# The Learning Signal Was Always There. We Just Never Saw It.


Bouarfa Mahi
Quantiota
Université Joseph Fourier
Grenoble, Auvergne-Rhône-Alpes, FR
info@quantiota.org


January 7, 2026


**Abstract**

For fifty years, every neural network has computed a local learning signal at every layer, at every step, in every forward pass. This signal—the local entropy—was hidden in plain sight within the activation function and its outputs. This article reveals how the sigmoid function emerges from entropy minimization, and how the simple dot product $z \cdot dD$ contains all the information needed for learning without backpropagation. The implications challenge fundamental assumptions about neural network training.


## 1 The Equation Everyone Knows

If you've ever studied neural networks, you've seen this equation:

$$D = \sigma(z) = \frac{1}{1 + e^{-z}} \tag{1}$$

The sigmoid function. The classic activation. Written in countless textbooks, implemented in millions of models, computed trillions of times.

We called $z$ the "pre-activation." We called $\sigma$ the "activation function." We called $D$ the "output."

Then we moved on—passing $D$ to the next layer, waiting until the final output to compute a loss, then backpropagating errors through the entire network.

What if the learning signal was right there, in every layer, the whole time?

## 2 What We Missed

Consider what every layer actually produces:

- $z$: accumulated information from inputs and weights
- $D$: a value between 0 and 1, representing certainty
- $\Delta D$: how that certainty changes as learning proceeds

Now consider this simple operation:

$$h = z \cdot \Delta D \tag{2}$$

A dot product. Multiplication and sum. The simplest possible operation.

This quantity $h$ is the local entropy—a measure of uncertainty at that specific layer. And it requires no information from any other part of the network.

Every layer has always had access to its own knowledge state ($z$), its own decision certainty ($D$), and its own entropy ($h$).

**Every layer has always had everything it needs to learn. We just never asked it to.**

## 3 The Sigmoid's Secret

Why does the sigmoid function work so well in neural networks?

Standard answers include smooth gradients, bounded outputs, and biological inspiration. But there's a deeper answer.

The sigmoid is the **unique** function that makes local entropy equivalent to Shannon entropy. When $D = \sigma(z)$, the entropy computed locally at each layer matches exactly the classical information-theoretic measure of uncertainty:

$$H_{\text{SKA}} = -\frac{1}{\ln 2} \int z \, dD = H_{\text{Shannon}} \tag{3}$$

The sigmoid wasn't an arbitrary choice that happened to work. It was **mathematically necessary** for entropy-optimal learning.

We used it for decades without knowing why it was special.

## 4 The Tragedy of Oversight

Here's what happened in every neural network ever trained:

| | |
|---|---|
| **Forward pass:** | |
| Layer 1: | Compute $z_1 \to D_1 \to$ [Entropy available] |
| Layer 2: | Compute $z_2 \to D_2 \to$ [Entropy available] |
| Layer 3: | Compute $z_3 \to D_3 \to$ [Entropy available] |
| Output: | Compute loss from final layer only |
| | |
| **Backward pass:** | Propagate gradients back through all layers |

Each layer computed its own entropy. Each layer had its own learning signal. But we **threw it all away**, waiting instead for a global error to travel backward from the output.

We had local information and demanded global coordination.

## 5 Why Didn't Anyone See This?

Several reasons explain this collective oversight:

**Success blinds.** Backpropagation worked. When something works, you don't question its foundations.

**Framing constrains.** We called $z$ a "pre-activation"—a transient computational step. We didn't call it "knowledge." We called $D$ an "activation output"—something to pass forward. We didn't call it a "decision probability."

**Complexity attracts.** Researchers are drawn to sophisticated mathematics. A dot product seemed too simple to be fundamental.

**Assumptions hide.** Everyone assumed learning required global information. So no one looked for local signals.

**Computation isn't comprehension.** The quantity $z \cdot dD$ was locally computable but not conceptually visible. Every forward pass calculated it. No one recognized its meaning.

## 6 The Hidden Learning Signal

The quantity that was always there:

$$h = -\frac{1}{\ln 2} \cdot z \cdot dD \tag{4}$$

This is the local entropy at any point in the network. It measures how much uncertainty exists at that location, based only on local information.

When each unit minimizes its own entropy, something remarkable happens: knowledge structures organize, decision boundaries form, and classification emerges.

No targets required. No backward pass. No global coordination.

**The learning signal was never at the output. It was everywhere.**

## 7 What This Means

If local entropy has always been present at every layer, several implications follow:

**Backpropagation is one way, not the only way.** We can learn by following global gradients backward. We can also learn by reducing local entropy forward.

**The sigmoid is derived, not assumed.** It emerges from the mathematics of entropy minimization, not from trial and error.

**Every layer can be autonomous.** Layers don't need to wait for instructions from the output. They contain their own learning signal.

**The architecture of computation changes.** Forward-only learning becomes possible. Memory requirements drop. Parallelization becomes natural.

## 8 Hidden in Plain Sight

The history of science is filled with discoveries that were always visible but never seen:

- Oxygen existed in every breath before Lavoisier named it
- Gravity pulled every falling object before Newton explained it
- DNA sat in every cell before Watson and Crick decoded it

And perhaps: **local entropy computed in every layer before anyone recognized its meaning**.

Sometimes the most fundamental truths are not hidden in complexity. They are hidden in simplicity—so obvious that we look right past them.

## 9 A Different Way to See

I'm not claiming that fifty years of neural network research was wrong. Backpropagation works. Global optimization works. These are genuine achievements.

But I am suggesting that we may have been solving a harder problem than necessary. We built elaborate systems to propagate global errors when local signals were available all along.

| **What We Called It** | **What It Actually Is** |
|---|---|
| Pre-activation ($z$) | Knowledge |
| Activation function ($\sigma$) | Entropy-optimal transform |
| Activation output ($D$) | Decision probability |
| Gradient component ($dD$) | Decision shift |
| $z \cdot dD$ | Local entropy (learning signal) |

Table 1: Reinterpreting neural network components.

We had the answer. We just didn't know the question.

## Acknowledgment

This perspective emerges from the Structured Knowledge Accumulation (SKA) framework, which reinterprets neural network learning as local entropy reduction. The framework demonstrates that the sigmoid function emerges from entropy minimization and that each layer contains sufficient information for autonomous learning.

## References

[1] Mahi, B. (2025). Structured Knowledge Accumulation: An Autonomous Framework for Layer-Wise Entropy Reduction in Neural Learning. *arXiv preprint*, arXiv:2503.13942 [cs.LG].

[2] Mahi, B. (2025). Structured Knowledge Accumulation: The Principle of Entropic Least Action in Forward-Only Neural Learning. *arXiv preprint*, arXiv:2504.03214 [cs.LG].

[3] Mahi, B. (2025). Structured Knowledge Accumulation: Geodesic Learning Paths and Architecture Discovery in Riemannian Neural Fields. *arXiv preprint*.

[4] Shannon, C. E. (1948). A Mathematical Theory of Communication. *Bell System Technical Journal*, 27, 379–423, 623–656.