A Two-Phase Perspective on Deep Learning Dynamics

Grokking, double descent, information bottleneck, and circuit complexity

Robert de Mello Koch

Huzhou Normal University

Based on arXiv:2504.12700 with Animik Ghosh, as well as work in progress

The central puzzle: zero training error is not the end

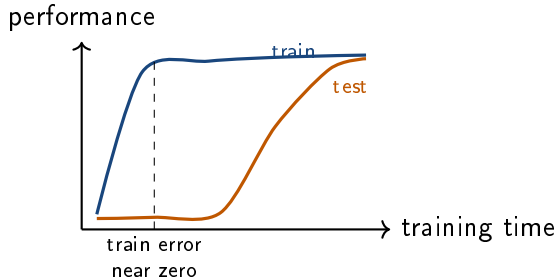
Naive expectation

Once the loss is minimized, the essential learning should be complete.

Observed behavior

In several settings the network continues to reorganize internally long after the training error has effectively vanished.

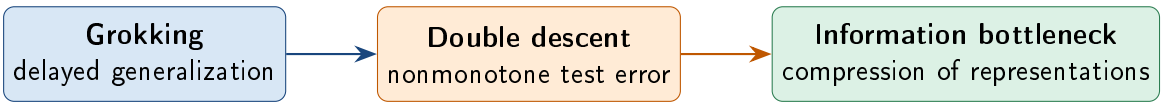
The delayed improvement is not just “more fitting”.



Roadmap

1. Three distinct phenomena: grokking, double descent, information bottleneck.
2. Why each is robust, and why each is surprising.
3. Numerical observation: their characteristic time scales align.
4. The two-phase perspective: curve fitting followed by compression/coarse graining.
5. Circuit language for the second phase: local complexity, LMN, and complexity reduction.
6. Future directions.

The three phenomena are distinct



Different diagnostics, different languages, but potentially the same transition time.

Working principle for the talk

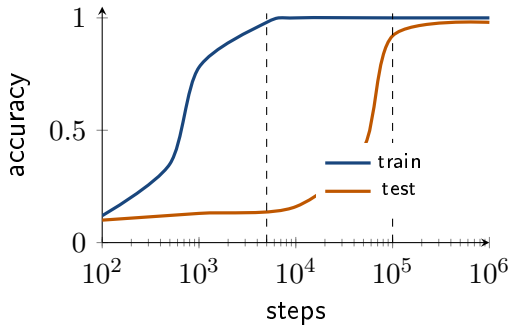
Do not collapse the concepts. Use each as an independent probe of the same hidden learning dynamics.

Grokking: what it is

Definition

A network first fits the training data, often reaching near-perfect training accuracy, but only much later abruptly improves on the test set.

- Training success comes early.
- Generalization is delayed.
- The late transition can be sharp.



Grokking: robustness and surprise

How robust?

- Originally seen in clean algorithmic tasks.
- Now observed across a broader range of tasks and architectures.
- The onset and sharpness depend strongly on data fraction, regularization, optimization, and training time.

What is surprising?

- Memorization and generalization separate in time.
- Loss can look almost finished while representations are still reorganizing.
- Standard training curves hide important late-stage structure.

Grokking says: fitting the examples is not the same as learning the rule.

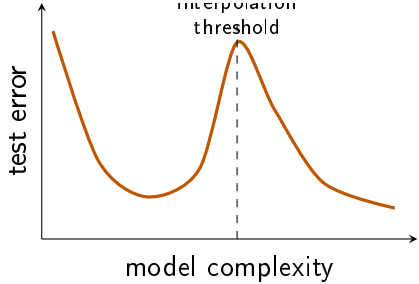
Double descent: what it is

Classical expectation

Test error follows a U-shaped curve as model complexity increases: underfit, optimal, overfit.

Double descent

Past the interpolation threshold, where the model can fit the training set, test error can decrease again.



Double descent: robustness and surprise

How robust?

- Seen in neural networks, kernel methods, and simple regression models.
- Appears as model-wise, sample-wise, or epoch-wise double descent.
- The peak location and height depend on data noise, labels, architecture, and optimization.

What is surprising?

- Bigger models can do worse near interpolation, then better again.
- Exact fitting need not imply catastrophic overfitting.
- Overparameterization can improve generalization.

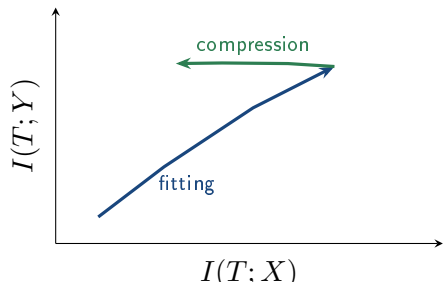
Double descent says: the interpolation threshold is not the end of generalization theory; it is a transition point.

Information bottleneck: what it is

For a hidden representation T, input X, and label Y, the information bottleneck objective is

$$\mathcal{L}_{\mathrm{IB}} = I(T;X) - \beta I(T;Y).$$

- Keep information useful for predicting Y.
- Discard information about X that is irrelevant to Y.
- Visualize layers in the information plane $(I(T;X), I(T;Y))$.



Information bottleneck: robustness and surprise

How robust?

- The principle is a robust conceptual framework for compression versus prediction.
- Empirical compression is more delicate than grokking or double descent: it depends on estimators, stochasticity, activations, and representation noise.
- In this paper it is used as a diagnostic rather than as a dogma.

What is surprising?

- Generalization is associated with forgetting, not merely with storing more structure.
- The useful representation can become less informative about the input while remaining predictive.
- This suggests a physical analogy with coarse graining.

Information bottleneck says: a good representation remembers less, but remembers the right things.

Why compare their time scales?

The key observation

Each phenomenon supplies a candidate transition time:

Phenomenon	Signature	Crossover
Grokking	Train $\rightarrow$ test accuracy	Delayed generalization
Double descent	Error peak $\rightarrow$ second descent	Beyond interpolation
Information bottleneck	$I(T;X)$ decreases	Compression begins

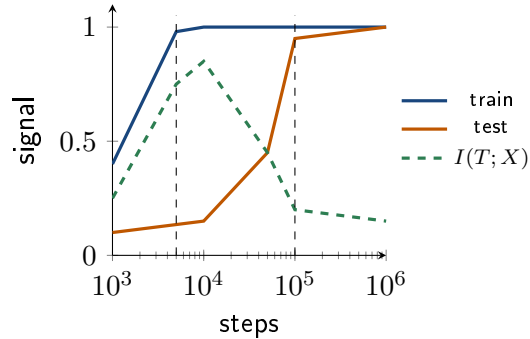
If these times agree, the three diagnostics may be seeing the same phase transition in learning dynamics.

Modular division: aligned time scales

- Training error effectively vanishes at roughly 5×10^3 steps.
- Generalization occurs at roughly 10^5 steps.
- The onset of the second descent and the rightmost point of the information-plane trajectory occur in the same window.

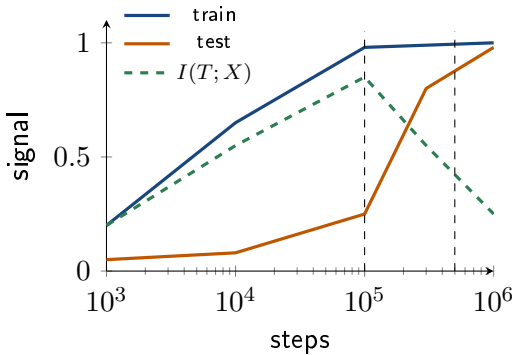
Interpretation

The network quickly fits the training set, then spends a much longer time simplifying its internal representation.



S_5: a harder algorithm, a longer compression phase

- Generalization begins around 10^5 steps.
- This coincides with the onset of second descent and the rightmost point of the information-plane trajectory.
- The fixed point in the information plane is reached later, beyond the midpoint between 10^5 and 10^6 steps.



Interpretation

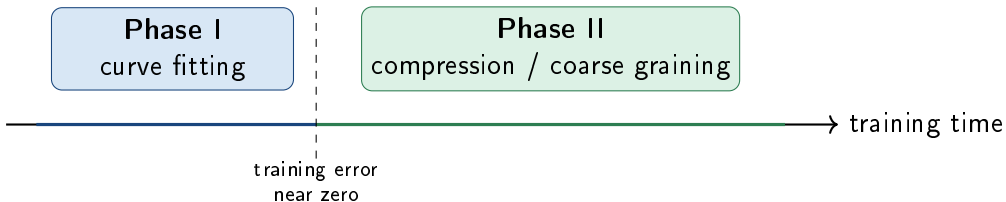
The second phase is longer for the harder S_5 task, consistent with a more demanding compression/coarse-graining process.

The alignment in one table

Dataset	Phase boundary	End of compression/generalization
Division mod 96	$\sim 5 \times 10^3$ steps: fitting ends; onset of second descent; rightmost point in information plane	$\sim 10^5$ steps: test accuracy maximal; mutual-information trajectory approaches fixed point
S_5 multiplication	$\sim 10^5$ steps: generalization begins; onset of second descent; rightmost point in information plane	beyond midpoint of 10^5–10^6 steps: information-plane fixed point approached

The same clock seems to be visible in three different languages.

The two-phase proposal

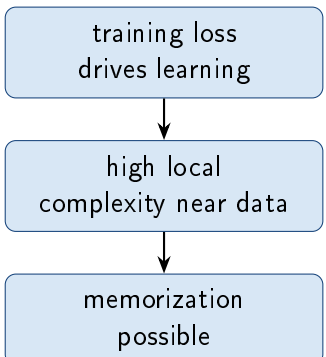


Hypothesis

Learning in a deep network proceeds through a rapid curve-fitting phase followed by a slower phase in which the network forgets irrelevant details and simplifies its effective computation.

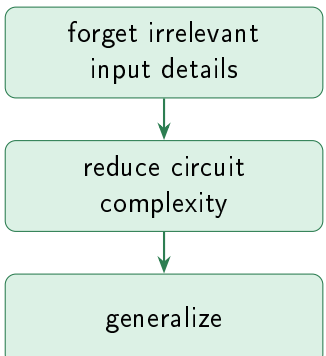
Phase I: curve fitting

- The loss is nonzero and supplies a strong optimization signal.
- The model learns enough structure to fit the training labels.
- From the information-bottleneck viewpoint, useful label information $I(T;Y)$ increases.
- From the double-descent viewpoint, the model approaches interpolation.



Phase II: compression and coarse graining

- Training loss is already tiny, but test performance still improves.
- $I(T;X)$ decreases while predictive information is retained.
- Local complexity and the number of effective circuits decrease.
- The network becomes smoother and more robust near training samples.



Why identify the phases this way?

Probe	Phase I marker	Phase II marker
Grokking	train accuracy reaches ≈ 1	test accuracy rises late
Double descent	interpolation threshold / peak	second descent
Information bottleneck	$I(T;Y)$ grows	$I(T;X)$ decreases

Point of the identification

The phase boundary is where ordinary loss minimization has done its job, but the network still has to simplify the function it has learned.

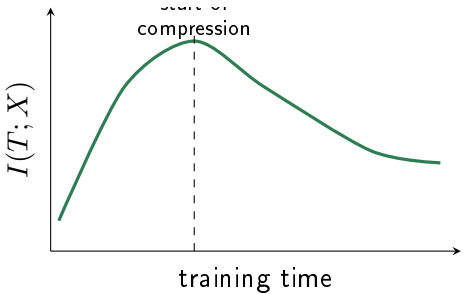
$I(T;X)$ as a progress measure

Training loss is a poor clock in Phase II

Once training error is near zero, the loss can be nearly flat while the representation continues to evolve.

Mutual information is a better clock

A decrease in $I(T;X)$ signals that the representation is discarding input details that are not needed for predicting Y.



The circuit view of a ReLU network

A feedforward network computes

$$f_\theta(\vec{x}) = a\big(W^{(L)} a(\cdots a(W^{(1)}\vec{x} + b^{(1)})\cdots) + b^{(L)}\big).$$

For ReLU,

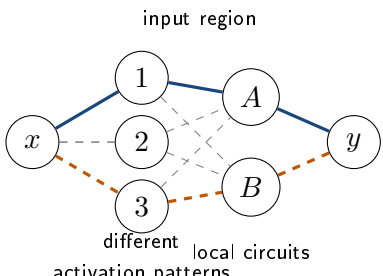
$$a(t) = \max(0, t).$$

Circuit for an input

For a fixed input, only a subset of neurons is active. The active neurons and the weights connecting them form a *circuit*. Within one activation region, the network implements a linear map.

A nonlinear network can be viewed as a patchwork of input-dependent linear computations.

The circuit picture: learning as path simplification



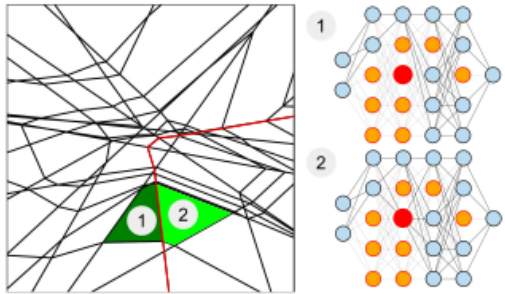
- A neural network contains many possible computational paths.
- For a given input, the activation pattern selects one effective circuit.
- Nearby inputs can activate different circuits.
- Learning can reduce the number of relevant circuits:
  - boundaries align,
  - regions disappear,
  - circuits merge.
- Generalization corresponds to a smoother, simpler local computation.

Activation boundaries partition input space

For neuron i in layer l, the activation boundary is

$$\mathcal{H}_i^{(l)} = \left\{ x_i^{(l)} : \sum_j W_{ij}^{(l)} x_j^{(l)} + b_i^{(l)} = 0 \right\}.$$

- Crossing a boundary changes which neuron is active.
- Neighboring regions differ by a small circuit change.
- Each region carries its own effective linear map.

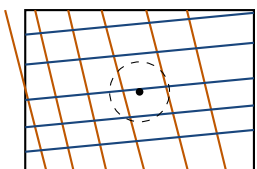


Local complexity

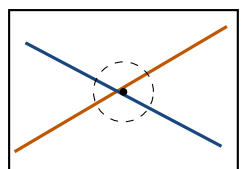
Definition in words

Local complexity counts how many distinct circuits appear in a fixed-volume neighborhood of the input space.

High local complexity



Low local complexity



During the second phase, local complexity decreases and the learned map becomes smoother around data points.

The Linear Mapping Number

Construction

Build a linear connectivity matrix L_{ij} on samples, using squared Pearson correlation coefficients to quantify how linearly related the outputs are near sample pairs.

Normalize eigenvalues by

$$\tilde{\lambda}_i = \frac{|\lambda_i|}{\sum_j |\lambda_j|}.$$

Define

$$S_{\mathrm{NL}} = -\sum_i \tilde{\lambda}_i \log_2 \tilde{\lambda}_i, \qquad \mathrm{LMN} = 2^{S_{\mathrm{NL}}}.$$

In simple settings LMN counts effective linear regions; more generally it tracks nonlinear complexity.

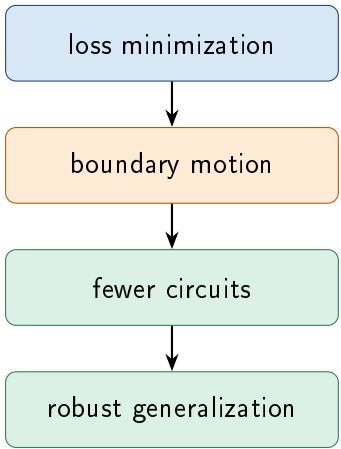
How is circuit complexity minimized?

Not by the usual loss directly

Standard training optimizes the loss. After the loss is near zero, compression is not usually an explicit objective.

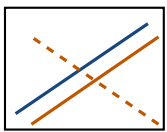
Instead

By moving activation boundaries, reducing redundant local structure.



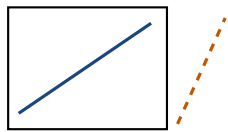
What complexity minimization looks like

Alignment



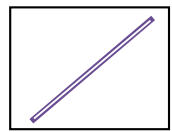
fewer intersections

Disappearance



boundary exits domain

Merging



shared boundary

Three mechanisms

Alignment, disappearance, and merging of activation zero-loci are elementary ways to reduce the number of linear regions and hence the number of effective circuits.

The renormalization-group analogy

RG in physics

Coarse graining removes microscopic details and leaves only relevant or marginal data controlling long-distance behavior.

Phase II in learning

Compression removes input-specific details and leaves only the structure relevant for prediction.

Universality in physics suggests a language for why forgetting details can improve prediction.

Practical implication: Phase II may be unnecessarily slow

Observation

The first phase is explicitly optimized by the training objective. The second phase appears as a byproduct of training dynamics.

Opportunity

If we can target compression or complexity reduction directly, we may be able to shorten the delayed generalization regime.

Future direction 1: make compression an objective

Possible training objectives:

- Add a penalty for $I(T;X)$ while maintaining $I(T;Y)$.
- Penalize LMN or a differentiable proxy for local complexity.
- Encourage merging or disappearance of redundant activation boundaries.
- Design schedules that switch objective after interpolation.

Target

Replace passive late-time compression by an explicitly optimized second-stage algorithm.

Future direction 2: a dynamical theory of circuits

- Track zero-locus motion during training.
- Derive effective equations for alignment, disappearance, and merging.
- Relate boundary dynamics to optimizer choice, weight decay, data fraction, and architecture.
- Ask whether a monotone complexity function exists during Phase II.

The goal is an RG-like flow on circuits rather than on microscopic parameters.

Future direction 3: test the universality of the two-phase picture

Broaden tasks

- Natural language tasks.
- Vision tasks.
- Noisy labels and structured noise.
- Out-of-distribution generalization.

Broaden diagnostics

- Mutual information estimates.
- Circuit/feature attribution measures.
- Representation geometry.
- Robustness to perturbations.

The strongest test is whether different diagnostics continue to identify the same clock.

Take-home message

1. Grokking, double descent, and information bottleneck are distinct phenomena.
2. Each contains a delayed transition beyond ordinary curve fitting.
3. In experiments, their characteristic time scales align.
4. This motivates a two-phase view: fast fitting followed by slow compression/coarse graining.
5. Circuits give a concrete geometric language for Phase II: complexity falls as activation regions align, disappear, or merge.

Learning may be as much about forgetting the irrelevant as fitting the data.

References

- R. de Mello Koch and A. Ghosh, *A Two-Phase Perspective on Deep Learning Dynamics*, arXiv:2504.12700.
- A. Power et al., *Grokking: Generalization Beyond Overfitting on Small Algorithmic Datasets*, arXiv:2201.02177.
- V. Varma et al., *Explaining Grokking Through Circuit Efficiency*, arXiv:2309.02390.
- N. Nanda et al., *Progress Measures for Grokking via Mechanistic Interpretability*, arXiv:2301.05217.
- P. Nakkiran et al., *Deep Double Descent: Where Bigger Models and More Data Hurt*, J. Stat. Mech. 2021, 124003.
- N. Tishby, F. Pereira, and W. Bialek, *The Information Bottleneck Method*, arXiv:physics/0004057.
- R. Shwartz-Ziv and N. Tishby, *Opening the Black Box of Deep Neural Networks via Information*, arXiv:1703.00810.
- C. Olah et al., *Zoom In: An Introduction to Circuits*, Distill, 2020.
- K. G. Wilson, *The Renormalization Group: Critical Phenomena and the Kondo Problem*, Rev. Mod. Phys. 47, 773 (1975).

Questions?