# Estimating the Probability of Sampling a Trained Neural Network at Random

#### Adam Scherlis 1 Nora Belrose 1

# **Abstract**

We present an algorithm for estimating the probability mass, under a Gaussian or uniform prior, of a region in neural network parameter space corresponding to a particular behavior, such as achieving test loss below some threshold. When the prior is uniform, this problem is equivalent to measuring the volume of a region. We show empirically and theoretically that existing algorithms for estimating volumes in parameter space underestimate the true volume by millions of orders of magnitude. We find that this error can be dramatically reduced, but not entirely eliminated, with an importance sampling method using gradient information that is already provided by popular optimizers. The negative logarithm of this probability can be interpreted as a measure of a network's information content, in accordance with minimum description length (MDL) principles and rate-distortion theory. As expected, this quantity increases during language model training. We also find that badly-generalizing behavioral regions are smaller, and therefore less likely to be sampled at random, demonstrating an inductive bias towards well-generalizing functions.

### 1. Introduction

There is a long line of research which finds that *flat* minima in a neural network parameter space, defined as weight vectors surrounded by *large regions* "with the property that each weight vector from that region leads to similar small error" generalize better than *sharp* minima (Hochreiter & Schmidhuber, 1997). While there are counterexamples to this tendency (Dinh et al., 2017), it seems to be empirically and theoretically fairly robust, and has inspired the development of optimizers that explicitly search for flatter minima (Foret et al., 2021).

In a related line of work, Chiang et al. (2022) put forward the *volume hypothesis*, which states that "...the implicit bias

of neural networks may arise from the volume disparity of different basins in the loss landscape, with good hypothesis classes occupying larger volumes." They evaluate simple gradient-free learning algorithms, such as the "Guess & Check" optimizer which randomly samples parameters until it stumbles upon a network that achieves training loss under some threshold, and find that these methods have similar generalization behavior to gradient descent, at least on the very simple tasks they tested. Teney et al. (2024) find that randomly initialized networks represent very simple functions, which would explain the simplicity bias of deep learning if SGD behaves similarly to Guess & Check.

Additionally, Mingard et al. (2021) provide evidence that SGD may be an approximate Bayesian sampler, where the prior distribution over functions is equal to the distribution over functions represented by randomly initialized networks. Since networks are usually initialized using a uniform or Gaussian distribution, the Bayesian sampling hypothesis makes similar predictions to the volume hypothesis. Finally, recent work suggests that singular learning theory (Watanabe, 2009), originally developed to analyze the learning dynamics of overparameterized Bayesian models, can be profitably used to understand deep learning (Hoogland et al., 2024; Lau et al., 2024).

In this work, we propose an efficient algorithm for estimating the probability that a network from some behaviorallydefined region would be sampled from a Gaussian or uniform prior. This is equivalent to the volume of the region, if the prior is uniform. We define the local volume of a weight vector  $\theta \in \mathbb{R}^N$  relative to a cost function  $\mathcal{C}: \mathbb{R}^N \to [0, \infty)$ , threshold  $\epsilon > 0$ , and measure  $\mu$  to be the size according to  $\mu$  of the star domain S anchored at  $\theta$ containing points  $\theta'$  such that  $\mathcal{C}(\theta') < \epsilon$ , which we call the **neighborhood** of  $\theta$ . This is equivalent to the probability of sampling a network inside S from a prior proportional to the measure  $\mu$ . While prior work has assumed  $\mu$  to be the Lebesgue measure (i.e. volume), we also consider the probability measure used to initialize the network before training, which guarantees that the measure of any neighborhood must be finite. We find empirically that some realworld neighborhood actually have infinite Lebesgue volume, creating difficulties for analysis.

We use the term "cost" rather than "loss" intentionally be-

<sup>\*</sup>Equal contribution <sup>1</sup>EleutherAI. Correspondence to: Adam Scherlis <adam@eleuther.ai>.

cause we are interested in a broader class of functions than just the loss function used to train the model. In particular, for networks that output a predictive probability distribution we find it useful to consider the expected KL cost function  $\mathbb{E}_x[D_{KL}(f(x;\theta)||f(x;\theta')]]$ , which measures how behaviorally different  $\theta'$  is from  $\theta$ , independent of any ground truth labels. This cost also has the benefit that it is always zero at the minimum point  $\theta' = \theta$ .

We estimate that the probability of randomly sampling the trained Pythia 31M language model from its initialization distribution, within an accuracy of 0.01 nats, is about

$$\Pr(\text{language model}) \approx \frac{1}{10^{3.6 \times 10^8}}$$
 (1)

or one in 1 followed by 360 million zeros. For comparison, there are about  $10^{80}$  atoms in the observable universe, so this is about the same as the probability of correctly guessing a specific atom 4.5 million times in a row.

#### 2. Motivation

As mentioned in the introduction, one major motivation behind this work is to test the volume hypothesis: the idea that the relative volumes of different regions in parameter space strongly shape the kinds of networks that are produced by gradient descent. There are many variants of this hypothesis, and we detail two of them below.

The Bayesian volume hypothesis. Here is a simplistic version of the volume hypothesis which we think is likely false for real-world neural networks, but which may be helpful as an intuition pump.

Consider Bayesian inference with prior density  $\rho(\theta)$  and likelihood function -L. The posterior distribution is proportional to  $\rho(\theta) \exp(-L(\theta))$ . Since many neural-network losses can be interpreted as negative log-likelihoods, we can think of L as the loss function of a neural net and  $\rho$  as a prior related to initialization and regularization of the network. If neural-net training were perfectly Bayesian, the probability density for obtaining some parameter  $\theta$  from training would depend only on the prior and the loss.

This is of interest because it attributes generalization entirely to the architecture and loss function: under this hypothesis, the only way for one low-training-loss solution to be favored over another is if it simply occupies more of parameter space. In effect, the architecture imposes a sophisticated inductive prior (on top of the simple prior  $\rho$ ) by overrepresenting "good", well-generalizing functions, and underrepresenting "bad" ones.

This hypothesis is true for stochastic gradient Langevin dynamics (Welling & Teh, 2011), which is an efficient Bayes sampler for deep neural networks, but only with unrealisti-

cally long mixing times.

Quadratic toy model. Consider a quadratic loss function with Hessian H. If the initialization distribution  $\mu_0$  has covariance matrix I, then at timestep t the covariance is  $\exp(-\mathbf{H}t)\exp(-\mathbf{H}t)^T$ . Assuming  $\mu_0$  is a zero-mean Gaussian, the log density of parameters  $\theta$  at time t is proportional to  $\theta \exp(2\mathbf{H}t)\theta^T$ , which is in general not proportional to the loss  $\frac{1}{2}\theta\mathbf{H}\theta^T$ . The probability mass becomes concentrated along directions of higher curvature (larger Hessian eigenvalues) exponentially faster than along directions of lower curvature.

If we introduce isotropic noise and solve the resulting Fokker-Planck equation, it can be shown that the log-density instead converges to something proportional to the loss. However, if the noise is not isotropic – in particular if it is stronger in more steeply-curved directions, as is true in practice – then this fails (Mandt et al., 2018).

This shows that, *within* basins of non-isotropic curvature, the posterior density of popular optimizers does not satisfy the Bayesian volume hypothesis. We can, however, restrict the hypothesis to apply only *between* different basins.

The basin volume hypothesis. Let  $\lambda$  be the Lebesgue measure on  $\mathbb{R}^N$ , and let  $\mu_0$  be the probability measure on  $\mathbb{R}^N$  from which the initial network parameters  $\theta_0$  are sampled, usually a uniform distribution on a compact set or a Gaussian. Let  $\mu_t$  be the distribution over network parameters at timestep t in training, and let  $f_t(x) = \frac{d\mu_t}{d\mu_0}$  be the probability density of parameters x at time t. We can decompose the posterior probability of behaviorally distinct regions of parameter space, such as basins of low loss with differing degrees of generalization, as follows.

Let  $A\subset\mathbb{R}^N$  and  $B\subset\mathbb{R}^N$  be two disjoint regions of parameter space, perhaps defined by their performance on a held-out test set. The probability that training will yield an element of A can be decomposed as

$$\log \mathbb{P}(\theta \in A) = \log \left[ \mu_0(A) \cdot \frac{1}{\mu_0(A)} \int_A f_t d\mu_0 \right]$$
(2)  
= 
$$\underbrace{\log \mu_0(A)}_{\text{volume}} + \underbrace{\log \mathbb{E}_{x \sim \text{Unif}(A)} \left[ f_t(x) \right]}_{\text{mean density}}$$
(3)

and the log probability ratio is

$$\log \frac{\mathbb{P}(\theta \in A)}{\mathbb{P}(\theta \in B)} = \underbrace{\log \frac{\mu_0(A)}{\mu_0(B)}}_{\text{volume ratio}} + \underbrace{\log \frac{\mathbb{E}_{x \sim \mu_0 \mid A} \left[ f_t(x) \right]}{\mathbb{E}_{x \sim \mu_0 \mid B} \left[ f_t(x) \right]}}_{\text{density ratio}}, \quad (4)$$

where  $\mu_0|A$  denotes the restriction of  $\mu_0$  to A. Note that

<sup>&</sup>lt;sup>1</sup>Formally, the Radon-Nikodym derivative of  $\mu_t$  w.r.t.  $\mu_0$ . This quantity exists if  $\mu_t$  is absolutely continuous w.r.t.  $\mu_0$ .

<sup>&</sup>lt;sup>2</sup>For example, if  $\mu_0 = \text{Unif}(S)$  for some compact  $S \subset \mathbb{R}^N$ ,

at t=0 we have  $f_0(x)=\frac{d\mu_0}{d\mu_0}(x)=1$  for any x, so that at early times t the density ratio term in Eq. 4 should be small.

A restricted form of the volume hypothesis is as follows: Even at the end of training, the volume ratio term in Eq.4 should be larger than the density ratio term for suitable choices of A and B.

Of course, if the networks in A and the networks in B differ significantly in terms of their performance on the training set, the density ratio term must become very large as  $t \to \infty$ , since a well-tuned optimizer is guaranteed to bring the loss close to a local minimum. When analyzing generalization, then, we should select A and B such that are both contained in a low-loss manifold (Benton et al., 2021).

#### 2.1. Minimum description length

Basin volume can be connected directly to generalization using the notion of minimum description length (MDL). The idea is that a statistical model is more likely to generalize if it compresses its training data effectively, while not being too complex itself. Since we are assuming that all networks in the neighborhood perform similarly, we will treat the neighborhood itself as an ensemble over networks, and use it as our statistical model. In Bayesian terms, our posterior is a uniform distribution over the neighborhood, and we assume that our receiver is using the initialization distribution  $\mu_0$  as a prior. The bits-back argument (Hinton & Van Camp, 1993) shows that the MDL of this model plus the training data  $x_{1:n}$  is

$$\mathrm{KL}\big(\mathrm{Unif}(A)||\mu_0\big) + \mathbb{E}_{\theta \sim \mathrm{Unif}(A)}\Big[\sum_{i=1}^n \log_2 p_{\theta}(x_i)\Big],$$
 (5)

where  $A \subset \mathbb{R}^N$  is the neighborhood, and  $p_{\theta}(x_i)$  is the probability that the network with parameters  $\theta$  assigns to datapoint  $x_i$ .

In practice,  $\mu_0$  is either a uniform distribution over a simple polytope  $S \subset R^N$ , or a (possibly truncated) Gaussian  $\mathcal{N}(0,\Sigma)$  with diagonal covariance. In the former case, the KL term simplifies to  $\log \lambda(S) - \log \lambda(A)$ , and in the latter, it simplifies to

$$\frac{n}{2}\log(2\pi) + \frac{1}{2}\log|\Sigma| + \frac{1}{2}\mathbb{E}_{\theta \sim \mathrm{Unif}(A)}[\theta^T \Sigma^{-1}\theta] - \log\lambda(A),$$

which only depends on A is through its volume and its mean Mahalanobis distance from the origin. Neighborhoods with large Lebesgue volume and small average Mahalanobis norm will have lower description length than neighborhoods with smaller volume or higher Mahalanobis norm.

then  $\mu_0|A=\mathrm{Unif}(A\cap S)$ . If  $\mu_0$  is a Gaussian, then  $\mu_0|A$  is a truncated Gaussian with support A.

# 2.2. Singular Learning Theory and the Local Learning Coefficient

The local learning coefficient (LLC) was introduced by Lau et al. (2024), extending concepts from singular learning theory (Watanabe, 2009), and has proved to be useful as a measure of the complexity of neural networks and their components (Hoogland et al., 2024; Wang et al., 2024).

Consider a local minimum  $\theta^*$  in the loss landscape  $L(\theta)$ . Consider the volume V(c) of the "basin" of nearby parameters  $\theta$  with loss  $L(\theta) \leq L(\theta^*) + c$ . Under some fairly general smoothness assumptions,  $V(c) \to 0$  as  $c \to 0$ , with some asymptotic scaling of the form

$$V(c) \sim c^{\lambda}$$
 (6)

The LLC is defined as the exponent  $\lambda$ . Note that  $\lambda = \frac{N}{2}$  whenever the Hessian is full-rank. In the context of singular learning theory, this is derived from a Bayesian perspective on deep learning, somewhat along the lines of the Bayesian volume hypothesis described above, albeit with much more mathematical sophistication.

Our measure is derived from somewhat similar considerations, and takes a similar form, with some key differences:

- We are interested in the behavior of V(c) away from the  $c \to 0$  limit.
- We want to compare the value V(c) across different regions, such as better- or worse-generalizing networks, or multiple training checkpoints.
- We want to apply this framework to cost functions other than the loss, allowing us to study neural nets far to local minima, without a localizing term.

# 2.3. Predictions

The considerations above lead us to expect the following:

- Among trained networks with low training loss, better-generalizing networks (lower validation loss) should have larger KL neighborhoods (shorter description lengths) than worse-generalizing ones.
- During training, KL local volume should tend to decrease (description length should increase), with possible exceptions when networks consolidate their knowledge (as seen for LLC).

# 3. Method

Our method builds on the work of Huang et al. (2020), who define 'basin' as "the set of points in a neighborhood of the

minimizer that have loss value below a cutoff." This definition is ambiguous because it leaves the notion of "neighborhood" undefined. We will show below that their method in fact estimates the volume of the star domain anchored at the minimizer such that all networks in the domain have loss value (or more generally, cost) below a cutoff.

#### 3.1. Naïve approach

Recall that a star domain  $S \subseteq \mathbb{R}^N$  is a set containing an **anchor**  $s_0$  such that for all  $s \in S$ , the line segment from  $s_0$  to s lies in S. This property allows us to define S in terms of a radial function  $r: \mathbb{S}^{N-1} \to [0,\infty)$  which takes in a unit vector u and outputs a non-negative number corresponding to the "radius" of S along u, or the length of the line segment from  $s_0$  to the boundary of S along the direction u. Given this parameterization, the volume of S can be written as

$$\operatorname{vol}(S) = \int_{\mathbb{S}^{N-1}} \int_{0}^{r(\boldsymbol{u})} r^{n-1} dr d\Omega \tag{7}$$

$$=\frac{1}{n}\int_{\mathbb{S}^{N-1}}r(\boldsymbol{u})^nd\Omega\tag{8}$$

$$= \frac{|\mathbb{S}^{N-1}|}{n} \mathbb{E}_{\boldsymbol{u} \sim \mathrm{Unif}(\mathbb{S}^{N-1})}[r(\boldsymbol{u})^n], \qquad (9)$$

where  $|\mathbb{S}^{n-1}| = \frac{2\pi^{n/2}}{\Gamma(n/2)}$  is the surface area of a unit N-ball. We can estimate this using k Monte Carlo samples:<sup>3</sup>

$$\operatorname{vol}(S) \approx \widehat{\operatorname{vol}(S)} = \frac{|\mathbb{S}^{n-1}|}{nk} \sum_{i=1}^{k} r(\boldsymbol{u}_i)^n$$
 (10)

Equation 10 is an unbiased estimator for the volume. It is also, very reliably, millions of orders of magnitude too small in practice. Below, we explain the source of this phenomenon and a method for ameliorating it.

**Bias of the log-estimator.** In practice, we estimate  $\log \operatorname{vol}(S)$ , rather than  $\operatorname{vol}(S)$  itself, to prevent numerical overflow or underflow. Jensen's inequality tells us that the logarithm of an unbiased estimator is a *downwardly biased* estimator for the logarithm of the population parameter:

$$\log \operatorname{vol}(S) \ge \mathbb{E}[\log \widehat{\operatorname{vol}(S)}],\tag{11}$$

with equality if and only if the estimator is constant. This "Jensen gap" is especially large when the variance of the log-estimator is large. For example, if the log-estimator is normally distributed with standard deviation  $\sigma$ , the gap is  $\sigma^2/2$ .

**Smooth maximum.** In practice, n will be extremely large, ranging from  $10^6$  to  $10^{12}$  parameters. It is therefore worth considering the limit of our estimator as n tends to infinity. First note that

$$\mathbb{E}[\log \widehat{\operatorname{vol}(S)}] \propto \mathbb{E}[\log \sum_{i=1}^{k} \exp(n \log r(\boldsymbol{u}_i))].$$
 (12)

LogSumExp is sometimes used as a continuous relaxation of the max function, because for any fixed set of values  $\{x_1, \ldots, x_k\}$  we have:

$$\lim_{n \to \infty} \frac{1}{n} \log \sum_{i=1}^{k} \exp\left(nx_i\right) = \max(\{x_1, \dots, x_k\}). \quad (13)$$

This suggests that, in the large-n limit, the normalized log volume estimate  $\frac{1}{n}\mathbb{E}[\log \operatorname{vol}(S)]$  will be proportional to the *maximum* of our log-radius samples. Empirically, we find that this is already very nearly true for tiny networks of a few thousand parameters (Figure 1).

**Markov's inequality.** Since our estimator is a non-negative random variable, we can use Markov's inequality to show that with high probability, our estimate of the log-volume will not significantly *overestimate* the true value:

$$\mathbb{P}\Big(\log\widehat{\text{vol}(S)} - \log\text{vol}(S) \ge \log k\Big) \le \frac{1}{k} \tag{14}$$

That is, the probability that we overestimate the true volume by m>0 orders of magnitude is at most one in  $10^m$ . Empirically, since our Monte Carlo samples vary over thousands or millions of orders of magnitude, we can view our estimate as a high-confidence approximate lower bound on the true log-volume, with error  $\ll \text{Var}(\log \widehat{\text{vol}(S)})$ .

# 3.2. Preconditioning

We propose to reduce the variance of the volume estimator with importance sampling. We still begin by sampling isotropic unit vectors u. However, we then multiply these by a positive-definite preconditioner  $\mathbf{P}$  with unit determinant, to obtain vectors  $v = \mathbf{P}u$ . We then unit-normalize these to obtain unit vectors  $\hat{v}$ , and use the estimator

$$\widehat{\operatorname{vol}(S)} = \frac{|\mathbb{S}^{N-1}|}{nk} \sum_{i=1}^{k} \frac{r(\hat{\boldsymbol{v}}_i)^n}{|\boldsymbol{v}|^n}$$
(15)

where the denominator is the usual importance-sampling correction. Under the stated conditions on  ${\bf P}$ , this is still unbiased.

The purpose of  $\mathbf{P}$  is to more aggressively sample directions that are flatter. We can interpret the formula above as our original estimator under a change of coordinates by  $\mathbf{P}$ , with

<sup>&</sup>lt;sup>3</sup>For each sample, the radial function is computed via binary search in a uniformly-random direction.

the unit-determinant condition ensuring that the volume of the neighborhood is unchanged in the new coordinates. For a good choice of **P**, the neighborhood will be more spherical in the new coordinates. With this in mind, we refer to the matrix **P** as a preconditioner.

In the case where the neighborhood is perfectly ellipsoid, a perfect choice of **P** would have eigenvectors aligned with principal axes and eigenvalues proportional to the lengths of those axes. This would result in an estimator with zero variance, returning the exact volume every time. Note that for a quadratic cost function, this is proportional to the inverse square root of the Hessian,

$$\mathbf{P} \propto \mathbf{H}^{-\frac{1}{2}} = \mathbf{V} \mathbf{D}^{-\frac{1}{2}} \mathbf{V}^T \tag{16}$$

where V, D are the eigenvectors and eigenvalues of H.

For very small neural nets, we use a form of this Hessian preconditioner that is modified to ensure positive-definiteness:

$$\mathbf{P} \propto \mathbf{V} \frac{1}{|\mathbf{D}|^{\frac{1}{2}} + \epsilon} \mathbf{V}^{T}$$
 (17)

We can further economize by using the Hessian diagonal:

$$\mathbf{P} \propto \frac{1}{|\operatorname{diag}(\mathbf{H})|^{\frac{1}{2}} + \epsilon} \tag{18}$$

where diag(**H**) is a matrix equal to **H** along its diagonal and zero elsewhere. While exactly computing the Hessian diagonal is no more computationally efficient than computing the entire Hessian, in practice we use the HesScale approximation (Elsayed & Mahmood, 2022), which is deterministic, highly efficient, and empirically very accurate.

Finally, for arbitrarily large networks we can use Adam's second moment buffers to estimate  $\operatorname{diag}(\mathbf{H})$ . In general, we can use any vector or matrix in place of  $\mathbf{H}$  and its diagonal, and can optionally replace  $\frac{1}{2}$  with another exponent to obtain a better preconditioner.

Because of the Markov-inequality bound above, we can test preconditioners very easily: larger numbers are always more accurate, so long as the preconditioner is unit-determinant. This also gives us, retroactively, a lower bound on how badly the naive (un-preconditioned) estimator undershoots.

#### 3.3. Gaussian volume

Behaviorally defined neighborhoods can often have infinite Lebesgue volume, making them hard to analyze. If there is any direction along which perturbations have precisely zero effect on the model's behavior on the validation set, that direction will have an infinite radius. There are often many of these. As an example, we find that several pixel locations are never used in the digits validation set, so the corresponding input weight parameters in any network will have no effect.

If we view neural network training as Bayesian inference, it is natural to think of the distribution used to initialize the parameters as a prior, and in practice this is often a Gaussian distribution. We therefore replace the Lebesgue measure with the Gaussian initialization measure with PDF  $\rho$ . Our preconditioned volume estimator becomes

$$\widehat{\text{vol}(S)} = \frac{|\mathbb{S}^{N-1}|}{k} \sum_{i=1}^{k} \frac{\int_{0}^{r(\widehat{\mathbf{v}_i})} \rho(s_0 + r\mathbf{u}_i) r^{n-1} dr}{|\mathbf{V}|^n}$$
(19)

Note that the integrand is of the form  $\exp(\operatorname{quadratic}(r) + n\log r)$  and varies rapidly when n is large. We evaluate these integrals numerically using an approximation similar to Lagrange's method, expanding the exponent to second-order and performing a Gaussian integral using a numerically-stable implementation of the error function. In practice, the error from the approximation is less than floating-point rounding error.

#### 3.4. From loss to KL divergence

The forms of the volume hypothesis above deal with the training loss. In practice, however, the Hessian of the loss function is closely related to the Fisher information matrix for the model (Martens & Grosse, 2015), which is the Hessian of KL divergence. We therefore use KL from the anchor point as our cost function, as well as our defining criterion for our neighborhoods. This has the added benefit of putting the anchor point at a global minimum by definition, simplifying some of the practical challenges. It can also be interpreted within an MDL framework as the description length of the network, as shown below.

#### 3.5. Poisoned networks

We produce "poisoned" ConvNeXt networks on CIFAR-10 using the methodology of (Huang et al., 2020), where the standard training loss is augmented with a term encouraging the model to perform poorly on a held-out "poison" set. These networks generalize worse than the unpoisoned ones, while still achieving low train loss. Our hypothesis is that poisoned networks should have smaller local volumes than unpoisoned ones.

# 4. Results

We test our method in three settings: a small MLP (4810 parameters) trained on the UCI handwritten digits dataset

<sup>&</sup>lt;sup>4</sup>The denominator, in this interpretation, can be seen as resulting from differing notions of "unit length" in the original and new coordinates.

 $<sup>{}^5\</sup>mathbf{P}$  is "proportional to" this quantity because it must be normalized to determinant 1.

(Alpaydin & Kaynak, 1998), a variant of the ConvNeXt Atto model (Woo et al., 2023) (3.4M parameters) trained on CIFAR-10 (Krizhevsky & Hinton, 2009), and checkpoints from the Pythia 31M language model (Biderman et al., 2023).

We compute KL divergence on held-out sets consisting of 773 images from digits, 1024 images from CIFAR-10, and 20 text sequences (10926 tokens) from the Pile, respectively. Except where otherwise specified, all results are for k=100 samples per data point and with a KL cutoff of  $10^{-2}$  nats. In the plots that follow, note that the base-ten logarithms of the probability estimates are themselves on the order of  $-10^6$  or  $-10^8$ , as shown by the " $\times 10^6$ " and " $\times 10^8$ " annotations on the x-axis labels.

#### 4.1. Preconditioners

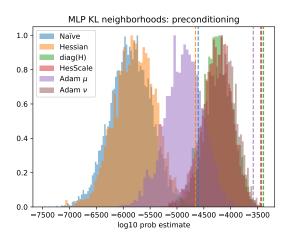


Figure 1. Results (k=3000) for various preconditioners on a small MLP. Vertical dashed lines indicate the aggregated log-volume estimate, which is very close to the maximum sample.

We estimate the local volume for our small MLP, using the various preconditioners described above, as shown in 1. We show a histogram of the individual samples, with vertical dashed lines for the aggregated estimate. Note that, on a log scale, the estimate is extremely close to the largest individual sample.

Interestingly, results when preconditioning with the Hessian of the KL (the Fisher matrix) are very similar to the unpreconditioned ones. The  $\operatorname{diag}(\mathbf{H})$ , HesScale, and Adam second-moment ( $\nu$ ) preconditioners perform much better, and very similarly to each other. The Adam first-moment ( $\mu$ ) preconditioner is somewhere in between.

The hyperparameter  $\epsilon$  is tuned separately for each of these,

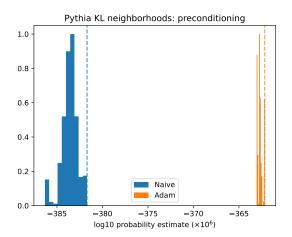


Figure 2. Results (k=1000) with and without Adam preconditioner on Pythia 31M

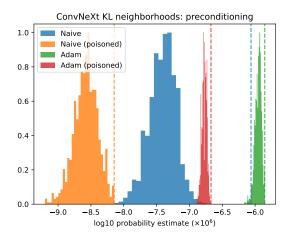


Figure 3. Results (k=1000) with and without Adam preconditioner on ConvNeXt Atto poisoned and unpoisoned

to obtain the largest (hence most accurate) result. We find that  $\epsilon=0.1$  works best for the Hessian, while  $\epsilon=0.01$  is best for diag(H) and HesScale and  $\epsilon=0.001$  is best for both Adam preconditioners.

We find it surprising that the full Hessian performs so poorly, especially given the success of  $\operatorname{diag}(\mathbf{H})$  and its approximations. This may be some form of overfitting, if the locally-flattest directions are slightly misaligned with the longest directions of the neighborhood, but if so, it is unclear why constraining to axis-aligned directions helps so much.

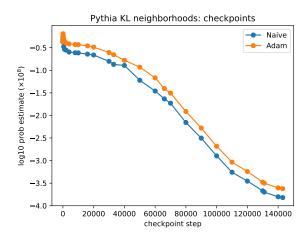
We also use the second-moment Adam preconditioner for Pythia and ConvNeXt, where it shows both a clear improvement in the value of the estimates and a smaller sample variance (Figures 2 and 3). The improvement is several

 $<sup>^6</sup>$ We changed the default patch size, which was optimized for ImageNet, from  $4 \times 4$  to  $1 \times 1$ . This significantly improves accuracy on smaller images like those in CIFAR-10.

standard deviations above most of the naïve estimates, suggesting that it would be infeasible to merely increase the sample size to try to get the same result.<sup>7</sup>

For ConvNeXt, we find that the poisoned network has a smaller local volume (with or without preconditioning), in agreement with the results of Huang et al. (2020) on small networks and in line with our expectation from the MDL and compression perspective.

#### 4.2. Across training checkpoints



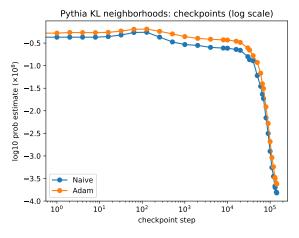
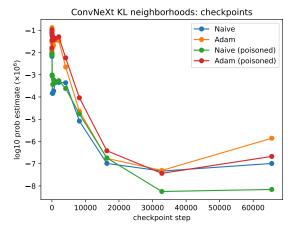
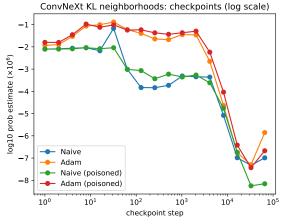


Figure 4. Local volume decrease while training Pythia 31M

As expected, local volume tends to decrease during training, as the network learns more and its description length increases. For Pythia, this decrease is smooth and approximately exponential after an rapid drop early in training (Figure 4). In this case, the Adam preconditioner yields modestly larger local volume estimates than the unprecon-





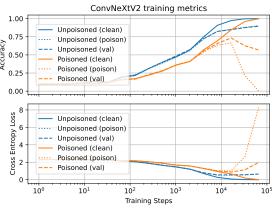


Figure 5. Local volume decrease while training ConvNeXt V2 Atto, and training metrics across datasets

ditioned method throughout training, but the two estimates follow parallel trends.

For ConvNeXt, the poisoned network actually has larger local volume for much of training, and then drops below the unpoisoned network around 30,000 steps, which is also when the val-set and poison-set losses diverge strongly

<sup>&</sup>lt;sup>7</sup>Note, however, that the naïve estimate for the unpoisoned ConvNeXt network has a large outlier sample that completely dominates the aggregated estimate, nearly reaching the bulk of the preconditioned estimates.

from each other. This makes sense: early in training, the poisoned loss is just holding back the network (worse loss across all three datasets), slowing the decrease in local volume. Later in training, the network overfits, decreasing its local volume to below the unpoisoned network's. This corresponds to a larger description length for the poisoned (overfit, poorly-generalizing) network.

#### 4.3. Across cutoffs

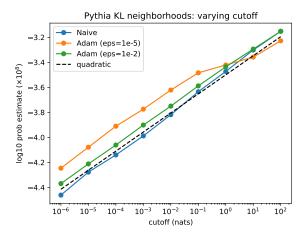


Figure 6. Results for various cutoffs on Pythia 31M

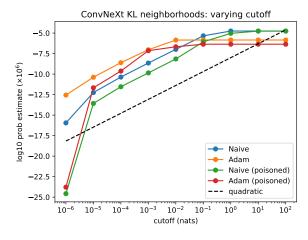


Figure 7. Results for various cutoffs on ConvNeXt V2 Atto

When varying the cutoff, we see a roughly power-law trend for local volume on Pythia (6). The log-log slope is consistent with n/2, where n is the model dimension; this is what would be expected for a purely quadratic cost function. A line with this slope is shown in black for comparison. At very high cutoffs ( $\geq 10$  nats), the Adam preconditioner begins to fail, although raising  $\epsilon$  partially counteracts this.

For ConvNeXt, the result is similar for cutoffs between

 $10^{-5}$  and  $10^{-2}$ . The preconditioner again fails at high cutoff, sooner than for Pythia. At very low cutoffs ( $10^{-6}$  nats), the poisoned network's local volume suddenly plummets. We have not investigated this phenomenon in detail, but we suspect it may have to do with floating point rounding error.

The slope of n/2 is somewhat in conflict with empirical results for the Local Learning Coefficient (Hoogland et al., 2024; Wang et al., 2024), which we suspect may be an artifact of the Monte Carlo local volume estimator. In particular, at low k, our estimator is nearly insensitive to very flat (long) directions, unless they are adequately corrected by the preconditioner. If the steeper (shorter) directions are close to quadratic in shape, this could cause the cost function to appear more quadratic than it really is.

#### 5. Conclusion

In this work, we introduced an efficient algorithm for estimating the probability that a network from some behaviorally-defined region would be sampled from a Gaussian or uniform prior, or equivalently, the network's *local volume*. While the method is demonstrably more accurate than prior state of the art, it is still unclear how close our estimates are to the ground truth. Nevertheless, we find that our estimated local volume decreases with training time, and is smaller for networks that overfit than for generalizing networks, suggesting that it at least correlates with the true local volume.

Our results are broadly consistent with a weak form of the volume hypothesis. As expected, poisoned networks were observed to have smaller local volumes than unpoisoned ones. That said, more research is needed to confirm or refute any specific version of the volume hypothesis.

One promising direction for future work may be to use stochastic gradient Langevin dynamics (SGLD) to propose directions along which to measure the neighborhood's radius. We also are excited to see practical applications of local volume estimation. We think it may be useful for predicting generalization performance. More speculatively, if we define the cost function to be the model's behavior on a relatively narrow distribution—say, a set of math problems fed to a large language model—the local volume may tell us something about how "difficult" these problems are for the model, or how hard it is "thinking."

Another possible direction may be to estimate the probability measure of neighborhoods around initializations that lead to a given final behavior after training, which corresponds almost exactly to the probability of SGD producing that trained behavior. This would allow for a precise quantitative evaluation of the basin volume hypothesis, and could potentially be accomplished via the training Jacobian (Belrose & Scherlis, 2024).

# **Contributions and Acknowledgements**

Adam Scherlis came up with the main ideas of this paper, wrote the code and performed all experiments and data analysis, and did much of the writing. Nora Belrose proposed the idea of using expected KL divergence as a cost function in lieu of training loss, pointed out the connection with minimum description length, proposed a last-minute reframing of the paper in terms of "sampling trained networks at random," and wrote various parts of the paper.

Adam and Nora are funded by a grant from Open Philanthropy. We thank Coreweave for computing resources.

# **Impact Statement**

This paper presents work whose goal is to advance the science of deep learning. At this early stage of research the social impacts are uncertain and indirect, although we hope that future research building on this work may be used to enhance generalization and reliability of neural networks.

#### References

- Alpaydin, E. and Kaynak, C. Optical Recognition of Handwritten Digits. UCI Machine Learning Repository, 1998. DOI: https://doi.org/10.24432/C50P49.
- Belrose, N. and Scherlis, A. Understanding gradient descent through the training jacobian, 2024. URL https://arxiv.org/abs/2412.07003.
- Benton, G., Maddox, W., Lotfi, S., and Wilson, A. G. G. Loss surface simplexes for mode connecting volumes and fast ensembling. In *International Conference on Machine Learning*, pp. 769–779. PMLR, 2021.
- Biderman, S., Schoelkopf, H., Anthony, Q. G., Bradley, H., O'Brien, K., Hallahan, E., Khan, M. A., Purohit, S., Prashanth, U. S., Raff, E., et al. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pp. 2397–2430. PMLR, 2023.
- Chiang, P.-y., Ni, R., Miller, D. Y., Bansal, A., Geiping, J., Goldblum, M., and Goldstein, T. Loss landscapes are all you need: Neural network generalization can be explained without the implicit bias of gradient descent. In *The Eleventh International Conference on Learning Representations*, 2022.
- Dinh, L., Pascanu, R., Bengio, S., and Bengio, Y. Sharp minima can generalize for deep nets. In *International Conference on Machine Learning*, pp. 1019–1028. PMLR, 2017.

- Elsayed, M. and Mahmood, A. R. Hesscale: Scalable computation of hessian diagonals. *arXiv* preprint *arXiv*:2210.11639, 2022.
- Foret, P., Kleiner, A., Mobahi, H., and Neyshabur, B. Sharpness-aware minimization for efficiently improving generalization. In *International Conference on Learning Representations (ICLR)*, 2021.
- Hinton, G. E. and Van Camp, D. Keeping the neural networks simple by minimizing the description length of the weights. In *Proceedings of the sixth annual conference on Computational learning theory*, pp. 5–13, 1993.
- Hochreiter, S. and Schmidhuber, J. Flat minima. *Neural computation*, 9(1):1–42, 1997.
- Hoogland, J., Wang, G., Farrugia-Roberts, M., Carroll, L., Wei, S., and Murfet, D. The developmental landscape of in-context learning. arXiv preprint arXiv:2402.02364, 2024.
- Huang, W. R., Emam, Z. A. S., Goldblum, M., Fowl, L. H., Terry, J. K., Huang, F., and Goldstein, T. Understanding generalization through visualizations. In "I Can't Believe It's Not Better!" NeurIPS 2020 workshop, 2020. URL https://openreview.net/forum? id=pxqYT\_7gToV.
- Krizhevsky, A. and Hinton, G. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.
- Lau, E., Furman, Z., Wang, G., Murfet, D., and Wei, S. The local learning coefficient: A singularity-aware complexity measure, 2024. URL https://arxiv.org/abs/2308.12108.
- Mandt, S., Hoffman, M. D., and Blei, D. M. Stochastic gradient descent as approximate bayesian inference, 2018. URL https://arxiv.org/abs/1704.04289.
- Martens, J. and Grosse, R. Optimizing neural networks with kronecker-factored approximate curvature. In *International conference on machine learning*, pp. 2408–2417. PMLR, 2015.
- Mingard, C., Valle-Pérez, G., Skalse, J., and Louis, A. A. Is sgd a bayesian sampler? well, almost. *Journal of Machine Learning Research*, 22(79):1–64, 2021.
- Teney, D., Nicolicioiu, A. M., Hartmann, V., and Abbasnejad, E. Neural redshift: Random networks are not random functions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4786–4796, 2024.

- Wang, G., Hoogland, J., van Wingerden, S., Furman, Z., and Murfet, D. Differentiation and specialization of attention heads via the refined local learning coefficient, 2024. URL https://arxiv.org/abs/2410.02984.
- Watanabe, S. *Algebraic geometry and statistical learning theory*, volume 25. Cambridge university press, 2009.
- Welling, M. and Teh, Y. W. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pp. 681–688. Citeseer, 2011.
- Woo, S., Debnath, S., Hu, R., Chen, X., Liu, Z., Kweon, I. S., and Xie, S. Convnext v2: Co-designing and scaling convnets with masked autoencoders. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16133–16142, 2023.

# A. Variance of the log-estimator

The variance of our local volume estimator is large when S contains outlier directions which have a large effect on the volume. Intuitively, it is difficult to estimate the volume of a needle or pancake by measuring its size along uniformly sampled directions. Most samples will be far closer to the minimum than to the maximum radius. We can formalize this intuition in the following way.

Consider an ellipsoid  $S = \{x \in \mathbb{R}^N : x^T A x \leq 1\}$  for some p.s.d. matrix A. Assume also that our anchor  $s_0$  is equal to the centroid of S. Now the radial function has the closed form:

$$r(\boldsymbol{u})^{-2} = \boldsymbol{u}^T \mathbf{A} \boldsymbol{u} \tag{20}$$

It turns out that the variance of this quadratic form, assuming u is uniformly distributed on the unit sphere, is

$$Var(\boldsymbol{u}^{T}\mathbf{A}\boldsymbol{u}) = \frac{2}{n+2}Var(\lambda), \tag{21}$$

where  $Var(\lambda)$  is the variance of the eigenvalues of **A**. Using a Taylor expansion around the mean, the variance of the log radial function is roughly half the squared coefficient of variation of the spectrum. As a result, the variance of the log-estimator is approximately:

$$\operatorname{Var}\left[-\frac{n}{2}\log(\boldsymbol{u}^{T}\mathbf{A}\boldsymbol{u})\right] \approx \frac{n^{3}}{2(n+2)} \cdot \frac{\operatorname{Var}(\lambda)}{\mathbb{E}[\lambda]^{2}}.$$
(22)

If the spectrum of A has any significant variance, as is empirically often the case for neural network Hessians, this variance will be extremely large.