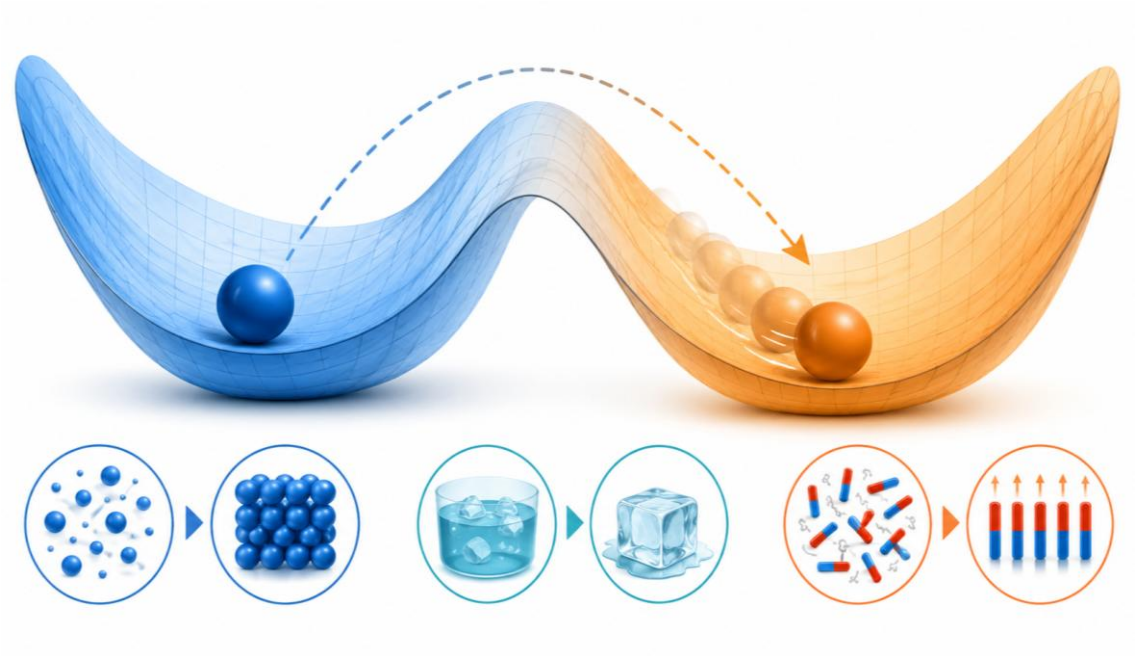
Grokking: Sudden Delayed Generalization

When a neural net memorizes first — then suddenly discovers the rule

1 The phenomenon

- Training accuracy rises quickly
- Test accuracy stays flat for a long plateau
- Then test performance climbs abruptly
- The model shifts from examples to rule



2 The physics lens

- Training is movement in a loss landscape
- Memorization and rules are competing basins
- Weight decay, noise, and time change stability
- The jump resembles a phase transition

3 What makes it weird?

- The model already fits the training set
- Nothing new is added to the data
- Yet internal representations reorganize
- Generalization emerges late

4 Practical warning

- Perfect train performance can hide shortcuts
- Early stopping may freeze memorization
- Evaluation must break superficial cues
- Longer training can reveal robust structure

Grocking is not magic: it is delayed movement from a memorizing basin to a rule-like basin.

Modular Arithmetic: A Clean Test Case

A toy task where memorization and rule learning are easy to separate

1 The task: addition with wraparound

- On a 12-hour clock: 9 + 5 = 2
- For modulus 17: numbers are 0…16
- Input pair: (a,b); target: (a+b) mod 17
- There are 17×17 = 289 possible pairs

2 Train/test split

- Show only part of the table, e.g. 40%
- Hold back the rest as unseen test cases
- Memorizer: good only on shown pairs
- Rule learner: works almost everywhere

3 Why it is ideal for grokking

- A compact algorithm exists
- Surface similarity is not enough
- Unseen pairs expose shortcuts
- Plots clearly show delayed generalization

4 Memorize vs rule

M Lookup individual pairs

R Learn wraparound structure

? Test pairs reveal which strategy won

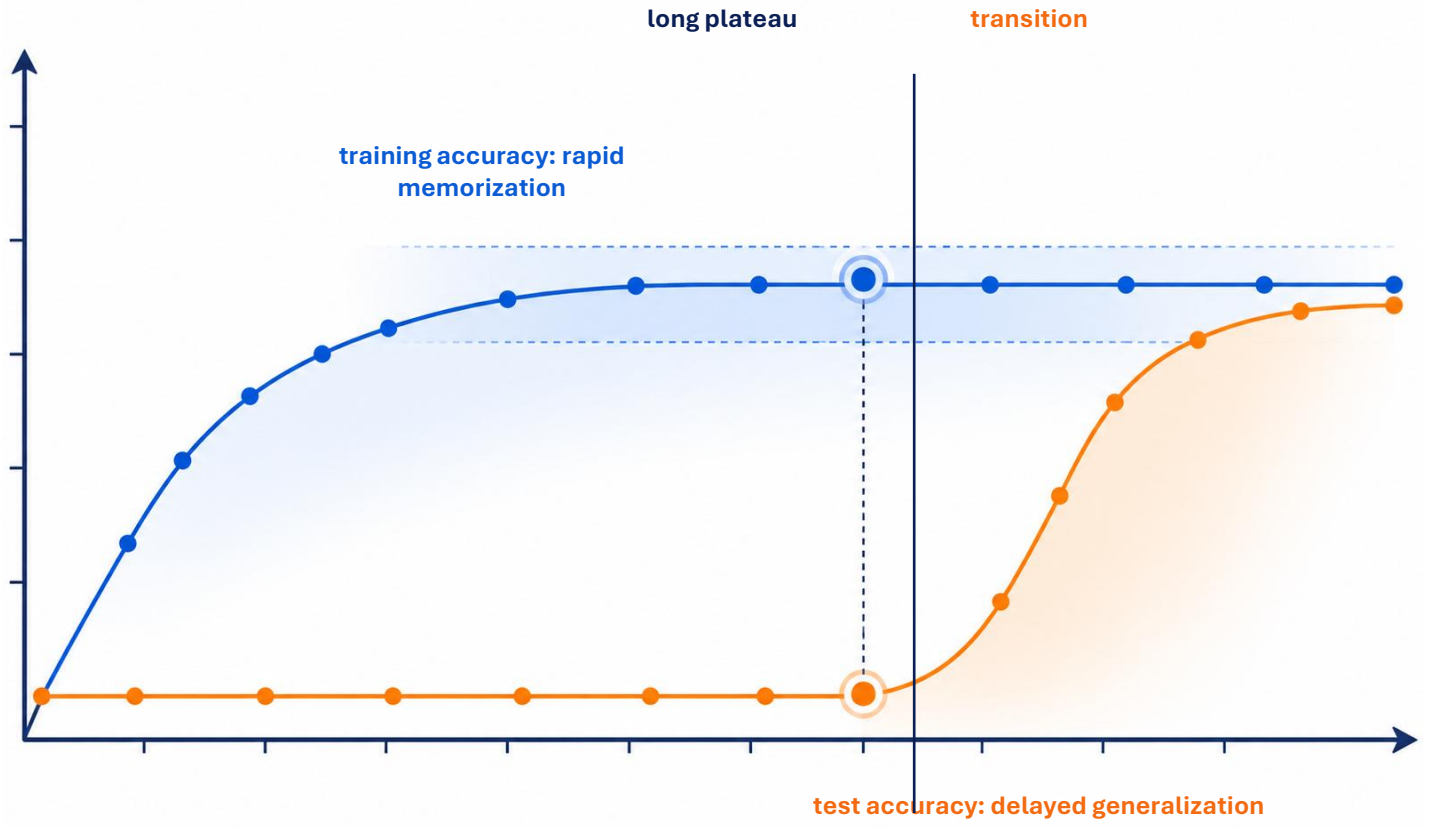
5 Key idea

- Both strategies can fit training examples
- Only the rule extrapolates across the full grid
- Grokking is the late selection of the rule

The task is deliberately simple so that "generalization" has an unambiguous meaning.

The Grokking Curve

Train accuracy saturates early; test accuracy waits, then catches up



1 What the plot says

- First phase: the model fits seen examples
- Middle phase: train/test gap persists
- Late phase: test accuracy rises sharply

2 Accuracy can exaggerate cliffs

- Accuracy is step-like: right or wrong
- If measured sparsely, a fast ramp looks instant
- Test loss often shows a smoother rise

3 Still, a real change occurs

- Internal strategy changes
- The generalization gap collapses
- Behavior becomes rule-like

Grokking is about a delayed strategy shift, not simply a dramatic-looking plot.

Memorization vs. Rule Discovery

The training set permits many solutions; only some generalize

1 Memorization basin

- Fits shown examples perfectly
- Behaves like a complicated lookup table
- Can use many special-case patches
- Often quick for gradient descent to find
- Fragile on renamed, permuted, or larger cases

Great train score
Poor transfer

3 Training is selection

many functions fit the training data

Which solution does optimization choose?

2 Rule basin

- Implements the underlying algorithm
- Works on unseen combinations
- Uses coordinated internal structure
- May require longer training to assemble
- Robust under irrelevant transformations

Good train score
Good transfer

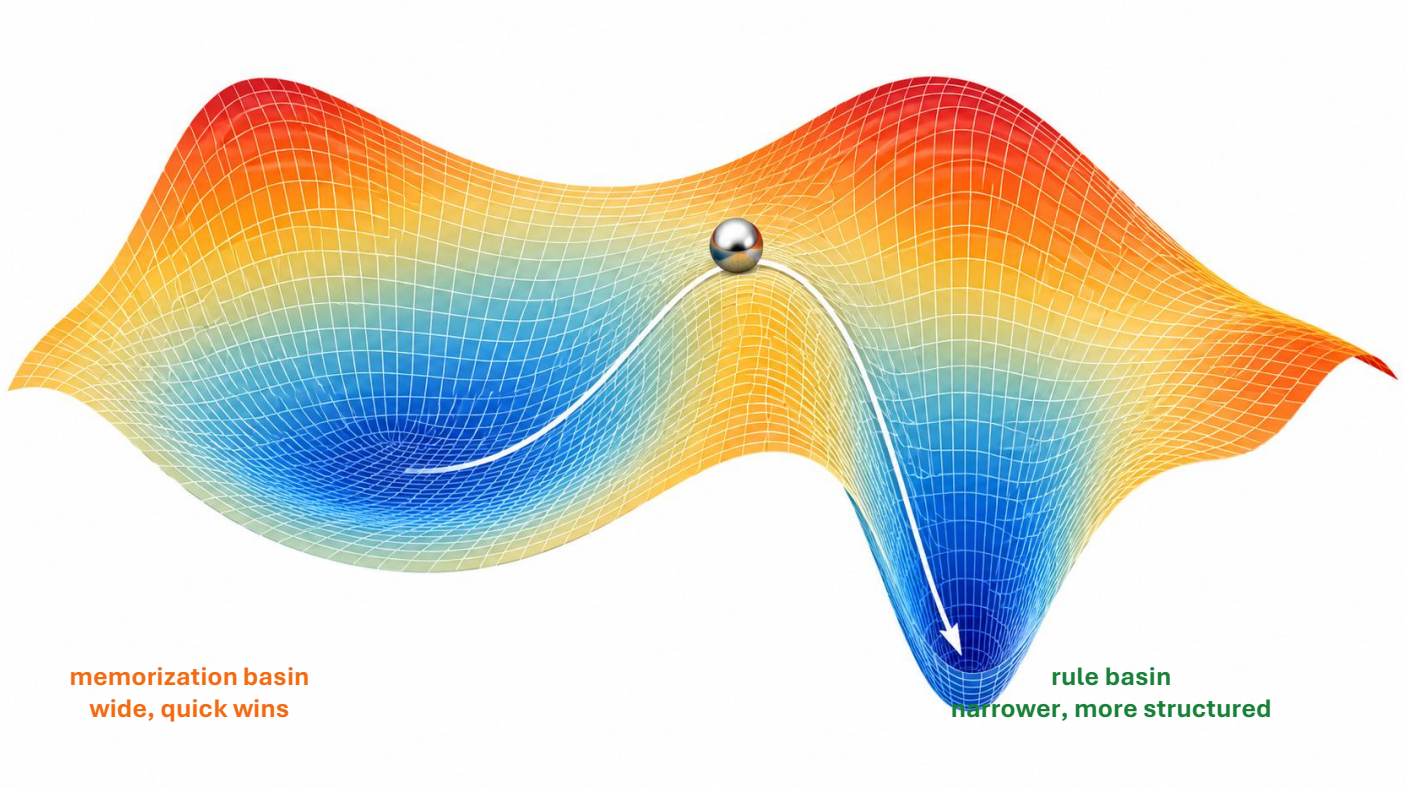
Grocking is the migration from a solution that fits examples to one that captures the rule.

Optimization as an Energy Landscape

Loss, regularization, and noise create the terrain that training explores

1 What height means

- Height ≈ training objective
- Lower is better
- Loss rewards fitting data
- Penalties reshape the surface



2 What motion means

- Gradient descent moves locally downhill
- Noise shakes the trajectory
- Weight decay pulls toward smaller norms
- Schedules cool or heat the search

3 Physics translation

- **E** Energy: lower loss and lower penalty
- **S** Entropy: how many parameter settings realize a behavior
- **F** Free energy: fit, simplicity, and volume trade off

The model is not "choosing elegantly"; it follows the terrain created by objective, architecture, data, and optimizer.

Why Memorizing Is Often Easier First

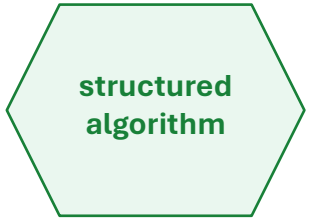
A simple rule for humans may be hard for gradient descent to discover early

1 Memorization has high volume

- Many parameter settings can patch training errors
- Large models have many adjustable knobs
- Local fixes produce immediate loss reduction
- This creates a roomy basin of solutions

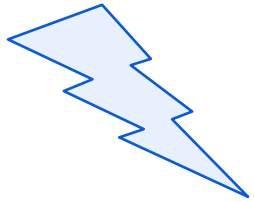
2 Rule learning needs coordination

- Representations must organize coherently
- Weights must align to implement an algorithm
- The payoff may appear only after structure forms
- The basin can be narrower early on



3 Why gradient descent follows shortcuts

- It is a local downhill process
- It follows the fastest error reduction now
- It does not search for elegance or meaning
- Patchwork solutions often win the early race



Early training rewards whatever reduces loss fastest, not necessarily what generalizes best.

Weight Decay and Optimization Noise

During the plateau, the model is not idle — it drifts

1 1. Weight decay

- Gently penalizes large weights
- Keeps acting after training accuracy is high
- Shrinks contorted special-case hacks
- Creates pressure toward simpler parameterizations

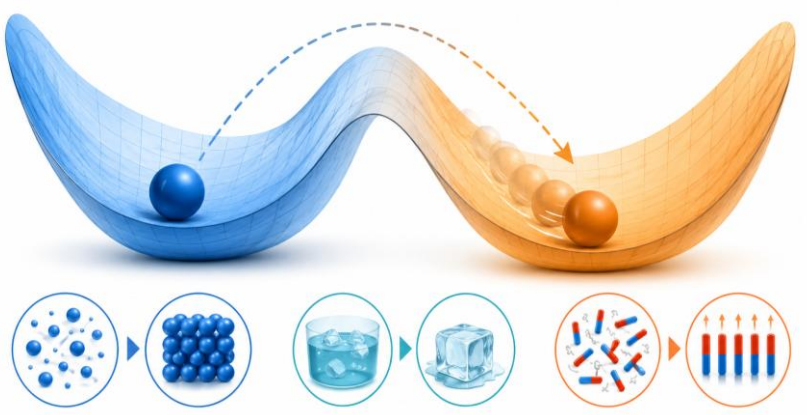
smaller weight norms

2 2. SGD noise

- Mini-batches add random jitter
- Noise acts like temperature
- It helps exploration of nearby configurations
- Too much noise prevents settling

jitter

3 3. Slow drift changes stability

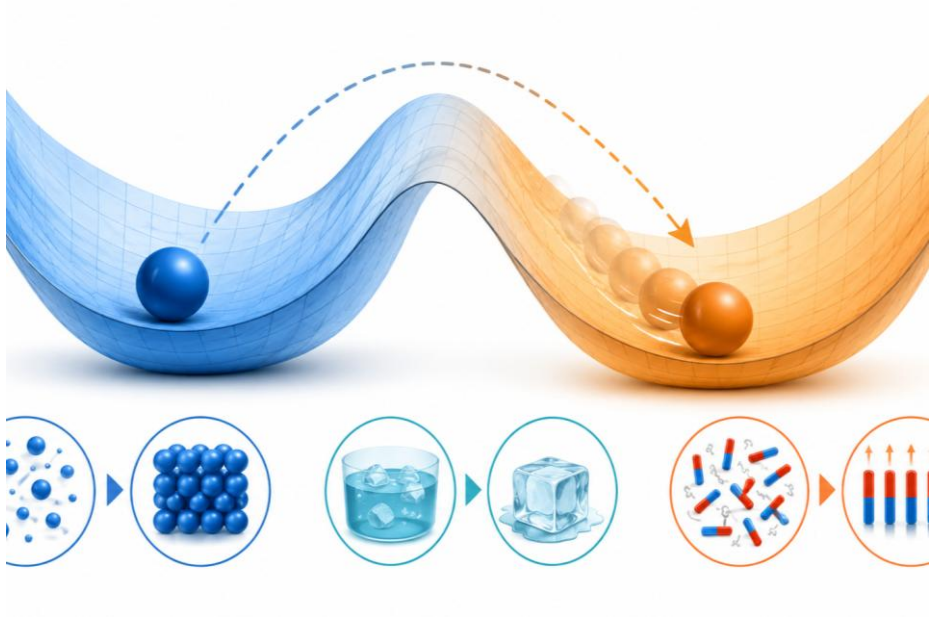


- After memorization, strong loss gradients fade
- Regularization and noise become relatively important
- Special-case memorization erodes
- Economical rule strategies become easier to maintain

The plateau is active: small forces accumulate until the preferred solution changes.

Grokking as a Phase Transition

A smooth control knob can produce an abrupt behavioral switch



1 Physics analogy

- Water freezes when temperature crosses a threshold
- Magnets align when fields and temperature favor order
- The system can sit in one regime, then snap to another

2 Grokking analogy

- Control knobs: training time, weight decay, batch noise, data size
- Order parameter: test accuracy or generalization gap
- Regime shift: memorizer → rule learner

3 Free-energy intuition

1. Early: reducing training loss dominates → memorization is rewarded quickly
2. Plateau: loss is low; regularization and noise keep pushing
3. Tipping point: the rule basin becomes cheaper and more stable

The jump looks sudden because the system crosses a stability threshold.

Control Knobs: Why Grokking Is Sensitive

Capacity, data, regularization, and training dynamics interact

1 Weight decay

- Too weak: memorization remains comfortable
- Too strong: model cannot fit training data
- Goldilocks: destabilizes brittle hacks

2 Batch size / noise

- Small batch: more temperature/exploration
- Large batch: smoother, colder trajectory
- Too much noise can prevent settling

3 Learning rate schedule

- Cooling can freeze the current basin
- Cool too early: lock in shortcuts
- Cool later: settle into rule once favorable

4 Model size + data

- Larger models memorize more easily
- But can also represent the rule cleanly
- Data fraction changes how obvious the rule is

5 Goldilocks zone

Grocking often appears when the model can memorize first, but continued training plus simplicity pressure eventually makes memorization unstable.

No single knob explains grokking; it emerges from the interaction of optimizer, regularization, capacity, and task structure.

How to Detect Rule Learning — and Why It Matters

Grokking is both a warning and a promise for ML practice

1 Test beyond the training distribution

- Rename symbols or permute labels
- Increase sequence length or number size
- Change irrelevant details
- Evaluate compositional combinations
- Measure test loss, not only accuracy

2 Warning

- Perfect train performance can mean lookup-table behavior
- A long flat test curve may hide internal drift
- Early stopping can freeze a brittle phase
- Weak evaluations can mistake shortcuts for understanding

memorize → grok → generalize

3 Promise

- Longer training can reveal robust structure
- Simplicity pressure can favor algorithms
- No new architecture or data may be needed
- The model may discover a rule it was capable of all along

Conceptual takeaway: learning is not just fitting examples; it is selecting one way to fit them from a vast solution space.