A Generalization Result for Overparametrized Neural Net Bayesian Learning

Dmitry Vaintrob Kaarel Hänni

ILIAD 2026

Kaarel used Claude's help when creating these slides.

Plan for the talk

1. Introduction and motivation.
2. The main toy learning theorem, proved in four parts:
   - a learning lemma,
   - the small-width case,
   - cloning,
   - subsampling.
3. Discussion.

The intention is to do $\sim$1 hour of main talk, and then have open discussion. Please feel free to leave for lunch after the main talk (and also, of course, feel free to leave earlier).

Part 1: Introduction and motivation

Some interesting questions about neural net learning

- When and why does neural net learning have good generalization behavior?
- In many cases, we see good generalization while the network could implement many more functions that label training data just as well but would do poorly on test data. Why does a well-generalizing function get picked out (in cases where it does)?
- Good generalization should relate to having a preference toward simpler functions. Why does neural net learning have this preference? Wait, what should we even mean by "simple" here?
- When, and why, does it generalize to genuinely novel inputs? Can we get a better sense of what NN out-of-distribution generalization is like?

What this talk does

We study a **Bayesian** version of neural net learning: put a prior on the weights, condition on the training data, and let the posterior predict on test inputs.

Our main results are (much more caveated and detailed versions of):

- The NNbayes prior of a function roughly tracks its circuit complexity, even when the neural net is overparametrized.
- Consequently, when the function being learned is circuit-simple, it is learned quickly (again, even when the neural net is overparametrized).

In this talk, we only prove the above in a simple case: when the prior is *very strong*, i.e. strongly prefers weight vectors of small ℓ^2 norm. But we believe the proof ideas generalize to other cases of NNbayes — we will discuss this at the end of the talk.

Speculation on the relevance to AI alignment

- **Relevance to understanding generalization of NN-based learning:**
  - Maybe it'd be helpful if we had a clearer way to think about what is happening during (supervised) NN learning. This could be helpful even if that description is mostly behavioral — for instance, if the description were that NN SGD learning is close in its input–output behavior to some sort of local minimal discrete circuit search.
  - One major problem here is that even if one establishes a correspondence with some crisper learning process, that's probably the easy part, with understanding generalization of that crisper process (or really designing some learning process with good high-level properties — perhaps one that generalizes in a benign fashion) being much more difficult.
  - In a different direction, maybe a clearer understanding of neural net gradient descent learning could lead to the construction of some more crisp learning algorithm that is still somewhat practical, and that could be useful for designing better-understood learners.

Speculation on the relevance to AI alignment

- **Relevance to interpretability:**
  - I think we are very confused about how to think about the structure in/of neural nets (or minds). Maybe trying to get a better understanding of NN learning is a problem that pushes you toward better ways to think about this structure. Maybe ideas so obtained could contribute to the project of rewriting neural nets into some more understandable form ("full reverse-engineering").
  - Concretely, the present line of research suggests the ideas of thinking about the distribution of neuron weight vectors, and about kernels computed at each layer.

Speculation on the relevance to AI alignment

- **But isn't this work about the overparametrized case, whereas real neural nets are usually not that overparametrized?** Two thoughts in response:
  - A lot of learning probably happens quite quickly as you start seeing data, in a regime where you are effectively overparametrized.
  - If we believe in a view of NN structure that says there are many different mechanisms, with each only being active on a sparse set of data points, then learning is probably overparametrized "from the point of view of such a sparse mechanism".
- **But isn't this about the Bayesian case, not SGD?**
  - Yes, I think these are importantly different. But also, probably some ideas carry over.
- Ultimately, tbh, my own view is that NN learning theory and principled interpretability are not among the better bets for alignment (but also I'm very pessimistic about all bets). Dmitry disagrees, and I encourage you to go to his talk tomorrow to hear his thoughts.

Part 2: The main learning theorem

Warm-up: Solomonoff function induction

We want to learn a function $f : X \to Y$ from examples: draw $x_1, \ldots, x_n \sim \mathcal{D}$ i.i.d., see the labels $y_i = f(x_i)$, and then predict f on fresh inputs $x \sim \mathcal{D}$.

Be Bayesian, with *Python programs* as the hypotheses:

- a program p computes a function $X \to Y$ (technically, elements of X and Y are presented as strings);
- the prior favors short source code: $\pi(p) \propto 3^{-|p|}$, where $|p|$ is the length of p in bits. (Base 3 rather than 2 makes sure the total mass $\sum_p 3^{-|p|}$ is finite.)

Learning is conditioning: the posterior is the prior restricted to the programs that get all n examples right.

Write C for the length in bits of the shortest program computing f, and $\mathrm{err}(g) = \Pr_{x\sim\mathcal{D}}[\, g(x) \neq f(x)\,]$ — test error is *in-distribution*. Our figure of merit is the posterior's *average probability on the correct answer*,

$$\mathbb{E}_{x\sim\mathcal{D}}\big[\,\mathrm{posterior}\{\, g : g(x) = f(x)\,\}\big] \;=\; 1 - \mathbb{E}_{g\sim\mathrm{posterior}}\big[\mathrm{err}(g)\big].$$

Warm-up: Solomonoff function induction

If the target has a short program, few examples suffice:

Proposition

After $n = O(C)$ samples, with probability $1 - \exp(-\Omega(C))$ over the draw of the samples, the average probability on the correct answer is $> 99.9\%$.

Corollary

Predicting the label with the most posterior mass then gets $< 0.2\%$ of the test labels wrong.

(To be wrong at x, that prediction needs the right answer to carry $\leq \frac{1}{2}$ of the posterior at x; Markov.)

Proof

(We give a proof that is more complicated than necessary, because this proof will generalize to our neural net case.)

Order all programs by length — i.e., by decreasing prior mass. Three kinds of programs:

The long tail never matters. The programs of length $> 3C + 25$ together have prior mass $< \pi(p^*)/2000$ (check: $\sum_{\ell \geq 3C+26} 2^\ell \, 3^{-\ell} = 3\,(2/3)^{3C+26} < 3^{-C}/2000$). And on *every* training set f's shortest program p^* is consistent, so the posterior denominator is $\geq \pi(p^*)$: the tail always carries $< \frac{1}{2000}$ of the posterior.

Bad short programs die. Call g *bad* if $\mathrm{err}(g) \geq \frac{1}{2000}$; it stays consistent with all n samples with probability $\leq e^{-n/2000}$. Union bound over the $< 2^{3C+26}$ short programs: with probability $\geq 1 - e^{-C}$ — at $n = 2000\big((3C + 26)\log 2 + C\big) = O(C)$ — *every* bad short program is eliminated outright.

Good programs are good. Everything else is wrong on a fresh input with probability $< \frac{1}{2000}$.

On that event, the average probability on the correct answer is $> 1 - \frac{1}{2000} - \frac{1}{2000} = 99.9\%$. □

The learning proposition really proved by the warm-up

Proposition (relevance-based sample complexity bound — informal version)

Suppose all but M other hypotheses are "irrelevant" when learning f (because their priors are too small compared to the prior of f). Then the sample complexity of learning f is $O(\log M)$.

Proposition (relevance-based sample complexity bound)

We Bayesian-learn $f : X \to Y$ using a countable hypothesis class $\mathcal{H} \ni f$ and a prior π on $\mathcal{H}$. Order the hypotheses in decreasing order of prior, and let M be an index such that the *tail* after it has at most $\pi(f)/2000$ mass. Then after $4000\,(\log M + 4)$ samples, with probability $\geq 1 - \frac{1}{1000M}$ over the draw of the samples, the average probability on the correct answer is $> 99.9\%$.

The proof is the warm-up's argument run in general; it is written out, with the error levels as free parameters, in Appendix A.

The proposition, instantiated

- **The warm-up** was exactly this: the top-mass hypotheses are the shortest programs, and the tail of programs of length $> 3C + 25$ has mass $< \pi(p^*)/2000$ — so $M < 2^{3C+26}$.
- **This talk**: $\mathcal{H} =$ the functions implementable by a width-N neural net, with the prior induced by Gaussian weights. We will be showing that the number of functions relevant when learning f is at most $\exp(\mathrm{poly}(d, \text{circuit complexity of } f))$. By our relevance-based bound, this means the sample complexity of learning f is $O(\mathrm{poly}(d, \text{circuit complexity of } f))$.

Our neural net Bayesian learning setting

- **Target.** A boolean function $f : \{1, -1\}^d \to \{1, -1\}$.
- **Architecture.** A standard MLP of depth L and width N, sigmoid activation ϕ bounded by ± 1, one output node $z_w(x)$. We ask three things of ϕ: that it be *bounded*, *Lipschitz*, and *non-polynomial*.
- **Biases.** No separate bias parameters: we append a constant input $x_0 = 1$.
- **Interpreting a weight vector as a boolean function.** w outputs $+1$ on x if $z_w(x) > \frac{1}{2}$ and -1 if $z_w(x) < -\frac{1}{2}$; we restrict the prior ahead of time to w implementing *full* boolean functions: $|z_w(x)| > \frac{1}{2}$ for every x.
- **Prior.** Independent Gaussian weights: standard deviation $\sigma_1 = \sqrt{N/d}\,\sigma$ (first layer), $\sigma_\ell = \sigma$ (inner), $\sigma_L = \sigma/\sqrt{N}$ (output) — choice explained later. This talk is the **small (strong) prior limit**, $\sigma \to 0$ at fixed N.
  - We don't actually consider any sort of limiting learner. For each N, we just take σ tiny enough that when learning any function f, the tail of functions with larger weighted ℓ^2 norm is irrelevant.
- **Learning is conditioning.** $(x, 1)$ conditions on $\{z_w(x) > \frac{1}{2}\}$; $(x, -1)$ on $\{z_w(x) < -\frac{1}{2}\}$.

Two complexity measures

Definition: the MLP-circuit complexity of f at depth L

$C_L(f)$ is the smallest C such that some MLP of depth L and width C, with all weights of absolute value at most C, implements f.

This is never infinite: one hidden layer with 2^d neurons can already interpolate all inputs exactly; larger L pads with near-identity layers (cf. Appendix B).

For any reasonable circuit convention — at least given constant fan-in — a function implemented by a depth-L circuit of complexity C has $C_L(f) = O(C)$. Conversely, a function with a small MLP-circuit has a small circuit in any reasonable convention, though there we may have to fudge the complexity polynomially, and the depth. So $C_L(f)$ is related to other fixed-depth circuit complexities of f. (First direction: Appendix B.)

And what the learner ≈minimizes is the weighted squared ℓ^2 norm matching that prior, $-\log(\text{prior density}) = \kappa_N(w)/2\sigma^2 + \text{const}$ with $c_1 = d/N$, inner $c_\ell = 1$, $c_L = N$:

$$\kappa_N(w) = \textstyle\sum_\ell c_\ell \|W^\ell\|^2 \quad \text{for a weight vector, and} \quad \kappa_N(f) = \inf\left\{\kappa_N(w) : w \text{ implements } f\right\}$$

for a function ($\|\cdot\|$ = Frobenius, and the Euclidean norm of the readout row).

Statement of the main learning theorem of this talk

Theorem

In the NNbayes setting of the setting slide: let $C = C_L(f)$, and let the width N be *anything* at least C — arbitrarily large. Then after $n = \mathrm{poly}(d, C)$ data points, with probability $1 - \exp(-\Omega(C))$ over the draw of the samples, the average probability on the correct answer is $> 99.9\%$.

Throughout, the depth L is held fixed and suppressed inside $\mathrm{poly}(d, C)$. How the bound depends on L is a story of its own, and we come back to it.

It turns out that it suffices to show the following:

Proposition (counting)

For every width $N \geq C$: $\#\{\, g \,:\, \kappa_N(g) \leq \kappa_N(f) \,\} \;\leq\; \exp\big(\mathrm{poly}(d, C)\big)$.

Why it suffices: at tiny σ, only these functions are relevant when learning f — everything costlier is in an irrelevant tail — so the relevance-based bound gives $n = O(\log M) = \mathrm{poly}(d, C)$ samples.

We get there by first proving the counting proposition with the width pinned small — and then bootstrapping.

The counting proposition — first version, at small width

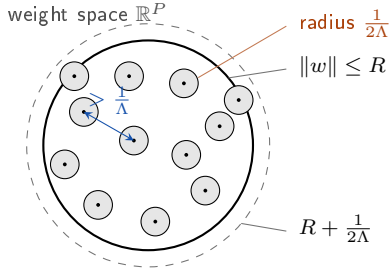
Proposition (counting, small width)

Let $C = C_L(f)$, and take the width N to be at least C and at most $\mathrm{poly}(C)$. Then $\#\{\, g : \kappa_N(g) \le \kappa_N(f) \,\} \le \exp\big(\mathrm{poly}(d, C)\big)$.

By definition of C, we have that f is realizable at width N with norm $\mathrm{poly}(d, C)$ — so every function to be counted has an implementation of norm at most $\mathrm{poly}(d, C)$. There is already something to prove here: the count is of *functions*, but they are implemented by a continuum of weight vectors.

The idea is that varying weights cannot change the function too quickly because of Lipschitzness, so we can put disjoint balls around points implementing different functions, and bound the total number of balls by a volume argument.

The small-width counting argument



One dot per boolean function of norm $\le R = \mathrm{poly}(d, C)$, in $\mathbb{R}^P$ with P the number of parameters — a ball the learner never leaves (in Appendix C's $\sqrt{c_\ell}$-rescaled coordinates, where $\kappa_N(w) = \|w\|^2$). If w, w' implement different functions then $|z_w(x) - z_{w'}(x)| > 1$ at some input x, and $z_w(x)$ is Λ-Lipschitz in w, so the dots sit $> 1/\Lambda$ apart:

$$\#\{\, g : \kappa_N(g) \le R^2 \,\} \;\le\; \left(\frac{R + \frac{1}{2\Lambda}}{\frac{1}{2\Lambda}} \right)^P \;=\; (1 + 2\Lambda R)^P \;=\; \exp\big(\mathrm{poly}(d, C)\big),$$

and that is the small-width counting proposition. (Details: Appendix C.)

High width: κ_N and C_L are polynomially related

Fix f, and now let N be *any* width at least $C_L(f)$ — arbitrarily large.

Theorem (equivalence)

$C_L(f)$ and $\kappa_N(f)$ are *polynomially related*:

$$\kappa_N(f) \;\le\; \mathrm{poly}(d, C_L(f)) \qquad \text{and} \qquad C_L(f) \;\le\; \mathrm{poly}(d, \kappa_N(f)),$$

with polynomials that do not depend on N.

Corollary (the counting proposition, at every width)

For every width $N \ge C_L(f)$: $\#\{\, g : \kappa_N(g) \le \kappa_N(f) \,\} \le \exp\big(\mathrm{poly}(d, C_L(f))\big)$.

We first derive the corollary, and then prove the two halves of the equivalence theorem by two constructions: one we call *cloning*, and one we call *subsampling*.

Proof of the corollary

Take any g with $\kappa_N(g) \le \kappa_N(f)$. The equivalence theorem — used in both directions — gives

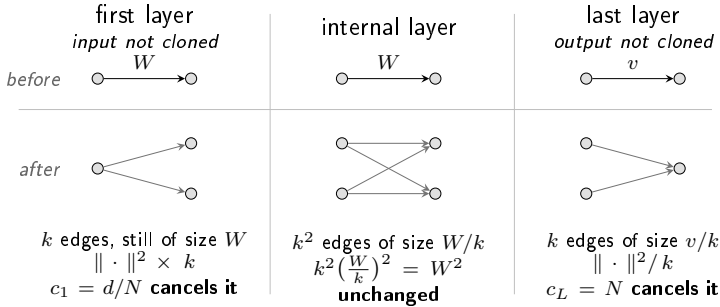
$$C_L(g) \;\le\; \mathrm{poly}(d,\, \kappa_N(g)) \;\le\; \mathrm{poly}(d,\, \kappa_N(f)) \;\le\; \mathrm{poly}(d,\, C_L(f)).$$

So every such g is implemented at width $\mathrm{poly}(d, C_L(f))$ with weights of absolute value at most $\mathrm{poly}(d, C_L(f))$ — and by the Lipschitz-plus-volume count of the small-width case, there are at most $\exp\big(\mathrm{poly}(d, C_L(f))\big)$ of them. □

Cloning: f stays cheap at every width

Claim (first half of the equivalence theorem). At every width $N \geq C = C_L(f)$, $\kappa_N(f) \leq \mathrm{poly}(d, C)$.

Replace every neuron by $k = \lfloor N/C \rfloor$ copies, dividing each weight *out of* a cloned neuron among its k clones; first-layer weights are copied whole. The function is untouched — $\sum_{b=1}^{k} \frac{W}{k}\,\phi(z) = W\phi(z)$ — and the norm does this:



This is the promised explanation of the boundary scalings: $c_1 = d/N$ and $c_L = N$ are exactly what cancels the k and $1/k$ (the d rides along at every width). (Details: Appendix D.)

Subsampling: small norm lets us discard almost all neurons

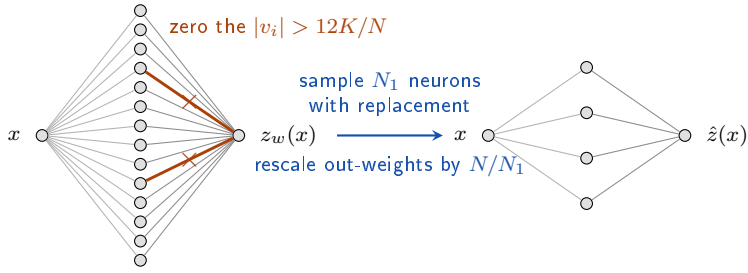
Cloning gives half of the equivalence theorem. To get the other half, i.e. $C_L(f) \le \mathrm{poly}(d, \kappa_N(f))$, we use *subsampling* to shrink any net of small ℓ^2 norm while preserving the function. (What follows is a sketch; see Appendix E for the full argument.)

Proposition (subsampling)

If a width-N net implements g with $\kappa_N(w) \le K$, then g is *also* implemented — at margin $\frac{1}{3}$ — by a net of width $N_1 = \mathrm{poly}(d, K)$ whose weighted norm is $\le 4K$.

This is the second half of the equivalence theorem: the small net's weights are all at most $\mathrm{poly}(d, K)$ (weighted norm $\le 4K$ at width N_1, read entrywise: first layer $\le 2\sqrt{KN_1/d}$, inner $\le 2\sqrt{K}$, output $\le 2\sqrt{K/N_1}$), so — after scaling the output layer by $\frac{3}{2}$ to turn margin $\frac{1}{3}$ back into margin $\frac{1}{2}$ — it witnesses $C_L(g) \le \mathrm{poly}(d, K) = \mathrm{poly}(d, \kappa_N(g))$.

Subsampling, in one picture



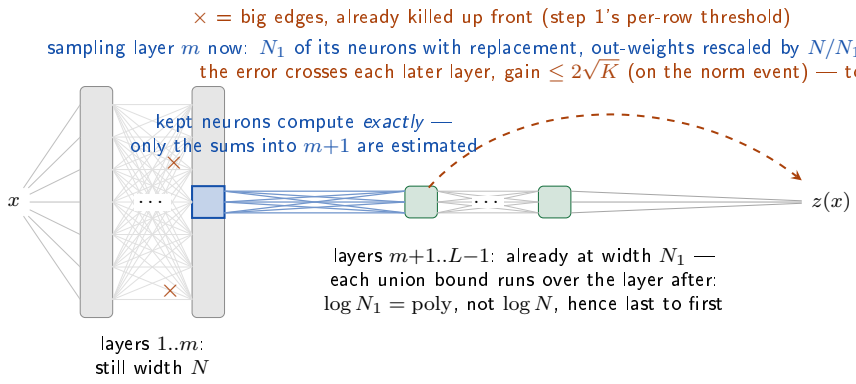
Why it works. $c_L = N$ forces $\sum_i v_i^2 \le K/N$, so the out-weights are tiny on average. Zero the few heavy out-edges — they can shift z_w by at most $\frac{1}{12}$ in total, on any input. What is left is a sum of N small terms, and a random N_1 of them, rescaled, is unbiased with Hoeffding error $\frac{1}{12}$ at every one of the 2^d inputs at once. Margin $\frac{1}{2}$ becomes margin $\frac{1}{3}$.

More hidden layers: the procedure

Error budget $\eta := \frac{1}{12(L-1)}$ per layer. Define $\rho := \eta\, 2^{-L} \min\{K^{-(L-1)/2}, 1\}$.

1. **Kill all big edges.** One deterministic pass over every W^s, $s \geq 2$: in row i, zero every entry with $|W_{ij}| > \tau_i := \frac{2}{\rho\sqrt{N}} \|W_{i\cdot}\|$.
   - *This moves the next preactivation vector in ℓ^2 by at most $\frac{\rho}{2}\sqrt{NK}$, so it moves the output by $\leq \frac{\rho}{2}\sqrt{NK} \cdot (\sqrt{K})^{L-s-1} \cdot \sqrt{K/N} = \frac{\rho}{2} K^{(L-s+1)/2} \leq \frac{\eta}{2}$* — the gains being operator norms, $\leq \sqrt{K}$ inside and $\leq \sqrt{K/N}$ at the output, forced by $\kappa_N(w) \leq K$.
2. **Subsample the hidden layers, every layer to the same size** $N_1 := \lceil 256\, \rho^{-4}\big(d + \log(L/\rho)\big)\rceil$: in stage t we subsample hidden layer $m := L - t$ (last hidden layer first), sampling N_1 neurons *independently*, with replacement, out-weights rescaled by N/N_1. Write W' for the subsampled matrices, S_{m+1} for the kept neurons of the layer after.
   - *Deviations are small:* by Hoeffding, with probability $\geq \frac{99}{100}$ for every stage and input at once, stage t moves the next preactivation vector in ℓ^2 by $\leq \frac{\rho}{2}\sqrt{N \sum_{i \in S_{m+1}} \|W^{m+1}_{i\cdot}\|^2}$.
   - *Kept-row masses stay small:* by Markov, $N \sum_m \sum_{i \in S_{m+1}} \|W^{m+1}_{i\cdot}\|^2 \leq 8 N_1 K$ with probability $\geq \frac{7}{8}$ — so every stage's term is, and the bound above reads $\leq \frac{\rho}{2}\sqrt{8N_1K}$.
   - *Norms stay small:* likewise the subsampled net's weighted norm is $\leq 4K$ with probability $\geq \frac{3}{4}$; hence $\|W'^s\|_{\mathrm{op}} \leq 2\sqrt{K}$ $(1 \leq s < L)$ and $\|W'^L\|_{\mathrm{op}} \leq 2\sqrt{K/N_1}$.
   - *All three at once* — one union bound, probability $\geq 1 - \frac{1}{100} - \frac{1}{8} - \frac{1}{4} > 0$; fix such a draw. *Then each stage moves the output by $\leq \frac{\rho}{2}\sqrt{8N_1K} \cdot (2\sqrt{K})^{t-2} \cdot 2\sqrt{K/N_1} = 2^{t+\frac{1}{2}} \frac{\rho}{2} K^{t/2} \leq \frac{\eta}{2}$,* and the $2(L-1)$ contributions of $\frac{\eta}{2}$ move it by $\leq \frac{1}{12}$ in all. **The subsampled net computes the same signs, at margin $\frac{1}{3}$, at width $N_1 = \mathrm{poly}(d, K)$, with weighted norm $\leq 4K$.**

More hidden layers, in one picture



Each layer has width $N_1 = O\big((d + L\log(K+1))\, L^4\, 16^L\, (K+1)^{2(L-1)}\big)$.

N-free, and the $K^{2(L-1)}$ is forced by the *first* layers: their errors cross all $L-1$ later gains of $\sqrt{K}$. So the proposition goes through — **at constant depth**, just $\mathrm{poly}(d, K)$.

Taking stock

Theorem (again)

Let $C = C_L(f)$, N anything at least C: after $\mathrm{poly}(d, C)$ data points, with probability $1 - \exp(-\Omega(C))$ over the draw of the samples, the average probability on the correct answer is $> 99.9\%$.

Cloning and subsampling made C_L and κ_N polynomially equivalent at every width; the small-width count and the relevance-based bound did the rest.

An alternative route. It turns out that one can also conclude this result by replacing the subsampling step (which is perhaps the mathematically trickiest part of the argument) with a norm-based capacity bound from the literature (Bartlett 1998; Neyshabur–Tomioka–Srebro; Golowich–Rakhlin–Shamir) — but as far as we know, the above theorem is novel. One reason to still care about subsampling is that the same idea also works for other cases of NNbayes, where we don't know how to get such a result using other means. We will say more about this later. (See Appendix F for more on the literature.)

Depth. Probably the main thing that sucks about this theorem is that the poly hides an $\exp(\text{depth})$ term. Can one get rid of that?

Is the $\exp(\text{depth})$ actually necessary?

One can show that the subsampled neuron count really must have an $\exp(\text{depth})$ term: for $K = \tilde{\Omega}(dL^2)$ and N large enough, the ball $\{\, g : \kappa_N(g) \le K \,\}$ genuinely contains $2^{\min\{2^d,\, (K/L)^{\Theta(L)}\}}$ boolean functions. (Appendix G.)

But this is not a conclusive argument that the sample complexity bound must have such a term.

Our guess is that it must — without some further assumption.

A way to remove $\exp(\text{depth})$: restrict the prior to robust weights

Store the activations at finite precision. Now restrict the prior to the **robust** weight vectors: those whose rounded activations do not change when the activations are perturbed by a quarter of a unit in the last place, on any input.

And that no weight is too big: $|W^s_{ij}| \leq \mathrm{poly}(d, C, L) \cdot \|W^s\|_F / N$ — each inner weight within a poly factor of its layer's root mean square weight. The cloned witness satisfies this.

Then a subsample that moves activations by less than that leaves every rounded activation *literally unchanged*: no error to propagate at all.

Theorem

On this prior, the exponential in the depth becomes polynomial:
the same guarantee from $\mathrm{poly}(d, C, L)$ data points, at every depth.

(C read as the width of the smallest *robust* implementation of f.) See Appendix H for an explanation of why the first condition above alone is insufficient.

These are sorta contrived restrictions, put in to make the proof work. A more principled variant: require a small Lipschitz constant under perturbing activations (obtainable for boolean implementations by building in *denoising* layers). But we'd like better ideas here.

Additional results

- **An online mistake bound.** Instead of the PAC statement, we can also prove: when doing *in-distribution online learning* — predict each fresh sample from $\mathcal{D}$ before seeing its label — the learner makes only $\mathrm{poly}(d, C_L(f))$ mistakes in expectation.
- **A bound in the other direction.** The *circuit* complexity C_L of the learned function is at most $\mathrm{poly}(d, \#\text{mistakes made while learning it})$. That is: if learning works well, the function you end up with is simple.

Part 3: Discussion

The big picture

A broader research aim

Establish that/when the neural net Bayesian prior *is* a circuit complexity prior.

So far, everything was for one choice of prior. In general there is one standard deviation per layer — σ_1 at the input, σ_ℓ in the intermediate layers, σ_L at the readout — so there is a whole space of scalings to survey, and which learner you get depends on where you sit in it. We conjecture that only three kinds of point live there — next slide.

The big picture: three scalings

We single out three scalings, listing each as (first layer σ_1, inner layers σ_ℓ, output layer σ_L):

- **strong prior**: $\sigma_1 = \sqrt{N/d}\,\sigma$, $\sigma_\ell = \sigma$, $\sigma_L = \sigma/\sqrt{N}$, with the free parameter $\sigma = N^{-\gamma/2}$ for some $\gamma > 1$. This presentation considered the $\gamma \to \infty$ case.
- **mean field**: $\sigma_1 = 1/\sqrt{d}$, $\sigma_\ell = 1/\sqrt{N}$, $\sigma_L = 1/N$ — the $\gamma \to 1$ point just outside the strong prior regime.
- **NNGP** (neural network Gaussian process): $\sigma_1 = 1/\sqrt{d}$, $\sigma_\ell = 1/\sqrt{N}$, $\sigma_L = 1/\sqrt{N}$.

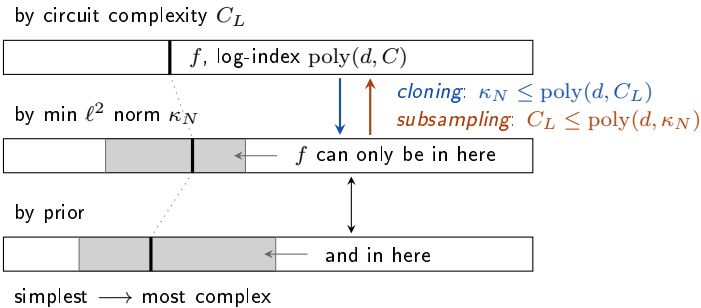
The big picture: what we expect

Our partly speculative picture is that:

- Every "reasonable" overparametrized NNbayes scaling resembles one of the three. By "reasonable", we mean that after conditioning on data, each layer has a constant fraction of neurons at $\Theta(1)$ preactivations — activations are not very small and also not saturated. (We currently have no rigorous argument for this, though we do have a handwavy one — Appendix I.)
- The strong prior and mean-field cases have a circuit prior and learn any target from $\mathrm{poly}(d, \text{its circuit complexity})$ samples.
  - Dmitry has figured out how to establish this in the strong prior case (write-up upcoming) and in the infinite-width mean-field case (Vaintrob & Hänni, `arXiv:2607.05735`).
- The NNGP case does *not* have a good circuit prior — XORs in particular are circuit-simple but slow for it to learn. Our heuristic picture is that it fits boolean functions in the Fourier basis, preferring low degree, so a size-k XOR takes $\sim d^k$ samples where $\mathrm{poly}(d, k)$ ought to do. (Appendix J has more discussion.)

The three orderings, scrambled only polynomially

Order all boolean functions three times over; a function's *log-index* is $\log \#\{g \text{ at least this simple}\}$.



Every bound runs both ways, so a log-index s in one ordering forces a log-index in $[\,(sd)^{1/a}, (sd)^a\,]$ in the others.

Open problems

- **Improve the polynomial.** We have not tried to optimize it. Perhaps *quadratic* in C_L is reachable — linear seems too good, there being $\sim \exp(C_L^2)$ functions at width C_L — or linear in some *other* circuit complexity, though that probably needs different ideas. (Matching the $O(C \log(C + d))$ of Solomonoff induction over boolean circuits looks out of reach: specifying the input directions the C first-layer neurons read is already $\sim dC$ bits — for a full XOR, $\sim d^2$. Not a lower bound, but why we expect a genuine d.)
- **Kill the $\exp(\text{depth})$.** We suspect this is impossible by default (open direction: understand the prior better; maybe it's roughly just the one-hidden-layer prior?), but we also suspect that there is some reasonable robustness assumption on the support of the prior that makes this true. We are not content with our current candidates for this assumption.
- **Give a lower bound on the log prior of a function.**
  - This bound probably must decay with width or be bad in the circuit complexity. This is because the NNGP does poorly on XORs (Appendix J) and we suspect that any scaling which does not give the NNGP has $\pi(f)$ decaying in width.
  - But even if that's right, one can ask for a bound that decays in width only slowly.

  Best such result we know of: Buzaglo et al. (ICML 2024) — see Appendix F.

More open problems

- **Classify and analyze the NNbayes regimes.** Either find scalings that behave in a genuinely new way, or prove that every reasonable scaling reduces to one of the regimes we can already deal with, or prove circuit prior theorems for other scalings directly (Appendix I is our current heuristic reduction.) Also, do other reasonable ways to set up NNbayes give similar behavior? (We think mostly yes.)
- **Out of distribution.** The in-distribution guarantee is simply false out of distribution — there are easy counterexamples. An *online mistake bound* might still hold, and we have a heuristic argument for one that we suspect is "morally right" — but the analogous claim for circuits is strictly false. (Appendix K.)
- **Other architectures.** State and prove analogous results for RNNs, transformers, CNNs. Oh, and also just MLPs with other activation functions (what we need of ϕ: Appendix L).
- **Gradient descent.** Do any of the results or ideas here extend to the local learning algorithms people actually use?

We think many of these are within reach — and we encourage people looking for concrete neural net learning theory problems to work on them.

Thank you for listening

Kaarel and Dmitry would like to thank the following for discussions and ideas:

Lucius Bushnaq, Sam Eisenstat, Lawrence Chan, Jake Mendel, Lauren Greenspan, Claude, ChatGPT

Appendices

Warning! Many of these appendices have largely been generated by Claude from bullet-point outlines and rough notes provided by me. I haven't meticulously reviewed many appendix slides yet, so there could be some errors, sorry.

- **A**: the relevance-based sample complexity bound, with free parameters
- **B**: implementing circuits inside MLPs
- **C**: the small-width counting argument, in detail
- **D**: cloning, in detail
- **E**: subsampling, in detail
- **F**: to what extent is this already in the literature?
- **G**: constructing $\exp\exp(L)$ functions of low norm
- **H**: why quantization alone does not kill the $\exp(\text{depth})$
- **I**: a handwavy typology of large-width NNbayes scalings
- **J**: what does NNbayes do in the NNGP regime?
- **K**: out of distribution
- **L**: wherefore our conditions on ϕ

Appendix A: the relevance-based bound, in general form

The setting: a countable class $\mathcal{H}$ of functions $g : X \to Y$ with prior π, a target $f \in \mathcal{H}$, data $x_1, \ldots, x_n \sim \mathcal{D}$ i.i.d. with labels $y_i = f(x_i)$, and the Bayesian posterior. Here S is the set of non-tail hypotheses of the body's statement, and the error levels are free.

Proposition (relevance-based sample complexity bound)

Suppose there is a set $S \subseteq \mathcal{H}$ with

(i) $f \in S$, $\pi(f) > 0$, and $|S| \le M$;
(ii) $\pi(\mathcal{H} \setminus S) \le T\,\pi(f)$.

Then for any $\varepsilon, \delta, \eta > 0$: after $n \ge \frac{1}{\varepsilon}\big(\log M + \log(2+T) + \log\frac{1}{\delta} + \log\frac{1}{\eta}\big)$ samples, w.p. $\ge 1 - 2\delta$ the posterior's average probability on the correct answer, $\mathbb{E}_{x\sim\mathcal{D}}\big[\text{posterior}\{g : g(x) = f(x)\}\big]$, is $\ge 1 - (\varepsilon + \eta)$; the posterior-majority predictor then errs on at most a $2(\varepsilon+\eta)$-fraction of inputs.

The 2 in $\log(2+T)$ only keeps that term nonnegative. The body's statement follows from this proposition's *proof* run at $\varepsilon = \eta = T = \frac{1}{2000}$, $\delta = \frac{1}{2000M}$: the union-bound and Markov requirements need only hold separately, and each reduces to $e^{-8} < \frac{1}{2000}$ — whence the body's $4000(\log M + 4)$.

Appendix A: proof

Call g *bad* if $\mathrm{err}(g) > \varepsilon$. A bad hypothesis is consistent with one fresh sample with probability $\leq 1 - \varepsilon$, hence with all n with probability $\leq e^{-\varepsilon n}$.

Inside S: union bound. $\Pr[\text{some bad } g \in S \text{ survives}] \leq M e^{-\varepsilon n} \leq \delta$ at the stated n. On the complement, every bad hypothesis in S has posterior mass *zero*.

Below S: Markov. The expected surviving prior mass of the *bad part* of $\mathcal{H} \setminus S$ is $\leq T\,\pi(f)\,e^{-\varepsilon n}$, while the posterior denominator is always $\geq \pi(f)$ (f never dies). So w.p. $\geq 1 - \delta$, the posterior mass of the bad part of $\mathcal{H} \setminus S$ is $\leq T e^{-\varepsilon n}/\delta \leq \eta$.

Average, and majority. On the intersection, every surviving bad hypothesis lies in $\mathcal{H} \setminus S$, with total posterior $\leq \eta$, and each good one is wrong at a fresh x with probability $\leq \varepsilon$ — so the average probability on the correct answer is $\geq 1 - (\varepsilon + \eta)$. The majority prediction is wrong at x only if the correct answer carries $\leq \frac{1}{2}$ of the posterior there; Markov over x gives the $2(\varepsilon + \eta)$-fraction. □

Appendix A: the classical special case, and why a posterior statement

The classical special case. For a *deterministic* learner that returns a simplest consistent hypothesis, the corresponding guarantee — $n \geq \frac{1}{\varepsilon}\big(\log M(C(f)) + \log \frac{1}{\delta}\big)$ for $M(s) := \#\{g : C(g) \leq s\}$ — is the classical Occam bound of Blumer–Ehrenfeucht–Haussler–Warmuth 1987; it is our proposition's union-bound half, with $S = \{g : C(g) \leq C(f)\}$.

Why we do not use that form. Our $\sigma \to 0$ posterior need not have a unique minimizer to return: it concentrates on consistent weight vectors with $\kappa_N(w)$ approaching $\kappa_N^* := \inf\{\kappa_N(w) : w \text{ consistent}\}$ — an infimum that need not be attained — and (at least a priori) several *functions* may sit at κ_N^*, by a symmetry of the input, say. The posterior is then spread over them. The posterior-mass conclusion needs no minimizer named: S is only ever *counted*.

Appendix A: how this talk instantiates it

In our setting $\mathcal{H}$ is the boolean functions implementable at the given width, and $S = \{\, g : \kappa_N(g) \le \kappa_N(f) \,\}$ — of size $\exp(\mathrm{poly}(d, C))$ by the equivalence theorem and the small-width count, which is what cloning and subsampling are for.

For the tail condition: every g with $\kappa_N(g) > \kappa_N(f)$ has prior mass vanishing *relative to* f*'s* as $\sigma \to 0$, and there are finitely many boolean functions at fixed d — so for σ small enough they form a tail of mass $< \pi(f)/2000$. At fixed small $\sigma > 0$, bounding the tail is genuinely open — the "bound the prior itself" problem from the open problems list.

When $\log M + \log(2 + T) \le \mathrm{poly}(d, C_L(f))$ at a given scaling — as we conjecture for the whole strong prior family and for mean field — every target is learned from $\mathrm{poly}(d, C_L(f))$ samples, with no single simplest hypothesis, just a quickly decaying prior.

Appendix B: implementing a circuit inside an MLP

Claim. A depth-L circuit of size C with constant fan-in has $C_L(f) = O(C)$, up to ± 1 in the depth convention — with the boolean values carried *exactly*, not approximately.

The construction. Each gate's value is carried by a *linear combination* of $O(1)$ neurons — the form universal approximation naturally produces. Such combinations compose at no cost: a neuron's preactivation is an arbitrary linear functional of the previous layer's activations, so any affine function of several gate values is again a legitimate preactivation.

Where the thresholds come from. The construction needs a constant available at every layer. Layer 1 reads it off the bias input; each later layer devotes one neuron to carrying $\phi(\beta) \neq 0$ forward. One extra neuron and $O(1)$ norm per layer, so still width $O(C)$.

Appendix B: building a gate inside an MLP

A gate of fan-in q has $Q = 2^q = O(1)$ input patterns $v^{(1)}, \dots, v^{(Q)}$, all distinct. We want Q neurons and coefficients with

$$\sum_{s=1}^{Q} \lambda_s \, \phi\big(a_s \cdot v^{(p)} - \theta_s\big) \;=\; \mathrm{gate}\big(v^{(p)}\big) \qquad \text{for all } Q \text{ patterns } p,$$

where a_s, θ_s and λ_s are all ours to choose.

Theorem (Pinkus 1999, Thm. 5.1)

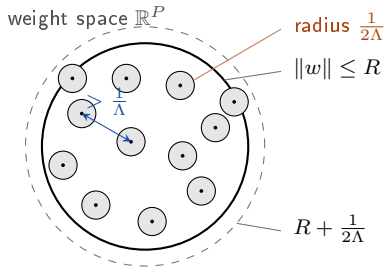
Let $\phi \in C(\mathbb{R})$ be non-polynomial. For any r distinct points $\xi^1, \dots, \xi^r$ and any values $\alpha_1, \dots, \alpha_r$, there are a_s, θ_s, λ_s with $\sum_{s=1}^{r} \lambda_s \, \phi(a_s \cdot \xi^p - \theta_s) = \alpha_p$ for every p.

Pinkus gives no bound on the weights, but none is needed: the interpolation problems range over a fixed finite family — exact boolean patterns, constant fan-in — so their solutions are constants depending only on ϕ.

Appendix C: the small-width count, restated

Proposition (counting, small width)

Let $C = C_L(f)$ and take the width N to be at least C and at most $\mathrm{poly}(C)$. Then $\#\{\, g : \kappa_N(g) \le \kappa_N(f) \,\} \le \exp\big(\mathrm{poly}(d, C)\big)$.



To apply the relevance-based bound we want a bound on $M(\kappa_N(f))$, the number of functions implementable at norm at most f's own. Three steps: the learner never leaves a ball of radius R; inside it, different functions sit $> 1/\Lambda$ apart; disjoint balls are counted by volume.

Appendix C: the learner stays in a ball

Coordinates. Rescale coordinate ℓ of weight space by $\sqrt{c_\ell}$. Then $\kappa_N(w) = \|w\|^2$ is an ordinary squared Euclidean norm on $\mathbb{R}^P$, P the number of parameters, and sublevel sets of κ_N really are balls.

f is cheap. At width $N \geq C$, embed the width-C net implementing f and set the remaining weights to 0. Its weights are at most C and there are $\mathrm{poly}(d, C)$ of them; the coefficients $c_1 = d/N$ and $c_L = N$ are themselves $\mathrm{poly}(d, C)$ at this width, so

$$\kappa_N(f) \;\leq\; \mathrm{poly}(d, C) \;=:\; R^2.$$

So the learner stays inside $\|w\| \leq R$: the functions with $\kappa_N(g) > R^2$ are *together* irrelevant — that is the strong prior tail condition — and only $\{g : \kappa_N(g) \leq R^2\}$ remains to count.

Appendix C: different functions are far apart

The separation. Suppose w, w' in the ball implement *different* boolean functions. Then they disagree at some input x, and both have margin, so — after swapping names if need be — $z_w(x) > \frac{1}{2}$ and $z_{w'}(x) < -\frac{1}{2}$, giving $|z_w(x) - z_{w'}(x)| > 1$. Since $w \mapsto z_w(x)$ is Λ-Lipschitz on $\|w\| \le R$,

$$\|w - w'\| \;>\; 1/\Lambda.$$

This is exactly what the $\pm\frac{1}{2}$ margin in the prior restriction bought us: without it, two weight vectors could implement different functions and be arbitrarily close.

And Λ is not too big. Perturbing one layer changes the output by at most the perturbation times the operator norms of the layers after it and the Lipschitz constant of ϕ — all bounded on the ball — so $\Lambda \le \mathrm{poly}(N) \cdot O(R)^L$. Only $\log \Lambda$ will enter the count, so this costs us nothing.

Appendix C: comparing volumes

One representative per function. Each g with $\kappa_N(g) \le R^2$ has an implementation in the ball — a subtlety: κ_N being an infimum, we should allow *non-strict* implementation here, margins $\ge \frac{1}{2}$ rather than $> \frac{1}{2}$ (take a limit point of implementations). Pick one arbitrarily for each g.

Disjoint, and contained. Put a radius-$\frac{1}{2\Lambda}$ ball around each chosen point. These balls have disjoint interiors — the separation argument, run with the non-strict margins, gives $\|w_g - w_{g'}\| \ge 1/\Lambda$ — and they are all contained in the bigger ball $B(0, R + \frac{1}{2\Lambda})$. Comparing volumes, there are not too many points:

$$\#\{\, g \;:\; \kappa_N(g) \le R^2 \,\} \;\le\; \left(\frac{R + \frac{1}{2\Lambda}}{\frac{1}{2\Lambda}} \right)^P \;=\; \big(1 + 2\Lambda R\big)^P.$$

Appendix C: what the count gives

At width $N \le \mathrm{poly}(C)$ we have $P = \mathrm{poly}(d, C, L)$ and $R^2 = \mathrm{poly}(d, C, L)$, and Λ enters only through its logarithm — so even $\Lambda \sim R^L$ is harmless:

$$M\big(\kappa_N(f)\big) \;\le\; \#\{\, g : \kappa_N(g) \le R^2 \,\} \;\le\; \exp\big(\mathrm{poly}(d, C, L)\big).$$

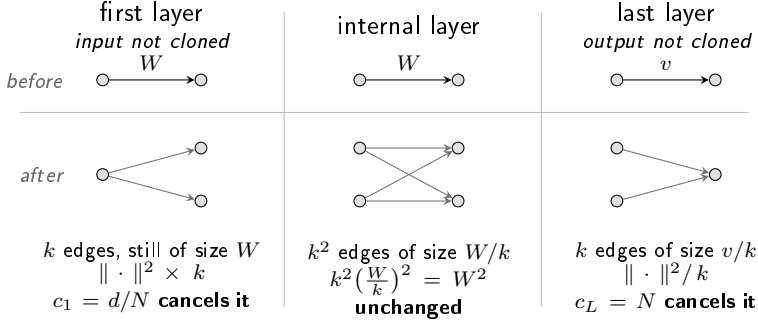
The relevance-based bound then gives $n = \mathrm{poly}(d, C, L)$. □

Note where this breaks. P is the number of *parameters*, so the bound degrades as the net gets wider — and says nothing at all in the overparametrized limit. That is the real problem, and cloning plus subsampling is how we get around it.

Appendix D: cloning, restated

Proposition

If f is implemented by a depth-L, width-C net with all weights at most C, then for every width $N \geq C$ we have $\kappa_N(f) \leq \mathrm{poly}(d, C)$ — with no dependence on N.



The next three slides give the construction, the norm accounting for the internal layers, and the two boundary layers — which is where the coefficients c_1 and c_L come from.

Appendix D: the construction

The construction. Put $k = \lfloor N/C \rfloor$. Replace each neuron i by k copies $i_1, \ldots, i_k$, and wire *every* copy of j to *every* copy of i, with weight W_{ij}/k. If C does not divide N, zero the leftover $N \bmod C$ neurons; the accounting below then holds up to a factor < 2, which $\mathrm{poly}(d, C)$ absorbs.

It computes the same function. Suppose all k copies of each neuron j in one layer carry the original activation $\phi(z_j)$. Then in the next layer, every copy of a neuron i has preactivation

$$\sum_j \sum_{b=1}^{k} \frac{W_{ij}}{k} \, \phi(z_j) \;=\; \sum_j W_{ij} \, \phi(z_j) \;=\; z_i,$$

so the next layer again computes the same function at each clone of a neuron. By induction on layers, the net still implements the same function.

Appendix D: the norm is preserved

Each original weight W_{ij} has become k^2 weights, each of size W_{ij}/k. So its contribution to the squared ℓ^2 norm is

$$k^2 \cdot \left(\frac{W_{ij}}{k}\right)^2 \;=\; W_{ij}^2 :$$

exactly what it was before. Summing over the layer, the squared norm is *unchanged*, however large N is.

The original net has $\mathrm{poly}(C)$ weights, each at most C — plus dC in the input layer — so this accounting gives

$$\sum_{\ell \text{ internal}} \|W^\ell\|^2 \;\le\; \mathrm{poly}(d, C) \qquad \text{at every width } N.$$

That is the accounting for *internal* layers only — the calculation is different for the first and last layer.

Appendix D: the boundary layers

The two ends of the net are not symmetric under cloning, because the input and the output are not themselves cloned:

- **First layer.** The inputs x_j are not duplicated, so each copy of neuron i needs the *full* weight W_{ij} — k copies at full size. The raw squared norm grows by $k = N/C$.
- **Last layer.** The single output reads all k copies of i, so each weight must be v_i/k — k copies at $1/k$ the size. The raw squared norm shrinks by k.

This is what fixed the boundary scalings of our prior. In the weighted ℓ^2 norm

$$\kappa_N(w) \;=\; \sum_{\ell} c_\ell \, \|W^\ell\|^2, \qquad c_1 = \frac{d}{N}, \quad c_\ell = 1 \;\text{ for } 1 < \ell < L, \quad c_L = N,$$

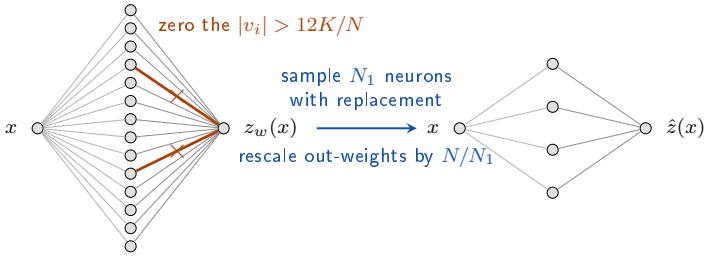
the coefficients exactly cancel the k and $1/k$ above, so cloning preserves the weighted norm at *every* layer — and the proposition holds as stated. □

In prior terms these are exactly the standard deviations we fixed at the start: $\sigma_1 = \sqrt{N/d}\,\sigma$, inner $\sigma_\ell = \sigma$, $\sigma_L = \sigma/\sqrt{N}$.

Appendix E: subsampling, restated

Proposition

If a width-N net implements g with $\kappa_N(w) \le K$, then g is *also* implemented — at margin $\frac{1}{3}$ — by a net of width $N_1 = \mathrm{poly}(d, K)$ (L fixed, as throughout) whose weighted norm is $\le 4K$ (so $\kappa_{N_1}(g) \le 9K$ after the $\frac{3}{2}$ rescale to margin $\frac{1}{2}$).



One hidden layer first, then the general case, where the error injected at one layer has to cross all the later layers.

Appendix E: the concentration bound we use

Proposition (Hoeffding)

Let $X_1, \dots, X_n$ be independent with $|X_k| \le B$, and $S = \sum_k X_k$. Then for every $t > 0$,

$$\Pr\big[\,|S - \mathbb{E}S| \ge t\,\big] \;\le\; 2\exp\Big(-\frac{t^2}{2nB^2}\Big).$$

Hoeffding 1963, Theorem 2, in the $a_k \le X_k \le b_k$ form: take $b_k - a_k \le 2B$ and union the two tails. (Textbook proof: Boucheron–Lugosi–Massart, *Concentration Inequalities*, Theorem 2.8.) In words: *a sum of independent, individually small terms is very likely to be close to its expectation.* The sum could be as large as nB, but deviations past $\sim B\sqrt{n}$ are exponentially unlikely, and the exponent sees the terms only through B.

Every estimator below has this shape: sampling $n = N_1$ neurons with replacement makes the terms independent, and each is bounded by $\frac{N}{N_1}\tau$, since the zeroing capped the weights at τ and $|\phi| \le 1$. **That is what step 1 is for —** it buys a small B, and the widths cancel inside it.

Appendix E: one hidden layer, the estimator

Write u_i for neuron i's incoming weights and v_i for its outgoing one:

$$z_w(x) \;=\; \sum_{i=1}^{N} v_i\, \phi(u_i \cdot x).$$

With $c_L = N$ the complexity charges $N \sum_i v_i^2$, so $\sum_i v_i^2 \le K/N$.
Concentration will come from Hoeffding, which needs each term of the sum to be small — most are, but a few need not be, so neutralize those first.
Zero the heavy out-edges. Take $\tau = 12K/N$ and set $v_i := 0$ wherever $|v_i| > \tau$.
Those edges had $|v_i| \le v_i^2/\tau$, so together they contributed at most $\big(\sum_i v_i^2\big)/\tau \le \frac{1}{12}$ to $z_w(x)$ in absolute value, on *any* input x (using $|\phi| \le 1$, per the setup). Write z_{light} for what is left — the neurons stay, only their out-edges go.
Subsample. Pick N_1 of the N neurons at random, with replacement, and rescale:

$$\hat{z}(x) \;=\; \frac{N}{N_1} \sum_{i \in S} v_i\, \phi(u_i \cdot x), \qquad |S| = N_1, \qquad \mathbb{E}[\hat{z}(x)] = z_{\text{light}}(x).$$

Appendix E: one hidden layer, concentration

$\hat{z}(x)$ is a sum of N_1 independent terms, each bounded by $\frac{N}{N_1}\tau = 12K/N_1$ — the width has canceled — so Hoeffding gives, for each fixed x,

$$\Pr\Big[\,|\hat{z}(x) - z_{\text{light}}(x)| > t\,\Big] \;\le\; 2\exp\Big(-\frac{t^2 N_1}{2N^2\tau^2}\Big) \;=\; 2\exp\Big(-\frac{t^2 N_1}{288\,K^2}\Big).$$

Union bound over all 2^d inputs. At $t = \frac{1}{12}$ the exponent is $N_1/(41472\,K^2)$, so $N_1 = 165888\,(d+2)\,K^2$ makes some input fail with probability at most $2^d \cdot 2e^{-4(d+2)} < 10^{-3}$ — and this is where d enters the theorem.

The norm. Both coefficients move with the width, and both come out clean: the output layer charges $N_1 \sum_{i\in S}(\frac{N}{N_1}v_i)^2$, of expectation $N\sum_i v_i^2 \le K$, and the input layer $\frac{d}{N_1}\sum_{i\in S}\|u_i\|^2$, of expectation $\frac{d}{N}\sum_i\|u_i\|^2 \le K$ — so the subsampled net's weighted norm has expectation $\le \kappa_N(w) \le K$, and Markov puts it above $4K$ w.p. $\le \frac{1}{4}$. So the probability that *either* fails is at most $10^{-3} + \frac{1}{4} < 1$: a good subsample exists, within $\frac{1}{6}$ of z_w on every input. □

And $\frac{1}{6}$ is enough. A net with $|z_w(x)| > \frac{1}{2}$ subsamples to one with $|\hat{z}(x)| > \frac{1}{3}$, correct sign — and Lipschitz-plus-volume applies verbatim at threshold $\frac{1}{3}$, distinct functions separated by $2/3$.

Appendix E: more hidden layers, the plan

The body's two steps, with its facts refined so the order in which things get determined is explicit: (In this appendix A, B, C name events, not the main thread's $C = C_L(f)$.)

1. **Kill all big edges**, in one deterministic pass — this will make it possible to bound the deviations caused by subsampling. (Next slide.)
2. **The zeroing was safe** (the body's two step-1 facts): since operator norms of later layers are bounded, zeroing a layer's big edges moves the output by at most $\frac{\eta}{2}$, on every input ($\eta := \frac{1}{12(L-1)}$).
3. **Draw the subsamples in order:** S_{L-1} for the last hidden layer first, down to S_1 — each N_1 neurons independently, with replacement.
4. **The deviation event** A (the body's "deviations are small"), about the moments in between: just after layer m is subsampled — layers before it still original — the sums into layer $m+1$'s kept neurons are accurate to $\frac{\rho}{2}\sqrt{N}\|W_{i\cdot}\|$, on every input. Hoeffding: $\Pr[A] \geq \frac{99}{100}$.
5. **The row-mass event** B: the kept rows carry $N\sum_m \sum_{i\in S_{m+1}} \|W_{i\cdot}^{m+1}\|^2 \leq 8N_1K$. Markov, $\Pr[B] \geq \frac{7}{8}$ — what turns A's bound into $\frac{\rho}{2}\sqrt{8N_1K}$.
6. **The norm event** C, about the net after *all* the subsampling: its weighted norm is $\leq 4K$ (a claim about the weights — κ enters only in fact 6). Markov, $\Pr[C] \geq \frac{3}{4}$ — and C alone bounds every subsampled *operator* norm.
7. **One union bound, then one implication.** $\Pr[A \cap B \cap C] \geq 1 - \frac{1}{100} - \frac{1}{8} - \frac{1}{4} > 0$: fix such a draw. The three *together* imply the output moves by $\leq \frac{1}{12}$ on every input — errors added once, at the end; no conditioning anywhere.

Appendix E: step 1 — one threshold rule

For every layer $s \geq 2$ and every row i of W^s, zero every entry with

$$|W^s_{ij}| \;>\; \tau_i \;:=\; \frac{2}{\rho\sqrt{N}}\,\|W^s_{i\cdot}\|, \qquad \text{the } \textit{same}\ \rho := \eta\, 2^{-L} \min\{K^{-(L-1)/2}, 1\} \text{ everywhere.}$$

A zeroed entry has $|W_{ij}| \leq W^2_{ij}/\tau_i$, so row i's zeroed entries together move its preactivation by at most (using $|\phi| \leq 1$)

$$\frac{\|W^s_{i\cdot}\|^2}{\tau_i} \;=\; \frac{\rho}{2}\sqrt{N}\,\|W^s_{i\cdot}\| \qquad \text{— the row's budget, on any input.}$$

Entries are zeroed, not neurons: a neuron heavy for one row is light for others, so none is ever discarded. And τ_i reads only the original row — the pass is deterministic. (With one hidden layer, the output was a single row, which is why a flat $\tau = 12K/N$ sufficed there.)

The two factors in ρ. The $\min$ binds only for $K < 1$, where $K^{-(L-1)/2} > 1$ would break the chains below (a formality: $\kappa \geq \frac{1}{4}$ always, as $\frac{1}{2} < |z| \leq \sqrt{N}\|W^L\|_2$). The 2^{-L} pays for the factor 2 each *subsampled* layer's operator norm carries on the norm event — fact 5.

Why per row? Rows of one layer differ in scale, and each may only be disturbed in proportion to *its own* norm — that is what makes the Hoeffding exponent row-free (fact 3) and the ℓ^2 aggregate collapse to $\|W^s\|_F$ (fact 1). A single flat τ cannot serve a row of norm $\sqrt{K}$ and one of norm $\sqrt{K/N}$ at once: loose enough to spare the first, it leaves the second's terms far above its budget.

Appendix E: facts 1–2 — the zeroing is safe

The gains, deterministically. In ℓ^2, ϕ contracts — normalize $\mathrm{Lip}(\phi) \le 1$, as for $\tanh$ — and each layer multiplies by an operator norm, which $\kappa_N(w) \le K$ bounds: $\|W^s\|_{\mathrm{op}} \le \|W^s\|_F \le \sqrt{K}$ $(1 < s < L)$ and $\|W^L\|_{\mathrm{op}} = \|W^L\|_2 \le \sqrt{K/N}$ — the output layer is a single row, so its operator norm *is* its Euclidean norm. Zeroing only shrinks the Frobenius norms (and the output row's Euclidean norm), so these bounds survive it. (Frobenius already dominates the operator norm — no row-wise ℓ^1 control needed.)

Fact 1, aggregated. Row i moves by at most its budget $\frac{\rho}{2}\sqrt{N}\|W^s_{i\cdot}\|$, so in ℓ^2 the whole vector moves by

$$\|\Delta z^s\|_2 \;\le\; \tfrac{\rho}{2}\sqrt{N}\left(\textstyle\sum_i \|W^s_{i\cdot}\|^2\right)^{1/2} \;=\; \tfrac{\rho}{2}\sqrt{N}\,\|W^s\|_F \;\le\; \tfrac{\rho}{2}\sqrt{NK}.$$

Fact 2. For $2 \le s \le L-1$, crossing layers $s{+}1, \dots, L{-}1$ and then the output row multiplies by $(\sqrt{K})^{L-s-1}\sqrt{K/N}$:

$$\tfrac{\rho}{2}\sqrt{NK} \cdot K^{\frac{L-s-1}{2}}\sqrt{\tfrac{K}{N}} \;=\; \tfrac{\rho}{2}\,K^{\frac{L-s+1}{2}} \;\le\; \tfrac{\eta}{2}.$$

The width N cancels exactly. (At $s = L$ the zeroed row *is* the output row, so its shift is already at the output: $\le \frac{\rho}{2}\sqrt{N}\|W^L\|_2 \le \frac{\rho}{2}\sqrt{K} \le \frac{\eta}{2}$ — the same bound, no crossing.)

Appendix E: fact 3 — Hoeffding, and the width N_1

Draw the subsamples in order, S_{L-1} first down to S_1, each N_1 neurons *independently*, with replacement. Subsampling layer m estimates each sum into layer $m{+}1$ by an average of N_1 terms; the kept rows of layer m are verbatim, so those sums are all that is estimated.

The exponent is the same everywhere. After zeroing, the terms are bounded by $\frac{N}{N_1}\tau_i$; aiming at the row's budget $\frac{\rho}{2}\sqrt{N}\|W_{i\cdot}\|$, the exponent is

$$\frac{\left(\frac{\rho}{2}\sqrt{N}\|W_{i\cdot}\|\right)^2 N_1}{2\,(N\tau_i)^2} \;=\; \frac{\frac{\rho^2}{4}N\|W_{i\cdot}\|^2\,N_1}{\frac{8}{\rho^2}N\|W_{i\cdot}\|^2} \;=\; \frac{\rho^4 N_1}{32} \qquad \text{— } \|W_{i\cdot}\| \text{ cancels.}$$

The deviation event A: *every* kept neuron of *every* layer is accurate to its row's budget on *every* input. Union bounding over the 2^d inputs, the N_1 kept neurons of the layer after, and the $L-1$ layers needs $\rho^4 N_1/32 \geq d + \log(N_1 L) + 5$ — which the body's $N_1 = \lceil 256\rho^{-4}(d + \log(L/\rho))\rceil$ satisfies, giving $\Pr[A] \geq \frac{99}{100}$ and

$$N_1 \;=\; O\big((d + L\log(K+1))\,L^4\,16^L\,(K+1)^{2(L-1)}\big) \quad \text{— } N\text{-free.}$$

The union runs over the *random* S_{m+1}, drawn at the previous stage — legal because the draws are independent and the exponent does not depend on the row, so the same bound holds for every realization and $\Pr[A]$ is unconditional. On A, stage t moves the next preactivation vector by $\leq \frac{\rho}{2}\sqrt{N\sum_{i\in S_{m+1}}\|W_{i\cdot}\|^2}$, as in fact 1.

Appendix E: fact 4 — the norm event

The row-mass event B. Summed over all stages — one Markov, not one per layer. Each kept set averages its layer, and at the top stage layer L is the single output row, taken whole:

$$\mathbb{E}\sum_m N \sum_{i\in S_{m+1}} \|W_{i\cdot}^{m+1}\|^2 \;=\; N_1 \sum_{1<s<L} \|W^s\|_F^2 \;+\; N\|W^L\|_2^2 \;\le\; N_1\,\kappa_N(w) \;\le\; N_1 K,$$

using $N_1 \ge 1$ on the output term. So all stages' row masses are $\le 8N_1K$ at once, $\Pr[B] \ge \frac{7}{8}$ — which turns fact 3's aggregate into $\frac{\rho}{2}\sqrt{8N_1K}$.

The norm event C. Subsampling is unbiased for every term of the weighted norm: each term is an average over the kept rows and columns of the original one, and the N/N_1 rescalings are eaten by exactly as many N_1/N sampling factors — boundary coefficients included. So the subsampled net's weighted norm has expectation $\le \kappa_N(w) \le K$; Markov puts it $\le 4K$, $\Pr[C] \ge \frac{3}{4}$. And C bounds the crossing gains too: every subsampled inner layer has $\|W'^s\|_{\mathrm{op}} \le \|W'^s\|_F \le 2\sqrt{K}$, the output row $\|W'^L\|_{\mathrm{op}} \le 2\sqrt{K/N_1}$. **The proposition's norm conclusion and the gains come from one event.**

And Markov is all that is true here: the norm is unbiased but *not* concentrated — a κ-legal heavy row drawn into a subsample really does inflate it. All the exponential concentration lives in fact 3, where the zeroing bounds the terms.

Appendix E: fact 5 — the stage crossing

Now combine, at stage t, i.e. subsampling layer $m = L - t$. (The layers after $m{+}1$ are the *same matrices* in the intermediate net as in the final one, so C's operator-norm bounds apply at every stage.) On A each kept neuron of layer $m{+}1$ is accurate to $\frac{\rho}{2}\sqrt{N}\|W_{i\cdot}\|$; on B, $N\sum_{i\in S_{m+1}}\|W_{i\cdot}\|^2 \le 8N_1K$; and on C the crossing gains are $(2\sqrt{K})^{t-2}\cdot 2\sqrt{K/N_1}$. So the output moves by at most

$$\frac{\rho}{2}\sqrt{8N_1K}\cdot\frac{2^{t-1}K^{\frac{t-1}{2}}}{\sqrt{N_1}} \;=\; 2^{t+\frac{1}{2}}\,\frac{\rho}{2}\,K^{\frac{t}{2}} \;\le\; \frac{\eta}{2} \qquad (t \le L-1),$$

the last step by ρ's factor 2^{-L}, since $2^{t+\frac{1}{2}-L} \le 2^{-\frac{1}{2}}$. Again N_1 cancels under the square root.

(At the top stage $t = 1$ the error hits the output row directly, shift $\le \frac{\rho}{2}\sqrt{K}$ — the same bound.)

All three at once. One union bound: $\Pr[A \cap B \cap C] \ge 1 - \frac{1}{100} - \frac{1}{8} - \frac{1}{4} > 0$, so a draw with all three properties exists. Fix one.

Appendix E: fact 6 — totals, and why last to first

The implication, spelled out. Telescope: net H_m has layers $\geq m$ subsampled, layers $< m$ original — so H_L is the zeroed original, H_1 the final net. Consecutive nets differ only in one layer's estimation, applied to *original, exactly computed* neuron functions; fact 5 bounds each H-difference at the output by $\frac{\eta}{2}$, and fact 2 separately bounds each of the $L-1$ zeroing steps (original net $\to H_L$). Errors are summed once, at the end — no conditioning anywhere.

Why the telescope runs last to first. Comparing H_{m+1} to H_m injects error at the N_1 kept neurons of layer $m+1$ — so A's union runs over $N_1 = \mathrm{poly}$ rows per layer, costing $\log N_1$ in its exponent. A first-to-last telescope would inject error at a still-original layer: N rows, and the $\log N$ would poison the width.

Fact 6, the totals. $L-1$ zeroing shifts and $L-1$ stage shifts, each $\leq \eta/2$, move the output by $\leq (L-1)\eta = \frac{1}{12} < \frac{1}{6}$ in all: margin $\frac{1}{2}$ becomes margin $\frac{1}{3}$, at which C bounds this net's weighted norm by $4K$ — hence, entrywise, first-layer weights $\leq 2\sqrt{KN_1/d}$ (the binding ones, since the coefficient there is the small d/N_1), inner $\leq 2\sqrt{K}$, output $\leq 2\sqrt{K/N_1}$: all $\mathrm{poly}(d, K)$, because N_1 is. *Only now* is the function known to be preserved — and to make that a statement about the *function* we must first restore margin $\frac{1}{2}$, which is what κ's infimum ranges over: the $\frac{3}{2}$ output rescale multiplies the output term by $\frac{9}{4}$, so $\kappa_{N_1}(g) \leq 4K + \frac{5}{4} \cdot 4K = 9K$, and that rescaled net is the C_L witness.

So the multilayer proposition holds, and the equivalence theorem's second half goes through — **at constant depth**, where that is just $\mathrm{poly}(d, K)$. But the depth dependence is exponential: each layer crossed contributes a $\sqrt{K}$, and L multiply.

Appendix F: to what extent is this already in the literature?

Most of the machinery used here is not novel. But as far as we know, the bayesian learning result proved in this talk has not been stated before. And the ideas work in more cases than just the strong prior limit studied here — these other cases are also novel as far as we know.

The small-width count is classical. The Lipschitz-plus-volume step is the standard volumetric covering bound, and its consequence — sample complexity polynomial in the *parameter count* — is known for the standard sigmoid: Karpinski and Macintyre bound the VC dimension of sigmoid nets by a polynomial in the number of parameters (*JCSS* 1997), building on Goldberg and Jerrum's bound for classes parametrized by real numbers (1995).

Appendix F: width-independence is not new either

The subsampling step can be replaced by known norm-based capacity bounds (see Bartlett 1998, Neyshabur–Tomioka–Srebro and Golowich–Rakhlin–Shamir).

But we believe the subsampling idea generalizes to other cases of NNbayes as well — and we conjecture that, in some sense, it generalizes to *all* interesting cases of NNbayes. What you have seen is the simplest instance of a more general argument.

Appendix F: cloning, and the closest prior work

Cloning's mechanism is Net2WiderNet — Chen, Goodfellow and Shlens, ICLR 2016. Duplicate a unit and divide the outgoing weights by the replication factor, and the function is preserved exactly. What is not there is the *norm* accounting: that the layer-weighted κ comes out unchanged, and that requiring this at the two ends is what *forces* $c_1 \propto 1/N$ and $c_L = N$.

The closest prior work is Buzaglo et al. (ICML 2024): the same Bayesian-interpolator framing and a "sample complexity $\approx$ teacher complexity" conclusion — but their bound degrades *linearly in the student width*, which is precisely the dependence we remove.

Same object, other angles. Our κ_N is a layer-weighted analogue of the *representation cost* $R_L(f)$ of Parkinson, Ongie, Willett, Shamir and Srebro, who prove depth separations in it. For two-layer ReLU nets the min-norm infinite-width interpolant is characterized by Savarese et al. (COLT 2019) and, multivariately, by Ongie et al. (ICLR 2020).

So: width-independence is a consequence; the headline is norm $\leftrightarrow$ circuits.

Appendix F: is the NNbayes prior a Kolmogorov prior?

A related line of work argues the neural net parameter–function map is biased toward *Kolmogorov-simple* functions: Dingle, Camargo and Louis (2018) bound $P(f) \leq 2^{-a\tilde{K}(f)+b}$ for generic input–output maps via the coding theorem; Valle-Pérez, Camargo and Louis (ICLR 2019) bring this to deep nets; Mingard et al. argue SGD approximately samples this prior (JMLR 2021), and that nets have "an inbuilt Occam's razor" (Nature Communications 2025).

But only one direction of that can hold. Bounds of this shape say: large prior mass $\Rightarrow$ compressible. A genuine Kolmogorov prior would also need the converse — every $\tilde{K}$-simple function gets mass $\sim 2^{-\tilde{K}}$ — and that is impossible. There are functions with $O(\log d)$-bit descriptions (diagonalize against all small circuits, by brute force) whose circuit complexity is $2^{\Omega(d)}$: they have no small circuits, so NNbayes will very likely *not* learn them quickly — and in our strong prior regime the prior tracks κ_N, which subsampling ties to circuit complexity, so their mass sits far below $2^{-\tilde{K}}$.

The warm-up's programs-versus-circuits distinction again: the NNbayes prior charges for *computation*, not just description. On small instances the two are hard to tell apart empirically; they separate exactly on the computation-heavy functions.

Appendix G: constructing $\exp\exp(L)$ functions of low norm

The claim. The $\exp(\text{depth})$ in our counting bound is not an artifact of the proof. It is really there — for $K = \tilde{\Omega}(dL^2)$:

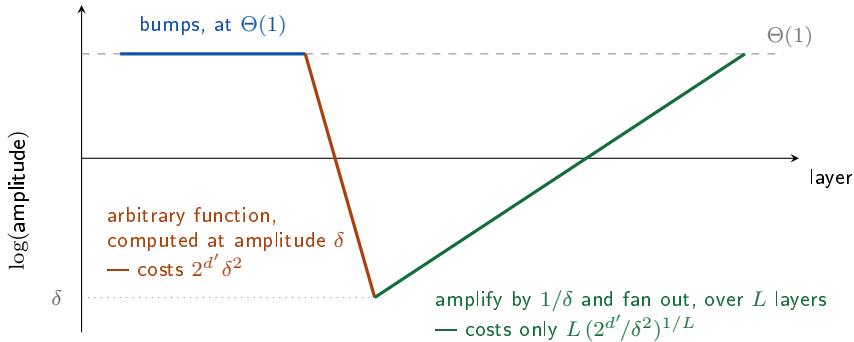
$$\log_2 \#\{\, g : \kappa_N(g) \le K \,\} \;\ge\; \min\Big\{\, 2^d,\; (K/L)^{\Theta(L)} \Big\}.$$

The method. We exhibit weight vectors. For *every* boolean function on a subcube of $d' = \Theta(L \log(K/L))$ of the d coordinates — taking $K = \tilde{\Omega}(dL^2)$; smaller K caps d' at $\sim\sqrt{K/d}$ — we build a w with $\kappa_N(w) \le K$ implementing it. There are $2^{2^{d'}}$ such functions, so the ball is at least that rich.

Why it is possible. Computing an arbitrary function is expensive, but only if you insist on an answer of size $\Theta(1)$. So compute at a tiny amplitude δ, where it is nearly free, and then spend the depth amplifying δ back up — and depth makes amplification exponentially cheap.

Three steps, then: a cheap first layer, an arbitrary function at amplitude δ, and amplification back to $\Theta(1)$.

Appendix G: why depth makes this cheap



Computing is expensive only if you insist on a $\Theta(1)$ answer. Compute tiny, then climb back — and the climb is cheap because we can be gaining exponentially in amplitude over L layers. The steeper the drop you can afford, the more functions fit in the ball: δ down means $2^{d'}$ up.

Appendix G: the blow-up construction

Budget K, depth L, width $N = 2^{d'+1}$ (one bump per point, plus a constant bank of the same size — see the fine print), working on a subcube of d' of the d coordinates. We implement an *arbitrary* boolean function on it.

Layer 1: one bump per point — cheap. Take $\phi\big(\gamma(\langle x^{(i)}, x\rangle - (d'-1))\big)$, which is $\phi(\gamma)$ at $x^{(i)}$ and $\phi(\leq -\gamma)$ elsewhere; $\gamma = \Theta(\log d')$ keeps even the sum of all $2^{d'}$ soft tails negligible. With $\|u_i\|^2 = \gamma^2\big(d' + (d'-1)^2\big)$ — the offset rides on the bias input —

$$\text{cost} \;=\; \frac{d}{N}\sum_{i=1}^{N}\|u_i\|^2 \;=\; \tilde{O}(d\,d'^2) \qquad \text{— the source of the } \sqrt{K/d} \text{ cap on } d'.$$

Layer 2: read them all, at tiny amplitude. One unit forms $s(x) = \sum_i v_i \phi(u_i \cdot x) = \delta f(x)$, at cost $\sum_i v_i^2 = \Theta(2^{d'}\delta^2)$. Small output, small weights, cheap — and an arbitrary boolean function has now been computed.

Appendix G: the blow-up construction

Layers $3, \dots, L-1$: fan out to N clones while amplifying by $G = 1/\delta$. A layer of gain a_ℓ going from n_ℓ to $n_{\ell+1}$ units costs $a_\ell^2 r_\ell$, with $r_\ell := n_{\ell+1}/n_\ell$; minimizing $\sum_\ell a_\ell^2 r_\ell$ subject to $\prod_\ell a_\ell = G$, $\prod_\ell r_\ell = N$ gives cost $m\,(G^2 N)^{1/m}$ over the $m = L-3$ amplifying layers; we write L for m below, hiding a constant. This is legitimate for a saturating ϕ: every step runs in the linear regime $\phi(az) \approx \phi'(0)az$, and only the last saturates.

Output layer. N units carrying ± 1, weights $1/N$: cost $N \cdot N \cdot N^{-2} = 1$, margin $\Theta(1)$ — so the prior's restriction to full boolean functions is met.

Any wider net works too. We wrote this at $N = 2^{d'+1}$, the smallest width that fits the $2^{d'}$ distinct bumps together with the constant bank. At a larger width, clone the whole net — each bump included — which leaves the weighted norm unchanged. So the same d' is reached at every $N \geq 2^{d'+1}$, and the bound is not a statement about one special width.

Appendix G: balancing the two costs

Balancing. Take $2^{d'}\delta^2 \approx K$, so $G \approx 2^{d'/2}/\sqrt{K}$ and the fan-out cost is $m(4^{d'}/K)^{1/m} \le K$ over the $m = L-3$ amplifying layers:

$$4^{d'} \;\le\; K\Big(\frac{K}{m}\Big)^{m} \qquad\Longleftrightarrow\qquad d' \;\approx\; \frac{m}{2}\log_2\frac{K}{m}, \qquad m = L-3.$$

The depth enters through the *boosting*, which evades the $c_L = N$ penalty: emitting the answer straight out of layer 2 would need $|v_i| \sim 1$, hence cost $\sim 4^{d'}$. A bump reads $\phi(-\gamma)$ off its own point rather than 0, leaving a $\Theta(\delta 2^{d'})$ offset at layer 2; a bank of $2^{d'}$ constant units absorbs it at the same $\Theta(K)$ cost — one unit alone would cost $K2^{d'}$. For scale: at $L = 10$, $d = 40$, $K = 3\times 10^6$, sharing the budget across the three stages honestly gives $d' = 40$ — the full cube — and almost all boolean functions on 40 bits need a circuit of size $\sim 3\times 10^{10}$.

Appendix G: what it does and does not settle

Proposition

For $L \geq 4$, $K > L$ and $N \geq 2^{d'+1}$, every boolean function on $d' = \tilde{\Theta}\big(\min\{L\log(K/L),\ \sqrt{K/d}\}\big)$ coordinates has $\kappa_N \leq K$ — the cap comes from the first layer's $\tilde{\Theta}(d\,d'^2)$ bill. Hence, for $K = \tilde{\Omega}(dL^2)$, $\log_2 \#\{g : \kappa_N(g) \leq K\} \geq \min\{2^d, (K/L)^{\Theta(L)}\}$.

This matches the ceiling from the other side: subsampling to $N_1 = \mathrm{poly}(d, K)K^{O(L)}$ and counting gives $\log \# \leq \tilde{O}(N_1^2 L)$, i.e. $K^{O(L)}$.

What it settles. The $\exp(\text{depth})$ in the width we subsample to is not an artifact. A width-N_1 net carries $\tilde{O}(N_1^2 L)$ bits, while the ball already holds $2^{d'}$ of them — so subsampling, with no further assumption on what the posterior is supported on, cannot reach any width below $(K/L)^{\Theta(L)}$.

What it does not settle. It does not show that ℓ^2 min-norm learning is bad here. For that, these functions would have to sit at or below the minimum norm over hypotheses consistent with the data — and a circuit-simple target may well have a cheaper implementation than any of them. Our guess is that things are not nice here, but we have no proof.

Appendix H: why quantization alone is not enough

The gap. Drop the no-big-weights condition of the body's robust-prior slide and keep only quantization. Then *this proof route dies*: there is an explicit family of *robust* nets with $\kappa_N \le K$ realizing *every* boolean function on d bits, so the count is back to $\exp(\text{depth})$ and no $\mathrm{poly}(d, K, L)$-width compression can exist. (Whether the *learning* statement is false we do not know — that needs prior mass, not cardinality.)

What quantization was supposed to stop. Appendix G's blow-up hides an arbitrary function in a tiny *amplitude* δ, then spends the depth amplifying it back. Quantization forbids exactly that: an activation of size $\delta \ll \delta_q$ rounds to 0.

But amplitude is not the only hiding place. Hide the function in a tiny relative *cancellation* instead. A reader neuron sums $\sim N$ terms that cancel down to a residual of size δ_q — a relative cancellation of 2^{-d} — while every activation in the net is 0, $\pm\delta_q$, or within $\delta_q/4$ of ± 1: margin $> \delta_q/4$ at every neuron, and $\delta_q/2$ at the signal.
The margin constrains each neuron's distance from a bucket boundary. It says nothing about its sensitivity to perturbations of its *inputs*. And quantization does not merely fail to bite — it *helps*: rounding collapses the "off" cluster into one bucket, which is what makes that cancellation exact.

Appendix H: the construction, layers 1 and 2

Write δ_q for the quantization step and $\beta = \delta_q/4$ for the required margin. Representatives are the integer multiples of δ_q, boundaries the half-integer multiples, rounding is to nearest. Fix a target $g : \{\pm 1\}^d \to \{\pm 1\}$ and set $\alpha := \delta_q$, $z_t := \phi^{-1}(\alpha)$, $c' := z_t/\alpha = 1 + O(\delta_q^2)$.

Layer 1: a *replicated* bump bank, and a saturated constant bank. n copies of each of the 2^d point bumps $\phi\big(\gamma(\langle x_p, x\rangle - (d-1))\big)$, together with $n2^d$ units of bias γ; width $N = 2n2^d$. The steepness obeys $\gamma > \operatorname{atanh}(1 - \delta_q/4) = \frac{1}{2}\ln(8/\delta_q) + O(\delta_q)$, independent of d — this is what makes all d "off" shells round into the *same* bucket -1. Cost $\frac{d}{N}\|W^1\|_F^2 = \frac{d}{2}\gamma^2(d^2 - d + 2) = \Theta(\gamma^2 d^3)$, independent of n and of N. (That is $c_1 = d/N$ doing its work: a 2^d-entry dictionary for $\mathrm{poly}(d)$.)

Layer 2: readers. n_2 neurons, each carrying weight $Wg(p)$ on every copy of bump p, with $W = z_t/(2n)$. Because the off-shells all round to -1, the background is one number independent of x, and the constant bank cancels it *exactly, on every input* — not on average. Preactivation $z_t g(x)$, hence activation $\alpha g(x)$.

Appendix H: the construction, fan-out and output

Layers $3, \dots, L-1$: fan out at *constant* amplitude. Layer $j{+}1$'s neurons each read all n_j neurons of layer j — which all compute g, so they all fire together — with weight c'/n_j. The preactivation is $z_t g(x)$ again, i.e. α is an exact fixed point of $\mathrm{round} \circ \phi \circ (c' \cdot)$: no drift, at any depth. Layer cost $c'^2 n_{j+1}/n_j$; writing κ_{fan} for the total spent on these $L-3$ layers, AM–GM lets the fan-out $F := n_{L-1}/n_2$ reach $\big(\kappa_{\mathrm{fan}}/(c'^2(L-3))\big)^{L-3}$ — exponential in the depth, for a fixed bill.

Output, and padding. The output reads the n_{L-1} final neurons with equal weights, at cost $\Theta\big(N/(\alpha^2 n_{L-1})\big)$, and $|z_{\mathrm{out}}| > \frac{1}{2}$. Layers $2, \dots, L-1$ are padded out to width N with zero rows: activation 0, which is a representative, at cost 0.

The bucket alignment is load-bearing, and it is forced. Three values have to clear the margin at once: 0 (the padding, unavoidable since the live layers are narrower than N), $\pm\alpha$ (the signal) and ± 1 (the saturated bank). Only the integer-representative grid hosts all three — put the signal mid-bucket instead and both 0 and ± 1 land on boundaries. This also forces $\delta_q = 1/m$ for an integer $m \geq 2$, so the claim is "for suitable δ_q", not "for all".

Appendix H: the budget, and the count

Let $\mu = 1 + \left(\sum_p g(p)\right)^2 / 4^d \in [1, 2]$ be the reader overhead and B what is left of K for the reader and output layers. The reader and output bills multiply, and in their product n, N, α and δ_q all cancel identically:

$$\text{(i) } \tfrac{\gamma^2 d^3}{2} + \kappa_{\text{fan}} + B \le K, \qquad \text{(ii) } 4^d \le \frac{2B^2 F}{\mu c'^2}, \qquad \text{(iii) } N \ge n_{L-1} \ge F.$$

Since $g \mapsto f$ is the identity here, $\log_2 \#\{\text{robust } f : \kappa_N(f) \le K\} \ge 2^d$, which is $K^{(L-1)/2 - o(1)}$ at fixed depth and $2^{\Theta((K/\gamma^2)^{1/3})}$ with the depth free. A net of width N_1 carries $\tilde{O}(N_1^2 L)$ bits, so no $\mathrm{poly}(d, K, L)$ width suffices. **The fan-out is the whole source of this:** at $F = 1$, i.e. $L = 3$, (ii) gives merely $2^d \lesssim K$.

Verified end to end by a script that builds the weights explicitly, rounds between layers, and checks the computed function, the margin at *every* neuron (bumps, constants, readers, each fan-out layer, padding) and κ_N from the definition: $d = 3, 4, 5$, $\delta_q \in \{\frac{1}{4}, \frac{1}{5}, \frac{1}{10}\}$, fan-out depth 3 and 5, random and worst-case g. E.g. at $d = 4$, $\delta_q = \frac{1}{5}$, $K = 4000$: $\kappa_N = 1832$, worst margin 0.100 against the required $\beta = 0.05$, function exact on all 2^d inputs.

Appendix H: and what the weight condition does to it

Every layer of the construction is a small *live block* inside an $N \times N$ frame: $n_{j+1} \times n_j$ nearly equal entries, zeros elsewhere. For such a layer

$$\|W^j\|_F = c'\sqrt{n_{j+1}/n_j}, \qquad \max_{ik} |W^j_{ik}| = c'/n_j, \qquad \text{so} \qquad \frac{\max_{ik} |W^j_{ik}|}{\|W^j\|_F/N} = \frac{N}{\sqrt{n_j n_{j+1}}}.$$

At the bottom of the fan-out, where the live block is smallest, this is at least $\sqrt{F}$ (and $F^{1-1/(2(L-3))}$ at the AM–GM optimum), so by (ii) at least $2^d/(K\sqrt{2})$: the violation is exactly as large as the count it buys. Requiring that no weight sit more than a poly factor above its layer's root mean square therefore caps F at poly, and then (ii) caps 2^d at poly: the construction collapses to the $L = 3$ case it started from.

Why relative to the layer's own norm. Asking for small weights outright buys nothing: every inner weight here is already P/N with an N-independent P, so all of them vanish as the net widens. What is exponential in the depth is P. Equivalently: the condition forbids a layer from routing the function through a vanishing fraction of its width, which is precisely what the fan-out chain does.

Appendix I: a handwavy large-width NNbayes typology

The prior has one variance σ_ℓ^2 per layer, so *a priori* the space of scalings is L-dimensional. What actually lives in it?

The standing assumption. *After conditioning the prior on a small number of data points (x, y) from a boolean function*, preactivations are $\Theta(1)$ at every hidden layer: activations neither *collapsed* — ϕ never leaves its linear-ish regime and the layer is nearly affine — nor *saturated* — every neuron ± 1 and the derivative gone. This is a condition on where the posterior sits, not on a typical prior draw — the two coincide only at NNGP. Everything below classifies the scalings compatible with it.

The protocol. Condition on data and sample the layers *in order*, each given the ones before. By the chain rule the total log prior cost is a sum of per-layer conditional costs, and by the time we reach layer ℓ the incoming activation vector $\phi^{\ell-1}$ is a fixed vector rather than something to solve for self-consistently.

One thing we grant ourselves: the first layer is a fixed random embedding $\mathbb{R}^d \to \mathbb{R}^N$ with $\sigma_1 = \Theta(1/\sqrt{d})$, which gives $\Theta(1)$ activations. Everything else is up for grabs.

Appendix I: what a layer costs

A row enters only through $z_i = \langle W_{i\cdot}, \phi^{\ell-1} \rangle$, so for an isotropic Gaussian row the whole N-dimensional prior collapses to one scalar:

$$z_i \sim \mathcal{N}(0, \tau_\ell^2), \qquad \tau_\ell \;=\; \sigma_\ell \big\| \phi^{\ell-1} \big\| \;=\; \Theta\big(\sigma_\ell \sqrt{N}\big).$$

τ_ℓ is the preactivation scale the prior supplies *for free*.

Asking a constant fraction of a layer's neurons to clear $|z_i| \geq A$ then costs

$$0 \;\text{ if } \tau_\ell = \Theta(1), \qquad\qquad \Theta\left(\frac{N}{\tau_\ell^2}\right) \;\text{ if } \tau_\ell \ll 1.$$

The readout is a *single* neuron, with no averaging available to it. So

$$F_{\text{hid}} \;=\; \Theta\left(\frac{1}{\sigma_\ell^2}\right), \qquad F_L \;=\; \Theta\left(\frac{1}{N\sigma_L^2}\right),$$

each when that layer is paying at all.

The bounds are tight only up to a constant per neuron, i.e. $e^{O(N)}$ per layer. So comparisons decide whenever per-neuron costs differ by $\omega(1)$ — which is everything below.

Appendix I: three reductions

1. The *hidden* σ_ℓ are $\leq N^{-1/2}$. Above that $\tau \gg 1$ and a typical draw saturates; nothing pushes back — a saturated net computes fine and the prior prefers it.

2. Only the per-layer prices matter. With m data points, a layer that *computes* must clear the bar in m different directions, cost $\propto m/\sigma_\ell^2$; a layer that *copies* makes them parallel, cost $\propto 1/\sigma_\ell^2$. So the scaling enters only through the *price* $1/\sigma_\ell^2$ it puts on each layer's work. (Which layer carries which price still matters once a layer can run collapsed and be rescaled downstream — that is the next three slides.)

3. The spread decides, by one case check. Prices within a poly of one another act as one price, up to poly. A superpolynomial step *up* in σ is ruled out by the standing assumption: there the posterior prefers to compute at collapsed amplitude and rescale. So σ is non-increasing — and at the first superpolynomial step *down*, the pricier tail can only copy, so the net truncates to its flat-price prefix. (Next three slides.)

So we are down to two knobs: the internal σ, and the readout σ_L.

Appendix I: the price argument

In the paying region the learner ≈minimizes the *weighted* norm $\kappa_\sigma = \sum_\ell \|W^\ell\|^2/\sigma_\ell^2$ — the overall scale is shared by every candidate function, so only the relative prices $1/\sigma_\ell^2$ are visible.

Moderate spread ⇒ strong prior, up to poly. Everything in this talk is per-layer for the *internal* layers — cloning preserves each internal $\|W^\ell\|^2$ (the boundary terms only at their canonical prices, Appendix D) — so for internal prices within a poly of one another, every bound runs for κ_σ with poly-degraded constants. (*Moderate* = the same power of N up to $\mathrm{poly}(d, C_L)$ factors; *extreme* = different powers.)

Extreme spread is a case check on the direction of the step — and neither direction makes a new regime. A step up in σ (a tight layer feeding a looser one) is *excluded by the standing assumption*: the posterior prefers to collapse the tight layer's activations and rescale in the looser one (next slide). So σ is non-increasing along the net. A step down leaves the tail with no cheaper layer after it to rescale: it can only copy, and the flat prefix finishes the computation (slide after next).

Appendix I: a step up in σ is excluded by the assumption

The worrying case: an earlier inner layer *tighter* than a later one, $\sigma_\ell \ll \sigma_{\ell+1}$ ($p_\ell := 1/\sigma_\ell^2 \gg p_{\ell+1}$); one adjacent pair suffices.

What the posterior prefers there. Run layer ℓ at amplitude t, let layer $\ell+1$ scale back up. If the computation uses the degree-k part of ϕ, the nonlinear content of layer ℓ's output is suppressed by t^{k-1}, and layer $\ell+1$ must dig it back out:

$$\mathrm{cost}(t) \;=\; t^2 p_\ell \;+\; t^{-2k} p_{\ell+1}, \qquad t_* = \Big(\tfrac{k\,p_{\ell+1}}{p_\ell}\Big)^{\frac{1}{2k+2}}, \qquad \mathrm{cost}_* = \Theta\big(p_\ell^{\frac{k}{k+1}}\, p_{\ell+1}^{\frac{1}{k+1}}\big).$$

Going linear is not a cliff, just the t^{-2k} recovery price; the saving $(p_\ell/p_{\ell+1})^{1/(k+1)}$ is unbounded, so nothing internal stops the collapse.

But the preferred configuration has layer ℓ at amplitude $t_* \to 0$: collapsed activations, precisely what the standing assumption excludes. A step up is not a new regime; it is a scaling whose posterior walks out of the classified region. So within the assumption, the internal σ is non-increasing.

(The assumption-compatible alternative — the tight layer only copies — is the next slide's truncation. `scaling_pricecheck.py`: interior optimum at every ratio; slopes 0.7498, -0.1251 vs the predicted $\frac{3}{4}$, $-\frac{1}{8}$ at $k=3$; t_* falls $0.38 \to 0.02$ over ratios $10^2 \to 10^{12}$.)

Appendix I: a step down in σ truncates the net

So the internal σ is non-increasing. Take the *first* superpolynomial step down: the prefix before it is one flat price — that is what those layers are in fact like, up to poly — and every layer after it is superpolynomially pricier.

The tail cannot compute within poly budgets. Paying its price outright is out, and the collapse trick of the previous slide needs a *cheaper later* layer to do the rescaling — with σ non-increasing there is none. So the tail can only copy: function-independent overhead, invisible to every comparison.

The prefix can finish. By universal approximation, whatever the full net was going to compute, the flat-price prefix can compute by itself and hand forward. So the learner is that of a flat-σ net at the prefix depth: one of the three regimes, at a possibly smaller effective depth.

Fine print. Functions cheap only at the full depth are not lost, just repriced: through the tail they cost superpoly, through the shallow prefix exponential — either way invisible at the poly budgets everything here runs at, so the reduction is exact at the classification's up-to-poly grain.

Appendix I: the strong prior case is a line, not a region

Take $\sigma \ll N^{-1/2}$, so the hidden layers pay, and suppose the readout pays as well (if not, mass moves into it — the first off-line case below). Neither live cost may dominate, so equate them:

$$\frac{1}{\sigma^2} = \Theta\left(\frac{1}{N\sigma_L^2}\right) \qquad \Longleftrightarrow \qquad \boxed{\sigma_L = \Theta\left(\frac{\sigma}{\sqrt{N}}\right).}$$

So this is a *one*-parameter family, not a two-parameter region — matching this talk's internal and readout coefficients $(1, N)$, σ the free parameter (the input layer sat outside the protocol).

Both sides of the line reduce rather than being new. Above it, mass moves into the cheaper readout and the hidden preactivations fall below $\Theta(1)$: the hidden layers go linear, the net effectively shallower. Below it the readout can only afford to copy: the last hidden layer already carries the answer — one fewer layer (off the line, the reduction fires again and cascades).

The line is exactly the set of scalings at which a depth-L net uses all L layers.

Appendix I: the readout axis at $\sigma = N^{-1/2}$

Now the hidden layers sit exactly at their free threshold and pay nothing; only σ_L varies. Write $\sigma_L^2 = N^{-\beta}$, so $F_L = \Theta(N^{\beta-1})$.

$\beta \leq 1$: **nothing happens.** $\tau_L = \Omega(1)$, so the $\pm\frac{1}{2}$ margin is an $\Theta(1)$-probability event and costs nothing; rescaling the output changes the classifier only up to constants (exactly not at all in the $\tau_L \to \infty$ limit).

$1 < \beta < 2$: **still NNGP, and for a specific reason.** The margin is now a large deviation, so it is *not* that things still look Gaussian. But the hidden layers are untouched, so the feature map — hence the kernel, its eigenbasis and its eigenvalues — is exactly the NNGP one. Only the *estimator* moves: conditioning a Gaussian on a far-away convex set concentrates it on the least-RKHS-norm point, so the posterior slides from GP classification toward the *hard-margin SVM in the NNGP kernel*. Same spectrum, same d^k for parities.

$\beta = 2$: **mean field.** Forcing the readout costs $\Theta(N^{\beta-1})$; moving the features costs $\Theta(N)$, since each hidden neuron sits at $\tau = \Theta(1)$ and a functionally real change needs $\Theta(N)$ of them. Equal exactly at $\beta = 2$.

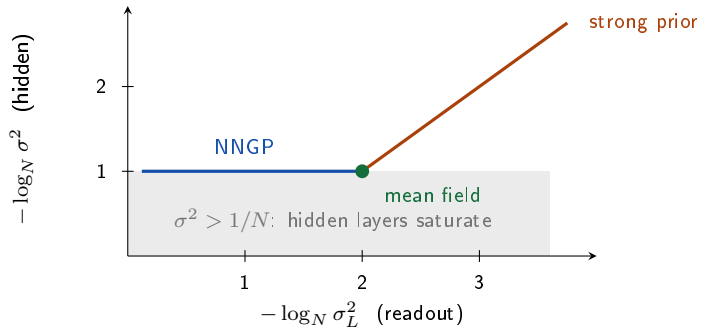
Appendix I: the classification

σ	σ_L	name	what it is
$N^{-1/2}$	$\Omega(N^{-1/2})$	NNGP	GP classification; the margin is vacuous
$N^{-1/2}$	$(N^{-1}, N^{-1/2})$	NNGP	same kernel, estimator slides to max-margin
$N^{-1/2}$	N^{-1}	mean field	the junction of the two families
$\ll N^{-1/2}$	$\sigma/\sqrt{N}$	strong prior	ℓ^2 min-norm; $\sigma \to 0$ is this talk

Two one-parameter families, joined at mean field.
On the first the kernel is fixed and the estimator varies; on the second the estimator is fixed and the prior strength varies.

Handwaving used. That a real feature change needs $\Theta(N)$ neurons is a scaling guess. The price argument identifies off-line profiles with strong prior only up to polynomial factors and effective-depth changes — it does not exclude them. The step-up case is where the assumption does real work: those scalings are *excluded*, not classified — their posterior prefers collapsed activations, outside the three regimes. And all of this prices *one* configuration; whether the cost *varies* across functions, which is what separates kernel from min-norm, is separate.

Appendix I: the space of scalings



What is left of the L-dimensional space after the reductions: one horizontal segment (NNGP — hidden layers at fan-in, kernel fixed while the readout price varies) and one diagonal ray, $\sigma_L^2 = \sigma^2/N$ (strong prior — readout *variance* N times smaller than hidden), meeting at mean field. Everything else either truncates to a shallower flat-σ net (steps down in σ) or leaves the standing assumption — saturating in the shaded region, collapsing at any step up in σ.

Appendix J: what does NNbayes do in the NNGP regime?

The talk is about the $\sigma \to 0$ end of the strong prior family; this appendix is about the NNGP.

In the NNGP scaling, the posterior is a kernel method and we can identify the kernel *exactly*. It is the **Fourier kernel**: diagonal in the parities, eigenvalues depending only on the degree. So the whole inductive bias of the regime is a preference for low Fourier degree.

The claim

In the NNGP regime there is an inductive bias, but it is the wrong one:
a bias toward low *degree* is not a bias toward low *complexity*.

Degree and circuit size are not comparable in the direction that matters: XOR on k bits has a circuit of size $O(k)$ and degree k — the largest possible for a function of k bits. A prior that charges for degree charges maximally for some of the very simplest functions — and no compositional path talks it out of that.
(Our own $\exp(\text{depth})$ costs nothing here: a k-bit XOR is symmetric, so k thresholds do it in *one* hidden layer, and our bound is $\mathrm{poly}(d, k) \ll d^k$.)

Appendix J: what a kernel method does

(In this appendix K denotes a kernel, not the main thread's norm budget.) A kernel is a covariance, $K(x, x') = \mathrm{Cov}(z(x), z(x'))$ under the prior — hence two arguments. Pinning the second one at a training point gives n single-input *kernel functions*

$$k_i(x) \;:=\; K(x, x_i), \qquad i = 1, \ldots, n,$$

and the method regresses on those: $h = \sum_i \alpha_i k_i$.

Equivalently, diagonalize: $K(x, x') = \sum_S \lambda_S \psi_S(x) \psi_S(x')$ gives the feature map $x \mapsto (\sqrt{\lambda_S}\, \psi_S(x))_S$, and $h = \sum_S c_S \sqrt{\lambda_S}\, \psi_S$ with $\|h\|^2_{\mathcal{H}} = \sum_S c_S^2$.

We are heavily overparametrized: there are 2^d features and only n constraints, so interpolation does not pin h down. The method returns the *minimum-*$\|c\|_2$ solution, $h(x) = K(x, \mathbf{X}) K(\mathbf{X}, \mathbf{X})^{-1} y$, with $\mathbf{X}$ the training inputs. In our case $\psi_S = \chi_S$ and $\lambda_S = \lambda_{|S|}$, so $\|h\|^2_{\mathcal{H}} = \sum_S \hat{h}(S)^2 / \lambda_{|S|}$: high degree is penalized by $1/\lambda_k$.

This is exactly the NNGP posterior mean — Lee, Bahri, Novak, Schoenholz, Pennington and Sohl-Dickstein, *Deep Neural Networks as Gaussian Processes*, ICLR 2018.

Appendix J: our $\frac{1}{2}$ rule versus the posterior mean

We condition on the *events* $\{z_w(x_i) > \frac{1}{2}\}$ and predict by posterior majority — the *median*; the NNGP setting conditions on *equalities* $z(x_i) = y_i$ and predicts by the *mean*. GP classification versus GP regression: no closed form, and median and mean can disagree.

But the conclusion crosses. The eigenstructure belongs to the *prior*, so changing the likelihood does not touch it. Redo the estimate on the prior side: for z_w to implement χ_S with margin its χ_S-coefficient must be $\Theta(1)$, and under the prior that coefficient is Gaussian with variance $\lambda_{|S|}$, so

$$-\log \pi\big(w \text{ implements } \chi_S\big) \;\approx\; \frac{\Theta(1)}{\lambda_{|S|}} \;\approx\; d^{|S|},$$

so our relevance-based accounting cannot kick in before $\sim d^k$ samples for a degree-k parity — a prior-mass statement, exactly what we condition on.

Caveat: this bounds the prior mass from *above*, so it limits what the relevance-based bound can promise rather than proving d^k are needed; the genuine lower bound we cite is for kernel ridge regression.

Appendix J: the NNGP kernel is the Fourier kernel

Take $G = (\{1, -1\}^d, \odot)$ under the Hadamard product, so $G \cong (\mathbb{Z}/2)^d$ and every element is its own inverse. Its characters are exactly the parities

$$\chi_S(x) = \prod_{i \in S} x_i, \qquad S \subseteq [d],$$

i.e. the Fourier basis of boolean functions.

Claim

For any MLP with i.i.d. Gaussian weights, the NNGP kernel on the hypercube is G-invariant, hence diagonal in the parities.

Two steps: an invariant kernel is always diagonalized by the characters, and the NNGP kernel is invariant. The NTK turns out to be in the same boat.

Appendix J: an invariant kernel is diagonal in the characters

Call K *G-invariant* if $K(g \odot x, g \odot x') = K(x, x')$ for all g. Equivalently $K(x, x') = \tilde{\varrho}(x \odot x')$ for some $\tilde{\varrho} : G \to \mathbb{R}$, using $x^{-1} = x$. Let $(C_K h)(x) = \mathbb{E}_{x'}\big[K(x, x')\, h(x')\big]$, the average over the cube.

Theorem

Every character χ is an eigenvector of C_K, with eigenvalue $\lambda_\chi = \mathbb{E}_z\big[\tilde{\varrho}(z)\chi(z)\big]$.

Proof. Substitute $z = x \odot x'$, a bijection for fixed x, so $\chi(x') = \chi(x)\chi(z)$:

$$(C_K\chi)(x) \;=\; \mathbb{E}_{x'}\big[\tilde{\varrho}(x\odot x')\,\chi(x')\big] \;=\; \mathbb{E}_z\big[\tilde{\varrho}(z)\,\chi(x)\chi(z)\big] \;=\; \mathbb{E}_z\big[\tilde{\varrho}(z)\chi(z)\big]\,\chi(x). \qquad \square$$

The 2^d characters are orthogonal in a 2^d-dimensional space, so they are a full eigenbasis.

Appendix J: the NNGP kernel is invariant

The NNGP recursion for an MLP with i.i.d. Gaussian weights is

$$\Sigma^{\ell}(x,x') \;=\; \sigma_w^2\,\mathbb{E}_{(u,v)\sim\mathcal{N}(0,Q^{\ell-1})}\big[\phi(u)\phi(v)\big] + \sigma_b^2,$$

with $Q^{\ell-1}$ the 2×2 Gram matrix $\big(\Sigma^{\ell-1}(x_i,x_j)\big)_{x_i,x_j\in\{x,x'\}}$ — so $\Sigma^{\ell}(x,x')$ depends on the previous layer only through those three Gram entries.

On the hypercube the diagonal is constant. $\langle x,x\rangle = d$ for every x, so $\Sigma^1(x,x)$ is a constant, and by induction so is every $\Sigma^{\ell}(x,x)$. Hence $\Sigma^{\ell}(x,x') = F_{\ell}(\Sigma^{\ell-1}(x,x'))$ for a univariate F_{ℓ}, and therefore $K(x,x') = \varrho(\langle x,x'\rangle)$.

And the Hadamard action preserves the inner product, since $g_i^2 = 1$:

$$\langle g\odot x,\; g\odot x'\rangle \;=\; \sum_i g_i^2 x_i x_i' \;=\; \langle x,x'\rangle.$$

(G acts on the d genuine coordinates only; the bias input just adds a constant 1 to every inner product, and $d+1$ on the diagonal.) So K is G-invariant, with $\tilde{\varrho}(z) = \varrho(\sum_i z_i)$. Its eigenvalue on χ_S is $\lambda_S = \mathbb{E}_z\big[\varrho(\sum_i z_i)\chi_S(z)\big]$, which by permutation symmetry depends only on $|S|$.

Appendix J: the same holds for the NTK

The argument used exactly one property: *the kernel sees the inputs only through their Gram matrix*. Nothing cared which recursion produced it.

The NTK recursion is of that form too:

$$\Theta^{\ell}(x,x') \;=\; \Sigma^{\ell}(x,x') \;+\; \Theta^{\ell-1}(x,x')\cdot \dot{\Sigma}^{\ell}(x,x'), \qquad \dot{\Sigma}^{\ell}(x,x') = \mathbb{E}\big[\phi'(u)\phi'(v)\big],$$

the expectation over the same $\mathcal{N}(0, Q^{\ell-1})$. Every ingredient depends on the inputs only through the Gram entries, and sums and products of functions of $\langle x, x'\rangle$ are again functions of $\langle x, x'\rangle$. So $\Theta(x,x') = \varrho_{\Theta}(\langle x, x'\rangle)$, and the parities are its eigenfunctions too, with eigenvalues graded by degree.

So the low-degree-fit picture is equally our picture of what *gradient descent* does in the linearized regime. Not needed for the Bayesian story, but it is the same phenomenon — and indeed the lower bound we cite is stated for the NTK, which is legitimate to use here precisely because the two kernels share an eigenbasis and a degree-graded spectrum.

Appendix J: so what does the NNGP learn?

The kernel is diagonal in the parities with eigenvalues λ_k depending only on the degree, so the RKHS norm is

$$\|h\|_{\mathcal{H}}^2 \;=\; \sum_S \frac{\hat{h}(S)^2}{\lambda_{|S|}},$$

and minimum-norm interpolation penalizes degree-k coefficients by $1/\lambda_k$.

A heuristic picture. Kernel regression resolves the eigendirections with $\lambda \gg 1/n$. The degree-$\leq k$ eigenspace has dimension $\sim d^k$, so with n samples the learned function is roughly the Fourier truncation of the target to degree

$$k \;\approx\; \frac{\log n}{\log d}.$$

So it really is "fit the lowest-degree thing consistent with the data so far" — in a soft form, with low degrees strongly preferred rather than a hard cutoff.

Hence the XOR problem, for which there are theorems — next slide.

Appendix J: no kernel at all likes parities

The obstruction is not invariance — it is a trace budget. Normalize $K(x,x) = 1$ and write $\lambda_S := \langle \chi_S, C_K\, \chi_S \rangle$ — the eigenvalue in the invariant case, the diagonal entry in general. The trace is basis-free, so $\sum_S \lambda_S = \mathbb{E}_x K(x,x) = 1$, and there are $\binom{d}{k} \approx d^k$ parities at degree k. So for *any* kernel whatsoever

$$\#\{\, S : \lambda_S \geq 1/n \,\} \;\leq\; n.$$

Since resolving the χ_S direction needs $n = \Omega(1/\lambda_S)$: with n samples at most n of the d^k degree-k parities are learnable at all. A single fixed parity is of course easy for a kernel built for it — which is why any kernel-independent statement has to quantify over a class.

What the invariance adds is only that the λ_S are *equal* within a degree. It upgrades "the average degree-k parity has $\lambda \approx d^{-k}$" to "*every* one has $\lambda \lesssim d^{-k}$" — it removes the escape hatch of a kernel that favors a few chosen parities, rather than being the source of the failure.

Appendix J: the citations

Kernel-independent, class level. Kamath, Montasser and Srebro (COLT 2020): for any class of pairwise uncorrelated functions the dimension complexity is exponential (Example 1, square loss) — parities being the canonical instance. Ben-David, Eiron and Simon (JMLR 2002) supply the other half of the picture: most classes of bounded VC dimension admit no large-margin embedding at all.

For our kernel specifically. Donhauser, Wu and Yang (ICML 2021): for *inner product kernels* $\varrho(\langle x, x' \rangle)$ and the fully-connected NTK at any depth, rotational invariance "induces a bias towards low-degree polynomials in high dimensions", with a lower bound on generalization error; their distributional assumptions cover i.i.d. ± 1 coordinates, so the cube qualifies. Our structural theorem shows the NNGP kernel is of that form: their theorem plus our step, not an analogy.

Quantitatively. Ghorbani, Mei, Misiakiewicz and Montanari: $\tilde{\Omega}(d^k)$ samples for a degree-k target — stated on the sphere, but Yang and Salman's spectra agree across cube, sphere and isotropic Gaussian in high dimensions.

And the spectra are computed in exactly our basis by Yang and Salman: both the NNGP kernel and the NTK, as operators on the boolean cube, "are diagonalized by the natural Fourier basis".

Appendix J: maybe that is just what real nets do?

The objection. Real networks are also terrible at XORs. Abbe, Boix-Adserà and Misiakiewicz *conjecture* SGD's time to be $\tilde{\Theta}\big(d^{\max(\mathrm{Leap}(f),2)}\big)$ — proved for two-layer nets on isotropic Gaussian data under technical assumptions on how SGD is run, open in general.

Here $\mathrm{Leap}(f)$ is the most *new* coordinates any single Fourier term must add, building the support up in the best order. A plain size-k XOR has $\mathrm{Leap} = k$ — the same d^k the kernel pays.

So perhaps the NNGP is simply right about neural nets, and the failure we are complaining about is a failure real networks share.

Appendix J: no — it fails for the wrong reason

The response. Take the balanced tree of pairwise XORs computing XOR_k, with gates $g_1, \ldots, g_r$, and learn

$$f_\varepsilon(x) \;=\; \mathrm{XOR}_k(x) \;+\; \varepsilon \sum_j g_j(x)$$

(from upcoming work of Alexeev and Vaintrob). The degree is still k, but the Fourier support now contains the circuit's intermediate gates $\chi_{\{1,2\}}, \chi_{\{3,4\}}, \ldots$, so no term has to add more than two new coordinates and $\mathrm{Leap}(f_\varepsilon) = 2$:

$$\text{SGD: } \tilde{\Theta}(d^2) \text{ conjecturally,} \qquad \text{kernel: still } \tilde{\Omega}(d^k).$$

Both fail on the bare XOR, for different reasons. A real net fails because nothing clicks — build it a path and it walks up. The NNGP fails because it dislikes high degree, however good the path.

Appendix J: in fairness to the NNGP

We rescued our own regime by restricting the support of the prior, so the same question deserves asking here.

An asymmetry makes it look harder. Ours was a *competitor* problem — too many functions in the ball — and cutting the support is the natural tool for that. The NNGP's is a *target-mass* problem: a degree-k parity has prior mass $\exp(-\Theta(1/\lambda_k)) \approx \exp(-d^k)$ by itself. Conditioning on A rescales every mass by $1/\Pr[A]$, so a restriction keeping both the parity and its low-degree competitors leaves the ratio untouched.

But it is not ruled out. Any restriction making the prior roughly uniform over a small class of low-complexity functions would give good learning (via the relevance-based bound) whatever the prior's shape. Whether a *natural* one does — as opposed to importing a circuit prior, which begs the question — we do not know.

Open problem: is there a natural support restriction under which the NNGP learns simple functions quickly?

Appendix K: out of distribution, and what replaces the theorem

Our theorem is in-distribution, and out of distribution the statement is simply false: nothing stops the learner from being wrong on inputs the training distribution never reaches.

But our results still give a handle here: the equivalence theorem and the count make NNbayes an Occam learner over a discrete, complexity-ordered class, no matter where the inputs come from — so one can now think of neural net out-of-distribution learning as out-of-distribution *Occam* learning. That is the move this appendix makes.

The natural substitute is an online mistake bound. Fix any ordering of all 2^d inputs, predict each one with the current hypothesis before seeing its label, and count mistakes. Is the count $\mathrm{poly}(d, C)$? (In this appendix C is description length in a formula code — a cousin of, not the same as, the talk's C_L.)

A physicist's argument that it is. When the current hypothesis is wrong, presumably a constant fraction of the hypotheses between it and f are wrong too, so each mistake retires a constant fraction of the survivors. There are $2^{O(C)}$ of those, so $O(C)$ mistakes.

In the description-length version of the setting — Occam over boolean circuits, which is the discrete model of our learner — this argument is *false*, and not for an exotic reason.

Appendix K: the conjunction ladder

Theorem

Let $f = x_1 \wedge \cdots \wedge x_d$, so $C(f) = \Theta(d \log d)$. Present the all-ones point, then the all-zeros point, then every remaining input in *ascending Hamming weight*, within each level in the learner's own tie-breaking order. The Occam learner is wrong on *all but at most one* of the 2^d inputs: writing $M^*(f)$ for the worst-case mistake count over input orderings, $M^*(f) = 2^{\Theta(C/\log C)}$.

Why. At stage k the cheapest surviving hypotheses are exactly the $\binom{d}{k}$ conjunctions of k variables. They all cost the same — variable naming is symmetric — and $\mathrm{AND}_{S'}(1_S) = 1$ only if $S' \subseteq S$, impossible between distinct sets of equal size. So each mistake kills exactly one hypothesis while every other survivor is *correct* at that point — the next slide makes this rigorous. The attrition assumption fails maximally, at every single step.

Neither ingredient is contrived: the target is a very neat function, and the ordering is an easy-to-hard curriculum. At $d = 40$ that is $\sim 10^{12}$ mistakes, where the physicist's argument had promised $O(C) \approx$ a few hundred.

Appendix K: why the conjunctions are the cheapest survivors

At the start of stage k the data says: $h(1\cdots1)=1$, and $h(x)=0$ for every $|x|<k$. (Cost model: leaf $v=2+\lceil\log_2 d\rceil$, AND/OR/XOR gate 4, NOT 2, constant 3.)

1 (Möbius). In h's GF(2) polynomial the coefficient of monomial S is $c_S=\bigoplus_{T\subseteq S}h(1_T)$. Vanishing below weight k kills every coefficient of degree $<k$, and leaves $c_S=h(1_S)$ at degree k.

2 (leaves). So every monomial of h has degree $\geq k$, and $h\neq 0$ — so h depends on $\geq k$ distinct variables: any formula for it has $\geq k$ leaves and $\geq k-1$ binary gates, cost $\geq kv+4(k-1)$.

3 (equality). At exactly $kv+4(k-1)$ there is no room for constants, NOTs, or a repeated variable: h is a bare gate-tree on k distinct variables. And any assignment to them other than all-ones zero-fills to an input of weight $<k$, where h must vanish — so $h=\mathrm{AND}$ of its variables. (This is what killed OR_S and XOR_S: both are nonzero at weight-1 points inside S.)

Checked by exact dynamic programming at $d=4$, over all 2^{16} data-subsets: the optimal adversarial ordering is precisely this ladder.

Appendix K: the ladder at $d = 4$

step	input	label	learner's hypothesis (cost)	prediction
1	1111	1	constant 0 (3)	0 ✗
2	0000	0	constant 1 (3)	1 ✗
3–6	1000, 0100, 0010, 0001	0	x_1, x_2, x_3, x_4 (4)	1 ✗ each
7–12	the six weight-2 points	0	the six $x_i \wedge x_j$ (12)	1 ✗ each
13–16	the four weight-3 points	0	the four $x_i \wedge x_j \wedge x_k$ (20)	1 ✗ each

$2 + 4 + 6 + 4 = 16 = 2^4$ mistakes, and only then does the learner reach AND_4 (cost 28). Showing 1111 first is the key move: the one positive example never rules out any monotone conjunction, and each negative example kills exactly one survivor.

Here *simplest* = *fewest literals* = *most general*, and ascending-weight data punishes over-generality one point at a time. The most-*specific* consistent learner handles the same stream with $\leq d + 1$ mistakes.

Appendix K: it is the argmax that fails, not the prior

Same prior $\propto 2^{-C(g)}$, same data. Only the selection rule changes.

selection rule	worst case over orderings
argmax — Occam / MAP / MDL, deterministic ties	$2^{\Theta(C/\log C)}$
argmax with randomized tie-breaking within a cost level	$\tilde{\Theta}(C^2)$
uniformly random consistent hypothesis, doubling budget	$O(C)$
weighted majority vote — Bayesian model averaging	$\leq C$

The last line is one paragraph: total weight is ≤ 1 by Kraft, each mistake halves the surviving weight, and f's weight $2^{-C(f)}$ is never removed. It holds on *every* ordering.

Where this touches us. Our $\sigma \to 0$ learner is *not* the deterministic argmax: ties spread symmetrically (the Gaussian prior has no convention to conspire with) and we predict by majority — the good side of this table. On the ladder the surviving conjunctions get equal mass, so the majority errs $O(d)$ times. The caution is for the *abstraction*: modeling NNbayes as a single-simplest-hypothesis Occam learner can be off by $2^{\Theta(C/\log C)}$ out of distribution; the Bayesian version (Appendix A) survives.

Appendix K: does it transfer to κ_N? which orderings are bad

Transfer. The mechanism needs two things: an antichain of equal-complexity hypotheses, and each one wrong at a point all the others get right. The second is a fact about conjunctions, not about the language. The first holds for κ_N too — the Gaussian prior is invariant under permuting input coordinates, so all $\binom{d}{k}$ conjunctions of k variables have the same κ_N. But note the symmetry cuts both ways: it also makes the posterior *tie* across the antichain, which is exactly what rescues the majority vote (previous slide).

Which orderings. Random order is fine in any language, $\mathbb{E}[M] = O\big((C+d)\,d\big)$. But "structured $\Rightarrow$ safe" is false: the catastrophic ordering has description complexity $O(\log d)$, and smoothing it by w helps only by a factor w. Descending weight — anti-curriculum — gives exactly $d+1$. The real axis is *correlation between the data order and the arbitrary tie-breaking conventions of the language*; catastrophe needs a conspiracy between the two.

A discrete, realizable, noise-free model — no SGD, no overparametrization, not a theorem about a net. Nearest relative: Poland & Hutter (MDL loss $\Theta(2^{K_{\mathrm{PH}}})$ vs Bayes mixture $O(K_{\mathrm{PH}})$); NN-side, Wilson & Izmailov's "marginalization instead of optimization".

Appendix L: wherefore our conditions on ϕ

Bounded, $|\phi| \leq \text{const}$. Used in subsampling: for concentration bounds to be good, we need to bound the contributions of individual neurons.

Lipschitz, $|\phi'| \leq \text{const}$. Used twice. In the small-width count, it is what makes $w \mapsto z_w(x)$ Lipschitz on the norm ball, so that distinct functions are separated in weight space and the volume argument runs. In the multilayer case, it is what carries a perturbation through the nonlinearity, so that it is amplified only by the operator norms.

Non-polynomial. Used in Appendix B, to relate other circuit complexities to MLP-circuit complexity: it is exactly the hypothesis of the interpolation theorem.

A saturating sigmoid has all three. ReLU is Lipschitz and non-polynomial, but bounded only on a bounded range. Appendices G and H additionally use a *saturating* ϕ with $\phi'(0) \neq 0$ and $\phi(0) = 0$ — nondegeneracies rather than genuinely new conditions — and H's fixed point also needs ϕ *odd*, so that the $g = -1$ branch lands on $-\alpha$; that one is a genuine symmetry assumption. $\tanh$ has all of them.