A Perspective: Sparse Compositionality and Efficiently Computable Intelligence

Tomaso Poggio

Center for Brains, Minds, and Machines, CSAIL, McGovern Institute, MIT

January 18, 2026

We argue that sparse compositionality is a fundamental structural property of functions that can be computed or learned efficiently by digital systems. Any function that is efficiently computable in the Church–Turing sense admits a representation as a composition of low-arity constituent functions arranged in a bounded-fan-in directed acyclic graph, and can therefore be approximated by a deep network with sparse connectivity. This principle provides a unified explanation for why deep learning avoids the curse of dimensionality, why optimization remains tractable despite nonconvexity, and why generalization often exceeds classical capacity-based predictions. Because all artificial intelligence systems are implemented on digital computers, the theory applies fully to machine learning. In contrast, it is not known whether all components of biological intelligence are efficiently computable, even if they are Turing computable in principle. This distinction suggests that some higher cognitive functions, such as language and abstraction, may be more accessible to computational modeling than evolutionarily older processes related to affect, motivation, or internal bodily states. Sparse compositionality thus clarifies both the power and the limits of machine intelligence, and points toward a theory of what can be learned efficiently from data.

Deep neural networks achieve remarkable performance across diverse domains, yet the nature of the functions they can represent and learn remains only partially understood. Here we argue that a single structural property—*compositional sparsity*—characterizes essentially all functions that can, in practice, be computed, simulated, or analyzed by a digital computer, as well as those that can be learned from data.

A function is compositionally sparse if it factors through a directed acyclic graph (DAG) of intermediate variables with bounded local fan-in, independent of the overall input dimension. Under the standard Church–Turing viewpoint, "practically computable" means computable by a Turing machine in polynomial time. Every such function admits a representation by a bounded-fan-in compositional DAG and, consequently—as proven in Mhaskar and Poggio[12]—by a deep network with sparse connectivity.

Compositional sparsity thus provides a common structural explanation for three central properties of modern learning systems: (1) avoidance of the curse of dimensionality in approximation, (2) tractable optimization through locality of dependencies, and (3) improved generalization through

reduced effective dimensionality. This perspective links practical computability, approximation theory, and statistical learning, suggesting that compositional sparsity is a fundamental and unifying principle of machine learning.[13;14]

Deep networks approximate and learn high-dimensional functions that arise in vision, language, and scientific modeling. Surprisingly, they appear to overcome the so-called *curse of dimensionality*: despite operating on inputs with millions of dimensions—such as the function defining object classification in CIFAR images—they perform accurate learning and generalization without an explosion in the number of their weights. Their empirical success raises a foundational question: what structural property must a function possess to be realizable and learnable in practice by a deep network and thus by a computer?

Recent theoretical work suggests that the answer lies in compositional sparsity.[4;13;14] Functions that can be computed by a Turing machine within polynomial time—and thus by any realistic digital computer— admit representations as bounded-fan-in compositions of simpler subfunctions. This structure corresponds to a directed acyclic graph (DAG) whose nodes represent intermediate variables and whose local connectivity does not grow with input dimension.

In this view, the architectures of deep networks are not arbitrary engineering choices. Rather, they naturally reflect the sparse compositional structure of those functions that can be computed or learned in practice. The correspondence between compositional sparsity and practical computability provides a bridge between Turing complexity and modern learning theory. Following standard usage in theoretical computer science, we call a function *efficiently computable* if it can be computed by a Turing machine in polynomial time; in practice, this coincides with what can be computed by an ordinary digital computer. The key notions are summarized below.

1 Efficient Computability and Sparse Compositionality

A central observation is that any function f that is efficiently computable, in the sense of Box 1, can be represented, to arbitrary precision, by a circuit of polynomial size and bounded fan-in. Translating this into continuous mathematics yields a hierarchy of compositions of low-arity functions. The resulting function class—compositionally sparse functions—is exponentially smaller than the space of all continuous functions on high-dimensional domains, yet it contains those that can be evaluated or learned in practice.[13;14]

Formally, a function $f(x_1, \ldots, x_n)$ is compositionally sparse if there exists a DAG G with finite depth L and fan-in $k = O(1)$ such that each node computes a function h_i of at most k inputs from its predecessors and f equals the output at the final node. This structure corresponds to a layered composition

$$f(x) = h^{(L)} \circ h^{(L-1)} \circ \cdots \circ h^{(1)}(x),$$

where each $h^{(\ell)}$ acts on only a few variables. Such sparsity drastically reduces the number of parameters required to approximate f and constrains the space of admissible dependencies. This exponential explosion of parameters with input dimension—known as the curse of dimensionality—is summarized in Box 1.

Box 1 – Key definitions

Efficiently computable. A function f is said to be *efficiently computable* if there exists an algorithm on a standard digital computer—or equivalently, a Turing machine—that, given an input x encoded in n bits and a desired output precision ε (encoded in $\lceil \log_2(1/\varepsilon) \rceil$ bits), computes an ε-approximation to $f(x)$ in time $\mathrm{poly}(n, \log(1/\varepsilon))$. In theoretical computer science, this marks the boundary between functions that can be computed in practice and those that cannot. Throughout this article, we treat "efficiently computable" and "practically computable" as synonymous.

Compositionally sparse function. A function $f(x_1, \ldots, x_n)$ is *compositionally sparse* if it can be expressed as a composition of low-arity constituent functions arranged in a directed acyclic graph (DAG) of fixed depth. Formally, there exists a DAG G with depth L such that each node computes a function h_i that depends on at most k of its input coordinates, where k is a constant independent of n, and all other coordinates are passed forward unchanged. Thus each layer $h^{(\ell)} : \mathbb{R}^n \to \mathbb{R}^n$ modifies at most k coordinates of its input, with identity mappings on the remaining coordinates. The full function is given by

$$f(x) = h^{(L)} \circ h^{(L-1)} \circ \cdots \circ h^{(1)}(x).$$

This formulation avoids the implicit "deletion" of unused variables and corresponds exactly to the standard bounded-fan-in circuit model.

Curse of dimensionality. For a generic smooth function $f : \mathbb{R}^n \to \mathbb{R}$, the number of parameters $N(\varepsilon)$ required to approximate f with uniform error ε by standard (non-compositional) methods typically scales as

$$N(\varepsilon) \propto \varepsilon^{-n/s},$$

where s measures smoothness (for example, the number of bounded derivatives). When n is large, this dependence becomes prohibitive: for $n = 1000$ and $\varepsilon = 0.1$ with $s = 1$, one obtains $N \sim 10^{1000}$ parameters. This exponential growth in required resources is the classical *curse of dimensionality*.

2 Approximation, Optimization, and Generalization

Compositional sparsity leads to quantitative advantages in all major aspects of learning:

1. **Approximation.** For functions with compositional depth L and arity k, deep networks achieve error $O(N^{-\alpha/k})$ using $O(N)$ parameters, while shallow or dense architectures require exponentially more.[1;12;13]

2. **Optimization.** Because of the non-exponential number of parameters, gradient-based optimization avoids the curse of dimensionality. Note that while the target function is compositionally sparse, finding this structure via gradient descent often requires highly overparameterized, dense networks during training. This provides the optimization landscape necessary to locate the sparse solution, even if the final effective dimensionality of the learned function remains low.

3. **Generalization.** The effective dimensionality of the hypothesis space scales with the number of parameters for the "worst" subfunctions rather than the total number of parameters, yielding tighter bounds on sample complexity.[2;5;10;16;18]

These effects emerge naturally from the DAG structure and do not depend on the specific activation or architecture details. The corresponding formal results are summarized in Box 2.

3 Relation to Prior Theories of Deep Learning

The principle of compositional sparsity extends and unifies several earlier theoretical accounts of deep learning. Classical approximation theory established that deep networks can represent certain function classes with exponentially fewer parameters than shallow networks.[1;12] The present framework identifies the source of this efficiency: compositional structure limits the effective dimensionality of the function space. Similarly, circuit complexity results[6] demonstrate that functions computable by small-depth, bounded-fan-in circuits correspond to hierarchically compositional functions. Recent analyses in statistical learning[3;15] emphasized hierarchical feature composition, but without connecting it to formal notions of practical computability or sample efficiency.

Compositional sparsity provides this missing link, showing that the same structural property explains advantages in representation, optimization, and generalization within a unified mathematical framework. The theorems above suggest that modern large-scale architectures benefit precisely because they learn and compose local constituents. Two prominent examples are transformer-based sequence models and diffusion generative models.[7;17;19]

Implication for transformer MLP blocks (falsifiable prediction). If efficient computability implies compositional sparsity, then the *feed-forward (MLP) sublayers* in a transformer must implement *low-arity constituents*. Concretely, for each layer there exist permutations of coordinates under which the two linear maps $(W_{\mathrm{in}}, W_{\mathrm{out}})$ of the MLP can be written (up to an ε-approximation error) as *block-sparse/column-sparse* matrices with at most $O(k)$ nonzeros per column/row, where k is the constituent arity and is independent of the embedding dimension. Equivalently, the effective ℓ_0 (and hence ℓ_1) sparsity of the MLP weight matrices scales with k rather than with width. This yields a concrete, testable consequence: at fixed accuracy, one should be able to prune the MLP weights down to a sparsity level proportional to k (up to logarithmic factors), with minimal fine-tuning, and the resulting norm-based generalization bounds (Box 2, Theorem C) should tighten accordingly.[2;5;10] In short, *efficient computability predicts intrinsic sparsity of transformer MLP weights*, reflecting the low-arity structure of the underlying constituents. This prediction is consistent with emerging empirical evidence suggesting that large language models can be pruned to extreme levels of sparsity—up to 99% in some regimes—without significant performance degradation, provided the pruning aligns with the underlying compositional structure.

Box 2 – Main results on sparse compositionality

Theorem A (Mhaskar & Poggio, informal). *Sparse compositional functions escape the curse of dimensionality.*
If $f(x_1, \ldots, x_n)$ is a *compositionally sparse* function built from constituent functions in a Sobolev space $\mathbf{W}_s^k$, each depending on at most k variables (independent of n), then a deep network whose architecture mirrors this compositional structure can approximate f with error ε using $N_{\text{deep}}(\varepsilon) \propto \varepsilon^{-k/s}$ parameters, where s depends on the smoothness of the constituents. In contrast, shallow or dense architectures typically require $N_{\text{shallow}}(\varepsilon) \propto \varepsilon^{-\frac{n}{s}}$ parameters. Compositional sparsity thus removes the exponential dependence on n and breaks the curse of dimensionality.[12]

Theorem B (informal, see Poggio[14]). *Efficient computability implies sparse compositionality.*
Any efficiently computable function admits a representation as a directed acyclic graph of bounded-fan-in constituent functions of polynomial size. Equivalently, every efficiently computable function can be approximated, to arbitrary precision, by a deep network whose connectivity is sparse and locally bounded.

Scope of the result. Following Poggio,[14] we adopt the standard bit model of computation: an "efficiently computable" function $f : [0,1]^d \to \mathbb{R}^m$ is one for which there exists a deterministic Turing machine that, given an n-bit encoding of x and an accuracy parameter $\varepsilon > 0$ (encoded in $\lceil \log_2(1/\varepsilon) \rceil$ bits), runs in time $\text{poly}(n, \log(1/\varepsilon))$ and outputs an ε-approximation to $f(x)$. In Poggio[14] it is shown that such a machine can be unfolded into a family of polynomial-size, bounded-fan-in circuits that, in turn, can be compiled into deep neural networks with sparse connectivity and size and depth polynomial in $n + \log(1/\varepsilon)$.

Theorem C (informal, see Galanti et al.[5]). *Sparsity of the network's weight matrices implies better generalization bounds.*
For a deep network whose connectivity matches a compositionally sparse DAG, the empirical Rademacher bounds $\mathfrak{R}_S(\mathcal{F})$ is are significantly smaller than the standard Rademacher-based bounds for dense networks.[2;10;16;18] For example, if a layer is convolutional and the connectivity is equivalent to non-overlapping patches, that is, each node in the DAG receives different inputs, its contribution to the Rademacher complexity is a factor $\frac{1}{\sqrt{N}}$ smaller when sparsity is taken into account than in standard bounds.

Theorem D (informal; see Poggio[14] and Malach[?]). *Efficient computability implies constituent learnability.*
For efficiently computable functions, the constituent functions are sparse and thus each of them is efficiently learnable from polynomially many examples. Moreover, the overall function can be learned by *local* training of these constituents, using data that only need to expose the relevant low-arity dependencies (as in next-token prediction or local denoising objectives). Efficient computability of the composite function implies *local learnability of its parts thus yielding practical learnability of the composite function*.

4 Discussion

Compositional sparsity offers a structural characterization of the class of functions that can be computed or learned in practice by digital computers. Under the Church–Turing viewpoint, every efficiently (and thus practically) computable function is compositionally sparse and can therefore be represented by a deep network with bounded-fan-in connectivity.[4;13;14] This perspective unifies several empirical and theoretical observations: that deep architectures overcome the curse of dimensionality, that their optimization is tractable despite nonconvexity, and that their generalization performance exceeds what classical capacity measures would predict.[1;2;5;10;12;16;18]

For artificial intelligence systems, the scope of this theory is clear. All AI algorithms are implemented on digital computers and are therefore efficiently computable by construction. As a consequence, the theory developed here applies fully to artificial learning systems: any function realized by an AI model must admit a compositionally sparse representation, even if that structure is not explicit in the trained architecture. From this perspective, the success of modern foundational models is not accidental but reflects their ability to approximate and compose low-arity constituent functions drawn from this restricted, efficiently computable function class.

The situation is more subtle for biological intelligence. A crucial distinction in this context is between *Turing computability* and *efficient* Turing computability. The physical Church–Turing thesis asserts that the behavior of any physically realizable system can, in principle, be simulated by a Turing machine. However, this does not imply that such a simulation can be carried out with resources that scale polynomially in the relevant problem size. Many physical and biological systems may therefore be fully Turing computable yet computationally intractable, requiring exponential time or space to simulate with useful precision.

From this perspective, it is not obvious that all functions implemented by the brain are efficiently computable in the sense adopted here. Higher-level cognitive functions such as language, abstraction, and reasoning exhibit clear signatures of hierarchical and compositional structure and are natural candidates for efficient computation. In contrast, other processes—such as affect, emotions, drives, homeostatic regulation, and internal bodily states—may rely on tightly coupled, state-dependent, or continuous dynamics that do not admit an efficient sparse compositional representation. Such processes may remain fully Turing computable, yet fall outside the class of functions that can be approximated or learned efficiently by digital systems.

This distinction suggests an apparent but instructive irony. It may be easier to replicate and understand higher-level cognitive functions using digital computers than more basic, evolutionarily older ones. The reason is not that language or reasoning are simpler in an absolute sense, but that they may be more structured, more compressible, and more amenable to sparse compositional representations. Conversely, functions that are behaviorally simple or automatic may nonetheless be computationally inefficient, poorly modularized, or resistant to decomposition into reusable constituents.

Importantly, this perspective does not imply that such functions are non-computable or beyond physical realization. Rather, it highlights a distinction between efficient computability and mere computability, and between learnability from data and implementation by embodied, closed-loop biological systems. From this viewpoint, the current strengths and limitations of AI systems are not paradoxical but follow naturally from the structural constraints imposed by efficient computation.

Ultimately, sparse compositionality points toward a unifying theory of machine learning and computation—one that explains what digital systems can learn well, what they struggle with, and

why. Whether sparse compositionality also captures the full range of biological intelligence remains an open question, but one that can now be stated with greater precision.

Box 3 – Modern architectures as realizations of efficiently computable

Scope clarification. The interpretations in this box concern those aspects of modern learning architectures that correspond to *efficiently computable* functions. Since all artificial intelligence systems are implemented on digital computers, the functions they realize necessarily lie within the class of efficiently computable—and hence compositionally sparse—mappings. The discussion below does not claim that such architectures capture all aspects of biological intelligence, but illustrates how sparse compositionality manifests in current large-scale models.

Informal principle. Contemporary large-scale models, including transformers and diffusion networks, can be interpreted as concrete realizations of compositionally sparse functions, in the sense that their global input–output behavior arises from the composition of locally learnable, low-arity constituent functions.

Transformers. A transformer implements a hierarchy of low-arity constituent functions operating on token embeddings. Multi-head attention computes context vectors c_t from bounded subsets of positions or features, and the output heads map c_t to conditional probabilities $p(x_t \mid c_t)$. Training by next-token prediction,

$$\min_{\theta} \mathbb{E}[-\log p_\theta(x_t \mid x_{<t})] \,,$$

corresponds to fitting local conditional constituents.[?] The global mapping from prompts to continuations can therefore be viewed as a composition of locally learnable subfunctions, accounting for the emergence of coherent text generation within the class of efficiently computable mappings.[13;14]

Diffusion models. Diffusion generative models construct samples by composing a sequence of local denoising steps.[7;17] Let $(X_t)_{t=0}^{T}$ be a Markov chain with known forward kernels $q(x_t \mid x_{t-1})$ and learnable reverse kernels $p_\theta(x_{t-1} \mid x_t)$. Each reverse step acts on a low-dimensional state (x_t, t) and can be represented by a constituent neural function in a compositional DAG. Training objectives such as denoising score matching fit these local transitions; their composition yields the global generator

$$x_T = z \mapsto x_{T-1} \mapsto \cdots \mapsto x_0 = G_\theta(z).$$

Both architectures illustrate the same principle: high-dimensional predictive or generative behavior can arise from the composition of locally learnable, low-arity constituents, consistent with the constraints imposed by efficient computation and sparse compositionality.[4;13]

References

[1] Barron, A. (1993). Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information Theory*, 39(3):930–945.

[2] Bartlett, P. L. and Mendelson, S. (2002). Rademacher and Gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3:463–482.

[3] Bengio, Y., Courville, A., and Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828.

[4] Danhofer, D. A., D'Ascenzo, D., Dubach, R., and Poggio, T. A. (2025). Position: A Theory of Deep Learning Must Include Compositional Sparsity. In *Proceedings of the 42nd International Conference on Machine Learning (ICML 2025)*. Also available as arXiv:2507.02550.

[5] Galanti, T., Xu, M., Galanti, L., and Poggio, T. (2023). Norm-based generalization bounds for sparse neural networks. In *Advances in Neural Information Processing Systems*, Vol. 36, pp. 42482–42501.

[6] Håstad, J. (1986). Almost optimal lower bounds for small depth circuits. In *Proceedings of the 18th ACM Symposium on Theory of Computing*, 6–20.

[7] Ho, J., Jain, A., and Abbeel, P. (2020). Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, 33:6840–6851.

[8] Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79(8):2554–2558.

[9] Kanerva, P. (1988). *Sparse Distributed Memory*. MIT Press.

[10] Koltchinskii, V. (2001). Rademacher penalties and structural risk minimization. *IEEE Transactions on Information Theory*, 47(5):1902–1914.

[11] Malach, E. (2023). Auto-regressive next-token predictors are universal learners. *arXiv preprint arXiv:2309.06979*.

[12] Mhaskar, H., Liao, Q., and Poggio, T. (2016). Learning functions: when is deep better than shallow. *Annals of Mathematics and Artificial Intelligence*, 86(1):1–26.

[13] Poggio, T. and Fraser, M. (2024). Compositional sparsity of learnable functions. *Bulletin of the American Mathematical Society*, 61(3):438–456.

[14] Poggio, T. (2025). On Efficiently Computable Functions, Deep Networks, and Sparse Compositionality. CBMM Memo / Preprint.

[15] Poole, B., Lahiri, S., Raghu, M, Sohl-Dickstein, J., and Ganguli, S. (2016). Exponential expressivity in deep neural networks through transient chaos. In *Advances in Neural Information Processing Systems*, 3360–3368.

[16] Shalev-Shwartz, S. and Ben-David, S. (2014). *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press.

[17] Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. (2015). Deep unsupervised learning using nonequilibrium thermodynamics. In *Proceedings of the 32nd International Conference on Machine Learning*, 2256–2265.

[18] Vapnik, V. N. (1998). *Statistical Learning Theory*. Wiley.

[19] Vaswani, A., Shazeer, N., Parmar, N. et al. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems*, 5998–6008.