Scaling laws and representation learning in simple hierarchical languages: Transformers versus convolutional architectures

Francesco Cagnetta[*]
Scuola Internazionale Superiore di Studi Avanzati (SISSA), Via Bonomea 265, 34136 Trieste, Italy

Alessandro Favero[†]
Institute of Physics, École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland

Antonio Sclocchi[‡]
Gatsby Computational Neuroscience Unit, University College London, London, United Kingdom

Matthieu Wyart[§,‖]
Department of Physics & Astronomy, Johns Hopkins University, Baltimore, MD, USA

(Received 30 April 2025; accepted 31 October 2025; published 23 December 2025)

How do neural language models acquire a language's structure when trained for next-token prediction? We address this question by deriving theoretical scaling laws for neural network performance on synthetic datasets generated by the random hierarchy model (RHM)—an ensemble of probabilistic context-free grammars designed to capture the hierarchical structure of natural language while remaining analytically tractable. Previously, we developed a theory of representation learning based on data correlations that explains how deep learning models capture the hierarchical structure of the data sequentially, one layer at a time. Here, we extend our theoretical framework to account for architectural differences. In particular, we predict and empirically validate that convolutional networks, whose structure aligns with that of the generative process through locality and weight sharing, enjoy a faster scaling of performance compared to transformer models, which rely on global self-attention mechanisms. This finding clarifies the architectural biases underlying neural scaling laws and highlights how representation learning is shaped by the interaction between model architecture and the statistical properties of data.

DOI: 10.1103/qtd6-nl8p

I. INTRODUCTION

The exceptional success of deep learning methods in vision, language, and other domains has prompted an urgent search for a theory that can explain their effectiveness. Despite the general absence of a fundamental understanding, practitioners have identified several empirical regularities that any such theory should ultimately aim to predict. Among these, the most prominent are neural scaling laws [1–4], which describe power-law relationships between model performance and resource variables such as dataset size, model parameters, and compute budget. On the one hand, these laws have proven remarkably consistent across domains and architectures and now serve as practical heuristics for guiding resource allocation in large-scale machine learning. On the other hand, their ubiquity suggests the presence of fundamental statistical phenomena underlying the behavior of modern deep learning methods—much like the scaling relations observed near critical points in statistical physics [5].

If these statistical phenomena are truly universal, then they must reside in the data itself. After all, a learner can generalize only if the data exhibits patterns that are discoverable from finite examples, i.e., learning is impossible if there is nothing to learn. While early theoretical studies often assumed overly simplistic data models—with inputs being independent or exhibiting shallow statistical dependencies, and tasks corresponding to low-dimensional or extremely smooth functions—these assumptions fail to capture the richness of real data. Natural data such as language and images exhibit structure in the form of long-range dependencies and compositionality, where complex features are constructed from simpler parts according to systematic rules. Recent work has introduced hierarchically compositional models of data that better reflect the kind of structure deep learning methods exploit, while retaining some degree of analytical tractability [6–14]. In this work, we build on this perspective by studying how different architectures learn hierarchically compositional data, revealing the impact of architectural differences on the scaling of performance.

[*]Contact author: francesco.cagnetta@sissa.it
[†]Contact author: alessandro.favero@epfl.ch
[‡]Contact author: a.sclocchi@ucl.ac.uk
[§]Contact author: mwyart1@jh.edu
[‖]On Leave from Institute of Physics, École polytechnique fédérale de Lausanne (EPFL), Lausanne, Switzerland.

Specifically, we focus on synthetic datasets generated by the random hierarchy model (RHM)—an ensemble of hierarchically compositional generative processes corresponding to simple context-free grammars (CFGs) [15], introduced in Ref. [8] to study the role of depth in supervised learning tasks. The learning setup is a last-token prediction task, where a deep neural network is tasked with predicting the final token of a sequence of d, given the first $d-1$ ones. In Ref. [10], we showed that deep networks use correlations between the tokens to infer the hierarchical structure of the data and derived the resulting scaling of the test error with the number of training data. The present paper extends this correlation-based scaling analysis to incorporate architectural differences. In particular, we predict that translational invariant architectures such as convolutional networks have access to stronger correlations and therefore enjoy a faster improvement in performance over training than transformers. We empirically validate these theoretical predictions by examining the learning dynamics of deep convolutional networks and transformers trained with standard gradient-based algorithms on our synthetic dataset. Our analysis reveals how fundamental architectural biases affect neural scaling laws.

The remainder of the paper is organized as follows: In Sec. II, we review related works, including theories of neural scaling laws based on simpler data structures, theories of representation learning based on correlations, and empirical and theoretical works using several classes of formal languages to understand the behavior of neural language models. In Sec. III, we introduce the RHM, an ensemble of synthetic datasets based on probabilistic context-free grammars (PCFGs). The specific assumptions made on the data-generating process render the dataset analytically tractable, enabling the explicit computation of data statistics and their ensemble averages. In Sec. IV, we characterize the fundamental statistical properties of RHM data, focusing on token correlations and their relationship with the hidden hierarchical structure. In Sec. V, we develop a theoretical framework for learning RHM data in the last-token prediction setting. This framework leverages the link between observable correlations and the hidden tree structure, allowing us to compute sample complexities for inferring hidden variables from a finite dataset of example sentences. In Sec. VI, we describe the neural network architectures studied in this work: convolutional networks, which implement local connectivity and weight sharing, and transformers, which use global self-attention mechanisms to capture long-range dependencies. In Sec. VII, we compare our theory with the empirical learning curves of deep neural networks. This is our main result, highlighting systematic differences in the scaling behavior of the test loss between convolutional networks and transformers. In Sec. VIII, we qualitatively validate our theory of representation learning by studying how the internal representations of deep networks evolve during training. In Sec. IX, we extend the analysis of the previous two sections to locally connected networks (LCNs), which retain the local receptive field of convolutional networks but without weight sharing, to isolate the role of architectural priors in learning hierarchical data. Finally, Sec. X summarizes our main finding about the impact on architectural differences on neural scaling law and discusses implications and possible future directions.

II. RELATED WORKS

A. Theories of neural scaling laws

Power-law learning curves can arise in models that memorize training data drawn from a Zipfian (i.e., power-law) distribution [16,17], but such models fail to account for generalization beyond the training set. Alternatively, power laws emerge in kernel methods when the target function has a power-law spectrum in the kernel eigenbasis [18], a mechanism that underpins many theoretical analyses of scaling in neural networks [19–26]. However, these analyses are restricted to kernel or lazy training regimes [27,28] and do not capture the representation learning capabilities of modern deep networks. A number of works have attempted to go beyond this limitation in simplified models [29–33], showing that feature learning can modify scaling exponents and improve generalization.

B. Correlation-based representation learning

Hierarchical generative models on trees, initially developed for phylogeny [34], have become a powerful tool in understanding both supervised [6–9] and self-supervised [8,11,12,14,35] learning algorithms. In supervised settings, [6] introduced a sequential clustering algorithm that highlights the importance of correlations between the input features and the labels for learning. Reference [8] introduced the RHM as a framework to show how correlations emerge from the generative model and can be used by deep networks to represent the latent hierarchical structure. In self-supervised settings, several works [8,11,14] have shown that neural networks progressively learn longer-range correlations, corresponding to deeper layers of the data hierarchy, throughout the course of training. This staged process aligns with the so-called distributional simplicity bias, where neural networks sequentially learn statistics of increasing order during training [36–38].

C. Learning formal languages

There is a growing number of studies using generative models from theoretical linguistics to understand the capabilities of large language models, including n grams [39–41], and regular [42,43] and context-free grammars [44,45]. All these works concern either expressivity or the interpretability of the representations of trained transformers. Reference [45], in particular, showed that the operations performed by BERT-like transformers resemble well-known algorithms for grammatical inference, and proved that, for PCFG data, these algorithms are optimal solutions of the masked language modeling objective. However, when the training data is compatible with both a PCFG and a nonhierarchical generative model, neither recurrent language models [46] nor transformers [47] consistently prefer the hierarchical explanation. In addition, none of these works studied the learning process and the sample complexity.

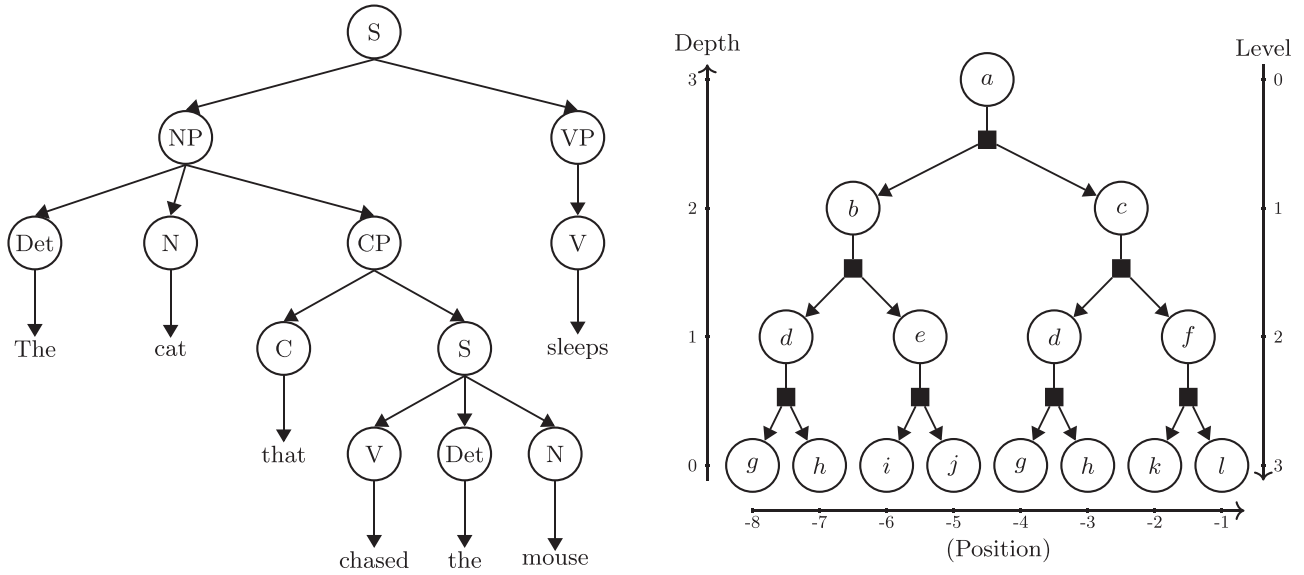

FIG. 1. Comparison of structural hierarchies in (left) natural language and (right) an instance of the random hierarchy model (RHM).

III. CONTEXT-FREE GRAMMARS AND THE RANDOM HIERARCHY MODEL

In this section, we introduce the notion of context-free grammar and our model of hierarchically compositional data.

A. Context-free grammars and compositional structure

Many real-world data modalities—such as natural language, visual scenes, and even code—exhibit a hierarchical and compositional structure. CGFs from theoretical linguistics [15,48] offer a natural formalism to describe such structures [49–52]. A CFG consists of a set of production rules for the recursive expansion of abstract hidden (nonterminal) symbols into sequences of other nonterminal and observable (terminal) symbols. The recursive nature of the rules induces a treelike structure, called derivation—see Fig. 1, left panel, for an illustration. The leaves of the tree form the observable data, whereas the internal nodes represent the hidden hierarchical structure. The arrows connecting hidden to other hidden or observable nodes represent production rules.

Each hidden symbol can typically expand into several distinct sequences, each described by a separate production rule. By assigning probabilities to each production rule, a CFG becomes a PCFG, defining a probability distribution over sequences of symbols. Samples from a PCFG are generated by recursively applying production rules until a sequence of terminal symbols is obtained. Compared to simpler generative models such as Markov processes and regular grammars, PCFGs can model complex dependencies spanning unbounded distances [53,54], making them particularly suited for modeling structured real-world data. However, these models are not analytically tractable in general. To address this issue, we consider a simplified yet expressive class of PCFGs: the RHM.

B. The random hierarchy model

The RHM is an ensemble of PCFGs, where, as is customary in statistical mechanics, the production rules are drawn uniformly at random, compatibly with the following set of constraints.

(C1) *Fixed tree topology.* All grammars share a common tree structure: a fixed regular tree of arity s and depth L. An example having $s=2$ and $L=3$ is displayed in Fig. 1, right, where hidden variables are represented as variable nodes (empty circles) and we used factor nodes (full squares) to represent production rules. This tree serves as the shared backbone for all the generated data, which consists of sequences of $d=s^L$ symbols. Because of the fixed topology constraint, exact inference on data generated by the RHM can be performed using the belief propagation (BP) algorithm [55]. BP enables the efficient computation of posterior distributions over hidden symbols given observed data, as well as marginal distributions over observed symbols. In particular, it yields closed-form expressions for data statistics—such as symbol frequencies and pairwise correlations—for any given grammar in the ensemble. Moreover, since the grammars are sampled randomly, these quantities can be further averaged over the ensemble, allowing us to calculate expected values and variances of relevant observables analytically.

(C2) *Unambiguity.* Production rules are chosen so no two distinct hidden symbols are allowed to generate the same sequence of children. This constraint ensures that the observable datum uniquely determines the hidden structure of its derivation. Without some form of unambiguity constraint, a PCFG with randomly sampled production rules would typically generate sequences with weak or trivial correlations: the observable tokens would become nearly independent, losing any statistical imprint of the underlying hierarchical structure [56,57].

(C3) *Vocabulary size* v. Hidden symbols are split into L vocabularies $\mathcal{V}_\ell$, with $\ell=0,\dots,L-1$ and $\mathcal{V}_L\equiv\mathcal{V}$ denoting the vocabulary of observable symbols. All vocabularies have the same cardinality v.

(C4) *m production rules per symbol.* Each hidden symbol is associated with m distinct and equiprobable production

rules that yield symbols of the next level. Due to unambiguity, $mv \leqslant v^s$, i.e., $f := (m/v^{s-1}) \leqslant 1$.

Given an instance of the RHM, the data-generating process works as follows:

(i) Sample a root symbol $\mu^{(0)} \in \mathcal{V}_0$ according to the uniform probability $1/v$ (e.g., a in Fig. 1, right);

(ii) Sample one of the m production rules expanding $\mu^{(0)}$ into s symbols from $\mathcal{V}_1$,

$$\mu^{(0)} \to \mu_1^{(1)}, \ldots, \mu_s^{(1)}, \tag{1}$$

(e.g., $a \to b, c$ in Fig. 1, right);

(iii) replace $\mu^{(0)}$ with the right-hand side of Eq. (1) to obtain the next-level representation; and

(iv) Repeat steps (ii) and (iii) above for each element of the new representation and iterate L times.

This process defines a probability distribution over sequences in $(\mathcal{V})^d$,

$$P_X(\boldsymbol{x}) = \mathbb{P}\{X_1 = x_1, \ldots, X_d = x_d\}, \tag{2}$$

where the probability of each datum $\boldsymbol{x}$ is the product of probabilities for all production rules involved in its generation. Following standard language modeling jargon, we refer to the components of $\boldsymbol{x}$ as tokens.

IV. STATISTICS OF RHM DATA

Before turning to the question of learnability, we first examine the statistical structure of the sequences generated by the RHM. In particular, we study the correlations between tokens that are induced by the underlying hierarchical generative process. These correlations reflect the hidden tree structure shared by data and form the primary observable cues that a learning algorithm can exploit to infer such structure.

We define the correlation between the ith and jth tokens of the sequence via the $v \times v$ cooccurrence matrix $C(X_i, X_j)$, having elements

$$C(X_i, X_j)_{\mu,\nu} := \mathbb{P}\{X_i = \mu, X_j = \nu\} - \mathbb{P}\{X_i = \mu\}\mathbb{P}\{X_j = \nu\}. \tag{3}$$

The treelike structure gives a simple recipe for computing all the marginals appearing in Eq. (3): draw the path that connects the root of the tree to the tokens X_i and X_j, multiply the root prior probability $p(\mu^{(0)})$ by one transition probability per every factor node traversed by the path, then sum over all the hidden variables along the path. For instance, the one-token marginal for the first token of the sequence in Fig. 1, right, is given by

$$\mathbb{P}\{X_1 = \mu\} = \sum_{\mu_1^{(2)}} p\big(\mu|\mu_1^{(2)}\big) \sum_{\mu_1^{(1)}} p\big(\mu_1^{(2)}|\mu_1^{(1)}\big) \times \sum_{\mu^{(0)}} p\big(\mu_1^{(1)}|\mu^{(0)}\big) p(\mu^{(0)}), \tag{4}$$

where $\mu^{(0)}$ denotes the root symbol, $\mu_1^{(1)}$ the first level-1 symbol, $\mu_1^{(2)}$ the first level-2 symbol, and $p(\mu^{(\ell)}|\mu^{(\ell-1)})$ the transition probability associated with the corresponding production rule. The definition in Eq. (3) can be directly extended from observable tokens to hidden variables and tuples thereof.

The RHM ensemble provides specific statistics for the transition probabilities associated with the production rules. Using these statistics, we can obtain analytical expressions for the ensemble averages and variances of various correlation functions, as shown in Ref. [8] for the token-root and tuple-root correlations, in Ref. [10] for token-token and tuple-token correlations, and in Ref. [14] for the correlations between observable tokens and tuples of hidden variables. All correlations have zero mean over RHM realizations, whereas the variance is given by the following simple formula. Given two hidden or observable variables (X_i, X_j) and their tree distance $d_{\text{tree}}(X_i, X_j)$, then, if X_i is an ancestor of X_j or vice versa [8],

$$\langle (C(X_i, X_j)_{\mu,\nu})^2 \rangle_{\text{RHM}} \simeq \frac{1}{v^2} \frac{1}{v m^{d_{\text{tree}}(X_i, X_j)}}, \tag{5}$$

otherwise, if X_i and X_j are not on the same branch [10],

$$\langle (C(X_i, X_j)_{\mu,\nu})^2 \rangle_{\text{RHM}} \simeq \frac{1}{v^2} \frac{(1-f)}{v m^{d_{\text{tree}}(X_i, X_j)}}, \tag{6}$$

where the approximation $\simeq$ is exact in the limit of large m and v. This result represents a generic feature of RHM data: the correlations between any two symbols decay exponentially with the tree distance, i.e., the number of links in the shortest path connecting said two symbols.

A. Correlations and hidden variables

Due to the context-free structure, correlations involving the entire span of some hidden variable are representative of the value of such a variable. Consider, for instance, the joint probability $\mathbb{P}\{X_1 = \mu\}$ of the first 2 tokens for sentences generated by a RHM with $s = 2$ and $L = 3$ (as in the tree of Fig. 1, right).

$$\mathbb{P}\{X_1 = \mu_1, X_2 = \mu_2\} = \sum_{\mu_1^{(2)}} p\big(\mu_1, \mu_2|\mu_1^{(2)}\big) \times \sum_{\mu_1^{(1)}} p\big(\mu_1^{(2)}|\mu_1^{(1)}\big) \sum_{\mu^{(0)}} p\big(\mu_1^{(1)}|\mu^{(0)}\big) p(\mu^{(0)}). \tag{7}$$

Due to the unambiguity constraint C2, there is only one value of the latent variable $\mu_1^{(2)}$ compatible with μ_1, μ_2. Hence, the right-hand side of Eq. (7) simplifies, revealing that the joint probability $\mathbb{P}\{X_1 = \mu_1, X_2 = \mu_2\}$ depends on $\mu_1^{(2)}$ rather than (μ_1, μ_2). In other words, given the latent variable $\mu_1^{(2)}$, this probability is conditionally independent of (μ_1, μ_2). This property extends to all joint probabilities that include the entire span of some hidden variable in their argument and implies that the statistics of the span can be used to infer the corresponding hidden variable.

V. A THEORY OF LEARNING THE RHM

We now turn to the question of how the structured data generated by the random hierarchy model can be learned. In this section, we develop a theoretical framework for analyzing the learnability of the RHM by focusing on a simplified prediction task, which is defined precisely in Sec. V A. Our approach centers on understanding how a learner can exploit the statistical dependencies induced by the generative process, described in Sec. IV, to recover information about the hidden

hierarchical structure. This setting provides a tractable and analytically grounded perspective on the general problem of learning compositional data.

A. The last-token prediction task

Several language modeling techniques have been developed to provide approximations of Eq. (2). For example, Masked Language Modeling randomly replaces a fraction of tokens with a special symbol x_{mask}, and the learner is tasked with inferring their original value from the context [58]. In contrast, autoregressive language models sequentially predict each token based on all preceding tokens in the sequence [59]. Here, following Ref. [10], we consider a simplified setup where the learner is tasked with predicting the last token of the sequence from the remaining tokens or context. Due to the focus on the end of the sequence, we will follow the notation of Fig. 1, right, and use the subscript -1 for the last token of the sequence, -2 for the penultimate, and so on. In this notation, the learner aims to approximate the conditional probability

$$P(X_{-1} \mid X_{-2} = x_{-2}, \ldots, X_{-d} = x_{-d}) \tag{8}$$

from a training set of P samples of valid RHM sequences. We denote the learner's approximation with $\hat{P}(X_{-1}|x_{-(2:d)})$. The quality of the approximation is measured via the test cross-entropy loss:

$$\mathcal{L}(\hat{P}) = -\mathbb{E}_{\boldsymbol{x}\sim P_X}[\ln \hat{P}(X_{-1} = x_{-1}|x_{-(2:d)})]. \tag{9}$$

B. Flat vs hierarchical approximation: the role of *n* grams

A standard strategy in classical language modeling is to approximate the conditional distribution of the next token based on fixed-length contexts. In the last-token prediction task, this leads to an n-gram approximation of the form

$$P(X_{-1} \mid X_{-2}, \ldots, X_{-d}) \approx P(X_{-1} \mid X_{-2}, \ldots, X_{-n}), \tag{10}$$

where the model estimates the probability of X_{-1} given the most recent $n-1$ tokens. Being agnostic of the underlying structure, this approximation requires sampling all length-n contexts. As the number of these contexts grows exponentially with n, this approximation is not data efficient.

In contrast, taking the hidden context-free structure into account yields a much more efficient strategy. Indeed, as shown in Sec. IV A, given a hidden variable, its span is conditionally independent of the rest of the sequence. As a result, the content of a length-n context is determined by fewer than n variables: using these variables would dramatically reduce the number of contexts to sample. In the specific case of last-token prediction, the variables contributing to the prediction of X_{-1} include

(1) The $(s-1)$ siblings $(X_{-s}, \ldots, X_{-2})$, generated with X_{-1} by the last level-L hidden variable $\mu_{-1}^{(L)}$.

(2) The $(s-1)$ siblings of $\mu_{-1}^{(L)}$, and of all the other ancestors $\mu_{-1}^{(\ell)}$ of X_{-1}.

(3) The root symbol $\mu^{(0)}$.

The total count sums up to $(s-1)L+1$, exponentially smaller than the full input length $d = s^L$.

The contrast between the flat and hierarchical n-gram approximations reveals the possibility of learning the conditional last-token probability efficiently by leveraging the hierarchical structure of the data. The efficient strategy requires knowledge of the hidden structure of the data, which is not explicitly provided to the learner. However, as we will demonstrate in the next subsection, an ideal learner could use observable statistics—specifically, token-token correlations—to reconstruct the hidden hierarchical structure of the data.

C. Reconstruction of the hidden variables via correlations

To access the hierarchical n-gram approximation, a learner requires access to the hidden variables, which are not explicitly provided in the training data. However, as shown in Sec. IV A, the joint statistics of spans of tokens generated by the same hidden variable are representative of the hidden variable itself. As a result, a learner could use these statistics to cluster all the spans corresponding to the same hidden variable, i.e., infer this variable. The only obstacle to inference is the finite number of training examples, implying that the learner has access only to noisy measurements of such statistics.

1. Reconstruction of depth-1 hidden variables

For concreteness, let us consider the correlations between the last token X_{-1} and one of the entire s tuples in the context, $\boldsymbol{X}_{-t} := (X_{-ts}, \ldots, X_{-(t-1)s+1})$ with $t = 2, \ldots, s^{L-1}$. These correlations are given by the following $(vm) \times v$ matrix:

$$C(\boldsymbol{X}_{-t}, X_{-1})_{\boldsymbol{\mu},\nu} := \mathbb{P}\{\boldsymbol{X}_{-t} = \boldsymbol{\mu}, X_{-1} = \nu\} - \mathbb{P}\{\boldsymbol{X}_{-t} = \boldsymbol{\mu}\}\mathbb{P}\{X_{-1} = \nu\}. \tag{11}$$

According to Sec. IV, entries corresponding to the m distinct $\boldsymbol{\mu}'$s generated by the same latent variable $\mu_{-t}^{(L)}$ are all equal to each other. In other words, there are only v independent rows of $C(\boldsymbol{X}_{-t}, X_{-1})$, each repeating m times. As a result, a perfect measurement of correlations allows for clustering the s-tuple index $\boldsymbol{\mu}$ by the associated latent variable. However, the noise induced by the finite training set might hinder the inference of $\mu_{-t}^{(L)}$. For a large training set size P, such noise is equivalent to an additive Gaussian noise with zero mean and variance [10]:

$$\sigma_P^2 = \frac{1}{(vm)vP}. \tag{12}$$

If this variance is larger than the difference between correlations associated with distinct hidden variables, then inference of the hidden variable is impossible.

We estimate the difference between correlations corresponding to different hidden variables with the variance of $C(\boldsymbol{X}_{-t}, X_{-1})_{\boldsymbol{\mu},\nu}$ over realizations of the RHM [10]:

$$\left\langle (C(\boldsymbol{X}_{-t}, X_{-1})_{\boldsymbol{\mu},\nu})^2 \right\rangle_{\text{RHM}} = \left\langle \frac{1}{m}\left(C\left(X_{-t}^{(L)}, X_{-1}\right)_{\boldsymbol{\mu},\nu}\right)^2 \right\rangle_{\text{RHM}} \simeq \frac{1}{(vm)v}\frac{(1-f)}{vm^{d_{\text{tree}}(X_{-t}^{(L)}, X_{-1})}}. \tag{13}$$

Comparing Eq. (12) with Eq. (13), we get the following condition for the inference of the depth-1 hidden variable $\mu_{-t}^{(L)}$:

$$P \gg (1-f)^{-1} vm^{d_{\text{tree}}(X_{-t}^{(L)}, X_{-1})}. \tag{14}$$

Due to the fixed tree topology assumption C1, the distance $d_{\text{tree}}(X_{-t}^{(L)}, X_{-1})$ depends on the depth of the lowest common ancestor of $X_{-t}^{(L)}$ and X_{-1}: for all $s^{\ell-1} < t < s^{\ell}$, $d_{\text{tree}}(X_{-t}^{(L)}, X_{-1}) = 2\ell + 1$. Therefore, Eq. (14) yields $L-1$ different sample complexities for the inference of the hidden variables $\mu_{-t}^{(L)}$ with $t < s^{\ell}$ with $\ell = 1, \dots, L-1$,

$$P_\ell = (1-f)^{-1} v m^{2\ell+1}. \tag{15}$$

Note that the exponential increase of P_ℓ with the tree distance $2\ell+1$ mirrors the exponential decrease of correlations as the number of edges between the variables on the tree graph increases.

2. Reconstruction of deeper hidden variables

The same ideas apply for the inference of the next layer of hidden variables, i.e., depth-2 or level-$(L-1)$ variables, via correlations between X_{-1} and s-tuples of level-L variables. These tuples have a smaller tree distance from X_{-1} than tuples of input tokens, thus the corresponding sample complexities are lower than those required by Eq. (15). Hence, when $P \gg P_\ell$, all the latent variables spanning the context going $X_{-s^{\ell+1}}$ to $X_{-(s+1)}$ can be inferred. We can formalize this concept via the following assumption:

Assumption 1. A learner discovers the hidden structure up to depth ℓ when the correlations between the last token X_{-1} and the s tuples of observable tokens $\boldsymbol{X}_{-t}$ at tree distance $d_{\text{tree}}(\boldsymbol{X}_{-t}, X_{-1}) = 2\ell + 1$ become detectable from the training data.

This assumption is crucial for predicting scaling laws. Given a data budget P, the assumption determines the portion of the data-generating model available to the learner, which, in turn, as we explain in Sec. VII, determines the last-token prediction performance.

3. Improved sample complexities due to translation invariance

In the RHM, the same set of production rules applies to all variables at a given level, independently of the position. Therefore, a learner could infer all level-L latents as soon as it is able to infer $\mu_{-2}^{(L)}$, which is the easiest to infer as it has the strongest correlation with X_{-1}. If this were the case, we could assume the learner can access all the level-L hidden variables with $P \gg P_1$ data. This would improve the sample complexities for the inference of deeper hidden variables, as the learner could correlate the last token directly with tuples of hidden variables, having stronger correlations than the tuples of observable tokens. The corresponding condition for the sample complexities required to infer all latents of depth ℓ and smaller is [14]

$$P \gg \bar{P}_\ell := (1-f)^{-1} v m^{\ell+2}. \tag{16}$$

As we demonstrate empirically in Sec. VII, the performance scaling of convolutional neural networks (CNNs) is compatible with this improved strategy, whereas the scaling of transformers follows Eq. (15).

VI. ARCHITECTURES

In this section, we describe the different architectures considered in this paper.

A. Convolutional neural networks

CNNs are a class of architectures originally developed for image processing [60], where they exploit local connectivity and translation invariance to efficiently model spatial patterns. Beyond images, CNNs have also been successfully applied to sequential data, such as text [61,62], by adapting the operations to one-dimensional inputs. In this work, we focus on one-dimensional CNNs, which are well-suited for sequence modeling. More specifically, our CNN architectures are designed to match the hierarchical structure of the RHM. Each network consists of L hidden layers, with each layer corresponding to one level of the underlying tree. Filters operate without overlap: the filter size and stride are both set to the RHM branching factor s, and no padding is used.

CNNs receive as input a sequence of one-hot encodings of the d input features: each token x_i is a v-dimensional vector, whose only nonvanishing component coincides with the vocabulary entry. The $(d-1)$ observable tokens are whitened, so as to have $\sum_{\mu=1}^{v} x_{i,\mu} = 0$, $\sum_{\mu=1}^{v} (x_{i,\mu})^2 = 1$ for all i's. The last token is replaced with the dummy token $x_\mu = v^{-1/2}$. The activations h_ℓ at layer ℓ are defined recursively as

$$h_{\ell,i} = \sigma\left(W^{(\ell)} h_{\ell-1,si:s(i+1)-1} + b^{(\ell)}\right), \tag{17}$$

where $W^{(\ell)}$ and $b^{(\ell)}$ denote the weight matrix and bias at layer ℓ, $\sigma(\cdot)$ is a pointwise nonlinearity (e.g., ReLU), and $h_{\ell-1,si:s(i+1)-1}$ denotes the s-dimensional slice of the previous layer's activations corresponding to the ith output at level ℓ. The weights $W^{(\ell)}$ are order-3 tensor, $W^{(\ell)} \in \mathbb{R}^{w_\ell \times w_{\ell-1} \times s}$, with $w_0 = v$ and $w_\ell = w$ for $\ell \geqslant 1$. The output is obtained by applying a $v \times w$ linear readout a on the last hidden layer:

$$f_{\text{CNN}}(x_{-(2:d)}) = w^{-1} a \cdot h_L(x_{-(2:d)}). \tag{18}$$

We consider the infinite-width limit $w \to \infty$, which in our setting is realized around $w = 512$, and we verified empirically that increasing the width further does not affect the results presented in the paper. The parameters are trained with stochastic gradient descent (SGD) on the cross-entropy loss Eq. (9), with learning rate w and batch size 128. We consider the setting of online training, where a new batch of data is generated independently at each training step, so the total number of training data is linearly proportional to the number of steps.

B. Transformers

Transformer architectures have emerged as a central paradigm in modern machine learning [63]. Their key innovation is the self-attention mechanism, which enables models to aggregate information from different parts of an input sequence, irrespective of their distance, and thus capture long-range dependencies effectively.

We consider a simplified one-dimensional transformer consisting of L hidden layers. Each layer applies a multihead self-attention operation over the full sequence, without any additional feedforward operations. Unlike CNNs, transformers preserve the sequence length across layers and can directly model global interactions between all input tokens.

Mathematically, the transformer requires first a projection of the input tokens into an embedding space of dimension d_K,

which we achieve via a learnable $v \times d_K$ matrix acting on the one-hot encodings of the tokens. A random and learnable d_K-dimensional sequence, known as positional encoding, is added to each token to break the permutation equivariance of the rest of the architecture. The following activations $h_{\ell,i}$ at layer ℓ and position i are given by

$$h_{\ell,i} = \mathrm{MHA}^{(\ell)}(h_{\ell-1,1}, \ldots, h_{\ell-1,d})_i, \tag{19}$$

where $\mathrm{MHA}^{(\ell)}$ denotes the multihead attention operation at layer ℓ, and h_0 represents the input embeddings. In particular, each MHA layer first computes, for each head h, learned projections of the previous layer outputs: $Q_i^{(\ell,h)} = W_Q^{(\ell,h)} h_{\ell-1,i}$, $K_i^{(\ell,h)} = W_K^{(\ell,h)} h_{\ell-1,i}$, $V_i^{(\ell,h)} = W_V^{(\ell,h)} h_{\ell-1,i}$, where $W^{(\ell,h)}$'s are trainable weight matrices. For each head, attention scores $\alpha^{(\ell,h)}$'s and head outputs are then computed as $\alpha_{ij}^{(\ell,h)} = \mathrm{softmax}_j(d_k^{-1/2} Q_i^{(\ell,h)} \cdot K_j^{(\ell,h)})$, $\mathrm{head}_i^{(\ell,h)} = \sum_{j=1}^{d} \alpha_{ij}^{(\ell,h)} V_j^{(\ell,h)}$, with d_k denoting the dimension of the key vectors. The outputs of all heads are concatenated and projected to produce the layer output:

$$\mathrm{MHA}^{(\ell)}(h_{\ell-1,1}, \ldots, h_{\ell-1,d})_i = W_O^{(\ell)} \mathrm{multihead}^{(\ell)}, \tag{20}$$

where $W_O^{(\ell)}$ is learned and $\mathrm{multihead}^{(\ell)} = \mathrm{concat}(\mathrm{head}_i^{(\ell,1)}, \ldots, \mathrm{head}_i^{(\ell,H)})$. This global operation enables each token to aggregate information from the entire sequence at each layer. However, the absence of built-in locality biases requires the model to learn the hierarchical structure from the data.

The output of the architecture is obtained by applying a $v \times w$ linear readout a on the last token of the last hidden layer:

$$f_{\mathrm{MLA}}(x_{-(2:d)}) = w^{-1} a \cdot h_{L,-1}(x_{-(2:d)}). \tag{21}$$

We set $d_K = 512$ and the number of heads to 16, and we verified empirically that our results are stable to changing the number of heads at fixed d_K and to varying d_K in the range [256,512] with the number of heads fixed to $d_K/32$. The parameters of the transformers are trained with the Adam optimizer—a variant of SGD that adapts the learning rate for each parameter based on its gradient history—on the cross-entropy loss Eq. (9), with learning rate 10^{-3} and batch size 128. As for CNNs, we consider the online training setting.

VII. PREDICTED SCALING LAWS VS EMPIRICAL LEARNING CURVES

In this section, we compare the predictions outlined in Sec. V with the training dynamics of deep transformers and CNNs. We consider an online training setup, where a new batch of $B = 128$ training data is sampled independently at each training step. As a result, the total number of training data is proportional to the number of training steps.

A. Transformers

Figure 2 illustrates the stagewise dynamics of deep transformers, mirroring the stagewise learning curves observed in Ref. [10]. The dashed lines represent the random-choice loss (or the s^0-gram) and the ensemble-averaged s^ℓ-grams losses,

$$\mathcal{L}_\ell := \langle \mathbb{E}_{\boldsymbol{x}\sim P_X}[-\ln P(X_{-1} = x_{-1} | x_{-(2:s^\ell)})] \rangle_{\mathrm{RHM}}, \tag{22}$$

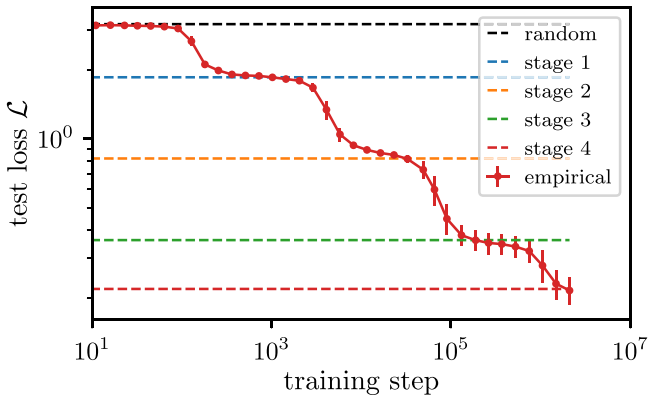

FIG. 2. Training dynamics of a depth-4 Transformer trained on RHM data with $L = 4$, $s = 2$, $v = 24$, and $m = 6$. At each step of training, the test cross-entropy loss is evaluated on a test set of size 2^{15} and averaged over eight independent realizations of the RHM. This procedure applies to all figures in the paper unless otherwise stated. Colored dashed lines show the averaged s^ℓ-gram losses defined in Eq. (22): as the number of training steps increases, the transformer's performance transitions between successive approximations.

with $\ell = 1, \ldots, L$. While the random-choice loss is simply given by $\ln v$, the $\mathcal{L}_\ell$'s are computed by sampling 64 independent instances of the RHM with fixed L, s, v, and m, obtaining the losses of each instance exactly via belief propagation, then averaging over the sampled instances. As the number of training steps increases, the performance jumps from one approximation to the next.

The scaling law is obtained following Assumption 1: A learner having access to the hidden hierarchical structure up to depth ℓ outputs the true s^ℓ-gram probability of the data. Therefore, we get the scaling law by inverting the right-hand side of Eq. (15) for ℓ as a function of P, then substituting it into the asymptotic behavior of $\mathcal{L}_\ell$ with ℓ found in Ref. [8]:

$$\mathcal{L}_\ell - \mathcal{L}_\infty \sim f^\ell, \quad P_\ell \sim m^{2\ell} \Rightarrow \mathcal{L}(P) - \mathcal{L}_\infty \sim P^{\frac{\ln f}{2 \ln m}}. \tag{23}$$

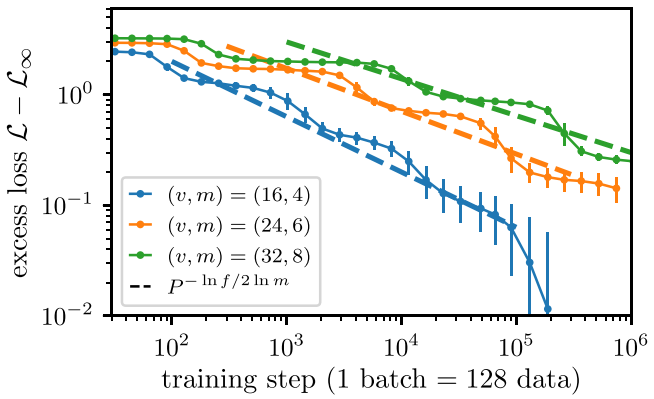

FIG. 3. Scaling of the excess test loss, $\mathcal{L} - \mathcal{L}_\infty$, as a function of the number of training steps for depth-4 transformers trained on RHMs with shared tree structure ($L = 4$, $s = 2$) but varying values of m and v (indicated in the legend). Each parameter set is associated with a distinct color. Solid lines show empirical results, while dashed lines represent scaling predictions from Eq. (23).

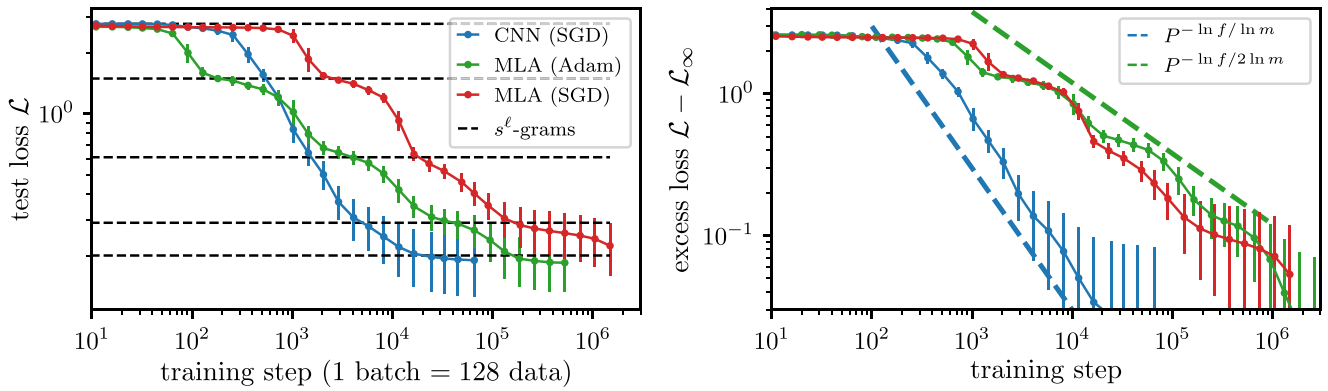

FIG. 4. Training dynamics of depth-4 models on RHM data with $L = 4$, $s = 2$, $v = 16$, and $m = 4$. Left: Test cross-entropy loss as a function of the number of training steps for depth-4 transformers trained with Adam and vanilla SGD, and for depth-4 CNNs trained with SGD. Black dashed lines indicate the s^ℓ-gram losses. CNNs exhibit a much faster loss decay compared to Transformers, and a constant delay is observed between the Adam- and SGD-trained transformers, with Adam reaching lower losses earlier. Right: Scaling of the excess test loss, $\mathcal{L} - \mathcal{L}_\infty$, as a function of the number of training steps. The Adam curve is rescaled to overlap with the transformer SGD curve, highlighting their similar scaling behaviors. By contrast, CNNs display a different scaling behavior. Both CNN and transformer scalings are compared with the corresponding theoretical predictions reported in the legend.

As shown in Fig. 3, Eq. (23) correctly predicts the decay of the loss of deep transformer toward the limiting value $\mathcal{L}_\infty$ for several values of the RHM parameters m and v.

B. Hierarchical CNNs

Figure 4 compares dynamics of a deep transformer (green curve) to that of a deep CNN tailored to the structure of the RHM (blue curve), i.e., having L hidden layers, with filter size $s = 2$ and stride equal to s. The CNN dynamics crosses the four s^ℓ-gram approximation stages sequentially, although the decay is significantly faster than for the transformer. This is highlighted in the right panel, where the asymptotic entropy of the dataset $\mathcal{L}_\infty$ is subtracted. As the CNN is trained with vanilla SGD instead of Adam, we also included the dynamics of a transformer trained with vanilla SGD to check that the difference in scaling is not due to the difference in the optimizer. While the optimizer causes a shift of the dynamics, the two transformer curves (green and red) display the same decay

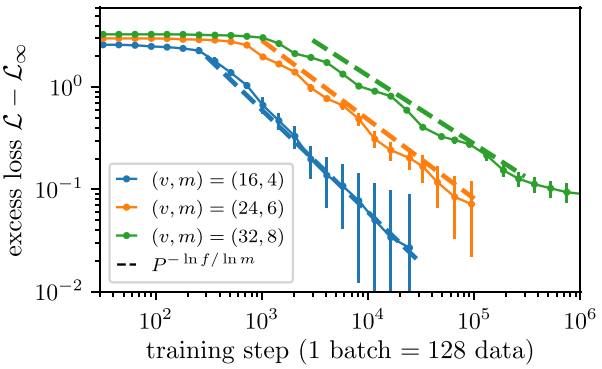

FIG. 5. Scaling of the excess test loss, $\mathcal{L} - \mathcal{L}_\infty$, as a function of the number of training steps for depth-4 CNNs trained on RHMs with shared tree structure ($L = 4$, $s = 2$) but varying values of m and v (indicated in the legend). Each parameter set is associated with a distinct color. Solid lines show empirical results, while dashed lines represent scaling predictions from Eq. (24).

towards $\mathcal{L}_\infty$. This is confirmed in the right panel of the figure, where the two curves are made to overlap after rescaling the x axis of the green curve by a constant factor.

As discussed in Sec. V C 3, CNNs could use the weight-sharing property of the architecture to exploit stronger correlations than those used by the transformer, thus enjoying a faster loss decay. To test this hypothesis, we compare the CNN dynamics with the scaling obtained by substituting the relationship between ℓ and P from Eq. (16) into the asymptotic behavior of $\mathcal{L}_\ell$ with ℓ:

$$\mathcal{L}_\ell - \mathcal{L}_\infty \sim f^\ell, \quad \bar{P}_\ell \sim m^\ell \Rightarrow \mathcal{L}(P) - \mathcal{L}_\infty \sim P^{\frac{\ln f}{\ln m}}, \tag{24}$$

to be compared with Eq. (23) of the previous section. Fig. 5 shows that the loss scaling of deep CNNs agrees with Eq. (24) for several values of the RHM parameters v and m.

VIII. DYNAMICS OF THE HIDDEN REPRESENTATIONS

To further understand how models trained on RHM data build hierarchical representations, we study the behavior of hidden activations under specific input transformations [10]. Given an input sequence $\boldsymbol{x}$ and its associated generative tree, we consider two types of transformations:

(1) *Random variable replacement* $\mathcal{S}_{\ell,i}$. Replace the ith hidden variable at level ℓ of the tree with another symbol randomly chosen from the vocabulary. This modifies the entire subtree rooted at that variable.

(2) *Random rule replacement* $\mathcal{R}_{\ell,i}$. Resample the production rule emanating from the ith variable at level ℓ, while keeping the variable itself fixed. This alters the subtree while preserving the identity of the hidden variable.

Following Sec. VI, we denote the hidden representation at layer ℓ and position i by $h_{\ell,i}(x)$. For transformers, $i = -1, \ldots, -d$ independently of ℓ, whereas, for hierarchical CNNs, $i = -1, \ldots, -s^{L-\ell}$. To probe their structure, we consider the standardized representations

$$\hat{h}_{\ell,i}(\boldsymbol{x}) = \frac{h_{\ell,i}(\boldsymbol{x}) - \mathbb{E}_{\boldsymbol{x}}[h_{\ell,i}(\boldsymbol{x})]}{\sqrt{\mathrm{Var}_{\boldsymbol{x}}[h_{\ell,i}(\boldsymbol{x})]}}, \tag{25}$$

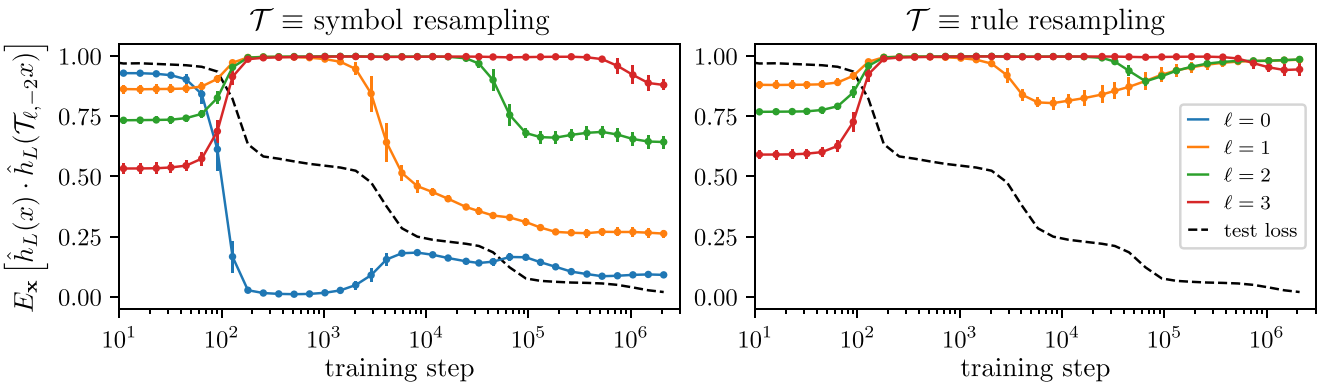

FIG. 6. Dynamics of hidden representations of depth-4 transformers trained on RHM data with $L=4$, $s=2$, $v=24$, and $m=6$. The solid curves measure the cosine similarity between the last-layer representation of the original and transformed data, where transformations alter the RHM tree structure. Left: Random replacements of hidden variables at depth ℓ indicated by color, always applied to the penultimate hidden variable ($i=-2$). Right: Random replacements of production rules emanating from the penultimate hidden variable at depth ℓ. The test loss dynamics are shown as a black dashed line for reference.

where the mean and variance are taken over the RHM data. The invariance of a representation to a transformation $\mathcal{T}$ is quantified via the cosine similarity between the original and transformed standardized representations:

$$q_{\mathcal{T}}(h_{\ell,i}) = \mathbb{E}_{\boldsymbol{x}}[\hat{h}_{\ell,i}(\boldsymbol{x}) \cdot \hat{h}_{\ell,i}(\mathcal{T}\boldsymbol{x})]. \tag{26}$$

A drop in $q_{\mathcal{T}}(h)$ indicates that the representation is sensitive to the transformation. In particular, invariance to $\mathcal{R}_{\ell,i}$ but sensitivity to $\mathcal{S}_{\ell,i}$ signals that the representation encodes the hidden symbol itself but not the details of its generated subtree.

Figure 6 reports the evolution of $q_{\mathcal{T}}$ for the last-layer representation h_L of a Transformer during training. The figure shows the last representation, with $i=-1$, but all positions display the same behavior. The transformations considered are the random replacement of the penultimate observable token, x_{-2}, the penultimate hidden variable in each level of the tree, $x_{-2}^{L-\ell}$, and the production rule immediately below. Initially, all the cosine similarities start at different baseline values, reflecting the varying extent to which each transformation alters the input sequence. In correspondence with the first sharp drop of the test loss, all invariance scores sharply increase to 1, except for the one associated with resampling the penultimate observable token (blue line on the left panel). This behavior indicates that, at this stage, the model relies predominantly on the token immediately preceding the masked token for its prediction.

Subsequently, as training progresses, the invariances to resampling deeper hidden symbols drop sequentially, in order of depth. This sequence of drops signals that the model progressively incorporates longer-range dependencies, leveraging increasingly larger portions of the context window. Notably, the invariance to symbol replacements drops significantly at each stage, whereas the invariance to rule replacements recovers and approaches unity after each transition. This behavior strongly suggests that the hidden representations encode the high-level hidden variables associated with the hierarchical structure, while becoming increasingly insensitive to the specific realizations of the subtrees they generate.

As shown in Fig. 7, the hidden representations of hierarchical CNNs display the same phenomenology.

IX. LOCALITY WITHOUT WEIGHT-SHARING: LOCALLY CONNECTED NETWORKS

In this section, we extend our analysis to LCNs. These architectures retain the local connectivity of CNNs but remove weight sharing across spatial locations. By comparing their

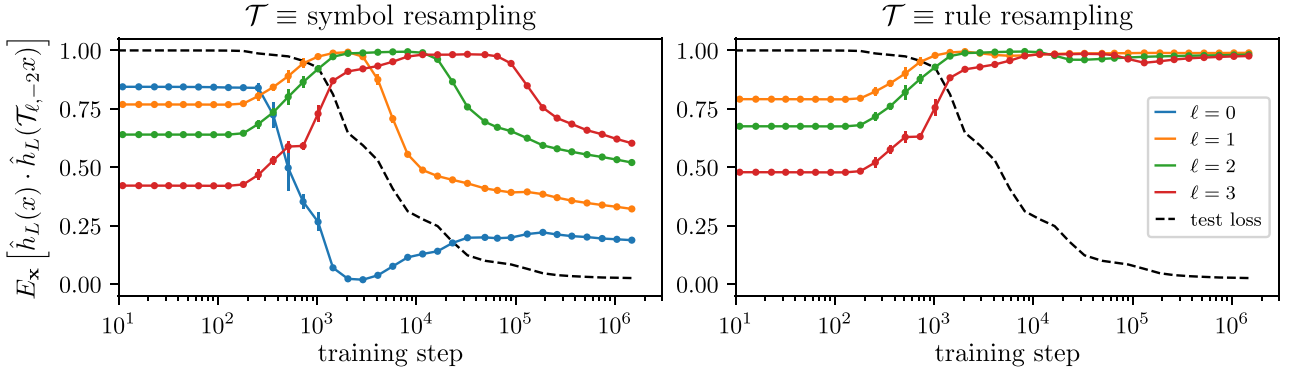

FIG. 7. Dynamics of hidden representations for a depth-4 CNN trained on RHM data with $L=4$, $s=2$, $v=32$, and $m=8$. The figure follows the layout of Fig. 6.

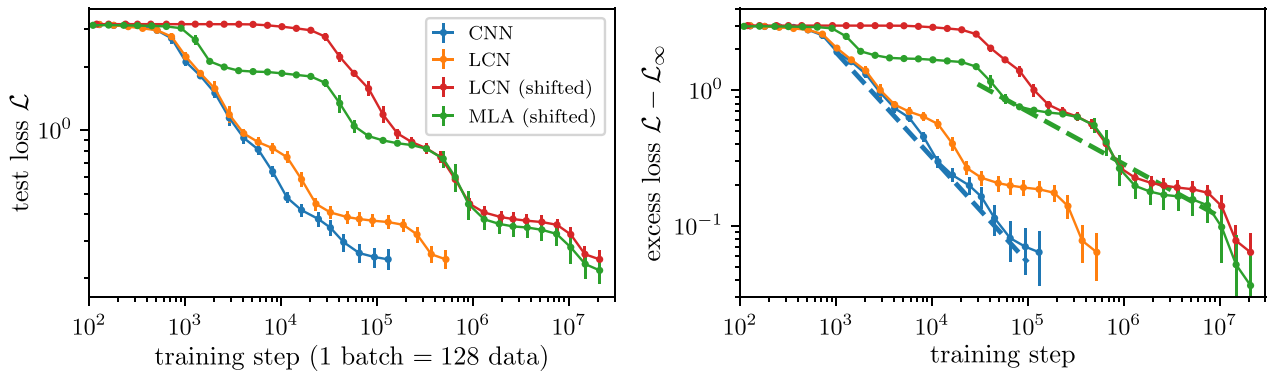

FIG. 8. Comparison of training dynamics of depth-4 CNNs, transformers, and LCNs on RHM data with $L = 4$, $s = 2$, $v = 24$, and $m = 6$. Left: Test cross-entropy loss as a function of the number of training steps. The LCN loss curve (orange) follows that of CNNs (blue) for the first two loss steps, but deviates afterward. Rescaling the LCN curve by multiplying the x axis by a constant (red curve) reveals that the later steps overlap with those of the transformer loss curve (green curve, also shifted for visualization purposes). Right: Scaling of the excess test loss, $\mathcal{L} - \mathcal{L}_\infty$, confirming the shared asymptotic behavior after the shift.

performance to that of CNNs and transformers, we aim to isolate the role of weight sharing in achieving the improved scaling of the test loss displayed by CNNs.

Figure 8 shows the evolution of the test loss of LCNs as a function of training steps, alongside those of CNNs and transformers. Initially, the LCN loss follows the CNN loss closely, with the first two steps—corresponding to the inference of depth-1 hidden variables—occurring at similar times. However, from the third step onward, the LCN loss deviates: the number of training steps required for the third loss step is noticeably larger than for CNNs, and the fourth step requires even more training. The shape of the loss curve beyond the second step closely matches that of the transformer, up to a constant rescaling of the number of training steps. This observation suggests that the improved scaling of CNNs at later stages is primarily due to their weight-sharing structure.

X. CONCLUSIONS

In this work, we studied the learning dynamics of deep neural networks trained on hierarchically structured data. Leveraging synthetic datasets generated by the RHM—an analytically tractable ensemble of probabilistic context-free grammars—we analyzed how the test loss evolves as a function of training steps under online stochastic gradient descent. Our theoretical framework elucidates how the observed power-law scaling of performance with the number of training steps (or, equivalently, training samples) arises naturally through the progressive acquisition of increasingly deeper layers of the underlying hierarchical structure.

By comparing convolutional networks, transformers, and locally connected networks, we demonstrated that different architectural priors lead to systematic differences in learning dynamics and scaling laws. In particular, convolutional networks tailored to the hierarchical structure of the data achieve a faster scaling of performance. This finding seems to contradict the common intuition that transformer-based models are better suited for sequence-modelling tasks. However, this contrast highlights the opportunity to use controlled synthetic datasets as probes to uncover the conditions under which different inductive biases achieve better performances. In our case, the strict hierarchical structure of the data aligns naturally with the design of convolutional networks, giving them the edge. One natural extension is to consider variable tree topologies, introducing spatial heterogeneities that might be better exploited by the flexibility of attention mechanisms. Another is to incorporate context-sensitive dependencies, pushing beyond the context-free structure. Exploring such modifications offers a systematic path to understanding when and how transformer architectures gain their empirical edge.

Several extensions are also possible within the framework of the current data model. On the theoretical side, a formal characterization of the training dynamics of deep networks on hierarchical data could sharpen our understanding of the mechanisms driving representation learning and the ensuing scaling behavior. The empirical analysis of internal representations presented in Sec. VIII provides a promising starting point, as it identifies the information encoded at different layers of trained networks and how this evolves with training. On the empirical side, applying similar analyses to real-world datasets with known hierarchical structure would test the generality of our framework and could guide the design of architectures better suited for compositional learning in practice.

ACKNOWLEDGMENT

The work of FC was supported by the European Union's Horizon Europe program under the Marie Skłodowska-Curie Grant Agreement No. 101154584.

DATA AVAILABILITY

The data that support the findings of this article are openly available [64], embargo periods may apply.

[1] J. Hestness, S. Narang, N. Ardalani, G. Diamos, H. Jun, H. Kianinejad, M. M. A. Patwary, Y. Yang, and Y. Zhou, Deep learning scaling is predictable, empirically, arXiv:1712.00409.

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

[3] T. Henighan, J. Kaplan, M. Katz, M. Chen, C. Hesse, J. Jackson, H. Jun, T. B. Brown, P. Dhariwal, S. Gray, *et al.*, Scaling laws for autoregressive generative modeling, arXiv:2010.14701.

[4] J. Hoffmann, S. Borgeaud, A. Mensch, E. Buchatskaya, T. Cai, E. Rutherford, D. de Las Casas, L. A. Hendricks, J. Welbl, A. Clark, *et al.*, *Training Compute-Optimal Large Language Models* (NeurIPS, 2022).

[5] M. E. Fisher, The theory of equilibrium critical phenomena, Rep. Prog. Phys. **30**, 615 (1967).

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

[7] E. Malach and S. Shalev-Shwartz, The implications of local correlation on learning some deep functions, in *Advances in Neural Information Processing Systems*, Vol. 33 (Curran Associates, Inc., virtual, 2020), pp. 1322–1332.

[8] F. Cagnetta, L. Petrini, U. M. Tomasini, A. Favero, and M. Wyart, How deep neural networks learn compositional data: The random hierarchy model, Phys. Rev. X **14**, 031001 (2024).

[9] U. M. Tomasini and M. Wyart, How deep networks learn sparse and hierarchical data: The sparse random hierarchy model, in *Forty-first International Conference on Machine Learning* (PMLR, Vienna, Austria, 2024).

[10] F. Cagnetta and M. Wyart, Towards a theory of how the structure of language is acquired by deep neural networks, in *The Thirty-eighth Annual Conference on Neural Information Processing Systems* (Cambridge University Press, Vancouver, Canada, 2024).

[11] J. Garnier-Brun, M. Mézard, E. Moscato, and L. Saglietti, *How Transformers Learn Structured Data: Insights from Hierarchical Filtering* (ICML, 2025).

[12] A. Sclocchi, A. Favero, and M. Wyart, A phase transition in diffusion models reveals the hierarchical nature of data, Proc. Natl. Acad. Sci. USA **122**, e2408799121 (2025).

[13] A. Sclocchi, A. Favero, N. I. Levi, and M. Wyart, Probing the latent hierarchical structure of data via diffusion models, in *The Thirteenth International Conference on Learning Representations* (OpenReview.net, Singapore, 2025).

[14] A. Favero, A. Sclocchi, F. Cagnetta, P. Frossard, and M. Wyart, How compositional generalization and creativity improve as diffusion models are trained, in *Proceedings of the 42nd International Conference on Machine Learning (ICML)* (PMLR, 2025).

[15] N. Chomsky, Three models for the description of language, IEEE Trans. Inform. Theory **2**, 113 (1956).

[16] M. Hutter, Learning curve theory, arXiv:2102.04074.

[17] E. J. Michaud, Z. Liu, U. Girit, and M. Tegmark, The quantization model of neural scaling, in *Thirty-seventh Conference on Neural Information Processing Systems* (Curran Associates, Inc., New Orleans, Louisiana, USA, 2023).

[18] A. Caponnetto and E. De Vito, Optimal rates for the regularized least-squares algorithm, Found. Comput. Math. **7**, 331 (2007).

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

[20] B. Bordelon, A. Canatar, and C. Pehlevan, Spectrum dependent learning curves in kernel regression and wide neural networks, in *International Conference on Machine Learning* (PMLR, 2020), pp. 1024–1034.

[21] Y. Bahri, E. Dyer, J. Kaplan, J. Lee, and U. Sharma, Explaining neural scaling laws, Proc. Natl. Acad. Sci. USA **121**, e2311878121 (2024).

[22] A. Favero, F. Cagnetta, and M. Wyart, Locality defeats the curse of dimensionality in convolutional teacher-student scenarios, J. Stat. Mech. (2022) 114012.

[23] H. Cui, B. Loureiro, F. Krzakala, and L. Zdeborová, Generalization error rates in kernel regression: The crossover from the noiseless to noisy regime, in *35th Conference on Neural Information Processing Systems* (NeurIPS, 2021), Vol. 34, pp. 10131–10143.

[24] A. Maloney, D. A. Roberts, and J. Sully, A solvable model of neural scaling laws, arXiv:2210.16859.

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

[26] B. Bordelon, A. Atanasov, and C. Pehlevan, A dynamical model of neural scaling laws, in *Forty-first International Conference on Machine Learning* (PMLR, Vienna, Austria, 2024).

[27] A. Jacot, F. Gabriel, and C. Hongler, Neural tangent kernel: Convergence and generalization in neural networks, in *Advances in Neural Information Processing Systems* (NeurIPS, 2018), Vol. 31, pp. 8571–8580.

[28] L. Chizat, E. Oyallon, and F. Bach, On lazy training in differentiable programming, in *Advances in Neural Information Processing Systems* (NeurIPS, 2019), pp. 2937–2947.

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

[30] 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 Inform. Proc. Syst. **35**, 37932 (2022).

[31] A. Bietti, J. Bruna, C. Sanford, and M. J. Song, Learning single-index models with shallow neural networks, Adv. Neural Inform. Proc. Syst. **35**, 9768 (2022).

[32] Y. Dandi, F. Krzakala, B. Loureiro, L. Pesce, and L. Stephan, How two-layer neural networks learn, one (giant) step at a time, J. Mach. Learn. Res. **25**, 1 (2023).

[33] B. Bordelon, A. Atanasov, and C. Pehlevan, How feature learning can improve neural scaling laws, in *The Thirteenth International Conference on Learning Representations* (OpenReview.net, Vienna, Austria, 2025).

[34] E. Mossel, Deep learning and hierarchal generative models, arXiv:1612.09057.

[35] S. Mei, U-nets as belief propagation: Efficient classification, denoising, and diffusion in generative hierarchical models, in *The Thirteenth International Conference on Learning Representations* (ICLR, 2025).

[36] M. Refinetti, A. Ingrosso, and S. Goldt, Neural networks trained with SGD learn distributions of increasing complexity, in *International Conference on Machine Learning* (PMLR, Honolulu, Hawaii, USA, 2023), pp. 28843–28863.

[37] L. Bardone and S. Goldt, Sliding down the stairs: How correlated latent variables accelerate learning with neural networks, PMLR **235**, 3024 (2024).

[38] R. Rende, F. Gerace, A. Laio, and S. Goldt, A distributional simplicity bias in the learning dynamics of transformers, in *Advances in Neural Information Processing Systems* (Curran Associates, Inc., 2024), pp. 96207–96228.

[39] A. Svete and R. Cotterell, Transformers can represent *n*-gram language models, in *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)* (Association for Computational Linguistics, Mexico, 2024), pp. 6845–6881.

[40] T. Nguyen, Understanding transformers via *n*-gram statistics, in *The Thirty-eighth Annual Conference on Neural Information Processing Systems* (Cambridge University Press, Vancouver, Canada, 2024).

[41] A. Svete, N. Borenstein, M. Zhou, I. Augenstein, and R. Cotterell, Can transformers learn *n*-gram language models? in *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing* (Association for Computational Linguistics, Miami, Florida, USA, 2024), pp. 9851–9867.

[42] N. Borenstein, A. Svete, R. Chan, J. Valvoda, F. Nowak, I. Augenstein, E. Chodroff, and R. Cotterell, What languages are easy to language-model? A perspective from learning probabilistic regular languages, in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (Association for Computational Linguistics, Bangkok, Thailand, 2024), pp. 15115–15134.

[43] A. S. Shai, S. E. Marzen, L. Teixeira, A. G. Oldenziel, and P. M. Riechers, Transformers represent belief state geometry in their residual stream, in *The Thirty-eighth Annual Conference on Neural Information Processing Systems* (NeurIPS, 2024).

[44] Z. Allen-Zhu and Y. Li, Physics of language models: Part 1, learning hierarchical language structures, arXiv:2305.13673.

[45] H. Zhao, A. Panigrahi, R. Ge, and S. Arora, Do transformers parse while predicting the masked word? in *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing* (Association for Computational Linguistics, Singapore, 2023), pp. 16513–16542.

[46] R. T. McCoy, R. Frank, and T. Linzen, Does syntax need to grow on trees? Sources of hierarchical inductive bias in sequence-to-sequence networks, Trans. Assoc. Comput. Linguis. **8**, 125 (2020).

[47] K. Ahuja, V. Balachandran, M. Panwar, T. He, N. A. Smith, N. Goyal, and Y. Tsvetkov, Learning syntax without planting trees: Understanding hierarchical generalization in transformers, Trans. Assoc. Comput. Linguist. **13**, 121 (2025).

[48] G. Rozenberg and A. Salomaa, *Handbook of Formal Languages* (Springer, Berlin, Germany, 1997).

[49] G. K. Pullum and G. Gazdar, Natural languages and context-free languages, Linguist. Philos. **4**, 471 (1982).

[50] A. K. Joshi, Tree adjoining grammars: How much context-sensitivity is required to provide reasonable structural descriptions? in *Natural Language Parsing: Psychological, Computational, and Theoretical Perspectives*, edited by D. R. Dowty, L. Karttunen, and A. M. Zwicky (Cambridge University Press, Cambridge, UK, 1985), pp. 206–250.

[51] C. D. Manning and H. Schütze, *Foundations of Statistical Natural Language Processing* (MIT Press, Cambridge, MA, 1999).

[52] S.-C. Zhu and D. Mumford, A stochastic grammar of images, Found. Trends Comput. Graph. Vision **2**, 259 (2006).

[53] W. Ebeling and T. Pöschel, Entropy and long-range correlations in literary english, Europhys. Lett. **26**, 241 (1994).

[54] H. W. Lin and M. Tegmark, Critical behavior in physics and probabilistic formal languages, Entropy **19**, 299 (2017).

[55] M. Mezard and A. Montanari, *Information, Physics, and Computation* (Oxford University Press, Oxford, UK, 2009).

[56] D. J. Hsu, S. M. Kakade, and P. Liang, Identifiability and unmixing of latent parse trees, in *Advances in Neural Information Processing Systems (NeurIPS)*, edited by F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger (Curran Associates, Inc., Nevada, USA, 2012), pp. 37–45.

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

[58] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, BERT: Pre-training of deep bidirectional transformers for language understanding, in *North American Chapter of the Association for Computational Linguistics* (Association for Computational Linguistics (ACL), Minneapolis, Minnesota, USA, 2019).

[59] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, Improving language understanding with unsupervised learning, Technical report, OpenAI, 2018.

[60] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, Backpropagation applied to handwritten zip code recognition, Neural Comput. **1**, 541 (1989).

[61] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, A convolutional neural network for modeling sentences, in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (Association for Computational Linguistics, Baltimore, Maryland, 2014), pp. 655–665.

[62] Y. Kim, Convolutional neural networks for sentence classification proceedings of the 2014 conference on empirical methods in natural language processing, EMNLP 2014, in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (Association for Computational Linguistics, Doha, Qatar, 2014), pp. 1746–1751.

[63] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, Attention is all you need, in *Advances in Neural Information Processing Systems* (Curran Associates, Inc., 2017), Vol. 30.

[64] F. Cagnetta, Scaling laws and representation learning in simple hierarchical languages: Transformers vs. convolutional architectures, Zenodo (2025), https://doi.org/10.5281/zenodo.17683931.