map $\tau : T_u \to T_v$, which maps one-to-one the nodes and edges of T_u to nodes and edges of T_v. The same definition and notation apply to output trees.

For graph G,

1. An *fibration symmetry* of G is a surjective homomorphism $\Phi_{fib} : G \to B$ that preserves the input tree:

$$\forall u, v \in N_G : \varphi_{fib}(u) = \varphi_{fib}(v) \in N_B \iff T_u \sim T_v. \tag{7}$$

2. An *opfibration symmetry* of G is a surjective homomorphism $\Phi_{opf} : G \to B$ that preserves the output tree:

$$\forall u, v \in N_G : \varphi_{opf}(u) = \varphi_{opf}(v) \in N_B \iff \hat{T}_u \sim \hat{T}_v. \tag{8}$$

3. An *covering symmetry* of G is a surjective homomorphism $\Phi_{cov} : G \to B$ that preserves the input and output trees.

4. An *automorphism* of a graph G is a bijective map $\Phi_{auto} : G \to G$, such that the pair of nodes u and v forms an edge (u, v) if and only if $(\Phi(u), \Phi(v))$ also forms a edge.

Unlike automorphism symmetries, which are global and highly restrictive, fibration symmetry, opfibration symmetry, and covering symmetry are all local symmetries. Examples of these symmetries are shown in Fig. 1.

In particular, we show an automorphism that maps 1 to 2, 2 to 1, 3 to 4, 4 to 3, 5 to 6, 6 to 5, and all other nodes map to themselves (se Fig. 1d). It is denoted by:

$$\pi \;=\; \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 2 & 1 & 4 & 3 & 6 & 5 \end{pmatrix} , \tag{9}$$

in Cauchy's two-line notation which consists of a first row with the original labels of nodes, and a second row with the permuted labels.

7.2.1 Symmetries in Feedforward Networks

In feedforward networks (e.g. Multi-Layer Perceptrons), an automorphism symmetry between two nodes would require that they are in the same layer and that they have identical weight vectors at the input and output, so that permuting the two nodes keeps the entire network unchanged. Fibration and opfibration symmetries are less restrictive than automorphisms. As we shall see in the mathematical definition, this specifies a weaker constraint on the weights (Section 3).

The input tree of a node i in layer l consists of all paths from the input layer to node i. For example, for a node i in the first layer ($l = 1$), its input tree is the vector $T_i = [W_{i1}^{(1)}, W_{i2}^{(1)}, ..., W_{id_0}^{(1)}]$. Then, two nodes have isomorphic input trees if $T_i = T_j$. Using this criterion, we can identify fibers in the first layer. For layer $l = 2$, the input tree of node i can be expressed as the concatenation of the input trees T_m of nodes m in

layer $l = 1$, and their corresponding connection weights $W^{(2)}_{im}$. If the fibers c in the first layer are already known, T_i can be compactly represented as the concatenation of the input trees of the fibers T_c and the aggregated weights $\sum_{k \in c} W_{ic}$. Then, two nodes i and j of layer $l = 2$ have isomorphic input trees if $\sum_{k \in c} W_{ic}$ for all c. This argument can be extended to the next layers $l = 3, ..., N$ (see definition in Section 3). Also, a similar analysis applies to output trees and opfibration.

7.3 Balanced coloring

A *coloring* c of a graph G is a map $c : N_G \to \mathcal{C}$ where $c(u)$ is called color of the node u and $\mathcal{C}$ is called the set of *colors*. The coloring c is *balanced* if $c(u) = c(v)$ implies that T_u and T_v are color isomorphic. The minimal balanced coloring is a balanced coloring of a graph with the minimal number of colors. In terms of synchronization, nodes inside the same subset of the balanced coloring partition can synchronize (i.e. nodes with same color), since they receive the same color inputs from the same synchronized nodes.

7.3.1 Refinement Algorithms for Balanced Colorings in Feedforward Networks

A refinement algorithm of G starts with an initial coloring c_0 and produces a new coloring c_t in each iteration t, based on some criteria, until a desired property is achieved.

In a graph without weights edges and without labeled nodes, the method can be summarized as follows:

1. c_0 assigns a trivial color to each vertex v (e.g. $c_0(v)$=1).
2. $c_{i+1}(v) = (c_i(v), \{\{c_i(w) \mid w \text{ is a neighbor of } v\}\})$

At some point, it stabilizes $c_{t+1}(u) = c_t(u)\ \forall u \in N_G$, $t > T$.
In a layered feedforward network, the method can be expressed as follow:

1. $c^{(l=0)} \in \mathcal{C}^{(0)}$ assigns different colors to the input features ($l = 0$).
2. For $l = 1, ..., N$:
   2.1. Calculation of $\hat{W}^{(l)}_{ir} = \sum_{k|c(k)=r} W^{(l)}_{ik}\ \forall r \in \mathcal{C}^{(l)}$
   2.2. Normalization of $\hat{W}^{(l)}$.

7.4 Lifting property by Fibration symmetry

We want to find the parameters of a base network B_{fib} whose nodes are the fibers of the original network G and that maintains forward computation.

7.4.1 MLP

Let $\hat{h}_{c'}$ be the activity of a fiber $c' \in \mathcal{C}_l^{\text{fib}}$, then it follows that:

$$\begin{aligned}
\hat{h}_{c'}^{(l)} &= h_i^{(l)} \qquad \forall i \in c' \qquad \text{(activity synchronization)} \\
&= \frac{1}{|c'|} \sum_{i \in c'} h_i^{(l)} \\
&= \frac{1}{|c'|} \sum_{i \in c'} \left[W_{ik}^{(l)} h_k^{(l)} + b_i^{(l)} \right] \\
&= \frac{1}{|c'|} \sum_{i \in c'} \sum_{c \in \mathcal{C}_{l-1}^{\text{fib}}} \sum_{k \in c} W_{ik}^{(l)} h_k^{(l)} + \sum_{i \in c'} b_i^{(l)} \\
&= \sum_{c \in \mathcal{C}_{l-1}^{\text{fib}}} \left[\frac{1}{|c'|} \sum_{i \in c'} \sum_{k \in c} W_{ik}^{(l)} \right] \hat{h}_c^{(l)} + \left[\sum_{i \in c'} b_i^{(l)} \right] \\
&= \sum_{c \in \mathcal{C}_{l-1}^{\text{fib}}} \hat{W}_{c'c} \hat{h}_c^{(l)} + \hat{b}_c^{(l)}
\end{aligned} \tag{10}$$

where $\hat{W}_{c'c} = \frac{1}{|c'|} \sum_{i \in c'} \sum_{k \in c} W_{ik}^{(l)}$ and $\hat{b}_{c'}^{(l)} = \sum_{i \in c'} b_i^{(l)}$. These matrices $\hat{W}_{c'c}$ and vectors $\hat{b}_{c'}^{(l)}$ are the parameters of the base B_{fib}.

7.4.2 CNN

A similar line of reasoning works for CNNs where the nodes of the computational graph are the channels in the convolutions. More precisely,

$$\begin{aligned}
\hat{\mathbf{W}}[:, :, c, c'] &= \frac{1}{|c'|} \sum_{i \in c'} \sum_{k \in c} \mathbf{W}^{(l)}[:, :, k, i], \\
\hat{\mathbf{b}}_{c'}^{(l)} &= \sum_{i \in c'} \mathbf{b}_i^{(l)}.
\end{aligned} \tag{11}$$

7.4.3 LSTM

7.4.4 Attention

7.5 Proof of Theorem 4.1 point (1)

Osvaldo/Ali/Matteo:

7.6 Proof of Theorem 4.1 point (2)

Osvaldo/Ali/Matteo:

7.7 Proof of weight update Eq. 5

For a dataset $\mathcal{D} = \{(x, y)\}$, we denote the network's input as $h^{(0)} = x$ or $\mathbf{H}^{(0)} = x$. The error of the prediction is $L = \mathcal{L}(\hat{y}, y)$, where $\mathcal{L}$ is called *loss function*. The weights $W^{(l)}$ and $\mathbf{W}^{(l)}$ can then be adjusted based on corrections that minimize the error L. Using gradient descent, the change in the weight matrix/tensor is

$$W^{(l)}(t) = W^{(l)}(t-1) - \alpha \frac{\partial L}{\partial W^{(l)}}(t-1) \tag{12}$$

where α is called learning rate.
We calculate the derivative

$$\text{MLP}: \frac{\partial L}{\partial W^{(l)}} = \delta^{(l)} \cdot \left(h^{(l-1)}\right)^T,$$
$$\text{CNN}: \frac{\partial L}{\partial \mathbf{W}^{(l)}} = \delta^{(l)} * \mathbf{H}^{(l-1)}$$

where δ^l is the error signal for nodes in layer l.
For MLPs, we obtained:

$$\delta^{(l)} = \left(\prod_{k=l+1}^{N} \left(W^{(k)}\right)^T \text{diag}[\sigma'(z^{(k)})] \right) \frac{\partial L}{\partial h^{(N)}} \odot \sigma'(z^{(N)}), \tag{13}$$

where the matrix $\text{diag}[\sigma'(z^{(k)})]$ is a diagonal matrix, which diagonal is the vector $\sigma'(z^{(k)})$.

For CNNs, $\delta^{(l)} = \frac{\partial L}{\partial \mathbf{H}^{(l)}}$. Note $\delta^{(l-1)} = \delta^{(l)} *^T \mathbf{W}^{(l)}$ with initial condition $\delta^{(N)} = \frac{1}{L} \otimes \frac{1}{L} \otimes \frac{\partial L}{\partial \hat{y}}$.

7.8 Symmetries and synchronized learning

Suppose the linear case without bias (i.e $\sigma = Id$ and $b = 0$). The terms of $\frac{\partial L}{\partial W^{(l)}} = \delta^{(l)} \cdot \left(h^{(l-1)}\right)^T$ are

$$\begin{aligned} \delta^{(l)} &= \underbrace{\left(\prod_{k=l+1}^{N} W^{(k)} \right)^T}_{W_{out}^{(l)}} \delta^{(N)}, \\ h^{(l-1)} &= \underbrace{\left(\prod_{k=1}^{l-1} W^{(k)} \right)}_{W_{in}^{(l-1)}} x. \end{aligned} \tag{14}$$

The row m of the product $W_{out}^{(l)}$ is a summary representation of the output tree of the node m of the layer l. Similarly, the row i of the product $W_{in}^{(l)}$ is a summary representation of the input tree of the node i of the layer $l-1$.

Consider two nodes i and j in layer $l-1$ in the same fiber and two nodes m and n in layer l in the same opfiber (see red and green nodes in Fig. 2). It is clear that the rows i and j of $W_{in}^{(l)}$ will be identical; similarly, the rows m and n of $W_{out}^{(l)}$ will also be identical. Furthermore, $h_i^{(l-1)} = h_j^{(l-1)}$ and $\delta_m^{(l)} = \delta_n^{(l)}$. Then, we obtain

$$\frac{\partial L}{\partial W_{mi}^{(l)}} = \frac{\partial L}{\partial W_{mj}^{(l)}} = \frac{\partial L}{\partial W_{ni}^{(l)}} = \frac{\partial L}{\partial W_{nj}^{(l)}}$$

or equivalently,

$$\Delta W_{mi}^{(l)} = \Delta W_{mj}^{(l)} = \Delta W_{ni}^{(l)} = \Delta W_{nj}^{(l)}. \tag{15}$$

7.9 Proof of Theorem 4.2

Osvaldo/Ali/Matteo:

Proof of synchronized learning.

7.10 Proof of Theorem 4.3

Osvaldo/Ali/Matteo:
REPEAT - REMOVE

Before proving the theorem, let us refine the notation we will be using. If i and j are two nodes in layer l, we define at training epoch t the following symmetries, based on partitions of nodes $C^{fib}_{l-1}(t)$ and $C^{op}_{l+1}(t)$ of its neighboring layers.

1. **Fibration symmetry:**

$$i \underset{fib,l,t}{\sim} j \iff \forall c \in \mathcal{C}^{fib}_{l-1}(t) : \sum_{k\in c} W^{(l)}_{ik}(t) = \sum_{k\in c} W^{(l)}_{jk}(t)$$

2. **Opfibration symmetry:**

$$i \underset{op,l,t}{\sim} j \iff \forall \hat{c} \in \mathcal{C}^{op}_{l+1}(t) : \sum_{k\in \hat{c}} W^{(l+1)}_{ki}(t) = \sum_{k\in \hat{c}} W^{(l+1)}_{kj}(t)$$

3. **Covering symmetry:**

$$i \underset{cov,l,t}{\sim} j \iff i \underset{fib,l,t}{\sim} j \quad \& \quad i \underset{op,l,t}{\sim} j$$

Lemma 7.1 *During gradient descent training of an FNN, learning synchronization ensures that:*

1. *Nodes in the same covering fiber $i \underset{cov,l-1,t}{\sim} j$ will belong to the same opfiber at the next epoch $i \underset{op,l-1,t+1}{\sim} j$ if $\mathcal{C}^{op}_l(t+1)$ is coarser than $\mathcal{C}^{op}_l(t)$.*
2. *Nodes in the same covering fiber $m \underset{cov,l,t}{\sim} n$ will belong to the same fiber at the next epoch $m \underset{fib,l,t+1}{\sim} n$ if $\mathcal{C}^{fib}_{l-1}(t+1)$ is coarser than $\mathcal{C}^{fib}_{l-1}(t)$.*

Proof (1) Let $i \underset{cov,l-1,t}{\sim} j$ be nodes in layer $l-1$. For epoch $t+1$, for $\forall \hat{c} \in \mathcal{C}^{op}_l(t+1)$:

$$\begin{aligned} \sum_{k\in\hat{c}} W^{(l)}_{ki}(t+1) &= \sum_{k\in\hat{c}} W^{(l)}_{ki}(t) + \Delta W^{(l)}_{ki}(t) \\ &= \sum_{k\in\hat{c}} W^{(l)}_{ki}(t) + \Delta W^{(l)}_{kj}(t). \end{aligned} \tag{16}$$

The last equality is valid due to $i \underset{fib,l-1,t}{\sim} j$ and Lemma 4.1 (Learning synchronization). Now, note that

$$\begin{aligned} \sum_{k\in\hat{c}} W^{(l)}_{ki}(t) &= \sum_{r\in\mathcal{C}^{op}_l(t)} \sum_{k\in\hat{c}\cap r} W^{(l)}_{ki}(t) \\ &= \sum_{r\in\mathcal{C}^{op}_l(t)} \Theta(r\subset\hat{c}) \sum_{k\in r} W^{(l)}_{ki}(t) \qquad \textit{Hyp) } \mathcal{C}^{op}_l(t+1) \textit{ is coarser than } \mathcal{C}^{op}_l(t) \\ &= \sum_{r\in\mathcal{C}^{op}_l(t)} \Theta(r\subset\hat{c}) \sum_{k\in r} W^{(l)}_{kj}(t) \qquad \textit{Hyp) } i \underset{op,l-1,t}{\sim} j \\ &= \sum_{k\in\hat{c}} W^{(l)}_{kj}(t). \end{aligned} \tag{17}$$

Then, we obtain $\sum_{k\in\hat{c}} W_{ki}^{(l)}(t+1) = \sum_{k\in\hat{c}} W_{kj}^{(l)}(t+1)$. That means $i \underset{op,l-1,t+1}{\sim} j$.

The proof of (2) is similar.

Theorem 7.2 *During gradient descent training of an FNN, if* $\mathcal{C}_{l-1}^{fib}(t+1)$ *is coarser than* $\mathcal{C}_{l-1}^{fib}(t)$ *and* $\mathcal{C}_{l+1}^{op}(t+1)$ *is coarser than* $\mathcal{C}_{l+1}^{op}(t)$*, then* $\mathcal{C}_{l}^{cov}(t+1)$ *is coarser than* $\mathcal{C}_{l}^{cov}(t)$*.*

Proof. If $i \underset{cov,l,t}{\sim} j$, we obtain that $i \underset{op,l,t+1}{\sim} j$ using Lemma 7.1 (1) with the hypothesis that $\mathcal{C}_{l+1}^{op}(t+1)$ is coarser than $\mathcal{C}_{l+1}^{op}(t)$. Also, we obtain that $i \underset{fib,l,t+1}{\sim} j$ using Lemma 7.1 (2) with the hypothesis that $\mathcal{C}_{l-1}^{fib}(t+1)$ is coarser than $\mathcal{C}_{l-1}^{fib}(t)$.

Finally, $i \underset{cov,l,t+1}{\sim} j$. Then, $\mathcal{C}_{l}^{cov}(t+1)$ is coarser than $\mathcal{C}_{l}^{cov}(t)$.

7.10.1 Proof of Theorem 4.3

Theorem 3.1 indicates $\forall l$*:* $\mathcal{C}_{l}^{cov}(t+1)$ *is coarser than* $\mathcal{C}_{l}^{cov}(t)$*.*

Proof For $l = 0$ and $l = N$, $\mathcal{C}_0(t) = \{[1], [2], ..., [d_0]\}$, and $\mathcal{C}_N(t) = \{[1], [2], ..., [d_N]\}$ that remain constant over time t. The statement holds for both layers.

7.11 Results on major architecture

7.12 Generalization to other computational graphs

Here we generalize the concepts above to any computational graph involving weighted sums and point nonlinearities, including hypergraphs, where two or more nodes interact before acting on the subsequent node. For ANNs, this is most commonly a multiplicative interaction implementing a modulation or "gating" of activity. We develop this on concrete architectures that are currently dominant in deep-networks, starting with convolutional neural networks (CNN).

CNN. The definition of fibers above for MLPs can be generalized to CNNs (Fig. 7b). To do so, Equation (1) is replaced by a convolutional network, where the product and sum in each layer are now a convolution and average pooling represented as $*$: $\mathbf{H}^{(l)} = \sigma\left(\mathbf{H}^{(l-1)} * \mathbf{W}^{(l)} + \mathbf{b}^{(l)}\right)$, with $\mathbf{W}^{(l)} \in \mathbb{R}^{k\times k\times d_{l-1}\times d_l}$, $\mathbf{b}^{(l)} \in \mathbb{R}^{d_l}$, and $\mathbf{H}^{(l)} \in \mathbb{R}^{L\times L\times d_l}$, where l represents layers, $k \times k$ are the size of the 2D convolution kernels, and d_l are the number of channels in each layer. Now, Equation 7 is applied to channels i, j in a layer l; where $W_{iu}^{(l)}$ is a k^2-dimensional vector that represents connections between channels i and u.

Computational Hypergraph. The structure of MLP and CNN can be captured by a simple *graph*, which connects one node to the next with an edge (Fig. 7a b). More generally, modern network architectures such as LSTM and Transformers can be described as *hypergraph* (Fig. 7c d). In a hypergraph, two or more nodes interact before providing input to the next node. This interaction – often a product or a sum – can be represented as a "factor" (the squares in (Fig. 7c d).

LSTM and RNN The LSTM is a particular form of a recurrent neural network (RNN). Although LSTMs have been replaced by transformers, recurrence remains important because it can implement multiple processing stages in a more compact network, and because recurrence is the dominant architecture in biological neural network. The LSTM is shown as a hypergraph in Fig. 7. As with most RNN, it contains "gates" that multiply with the state of a node. For example, Cell State C_t takes as input the Cell gate g_t and input i_t combining them with $\odot$ — an element-wise multiplication: $C_t = f_t \odot C_{t-1} + i_t \odot g_t$. Let us consider the second term in this sum. It represents a multiplicative two-node interaction. For a fiber to emerge in this product,

two nodes must be a fiber in the corresponding nodes of i_t **and** in a fiber of nodes of g_t, as exemplified with colors in Fig. 7. Two nodes in the product have the same colors, if they had the same colors in both factors. The same is true for the element-wise product of f_t and C_{t-1}. — this color combination rule is the same *And operation* as we used for covers, but here it applies to the factors of the product. The sum can be treated as a concatenation of nodes with identity as weight matrix and the same isomorphism condition as in Eq. (**??**). This definition of isomorphic nodes (fibers) constitutes a general algorithm to identify fibers in any feedforward hypergraph composed of sums and products. To our knowledge, both the sum and product rules for fibers in hypergraphs are new to the literature.

In a recurrent network, as in the LSTM, the same fiber (color) propagation rules can be applied until convergence, i.e. the colors no longer change. Convergence is guaranteed after a number of steps of no more than the longest loop in the recurrence [**?**].

We have tested a typical use of LSTMs in agent training with reinforcement learning (RL). The agent is a deep neural network that maps states to actions. We have investigated fiber formation in reinforcement learning using the proximal policy optimization (PPO) algorithm in the beam-rider game [**?**]. The architecture consists of three Conv2d layers followed by a linear layer and LSTM with a linear policy and value head. After training, we identify fibers, collapse the network, and evaluate performance. The results in Fig. 4d show that the fibration collapse of the linear layers provides a reduction of 70% without loss of performance, as given by the score of the game. This study will be extended to a fibration analysis of the LSTM module, convolutional and linear layers in different games in RL training.

The study of symmetries in RL represents a new area of research. Since the environment works as the data source for the RL agent, the emergence of symmetries is inherently dependent on it. We will study how the environment and multi-agent interactions affect the formation of symmetries. On the other hand, LSTM represents one specific case of a recurrent architecture. We will extend our study to other existing recurrent models with different gating mechanisms, memory structures, number of feedback loops, and temporal delays.

Attention modules and Transformers. The main architectural innovation of Transformers is the "attention mechanism". It can be described as a sequence of weighted sums and multiplications similar to the gating mechanism in LSTMs. The attention mechanism consists of a query, key, and value matrix (with dimensions of token by features): $\mathbf{Q} = \mathbf{X}\mathbf{W}^Q, \mathbf{K} = \mathbf{X}\mathbf{W}^K, \mathbf{V} = \mathbf{X}\mathbf{W}^V$. Here, the $\mathbf{W}^Q, \mathbf{W}^K, \mathbf{W}^V$ weights are trainable parameters and define the symmetries in the network. The query and key matrices are multiplied to obtain an "attention" score: $\mathbf{A} = \frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}$ (see Fig. 7), which quantifies how much each token "attends" to another token. The scores are then used as weights to obtain a weighted average (over tokens) of the value vectors, $\mathbf{z} = \text{softmax}(\mathbf{A})\mathbf{V}$. In total, all operations are a product or a weighted sum of nodes, which can be treated with the same two rules for weighted sums and element-wise products described above to identify fibers. The attention mechanism is typically followed by dense layers that operate over channels for each time point independently and can be treated the same as in MLPs or CNN.

Finally, Transformers and some CNNs have "residual connections", which simply add the input to the output of multiple layers. These "skip connections" combine entirely different colors (from different layers), in which case the addition operation simplifies to the same *And operation* as for the element-wise multiplication. If the input does not have large fibers, this *And operation* will tend to break up the fibers in the later layers of the network.

The attention mechanism has both a feature dimension and a token dimension. Here we have implemented a coloring algorithm to find fibration symmetries for the Q, K,

and V matrices in the feature dimension, which can be propagated across all layers as before, similar to CNNs. The computation of symmetries for attention scores α requires defining symmetries across the token dimension while also incorporating the quadratic operation inherent in the feature dot product and will be part future work.

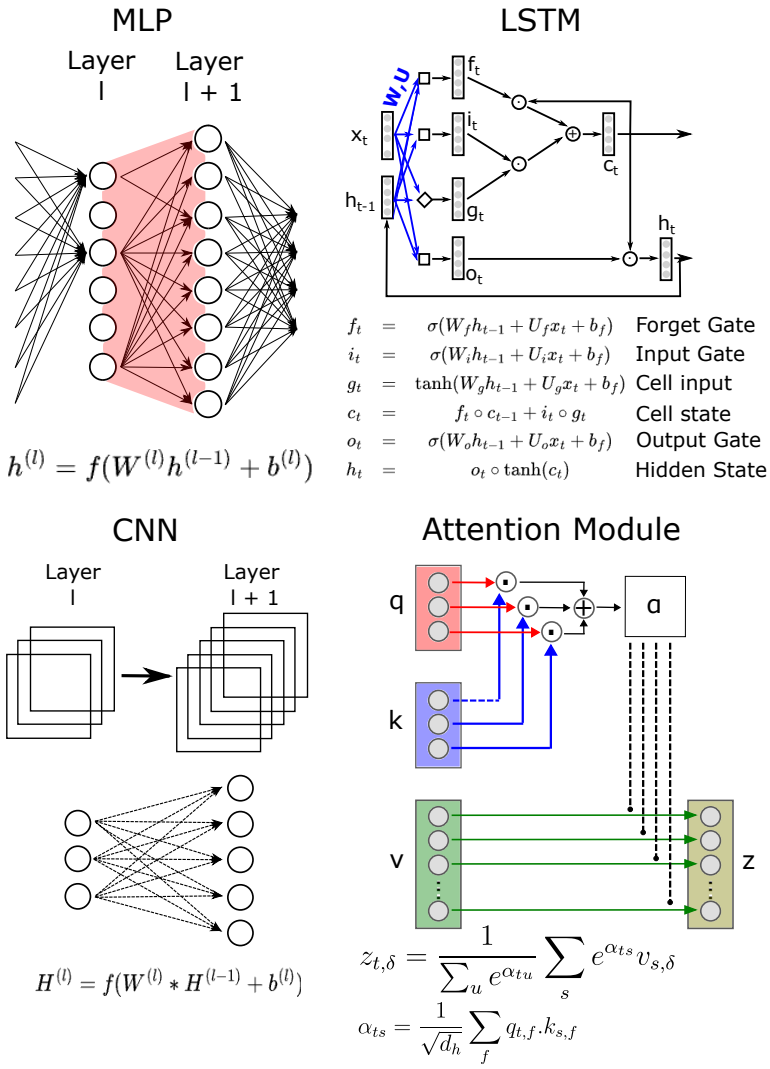

Figure 7. Schematic representation of deep learning architectures as computational graphs and hypergraphs. The figure illustrates the core topologies of: (a) MLP (Multilayer Perceptron), (b) CNN (Convolutional Neural Network), (c) LSTM (Long Short-Term Memory), and (d) Attention Module, along with their characteristic equations. In MLP and CNN, f is a non-linear activation. For LSTM, activations are represented with square (sigmoid) and diamonds (hyperbolic tangent) representing the factors in the factor graph representation of the hypergraph.In all the cases, W, U and b are trainable parameters. The symbol $\circ$ and $+$ represents the operations product and sum element-wise, resp. In Attention Module, α is called attention score calculated based on **q**uery and **k**ey. The output z is proportional to **v**alue.

7.13 Results on CL

Unlike humans, ANNs struggle with lifelong learning due to the progressive inability to acquire new associations without erasing previously acquired ones. In sequential training regimes, this manifests as plasticity loss that can ultimately stall learning altogether [30, 40]. Common heuristics, such as full or partial reinitialization and transfer-learning protocols that reset only the final layers, offer at best narrow, task-specific relief; they do not constitute a general mechanism for preserving plasticity across diverse tasks. As a result, plasticity loss remains a central obstacle on the path toward artificial general intelligence [41].

this can be removed from here or reduced and integrated

In contrast to humans, ANNs struggle to learn new tasks. Lifelong learning remains a challenge due to catastrophic forgetting. Recent work shows that deep networks in particular, after training on multiple tasks sequentially, will lose the ability to learn new tasks altogether. The reason for this loss of plasticity is not well understood and a clear solution has not yet been identified [30]. The brute-force approach to learning new tasks involves reinitializing the network with small, random weights throughout its structure, resulting in obvious catastrophic forgetting. A more nuanced heuristic for deep networks, known as "transfer learning," involves randomizing only the last layers of the network while preserving the "backbone." This approach works well when the input to the deep network remains the same and only the labels at the output change. However, a general solution to the loss of plasticity is still lacking. Plasticity loss together with catastrophic forgetting remains a key bottleneck to achieve artificial general intelligence [41].

Since the covering symmetry is an attractor of the learning dynamics, once the network settles into this symmetry, it remains there forever, explaining the network's inability to learn new associations. Drawing inspiration from the emergence of states of matter in physics through symmetry breaking [31], we demonstrate that ANNs can continue to learn new tasks by breaking the covering symmetries. This process of symmetry breaking is also inspired by how the human brain processes a task like language: the language state represents a broken fibration symmetry of the resting state [11].

We demonstrate that our symmetry-breaking method unifies previous methods of designed to avoid node collapse, including dropout [28], redundancy reduction [42, 43], and random perturbation [29], outperforming them in learning new tasks.

CL:Since symmetries imply redundancies, the symmetric wiring can be broken in an optimal way to keep old memories intact while letting the network learn new tasks as new examples are presented.

When learning a single task (e.g., binary classification between Class 1 and Class 2), identifying these symmetries enables effective network compression without loss of functionality. However, these symmetry patterns are not necessarily preserved when the network faces new tasks. Changes in these symmetry structures can significantly impact both the network's functional capacity and its ability to acquire new information (i.e. *plasticity*). While it is well-known that DNNs often exhibit plasticity loss even in simple learning scenarios [Understanding Plasticity in Neural Networks], the underlying mechanisms remain poorly understood.

We begin by investigating the Continual ImageNet introduced in [XX]. Our benchmark comprises a sequence of 5,000 binary classification tasks, each formed by pairing distinct classes from ImageNet. For every task, a deep neural network is trained on images from the two selected classes and evaluated on a corresponding held-out test set. The experiments use ImageNet, a standard object-recognition dataset with 1,000 classes, each represented by approximately 700 images. Each class is split into 600 training images and 100 test images. Tasks are presented sequentially, each new task sampling a new pair of classes. Training runs for 250 epochs per task, and performance

is assessed through test set accuracy. Since task difficulty remains constant throughout the sequence, any decrease in accuracy signals a loss of plasticity.

We use CNN with three convolutional layers followed by three fully connected layers. The final layer consists of two nodes, corresponding to the two classes of the current task. Following standard practices [52], this output layer is reinitialized at the start of each new task to accommodate the new class pair. We evaluate three optimization approaches: standard gradient descent, deep continual learning, and symmetry-breaking gradient descent. All three methods minimize cross-entropy loss, with detailed hyperparameters provided in tables. To ensure robust comparisons, each method undergoes 10 independent runs using the same predefined sequence of class pairs. The network weights are initialized randomly once before the first task.

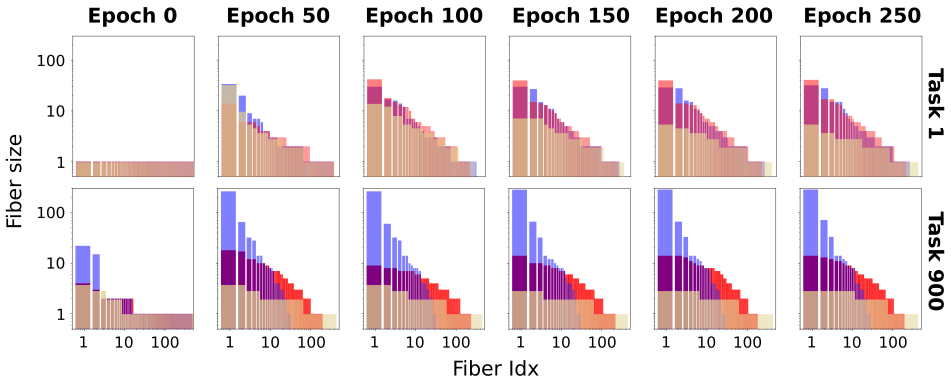

Figure 8. Distribution of covering fibers. Evolucion temporal de los tamanos de las covering fibers para diferente epochs (column's) y diferente tasks (filas). (blue) Training with BP with SGD, (red) Training with Deep Continual Learning, (yellow) Training with Symmetry-Breaking GD.

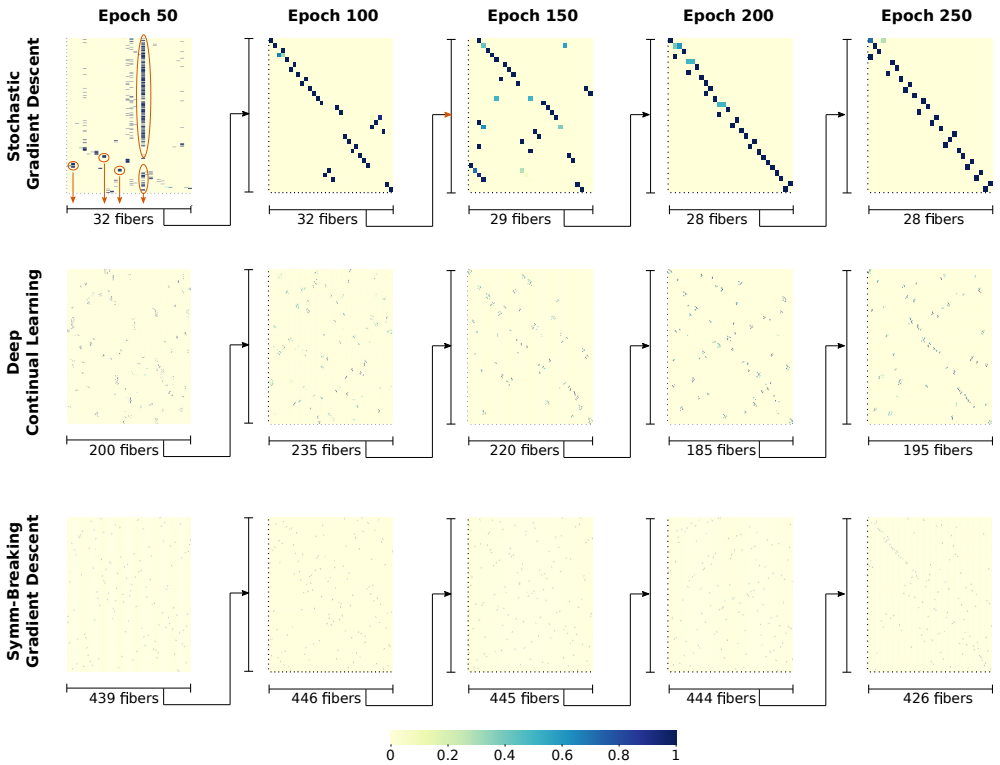

Figure 9. Transition of covering fibers. Para la tarea 900, ...

DCL proposes resetting connections for zero-utility nodes (see Definition 5.3) during training. For these nodes, input weights are reinitialized randomly and output weights are fixed at zero. This modifies the node's activity h while avoiding immediate disruption of the learned network behavior. Altering both input and output weights necessarily modifies the input and output trees of the nodes, thus breaking their associated covering fibers. Notably, zero-utility nodes typically exhibit null activity ($h = 0$) and belong to the same fiber. Thus, DCL inherently targets the breaking of inactive covering fibers.

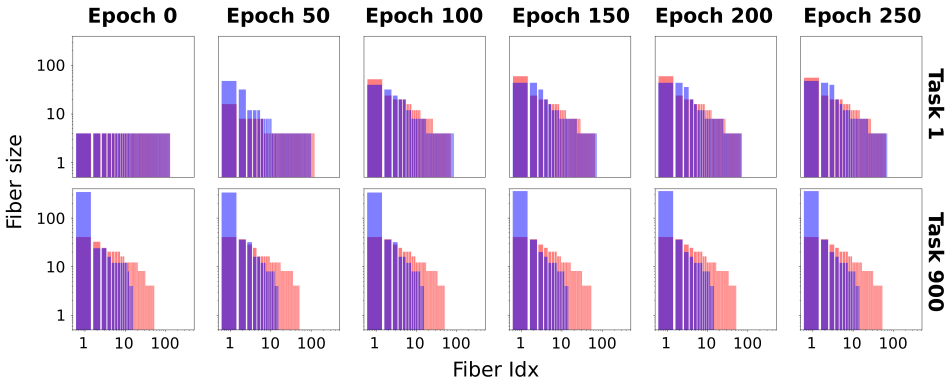

Figure 10. Layer 2 - 512 .

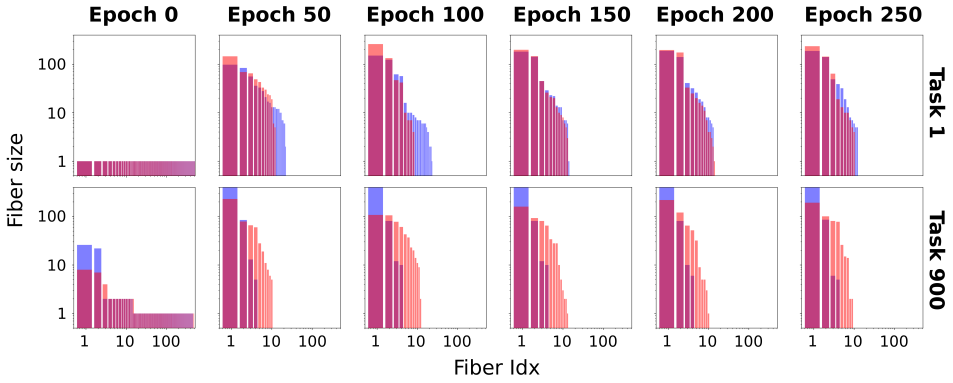

Figure 11. Layer 2 - 512 .

7.14 Deep Continual Learning is a particular case of symmetry breaking

We also show empirically that SGC with symmetry breaking outperforms conventional stochastic gradient descent (SGD) and Deep Continual Learning in classic datasets (ImageNet and CIFAR-100) and standard feedforward networks such as Linear, Convolution, and Residual Layers.

The contribution utility [3] is

$$u_i^{(l)}(t+1) = \eta . u_i^{(l)}(t) + (1-\eta) F_i^{(l)}(t),$$
$$F_i^{(l)}(t) = h_i^{(l)}(t) . \sum_{k=1}^{d_{l+1}} W_{ki}^{(l+1)}(t)$$

with $u_i^{(l)}(0) = 0$. Then,

$$u_i^{(l)}(t) = (1-\eta) \sum_{s=0}^{t-1} \eta^s F_i^{(l)}(t-1-s). \tag{18}$$

Consider two nodes $i \underset{cov,l,t}{\sim} j$ with identical utilities $u_i^{(l)}(t) = u_j^{(l)}(t)$. Then, we obtain activity synchronization $h_i^{(l)}(t) = h_j^{(l)}(t)$; and opfibration symmetry in DNNs $\sum_{k=1}^{d_{l+1}} W_{ki}^{(l+1)}(t) = \sum_{k=1}^{d_{l+1}} W_{kj}^{(l+1)}(t)$. Therefore, $F_i^{(l)}(t) = F_j^{(l)}(t)$ and $u_i^{(l)}(t+1) = u_j^{(l)}(t+1)$. Using Corollary 3.1.1, we obtain $u_i(s) = u_j(s) \quad \forall t \leq s$. That means, *covering symmetry implies utility synchronization*. Deep Continual Learning resets nodes belonging to the same covering fiber when their utility reaches zero.

The permutation invariant set introduced in [25] results from a particular case of the permutation symmetry discussed. Consider two hidden neurons p, q within the same layer l; then the subspace $A = \{\theta | W_{pi}^{(l)} = W_{qi}^{(l)}, W_{mp}^{(l+1)} = W_{mq}^{(l+1)} \quad \forall k, m\}$ is an invariant set.

7.15 EWC

$\mathcal{L}(\theta) = \mathcal{L}_B(\theta) + \frac{\lambda}{2} \sum_{W_{mi}^{(l)}} F(W_{mi,A}^{(l)*})(W_{mi}^{(l)} - W_{mi,A}^{(l)*})^2$

7.15.1 Expression of F

$$F(W^{(l)*}_{mi,A}) = \mathbb{E}_{x,y}\left[\left(\frac{\partial \log p(\hat{y}=y|x,\theta)}{\partial W^{(l)}_{mi}}\right)\right]_{\theta^*_A}$$

$$\begin{aligned} F^{(l)}_{mi} = F(W^{(l)*}_{mi,A}) &= \mathbb{E}_{x,y}\left[(\hat{y}-\vec{e}_y)^T\left(\prod_{k=l+1}^{L} W^{(l)*}\right)h^{(l-1)}(x)\right]^2_{ij} \\ &= \mathbb{E}_{x,y}[\frac{\partial MSE(y,\hat{y})}{\partial W^{(l)}_{mi}}]^2 \\ &= \mathbb{E}_{x,y}\left[r^{(l)}_m(n)h^{(l-1)}_i(x)\right]^2 \end{aligned}$$

If $i \sim_{fib} j, m \sim_{opf} n$, then $F^{(l)}_{mi} = F^{(l)}_{mj} = F^{(l)}_{ni} = F^{(l)}_{nj}$.

7.15.2 SGD

$$\Delta W^{(l)}_{mi} = -\alpha\delta^{(l)}_m h^{(l-1)}_i - \alpha\lambda F^{(l)}_{mi}(W^{(l)}_{mi} - W^{(l)*}_{mi,A})$$

Lemma 7.3 *During gradient descent training of an FNN, learning synchronization ensures that:*

1. *Nodes in the same covering fiber $i \underset{cov,l-1,t}{\sim} j$ will belong to the same opfiber at the next epoch $i \underset{op,l-1,t+1}{\sim} j$ if $\mathcal{C}^{op}_l(t+1)$ is coarser than $\mathcal{C}^{op}_l(t)$.*

Proof (1) Let $i \underset{cov,l-1,t}{\sim} j$ be nodes in layer $l-1$. For epoch $t+1$, for $\forall \hat{c} \in \mathcal{C}^{op}_l(t+1)$:

$$\begin{aligned} \sum_{k\in\hat{c}} W^{(l)}_{ki}(t+1) &= \sum_{k\in\hat{c}} W^{(l)}_{ki}(t)\left[1-\alpha\lambda F^{(l)}_{ki}\right] - \alpha\delta^{(l)}_k h^{(l-1)}_i + \alpha\lambda F^{(l)}_{ki} W^{(l)*}_{ki,A} \\ &= \sum_{k\in\hat{c}} W^{(l)}_{ki}(t)\left[1-\alpha\lambda F^{(l)}_{ki}\right] - \alpha\delta^{(l)}_k h^{(l-1)}_j + \alpha\lambda F^{(l)}_{ki} W^{(l)*}_{ki,A}. \end{aligned} \tag{19}$$

The last equality is valid due to $i \underset{fib,l-1,t}{\sim} j$. Now, note that

$$\begin{aligned} \sum_{k\in\hat{c}} W^{(l)}_{ki}(t)\left[1-\alpha\lambda F^{(l)}_{ki}\right] &= \sum_{r\in\mathcal{C}^{op}_l(t)}\sum_{k\in\hat{c}\cap r} W^{(l)}_{ki}(t)\left[1-\alpha\lambda F^{(l)}_{ki}\right] \\ &= \sum_{r\in\mathcal{C}^{op}_l(t)} \Theta(r\subset\hat{c})\sum_{k\in r} W^{(l)}_{ki}(t)\left[1-\alpha\lambda F^{(l)}_{k,i}\right] \qquad \textit{Hyp) Coarsing} \\ &= \sum_{r\in\mathcal{C}^{op}_l(t)} \Theta(r\subset\hat{c})\left[1-\alpha\lambda F^{(l)}_{:,i}\right]\sum_{k\in r} W^{(l)}_{ki}(t) \qquad \forall k\in r: F^{(l)}_{:,i} = F^{(l)}_{k,i} \\ &= \sum_{r\in\mathcal{C}^{op}_l(t)} \Theta(r\subset\hat{c})\left[1-\alpha\lambda F^{(l)}_{:,j}\right]\sum_{k\in r} W^{(l)}_{kj}(t) \qquad \textit{Hyp) } i \underset{op,l-1,t}{\sim} j \\ &= \sum_{k\in\hat{c}} W^{(l)}_{kj}(t)\left[1-\alpha\lambda F^{(l)}_{kj}\right]. \end{aligned} \tag{20}$$

$F^{(l)}_{ki} W^{(l)*}_{ki,A}$

$$\begin{aligned}\sum_{k\in\hat{c}} F^{(l)}_{ki} W^{(l)*}_{ki,A} &= F^{(l)}_{:,j} \sum_{k\in\hat{c}} W^{(l)*}_{ki,A} \\ &= F^{(l)}_{:,j} \sum_{k\in\hat{c}} W^{(l)*}_{kj,A} \qquad \textit{Hyp) Coarsing} \\ &= \sum_{k\in\hat{c}} F^{(l)}_{kj} W^{(l)*}_{kj,A}\end{aligned} \tag{21}$$

Then, we obtain $\sum_{k\in\hat{c}} W^{(l)}_{ki}(t+1) = \sum_{k\in\hat{c}} W^{(l)}_{kj}(t+1)$. That means $i \underset{op,l-1,t+1}{\sim} j$.

7.16 Parameters

Continual backpropagation has different hyperparameters: step size (0.01), replacement rate (3e-4), maturity threshold (100), and decay rate (0.99).

The values of hyperparameters used for the grid search are described in Extended Data Table 2.

models Network = 3 capas convolucionales + 3 capas lineales. Dataset = Imagenet. Cada tarea es un clasficiacion binaria entre dos clases del dataset. Se aprendieron 1000 tareas.

Minibatch sizes. downsampled 32x32 version of the ImageNet dataset

The momentum hyperparameter was 0.9. We tested various step-size parameters for backpropagation but only presented the performance for step sizes 0.01, 0.001 and 0.0001 for clarity of Fig. 1b.

7.17 Fibration symmetry as a generalization of orthogonal regularization.

Suponga una red lineal $y = Wx \in \mathbb{R}^{d_1}$ con $x \in \mathbb{R}^{d_0}$ y $W \in \mathbb{R}^{d_1 \times d_0}$. Por simplicidad, supongamos que cada fila de W esta normalizada. Para detectar las fibers en la capa de salida, debemos aplicar un algoritmo de clustering sobre la matriz de distancia $D = 1 - W^T W$.

7.18 Redundant feature pruning for accelerated inference in deep neural networks

$$\mathbf{W}^{(l)} \mapsto \begin{bmatrix} \text{vect}(W[:,:,1,1]) & ... & \text{vect}(W[:,:,1,d_L]) \\ \text{vect}(W[:,:,2,1]) & ... & \text{vect}(W[:,:,2,d_L]) \\ ... & ... & ... \\ \text{vect}(W[:,:,d_{l-1},1]) & ... & \text{vect}(W[:,:,d_{L-1},d_L]) \end{bmatrix} = \begin{bmatrix} w_1 & ... & w_{d_L} \end{bmatrix}$$

The paper uses this similarity metric for agglomerative clustering

$$\text{Sim}(C_a, C_b) = \frac{1}{|C_a||C_b|} \sum_{i\in C_a, j\in C_b} w_i \cdot w_j \tag{22}$$

In the strictest case (sim = 1), i, j are in the same cluster if $w_i = w_j$. That means, $\forall n = 1, ..., d_{l-1} : W_{in} = W_{jn}$ (permutation symmetry).

7.19 Optimizers

7.19.1 SGD with Momentum

$$\begin{aligned} v_t &= \gamma \cdot v_{t-1} + \eta \cdot \nabla J(\theta_t) \\ \theta_{t+1} &= \theta_t - v_t \end{aligned} \tag{23}$$

Equivalently, $\theta_{t+1} = \theta_t - \eta \sum_{i=0}^{t} \gamma^{t-i} \nabla J(\theta_i)$.

7.20 Parameters

References

1. J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

2. J. Hoffmann, S. Borgeaud, A. Mensch, E. Buchatskaya, T. Cai, E. Rutherford, D. de Las Casas, L. A. Hendricks, J. Welbl, A. Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

3. Shibhansh Dohare, J. Fernando Hernandez-Garcia, Qingfeng Lan, Parash Rahman, A. Rupam Mahmood, and Richard S. Sutton. Loss of plasticity in deep continual learning. *Nature*, 632(8026):768–774, August 2024. Publisher: Nature Publishing Group.

4. Terrence J Sejnowski. The unreasonable effectiveness of deep learning in artificial intelligence. *Proceedings of the National Academy of Sciences*, 117(48):30033–30038, 2020.

5. M. Belkin, D. Hsu, S. Ma, and S. Mandal. Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proc. Nat. Acad. Sci. USA*, 116(32):15849–15854, 2019.

6. P. Nakkiran, G. Kaplun, Y. Bansal, T. Yang, B. Barak, and I. Sutskever. Deep double descent: Where bigger models and more data hurt. In *International Conference on Learning Representations (ICLR)*, 2020. arXiv:1912.02292.

7. Michael M Bronstein, Joan Bruna, Taco Cohen, and Petar Veličković. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *arXiv preprint arXiv:2104.13478*, 2021.

8. Frank Wilczek. *A Beautiful Question: Finding Nature's Deep Design*. Penguin, 2016.

9. Steven Weinberg. *The Quantum Theory of Fields*, volume 2. Cambridge U.P., 1995.

10. Hernán A. Makse, Paolo Boldi, Francesco Sorrentino, and Ian Stewart. *Symmetry of Living and Artificial Intelligence Systems: Fibration Symmetry and Synchronization in Networks*. Cambridge University Press, 2026.

11. Tommaso Gili, Bryant Avila, Luca Pasquini, Andrei Holodny, David Phillips, Paolo Boldi, Andrea Gabrielli, Guido Caldarelli, Manuel Zimmer, and Hernán A. Makse. Fibration symmetry-breaking supports functional transitions in a brain network engaged in language. *Nat. Phys., under review, https://arxiv.org/abs/2409.02674*, 2025.

12. Bryant Avila, Pedro Augusto, Alireza Hashemi, David Phillips, Tommaso Gili, Manuel Zimmer, and Hernán A. Makse. 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.

13. Osvaldo Velarde, Lucas Parra, Paolo Boldi, and Hernan Makse. The Role of Fibration Symmetries in Geometric Deep Learning, August 2024. arXiv:2408.15894 [cs].

14. Alexandre Grothendieck. Technique de descente et théorémes d'existence en géométrie algébrique, I. Généralités. Descente par morphismes fidélement plats. *Seminaire Bourbaki*, 190, 1959–1960.

15. Paolo Boldi and Sebastiano Vigna. Fibrations of graphs. *Discrete Mathematics*, 243(1):21–66, January 2002.

16. Flaviano Morone, Ian Leifer, and Hernán A. Makse. Fibration symmetries uncover the building blocks of biological networks. *Proceedings of the National Academy of Sciences*, 117(15):8306–8314, April 2020. Publisher: Proceedings of the National Academy of Sciences.

17. Vardan Papyan, XY Han, and David L Donoho. Prevalence of neural collapse during the terminal phase of deep learning training. *Proceedings of the National Academy of Sciences*, 117(40):24652–24663, 2020.

18. Diego Doimo, Aldo Glielmo, Sebastian Goldt, and Alessandro Laio. Redundant representations help generalization in wide neural networks. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 19659–19672. Curran Associates, Inc., 2022.

19. Feng Chen, Daniel Kunin, Atsushi Yamamura, and Surya Ganguli. Stochastic collapse: How gradient noise attracts sgd dynamics towards simpler subnetworks. *Advances in Neural Information Processing Systems*, 36:35027–35063, 2023.

20. David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

21. Paolo Boldi and Sebastiano Vigna. Fibrations of graphs. *Discrete Math.*, 243(1-3):21–66, 2002.

22. Manuela AD Aguiar, Ana Paula S Dias, and Flora Ferreira. Patterns of synchrony for feed-forward and auto-regulation feed-forward neural networks. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 27(1), 2017.

23. Ian Leifer, David Phillips, Francesco Sorrentino, and Hernán A Makse. Symmetry-driven network reconstruction through pseudobalanced coloring optimization. *J. Stat. Mech.: Theor. Exp.*, 2022(7):073403, 2022.

24. P. Erdős and A. Rényi. Asymmetric graphs. *Acta Mathematica Academiae Scientiarum Hungaricae*, 14:295–315, 1963.

25. Feng Chen, Daniel Kunin, Atsushi Yamamura, and Surya Ganguli. Stochastic Collapse: How Gradient Noise Attracts SGD Dynamics Towards Simpler Subnetworks, May 2024. arXiv:2306.04251 [cs].

26. Lee DeVille and Eugene Lerman. Modular dynamical systems on networks. *J. Eur. Math. Soc.*, 17(12):2977–3013, 2015.

27. E. Nijholt, R. Rink, and J. Sanders. Graph fibrations and symmetries of network dynamics. *J. Diff. Eq.*, 261(9):4861–4896, 2016.

28. Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014.

29. Alexander Chebykin, Arkadiy Dushatskiy, Tanja Alderliesten, and Peter A. N. Bosman. Shrink-Perturb Improves Architecture Mixing during Population Based Training for Neural Architecture Search, July 2023. arXiv:2307.15621 [cs].

30. Clare Lyle, Zeyu Zheng, Khimya Khetarpal, Hado van Hasselt, Razvan Pascanu, James Martens, and Will Dabney. Disentangling the causes of plasticity loss in neural networks. *arXiv preprint arXiv:2402.18762*, 2024.

31. Philip W Anderson. More is different: Broken symmetry and the nature of the hierarchical structure of science. *Science*, 177(4047):393–396, 1972.

32. T. Gili, B. Avila, L. Pasquini, A. Holodny, D. Phillips, P. Boldi, A. Gabrielli, G. Caldarelli, M. Zimmer, and H. A. Makse. Fibration symmetry-breaking supports functional transitions in a brain network engaged in language. preprint, 2025.

33. S. Dohare, J. F. Hernandez-Garcia, Q. Lan, P. Rahman, A. R. Mahmood, and R. S. Sutton. Loss of plasticity in deep continual learning. *Nature*, 632:768–774, 2024.

34. P. W. Anderson. More Is Different. *Science*, 177(4047):393–396, August 1972. Publisher: American Association for the Advancement of Science.

35. Philip Ball. *The self-made tapestry: pattern formation in nature*. Oxford Univ. Press, Oxford, repr edition, 2004.

36. D'Arcy Wentworth Thompson. *On Growth and Form: The Complete Revised Edition*. Dover Publications, New York, June 1992.

37. Philip Ball. *Patterns in Nature: Why the Natural World Looks the Way It Does*. University of Chicago Press, Chicago, 2016.

38. Rong Li and Bruce Bowerman. Symmetry Breaking in Biology. *Cold Spring Harbor Perspectives in Biology*, 2(3):a003475, March 2010.

39. Antonio García-Bellido. Symmetries throughout organic evolution. *Proceedings of the National Academy of Sciences of the United States of America*, 93(25):14229–14232, December 1996.

40. Clare Lyle, Zeyu Zheng, Evgenii Nikishin, Bernardo Avila Pires, Razvan Pascanu, and Will Dabney. Understanding plasticity in neural networks. In *International Conference on Machine Learning*, pages 23190–23211. PMLR, 2023.

41. James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

42. Zi Wang, Chengcheng Li, and Xiangyang Wang. Convolutional neural network pruning with structural redundancy reduction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14913–14922, 2021.

43. Jiaxiong Qiu, Cai Chen, Shuaicheng Liu, Heng-Yu Zhang, and Bing Zeng. Slimconv: Reducing channel redundancy in convolutional neural networks by features recombining. *IEEE Transactions on Image Processing*, 30:6434–6445, 2021.

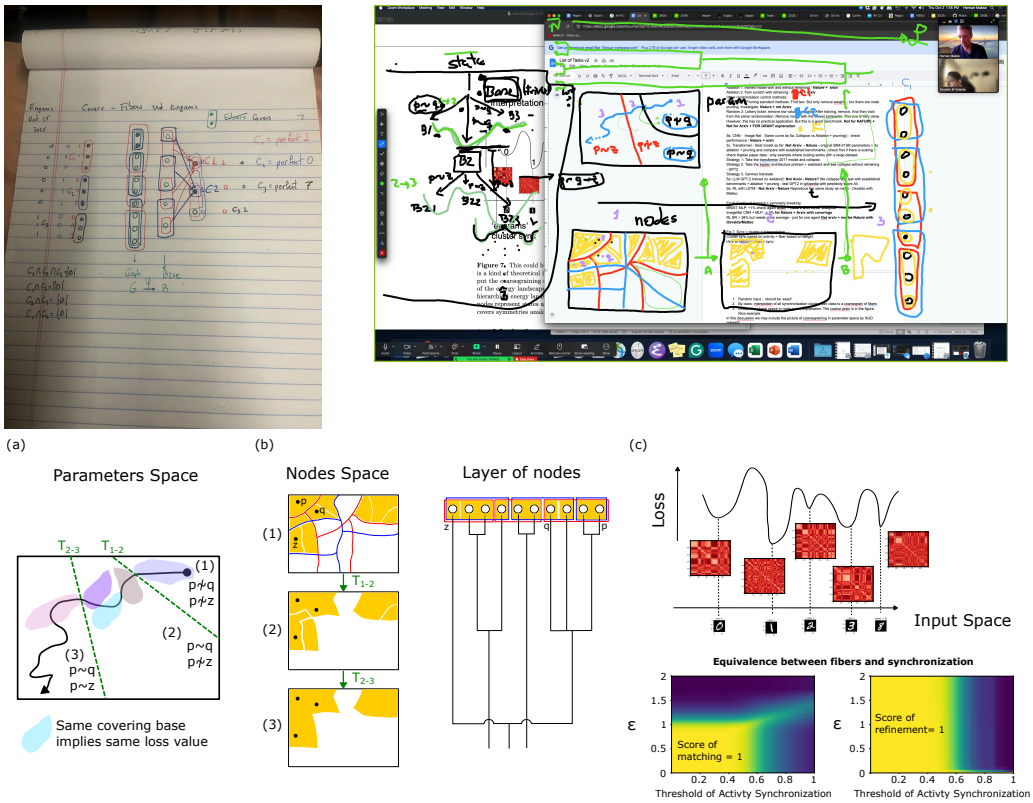

Figure 6. Theoretical interpretation of loss-energy landscape. The relation between synchrony clusters per class and fibers. The results provide an interpretation of the energy landscape in terms of an attractor of the SGD dynamics, giving rise to a hierarchical energy landscape that can be conceptually modeled as a tree structure where nodes represent states and internal nodes represent barriers, capturing the coarsening of covers symmetries analogous to spin glasses. (b) (c) Top: Correlation coefficients of activity in Layer 2 with imputs drawn from individual classes of the MINIST set after training. Left: measures the score of matching between node partitions obtained with activity synchronization versus clusters obtained with fibration (up to ϵ in Eq. 7, using random input images. Provided that threshold on activity synchronization and fibration equality are small, both method find the same partition. Right: here synchronization was measured with inputs only from the NIST data for individual classes, and partitions then intersected to produce a refinement that is synchronous for inputs from for all classes. This is a weaker conditions than fibration, and so we now measure if the fiber partition is insides of the synchronization partition, using as score of refinement.

FROM HERE ON IS ALL JUNK:

8 Discussion: bullet points

Remove: This occurs because symmetries enforce functional equivalence among certain nodes, thereby restricting the network's capacity to specialize for distinct tasks.

NOTE: ORTHOGONAL WEIGHTS PAPER. TRANSFER LEARNING (PICTURE: NUM FIBERS VS TIME) FUNCTIONALITY OF THE FIBERS.

Results

1. SGD/BP can break classes in fibrations (by different error) or opfibrations (by different activity), but cannot break classes in covering.
2. Continual Learning can also break classes in covering with activity = 0.

Explain the generalization. ** Comparison with our results.
Future works

1. Extension continual learning: Break classes in covering with activity = A.
2. Opposite version of Continual Learning: Break classes in covering with error = E.

Main conclusions:

1. Our fundamental result is that a local symmetry in the network of weights, called fibration and covering, is the 'engram' signature of the learned representation or memory of the task. We demonstrate that deep neural networks, during learning with gradient descent, converge to fibration, opfibration, and covering symmetries in their graph structure. These symmetries are local and less stringent than the global group symmetries of GDL, providing more flexible inductive biases to the learning process.
2. The process of creating covering symmetries is inherently facilitated by a phenomenon we term "synchronized learning". The errors of the backpropagation algorithm naturally group neurons into coverings by synchronizing the updates of weights connecting neurons that have synchronized activity (fibrations) to neurons with synchronized errors (opfibrations). This grouping also reduces the degree-of-freedom of the network, which helps avoid overtraining despite the large number of parameters of deep networks. Translating Hebb's dogma to the artificial brain, our result could be summarized as: 'neurons firing together wire to neurons erring together'.
3. Our formalism explains representation collapse, i.e., redundancy between neurons with redundant information is to be avoided, see Barlow.
4. Our framework explains why transfer learning works: swapping the output layer with randomization and retaining the backbone. The simplest form of randomization is found in the last layer. In a sense, we generalize this transfer learning to the entire network by carefully selecting the coverings in each layer to achieve optimized performance by targeted randomization of coverings.
5. Our framework explains the widespread use of tying weights: how networks become better at generalizing across tasks through task learning by tying weights. CNN is a form of inductive bias by tying the weights.
6. Tying weights is a solution for overfitting, so we provide a solution for overfitting too.

7. CNN is also an attractor, as starting from a dense network, the graph learns a CNN convolution. Our framework explains why tie weights are effective for improved generalization.

8. Our framework includes continual Deep Learning. We provide a novel interpretation of Deep Continual Learning, showing that its effectiveness arises from implicit symmetry-breaking mechanisms that preserve plasticity. Building on these insights, we propose new learning rules that generalize symmetry breaking as a fundamental mechanism for maintaining plasticity in DNNs.

9. Our approach is based on theory, and can guide future inductive biases to the learning protocols in a systematic way based on rigorous mathematical theorem of fibration theory. In contrast, Continual Learning, Transfer Learning, Dropout, and most of the improvement in deep learning so far have been guided by trial and error and brute force. They cannot easily generalized in a systematic way. This is in contrast to the philosophy of physics where theory guide the experiments in a synergetic way.

10. Do the forgetting and say that we also solve this long-standing problem.

11. Explain the mechanisms to generate covering and how synchronized learning works. Covering formation is highly improbable. But starting with the coverings in the last layer, there is a competition with the lack of fibers in the data. We impose regularization on the outputs from the loss function, and this trickles down to all the network, but not in the opposite direction.

12. Our results may explain why deep neural networks are better than shallow neural networks. Even though shallow nets are universal approximators, increasing the size of the hidden layer does not necessarily improve performance, despite their ability to approximate any function well. For instance, shallow nets do not work well, tend to overfit, and cannot generalize effectively. Now, by using the same parameter space but distributing the neurons in a deep network with layers that are not too large, performance is significantly enhanced. This is because this structure clearly promotes the formation of coverings. At the same time, the shallow net does not, since it is highly improbable that a symmetry will be formed by random fluctuations when there are large numbers of random connections to the neurons.

13. Why our theory is important: As we train, we start with a random structure, and learning creates the structure. What is the structure? It is the covering. Learning is the transition from randomness to covering symmetry. Some random event needs to initialize this process. We need deep neural networks, not shallow ones. Also, tying will help to initiate this process. Once the first covering is formed, then nothing can stop the networks from destroying this covering.

14. Our results explain how billions of parameters for 100 datapoints can be learned. This is a massive inductive bias that favors the formation of coverings. It is similar to tying, but in our case, it involves a highly intelligent tying technique designed to quickly cover the old memory and then break it, preserving the old memory. It does explain deep, continual learning obtained by trial and error, ie, by change. But in our approach, we leave nothing to chance or trial and error. Theory dictates the best algorithm for symmetry breaking.

15. Explains that even the inputs have symmetries, like a binary image, which has two fibers. An RGB image has 3 fibers by definition. This is propagated into the

network. What is important for the formation of fibers is the spatial correlation between these colors. A random binary image will also have two fibers, but it will not be possible to learn from it. Thus, the formation of fibers is related to how the two fibers are arranged in the location of the net and not just the existence of two fibers in the input layer.

OLD ABSTRACT

OLD Option 1: The recent boom in artificial intelligence (AI) is largely driven by scaling laws, which fuel the belief that increasing computational resources will enhance machine capabilities. Additionally, current extensive infrastructure enables to train large AI models using all data at once, overcoming the challenge that these models, unlike humans, struggle to learn sequentially and often experience a loss of plasticity. In this context, we demonstrate that symmetries and their broken counterparts provide a more effective inductive bias, helping to avoid the pitfalls associated with scaling. The symmetries that emerge in ANNs are not the global group symmetries typically observed in physics, which have been previously applied in deep convolutional and graph networks. Instead, these are less stringent local symmetries related to the input and output trees in the neural network. These local symmetries can be formalized by fibrations, opfibrations, and coverings, concepts introduced by Grothendieck in the context of category theory. We theoretically demonstrate that covering symmetries are preserved during gradient descent and explain how these symmetries naturally emerge through the "synchronized learning" of weights during training. By systematically breaking these symmetries, the network can overcome the scaling laws previously found in AI models and continue learning sequentially new tasks indefinitely. We provide both theoretical and empirical proof of this algorithm, termed Symmetry-Breaking Back Propagation (SBBP), across various network architectures, including multilayer perceptrons, convolutional, recurrent, as well as transformers for gradient descent and reinforcement learning. Our findings demonstrate that the theory of fibration symmetry unifies continuous learning protocols and network architectures within a common framework, paving the way for the development of optimized scaling architectures capable of lifelong learning.

OLD Option 2: Lifelong learning remains a challenge due to catastrophic forgetting. In contrast to humans, machines struggle to continuously learn new tasks. The brute solution to this problem has been based on the scaling hypothesis: take the biggest dataset, largest state-of-the-model and success. However, scaling has its limits. On the contrary, physical systems are typically reduced by its symmetries, a kind of inductive bias that allow us to understand the physical universe under a unifying framework of simple laws. Here we develop a symmetry approach that rather than using the global symmetry groups of physics, ANN learn by developing softer local symmetries that form groupoids, they are called fibrations, opfibrations and coverings. They form the conservation law of artificial learning.

OLD Option 2: Lifelong learning remains a challenge due to catastrophic forgetting. In contrast to humans, machines struggle to continuously learn new tasks. The brute solution to this problem has been based on the scaling hypothesis: take the biggest dataset, largest state-of-the-model and success. However, scaling has its limits. On the contrary, physical systems are typically reduced by its symmetries, a kind of inductive bias that allow us to understand the physical universe under a unifying framework of simple laws. Here we develop a symmetry approach that rather than using the global symmetry groups of physics, ANN learn by developing softer local symmetries that form groupoids, they are called fibrations, opfibrations and coverings. They form the conservation law of artificial learning.

OLD Option 3: In contrast to humans, artificial neural networks (ANNs) struggle to learn new tasks. Here, we solve this problem by first uncovering that neural networks learn and create memories by creating symmetries within their architecture analogous to engrams in the brain. We then demonstrate how breaking these symmetries allows the networks to learn new tasks indefinitely. The symmetries that emerge in ANNs are

not the global group symmetries typically observed in physics, which have been used previously in deep convolutional and graph networks. Instead, they are less stringent local symmetries related to the input and output trees in the network. These local symmetries can be formalized by fibrations, opfibrations, and coverings, concepts introduced by Grothendieck in the context of category theory. We demonstrate theoretically that covering symmetries are preserved during gradient descent and show how these symmetries emerge through the "synchronized learning" of weights during training. By systematically breaking these symmetries, the network can continue to learn new tasks forever. We provide theoretical and empirical proof of this algorithm, termed Symmetry-Breaking Back Propagation (SBBP), in various network architectures, including multilayer perceptrons, convolutional, recurrent, as well as transformers for gradient descent and reinforcement learning. Our findings demonstrate that the theory of fibration symmetry unifies continuous learning protocols and network architectures within a common framework, paving the way for the creation of optimized architectures that can learn forever.

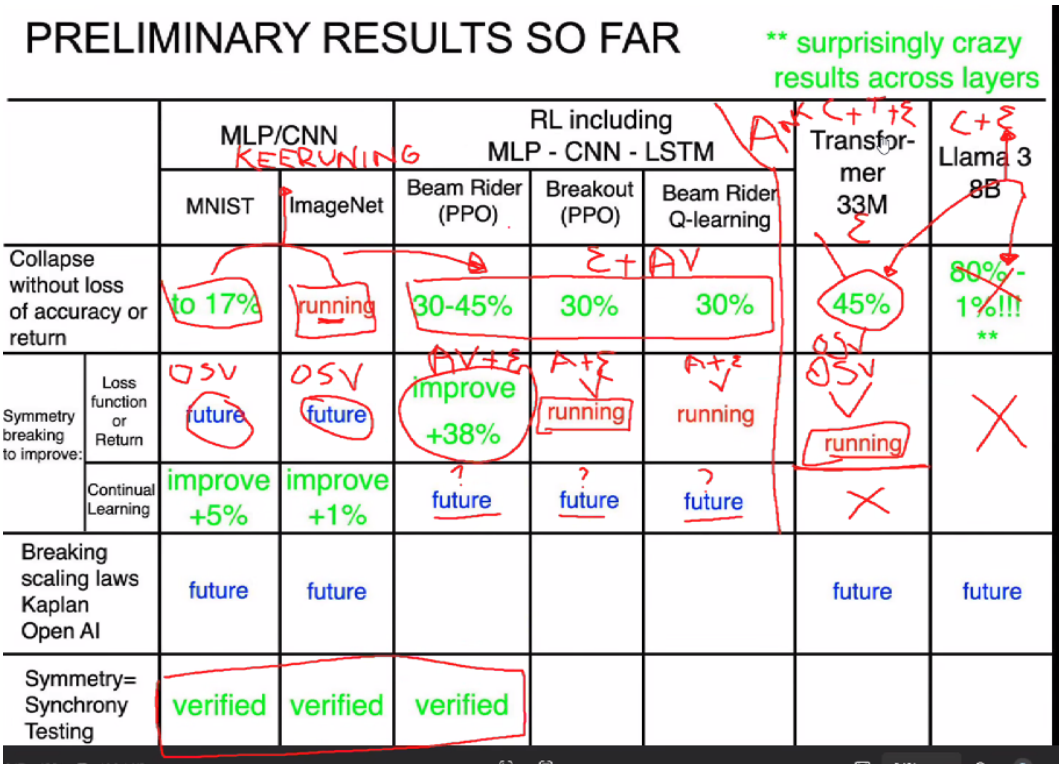

Figure 12. Results

Some conclusions

At the heart of this structure-formation is a phenomenon we call synchronized learning. Through gradient descent and backpropagation, neurons with correlated activations and error gradients self-organize into structured groups—forming coverings that both reduce the degrees of freedom and enhance generalization.

We have shown that local symmetries, i.e., fibrations, opfibrations, and coverings, emerge naturally in the graph topology of neural networks during training. These symmetries act as the engrammatic signatures of learned representations, forming the internal structure that supports generalization, robustness, and memory. Unlike the

rigid global symmetries of group-equivariant deep learning, these local symmetries offer a flexible and biologically inspired inductive bias, better suited to the heterogeneous and layered nature of deep models.

Traditionally, the concept of an engram refers to the physical trace of memory in biological brains, a hypothetical substrate encoding learned experience. In our framework, we redefine this classical notion within the context of artificial intelligence: the engram is not a static pre-defined set of weights, but a dynamic topological symmetry that emerges through training.

You do as you see it, but there is nothing in this paper to support the claim that that memory is stored in covers. We started a conversation on Signal on how one can maybe demonstrate that larger coverings with lots on neurons are more important than trivial covers with a single neuron (I gave very specific suggestions with equations for a possible mathematical argument), but I don't yet see results on this in the paper. So unless you want reviewers to kill you here, at the very minimum you should formulate this as postulate to be tested in future work.

As networks learn via gradient descent, their weight graphs evolve toward structured configurations that exhibit these local symmetries. Fibrations correspond to neurons that receive inputs in a coordinated, symmetric way—suggesting a shared representational function. Opfibrations encode symmetry in the outgoing error structure, revealing shared error propagation roles. Coverings integrate both directions, yielding balanced, bi-directional symmetries that define functionally coherent groups of neurons. These coverings are mathematically precise and biologically plausible, forming the minimal sufficient structure required for generalization without the rigidity of global group symmetries as used in equivariant deep learning.

Thus, the building blocks of the engram is the covering—a localized, structured, symmetry-based representation that preserves the memory of the task within the architecture itself. This reframing transforms our understanding of memory in AI: what is remembered is not merely parameterized values, but structural regularities—the way weights are wired to preserve function under symmetry. These symmetries can be reused, transferred, or selectively broken, allowing for continual learning, adaptation, and robust memory retention.

We have introduced the concept of synchronized learning, a phenomenon emerging naturally from gradient-based optimization. During training, neurons that exhibit similar activation patterns (fibrations) and similar error gradients (opfibrations) tend to be updated in a synchronized fashion. This

Vague and repetitive. suggest cutting: These two concepts—the engram as covering symmetry and the new Hebbian learning principle—together offer a new paradigm for understanding memory and learning in artificial systems. They suggest that intelligence, whether biological or artificial, is deeply linked to the emergence and evolution of symmetry: not just as a mathematical abstraction, but as the operational foundation of memory, generalization, and continual adaptation.

Crucially, our framework is theory-driven, offering systematic guidance for the design of inductive biases, in contrast to the empirical heuristics that have largely dominated the development of techniques such as dropout, transfer learning, and fine-tuning. By grounding learning protocols in rigorous theorems from fibration theory, we provide a path toward principled and generalizable architectures.

In essence, learning is not simply parameter tuning—it is structure formation. Our theory offers a rigorous and interpretable model of this process: not as black-box optimization, but as the emergence of symmetry from randomness, guided by well-defined mathematical principles. In doing so, it opens a path toward designing future learning systems where theory and structure—not brute force scaling dictate performance. The result is not only functionally powerful, but mathematically beautiful.