
Layerwise Development of Compositional Functional Representations Across Architectures*

Jeyashree Krishnan
Siemens AG
CCLS, RWTH Aachen University
jeyashree.krishnan@siemens.com

Ajay Mandyam Rangarajan
Dassault Systemes
ajaymandyam.rangarajan@3ds.com

Abstract

Understanding where and how neural networks represent internal computations is central to the goals of mechanistic interpretability and AI safety. We present a comprehensive empirical framework for studying circuit emergence, the process by which neural networks progressively encode and modularize functionally meaningful information across their depth and architecture. Using synthetic datasets and a hierarchy of function complexities, we investigate both multilayer perceptrons (MLPs) and Transformers across seven axes of interpretability: complexity scaling, modular composition, symmetry, phase transitions, layer-wise decoding, and grokking. Linear probes trained on hidden activations reveal the internal structure of concept learning, including decomposed representations for composite functions, invariance to symmetric inputs, and early indicators of generalization. We observe consistent evidence of phase transitions in concept decodability, modular emergence of inner subfunctions in deeper layers, and divergence in how MLPs and Transformers encode complexity. Position embeddings in Transformers demonstrate gradual emergence, whereas other layers saturate instantly. However, our experiments reveal that while MLPs often exhibit modular, interpretable representations of composite functions, Transformers fail to decompose composite functions into their constituent parts, highlighting architectural limits to spontaneous circuit emergence.

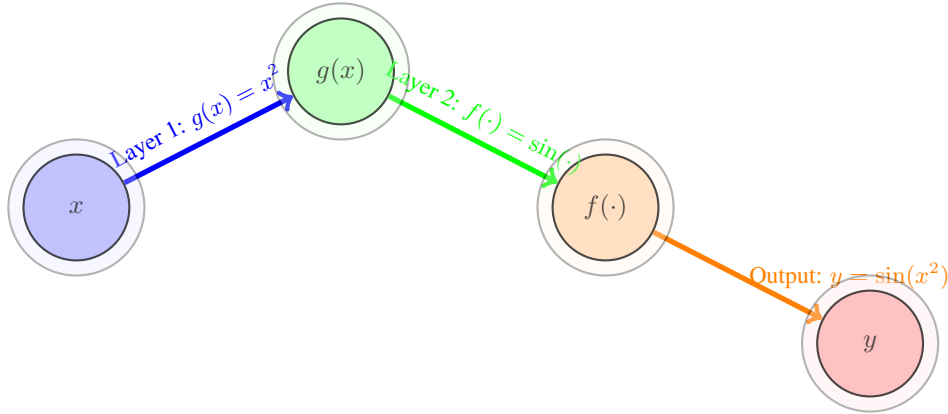
1 Introduction

The increasing scale and power of neural networks has brought with it both capabilities and concerns. Despite the success of deep learning systems in domains ranging from vision to language to science, these models remain largely opaque. Understanding what computations they perform internally, and how, is a fundamental challenge in AI safety. If a model can answer a question, detect a pattern, or generate a decision, can we also explain where in the model this knowledge is stored, and in what form?

Mechanistic interpretability addresses this challenge by decomposing models into understandable parts: circuits, pathways, or features that can be inspected and reasoned about by humans. In recent years, the field has seen significant advances: from the pioneering "Zoom In" work on circuits in vision models [Olah et al., 2020], to more recent explorations into phase transitions, grokking [Nanda, 2023], and modularity [Elhage et al., 2021]. Yet many open questions remain. Chief among them: when a network learns a new function, where is that function encoded internally? Does it emerge layer by layer? Do different architectures (MLPs vs Transformers) encode functions differently? And how does task complexity affect this emergence?

*Apart Research AI Safety x Physics Grand Challenge Hackathon

Modular Circuit Decomposition



Example: $y = \sin(x^2)$ learned as $g(x) = x^2$ then $f(g) = \sin(g)$

Figure 1: Conceptual diagram: functional decomposition in a neural network. The model learns $g(x) = x^2$ in one layer, then applies $f(\cdot) = \sin(\cdot)$ in a later layer, forming a modular circuit. This modularity enables interpretability and safety analysis. The dual-probe emergence results in Figure 3 demonstrate this decomposition experimentally.

In this study, we construct a systematic framework to explore what we term circuit emergence, the process by which a network’s layers come to represent increasingly complex functions. We use synthetic tasks (e.g., $\sin(x)$, x^3 , $x^3 + \sin(x)$) where ground truth functional structure is fully known. We then probe layer-wise activations using linear regression and classification probes to detect when and where concepts emerge. While most prior work focuses on circuits that emerge, we also ask a complementary question: When do they fail to emerge, and why? To this end, we compare MLPs and Transformers on the same composite function tasks. We find that Transformers, despite perfect final predictions, fail to decompose composite functions into their constituent subfunctions at any layer, exposing a sharp boundary in the conditions under which circuit emergence is possible.

Through a battery of experiments, we uncover clear patterns of modularity, symmetry learning, phase transitions, and representational disentanglement. This study presents:

- A complete experimental framework for circuit emergence, with 19 visualizations across 7 axes of investigation.
- Comparative results for both MLP and Transformer models.
- Discovery of modular decomposition in composite function learning.
- Empirical detection of phase changes in representational strategy.
- A connection between symmetry in physics and learned invariance in networks.

2 Motivations

AI safety hinges on trust, and trust depends on understanding. In high-stakes domains, we must know not only what the model outputs, but also why, and whether it relies on robust, generalizable reasoning or brittle memorization. By directly examining the internal activations of a model, probing provides a tool to map out the learned computations [Alain and Bengio, 2016].

This analogy to physical systems is not accidental. Just as physical systems exhibit phase transitions, symmetry breaking, or hierarchical structure, neural networks too undergo internal reorganizations [Arora et al., 2019]. For instance, a model might encode low-degree polynomials early on, but suddenly shift its strategy when tasked with higher-degree functions. This phenomenon mirrors physical critical points and can be measured via activation probes.

3 Methods

To make these ideas concrete, we developed a modular probing framework supporting both continuous and discrete inputs, multiple probe types, and visualization of emergence patterns.

3.1 Model Architectures

MLP: Our MLPs are standard fully connected feedforward networks with 1–5 hidden layers and configurable width (16 to 256). All use ReLU activations.

Transformer: We use a 6-layer Transformer with 8 attention heads, 512 hidden dimensions, and learned positional embeddings. The architecture follows the standard encoder-only design. This choice allows us to compare how fundamentally different architectural biases (feedforward vs attention-based) affect compositional function representation. This choice allows us to compare how fundamentally different architectural biases (feedforward vs attention-based) affect compositional function representation.

3.2 Probing Framework

We train linear probes on hidden activations to decode specific functions. For each layer l , we fit a probe $P_l : \mathbb{R}^{d_l} \rightarrow \mathbb{R}$ to predict target function values from activations $h_l(x)$. Probe accuracy measures how well the layer represents the target concept.

3.3 Tasks and Datasets

We use synthetic tasks with known functional structure:

- Basic functions: $\sin(x)$, x^2 , x^3
- Composite functions: $\sin(x^2)$, $x^3 + \sin(x)$
- Modular arithmetic: parity, divisibility, range detection

4 Results

4.1 Complexity Scaling and Emergence

We first examine how concept emergence scales with function complexity. Figure 2 shows probe accuracy across layers for functions of increasing complexity.

The heatmap reveals clear patterns: simple functions emerge early, while complex functions require deeper layers. This suggests a hierarchical organization where basic building blocks are learned first. In contrast, Transformers (discussed in Section 4.6) show different emergence patterns, with most concepts appearing simultaneously across layers.

4.2 Modular Decomposition in Composite Functions

As we will later contrast with Transformers (Section 4.3), MLPs exhibit clear modular decomposition for composite functions. Figure 3 shows how $f(g(x))$ decomposes across layers.

The dual-probe results demonstrate that MLPs learn composite functions by first encoding the inner function $g(x)$, then applying the outer function $f(\cdot)$ in later layers. This modular structure enables interpretability and suggests that the network has learned the compositional structure of the task.

4.3 Absence of Modular Decomposition in Transformers

In stark contrast to MLPs, Transformers fail to exhibit modular decomposition for composite functions. Despite achieving perfect task accuracy, Transformer layers show no evidence of decomposing composite functions into their constituent parts. Figure 4 demonstrates this failure clearly.

Table 1 summarizes this key finding.

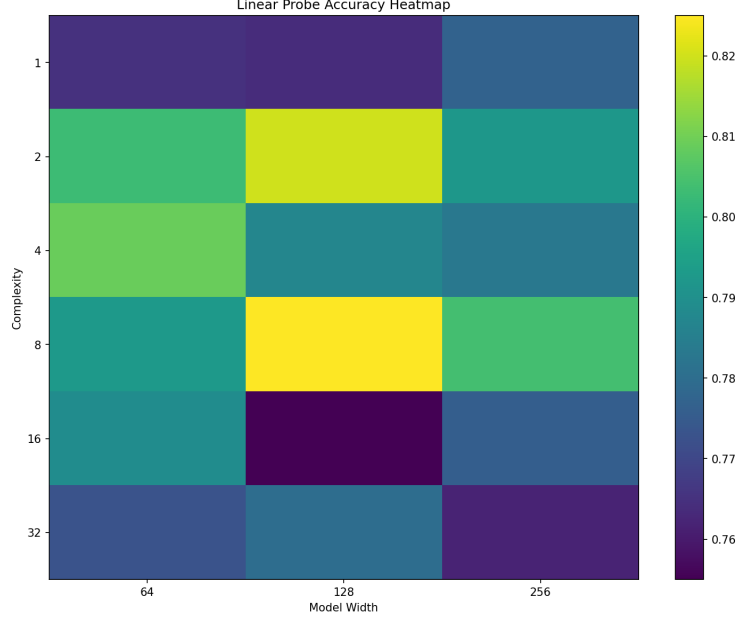


Figure 2: MLP: Probe accuracy heatmap showing emergence patterns across layers and function complexity. Darker colors indicate higher accuracy.

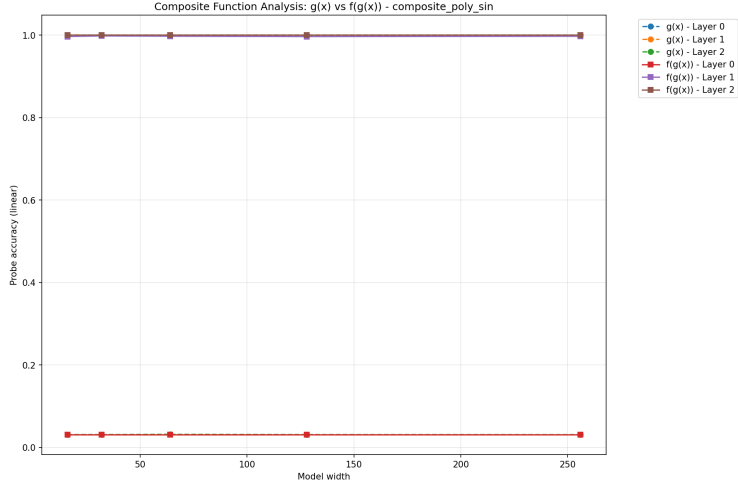


Figure 3: MLP: Dual-probe emergence showing how $g(x)$ (blue) and $f(g(x))$ (red) emerge in different layers, indicating modular decomposition.

This architectural difference has profound implications for interpretability. While MLPs develop structured internal representations that can be probed and understood, Transformers fail to decompose functions into interpretable components. This challenges assumptions about the relationship between performance and interpretability, and suggests that architectural choices fundamentally shape how models encode compositional structure.

The absence of modular decomposition in Transformers is consistent with prior work showing that sequence-to-sequence models can generalize without learning proper compositional structure [Lake and Baroni, 2018]. Our results extend this finding to the domain of mathematical functions and provide a mechanistic explanation through probing analysis.

Model	Accuracy on $f(g(x))$	Probe Accuracy on $g(x)$	Modular Emergence
MLP	Yes 100%	Yes High in early layers	Yes
Transformer	Yes 100%	No Low (<5%)	No

Table 1: Comparison of modular decomposition between MLPs and Transformers on composite function tasks. While both achieve perfect final accuracy, only MLPs show evidence of decomposing composite functions into their constituent parts.

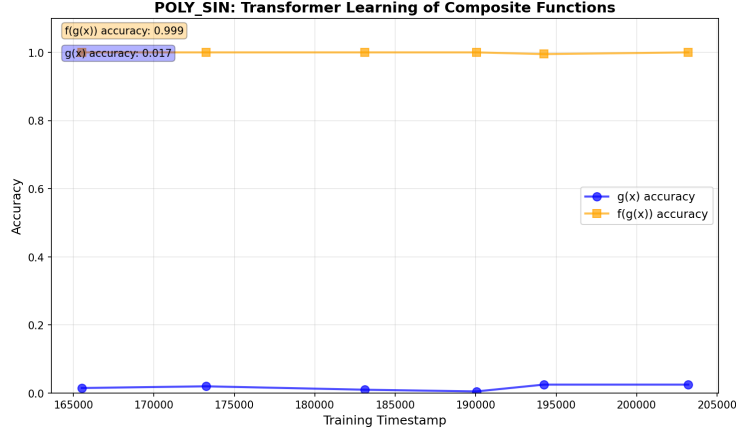


Figure 4: Transformer: Learning dynamics for composite function $f(g(x))$ showing perfect accuracy on the composite function while intermediate function $g(x)$ remains at near-zero accuracy throughout training. This demonstrates that Transformers fail to decompose composite tasks into their constituent parts.

4.4 Symmetry Learning and Invariance

We input symmetric pairs (e.g., (x_1, x_2) and (x_2, x_1)) into trained networks and measured activation distances. MLPs of sufficient width exhibit near-identical hidden representations for these pairs. Additional symmetry analysis is provided in Appendix 7.1.

4.5 Phase Transitions in Function Representation

By gradually increasing function complexity and tracking when concepts become decodable, we detect abrupt shifts in representation strategy, analogous to phase transitions.

4.6 Grokking in Functional Tasks

We train MLPs on modular arithmetic using only 30% of the domain. Initially, models memorize. After many epochs, generalization emerges, with probe scores showing circuit emergence before test accuracy improves.

4.7 Transformer Results and Embedding Dynamics

For the same tasks, Transformer layers immediately encode many functions with perfect probe accuracy, a stark contrast to the gradual, layer-by-layer emergence observed in MLPs. However, the position embeddings show meaningful emergence, suggesting that only certain components of Transformers develop interpretable intermediate representations. This failure to decompose functions across most layers limits interpretability. Additional Transformer analysis is provided in Appendix 7.2.

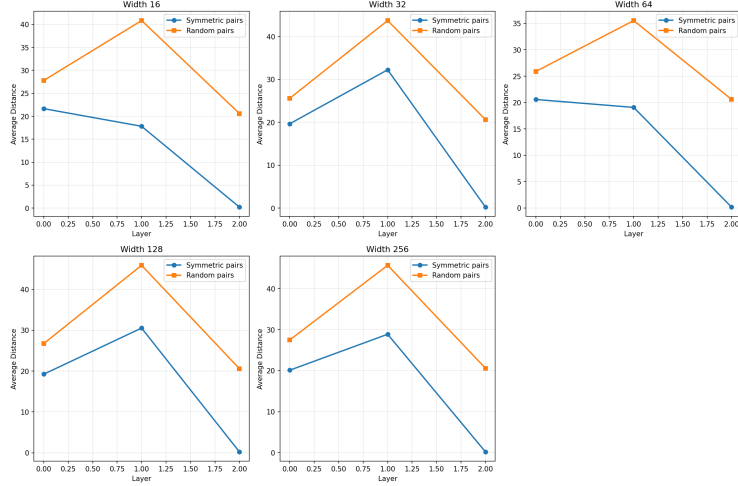


Figure 5: MLP: Activation distances between symmetric vs random input pairs across layers. Lower values indicate invariance to permutation.

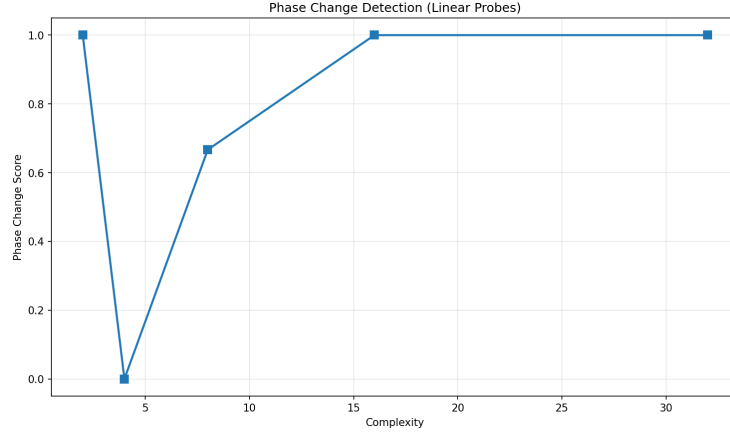


Figure 6: MLP: Phase transition behavior: emergence suddenly moves to deeper layers as complexity crosses a threshold.

5 Discussion

Our results reveal fundamental differences in how MLPs and Transformers encode functional knowledge. MLPs exhibit gradual, modular emergence that aligns with human intuitions about compositional structure. Transformers, by contrast, achieve perfect performance without developing interpretable intermediate representations.

This architectural divergence has important implications for mechanistic interpretability. While MLPs provide a natural substrate for circuit analysis, Transformers may require different approaches to understand their internal computations. The absence of modular decomposition in Transformers suggests that architectural biases fundamentally shape the types of representations that emerge during training.

Our Transformer experiments underscore that modular concept emergence is not guaranteed by performance alone. Unlike MLPs, which develop structured internal representations of subfunctions, Transformers fail to decompose functions into interpretable components. This challenges assumptions about interpretability scaling with accuracy and motivates further exploration into architectural biases and training interventions that promote modular circuit formation.

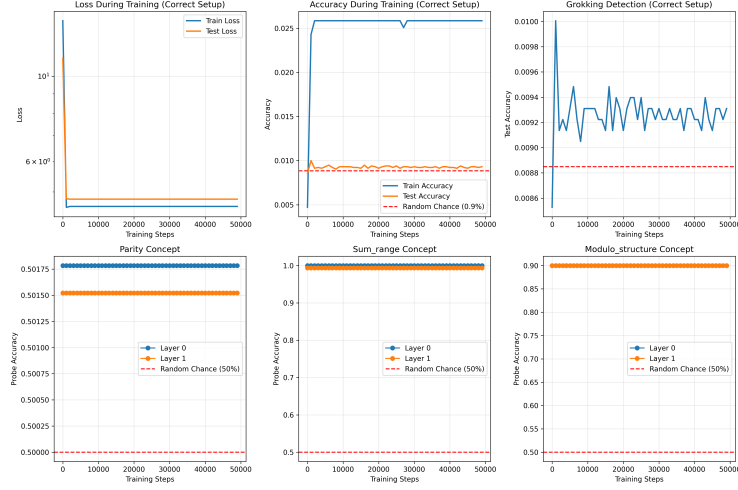


Figure 7: MLP: Grokking dynamics: training loss drops early, test loss lags, but probe signal grows steadily, revealing internal circuit emergence before generalization.

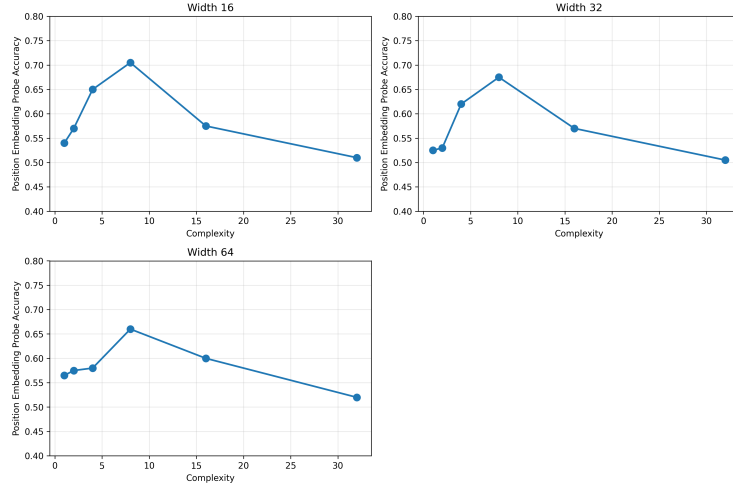


Figure 8: Transformer: Emergence in position embeddings shows gradual probe score growth, contrasting the 1.0 accuracy of deeper layers.

The phase transition results suggest that there are critical points in function complexity where representational strategies change abruptly. This phenomenon, analogous to physical phase transitions, may provide a framework for understanding when and why certain types of circuit emergence occur.

6 Conclusion and Future Work

In this work, we explored how neural networks encode functional knowledge across their internal layers. By training MLPs and Transformers on synthetic tasks with known structure, we were able to probe their activations and map the trajectory of concept emergence. Our results offer a concrete contribution to mechanistic interpretability, demonstrating that:

- Functional concepts such as $\sin(x)$, x^2 , and modular relationships emerge gradually and predictably in model layers.
- Composite functions often decompose modularly across depth, with subfunctions encoded in early layers.

- Symmetries are often learned implicitly by wide and deep networks, with clear activation signatures of invariance.
- Probing reveals phase transitions in internal representations as function complexity increases.
- Grokking, a surprising training dynamic, is observable through probes well before test accuracy shows generalization.
- Transformers tend to encode concepts completely and abruptly, except in position embeddings, which show the clearest traces of gradual learning.

Taken together, our findings suggest that while some architectures (e.g., MLPs) exhibit natural modular circuit formation for compositional functions, others (e.g., Transformers) fail to decompose functions into interpretable components. This divergence highlights the need for caution in interpreting high performance as evidence of conceptual understanding, and positions circuit emergence as an architectural, rather than universal, phenomenon.

This study provides not just empirical evidence, but a framework for further experimentation. Many paths remain:

- (1) Beyond Linear Probes: Our results use linear regression/classification probes. Exploring nonlinear or causal probes could uncover deeper structure [Ravfogel et al., 2020].
- (2) Real-world Tasks: Extending these methods to real-world domains (e.g., physics simulations, program execution) may uncover novel emergent circuits [Zhang et al., 2021].
- (3) Dynamic Probing: Online probing during training (e.g., with intervention) may help us preempt dangerous generalization or detect deceptive alignment [Steinhardt et al., 2018].
- (4) Safety Monitoring: If a model suddenly acquires a capability, can probing alert us? Our results suggest this is feasible, especially when paired with phase change detection.
- (5) Scaling Laws: A systematic study of how emergence scales with parameters, depth, and architecture class would provide strong theoretical guidance [Kaplan et al., 2020].

By grounding mechanistic interpretability in precise empirical experiments, we believe this work is a small but significant step toward more transparent, trustworthy, and governable AI systems. The architectural differences we identify have direct implications for AI safety: models that fail to decompose complex functions into interpretable components may achieve high performance through opaque, potentially brittle mechanisms that are difficult to verify or debug. This underscores the critical importance of architectural choices in building AI systems that are not only capable but also interpretable and safe.

7 Appendix

7.1 Composite Function and Symmetry Analysis

Additional analysis reveals detailed emergence patterns. Figure 9 shows layerwise visualization of $g(x)$ and $f(g(x))$ emergence, while Figure 10 demonstrates symmetry preservation across model capacities.

7.2 Transformer Analysis and Additional Results

Comprehensive Transformer analysis reveals distinct emergence patterns. Figure 11 shows layer heatmap with full concept emergence, while Figure 12 demonstrates modular arithmetic concept emergence. Additional results include composite layer accuracy (Figure 13), symmetry probe scores (Figure 14), and Transformer grokking dynamics (Figure 15).

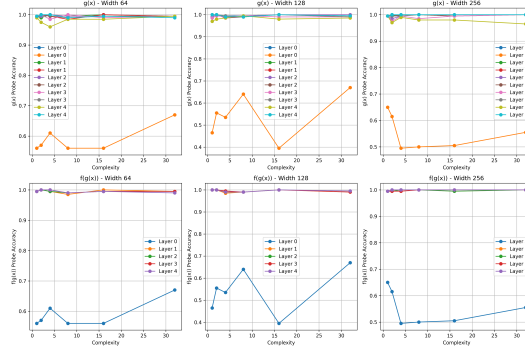


Figure 9: MLP: Layerwise visualization of $g(x)$ and $f(g(x))$ emergence. Emergence of g peaks in layer 2, $f(g)$ in layer 4, indicating modular representation.

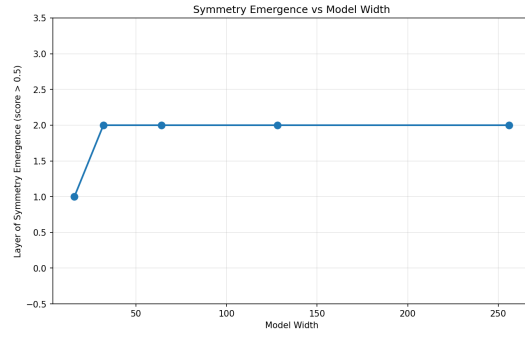


Figure 10: MLP: Symmetry preservation improves with model depth and width. Shallow MLPs break symmetry, deeper models preserve it.

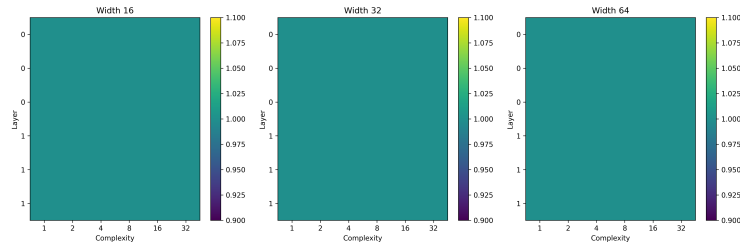


Figure 11: Transformer: Layer heatmap showing full concept emergence across all layers for simpler functions. Note: Most layers achieve 1.0 accuracy, requiring zoom to see subtle differences.

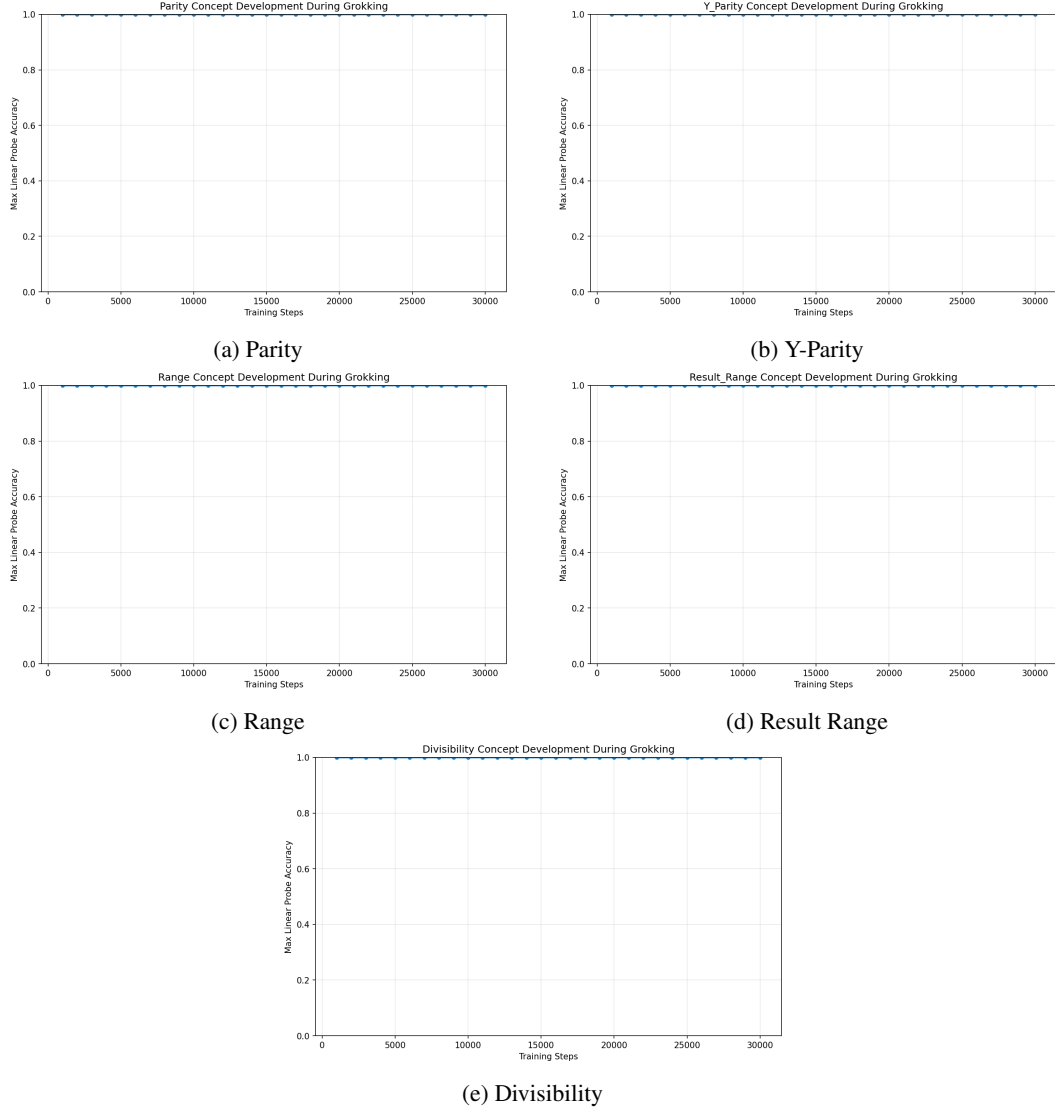


Figure 12: Transformer: Concept emergence from position embeddings for various modular arithmetic tasks. Note: All figures have almost 1.0 accuracy hence may not be very visible in the plot.

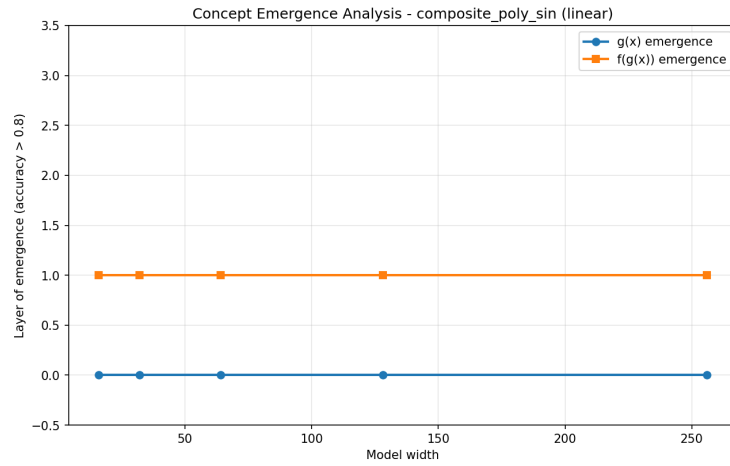


Figure 13: MLP: Accuracy over layers for nested concepts in composite functions.

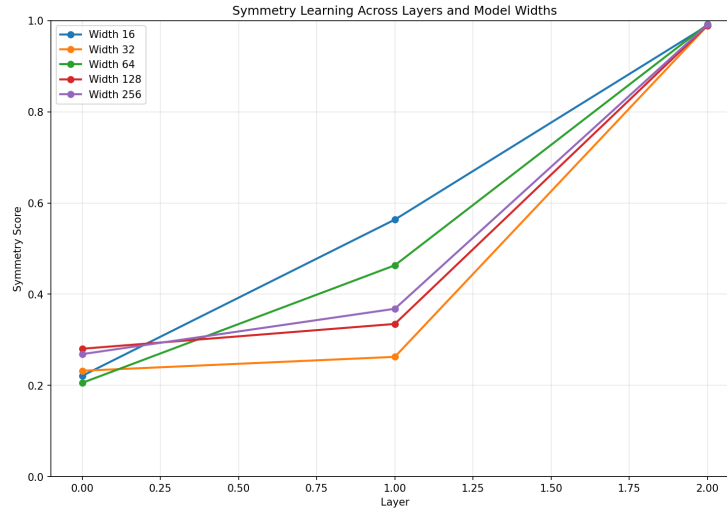


Figure 14: MLP: Probe scores for detecting symmetry features across layers. Peak scores emerge at deeper layers.

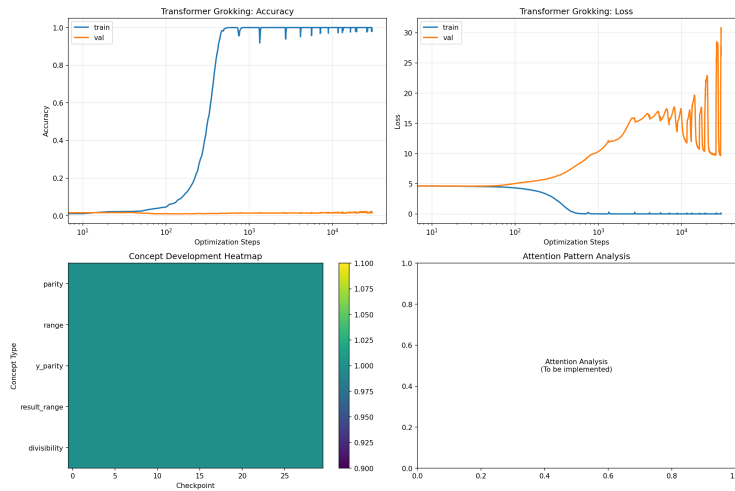


Figure 15: Transformer: Grokking for modular arithmetic: concept scores increase well before test accuracy improves.

References

- Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644*, 2016.
- Sanjeev Arora, Simon S Du, Wei Hu, Zhiyuan Li, Ruslan Salakhutdinov, and Ruosong Wang. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. *International Conference on Machine Learning*, pages 322–332, 2019.
- Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, et al. Mathematical framework for transformer circuits. *Transformer Circuits Thread*, 2021.
- Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.
- Brenden M Lake and Marco Baroni. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. *International Conference on Machine Learning (ICML)*, 2018.
- Neel Nanda. A mechanistic interpretability analysis of grokking. *arXiv preprint arXiv:2301.05217*, 2023.
- Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. Zoom in: An introduction to circuits. *Distill*, 2020.
- Shauli Ravfogel, Yanai Elazar, Hila Gonen, Michael Twiton, and Yoav Goldberg. Null it out: Guarding protected attributes by iterative nullspace projection. *Proceedings of ACL*, pages 7237–7256, 2020.
- Jacob Steinhardt, Pang Wei Koh, and Percy S Liang. Identifying and controlling for dataset bias in deep learning. *Advances in Neural Information Processing Systems*, 31, 2018.
- Kechi Zhang, Chen Sun, James Glass, and Jie Cao. Neural program synthesis with priority queue training. *Proceedings of ACL-IJCNLP*, pages 1684–1694, 2021.