NeurIPS 2026

Comparing Linear Regions in ReLU-Type Networks: Theory and Monte Carlo Methods

Yuan Wang · Independent Researcher, Singapore · yuanwang2011@outlook.com

Code & 3D models

TL;DR For generic parameters, linear patterns identify the non-constant linear regions, so Monte-Carlo comparison of region counts is theoretically sound. In expectation, width beats depth, $E(\mathcal{N}_{pq\times 2}) > E(\mathcal{N}_{p\times 2q})$, and neurons near the input tend to matter more.

1 Why compare linear regions?

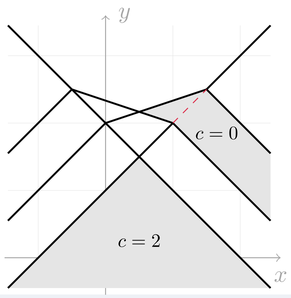
A ReLU or leaky-ReLU (*ReLU-type*) network computes a function F that is continuous and piecewise linear. Its number of **linear regions** is a standard measure of expressivity, and its worst-case maximum is often used to argue for depth [MPCB14].

- Counting the regions exactly is NP-hard [Wang22, SHR25].
- Direct comparisons between architectures, for typical random parameters, are largely missing [Wang22].

This work compares networks in theory (expected counts) and in practice (Monte-Carlo sampling), with minimal prerequisites.

2 Setting and definitions

- $\mathcal{N}_{(n_1,\dots,n_l)}$: fully connected, input dimension M, output dimension 1, and l hidden layers of widths $n_1,\dots,n_l$. $\mathcal{N}_{n\times l}$: all l widths equal n.
- Weights and biases: independent continuous random variables with even densities (e.g. PyTorch's default `nn.Linear` initialization). $E(\mathcal{N})$: the expected number of linear regions.
- **Activation region**: a full-dimensional set of inputs sharing one on/off pattern of all neurons (always convex).
- **Linear region**: the closure of a maximal connected open set on which F is affine.
- **Linear pattern** of a generic input x: the affine function that F equals near x.



Linear regions of a ReLU network with 2 inputs and two hidden layers of width 2. Grey regions are constant ($F \equiv c$). The dashed line splits the upper-right constant region into two activation regions, which together form one linear region.

References

[HR19b] B. Hanin, D. Rolnick. Deep ReLU networks have surprisingly few activation patterns. NeurIPS 2019.

[MPCB14] G. Montúfar, R. Pascanu, K. Cho, Y. Bengio. On the number of linear regions of deep neural networks. NeurIPS 2014.

[Wang22] Y. Wang. Estimation and comparison of linear regions for ReLU networks. IJCAI 2022.

[SHR25] M. Stargalla, C. Hertrich, D. Reichman. The computational complexity of counting linear regions in ReLU neural networks. NeurIPS 2025.

3 Uniqueness of linear patterns

Theorem 1 (generic uniqueness of linear patterns)

For generic weights and biases:

1. The non-constant linear regions are exactly the non-constant activation regions. In particular, they are convex, and no two linear regions share a linear pattern.
2. If $M \geq n_1$ and $l \leq 2$, no two constant linear regions share a linear pattern either.

For leaky ReLU there are generically no constant regions, so Part 1 covers every region.

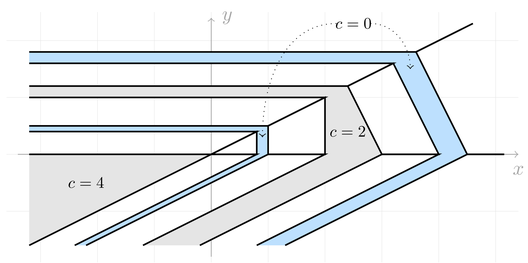
Proof idea. On the activation region of a pattern A, F is affine, and each of its coefficients is a polynomial in the parameters with one monomial per input-to-output path:

$$c_{A,i} = \sum_{\text{paths}} \Big(\prod_{r=1}^{l} s_A(i_r)\Big)\, w_1^{i,i_1} w_2^{i_1,i_2} \cdots w_{l+1}^{i_l,1},$$

where $s_A(i_r)$ is 1 for an active neuron and the negative slope α otherwise ($\alpha = 0$ for ReLU). Monomials of different paths never cancel, so two distinct patterns, at least one non-constant, give different polynomials. They can share a linear pattern only on a measure-zero set of parameters.

Consequence. Two generic inputs share a non-constant linear pattern *if and only if* they lie in the same linear region. Counting the distinct patterns of sampled inputs therefore counts the regions they hit (block 7).

4 The conditions in Part 2 are needed



A ReLU network with input dimension $M = 2$ and three hidden layers of width 2 has two *disconnected* constant regions with the same value $c = 0$ (blue), and this persists under small perturbations of the parameters. A one-dimensional example with $M = 1 < n_1 = 2$ and $l = 2$ also exists.

5 Width beats depth in expectation



$\mathcal{N}_{pq\times 2}$ (here 4×2) $>$ $\mathcal{N}_{p\times 2q}$ (here 2×4)

Theorem 2 (comparison)

For integers $p \geq 1$, $q \geq 2$ and input dimension $M \geq pq$, for ReLU and leaky ReLU,

$$E(\mathcal{N}_{pq\times 2}) > E(\mathcal{N}_{p\times 2q}).$$

For ReLU, moreover,

$$E(\mathcal{N}_{pq\times 2}) > \frac{2^{pq}-1}{2^{pq-q+1}}\, E(\mathcal{N}_{p\times 2q}),$$

with a factor of roughly 2^{q-1}.

Both networks have the same number of hidden neurons, $2pq$.

6 Exact counts and bounds along the way

With $G(m,n) = \frac{1}{2^m}\sum_{i=0}^{m}\binom{m+n}{i}$ and $T(m,n) = \frac{1}{2^m}\sum_{i=0}^{m}\binom{m}{i}G(i,n)$:

- **Upper bound** for $M \geq n_1 \geq \cdots \geq n_l$: $E(\mathcal{N}) \leq 2^{n_1}\prod_{i=1}^{l-1} T(n_i, n_{i+1})$ for ReLU, and the same with G for leaky ReLU.
- **Exact, two hidden layers** ($M \geq n_1$):

  $$E(\mathcal{N}_{(n_1,n_2)}) = 2^{n_1}\big(1-2^{-n_2}\big)T(n_1,n_2) + 2^{-n_2}G(n_1,n_2)$$

  for ReLU. For leaky ReLU, the count is *exactly* $2^{n_1}G(n_1,n_2)$ with probability 1; this fails from depth 3 on (an example has 15 vs. 16 regions).
- **Proof idea**: switch activations on one layer at a time, from the last hidden layer to the first. Each step slices the existing regions along the strata of the new layer's positive orthant. By symmetry, m random hyperplanes cut an orthant of $\mathbb{R}^n$ into $\frac{1}{2^n}\sum_{i=0}^{n}\binom{n+m}{i}$ regions in expectation.

Parameter counts do not determine region counts

Input dimension 10, ReLU, exact formula above.

$\mathcal{N}_{(10,10)}$: 231 parameters	$\mathcal{N}_{(5,30)}$: 266 parameters
176,694	**37,480**
expected linear regions	expected linear regions

7 Monte-Carlo comparison

1. Sample K inputs $x \sim N(0, \sigma^2 I)$.
2. Compute each input's linear pattern (gradient and intercept), at a cost linear in the number of parameters.
3. Count the distinct non-constant patterns, the *detected linear patterns*.

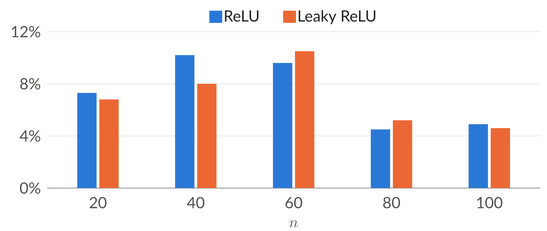
By Theorem 1, this is the number of non-constant regions hit by the sample, a lower bound on the true count. Its expectation is

$$\textstyle\sum_R \big(1-(1-\mu(R))^K\big),$$

so regions are weighted by their probability mass $\mu(R)$ under the input distribution.

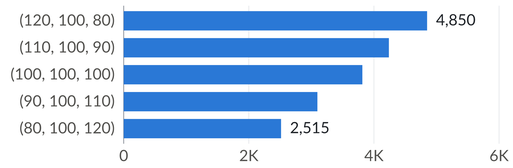
Wider beats deeper, at equal width + depth

% more detected patterns for width n, depth $n-1$ than for width $n-1$, depth n. $\sigma = 0.01$, 10^4 inputs, mean over 1000 random networks.



Front-loaded widths yield more patterns

Hidden widths (n_1, n_2, n_3) with 300 hidden neurons in total, input dimension 100, ReLU (leaky ReLU within 0.4%). Same sampling as above.



Calibration. On small networks with known counts, the detected number never exceeds the truth and approaches it as K grows (leaky-ReLU $\mathcal{N}_{2\times 2}$: $9.25 \to 10.76$ of 11 regions for $K = 10^4 \to 7\times 10^7$).

Takeaway

Neurons closer to the input layer tend to have more impact on the number of linear regions: adding them to early layers usually increases the count more than adding them to later ones.

Limitations. Theorem 2 compares $pq\times 2$ with $p\times 2q$ only. The Monte-Carlo count is a lower bound whose resolution is set by K and σ; saturated counts are inconclusive.