How Deep Neural Networks Learn Compositional Data: The Random Hierarchy Model

Francesco Cagnetta,[1,*,†] Leonardo Petrini,[1,*] Umberto M. Tomasini,[1]
Alessandro Favero,[1,2] and Matthieu Wyart[1,‡]

[1]*Institute of Physics, EPFL, Lausanne, Switzerland*
[2]*Institute of Electrical Engineering, EPFL, Lausanne, Switzerland*

(Received 19 December 2023; revised 15 April 2024; accepted 31 May 2024; published 1 July 2024)

Deep learning algorithms demonstrate a surprising ability to learn high-dimensional tasks from limited examples. This is commonly attributed to the depth of neural networks, enabling them to build a hierarchy of abstract, low-dimensional data representations. However, how many training examples are required to learn such representations remains unknown. To quantitatively study this question, we introduce the random hierarchy model: a family of synthetic tasks inspired by the hierarchical structure of language and images. The model is a classification task where each class corresponds to a group of high-level features, chosen among several equivalent groups associated with the same class. In turn, each feature corresponds to a group of subfeatures chosen among several equivalent groups and so on, following a hierarchy of composition rules. We find that deep networks learn the task by developing internal representations invariant to exchanging equivalent groups. Moreover, the number of data required corresponds to the point where correlations between low-level features and classes become detectable. Overall, our results indicate how deep networks overcome the curse of dimensionality by building invariant representations and provide an estimate of the number of data required to learn a hierarchical task.

DOI: 10.1103/PhysRevX.14.031001

Subject Areas: Complex Systems,
Interdisciplinary Physics,
Statistical Physics

I. INTRODUCTION

Deep learning methods exhibit superhuman performances in areas ranging from image recognition [1] to Go playing [2]. However, despite these accomplishments, we still lack a fundamental understanding of their working principles. Indeed, Go configurations and images lie in high-dimensional spaces, which are hard to sample due to the *curse of dimensionality*: The distance δ between neighboring data points decreases very slowly with their number P, as $\delta = \mathcal{O}(P^{-1/d})$, where d is the space dimension. Solving a generic task such as regression of a continuous function [3] requires a small δ, implying that P must be *exponential* in the dimension d. Such a number of data is unrealistically large: For example, the benchmark dataset ImageNet [4], whose effective dimension is estimated to be ≈50 [5], consists of only $\approx 10^7$ data, significantly smaller than $e^{50} \approx 10^{20}$. This immense difference implies that learnable tasks are not generic, but highly structured. What is then the nature of this structure, and why are deep learning methods able to exploit it?

A popular idea attributes the efficacy of these methods to their ability to build a useful representation of the data, which becomes increasingly complex across the layers [6]. Interestingly, a similar increase in complexity is also found in the visual cortex of the primate brain [7,8]. In simple terms, neurons closer to the input learn to detect simple features like edges in a picture, whereas those deeper in the network learn to recognize more abstract features, such as faces [9,10]. Intuitively, if these representations are also invariant to aspects of the data unrelated to the task, such as the exact position of an object in a frame for image classification [11], they may effectively reduce the dimensionality of the problem and make it tractable. This view is supported by several empirical studies of the hidden representations of trained networks. In particular, measures such as the mutual information between such representations and the input [12,13], their intrinsic dimensionality [14,15], and their sensitivity toward transformations that do not affect the task (e.g., smooth deformations for image classification [16,17]), all eventually decay with the layer depth. However, none of these studies addresses the *sample*

[*] These authors contributed equally to this work.
[†] Corresponding author: francesco.cagnetta@epfl.ch
[‡] Corresponding author: matthieu.wyart@epfl.ch

complexity, i.e., the number of training data necessary for learning such representations, and thus the task.

In this paper, we study the relationship between sample complexity, depth of the learning method, and structure of the data by focusing on tasks with a hierarchically compositional structure—arguably a key property for the learnability of real data [18–25]. To provide a concrete example, consider a picture that consists of several high-level features like face, body, and background. Each feature is composed of subfeatures like ears, mouth, eyes, and nose for the face, which can be further thought of as combinations of low-level features such as edges [26]. Recent studies have revealed that deep networks can represent hierarchically compositional functions with far fewer parameters than shallow networks [21], implying an information-theoretic lower bound on the sample complexity which is only *polynomial* in the input dimension [24]. While these works offer important insights, they do not characterize the performance of deep neural networks trained with gradient descent (GD).

We investigate this question by adopting the physicist's approach [27–32] of introducing a model of synthetic data, which is inspired by the structure of natural problems, yet simple enough to be investigated systematically. This model (Sec. II) belongs to a family of hierarchical classification problems where the class labels generate the input data via a hierarchy of composition rules. These problems were introduced to highlight the importance of input-to-label correlations for learnability [19] and were found to be learnable via an iterative clustering algorithm [22]. Under the assumption of randomness of the composition rules, we show empirically that shallow networks suffer from the curse of dimensionality (Sec. III), whereas the sample complexity P^* of deep networks (both convolutional networks and multilayer perceptrons) is only polynomial in the size of the input. More specifically, with n_c classes and L composition rules that associate m equivalent low-level representations to each class or high-level feature, $P^* \simeq n_c m^L$ asymptotically in m (Sec. III).

Furthermore, we find that P^* coincides with both (a) the number of data that allows for learning a representation that is invariant to exchanging the m semantically equivalent low-level features (Sec. III A) and (b) the size of the training set for which the correlations between low-level features and class label become detectable (Sec. IV). We prove for a simplified architecture trained with gradient descent that (a) and (b) must indeed coincide. Via (b), P^* can be derived analytically under our assumption of randomness of the composition rules.

A. Relationship to other models of data structure

Characterizing the properties that make high-dimensional data learnable is a classical problem in statistics. Typical assumptions that allow for avoiding the curse of dimensionality include (i) data lying on a low-dimensional manifold and (ii) the task being smooth [33]. For instance, in the context of regression, the sample complexity is not controlled by the bare input dimensionality d, but by the ratio d_M/s [34–36], where d_M is the dimension of the data manifold and s the number of bounded derivatives of the target function. However, d_M is also large in practice [5]; thus, keeping d_M/s low requires an unrealistically large number of bounded derivatives. Moreover, properties (i) and (ii) can already be leveraged by isotropic kernel methods, and thus cannot account for the significant advantage of deep learning methods in many benchmark datasets [37]. Alternatively, learnability can be achieved when (iii) the task depends on a small number of linear projections of the input variables, such as regression of a target function $f^*(x) = g(x_t)$, where $x \in \mathbb{R}^d$ and $x_t \in \mathbb{R}^t$ [38–41]. Methods capable of learning features from the data can leverage this property to achieve a sample complexity that depends on t instead of d [42]. However, one-hidden-layer networks are sufficient for that; hence, this property does not explain the need for deep architectures.

In the context of statistical physics, the quest for a model of data structure has been pursued within the framework of teacher-student models [43–45], where a teacher uses some ground truth knowledge to generate data, while a student tries to infer the ground truth from the data. The structural properties (i)–(iii) can be incorporated into this approach [28,46]. In addition, using a shallow convolutional network as a teacher allows for modeling (iv) the *locality* of imagelike datasets [31,47,48]. In the context of regression, this property can be modeled with a function $f^*(x) = \sum_i f_i^*(x_i)$ where the sum is on all patches x_i of t adjacent pixels. Convolutional networks learn local tasks with a sample complexity controlled by the patch dimension t [47], even in the "lazy" regime [49,50] where they do not learn features. However, locality does not allow for long-range nonlinear dependencies in the task. It might be tempting to include these dependencies by considering a deep convolutional teacher network, but then the sample complexity would be exponential in the input dimension d [25].

The present analysis based on hierarchical generative models shows that properties (i)–(iii) are not necessary to beat the curse of dimensionality. Indeed, for some choices of the parameters, the model generates all possible d-dimensional sequences of input features, which violates (i). Additionally, changing a single input feature has a finite probability of changing the label, violating the smoothness assumption (ii). Finally, the label depends on all of the d input variables of the input, violating (iii). Yet, we find that the sample complexity of deep neural networks is only polynomial in d. Since locality is incorporated hierarchically in the generative process, it generates long-range dependencies in the task, but it can still be leveraged by building a hierarchical representation of the data.

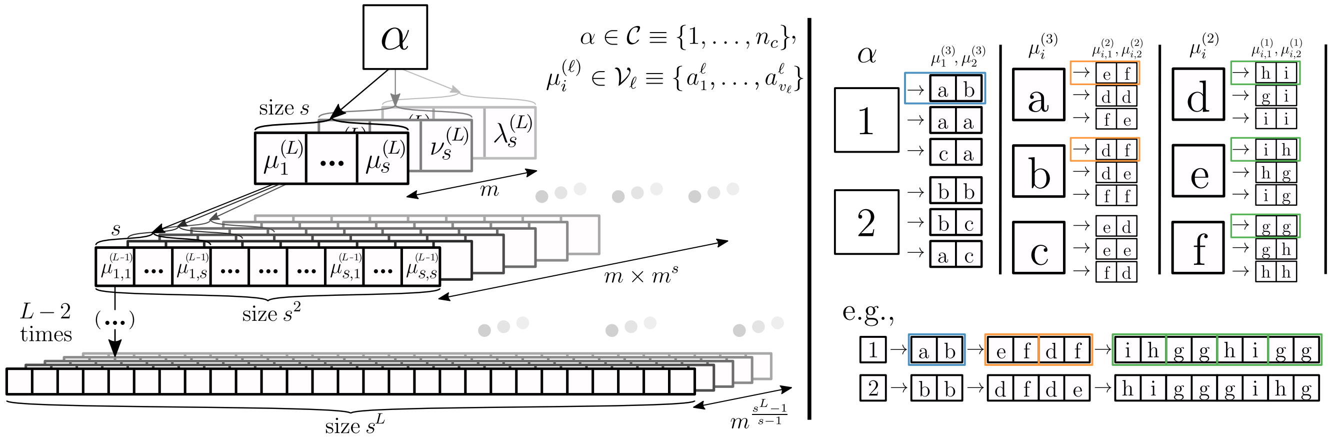
α
$\alpha \in \mathcal{C} \equiv \{1, \ldots, n_c\}$,
$\mu_i^{(\ell)} \in \mathcal{V}_\ell \equiv \{a_1^\ell, \ldots, a_{v_\ell}^\ell\}$
size s
$\mu_1^{(L)}$ … $\mu_s^{(L)}$ $\nu_s^{(L)}$ $\lambda_s^{(L)}$
m
s
$\mu_{1,1}^{(L-1)}$ … $\mu_{1,s}^{(L-1)}$ … … … $\mu_{s,1}^{(L-1)}$ … $\mu_{s,s}^{(L-1)}$
$m \times m^s$
size s^2
$L-2$ times
(…)
size s^L
$m^{\frac{s^L-1}{s-1}}$
α $\mu_1^{(3)}, \mu_2^{(3)}$
1 → a b → a a → c a
2 → b b → b c → a c
$\mu_i^{(3)}$ $\mu_{i,1}^{(2)}, \mu_{i,2}^{(2)}$
a → e f → d d → f e
b → d f → d e → f f
c → e d → e e → f d
$\mu_i^{(2)}$ $\mu_{i,1}^{(1)}, \mu_{i,2}^{(1)}$
d → h i → g i → i i
e → i h → h g → i g
f → g g → g h → h h
e.g.,
1 → a b → e f d f → i h g g h i g g
2 → b b → d f d e → h i g g g i h g

FIG. 1. The random hierarchy model. Left: structure of the generative model. The class label $\alpha = 1, \ldots, n_c$ generates a set of m equivalent (i.e., *synonymic*) high-level representations with elements taken from a vocabulary of high-level features $\mathcal{V}_L$. Similarly, high-level features generate m equivalent lower-level representations, taken from a vocabulary $\mathcal{V}_{L-1}$. Repeating this procedure $L-2$ times yields all the input data with label α, consisting of low-level features taken from $\mathcal{V}_1$. Right: example of random hierarchy model with $n_c = 2$ classes, $L = 3$, $s = 2$, $m = 3$, and homogeneous vocabulary size $v_1 = v_2 = v_3 = 3$. The three sets of rules are listed at the top, while two examples of data generation are shown at the bottom. The first example is obtained by following the rules in the colored boxes.

II. RANDOM HIERARCHY MODEL

In this section, we introduce our generative model, which can be thought of as an L-level context-free grammar—a generative model of language from formal language theory [51]. The model consists of a set of class labels $\mathcal{C} \equiv \{1, \ldots, n_c\}$ and L disjoint vocabularies $\mathcal{V}_\ell \equiv \{a_1^\ell, \ldots, a_{v_\ell}^\ell\}$ of low- and high-level features. As illustrated in Fig. 1, left-hand panel, data are generated from the class labels. Specifically, each label generates m distinct high-level representations via m composition rules of the form

$$\alpha \mapsto \mu_1^{(L)}, \ldots, \mu_s^{(L)} \quad \text{for } \alpha \in \mathcal{C} \quad \text{and} \quad \mu_i^{(L)} \in \mathcal{V}_L, \quad (1)$$

having size $s > 1$. The s elements of these representations are high-level features $\mu_i^{(L)}$ such as background, face, and body for a picture. Each high-level feature generates in turn m lower-level representations via other m rules,

$$\mu^{(\ell)} \mapsto \mu_1^{(\ell-1)}, \ldots, \mu_s^{(\ell-1)} \quad \text{for } \mu^{(\ell)} \in \mathcal{V}_\ell, \quad \mu_i^{(\ell-1)} \in \mathcal{V}_{\ell-1}, \quad (2)$$

from $\ell = L$ down to $\ell = 1$. The input features $\mu^{(1)}$ represent low-level features such as the edges in an image. Because of the hierarchical structure of the generative process, each datum can be represented as a tree of branching factor s and depth L, where the root is the class label, the leaves are the input features, and the hidden nodes are the level-ℓ features with $\ell = 2, \ldots, L$.

In addition, for each level ℓ, there are m distinct rules emanating from the same higher-level feature $\mu^{(\ell)}$; i.e., there are m equivalent lower-level representations of $\mu^{(\ell)}$ (see Fig. 1, right-hand panel, for an example with $m = 3$). Following the analogy with language, we refer to these equivalent representations as *synonyms*. We assume that a single low-level representation there is only one high-level feature that generates it, i.e., that there are no ambiguities. Since the number of distinct s-tuples at level ℓ is bounded by v_ℓ^s, this assumption requires $mv_{\ell+1} \leq v_\ell^s$ for all $\ell = 1, \ldots, L$ (with $v_{L+1} \equiv n_c$). If $m = 1$, each label generates only a single datum and the model is trivial. For $m > 1$, the number of data per class grows exponentially with the input dimension $d = s^L$:

$$m \times m^s \times \cdots \times m^{s^{L-1}} = m^{\sum_{i=0}^{L-1} s^i} = m^{(d-1)/(s-1)}. \quad (3)$$

In particular, in the case where $mv_{\ell+1} = v_\ell^s$, the model generates all the possible data made of d features in $\mathcal{V}_1$. Instead, for $mv_{\ell+1} < v_\ell^s$, the set of available input data is given by the application of the composition rules; therefore, it inherits the hierarchical structure of the model.

Let us remark that, due to the nonambiguity assumption, each set of composition rules can be summarized with a function g_ℓ that associates s-tuples of level-ℓ features to the corresponding level-$(\ell+1)$ feature. The domain of g_ℓ is a subset of $\mathcal{V}_\ell^s$ consisting of the $mv_{\ell+1}$ s-tuples generated by the features at level $(\ell+1)$. Using these functions, the label $\alpha \equiv \mu^{(L+1)}$ of an input datum $\boldsymbol{\mu}^{(1)} = (\mu_1^{(1)}, \ldots, \mu_d^{(1)})$ can be written as a hierarchical composition of L local functions of s variables [20,21]:

$$\mu_i^{(\ell+1)} = g_\ell\left(\mu_{(i-1)s+1}^{(\ell)}, \ldots, \mu_{(i-1)s+1}^{(\ell)}\right), \quad (4)$$

for $i = 1, \ldots, s^{L-\ell}$ and $\ell = 1, \ldots, L$.

Note that, while we keep s and m constant throughout the levels for ease of exposition, our results can be generalized without additional effort. Likewise, we will set the vocabulary size to v for all levels. To sum up, a single classification task is specified by the parameters n_c, v, m, and s and by the L composition rules. In the random hierarchy model (RHM) the composition rules are chosen uniformly at random over all the possible assignments of m representations of s low-level features to each of the v high-level features. An example of binary classification task ($n_c = 2$), with $s = 2$, $L = 3$, and $v = m = 3$, is shown in Fig. 1, right-hand panel, together with two examples of label-input pairs. Notice that the random choice induces correlations between low- and high-level features. In simple terms, each of the high-level features, e.g., the level-2 features d, e, or f in the figure, is more likely to be represented with a certain low-level feature in a given position, e.g., i on the right for d, g on the right for e, and h on the right for f. These correlations are crucial for our predictions and are analyzed in detail in Appendix C.

III. SAMPLE COMPLEXITY OF DEEP NEURAL NETWORKS

The main focus of our work is the answer to the following question.

How much data is required to learn a typical instance of the random hierarchy model with a deep neural network?

Thus, after generating an instance of the RHM with fixed parameters n_c, s, m, v, and L, we train neural networks of varying depth with stochastic gradient descent (SGD) on a set of P training points. The training points are sampled uniformly at random without replacement from the set of available RHM data; hence, they are all distinct. We adopt a one-hot encoding of the input features, so that each input point $\boldsymbol{x}$ is a $(d \times v)$-dimensional sequence where, for $i = 1, \ldots, d$ and $\nu \in \mathcal{V}_1$,

$$x_{i,\nu} = \begin{cases} 1 & \text{if } \mu_i^{(1)} = \nu \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

All our experiments consider overparametrized networks, which we achieve in practice by choosing the width H of the network's hidden layers such that (i) training loss reaches 0 and (ii) test accuracy does not improve by increasing H. To guarantee representation learning as H grows, we consider the maximal update parametrization [52], equivalent to having the standard $H^{-1/2}$ scaling of the hidden layer weights plus an extra factor of $H^{-1/2}$ at the last layer. Further details of the machine learning methods can be found in Appendix A.

(a) *Shallow networks are cursed.* Let us begin with the sample complexity of two-layer fully connected networks. As shown in Fig. 2, in the maximal case $n_c = v$, $m = v^{s-1}$ these networks learn the task only if trained on a significant fraction of the total number of data $P_{\max}$. From Eq. (3),

$$P_{\max} = n_c m^{(d-1)/(s-1)}, \tag{6}$$

which equals v^{s^L} in the maximal case. The bottom panel of Fig. 2, in particular, highlights that the number of training data required for having a test error $\epsilon \leq 0.7\epsilon_{\text{rand}}$, with $\epsilon_{\text{rand}} = 1 - n_c^{-1}$ denoting the error of a random guess of the label, is proportional to $P_{\max}$. Since $P_{\max}$ is exponential in d, this is an instance of the curse of dimensionality.

(b) *Deep networks break the curse.* For networks having a depth larger than that of the RHM L, the test error displays a sigmoidal behavior as a function of the training set size. This finding is illustrated in the top panels of Figs. 3 and 4 (and Fig. 12 of Appendix F for varying n_c) for convolutional neural networks (CNNs) of depth $L + 1$ (details in Appendix A). Similar results are obtained for multilayer perceptions of depth $> L$, as shown in Appendix F. All these results suggest the existence of a well-defined number of training data at which the task is learned. Mathematically, we define the sample complexity P^* as the smallest training set size P such that the test error $\epsilon(P)$ is smaller than $\epsilon_{\text{rand}}/10$. The bottom panels of Figs. 3 and 4 (and Figs. 12 and 13) show that

$$P^* \simeq n_c m^L \Leftrightarrow \frac{P^*}{n_c} \simeq d^{\ln(m)/\ln(s)}, \tag{7}$$

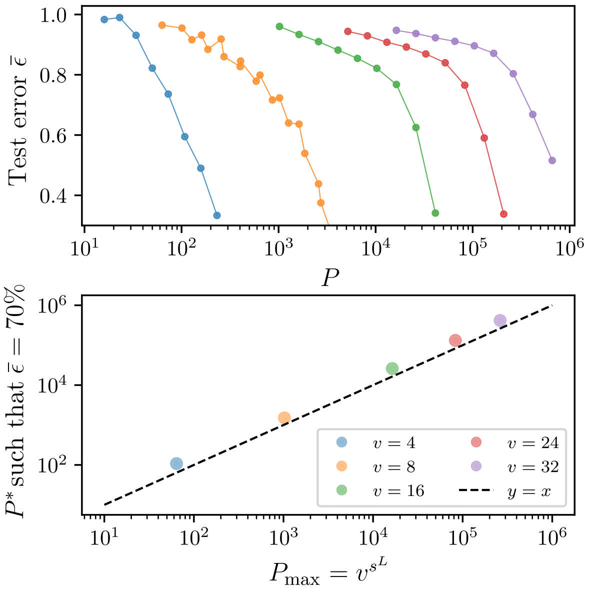

FIG. 2. Sample complexity of two-layer fully connected networks, for $L = s = 2$ and $v = n_c = m$. Top: test error versus the number of training data. Different colors correspond to different vocabulary sizes v. Bottom: number of training data resulting in test error $\bar{\epsilon} = 0.7$ as a function of $P_{\max}$, with the black dashed line indicating a linear relationship.

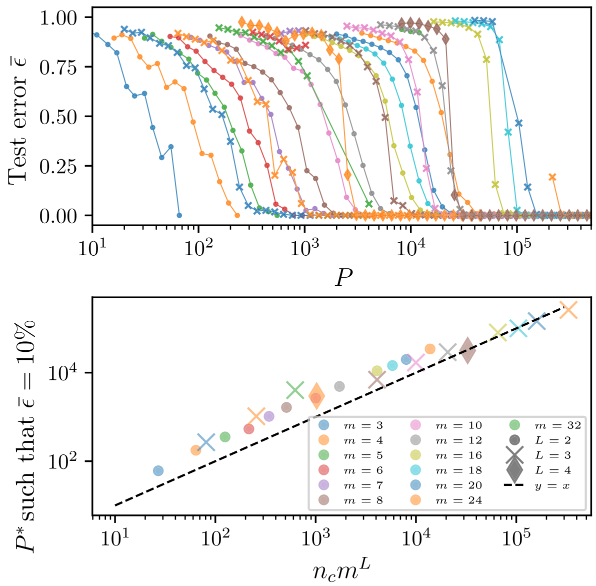

FIG. 3. Sample complexity of depth-$(L+1)$ CNNs, for $s = 2$ and $m = n_c = v$. Top: test error versus number of training points. Different colors correspond to different vocabulary sizes v while the markers indicate the hierarchy depth L. Bottom: sample complexity P^* corresponding to a test error $\epsilon^* = 0.1\epsilon_{\text{rand}}$. The empirical points show remarkable agreement with the law $P^* = n_c m^L$, shown as a black dashed line.

independently of the vocabulary size v. Since P^* is a power of the input dimension $d = s^L$, the curse of dimensionality is beaten, which evidences the ability of deep networks to harness the hierarchical compositionality of the task. It is crucial to note, however, that this ability manifests only in feature learning regimes, e.g., under the maximal update parametrization considered in this work. Conversely, as shown in Fig. 14 of Appendix F for the maximal case $n_c = v$, $m = v^{s-1}$, deep networks trained in the "lazy" regime [49]—where they do not learn features—suffer from the curse of dimensionality, even when their architecture is matched to the structure of the RHM.

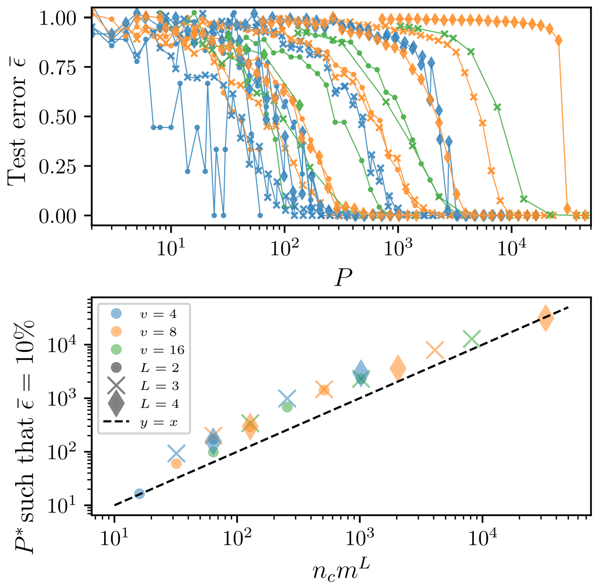

FIG. 4. Sample complexity of depth-$(L+1)$ CNNs, for $s = 2$, $n_c = v$ and varying $m \leq v$. Top: test error versus number of training points, with different colors corresponding to different vocabulary sizes v and markers indicating the hierarchy depth L. Bottom: sample complexity P^*, with the law $P^* = n_c m^L$ shown as a black dashed line.

We now turn to study the internal representations of trained networks and the mechanism that they employ to solve the task.

A. Emergence of synonymic invariance in deep CNNs

A natural approach to learning the RHM would be to identify the sets of s-tuples of input features that correspond to the same higher-level feature, i.e., synonyms. Identifying synonyms at the first level would allow for replacing each s-dimensional patch of the input with a single symbol, reducing the dimensionality of the problem from s^L to s^{L-1}. Repeating this procedure L times would lead to the class labels and, consequently, to the solution of the task.

To test if deep networks trained on the RHM resort to a similar solution, we introduce the *synonymic sensitivity*, which is a measure of the invariance of a function with respect to the exchange of synonymic low-level features. Mathematically, we define $S_{k,l}$ as the sensitivity of the kth layer representation of a deep network with respect to exchanges of synonymous s-tuples of level-l features. Namely,

$$S_{k,l} = \frac{\langle \| f_k(\boldsymbol{x}) - f_k(P_l \boldsymbol{x}) \|^2 \rangle_{\boldsymbol{x}, P_l}}{\langle \| f_k(\boldsymbol{x}) - f_k(\boldsymbol{y}) \|^2 \rangle_{\boldsymbol{x}, \boldsymbol{y}}}, \tag{8}$$

where f_k is the sequence of activations of the kth layer in the network, P_l is an operator that replaces all the level-l tuples with one of their $m-1$ synonyms chosen uniformly at random, $\langle\cdot\rangle$ with subscripts $\boldsymbol{x}$, $\boldsymbol{y}$ denotes average over pairs of input data of an instance of the RHM, and the subscript P_l denotes average over all the exchanges of synonyms.

Figure 5 reports $S_{2,1}$, which measures the sensitivity to exchanges of synonymic tuples of input features, as a function of the training set size P for deep CNNs trained on RHMs with different parameters. We focused on $S_{2,1}$—the sensitivity of the second layer of the network—since a single linear transformation of the input cannot produce an invariant representation in general [53]. Note that all the curves display a sigmoidal shape, signaling the existence of a characteristic sample size which marks the emergence of synonymic sensitivity in the learned representations. Remarkably, by rescaling the x axis by the sample

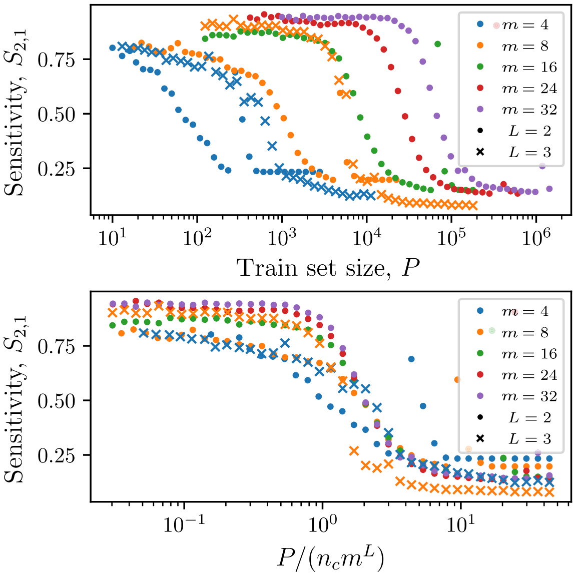

FIG. 5. Synonymic sensitivity $S_{2,1}$ for a depth-$(L+1)$ CNN trained on the RHM with $s = 2$, $n_c = m = v$ as a function of the training set size (L and v as in the key). The collapse achieved after rescaling by $P^* = n_c m^L$ highlights that the sample complexity coincides with the number of training points required to build internal representations invariant to exchanging synonyms.

complexity of Eq. (7) (bottom panel of Fig. 5), curves corresponding to different parameters collapse. We conclude that the generalization ability of a network relies on the synonymic invariance of its hidden representations.

Measures of the synonymic sensitivity $S_{k,1}$ for different layers k are reported in Fig. 6 (blue lines), showing indeed that the layers $k \geq 2$ become insensitive to exchanging level-1 synonyms. Figure 6 also shows the sensitivities to exchanges of higher-level synonyms: All levels are learned together as P increases, and invariance to level-l exchanges is achieved from layer $k = l + 1$. The test error is also shown (gray dashed lines) to further emphasize its correlation with synonymic invariance.

(a) *Synonymic invariance and effective dimension.* Note that the collapse of the representations of synonymic tuples to the same value implies a progressive reduction of the effective dimensionality of the hidden representations, as reported in Fig. 11 of Appendix E.

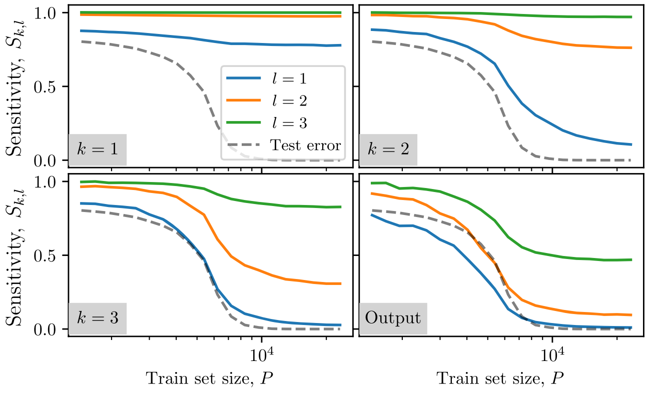

FIG. 6. Synonymic sensitivities $S_{k,l}$ of the layers of a depth-$(L+1)$ CNN trained on an RHM with $L = 3$, $s = 2$, $n_c = m = v = 8$, as a function of the training set size P. The colors denote the level of the exchanged synonyms (as in the key), whereas different panels correspond to the sensitivity of the activations of different layers (layer index in the gray box). Synonymic invariance is learned at the same training set size for all layers, and invariance to level-l exchanges is obtained from layer $k = l + 1$.

IV. CORRELATIONS GOVERN SYNONYMIC INVARIANCE

We now provide a theoretical argument for understanding the scaling of P^* of Eq. (7) with the parameters of the RHM. First, we compute a third characteristic sample size P_c, defined as the size of the training set for which the *local* correlations between any of the input patches and the label become detectable. Remarkably, P_c coincides with P^* of Eq. (7). Second, we demonstrate how a shallow (two-layer) neural network acting on a single patch can use such correlations to build a synonymic invariant representation in a single step of gradient descent so that P_c and P^* also correspond to the emergence of an invariant representation. Last, we show empirically that removing such correlations leads again to the curse of dimensionality, even if the network architecture is matched to the structure of the RHM.

A. Identify synonyms by counting

Groups of input patches forming synonyms can be inferred by counting, at any given location, the occurrences of such patches in all the data corresponding to a given class α. Indeed, tuples of features that appear with identical frequencies are likely synonyms. More specifically, let us denote $\boldsymbol{x}_j$ an s-dimensional input patch for j in $1, \ldots, s^{L-1}$, an s-tuple of input features with $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_s)$, and the number of data in class α having $\boldsymbol{x}_j = \boldsymbol{\mu}$ with $N_j(\boldsymbol{\mu}; \alpha)$ [54]. Normalizing this number by $N_j(\boldsymbol{\mu}) = \sum_\alpha N_j(\boldsymbol{\mu}; \alpha)$ yields the conditional probability $f_j(\alpha|\boldsymbol{\mu})$ for a datum to belong to class α conditioned on displaying the s-tuple $\boldsymbol{\mu}$ in the jth input patch:

$$f_j(\alpha|\boldsymbol{\mu}) \coloneqq \Pr\{\boldsymbol{x} \in \alpha | \boldsymbol{x}_j = \boldsymbol{\mu}\} = \frac{N_j(\boldsymbol{\mu}; \alpha)}{N_j(\boldsymbol{\mu})}. \qquad (9)$$

If the low-level features are homogeneously spread across classes, then $f = n_c^{-1}$ for all α, $\boldsymbol{\mu}$, and j. In contrast, due to the aforementioned correlations, the probabilities of the RHM are all different from n_c^{-1}—we refer to this difference as *signal*. Distinct level-1 tuples $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$ yield a different f (and thus a different signal) with high probability unless $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$ are synonyms, i.e., they share the same level-2 representation. Therefore, this signal can be used to identify synonymous level-1 tuples.

B. Signal versus sampling noise

When measuring the conditional class probabilities with only P training data, the occurrences in the right-hand side of Eq. (9) are replaced with empirical occurrences, which induce a sampling *noise* on the f's. For the identification of synonyms to be possible, this noise must be smaller in magnitude than the aforementioned signal—a visual representation of the comparison between signal and noise is depicted in Fig. 7.

The magnitude of the signal can be computed as the ratio between the standard deviation and mean of $f_j(\alpha|\boldsymbol{\mu})$ over realizations of the RHM. The full calculation is presented in Appendix C: Here we present a simplified argument based on an additional independence assumption. Given a class α, the tuple $\boldsymbol{\mu}$ appearing in the jth input patch is determined by a sequence of L choices—one choice per level of the hierarchy—of one among m possible lower-level representations. These m^L possibilities lead to all the mv distinct input s-tuples. $N_j(\boldsymbol{\mu};\alpha)$ is proportional to how often the tuple $\boldsymbol{\mu}$ is chosen—$m^L/(mv)$ times on average. Under the assumption of independence of the m^L choices, the fluctuations of $N_j(\boldsymbol{\mu};\alpha)$ relative to its mean are given by the central limit theorem and read $[m^L/(mv)]^{-1/2}$ in the limit of large m. If n_c is sufficiently large, the fluctuations of $N_j(\boldsymbol{\mu})$ are negligible in comparison. Therefore, the relative fluctuations of f_j are the same as those of $N_j(\boldsymbol{\mu};\alpha)$, and the size of the signal is $[m^L/(mv)]^{-1/2}$.

The magnitude of the noise is given by the ratio between the standard deviation and mean, over independent samplings of a training set of fixed size P, of the empirical conditional probabilities $\hat{f}_j(\alpha|\boldsymbol{\mu})$. Only $P/(n_c mv)$ of the training points will, on average, belong to class α while displaying feature μ in the jth patch. Therefore, by the convergence of the empirical measure to the true probability, the sampling fluctuations of $\hat{f}$ relative to the mean are of order $[P/(n_c mv)]^{-1/2}$—see Appendix C for a detailed derivation. Balancing signal and noise yields the characteristic P_c for the emergence of correlations. For large m, n_c, and P,

$$P_c = n_c m^L, \tag{10}$$

which coincides with the empirical sample complexity of deep networks discussed in Sec. III.

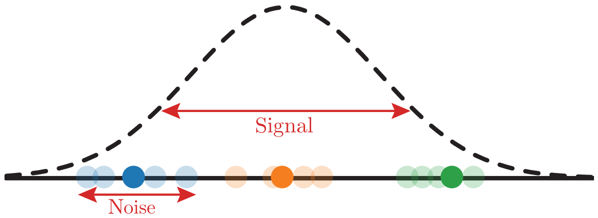

FIG. 7. Signal versus noise illustration. The dashed function represents the distribution of $f(\alpha|\boldsymbol{\mu})$ resulting from the random sampling of the RHM rules. The solid dots illustrate the *true* frequencies $f(\alpha|\boldsymbol{\mu})$ sampled from this distribution, with different colors corresponding to different groups of synonyms. The typical spacing between the solid dots, given by the width of the distribution, represents the *signal*. Transparent dots represent the empirical frequencies $\hat{f}_j(\alpha|\boldsymbol{\mu})$, with dots of the same color corresponding to synonymous features. The spread of transparent dots of the same color, which is due to the finiteness of the training set, represents the *noise*.

C. Learning level-1 synonyms with one step of gradient descent

To complete the argument, we consider a simplified one-step gradient descent setting [55,56], where P_c marks the number of training examples required to learn a synonymic invariant representation. In particular, we focus on the s-dimensional patches of the data and study how a two-layer network acting on one of such patches learns the first composition rule of the RHM by building a representation invariant to exchanges of level-1 synonyms.

Let us then sample an instance of the RHM and P input-label pairs $(\boldsymbol{x}_{k,1},\alpha_k)$ with $\alpha_k \coloneqq \alpha(\boldsymbol{x}_k)$ for all $k = 1,\dots,P$ and $\boldsymbol{x}_{k,1}$ denoting the first s patch of the datum $\boldsymbol{x}_k$. The network output reads

$$\mathcal{F}_{\mathrm{NN}}(\boldsymbol{x}_1) = \frac{1}{H}\sum_{h=1}^{H} a_h \sigma(\boldsymbol{w}_h\cdot\boldsymbol{x}_1), \tag{11}$$

where the inner-layer weights $\boldsymbol{w}_h$'s have the same dimension as $\boldsymbol{x}_1$, the top-layer weights a_h's are n_c dimensional, and $\sigma(x) = \max(0,x)$ is the rectified linear unit (ReLU) activation function. To further simplify the problem, we represent $\boldsymbol{x}_1$ as a v^s-dimensional one-hot encoding of the corresponding s-tuple of features. This representation is equivalent to an orthogonalization of the input points. In addition, the top-layer weights are initialized as i.i.d. Gaussian with zero mean and unit variance and fixed, whereas the $\boldsymbol{w}_h$'s are initialized with all their elements set to 1 and trained by gradient descent on the empirical cross-entropy loss:

$$\mathcal{L} = \frac{1}{P}\sum_{k=1}^{P}\left[-\log\left(\frac{e^{[\mathcal{F}_{\mathrm{NN}}(\boldsymbol{x}_{k,1})]_{\alpha(\boldsymbol{x}_k)}}}{\sum_{\beta=1}^{n_c} e^{[\mathcal{F}_{\mathrm{NN}}(\boldsymbol{x}_{k,1})]_{\beta}}}\right)\right]. \tag{12}$$

Finally, we consider the mean-field limit $W \to \infty$, so that, at initialization, $\mathcal{F}_{\mathrm{NN}}^{(0)} = 0$ identically.

Let us denote with $\boldsymbol{\mu}(\boldsymbol{x}_1)$ the s-tuple of features encoded in $\boldsymbol{x}_1$. Because of the one-hot encoding, $f_h(\boldsymbol{x}_1) \coloneqq \boldsymbol{w}_h\cdot\boldsymbol{x}_1$ coincides with the $\boldsymbol{\mu}(\boldsymbol{x}_1)$th component of the weight $\boldsymbol{w}_h$.

This component, which is set to 1 at initialization, is updated by (minus) the corresponding component of the gradient of the loss in Eq. (12). Recalling also that the predictor is 0 at initialization, we get

$$\begin{aligned}\Delta f_h(\boldsymbol{x}_1) &= -\nabla_{(\boldsymbol{w}_h)_{\boldsymbol{\mu}(\boldsymbol{x}_1)}}\mathcal{L}\\ &= \frac{1}{P}\sum_{k=1}^{P}\sum_{\alpha=1}^{n_c} a_{h,\alpha}\delta_{\boldsymbol{\mu}(\boldsymbol{x}_1),\boldsymbol{\mu}(\boldsymbol{x}_{k,1})}\left(\delta_{\alpha,\alpha(\boldsymbol{x}_k)} - \frac{1}{n_c}\right)\\ &= \sum_{\alpha=1}^{n_c} a_{h,\alpha}\left(\frac{\hat{N}_1(\boldsymbol{\mu}(\boldsymbol{x}_1);\alpha)}{P} - \frac{1}{n_c}\frac{\hat{N}_1(\boldsymbol{\mu})}{P}\right), \qquad (13)\end{aligned}$$

where $\hat{N}_1(\boldsymbol{\mu})$ is the empirical occurrence of the s-tuple $\boldsymbol{\mu}$ in the first patch of the P training points and $\hat{N}_1(\boldsymbol{\mu};\alpha)$ is the (empirical) joint occurrence of the s-tuple $\boldsymbol{\mu}$ and the class label α. As P increases, the empirical occurrences $\hat{N}$ converge to the true occurrences N, which are invariant for the exchange of synonym s-tuples $\boldsymbol{\mu}$. Hence, the hidden representation is also invariant for the exchange of synonym s-tuples in this limit.

This prediction is confirmed empirically in Fig. 8, which shows the sensitivity $S_{1,1}$ of the hidden representation [57] of shallow fully connected networks trained in the setting of this section, as a function of the number P of training data for different combinations of the model parameters. The bottom panel, in particular, highlights that the sensitivity is close to 1 for $P \ll P_c$ and close to 0 for $P \gg P_c$. In addition, notice that the collapse of the preactivations of synonymic tuples onto the same, synonymic invariant value, implies that the rank of the hidden weights matrix tends to v—the vocabulary size of higher-level features. This low-rank structure is typical in the weights of deep networks trained on image classification [58–61].

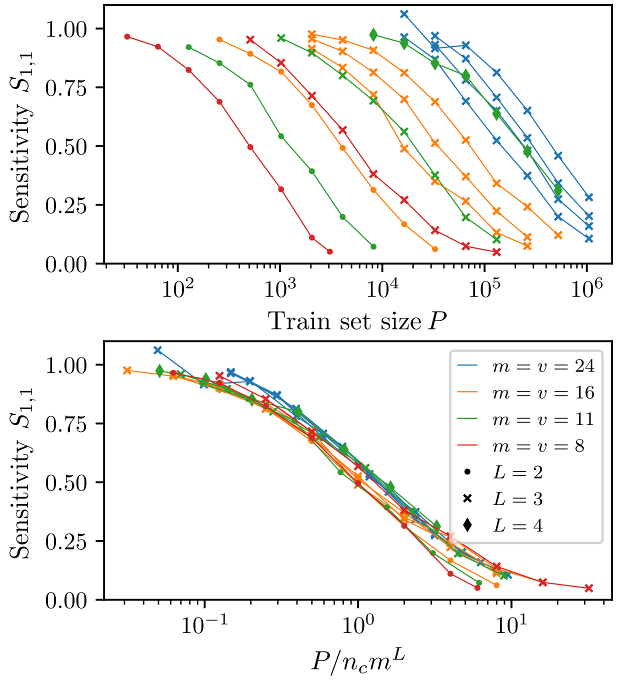

FIG. 8. Synonymic sensitivity of the hidden representation versus P for a two-layer fully connected network trained on the first patch of the inputs of an RHM with $s = 2$ and $m = v$, for varying L, v, and n_c. The top panel shows the bare curves whereas, in the bottom panel, the x axis is rescaled by $P_c = n_c m^L$. The collapse of the rescaled curves highlights that P_c coincides with the number of training data for building a synonymic invariant representation.

(a) *Including all patches via weight sharing.* Let us remark that one can easily extend the one-step setting to include the information from all the input patches, for instance, by replacing the network in Eq. (11) with a one-hidden-layer convolutional network with filter size s and nonoverlapping patches. Consequently, the empirical occurrences on the right-hand side of Eq. (13) would be replaced with average occurrences over the patches. However, this average results in a reduction of both the signal and the sampling noise contributions to the empirical occurrences by the same factor $\sqrt{s^{L-1}}$. Therefore, weight sharing does not affect the sample size required for synonymic invariance in the one-step setting.

(b) *Improved sample complexity via clustering.* A distance-based clustering method acting on the representations of Eq. (13) can actually identify synonyms at $P \simeq \sqrt{n_c} m^L = P_c/\sqrt{n_c}$, which is much smaller than P_c in the large-n_c limit. Intuitively, using a sequence instead of a scalar amplifies the signal by a factor n_c and the sampling noise by a factor $\sqrt{n_c}$, improving the signal-to-noise ratio. We show that this is indeed the case in Appendix D for the maximal dataset case $n_c = v$ and $m = v^{s-1}$. Previous theoretical studies have considered the possibility of intercalating clustering steps in standard gradient descent methods [22,62], but the question of whether deep learning methods can achieve a similar sample complexity with standard end-to-end training remains open.

D. Curse of dimensionality without correlations

To support the argument that learning is possible because of the detection of local input-label correlations, we show that their removal in the RHM leads to a sample complexity exponential in d, even for deep networks. Removing such correlations implies that, at any level, features are uniformly distributed among classes. This is achieved enforcing that a tuple $\boldsymbol{\mu}$ in the jth patch at level ℓ belongs to a class α with probability n_c^{-1}, independently on $\boldsymbol{\mu}$, j, ℓ, and α, as discussed in Sec. IV A. Such procedure produces an uncorrelated version of the RHM, which generalizes the parity problem (realized for $m = v = n_c = 2$), a task that cannot be learned efficiently with gradient-based

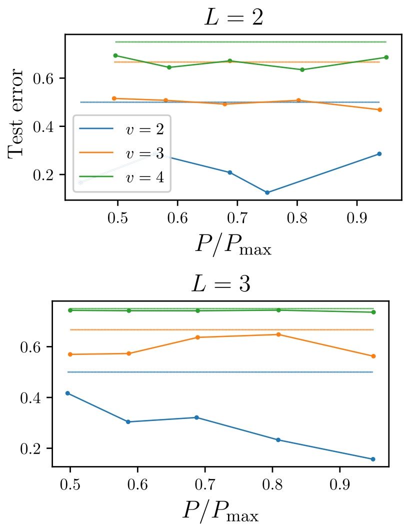

FIG. 9. Test error of depth-$(L+1)$ CNNs trained on uncorrelated RHM versus number P of training points rescaled with $P_{\max}$, with $s = 2$ and $m = n_c = v$ with different v (different colors), for $L = 2$ (top) and $L = 3$ (bottom). Horizontal dashed lines stand for ϵ_{rand}, given by guessing the label uniformly at random.

methods [63]. Indeed, deep CNNs with depth $L+1$, trained on this uncorrelated RHM, are cursed by dimensionality, as shown in Fig. 9. The CNN test error is close to ϵ_{rand}, given by randomly guessing the label, even for $P/P_{\max} > 0.9$, particularly for $v > 2$.

V. CONCLUSION

What makes real-world tasks learnable? This question extends from machine learning to brain science [64]. To start thinking quantitatively about it, we introduced the random hierarchy model: a family of tasks that captures the compositional structure of natural data. We showed that neural networks can learn such tasks with a limited training set, by developing a hierarchical representation of the data. Overall, these results rationalize several phenomena associated with deep learning.

First, our finding that for hierarchical tasks the sample complexity is polynomial in the input dimension (and not exponential) leads to a plausible explanation for the learnability of real-world tasks. Moreover, our results provide a rule of thumb for estimating the order of magnitude of the sample complexity of benchmark datasets. In the case of CIFAR10 [65], for instance, having 10 classes, taking reasonable values for task parameters such as $m \in [5, 15]$ and $L = 3$, yields $P^* \in [10^3, 3 \times 10^4]$, comparable with the sample complexity of modern architectures (see Fig. 15).

Second, our results quantify the intuition that depth is crucial to building a hierarchical representation that effectively lowers the dimension of the problem, and allows for avoiding the curse of dimensionality. On the one hand, this result gives a foundation to the claim that deep is better than shallow, beyond previous analyses that focused on expressivity [21,24] rather than learning. On the other hand, our result that the internal representations of trained networks mirror the hierarchical structure of the task explains why these representations become increasingly complex with depth in real-world applications [9,10].

Furthermore, we provided a characterization of the internal representations based on their sensitivity toward transformations of the low-level features that leave the class label unchanged. This viewpoint complements existing ones that focus instead on the input features that maximize the response of hidden neurons, thus enhancing the interpretability of neural nets. In addition, our approach bypasses several issues of previous characterizations. For example, approaches based on mutual information [12] are ill defined when the network representations are deterministic functions of the input [13], whereas those based on intrinsic dimension [14,15] can display counterintuitive results—see Appendix E for a deeper discussion of the intrinsic dimension and on how it behaves in our framework.

Finally, our study predicts a fundamental relationship between sample complexity, correlations between low-level features and labels, and the emergence of invariant representations. This prediction can be tested beyond the context of our model, for instance, by studying invariance to exchanging synonyms in language modeling tasks.

Looking forward, the random hierarchy model is a suitable candidate for the clarification of other open questions in the theory of deep learning. For instance, a formidable challenge is to obtain a detailed description of the gradient descent dynamics of deep networks. Indeed, dynamics may be significantly easier to analyze in this model, since quantities characterizing the network success, such as sensitivity to synonyms, can be delineated. In addition, the model could be generalized to describe additional properties of data, e.g., noise in the form of errors in the composition rules or inhomogeneities in the frequencies at which high-level features generate low-level representations. The latter, in particular, would generate data where certain input features are more abundant than others and, possibly, to a richer learning scenario with several characteristic training set sizes.

Beyond supervised learning, in the random hierarchy model the set of available input data inherits the hierarchical structure of the generative process. Thus, this model offers a new way to study the effect of compositionality on self-supervised learning or probabilistic generative models—extremely powerful techniques whose understanding is still in its infancy.

The codes that support the findings of this paper are openly available [66–68].

ACKNOWLEDGMENTS

The authors thank Antonio Sclocchi for fruitful discussions and helpful feedback on the manuscript. This work was supported by a grant from the Simons Foundation (No. 454953 for M. W.).

APPENDIX A: METHODS

1. RHM implementation

The code implementing the RHM is available online [66]. The inputs sampled from the RHM are represented as a one-hot encoding of low-level features so that each input consists of s^L pixels and v channels (size $s^L \times v$). The input pixels are whitened over channels; i.e., each pixel has zero mean and unit variance over the channels.

2. Machine learning models

We consider both generic deep neural networks and deep convolutional networks tailored to the structure of the RHM. Generic deep neural networks are made by stacking *fully connected* layers, i.e., linear transformations of the kind

$$x \in \mathbb{R}^{d_{\text{in}}} \to d_{\text{in}}^{-1/2} W \cdot x + b \in \mathbb{R}^{d_{\text{out}}}, \qquad \text{(A1)}$$

where W is a $d_{\text{out}} \times d_{\text{in}}$ matrix of weights, b is a d_{out} sequence of biases, and the factor $d_{\text{in}}^{-1/2}$ guarantees that the outputs remain of order 1 when d_{in} is varied. *Convolutional* layers, instead, act on imagelike inputs that have a spatial dimension d and c_{in} channels and compute the convolution of the input with a filter of spatial size f. This operation is equivalent to applying the linear transformation of Eq. (A1) to input patches of spatial size f, i.e., groups of f adjacent pixels [dimension $d_{\text{in}} = (f \times c_{\text{in}})$]. The output has an imagelike structure analogous to that of the input, with spatial dimension depending on how many patches are considered. In the *nonoverlapping patches* case, for instance, the spatial dimension of the output is d/f.

For all layers but the last, the linear transformation is followed by an elementwise nonlinear activation function σ. We resort to the popular ReLU $\sigma(x) = \max(0, x)$. The output dimension is always fixed to the number of classes n_c, while the input dimension of the first layer is the same as the input data: spatial dimension s^L and v channels, flattened into a single $s^L \times v$ sequence when using a fully connected layer. The dimensionalities of the other *hidden* layers are set to the same constant H throughout the network. Following the maximal update parametrization [69], the weights of the last layer are multiplied by an additional factor H^{-1}. This factor causes the output at initialization to vanish as H grows, which induces representation learning even in the $H \to \infty$ limit. In practice, we set $H = (4\text{–}8) \times v^s$. Increasing this number further does not affect any of the results presented in the paper.

To tailor deep CNNs to the structure of the RHM, we set $f = s$ so that, in the nonoverlapping patches setting, each convolutional filter acts on a group of s low-level features that correspond to the same higher-level feature. Since the spatial dimensionality of the input is s^L and each layer reduces it by s, the number of nonlinear layers in a tailored CNN is fixed to the depth of the RHM L, so that the network depth is $L + 1$. Fully connected networks, instead, can have any depth. The code for the implementation of both architectures is available online [67].

3. Training procedure

Training is performed within the PYTORCH deep learning framework [70]. Neural networks are trained on P training points sampled uniformly at random from the RHM data, using stochastic gradient descent on the cross-entropy loss. The batch size is 128 for $P \geq 128$ and P otherwise, the learning rate is initialized to 10^{-1} and follows a cosine annealing schedule which reduces it to 10^{-2} over 100 epochs. Training stops when the training loss reaches 10^{-3}. The corresponding code is available online [68].

The performance of the trained models is measured as the classification error on a test set. The size of the test set is set to $\min(P_{\max} - P, 20\,000)$. Synonymic sensitivity, as defined in Eq. (8), is measured on a test set of size $\min(P_{\max} - P, 1000)$. Reported results for a given value of RHM parameters are averaged over 10 jointly different instances of the RHM and network initialization.

APPENDIX B: STATISTICS OF THE COMPOSITION RULES

In this appendix, we consider a single composition rule, that is the assignment of m s-tuples of low-level features to each of the v high-level features. In the RHM these rules are chosen uniformly at random over all the possible rules; thus their statistics are crucial in determining the correlations between the input features and the class label.

1. Statistics of a single rule

For each rule, we call $N_i(\mu_1; \mu_2)$ the number of occurrences of the low-level feature μ_1 in position i of the s-tuples generated by the higher-level feature μ_2. The probability of $N_i(\mu_1; \mu_2)$ is that of the number of successes when drawing m (number of s-tuples associated with the high-level feature μ_2) times without replacement from a pool of v^s (total number of s-tuples with vocabulary size v) objects where only v^{s-1} satisfy a certain condition (number of s-tuples displaying feature μ_1 in position i):

$$\Pr\{N_i(\mu_0; \mu_1) = k\} = \binom{v^{s-1}}{k} \binom{v^s - v^{s-1}}{m - k} \Bigg/ \binom{v^s}{m}, \qquad \text{(B1)}$$

which is a hypergeometric distribution $\text{H}_{v^s, v^{s-1}, m}$, with mean

$$\langle N\rangle = m\frac{v^{s-1}}{v^s} = \frac{m}{v}, \tag{B2}$$

and variance

$$\sigma_N^2 := \langle (N-\langle N\rangle)^2\rangle = m\frac{v^{s-1}}{v^s}\frac{v^s - v^{s-1}}{v^s}\frac{v^s-m}{v^s-1} \\ = \frac{m}{v}\frac{v-1}{v}\frac{v^s-m}{v^s-1} \xrightarrow{m\gg 1} \frac{m}{v}, \tag{B3}$$

independently of the position i and the specific low- and high-level features. Note that, since $m \le v^{s-1}$ with s fixed, large m implies also large v.

2. Joint statistics of a single rule

(a) *Shared high-level feature.* For a fixed high-level feature μ_2, the joint probability of the occurrences of two different low-level features μ_1 and ν_1 is a multivariate hypergeometric distribution,

$$\Pr\{N_i(\mu_1;\mu_2)=k; N_i(\nu_1;\mu_2)=l\} \\ = \binom{v^{s-1}}{k}\binom{v^{s-1}}{l}\binom{v^s-2v^{s-1}}{m-k-l}\Big/\binom{v^s}{m}, \tag{B4}$$

giving the following covariance:

$$c_N := \langle (N_i(\mu_1;\mu_2)-\langle N\rangle)(N_i(\nu_1;\mu_2)-\langle N\rangle)\rangle \\ = -\frac{m}{v^2}\frac{v^s-m}{v^s-1} \xrightarrow{m\gg 1} -\left(\frac{m}{v}\right)^2\frac{1}{m}. \tag{B5}$$

The covariance can also be obtained via the constraint $\sum_{\mu_1} N_i(\mu_1;\mu_2)=m$. For any finite sequence of identically distributed random variables X_μ with a constraint on the sum $\sum_\mu X_\mu = m$:

$$\sum_{\mu=1}^{v} X_\mu = m \Rightarrow \sum_{\mu=1}^{v}(X_\mu - \langle X_\mu\rangle) = 0 \\ \Rightarrow (X_\nu - \langle X_\nu\rangle)\sum_{\mu=1}^{v}(X_\mu - \langle X_\mu\rangle) = 0 \\ \Rightarrow \sum_{\mu=1}^{v}\langle (X_\nu - \langle X_\nu\rangle)(X_\mu - \langle X_\mu\rangle)\rangle = 0 \\ \Rightarrow \mathrm{Var}[X_\mu] + (v-1)\mathrm{Cov}[X_\mu, X_\nu] = 0. \tag{B6}$$

In the last line, we used the identically distributed variables hypothesis to replace the sum over $\mu\neq\nu$ with the factor $(v-1)$. Therefore,

$$c_N = \mathrm{Cov}[N_i(\mu_1;\mu_2), N_i(\nu_1;\mu_2)] \\ = -\frac{\mathrm{Var}[N_i(\mu_1;\mu_2)]}{v-1} = -\frac{\sigma_N^2}{v-1}. \tag{B7}$$

(b) *Shared low-level feature.* The joint probability of the occurrences of the same low-level feature μ_1 starting from different high-level features $\mu_2\neq\nu_2$ can be written as follows,

$$\Pr\{N(\mu_1;\mu_2)=k; N(\mu_1;\nu_2)=l\} \\ = \Pr\{N(\mu_1;\mu_2)=k | N(\mu_1;\nu_2)=l\} \\ \times \Pr\{N(\mu_1;\nu_2)=l\} \\ = \mathrm{H}_{v^s-m, v^{s-1}-l, m}(k)\times \mathrm{H}_{v^s, v^{s-1}, m}(l), \tag{B8}$$

resulting in the following "interfeature" covariance:

$$c_{if} := \mathrm{Cov}[N_i(\mu_1;\mu_2), N_i(\mu_1;\nu_2)] = -\left(\frac{m}{v}\right)^2\frac{v-1}{v^s-1}. \tag{B9}$$

(c) *No shared features.* Finally, by multiplying both sides of $\sum_{\mu_1} N(\mu_1;\mu_2)=m$ with $N(\nu_1;\nu_2)$ and averaging, we get

$$c_g := \mathrm{Cov}[N_i(\mu_1;\mu_2), N_i(\nu_1;\nu_2)] \\ = -\frac{\mathrm{Cov}[N_i(\mu_1;\mu_2), N_i(\mu_1;\nu_2)]}{v-1} \\ = \left(\frac{m}{v}\right)^2\frac{1}{v^s-1}. \tag{B10}$$

APPENDIX C: EMERGENCE OF INPUT-OUT CORRELATIONS (P_c)

As discussed in the main text, the random hierarchy model presents a characteristic sample size P_c corresponding to the emergence of the input-output correlations. This sample size predicts the sample complexity of deep CNNs, as we also discuss in the main text. In this appendix, we prove that

$$P_c \xrightarrow{n_c, m\to\infty} n_c m^L. \tag{C1}$$

1. Estimating the signal

The correlations between input features and the class label can be quantified via the conditional probability (over realizations of the RHM) of a data point belonging to class α conditioned on displaying the s-tuple $\boldsymbol{\mu}$ in the jth input patch,

$$f_j(\alpha|\boldsymbol{\mu}) \coloneqq \Pr\{\boldsymbol{x}\in\alpha|\boldsymbol{x}_j=\boldsymbol{\mu}\}, \tag{C2}$$

where the notation $\boldsymbol{x}_j=\boldsymbol{\mu}$ means that the elements of the patch $\boldsymbol{x}_j$ encode the tuple of features $\boldsymbol{\mu}$. We say that the low-level features are correlated with the output if

$$f_j(\alpha|\boldsymbol{\mu}) \neq \frac{1}{n_c}, \tag{C3}$$

and define a "signal" as the difference $f_j(\alpha|\boldsymbol{\mu}) - n_c^{-1}$. In the following, we compute the statistics of the signal over realizations of the RHM.

a. Occurrence of low-level features

Let us begin by defining the joint occurrences of a class label α and a low-level feature μ_1 in a given position of the input. Using the tree representation of the model, we will identify an input position with a set of L indices $i_\ell = 1,\dots,s$, each indicating which branch to follow when descending from the root (class label) to a given leaf (low-level feature). These joint occurrences can be computed by combining the occurrences of the single rules introduced in Appendix B. With $L=2$, for instance,

$$N^{(1\to 2)}_{i_1 i_2}(\mu_1;\alpha) = \sum_{\mu_2=1}^{v}\left(m^{s-1}N^{(1)}_{i_1}(\mu_1;\mu_2)\right)\times N^{(2)}_{i_2}(\mu_2;\alpha), \tag{C4}$$

where

(i) $N^{(2)}_{i_2}(\mu_2;\alpha)$ counts the occurrences of μ_2 in position i_2 of the level-2 representations of α, i.e., the s-tuples generated from α according to the second-layer composition rule;

(ii) $N^{(1)}_{i_1}(\mu_1;\mu_2)$ counts the occurrences of μ_1 in position i_1 of the level-1 representations of μ_2, i.e., s-tuples generated by μ_2 according to the composition rule of the first layer;

(iii) the factor m^{s-1} counts the descendants of the remaining $s-1$ elements of the level-2 representation (m descendants per element);

(iv) the sum over μ_2 counts all the possible paths of features that lead to μ_1 from α across 2 generations.

The generalization of Eq. (C4) is immediate once one takes into account that the multiplicity factor accounting for the descendants of the remaining positions at the ℓth generation is equal to $m^{s^{\ell-1}}/m$ ($s^{\ell-1}$ is the size of the representation at the previous level). Hence, the overall multiplicity factor after L generations is

$$1\times\frac{m^s}{m}\times\frac{m^{s^2}}{m}\times\cdots\times\frac{m^{s^{L-1}}}{m} = m^{(s^L-1)/(s-1)-L}, \tag{C5}$$

so that the number of occurrences of feature μ_1 in position $i_1\dots i_L$ of the inputs belonging to class α is

$$N^{(1\to L)}_{i_{1\to L}}(\mu_1;\alpha) = m^{(s^L-1)/(s-1)-L}\sum_{\mu_2,\dots,\mu_L=1}^{v} N^{(1)}_{i_1}(\mu_1;\mu_2)\times\cdots\times N^{(L)}_{i_L}(\mu_L;\alpha), \tag{C6}$$

where we used $i_{1\to L}$ as a shorthand notation for the tuple of indices $i_1, i_2, \dots, i_L$.

The same construction allows us to compute the number of occurrences of up to $s-1$ features within the s-dimensional patch of the input corresponding to the path $i_{2\to L}$. The number of occurrences of a whole s-tuple, instead, follows a slightly different rule, since there is only one level-2 feature μ_2 which generates the whole s-tuple of level-1 features $\boldsymbol{\mu}_1 = (\mu_{1,1},\dots,\mu_{1,s})$—we call this feature $g_1(\boldsymbol{\mu}_1)$, with g_1 denoting the first-layer composition rule. As a result, the sum over μ_2 in the right-hand side of Eq. (C6) disappears and we are left with

$$N^{(1\to L)}_{i_{2\to L}}(\boldsymbol{\mu}_1;\alpha) = m^{(s^L-1)/(s-1)-L} \times \sum_{\mu_3,\dots,\mu_L=1}^{v} N^{(2)}_{i_2}[g_1(\boldsymbol{\mu}_1);\mu_3]\times\cdots\times N^{(L)}_{i_L}(\mu_L;\alpha). \tag{C7}$$

Coincidentally, Eq. (C7) shows that the joint occurrences of an s-tuple of low-level features $\boldsymbol{\mu}_1$ depend on the level-2 feature corresponding to $\boldsymbol{\mu}_1$. Hence, $N^{(1\to L)}_{i_{2\to L}}(\boldsymbol{\mu}_1;\alpha)$ is invariant for the exchange of $\boldsymbol{\mu}_1$ with one of its synonyms, i.e., level-1 tuples $\boldsymbol{\nu}_1$ corresponding to the same level-2 feature.

b. Class probability conditioned on low-level observations

We can turn these numbers into probabilities by normalizing them appropriately. Upon dividing by the total occurrences of a low-level feature μ_1 independently of the class, for instance, we obtain the conditional probability of the class of a given input, conditioned on the feature in position $i_1\dots i_L$ being μ_1:

$$f^{(1\to L)}_{i_{1\to L}}(\alpha|\mu_1) \coloneqq \frac{N^{(1\to L)}_{i_{1\to L}}(\mu_1;\alpha)}{\sum_{\alpha'=1}^{n_c} N^{(1\to L)}_{i_{1\to L}}(\mu_1;\alpha')} = \frac{\sum_{\mu_2,\dots,\mu_L=1}^{v} N^{(1)}_{i_1}(\mu_1;\mu_2)\times\cdots\times N^{(L)}_{i_L}(\mu_L;\alpha)}{\sum_{\mu_2,\dots,\mu_L=1}^{v}\sum_{\mu_{L+1}=1}^{n_c} N^{(1)}_{i_1}(\mu_1;\mu_2)\times\cdots\times N^{(L)}_{i_L}(\mu_L;\mu_{L+1})}. \tag{C8}$$

Let us also introduce, for convenience, the numerator and denominator of the right-hand side of Eq. (C8):

$$\begin{aligned} U^{(1\to L)}_{i_{1\to L}}(\mu_1\alpha) &= \sum_{\mu_2,\ldots,\mu_L=1}^{v} N^{(1)}_{i_1}(\mu_1;\mu_2)\times\cdots\times N^{(L)}_{i_L}(\mu_L;\alpha);\\ D^{(1\to L)}_{i_{1\to L}}(\mu_1) &= \sum_{\alpha=1}^{n_c} U^{(1\to L)}_{i_{1\to L}}(\mu_1;\alpha). \end{aligned} \tag{C9}$$

c. Statistics of the numerator *U*

We now determine the first and second moments of the numerator of $f^{(1\to L)}_{i_{1\to L}}(\mu_1;\alpha)$. Let us first recall the definition for clarity:

$$U^{(1\to L)}_{i_{1\to L}}(\mu_1;\alpha) = \sum_{\mu_2,\ldots,\mu_L=1}^{v} N^{(1)}_{i_1}(\mu_1;\mu_2)\times\cdots\times N^{(L)}_{i_L}(\mu_L;\alpha). \tag{C10}$$

(a) *Level 1:* $L=1$. For $L=1$, U is simply the occurrence of a single production rule $N_i(\mu_1;\alpha)$,

$$\left\langle U^{(1)}\right\rangle = \frac{m}{v}, \tag{C11}$$

$$\sigma^2_{U^{(1)}} \coloneqq \mathrm{Var}\left[U^{(1)}\right] = \frac{m}{v}\frac{v-1}{v}\frac{v^s-m}{v^s-1}\xrightarrow{v\gg1}\frac{m}{v}, \tag{C12}$$

$$c_{U^{(1)}} \coloneqq \mathrm{Cov}\left[U^{(1)}(\mu_1;\alpha),U^{(1)}(\nu_1;\alpha)\right] = -\frac{\mathrm{Var}[U^{(1)}]}{(v-1)} = -\left(\frac{m}{v}\right)^2\frac{v^s-m}{v^s-1}\frac{1}{m}\xrightarrow{v\gg1}\left(\frac{m}{v}\right)^2\frac{1}{m}, \tag{C13}$$

where the relationship between variance and covariance is due to the constraint on the sum of $U^{(1)}$ over μ_1; see Eq. (B6).

(b) *Level 2:* $L=2$. For $L=2$,

$$U^{(1\to 2)}_{i_{1\to 2}}(\mu_1;\alpha) = \sum_{\mu_2=1}^{v} N^{(1)}_{i_1}(\mu_1;\mu_2)\times N^{(2)}_{i_2}(\mu_2;\alpha) = \sum_{\mu_2=1}^{v} N^{(1)}_{i_1}(\mu_1;\mu_2)U^{(2)}_{i_2}(\mu_2;\alpha). \tag{C14}$$

Therefore,

$$\left\langle U^{(1\to 2)}\right\rangle = v\left(\frac{m}{v}\right)\times\left\langle U^{(1)}\right\rangle = v\left(\frac{m}{v}\right)^2, \tag{C15}$$

$$\begin{aligned} \sigma^2_{U^{(2)}} \coloneqq \mathrm{Var}\left[U^{(1\to 2)}\right] &= \sum_{\mu_2,\nu_2=1}^{v}\left(\left\langle N^{(1)}(\mu_1;\mu_2)N^{(1)}(\mu_1;\nu_2)\right\rangle\left\langle U^{(2)}(\mu_2;\alpha)U^{(2)}(\nu_2;\alpha)\right\rangle - \langle N\rangle^2\left\langle U^{(1)}\right\rangle^2\right)\\ &= \sum_{\mu_2,\nu_2=\mu_2}\cdots + \sum_{\mu_2}\sum_{\nu_2\neq\mu_2}\cdots\\ &= v\left(\sigma_N^2\sigma^2_{U^{(1)}} + \sigma_N^2\left\langle U^{(1)}\right\rangle^2 + \sigma^2_{U^{(1)}}\langle N\rangle^2\right) + v(v-1)\left(c_{if}c_{U^{(1)}} + c_{if}\left\langle U^{(1)}\right\rangle^2 + c_{U^{(1)}}\langle N\rangle^2\right)\\ &= v(\sigma_N^2\sigma^2_{U^{(1)}} + (v-1)c_{if}c_{U^{(1)}}) + v\left\langle U^{(1)}\right\rangle^2(\sigma_N^2 + (v-1)c_{if}) + v\langle N\rangle^2(\sigma^2_{U^{(1)}} + (v-1)c_{U^{(1)}})\\ &= v\sigma^2_{U^{(1)}}(\sigma_N^2 - c_{if}) + v\left\langle U^{(1)}\right\rangle^2(\sigma_N^2 + (v-1)c_{if}), \end{aligned} \tag{C16}$$

$$c_{U^{(2)}} = -\frac{\sigma^2_{U^{(2)}}}{(v-1)}. \tag{C17}$$

(c) *Level L.* In general,

$$U^{(1\to L)}_{i_{1\to L}}(\mu_1;\alpha) = \sum_{\mu_2=1}^{v} N^{(1)}_{i_1}(\mu_1;\mu_2)U^{(2\to L)}_{i_{2\to L}}(\mu_2;\alpha). \tag{C18}$$

Therefore,

$$\left\langle U^{(L)}\right\rangle = v\left(\frac{m}{v}\right)\times\left\langle U^{(L-1)}\right\rangle = v^{L-1}\left(\frac{m}{v}\right)^{L}, \tag{C19}$$

$$\begin{aligned}\sigma^2_{U^{(L)}} &= \sum_{\mu_2,\nu_1=1}^{v}\left(\left\langle N^{(1)}(\mu_1;\mu_2)N^{(1)}(\mu_1;\nu_2)\right\rangle\left\langle U^{(2\to L)}(\mu_2;\alpha)U^{(2\to L)}(\nu_1;\alpha)\right\rangle - \langle N\rangle^2\left\langle U^{(1\to(L-1))}\right\rangle^2\right)\\ &= \sum_{\mu_2,\nu_2=\mu_2}\cdots+\sum_{\mu_2}\sum_{\nu_2\neq\mu_2}\cdots\\ &= v\left(\sigma^2_N\sigma^2_{U^{(L-1)}}+\sigma^2_N\left\langle U^{(L-1)}\right\rangle^2+\sigma^2_{U^{(L-1)}}\langle N\rangle^2\right)+v(v-1)\left(\sigma^2_{if}c_{U^{(L-1)}}+c_{if}\left\langle U^{(L-1)}\right\rangle^2+c_{U^{(L-1)}}\langle N\rangle^2\right)\\ &= v\sigma^2_{U^{(L-1)}}(\sigma^2_N-c_{if})+v\left\langle U^{(L-1)}\right\rangle^2(\sigma^2_N+(v-1)c_{if}),\end{aligned} \tag{C20}$$

$$c_{U^{(L)}} = -\frac{\sigma^2_{U^{(L)}}}{(v-1)}. \tag{C21}$$

(d) *Concentration for large m.* In the large multiplicity limit $m \gg 1$, the U's concentrate around their mean value. Because of $m \leq v^{s-1}$, large m implies large v; thus, we can proceed by setting $m = qv^{s-1}$, with $q \in (0,1]$ and studying the $v \gg 1$ limit. From Eq. (C19),

$$\left\langle U^{(L)}\right\rangle = q^L v^{L(s-1)-1}. \tag{C22}$$

In addition,

$$\sigma^2_N \xrightarrow{v\gg1} \frac{m}{v} = qv^{(s-1)-1}, \qquad c_{if}\xrightarrow{v\gg1} -\left(\frac{m}{v}\right)^2\frac{1}{v^{s-1}} = -q^2v^{(s-1)-2}, \tag{C23}$$

so that

$$\begin{aligned}\sigma^2_{U^{(L)}} &= v\sigma^2_{U^{(L-1)}}(\sigma^2_N-\sigma^2_{if})+v\left\langle U^{(L-1)}\right\rangle^2(\sigma^2_N+(v-1)\sigma^2_{if})\\ &\xrightarrow{v\gg1}\sigma^2_{U^{(L-1)}}qv^{(s-1)}+\sigma^2_{U^{(L-1)}}q^2v^{(s-1)-1}+q^{2L-1}(1-q)v^{(2L-1)(s-1)-2}.\end{aligned} \tag{C24}$$

The second of the three terms is always subleading with respect to the first, so we can discard it for now. It remains to compare the first and the third terms. For $L = 2$, since $\sigma^2_{U^{(1)}} = \sigma^2_N$, the first term depends on v as $v^{2(s-1)-1}$, whereas the third is proportional to $v^{3(s-1)-2}$. For $L \geq 3$, the dominant scaling is that of the third term only: for $L = 3$ it can be shown by simply plugging the $L = 2$ result into the recursion, and for larger L it follows from the fact that replacing $\sigma^2_{U^{(L-1)}}$ in the first term with the third term of the precious step always yields a subdominant contribution. Therefore,

$$\sigma^2_{U^{(L)}}\xrightarrow{v\gg1}\begin{cases} q^2v^{2(s-1)-1}+q^3(1-q)v^{3(s-1)-2} & \text{for } L=2\\ q^{2L-1}(1-q)v^{(2L-1)(s-1)-2} & \text{for } L\geq 3.\end{cases} \tag{C25}$$

Upon dividing the variance by the squared mean, we get

$$\frac{\sigma^2_{U^{(L)}}}{\langle U^{(L)}\rangle^2} \xrightarrow{v\gg 1} \begin{cases} \frac{1}{q^2}\frac{1}{v^{2(s-1)-1}} + \frac{1-q}{q}\frac{1}{v^{(s-1)}} & \text{for } L=2 \\ \frac{1-q}{q}\frac{1}{v^{(s-1)}} & \text{for } L\geq 3, \end{cases} \tag{C26}$$

whose convergence to 0 guarantees the concentration of the U's around the average over all instances of the RHM.

d. Statistics of the denominator D

Here we compute the first and second moments of the denominator of $f^{(1\to L)}_{i_{1\to L}}(\mu_1;\alpha)$:

$$D^{(1\to L)}_{i_{1\to L}}(\mu_1) = \sum_{\mu_2,\dots,\mu_L=1}^{v}\sum_{\mu_{L+1}=1}^{n_c} N^{(1)}_{i_1}(\mu_1;\mu_2)\times\cdots\times N^{(L)}_{i_L}(\mu_L;\mu_{L+1}). \tag{C27}$$

(a) *Level 1:* $L=1$. For $L=1$, D is simply the sum over classes of the occurrences of a single production rule, $D^{(1)} = \sum_\alpha N_i(\mu_1;\alpha)$,

$$\langle D^{(1)}\rangle = n_c\frac{m}{v}, \tag{C28}$$

$$\sigma^2_{D^{(1)}} \coloneqq \mathrm{Var}\left[D^{(1)}\right] = n_c\sigma_N^2 + n_c(n_c-1)c_{if} = n_c\left(\frac{m}{v}\right)^2\frac{v-1}{v^s-1}\left(\frac{v^s}{m}-n_c\right)$$
$$\xrightarrow{v\gg 1} n_c\left(\frac{m}{v}\right)^2\left(\frac{v}{m}-\frac{n_c}{v^{s-1}}\right), \tag{C29}$$

$$c_{D^{(1)}} \coloneqq \mathrm{Cov}\left[D^{(1)}(\mu_1), D^{(1)}(\nu_0)\right] = -\frac{\mathrm{Var}[D^{(1)}]}{(v-1)} = n_c c_N + n_c(n_c-1)c_g, \tag{C30}$$

where, in the last line, we used the identities $\sigma_N^2 + (v-1)c_N = 0$ from Eq. (B5) and $c_{if} + (v-1)c_g = 0$ from Eq. (B10).

(b) *Level 2:* $L=2$. For $L=2$,

$$D^{(1\to 2)}_{i_{1\to 2}}(\mu_1) = \sum_{\mu_2}^{v}\sum_{\mu_3=1}^{n_c} N^{(1)}_{i_1}(\mu_1;\mu_2)\times N^{(2)}_{i_2}(\mu_2;\mu_3) = \sum_{\mu_2=1}^{v} N^{(1)}_{i_1}(\mu_1;\mu_2)D^{(2)}_{i_2}(\mu_2). \tag{C31}$$

Therefore,

$$\langle D^{(1\to 2)}\rangle = v\left(\frac{m}{v}\right)\times\langle D^{(1)}\rangle = \frac{n_c}{v}m^2, \tag{C32}$$

$$\begin{aligned}
\sigma^2_{D^{(2)}} \coloneqq \mathrm{Var}\left[D^{(1\to 2)}\right] &= \sum_{\mu_2,\nu_1=1}^{v}\left(\langle N^{(1)}(\mu_1;\mu_2)N^{(1)}(\mu_1;\nu_1)\rangle\langle D^{(2)}(\mu_2)D^{(2)}(\nu_1)\rangle - \langle N\rangle^2\langle D^{(1)}\rangle^2\right)\\
&= \sum_{\mu_2,\nu_1=\mu_2}\cdots + \sum_{\mu_2}\sum_{\nu_1\neq\mu_2}\cdots\\
&= v\left(\sigma_N^2\sigma^2_{D^{(1)}} + \sigma_N^2\langle D^{(1)}\rangle^2 + \sigma^2_{D^{(1)}}\langle N\rangle^2\right) + v(v-1)\left(c_{if}c_{D^{(1)}} + c_{if}\langle D^{(1)}\rangle^2 + c_{D^{(1)}}\langle N\rangle^2\right)\\
&= v(\sigma_N^2\sigma^2_{D^{(1)}} + (v-1)c_{if}c_{D^{(1)}}) + v\langle D^{(1)}\rangle^2(\sigma_N^2 + (v-1)c_{if}) + v\langle N\rangle^2(\sigma^2_{D^{(1)}} + (v-1)c_{D^{(1)}})\\
&= v\sigma^2_{D^{(1)}}(\sigma_N^2 - c_{if}) + v\langle D^{(1)}\rangle^2(\sigma_N^2 + (v-1)c_{if}),
\end{aligned} \tag{C33}$$

$$c_{D^{(2)}} = -\frac{\sigma^2_{D^{(2)}}}{(v-1)}. \tag{C34}$$

(c) *Level L.* In general,

$$D^{(1\to L)}_{i_{1\to L}}(\mu_1) = \sum_{\mu_2=1}^{v} N^{(1)}_{i_1}(\mu_1;\mu_2) D^{(2\to L)}_{i_{2\to L}}(\mu_2). \tag{C35}$$

Therefore,

$$\left\langle D^{(L)} \right\rangle = v\left(\frac{m}{v}\right) \times \left\langle D^{(L-1)} \right\rangle = \frac{n_c}{v} m^L, \tag{C36}$$

$$\begin{aligned}
\sigma^2_{D^{(L)}} &= \sum_{\mu_2,\nu_1=1}^{v} \left(\left\langle N^{(1)}(\mu_1;\mu_2) N^{(1)}(\mu_1;\nu_1) \right\rangle \left\langle D^{(2\to L)}(\mu_2;\alpha) D^{(2\to L)}(\nu_1;\alpha) \right\rangle - \langle N\rangle^2 \left\langle D^{(1\to(L-1))} \right\rangle^2 \right) \\
&= \sum_{\mu_2,\nu_1=\mu_2} \cdots + \sum_{\mu_2} \sum_{\nu_1\neq\mu_2} \cdots \\
&= v\left(\sigma^2_N \sigma^2_{D^{(L-1)}} + \sigma^2_N \left\langle D^{(L-1)} \right\rangle^2 + \sigma^2_{D^{(L-1)}} \langle N\rangle^2 \right) + v(v-1)\left(c_{if} c_{D^{(L-1)}} + c_{if} \left\langle D^{(L-1)} \right\rangle^2 + c_{D^{(L-1)}} \left\langle N \right\rangle^2 \right) \\
&= v\sigma^2_{D^{(L-1)}}(\sigma^2_N - c_{if}) + v\left\langle D^{(L-1)} \right\rangle^2 (\sigma^2_N + (v-1)c_{if}),
\end{aligned} \tag{C37}$$

$$c_{D^{(L)}} = -\frac{\sigma^2_{D^{(L)}}}{(v-1)}. \tag{C38}$$

(d) *Concentration for large m.* Since the D's can be expressed as a sum of different U's, their concentration for $m \gg 1$ follows directly from that of the U's.

e. Estimate of the conditional class probability

We can now turn back to the original problem of estimating

$$f^{(1\to L)}_{i_{1\to L}}(\alpha|\mu_1) = \frac{\sum_{\mu_2,\ldots,\mu_L=1}^{v} N^{(1)}_{i_1}(\mu_1;\mu_2) \times \cdots \times N^{(L)}_{i_L}(\mu_L;\alpha)}{\sum_{\mu_2,\ldots,\mu_L=1}^{v} \sum_{\mu_{L+1}=1}^{n_c} N^{(1)}_{i_1}(\mu_1;\mu_2) \times \cdots \times N^{(L)}_{i_L}(\mu_L;\mu_{L+1})} = \frac{U^{(1\to L)}_{i_{1\to L}}(\mu_1;\alpha)}{D^{(1\to L)}_{i_{1\to L}}(\mu_1)}. \tag{C39}$$

Having shown that both numerator and denominator converge to their average for large m, we can expand for small fluctuations around these averages and write

$$f^{(1\to L)}_{i_{1\to L}}(\alpha|\mu_1) = \frac{v^{-1} m^L \left(1 + \frac{U^{(1\to L)}_{i_{1\to L}}(\mu_1;\alpha) - m^L/v}{m^L/v}\right)}{n_c v^{-1} m^L \left(1 + \frac{D^{(1\to L)}_{i_{1\to L}}(\mu_1) - n_c m^L/v}{m^L}\right)} \tag{C40}$$

$$\begin{aligned}
&= \frac{1}{n_c} + \frac{1}{n_c} \frac{U^{(1\to L)}_{i_{1\to L}}(\mu_1;\alpha) - m^L/v}{m^L/v} - \frac{1}{n_c} \frac{D^{(1\to L)}_{i_{1\to L}}(\mu_1) - n_c m^L/v}{m^L/v} \\
&= \frac{1}{n_c} + \frac{v}{n_c m^L} \left(U^{(1\to L)}_{i_{1\to L}}(\mu_1;\alpha) - \frac{1}{n_c} D^{(1\to L)}_{i_{1\to L}}(\mu_1) \right).
\end{aligned} \tag{C41}$$

Since the conditional frequencies average to n_c^{-1}, the term in brackets averages to zero. We can then estimate the size of the fluctuations of the conditional frequencies (i.e., the signal) with the standard deviation of the term in brackets.

It is important to notice that, for each L and position $i_{1\to L}$, D is the sum over α of U, and the U with different α at fixed low-level feature μ_1 are identically distributed. In general, for a sequence of identically distributed variables $(X_\alpha)_{\alpha=1,\ldots,n_c}$,

$$\left\langle \left(\frac{1}{n_c} \sum_{\beta=1}^{v} X_\beta \right)^2 \right\rangle = \frac{1}{n_c^2} \sum_{\beta=1}^{n_c} \left(\langle X_\beta \rangle^2 + \sum_{\beta'\neq\beta} \langle X_\beta X_{\beta'} \rangle \right) = \frac{1}{n_c} \left(\langle X_\beta \rangle^2 + \sum_{\beta'\neq\beta} \langle X_\beta X_{\beta'} \rangle \right). \tag{C42}$$

Hence,

$$\left\langle\left(X_\alpha - \frac{1}{n_c}\sum_{\beta=1}^{n_c} X_\beta\right)^2\right\rangle = \langle X_\alpha^2\rangle + n_c^{-2}\sum_{\beta,\gamma=1}^{n_c}\langle X_\beta X_\gamma\rangle - 2n_c^{-1}\sum_{\beta=1}^{n_c}\langle X_\alpha X_\beta\rangle$$
$$= \langle X_\alpha^2\rangle - n_c^{-1}\left(\langle X_\alpha\rangle^2 + \sum_{\beta\neq\alpha}\langle X_\alpha X_\beta\rangle\right)$$
$$= \langle X_\alpha^2\rangle - n_c^{-2}\left\langle\left(\sum_{\beta=1}^{n_c} X_\beta\right)^2\right\rangle. \tag{C43}$$

In our case,

$$\left\langle\left(U_{i_{1\to L}}^{(1\to L)}(\mu_1;\alpha) - \frac{1}{n_c}D_{i_{1\to L}}^{(1\to L)}(\mu_1)\right)^2\right\rangle = \left\langle\left(U_{i_{1\to L}}^{(1\to L)}(\mu_1;\alpha)\right)^2\right\rangle - n_c{}^{-2}\left\langle\left(D_{i_{1\to L}}^{(1\to L)}(\mu_1)\right)^2\right\rangle$$
$$= \sigma^2_{U^{(L)}} - n_c^{-2}\sigma^2_{D^{(L)}}, \tag{C44}$$

where, in the second line, we have used that $\langle U^{(L)}\rangle = \langle D^{(L)}\rangle/n_c$ to convert the difference of second moments into a difference of variances. By Eqs. (C19) and (C36),

$$\sigma^2_{U^{(L)}} - n_c^{-2}\sigma^2_{D^{(L)}} = v\sigma^2_{U^{(L-1)}}(\sigma_N^2 - \sigma_{if}^2) + v\langle U^{(L-1)}\rangle^2(\sigma_N^2 + (v-1)\sigma_{if}^2)$$
$$- \frac{v}{n_c^2}\sigma^2_{D^{(L-1)}}(\sigma_N^2 - \sigma_{if}^2) - \frac{v}{n_c^2}\langle D^{(L-1)}\rangle^2(\sigma_N^2 + (v-1)\sigma_{if}^2)$$
$$= v(\sigma_N^2 - \sigma_{if}^2)(\sigma^2_{U^{(L-1)}} - n_c^{-2}\sigma^2_{D^{(L-1)}}), \tag{C45}$$

having used again that $\langle U^{(L)}\rangle = \langle D^{(L)}\rangle/n_c$. Iterating,

$$\sigma^2_{U^{(L)}} - n_c^{-2}\sigma^2_{D^{(L)}} = [v(\sigma_N^2 - \sigma_{if}^2)]^{L-1}\left((\sigma^2_{U^{(1)}} - n_c^{-2}\sigma^2_{D^{(1)}})\right). \tag{C46}$$

Since

$$\sigma^2_{U^{(1)}} = \frac{m}{v}\frac{v-1}{v}\frac{v^s-m}{v^s-1}\xrightarrow{v\gg1}\frac{m}{v},$$
$$n_c^{-2}\sigma^2_{D^{(1)}} = n_c^{-1}\sigma_N^2 + n_c^{-1}(n_c-1)\sigma_{if}^2\xrightarrow{v\gg1}n_c^{-1}\left(\frac{m}{v}\right)^2\left(\frac{v}{m} - \frac{n_c}{v^{s-1}}\right) = \frac{1}{n_c}\frac{m}{v}\left(1 - \frac{mn_c}{v^s}\right), \tag{C47}$$

one has

$$\sigma^2_{U^{(L)}} - n_c^{-2}\sigma^2_{D^{(L)}}\xrightarrow{v\gg1}\frac{m^L}{v}\left(1 - \frac{1 - n_c m/v^s}{n_c}\right), \tag{C48}$$

so that

$$\mathrm{Var}\left[f_{i_{1\to L}}^{(1\to L)}(\alpha|\mu_1)\right] = v^2\frac{\left\langle\left(U_{i_{1\to L}}^{(1\to L)}(\mu_1;\alpha) - \frac{1}{n_c}D_{i_{1\to L}}^{(1\to L)}(\mu_1)\right)^2\right\rangle}{n_c^2 m^{2L}}\xrightarrow{v,n_c\gg1}\frac{v}{n_c}\frac{1}{n_c m^L}. \tag{C49}$$

2. Introducing sampling noise due to the finite training set

In a supervised learning setting where only P of the total data are available, the occurrences N are replaced with their empirical counterparts $\hat{N}$. In particular, the empirical joint occurrence $\hat{N}(\mu;\alpha)$ (where we dropped level and positional indices to ease notation) coincides with the number of successes when sampling P points without replacement from a population of $P_{\max}$ where only $N(\mu;\alpha)$ belong to class α and display feature μ in position j. Thus, $\hat{N}(\mu;\alpha)$ obeys a hypergeometric distribution where P plays the role of the number of trials, $P_{\max}$ the population size, and the true occurrence

$N(\mu;\alpha)$ the number of favorable cases. If P is large and $P_{\max}$, $N(\mu;\alpha)$ are both larger than P, then

$$\hat{N}(\mu;\alpha) \to \mathcal{N}\left[P\frac{N(\mu;\alpha)}{P_{\max}}, P\frac{N(\mu;\alpha)}{P_{\max}}\left(1-\frac{N(\mu;\alpha)}{P_{\max}}\right)\right], \qquad \text{(C50)}$$

where the convergence is meant as a convergence in probability and $\mathcal{N}(a,b)$ denotes a Gaussian distribution with mean a and variance b. The statement above holds when the ratio $N(\mu;\alpha)/P_{\max}$ is away from 0 and 1, which is true with probability 1 for large v due to the concentration of $f(\alpha|\mu)$. In complete analogy, the empirical occurrence $\hat{N}(\mu)$ obeys

$$\hat{N}(\mu) \to \mathcal{N}\left[P\frac{N(\mu)}{P_{\max}}, P\frac{N(\mu)}{P_{\max}}\left(1-\frac{N(\mu)}{P_{\max}}\right)\right]. \qquad \text{(C51)}$$

We obtain the empirical conditional frequency by the ratio of Eqs. (C50) and (C51). Since $N(\mu) = P_{\max}/v$ and $f(\alpha|\mu) = N(\mu;\alpha)/N(\mu)$, we have

$$\hat{f}(\alpha|\mu) = \frac{\frac{f(\alpha|\mu)}{v} + \xi_P\sqrt{\frac{1}{P}\frac{f(\alpha|\mu)}{v}\left(1-\frac{f(\alpha|\mu)}{v}\right)}}{\frac{1}{v} + \zeta_P\sqrt{\frac{1}{P}\frac{1}{v}\left(1-\frac{1}{v}\right)}}, \qquad \text{(C52)}$$

where ξ_P and ζ_P are correlated zero-mean and unit-variance Gaussian random variables over independent drawings of the P training points. By expanding the denominator of the right-hand side for large P we get, after some algebra,

$$\hat{f}(\alpha|\mu) \simeq f(\alpha|\mu) + \xi_P\sqrt{\frac{vf(\alpha|\mu)}{P}\left(1-\frac{f(\alpha|\mu)}{v}\right)} - \zeta_P f(\alpha|\mu)\sqrt{\frac{v}{P}\left(1-\frac{1}{v}\right)}. \qquad \text{(C53)}$$

Recall that, in the limit of large n_c and m, $f(\alpha|\mu) = n_c^{-1}(1+\sigma_f\xi_{\text{RHM}})$, where ξ_{RHM} is a zero-mean and unit-variance Gaussian variable over the realizations of the RHM, while σ_f is the signal, $\sigma_f^2 = v/m^L$ by Eq. (C49). As a result,

$$\hat{f}(\alpha|\mu) \xrightarrow{n_c,m,P\gg 1} \frac{1}{n_c}\left(1 + \sqrt{\frac{v}{m^L}}\xi_{\text{RHM}} + \sqrt{\frac{vn_c}{P}}\xi_P\right). \qquad \text{(C54)}$$

3. Sample complexity

From Eq. (C54) it is clear that for the signal $\hat{f}$, the fluctuations due to noise must be smaller than those due to the random choice of the composition rules. Therefore, the crossover takes place when the two noise terms have the same size, occurring at $P = P_c$ such that

$$\sqrt{\frac{v}{m^L}} = \sqrt{\frac{vn_c}{P_c}} \Rightarrow P_c = n_c m^L. \qquad \text{(C55)}$$

APPENDIX D: IMPROVED SAMPLE COMPLEXITY VIA CLUSTERING

In this appendix, we consider the maximal dataset case $n_c = v$ and $m = v^{s-1}$, and show that a distance-based clustering method acting on the hidden representations of Eq. (13) would identify synonyms at $P \simeq \sqrt{n_c}m^L$. Let us then imagine feeding the representations updates $\Delta f_h(\boldsymbol{\mu})$ of Eq. (13) to a clustering algorithm aimed at identifying synonyms. This algorithm is based on the distance between the representations of different tuples of input features $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$,

$$\|\Delta f(\boldsymbol{\mu}) - \Delta f(\boldsymbol{\nu})\|^2 := \frac{1}{H}\sum_{h=1}^{H}(\Delta f_h(\boldsymbol{\mu}) - \Delta f_h(\boldsymbol{\nu}))^2, \qquad \text{(D1)}$$

where H is the number of hidden neurons. By defining

$$\hat{g}_\alpha(\boldsymbol{\mu}) := \frac{\hat{N}_1(\boldsymbol{\mu};\alpha)}{P} - \frac{1}{n_c}\frac{\hat{N}_1(\boldsymbol{\mu})}{P}, \qquad \text{(D2)}$$

and denoting with $\hat{\boldsymbol{g}}(\boldsymbol{\mu})$ the n_c-dimensional sequence having the $\hat{g}_\alpha$'s as components, we have

$$\begin{aligned}\|\Delta f(\boldsymbol{\mu}) - \Delta f(\boldsymbol{\nu})\|^2 &= \sum_{\alpha,\beta=1}^{n_c}\left(\frac{1}{H}\sum_{h}^{H} a_{h,\alpha}a_{h,\beta}\right)(\hat{g}_\alpha(\boldsymbol{\mu}) - \hat{g}_\alpha(\boldsymbol{\nu}))(\hat{g}_\beta(\boldsymbol{\mu}) - \hat{g}_\beta(\boldsymbol{\nu})) \\ &\xrightarrow{H\to\infty} \sum_{\alpha=1}^{n_c}(\hat{g}_\alpha(\boldsymbol{\mu}) - \hat{g}_\alpha(\boldsymbol{\nu}))^2 = \|\hat{\boldsymbol{g}}(\boldsymbol{\mu}) - \hat{\boldsymbol{g}}(\boldsymbol{\nu})\|^2,\end{aligned} \qquad \text{(D3)}$$

where we used the i.i.d. Gaussian initialization of the readout weights to replace the sum over neurons with $\delta_{\alpha,\beta}$.

Because of the sampling noise, from Eqs. (C50) and (C51), when $1 \ll P \ll P_{\max}$,

$$\hat{g}_\alpha(\boldsymbol{\mu}) = g_\alpha(\boldsymbol{\mu}) + \sqrt{\frac{1}{n_c m v P}}\eta_\alpha(\boldsymbol{\mu}), \qquad \text{(D4)}$$

where $\eta_\alpha(\boldsymbol{\mu})$ is a zero-mean and unit-variance Gaussian noise and g without hat denotes the $P \to P_{\max}$ limit of $\hat{g}$. In the limit $1 \ll P \ll P_{\max}$, the noises with different α and $\boldsymbol{\mu}$ are independent of each other. Thus,

$$\begin{aligned} &\|\hat{\boldsymbol{g}}(\boldsymbol{\mu}) - \hat{\boldsymbol{g}}(\boldsymbol{\nu})\|^2 \\ &\quad = \|\boldsymbol{g}(\boldsymbol{\mu}) - \boldsymbol{g}(\boldsymbol{\nu})\|^2 + \frac{1}{n_c m v P}\|\boldsymbol{\eta}(\boldsymbol{\mu}) - \boldsymbol{\eta}(\boldsymbol{\nu})\|^2 \\ &\qquad + \frac{2}{\sqrt{n_c m v P}}(\boldsymbol{g}(\boldsymbol{\mu}) - \boldsymbol{g}(\boldsymbol{\nu})) \cdot (\boldsymbol{\eta}(\boldsymbol{\mu}) - \boldsymbol{\eta}(\boldsymbol{\nu})). \end{aligned} \tag{D5}$$

If $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$ are synonyms, then $\boldsymbol{g}(\boldsymbol{\mu}) = \boldsymbol{g}(\boldsymbol{\nu})$ and only the noise term contributes to the right-hand side of Eq. (D5). If this noise is sufficiently small, then the distance above can be used to cluster tuples into synonymic groups.

By the independence of the noises and the central limit theorem, for $n_c \gg 1$,

$$\|\boldsymbol{\eta}(\boldsymbol{\mu}) - \boldsymbol{\eta}(\boldsymbol{\nu})\|^2 \sim \mathcal{N}(2n_c, \mathcal{O}(\sqrt{n_c})), \tag{D6}$$

over independent samplings of the P training points. The g's are also random variables over independent realizations of the RHM with zero mean and variance proportional to the variance of the conditional probabilities $f(\alpha|\boldsymbol{\mu})$ [see Eqs. (C40) and (C49)]:

$$\mathrm{Var}[g_\alpha(\boldsymbol{\mu})] = \frac{1}{n_c m v n_c m^L} = \frac{1}{n_c m v P_c}. \tag{D7}$$

To estimate the size of $\|\boldsymbol{g}(\boldsymbol{\mu}) - \boldsymbol{g}(\boldsymbol{\nu})\|^2$ we must take into account the correlations (over RHM realizations) between g's with different class label and tuples. However, in the maximal dataset case $n_c = v$ and $m = v^{s-1}$, both the sum over classes and the sum over tuples of input features of the joint occurrences $N(\boldsymbol{\mu}; \alpha)$ are fixed deterministically. The constraints on the sums allow us to control the correlations between occurrences of the same tuple within different classes and of different tuples within the same class, so that the size of the term $\|\boldsymbol{g}(\boldsymbol{\mu}) - \boldsymbol{g}(\boldsymbol{\nu})\|^2$ for $n_c = v \gg 1$ can be estimated via the central limit theorem:

$$\|\boldsymbol{g}(\boldsymbol{\mu}) - \boldsymbol{g}(\boldsymbol{\nu})\|^2 \sim \mathcal{N}\left(\frac{2n_c}{n_c m v P_c}, \frac{\mathcal{O}(\sqrt{n_c})}{n_c m v P_c}\right). \tag{D8}$$

The mixed term $(\boldsymbol{g}(\boldsymbol{\mu}) - \boldsymbol{g}(\boldsymbol{\nu})) \cdot (\boldsymbol{\eta}(\boldsymbol{\mu}) - \boldsymbol{\eta}(\boldsymbol{\nu}))$ has zero average (both with respect to training set sampling and RHM realizations) and can also be shown to lead to relative fluctuations of order $\mathcal{O}(\sqrt{n_c})$ in the maximal dataset case.

To summarize, we have that, for synonyms,

$$\begin{aligned} \|\hat{\boldsymbol{g}}(\boldsymbol{\mu}) - \hat{\boldsymbol{g}}(\boldsymbol{\nu})\|^2 &= \|\boldsymbol{\eta}(\boldsymbol{\mu}) - \boldsymbol{\eta}(\boldsymbol{\nu})\|^2 \\ &\sim \frac{1}{mvP}\left(1 + \frac{1}{\sqrt{n_c}}\xi_P\right), \end{aligned} \tag{D9}$$

where ξ_P is some $\mathcal{O}(1)$ noise dependent on the training set sampling. If $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$ are not synonyms, instead,

$$\begin{aligned} \|\hat{\boldsymbol{g}}(\boldsymbol{\mu}) - \hat{\boldsymbol{g}}(\boldsymbol{\nu})\|^2 \sim{} & \frac{1}{mvP}\left(1 + \frac{1}{\sqrt{n_c}}\xi_P\right) \\ & + \frac{1}{mvP_c}\left(1 + \frac{1}{\sqrt{n_c}}\xi_{\mathrm{RHM}}\right), \end{aligned} \tag{D10}$$

where ξ_{RHM} is some $\mathcal{O}(1)$ noise dependent on the RHM realization. In this setting, the signal is the deterministic part of the difference between representations of nonsynonymic tuples. Because of the sum over class labels, the signal is scaled up by a factor n_c, whereas the fluctuations (stemming from both sampling and model) are only increased by $\mathcal{O}(\sqrt{n_c})$. Therefore, the signal required for clustering emerges from the sampling noise at $P = P_c/\sqrt{n_c} = \sqrt{n_c} m^L$, equal to $v^{1/2+L(s-1)}$ in the maximal dataset case. This prediction is tested for $s = 2$ in Fig. 10, which shows the error achieved by a layerwise

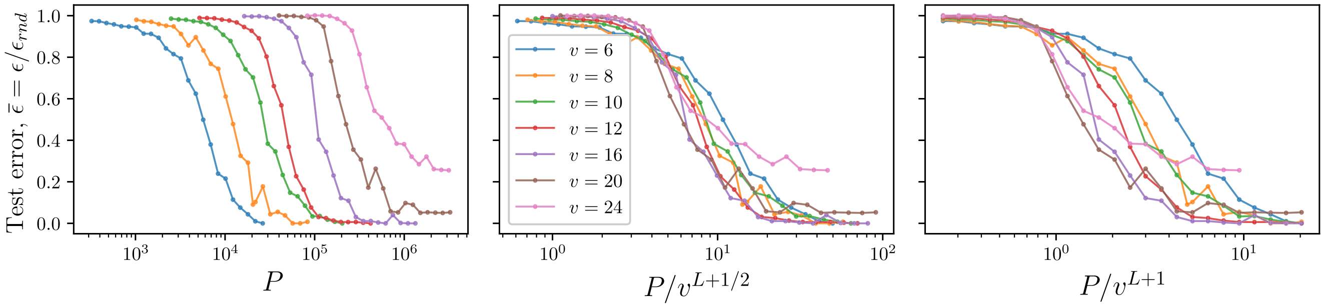

FIG. 10. Sample complexity for layerwise training, $m = n_c = v$, $L = 3$, $s = 2$. Training of an L-layers network is performed layerwise by alternating one-step GD as described in Sec. IV C and clustering of the hidden representations. Clustering of the $mv = v^2$ representations for the different one-hot-encoded input patches is performed with the k-means algorithms. Clustered representations are then orthogonalized and the result is given to the next one-step GD procedure. Left: test error versus number of training points. Different colors correspond to different values of v. Center: collapse of the test error curves when rescaling the x axis by $v^{L+1/2}$. Right: analogous, when rescaling the x axis by v^{L+1}. The curves show a better collapse when rescaling by $v^{L+1/2}$, suggesting that these layerwise algorithms have an advantage of a factor $\sqrt{v}$ over end-to-end training with deep CNNs, for which $P^* = v^{L+1}$.

algorithm which alternates single GD steps to clustering of the resulting representations [22,62]. More specifically, the weights of the first hidden layer are updated with a single GD step while keeping all the other weights frozen. The resulting representations are then clustered, so as to identify groups of synonymic level-1 tuples. The centroids of the ensuing clusters, which correspond to level-2 features, are orthogonalized and used as inputs of another one-step GD protocol, which aims at identifying synonymic tuples of level-2 features. The procedure is iterated L times.

APPENDIX E: INTRINSIC DIMENSIONALITY OF DATA REPRESENTATIONS

In deep learning, the representation of data at each layer of a network can be thought of as lying on a manifold in the layer's activation space. Measures of the *intrinsic dimensionality* of these manifolds can provide insights into how the networks lower the dimensionality of the problem layer by layer. However, such measurements have challenges. One key challenge is that it assumes that real data exist on a smooth manifold, while in practice, the dimensionality is estimated based on a discrete set of points. This leads to counterintuitive results such as an increase in the intrinsic dimensionality with depth, especially near the input. An effect that is impossible for continuous smooth manifolds. We resort to an example to illustrate how this increase with depth can result from spurious effects. Consider a manifold of a given intrinsic dimension that undergoes a transformation where one of the coordinates is multiplied by a large factor. This operation would result in an elongated manifold that appears one dimensional. The measured intrinsic dimensionality would consequently be one, despite the higher dimensionality of the manifold. In the context of neural networks, a network that operates on such an elongated manifold could effectively "reduce" this extra, spurious dimension. This could result in an increase in the observed intrinsic dimensionality as a function of network depth, even though the actual dimensionality of the manifold did not change.

In the specific case of our data, the intrinsic dimensionality of the internal representations of deep CNNs monotonically decreases with depth, see Fig. 11, consistently with the idea proposed in the main text that the CNNs solve the problem by reducing the effective dimensionality of data layer by layer. We attribute this monotonicity to the absence of spurious or noisy directions that might lead to the counterintuitive effect described above.

APPENDIX F: ADDITIONAL RESULTS ON SAMPLE COMPLEXITY

This appendix collects additional results on the sample complexity of deep networks trained on the RHM (Figs. 12 and 13), on the learning curves for "lazy" neural networks (Fig. 14), and for a ResNet18 trained on different subsamples of the benchmark dataset CIFAR10 (Fig. 15).

Figure 12 shows the behavior of the sample complexity with varying number of classes n_c when all the other parameters of the RHM are fixed, confirming the linear scaling discussed in the main text.

Figure 13 shows the behavior of the sample complexity for deep fully connected networks having depth larger than $L+1$, which are not tailored to the structure of the RHM. Notice that changing architecture seems to induce an additional factor of 2^L to the sample complexity, independent of v, n_c, and m. This factor is also polynomial in the input dimension.

Figure 14 presents the learning curves for deep CNNs tailored to the structure of the model and trained in the lazy regime on the maximal case, i.e., $n_c = v$ and $m = v^s$. In particular, we consider the infinite-width limit of CNNs with all layers scaled by a factor $H^{-1/2}$, including the last. In this limit, CNNs become equivalent to a kernel method [49], with an architecture-dependent kernel known as the *neural tangent kernel*. In our experiments, we use the

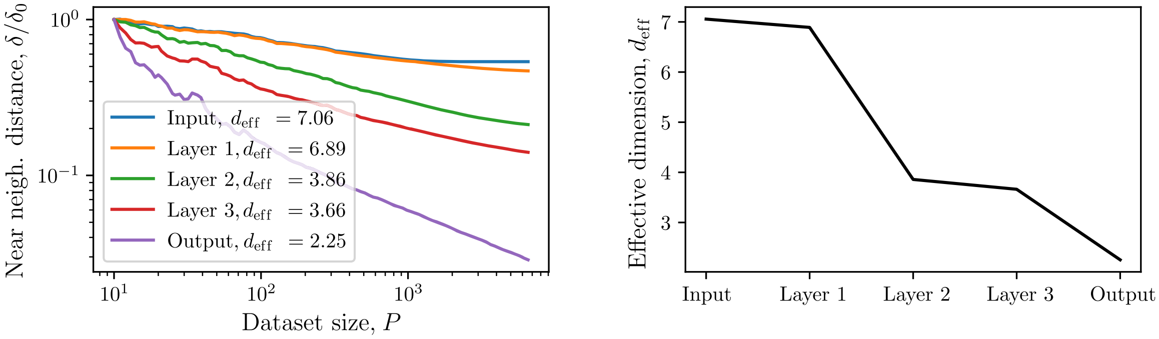

FIG. 11. Effective dimension of the internal representation of a CNN trained on one instance of the RHM with $m = n_c = v, L = 3$ resulting in $P_{\max} = 6232$. Left: average nearest neighbor distance of input or network activations when probing them with a dataset of size P. The value reported on the y axis is normalized by $\delta_0 = \delta(P = 10)$. The slope of $\delta(P)$ is used as an estimate of the effective dimension. Right: effective dimension as a function of depth. We observe a monotonic decrease, consistent with the idea that the dimensionality of the problem is reduced by deep neural networks with depth.

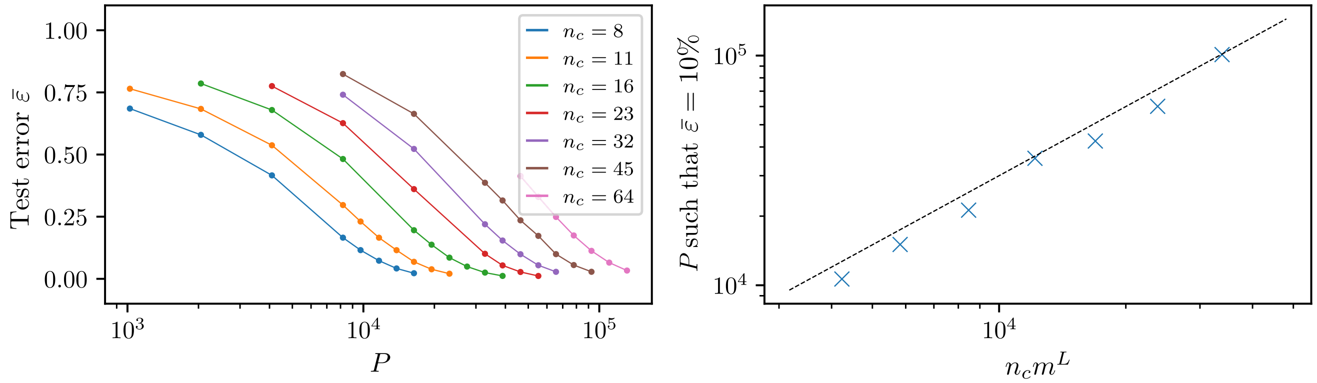

FIG. 12. Sample complexity of deep CNNs, for $L = s = 2$, $v = 256$, $m = 23$ and different values of n_c. Left: test error versus number of training points with the color indicating the number of classes (see key). Right: sample complexity P^* (crosses) and law $P^* = n_c m^L$ (black dashed).

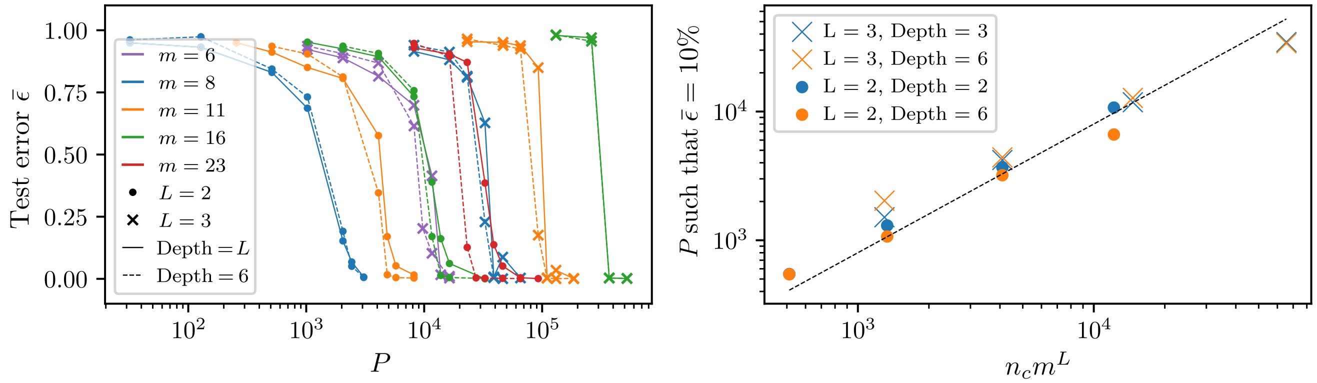

FIG. 13. Sample complexity of deep fully connected networks with different depth, for $s = 2$ and $m = n_c = v$. Left: test error versus number of training points. The color denotes the value of $m = n_c = v$, the marker the hierarchy depth of the RHM L. Solid lines represent networks having depth L, while dashed lines correspond to networks with depth $6 > L$. Note that, in all cases, the behavior of the test error is roughly independent of the network depth. Right: sample complexity P^* (crosses and circles). With respect to the case of deep CNNs tailored to the structure of the RHM, the sample complexity of generic deep networks seems to display an additional factor of s^L independently of n_c, m, and v.

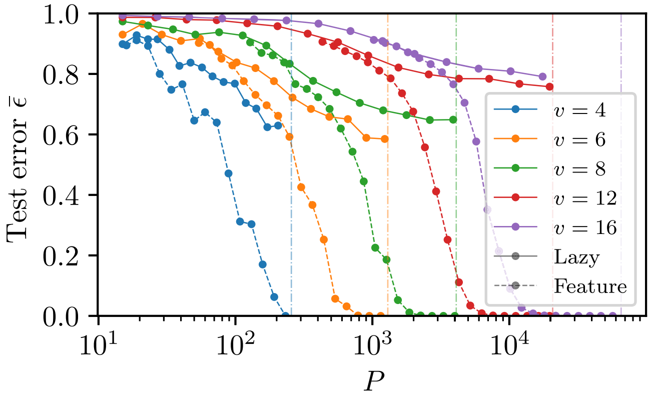

FIG. 14. Learning curves of depth-$(L + 1)$ CNNs, for $L = 2$, $s = 2$ and $m = n_c = v$ trained in the “lazy” regime (full lines)—where they are equivalent to a kernel method [49]—and in the “feature” learning regime (dashed lines). Different colors correspond to different vocabulary sizes v. Vertical lines signal $P_{\max} = v^{s^L}$.

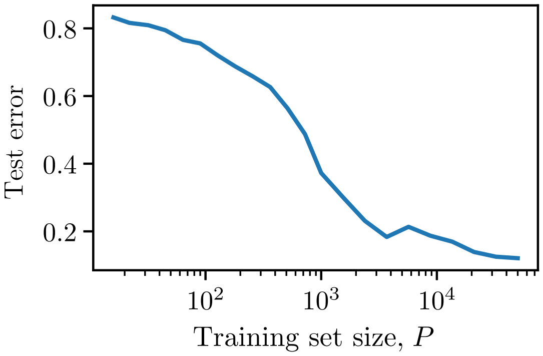

FIG. 15. Test error versus number of training points for a ResNet18 trained on subsamples of the CIFAR10 dataset. Results are the average of 10 jointly different initializations of the networks and dataset sampling.

analytical form of this kernel (see, e.g., Ref. [25]) and train a kernel logistic regression classifier up to convergence. Our main result is that, in the lazy regime, the generalization error stays finite even when $P \approx P_{\max}$; thus, kernels suffer from the curse of dimensionality.

Notice that the learning curves of the lazy regime follow those of the feature learning regime for $P \ll P^*$. This is because the CNN kernel can also exploit local correlations between the label and input patches [25] to improve its performance. However, unlike in the feature regime, kernels cannot build a hierarchical representation, and thus their test error does not converge to zero.

[1] A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis, *Deep learning for computer vision: A brief review*, Comput. Intell. Neurosci. **2018**, 1 (2018).

[2] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton *et al.*, *Mastering the game of Go without human knowledge*, Nature (London) **550**, 354 (2017).

[3] U. v. Luxburg and O. Bousquet, *Distance-based classification with Lipschitz functions*, J. Mach. Learn. Res. **5**, 669 (2004).

[4] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, *ImageNet: A large-scale hierarchical image database*, in *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, New York, 2009), pp. 248–255, https://ieeexplore.ieee.org/document/5206848.

[5] P. Pope, C. Zhu, A. Abdelkader, M. Goldblum, and T. Goldstein, *The intrinsic dimension of images and its impact on learning*, in *Proceedings of the International Conference on Learning Representations* (ICLR, 2021), https://openreview.net/forum?id=XJk19XzGq2J.

[6] Y. LeCun, Y. Bengio, and G. Hinton, *Deep learning*, Nature (London) **521**, 436 (2015).

[7] D. C. Van Essen and J. H. Maunsell, *Hierarchical organization and functional streams in the visual cortex*, Trends Neurosci. **6**, 370 (1983).

[8] K. Grill-Spector and R. Malach, *The human visual cortex*, Annu. Rev. Neurosci. **27**, 649 (2004).

[9] M. D. Zeiler and R. Fergus, *Visualizing and understanding convolutional networks*, in *Proceedings of the 13th European Conference Computer Vision—-ECCV 2014, Zurich* (Springer, Cham, 2014), pp. 818–833, 10.1007/978-3-319-10590-1_53.

[10] D. Doimo, A. Glielmo, A. Ansuini, and A. Laio, *Hierarchical nucleation in deep neural networks*, Adv. Neural Inf. Process. Syst. **33**, 7526 (2020), https://proceedings.neurips.cc/paper/2020/hash/54f3bc04830d762a3b56a789b6ff62df-Abstract.html.

[11] J. Bruna and S. Mallat, *Invariant scattering convolution networks*, IEEE Trans. Pattern Anal. Mach. Intell. **35**, 1872 (2013).

[12] R. Shwartz-Ziv and N. Tishby, *Opening the black box of deep neural networks via information*, arXiv:1703.00810.

[13] A. M. Saxe, Y. Bansal, J. Dapello, M. Advani, A. Kolchinsky, B. D. Tracey, and D. D. Cox, *On the information bottleneck theory of deep learning*, J. Stat. Mech. (2019) 124020.

[14] A. Ansuini, A. Laio, J. H. Macke, and D. Zoccolan, *Intrinsic dimension of data representations in deep neural networks*, Adv. Neural Inf. Process. Syst. **32**, 6111 (2019).

[15] S. Recanatesi, M. Farrell, M. Advani, T. Moore, G. Lajoie, and E. Shea-Brown, *Dimensionality compression and expansion in deep neural networks*, arXiv:1906.00443.

[16] L. Petrini, A. Favero, M. Geiger, and M. Wyart, *Relative stability toward diffeomorphisms indicates performance in deep nets*, Adv. Neural Inf. Process. Syst. **34**, 8727 (2021).

[17] U. M. Tomasini, L. Petrini, F. Cagnetta, and M. Wyart, *How deep convolutional neural networks lose spatial information with training*, Mach. Learn. Sci. Technl. **4**, 045026 (2023).

[18] A. B. Patel, T. Nguyen, and R. G. Baraniuk, *A probabilistic theory of deep learning*, arXiv:1504.00641.

[19] E. Mossel, *Deep learning and hierarchal generative models*, arXiv:18612.09057.

[20] H. Mhaskar, Q. Liao, and T. Poggio, *When and why are deep networks better than shallow ones?*, in *Proceedings of the AAAI Conference on Artificial Intelligence, 2017* (AAAI Press, San Francisco, 2017), Vol. 31, 10.1609/aaai.v31i1.10913.

[21] T. Poggio, H. Mhaskar, L. Rosasco, B. Miranda, and Q. Liao, *Why and when can deep—but not shallow—networks avoid the curse of dimensionality: A review*, Int. J. Autom. Comput. **14**, 503 (2017).

[22] E. Malach and S. Shalev-Shwartz, *A provably correct algorithm for deep learning that actually works*, arXiv:1803.09522.

[23] J. Zazo, B. Tolooshams, D. Ba, and H. J. A. Paulson, *Convolutional dictionary learning in hierarchical networks*, in *Proceedings of the IEEE 8th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing, 2019* (IEEE, New York, 2019), Vol. 131, 10.1109/CAMSAP45676.2019.9022440.

[24] J. Schmidt-Hieber, *Nonparametric regression using deep neural networks with ReLU activation function*, Ann. Stat. **48**, 1875 (2020), https://proceedings.mlr.press/v202/cagnetta23a.html.

[25] F. Cagnetta, A. Favero, and M. Wyart, *What can be learnt with wide convolutional neural networks?*, in *Proceedings of the International Conference on Machine Learning, PMLR, 2023* (PMLR, 2023), pp. 3347–3379.

[26] U. Grenander, *Elements of Pattern Theory* (Johns Hopkins University Press, Baltimore and London, 1996), pp. xiii + 222, 10.1002/bimj.4710390707.

[27] M. Mézard, *Mean-field message-passing equations in the Hopfield model and its generalizations*, Phys. Rev. E **95**, 022117 (2017).

[28] S. Goldt, M. Mézard, F. Krzakala, and L. Zdeborová, *Modeling the influence of data structure on learning in neural networks: The hidden manifold model*, Phys. Rev. X **10**, 041044 (2020).

[29] A. M. Saxe, J. L. McClelland, and S. Ganguli, *A mathematical theory of semantic development in deep neural networks*, Proc. Natl. Acad. Sci. U.S.A. **116**, 11537 (2019).

[30] Y. Bahri, J. Kadmon, J. Pennington, S. S. Schoenholz, J. Sohl-Dickstein, and S. Ganguli, *Statistical mechanics of deep learning*, Annu. Rev. Condens. Matter Phys. **11**, 501 (2020).

[31] A. Ingrosso and S. Goldt, *Data-driven emergence of convolutional structure in neural networks*, Proc. Natl. Acad. Sci. U.S.A. **119**, e2201854119 (2022).

[32] E. DeGiuli, *Random language model*, Phys. Rev. Lett. **122**, 128301 (2019).

[33] F. Bach, *The quest for adaptivity, Machine Learning Research Blog* (2021), https://francisbach.com/quest-for-adaptivity/.

[34] L. Györfi, M. Kohler, A. Krzyzak, H. Walk *et al.*, *A Distribution-Free Theory of Nonparametric Regression*, Vol. 1 (Springer, New York, 2002), 10.1007/b97848.

[35] S. Kpotufe, *k-NN regression adapts to local intrinsic dimension*, Adv. Neural Inf. Process. Syst. **24**, 729 (2011), https://proceedings.neurips.cc/paper_files/paper/2011/file/05f971b5ec196b8c65b75d2ef8267331-Paper.pdf.

[36] T. Hamm and I. Steinwart, *Adaptive learning rates for support vector machines working on data with low intrinsic dimension*, Ann. Stat. **49**, 3153 (2021).

[37] M. Geiger, L. Petrini, and M. Wyart, *Landscape and training regimes in deep learning*, Phys. Rep. **924**, 1 (2021).

[38] J. Paccolat, L. Petrini, M. Geiger, K. Tyloo, and M. Wyart, *Geometric compression of invariant manifolds in neural networks*, J. Stat. Mech. (2021) 044001.

[39] E. Abbe, E. Boix-Adsera, M. S. Brennan, G. Bresler, and D. Nagaraj, *The staircase property: How hierarchical structure can guide deep learning*, in *Advances in Neural Information Processing Systems*, Vol. 34, edited by M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan (Curran Associates, Inc., 2021), pp. 26989–27002, https://proceedings.neurips.cc/paper_files/paper/2021/file/e2db7186375992e729165726762cb4c1-Paper.pdf.

[40] B. Barak, B. Edelman, S. Goel, S. Kakade, E. Malach, and C. Zhang, *Hidden progress in deep learning: SGD learns parities near the computational limit*, Adv. Neural Inf. Process. Syst. **35**, 21750 (2022), https://proceedings.neurips.cc/paper_files/paper/2022/file/884baf65392170763b27c914087bde01-Paper-Conference.pdf.

[41] Y. Dandi, F. Krzakala, B. Loureiro, L. Pesce, and L. Stephan, *Learning two-layer neural networks, one (giant) step at a time*, arXiv:2305.18270.

[42] F. Bach, *Breaking the curse of dimensionality with convex neural networks*, J. Mach. Learn. Res. **18**, 629 (2017), http://jmlr.org/papers/v18/14-546.html.

[43] E. Gardner and B. Derrida, *Three unfinished works on the optimal storage capacity of networks*, J. Phys. A **22**, 1983 (1989).

[44] L. Zdeborová and F. Krzakala, *Statistical physics of inference: Thresholds and algorithms*, Adv. Phys. **65**, 453 (2016).

[45] M. Mézard, *Spin glass theory and its new challenge: Structured disorder*, Indian J. Phys., 1 (2023).

[46] S. Spigler, M. Geiger, and M. Wyart, *Asymptotic learning curves of kernel methods: Empirical data versus teacher–student paradigm*, J. Stat. Mech. (2020) 124001.

[47] A. Favero, F. Cagnetta, and M. Wyart, *Locality defeats the curse of dimensionality in convolutional teacher-student scenarios*, Adv. Neural Inf. Process. Syst. **34**, 9456 (2021), https://proceedings.neurips.cc/paper_files/paper/2021/file/4e8eaf897c638d519710b1691121f8cb-Paper.pdf.

[48] R. Aiudi, R. Pacelli, A. Vezzani, R. Burioni, and P. Rotondo, *Local kernel renormalization as a mechanism for feature learning in overparametrized convolutional neural networks*, arXiv:2307.11807.

[49] A. Jacot, F. Gabriel, and C. Hongler, *Neural tangent kernel: Convergence and generalization in neural networks*, Adv. Neural Inf. Process. Syst. **31**, 8580 (2018), https://proceedings.neurips.cc/paper_files/paper/2018/file/5a4be1fa34e62bb8a6ec6b91d2462f5a-Paper.pdf.

[50] L. Chizat, E. Oyallon, and F. Bach, *On lazy training in differentiable programming*, in Advances in Neural Information Processing Systems (Curran Associates, Inc., 2019), pp. 2937–2947, https://proceedings.neurips.cc/paper_files/paper/2019/file/ae614c557843b1df326cb29c57225459-Paper.pdf.

[51] G. Rozenberg and A. Salomaa, *Handbook of Formal Languages* (Springer Berlin, Heidelberg, 1997), 10.1007/978-3-642-59136-5.

[52] G. Yang and E. J. Hu, *Feature learning in infinite-width neural networks*, arXiv:2011.14522.

[53] Let us focus on the first s-dimensional patch of the input $\boldsymbol{x}_1$, which can take mv distinct values—m for each of the v level-2 features. For a linear transformation, insensitivity is equivalent to the following set of constraints: For each level-2 feature μ, and $\boldsymbol{x}_{1,i}$ encoding for one of the m level-1 representations generated by μ, $\boldsymbol{w} \cdot \boldsymbol{x}_{1,i} = c_\mu$. Since c_μ is an arbitrary constant, there are $v \times (m-1)$ constraints for the $v \times s$ components of $\boldsymbol{w}$, which cannot be satisfied in general unless $m \leq (s+1)$.

[54] The notation $\boldsymbol{x}_j = \boldsymbol{\mu}$ means that the elements of the patch $\boldsymbol{x}_j$ encode the tuple of features $\boldsymbol{\mu}$.

[55] A. Damian, J. Lee, and M. Soltanolkotabi, *Neural networks can learn representations with gradient descent*, in *Proceedings of Thirty-Fifth Conference on Learning Theory, 2022* (PMLR, 2022), Vol. 178, p. 5413, https://proceedings.mlr.press/v178/damian22a.html.

[56] J. Ba, M. A. Erdogdu, T. Suzuki, Z. Wang, D. Wu, and G. Yang, *High-dimensional asymptotics of feature learning: How one gradient step improves the representation*, Adv. Neural Inf. Process. Syst. **35**, 37932 (2022), https://proceedings.neurips.cc/paper_files/paper/2022/file/f7e7fabd73b3df96c54a320862afcb78-Paper-Conference.pdf.

[57] Here invariance to exchange of level-1 synonyms can already be achieved at the first hidden layer due to the orthogonalization of the s-dimensional patches of the input, which makes them linearly separable.

[58] M. Denil, B. Shakibi, L. Dinh, M. A. Ranzato, and N. de Freitas, *Predicting parameters in deep learning*, in *Advances in Neural Information Processing Systems*, Vol. 26 (2013), https://proceedings.neurips.cc/paper_files/paper/2013/file/7fec306d1e665bc9c748b5d2b99a6e97-Paper.pdf.

[59] E. L. Denton, W. Zaremba, J. Bruna, Y. LeCun, and R. Fergus, *Exploiting linear structure within convolutional networks for efficient evaluation*, in *Advances in Neural Information Processing Systems*, Vol. 27 (Curran Associates, Inc., 2014), https://proceedings.neurips.cc/paper_files/paper/2014/file/2afe4567e1bf64d32a5527244d104cea-Paper.pdf.

[60] X. Yu, T. Liu, X. Wang, and D. Tao, *On compressing deep models by low rank and sparse decomposition*, in *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (IEEE, New York, 2017), Vol. 67, https://openaccess.thecvf.com/content_cvpr_2017/html/Yu_On_Compressing_Deep_CVPR_2017_paper.html.

[61] F. Guth, B. Ménard, G. Rochette, and S. Mallat, *A rainbow in deep network black boxes*, arXiv:2305.18512.

[62] E. Malach and S. Shalev-Shwartz, *The implications of local correlation on learning some deep functions*, Adv. Neural Inf. Process. Syst. **33**, 1322 (2020), https://proceedings.neurips.cc/paper_files/paper/2020/file/0e4ceef65add6cf21c0f3f9da53b71c0-Paper.pdf.

[63] S. Shalev-Shwartz, O. Shamir, and S. Shammah, *Failures of gradient-based deep learning*, in *Proceedings of the International Conference on Machine Learning* (PMLR, 2017), pp. 3067–3075, https://proceedings.mlr.press/v70/shalev-shwartz17a.html.

[64] N. Kruger, P. Janssen, S. Kalkan, M. Lappe, A. Leonardis, J. Piater, A. J. Rodriguez-Sanchez, and L. Wiskott, *Deep hierarchies in the primate visual cortex: What can we learn for computer vision?*, IEEE Trans. Pattern Anal. Mach. Intell. **35**, 1847 (2012).

[65] A. Krizhevsky, *Learning multiple layers of features from tiny images* (2009), https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.

[66] L. Petrini and F. Cagnetta, *Random hierarchy model* (2023), 10.5281/zenodo.12509435; https://github.com/pcsl-epfl/hierarchy-learning/blob/master/datasets/hierarchical.py.

[67] https://github.com/pcsl-epfl/ hierarchy-learning/blob/master/models.

[68] https://github.com/pcsl-epfl/hierarchy-learning/blob/master.

[69] G. Yang, *Scaling limits of wide neural networks with weight sharing: Gaussian process behavior, gradient independence, and neural tangent kernel derivation*, arXiv:1902.04760.

[70] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, *PYTORCH: An imperative style, high-performance deep learning library*, Adv. Neural Inf. Process. Syst. **32**, 8026 (2019), https://proceedings.neurips.cc/paper_files/paper/2019/file/bdbca288fee7f92f2bfa9f7012727740-Paper.pdf.