The Scaling Laws of Classification Networks: Insights from Adaptive Exact Average Density Approximation

Anonymous authors
Paper under double-blind review

Abstract

Our main goal is to establish a generalization bound for classification tasks that aligns with the empirical scaling laws observed in deep neural networks (DNNs). Prior studies on scaling laws have not placed enough emphasis on network depth. This makes our approach particularly innovative. Suppose the boundary of the target classification function is a semi-algebraic set, we show that the generalization error can follow scaling laws for large networks. This study explores the relationship between scaling laws and sample size on data manifolds. The rate of scaling with respect to sample size is intrinsically linked to the effective dimension of the data manifold, independent of the specific network model or learning algorithm applied. In contrast, the rate per parameter exhibits variability across learning methods and network architectures. This variability can be quantified by the number of model parameters required to accurately characterize the target classification boundary at various radii of covering balls. Using this scaling law, we empirically demonstrate the feasibility of predicting the generalization errors of larger models from metrics of smaller models.

1 Introduction

Scaling laws in deep neural networks (DNNs) indicate that, across a wide range of architectures and tasks, the generalization error is primarily driven by scale. The generalization typically declines according to predictable power-law relationships associated with variables such as the number of parameters, the amount of training data, and the computational resources used for training. This observation suggests that larger models, trained on larger datasets, generally perform better in a smooth, quantifiable manner when properly optimized and regularized during training. Such regularities have been instrumental in guiding the design of large-scale models, such as GPT and PaLM, which use scaling curves to forecast performance before full-scale training.

Our approach builds on previous efforts in Bisla et al. (2021) and Sharma and Kaplan (2022) to explain scaling laws while introducing innovative elements. We highlight how increasing the depth of DNNs enhances their ability to capture finer local structures, thereby improving predictions. Our analysis relies on two conditions. First, the sample points must densely cover the input domain such that each decision region of a classification network contains at least one training point, and the histogram must accurately reflect the regional average density. Second, a polynomial function of network size should be inversely proportional to the radius of covering balls. By combining these elements, we first express the generalization error in terms of ball radius. We then demonstrate a scaling law that links a reduced ball radius to greater network depths and larger sample sizes.

Our contributions are twofold. First, consistent with the previous findings of Cortes et al. (1993), Bahri et al. (2024), and Bisla et al. (2021), we demonstrate that the relationship between generalization error and sample size adheres to a scaling law. This law features a slope that is inversely proportional to the effective or intrinsic dimension of the data manifold.

Second, we highlight another scaling law that relates generalization error to network size, particularly as network depth increases. We introduce the notion of box-module dimension to characterize the rate of

the scaling law. The The box-module dimension is a measure of the complexity of a set by quantifying the number of parameters in a model required to describe the shape at different scales of box size. The network architecture and learning methods can significantly influence the dimension. Furthermore, we show that this scaling law enables us to estimate the generalization errors of larger models using smaller models, thereby determining the resources required to increase model depth and adjust parameter settings to achieve a desired level of accuracy.

This document is organized as follows: Section 2 provides an overview of prior research in the field. In Section 3, we establish a bound on the generalization error using specific assumptions about the training points within the geometry induced by a classification network, based on the radius of covering balls. In Section 4, we present scaling laws linking the generalization bound to the number of training points by regularizing the density function. Following this, we introduce the box-module dimension to connect the radii of covering balls to the number of network parameters, thereby establishing scaling laws for network parameters. Finally, we show that the generalization error of a large model can be predicted from smaller models using published experimental data. Section 5 offers a mathematical justification for our assumptions, while Section 6 presents conclusions and suggestions for future research.

2 Related works

The paper by Hestness et al. (2017) serves as a precursor to the scaling law for accuracy. It demonstrates that large-scale empirical evidence across vision, speech, and language tasks that test loss follows a power-law scaling with dataset size N: $aN^{-\alpha}+b$. Rosenfeld et al. (2019; 2021) investigate how the generalization error of neural networks depends on both model size (specifically width and depth) and dataset size. They propose a parametric functional form that integrates model size and data size to predict the generalization error for a range of tasks in vision and language. Hutter (2021) derives power-law learning curves with a generalization error of $\mathcal{O}(1/N^{\beta})$, where N represents the number of training points. The rate β is related to the structural properties of the hypothesis space, the regularity of the data, and the complexity of hierarchical models. The Temporal Scaling Law examines test loss evolution in LLMs as training steps increase, using a dynamic hyperbolic model for predictions Xiong et al. (2024). Recent studies improve scaling law reliability through intermediate checkpoints and hyperparameter predictions Choshen et al. (2024) Li et al. (2025).

Some analysis aims to explain the scaling laws. Sharma and Kaplan (2022) introduce a model that explains the emergence of scaling laws based on the number of training points, denoted as D, and the number of network parameters, represented as P. Bahri et al. (2024) initializes the study of the generalization errors with respect to D and P in the resolution-limited and variance-limited regimes. In the resolution-limited regime, either D or P can approach infinity, allowing the other parameter to scale to infinity. In contrast, the variance-limited regime involves fixing a finite value on one of the parameters while the other can change. This work is compatible with the result by Park et al. (2025), which points out that benign overfitting can occur in the classical regime for large D. The study by Bisla et al. (2021) connects test points to the convex hull of their nearest training points within a hypercube, allowing for the approximation of density as uniform, and derives a closed-form expression for generalization error, which can be evaluated numerically and aligns with theoretical results.

From studying tractable shallow networks, it is possible to derive the factors governing the underlying dynamic behaviour at different scales. The foundational work on benign overfitting by Bartlett et al. (2020) closely links to scaling laws and highlights that, for effective generalization, the number of "unimportant" parameter directions must exceed the sample size by a significant margin. Cui et al. (2021) offers a bound on the generalization of kernel ridge regression, showing that eigenvalue decay influences learning rates. Maloney et al. (2022) developed a mathematically tractable joint model with one hidden layer and solved this model in a double limit—considering both large datasets and numerous parameters—they derived analytical formulas that connect the spectral properties of the data to the power-law scaling of test loss with dataset size or parameter count, explaining why scaling laws can eventually plateau.

3 Geometry, sample density, and generalization error

3.1 Assumptions and simplifications

We outline simplifications necessary to render our analysis tractable and to focus on the most critical parameters. This analysis focuses on the unit d-ball centered at $\underline{0}$, denoted as $B_d(\underline{0}, 1)$, within an input domain of dimension d. This choice does not limit our analysis, as it can be easily adapted to include any compact input domain typically encountered in real-world applications. Additionally, we will not consider the boundary in our analysis because of the compact support, which simplifies and yields more concise results.

Moreover, in this context, we do not differentiate between the dimension of the input domain and the effective dimension, also known as the intrinsic dimension. Given a density function, the effective dimension is the minimum number of independent variables or parameters needed to represent it without (substantial) loss of information Levina and Bickel (2004); Ansuini et al. (2019). This effective dimension is often smaller than the ambient dimension of the input domain. In this context, we present our findings in the input domain. Note that our results derived with d, the dimension of the input domain, can be replaced with d_e, the effective dimension of the density function $\mathcal{P}$. Additionally, we sometimes use d_e instead of d to align with concepts discussed in previous research.

The generalization error measures the difference between the expected test error and the expected training error, while the generalization bound represents an upper bound of that error. Our analysis makes two assumptions about training data distribution.

Assumption of densely covering: We assume that the data is sufficiently densely distributed, allowing spheres centered at a selected subset to cover the input domain within a defined radius.

To access the density of training data in the input domain, we use the γ-covering of the input domain, which means that for any $\mathbf{x}$ in the input domain, there is a training point $\mathbf{x}_i$ such as $\|\mathbf{x} - \mathbf{x}_i\|_2 \leq \gamma$. The γ-covering distribution in training points emphasizes that the input domain can be covered using balls with radius γ, and each ball contains at least one training point.

Assumption of histogram approximation: We assume sufficient training data are available for each covering ball to allow an accurate approximation of the average probability mass within the ball using a histogram over bins defined by the balls. Note that this assumption assumes that the average density functions within covering balls, not the actual density function.

These two assumptions form the basis of our analysis on generalization bounds of regression and classification functions, which help us identify the geometric parameters contributing to predicting errors in unseen data. In Section 5, we will provide an analysis that sufficiently supports these assumptions.

3.2 Generalization bound for covering with fixed radius balls

To understand the local geometry of the input of a neural network's function, we use covering balls in the input domain, focusing on two parameters: γ_s, the radius of the smallest enclosing ball for a set of regions, and β_l, the radius of the largest inscribed ball within that set. These regions relate to the arrangement of training points in the decision areas in classification networks. In Figure 1(a), γ_s is the radius of the ball encompassing both regions, while β_l represents the largest inscribed ball.

Our study utilizes specified balls to analyze local geometry, linking this analysis to generalization error and sample complexity. We demonstrate that the parameter γ_s effectively characterizes generalization errors. However, a higher sample density is essential to ensure that each decision region contains at least one training point. This necessary density is determined by the maximum radius of the balls, β_l, within the decision regions.

The lemma below shows that the sample density defined by β_l ensures that each region formed by the classification network has at least one training point.

Lemma 1. *Let β_l be the radius of the largest inscribed balls in the finest decision regions of the one-hot classification network $\mathcal{M}$. If samples cover the input domain with a β_l-covering, then each region of $\mathcal{M}$ contains at least one training point.* ■

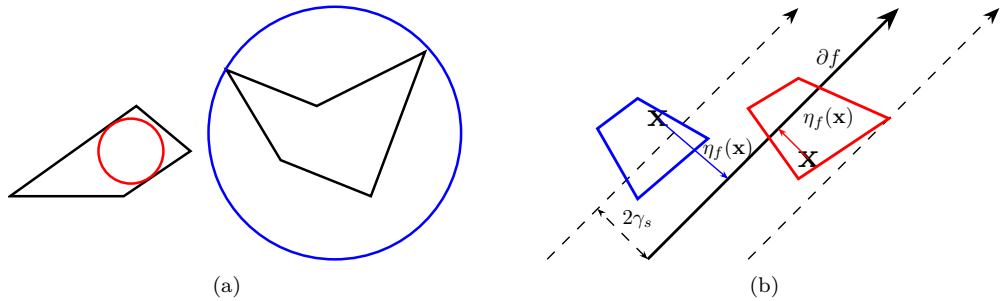

Figure 1: (a) Each polygon can be enclosed by and inscribed in a ball. The left (red) ball of radius β_l can be inscribed within both polygons, while the right (blue) ball of radius γ_s can enclose them. (b) Polygons represent decision regions of a classification network, each assigned a specific class. Any point $\mathbf{x}$ within the $2\gamma_s$-tube of the decision boundary ∂f (i.e., $\eta_f(\mathbf{x}) \leq 2\gamma_s$) can potentially cause a classification error, establishing an upper bound on generalization error. The red polygon intersects ∂f, with its two subregions representing different classes with respect to f, but they belong to the same class in the network. Conversely, any point farther than $2\gamma_s$ from ∂f cannot belong to a polygon that intersects the boundary, like the blue polygon.

Proof. Given that β_l is the radius of the largest inscribed ball within all decision regions in the input domain derived by network $\mathcal{M}$, it follows that for any region, there exists a ball with a radius of β_l contained within that region.

To reach a contradiction, let's assume there exists a region containing no training points. In this case, the center of the inscribed ball with a radius of β_l in that region would be devoid of training points within a distance of β_l. This situation contradicts the assumption that the input domain is β_l-covering. Q.E.D.

The derivation of Lemma 2 is detailed in Appendix A. The lemma shows that, for sufficiently large networks, the generalization error in classification tasks exhibits a linear behavior with respect to γ_s. The analysis is applicable to classification networks that define piecewise-constant functions over the input domain, with at least one training point assigned to each decision region of a specific class.

Let f be the target classification function. For generalization error, we use the volume of enclosing balls near the actual classification boundary, ∂f, based on the observation that when ∂f intersects with the network's decision regions, potential errors arise. Specifically, points within $2\gamma_s$ of ∂f are likely to belong to a region that could contribute to errors. This relationship emphasizes γ_s's significance in determining the network's generalization error, as shown in Figure 1(b). In addition, to measure the volume of balls proximity to ∂f, denoted by $2\gamma_s$, we need to characterize the boundary over the input domain. Here, we assume ∂f is a semi-algebraic set because a semi-algebraic set can realize an arbitrarily complicated shape.

The process of stratification involves decomposing a semi-algebraic set into a finite union of disjoint semi-algebraic subsets, known as strata, which are locally closed sub-manifolds (Benedetti et al., 1991). In a stratification, the boundary of the entire set S is composed of the union of all strata that have a lower dimension than the "top-level" interior strata. If S is a closed semi-algebraic set, then

$$\partial S = \bigcup_{X_j \cap \text{int}(S) = \emptyset} X_j \tag{1}$$

where X_j is a stratum of S and it does not touch the interior of S.

The target function f partitions the input domain into a finite union of disjoint sets, denoted as $\bigcup g(i)$ where $i \in [l]$. The set $g(i) = \{\mathbf{x} \in B_d(\underline{0}, 1) | f(\mathbf{x}) = \mathbf{e}_i\}$ corresponds to class i and is assumed to be a semi-algebraic set, meaning that its representation is a finite union of disjoint subsets, each defined by a finite combination of polynomial equations and inequalities. Formally, we can represent a stratification of $\bigcup g(i)$ by a finite collection of semi-algebraic subsets $\{f_k = \bigcup_j f_k^j\}$, where the dimension of f_k^j is k and f_k^j are disjoint semi-

algebraic sets. We refer to the volume of f_k^j as $|f_k^j|$. The concept of volume measures the size of an object: a 1-volume corresponds to length, while a 2-volume corresponds to area, and so on.

Hence, considering the boundary of the involved sets leads to the following relationship:

$$\partial f = \bigcup_{i=1}^{l} \partial g(i) = \bigcup_{k=0}^{d-1} f_k = \bigcup_{k=0}^{d-1} \bigcup_{j} f_k^j, \tag{2}$$

with $f_k = \bigcup_j f_k^j$ and

$$|f_k| = \sum_j |f_k^j|. \tag{3}$$

For example, consider the stratification of $g(1) = \{(x, y) \in \mathbb{R}^2 \mid |x| + |y| < 1\} \bigcup \{(x, y) \in \mathbb{R}^2 \mid |x| + |y| = 1\}$ with dimension 2. The first subset of that stratification is the interior of $g(1)$; therefore, that set does not contribute to the boundary of $g(1)$. We can decompose the second subset into four 1-dimensional semi-algebraic sets: $f_1^1 = \{x + y = 1 | x, y > 0\}$, $f_1^2 = \{-x + y = 1 | x < 0, y > 0\}$, $f_1^3 = \{-x - y = 1 | x, y < 0\}$, and $f_1^4 = \{x - y = 1 | x > 0, y < 0\}$, and four 0-dimensional vertices: $f_0^1 = \{(1, 0)\}$, $f_0^2 = \{(0, 1)\}$, $f_0^3 = \{(-1, 0)\}$, and $f_0^4 = \{(0, -1)\}$. The boundaries of their stratification of $g(1)$ are as follows:

$$\partial g(1) = f_1 \cup f_0 = (\cup_i f_1^i) \cup (\cup_j f_0^j). \tag{4}$$

The following lemma demonstrates that assuming the decision boundary is a semi-algebraic set leads to its decomposition into a combination of polynomial functions of varying degrees. This lemma connects the generalization error to the network's granularity, represented by γ_s and β_l, in approximating the classification boundary. For networks with a small γ_s, the decision boundary that intersects a given region can be considered a curved $(d-1)$-dimensional surface, homeomorphic to a hyperplane. The $2\gamma_s$-neighborhood of the union of the "hyperplanes" represents the area in the input domain that can contribute to a bound of generalization error.

Lemma 2. *Let f be the function $B_d(\underline{0}, 1) \to \{\mathbf{e}_1, \cdots, \mathbf{e}_l\}$, defined as $f = \bigcup_{i=1}^{l} g(i)$, where $g(i) = \{\mathbf{x} | f(\mathbf{x}) = \mathbf{e}_i\}$ is a semi-algebraic set with decision boundary ∂f adhering to the stratification equations (2) and (3). Let γ_s and β_l represent the smallest enclosing radius and largest inscribed radius of the decision regions in $B_d(\underline{0}, 1)$ of the one-hot classification network $\mathcal{M}$. Assume the training data satisfies the conditions for β_l-covering the unit d-ball and the histogram approximation. We use the notation $|S|$ to denote the volume of the set S.*
(i) For an arbitrary density function $\mathcal{P}$ on $B_d(\underline{0}, 1)$, the generalization error is given by:

$$\begin{aligned}\int_{B_d(\underline{0},1)} g(f(\mathbf{x}), \mathcal{M}(\mathbf{x}))\mathcal{P}(\mathbf{x})\, d\mathbf{x} \leq \frac{1}{N} \sum_{i=1}^{N} g(f(\mathbf{x}_i), \mathcal{M}(\mathbf{x}_i)) \\ + c_{\mathcal{P}} \sum_{k=0}^{d-1} (2\gamma_s)^{d-k} |B_{d-k}(\underline{0}, 1)||f_k|,\end{aligned} \tag{5}$$

where $g(f(\mathbf{x}), \mathcal{M}(\mathbf{x}))$ is the 0/1 error, and $c_{\mathcal{P}} \in (0, 1)$ depends on the density distributed over the $2\gamma_s$-neighborhood of ∂f.
(ii) We further suppose the volume of f_{d-1} is not zero. For small value of γ_s, we can approximate the above expression as follows:

$$\int_{B_d(\underline{0},1)} g(f(\mathbf{x}), \mathcal{M}(\mathbf{x}))\mathcal{P}(\mathbf{x})\, d\mathbf{x} \leq \frac{1}{N} \sum_{i=1}^{N} g(f(\mathbf{x}_i), \mathcal{M}(\mathbf{x}_i)) + c_{\mathcal{P}}(4\gamma_s)|f_{d-1}|. \tag{6}$$

■

4 Connecting geometry to scaling laws

4.1 Scaling laws with respect to sample size

The early work on learning curve theory by Cortes et al. (1993) illustrates how a binary classification network's performance changes as the amount of training data increases. This study reveals a scaling law related to data size: the log-log plot of generalization error versus the number of training samples, with nonzero training error, forms a straight line with a negative slope. For the MNIST dataset, the absolute value of slope ranges from $\frac{1}{2}$ to 1 (see Figures 5 and 6 of the paper).

The conclusions drawn from Lemma 2 support these findings with a regulatory assumption on the density function $\mathcal{P}$. By imposing $\mathcal{P} \geq k > 0$ over $B_{d_e}(\underline{0}, 1)$ where d_e is the effective dimension of $\mathcal{P}$, the probability mass of $\mathcal{P}$ over the ball $B_{d_e}(\mathbf{a}, \gamma)$, denoted as $P(\mathbf{a}, B_{d_e}(\mathbf{a}, \gamma))$, can be uniformly bounded below as follows:

$$P(\mathbf{a}, B_{d_e}(\mathbf{a}, \gamma)) = \int_{B_{d_e}(\mathbf{a},\gamma)} \mathcal{P}(\mathbf{x})\, d\mathbf{x} \geq |B_{d_e}(\underline{0}, \gamma)| \cdot \min_{\mathbf{x} \in B_{d_e}(\mathbf{a},\gamma)} \mathcal{P}(\mathbf{x}) \geq |B_{d_e}(\underline{0}, \gamma)| k = c\gamma^{d_e}, \tag{7}$$

where $c = |B_{d_e}(\underline{0}, 1)| k$.

Note that, in this context, we do not differentiate between effective and ambient dimensions. Therefore, we substitute the input domain dimension with the effective dimension in Lemma 2. The findings presented below indicating that the generalization error for a large network and sample size follow a scaling law with a slope of $1/d_e$.

Theorem 1. *The assumptions related to Lemma 2(ii) hold. Suppose the density function $\mathcal{P} \geq k > 0$. The generalization error with respect to sample size for a one-hot classification $\mathcal{M}$ is bounded by $\frac{4c\mathcal{P}\nu|\partial f_d|}{(cN)^{1/d_e}}$, where $\nu = \gamma_s/\beta_l$ represents the aspect ratio of the smallest enclosing ball to the largest enclosed ball within the decision regions over $B_{d_e}(\underline{0}, 1)$ incurred by the partitioning region of network $\mathcal{M}$.* ■

Proof. Let N be the number of training points that forms a β_l-covering. Then, the expected number of training points within a distance β_l at $\mathbf{x}$ is given by

$$\sum_{k=1}^{N} k \binom{N}{k} P(\mathbf{x}, B_{d_e}(\mathbf{x}, \beta_l))^k (1 - P(\mathbf{x}, B_{d_e}(\mathbf{x}, \beta_l)))^{N-k} = NP(\mathbf{x}, B_{d_e}(\mathbf{x}, \beta_l)) \geq 1 \tag{8}$$

By $\mathcal{P} \geq k > 0$ and Eq. (7), there exists a point $\mathbf{x}_0$ and integer t such that $NP(\mathbf{x}_0, B_{d_e}(\mathbf{x}_0, \beta_l)) = t \geq cN\beta_l^{d_e}$. Without loss of generality, we let $t = 1$. Consequently, we derive the inequality $\frac{\nu}{(cN)^{1/d_e}} \geq \gamma_s$. Applying this to Lemma 2(ii) provides us with the scaling law related to sample size for a large one-hot network, with the generalization error expressed in propositional form as $\frac{\nu|\partial f_d|}{(cN)^{1/d_e}}$. Q.E.D.

4.2 Box-model dimension and scaling laws with respect to network size

We have established a connection between the generalization bound and the radii of balls that represent the local geometry of a network function. Our next objective is to investigate how these radii relate to the number of parameters in a neural network.

4.2.1 Box-model dimension

The box-counting dimension is a way to measure the "ruggedness" or complexity of a shape by seeing how it occupies space at different scales. To calculate the box-counting dimension of a specific shape, we examine the relationship between the scaling factor and the number of boxes needed to cover it. The dimension D is the logarithmic power law relationship between the number of boxes and their size:

$$D = \lim_{s \to 0} \frac{\log \# B(s)}{\log(1/s)}, \tag{9}$$

where $\#B$ is the number of boxes (a grid of squares) with side length s over the object. This dimension is found by overlaying a box with side length s over the object, counting the number of boxes, then making the boxes smaller and counting again. Then, observe how the count increases as the scale s decreases. Practically, due to the box-counting dimension may not exist, or be numerically difficult to calculate, the upper box-counting dimension is applied to indicate that this set has a dimension at least $\bar{D}$ by considering

$$\bar{D} = \limsup_{s \to 0} \frac{\log \#B(s)}{\log(1/s)}. \tag{10}$$

The box-counting dimension measures the complexity of a set by quantifying how many boxes are needed to describe its shape at various scales of box size. In contrast, the box-module dimension focuses on the number of parameters, denoted as $\#W$, in model $\mathcal{M}$ that are required to construct the boxes of a given size, s. We define the box-model dimension as follows:

$$D_{\mathcal{M}} = \lim_{s \to 0} \frac{\log \#W(s)}{\log(1/s)}. \tag{11}$$

By this definition, we focus on how the radii decrease with depth as the number of parameters in a neural network increases. If we express the relationship between the number of parameters $\#W$ in network $\mathcal{M}$ and the radius γ_s as follows:

$$\gamma_s \leq \frac{1}{\#W^{1/\log_2(1+\alpha(\gamma_s))}}. \tag{12}$$

In our analysis of classification tasks, the radius γ_s is derived from the decision regions of network $\mathcal{M}$. Increasing the number of network parameters $\#W$ generates more decision regions, potentially reducing the radius. As a result, γ_s is a non-increasing function of $\#W$. The condition in Eq. (12) states that the growth rate of $\#W^{1/\log_2(1+\alpha)}$ must be slower than the decrease in $\gamma_s(\#W)$. This condition provides an upper bound of parameter size that prevents the network from overfitting or over-approximating the training points. The limitations depend on the network model and its derivation, which is characterized by the value of $\alpha(\gamma_s)$.

By taking the logarithm on both sides and then dividing by $\log_2 1/\gamma_s$:

$$\frac{\log_2 \#W}{\log_2 1/\gamma_s} \leq \log_2(1 + \alpha(\gamma_s)) \tag{13}$$

As $\gamma_s \to 0$, we obtain the upper box-model dimension:

$$\limsup_{\gamma_s \to 0} \frac{\log_2 \#W}{\log_2 1/\gamma_s} = \bar{D}_{\mathcal{M}} \to \log_2(1 + \alpha_{\mathcal{M}}). \tag{14}$$

Here, the parameter $\alpha_{\mathcal{M}} > 0$ is model-dependent and influenced by factors such as the network architecture and the specific training method used. Due to this model dependence, networks with the same input-domain configuration can produce different box-module dimensions. In Appendix B, we demonstrate that the exact configuration of the input domain, generated by shallow and deep network structures, can produce different box-module dimensions in the scaling laws (in our construction, the box-module dimension of the deep network is no greater than that of the shallow network), even though both achieve the same expected test error.

The logarithmic function in Eq. (12) emerged from our attempts to construct a network $\mathcal{M}_0$ and expand its depth to create $\mathcal{M}_1$. The goal was to ensure that γ_1 of $\mathcal{M}_1$ is reduced to less than half the value of γ_0 of $\mathcal{M}_0$, while also limiting the increase in the number of parameters in $\mathcal{M}_1$ to less than $1 + \alpha$ times of that in $\mathcal{M}_0$. Next, we apply the process recursively to decrease the enclosing radius to a desired value, γ_s. To reduce the radius from $\gamma_0 = 1$ to γ_s, we can apply the procedure L times with $L = -\log_2 \gamma_s$. Let the resulting network be $\mathcal{M}_L$. Using $\#W_{l_k} \leq (1 + \alpha)\#W_{l_{k-1}}$, with the initial condition specifying that $\#W_0 = 1$, we can deduce the number of parameters in network $\mathcal{M}_L$ with $\#W \leq (1 + \alpha)^L$. Hence, $\log_2 \#W \leq -\log_2(1 + \alpha) \log_2 \gamma_s$. This gives a relation between γ_s and $\#W$ to the power $\frac{1}{\log_2(1+\alpha)}$ with $\alpha > 0$ as depicted in Eq. (12). We note

that using the same value of α across all sub-networks and depths is not a requirement in this construction. We can easily adapt the design to accommodate different values of α in each sub-network.

Considering network $\mathcal{M}_L^0 = [I+\rho M_L]\circ[I+\rho M_{L-1}]\circ[I+\rho M_1]$, where M_l represents an affine linear mapping, ρ denotes the Rectified Linear Unit (ReLU) activation function, and I represents the identity function, which provides a direct connection in a layer. We show that it is possible to arrange the hyperplanes and the training points in such a way that the scaling law is satisfied with constant slope for any depth l in $\mathcal{M}_l^0$. A detailed derivation of this proposition can be found in Appendix C.

Proposition 1. *Let $\#W_l$ denote the number of parameters and γ_l represent the radius of the smallest enclosing balls of the regions induced by $\mathcal{P}_l^0$ in network $\mathcal{M}_l^0$. There exists a configuration of hyperplanes in layer ρM_{l+1} and a selection of training points such that the finest partition $\mathcal{P}_{l+1}^0$ of network $\mathcal{M}_{l+1}^0$ with $l \leq L-1$ is a refinement of $\mathcal{P}_l^0$ that satisfies $\gamma_{l+1} \leq \frac{\gamma_l}{2}$. Additionally, the number of parameters in $\mathcal{M}_{l+1}$ and $\mathcal{M}_l$ can be constrained by the inequality $\#W_{l+1} \leq (1+\alpha)\#W_l$, where $\alpha \geq 1$.* ■

4.2.2 Scaling laws with respect to network size and experiments

Now, let's examine the implications of condition on Eq. (12) on generalization bounds. By substituting that condition into Lemma 2 for γ_s, the generalization bounds become inversely proportional to the number of parameters in a network.

Theorem 2. *Suppose the classification boundary ∂f of target classification function $f : B_d(\underline{0},1) \to \{\mathbf{e}_1, \cdots, \mathbf{e}_l\}$ is a semi-algebraic set adhering to the stratification equations (2) and (3). Let γ_s and β_l denote the smallest enclosing radius and the largest inscribed radius of input domain partition regions derived by one-hot classification network $\mathcal{M} : B_d(\underline{0},1) \to \{\mathbf{e}_1, \cdots, \mathbf{e}_l\}$. We assume that the training data meets the assumptions of β_l-covering the unit d-ball and the histogram approximation. Let $\#W_{\mathcal{M}}$ represent the number of parameters in $\mathcal{M}$. Assume that $\gamma_s \leq \frac{1}{\#W_{\mathcal{M}}^{1/\log_2(1+\alpha_{\mathcal{M}})}}$ for some $\alpha_{\mathcal{M}} > 0$, which depends on the learning method used to obtain $\mathcal{M}$ from N i.i.d. training data points $\{(\mathbf{x}_i, f(\mathbf{x}_i))\}$ from $\mathcal{P}$. Let $|S|$ denote the volume of the set S.*
(i) We can bound the generalization error between classification function f and $\mathcal{M}$ as follows:

$$\int_{B_d(\underline{0},1)} g(f(\mathbf{x}), \mathcal{M}(\mathbf{x}))\mathcal{P}(\mathbf{x})\, d\mathbf{x} \leq \frac{1}{N}\sum_{i=1}^{N} g(f(\mathbf{x}_i), \mathcal{M}(\mathbf{x}_i)) + c_{\mathcal{P}} \sum_{k=0}^{d-1} \left(\frac{2}{\#W_{\mathcal{M}}^{1/\log_2(1+\alpha_{\mathcal{M}})}}\right)^{d-k} |B_{d-k}(\underline{0},1)||f_k|, \tag{15}$$

where $g(f(\mathbf{x}), \mathcal{M}(\mathbf{x}))$ is the 0/1 error and $c_{\mathcal{P}} \in (0,1)$ depends on the density distributed over the $2\gamma_s$-neighborhood of ∂f.
(ii) We further suppose the volume of f_{d-1} is not zero. For small value of γ_s, we can approximate the above expression as follows:

$$\int_{B_d(\underline{0},1)} g(f(\mathbf{x}), \mathcal{M}(\mathbf{x}))\mathcal{P}(\mathbf{x})\, d\mathbf{x} \leq \frac{1}{N}\sum_{i=1}^{N} g(f(\mathbf{x}_i), \mathcal{M}(\mathbf{x}_i)) + \frac{4c_{\mathcal{P}}|f_{d-1}|}{\#W_{\mathcal{M}}^{1/\log_2(1+\alpha_{\mathcal{M}})}}. \tag{16}$$

■

Estimating the bound presents particular challenges. However, in the log-log plots of the generalization bounds in Eq. (16) against the number of parameters, the slope is $-1/\log_2(1+\alpha_{\mathcal{M}})$. This slope serves as an index of how efficiently a network family performs as its parameter count increases while reducing generalization error. We will use the slope derived from smaller models to predict the performance of a larger model.

If we approximate the generalization error with the bound, we can use the following formulation to estimate $\alpha_{\mathcal{M}}$ from smaller models with known generalization errors and numbers of parameters:

$$\left(\frac{ge_1}{ge_2}\right)^{\log_2(1+\alpha_{\mathcal{M}})} \approx \frac{\#W_2}{\#W_1}, \tag{17}$$

where ge_i and $\#W_i$ represent the generalization error and the number of parameters for model i, respectively. This approximation does not necessarily require operating in the interpolation zone, where the training error is zero or nearly zero. However, within the interpolation zone, the generalization error can be approximated by the expected test error. Let $\hat{\alpha}$ represent the estimated value of $\alpha_{\mathcal{M}}$. We use this estimated value to predict the generalization error $\hat{ge}_{i+1}$ for model $i+1$: $\left(\frac{ge_i}{\hat{ge}_{i+1}}\right)^{\log_2(1+\hat{\alpha})} \approx \frac{\#W_{i+1}}{\#W_i}$. To evaluate the accuracy of this prediction, we calculate the relative error using the formula: $\frac{|\hat{ge}_{i+1} - ge_{i+1}|}{ge_{i+1}} \times 100\%$.

Findings on ResNet and VGG models for ImageNet and CIFAR-100

We evaluate the extrapolation performance of this approach on large models with numerous training points, which is more likely to align with our assumption that the histogram approximates the average density function, and less likely to impose a limitation on the depth increment in prediction.

We illustrate these scaling laws using experimental results from ResNet and VGG models on the ImageNet and CIFAR-100 datasets, drawn from various benchmark implementations, related papers, and reports. The learning algorithms discussed in those documents were implemented with precision, consistently achieving minimal training errors. This enables us to assume that the expected test error closely mirrors the generalization error and directly relates our findings to researchers and practitioners in the field.

The absolute value of the slope measures the change in test errors per infinitesimal increase in the number of model parameters on the log-log scale. In our analysis of VGGs and ResNets on ImageNet, we found that the absolute value of the slope for VGGs ≈ 1.5 is larger than that for ResNets ≈ 0.2. This result suggests that VGGs can achieve about a 7-fold reduction in test error for each additional parameter on a log-log scale. Nevertheless, this efficiency comes at the expense of increased parameter redundancy, as the VGG model has over 100M parameters.

I. ResNets on ImageNet:

ImageNet is a vast visual database containing over 14 million images across more than 20,000 categories, aimed at advancing research in object recognition. Table 1 and Figure 2 illustrate the predicted Top-1 Error for the ResNet model on this dataset. The estimated value of α from ResNet-18 and ResNet-34 is 35.14. This results in $\log_2(1+\hat{\alpha}) = 5.188$ and a slope of $-\frac{1}{\log_2(1+\hat{\alpha})} = -0.19$. This slope suggests that for the test error of a ResNet network to be reduced to half its original value, the number of parameters would have to increase to ten times its original size. We use this slope value to derive the predicted Top-1 error presented in the table and figure.

Model-Depth	Param. (M)	Test Error (%)	Pred. Test Error (%)	Relative Error (%)
ResNet-18	11.7	30.1	x	x
ResNet-34	21.8	26.7	x	x
ResNet-50	25.6	23.8	25.9	8.82
ResNet-101	44.5	22.6	21.4	5.31
ResNet-152	60.2	21.7	21.3	1.84

Table 1: Predict the top-1 test error of ResNet models on the ImageNet database, which contains about 6% label errors in the validation set Northcutt et al. (2021). The number of parameters and top-1 test errors were obtained from the original paper by He et al. (2016) as well as various benchmark re-implementations Wightman et al. (2021), including those in PyTorch and TensorFlow PyTorch Development Team (2025). There may be slight variations depending on the training setup, such as data augmentation techniques and learning rate schedules. We can assess the investment in larger models by analyzing the results predicted by a smaller model, with a relative error rate of less than 10%.

II. ImageNet on CIFAR-100:

The CIFAR-100 dataset consists of 60,000 color images, each measuring 32x32 pixels, and it includes 100 different classes, with 600 images per class. The dataset is divided into 50,000 training images and 10,000 test images. The Top-1 Error, as shown in Table 2, was obtained from the implementation of ResNet models

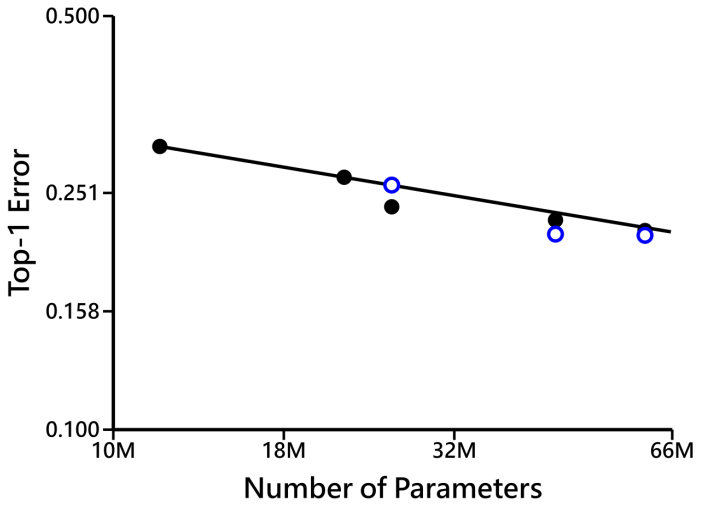

Figure 2: The log-log plot of the data presented in Table 1. The x-axis labels represent the number of parameters, while the y-axis labels indicate the top-1 test error. Neither label is on a logarithmic scale for clarity. The line passing through the two leftmost solid points, generated by the smaller models (ResNet-18, ResNet-34), was used to predict the performance of the larger models (ResNet-50, ResNet-101, ResNet-152). The actual data points are represented in solid, while the predicted points are shown in empty circles.

on the CIFAR-100 dataset, as reported by Jafar and Lee (2023). The slope derived from the ResNet-20 and ResNet-34 models is -0.15, with an estimated value of α being 93.2.

Model-Depth	Param. (M)	Test Error (%)	Pred. Test Error (%)	Relative Error (%)
ResNet-20	0.27	30.31	x	x
ResNet-34	0.46	27.95	x	x
ResNet-44	0.66	27.02	26.44	2.14
ResNet-101	1.7	23.86	23.39	1.96

Table 2: Predict the top-1 test error for ResNet models on the CIFAR-100 database, which contains approximately 5.85% label errors in the validation set Northcutt et al. (2021). We obtained the number of parameters and corresponding top-1 test errors from the original paper Jafar and Lee (2023). Note that the number of parameters in this table differs from those in Table 1 due to variations in the width of each layer and other adjustments. The relative errors for the ResNet-44 and ResNet-101 models are both less than 2.2%.

III. VGGs on ImageNet:

In our analysis of the VGG family of models tested on ImageNet, we first examine the least-squares regression lines presented on a log-log scale. The regression lines adhere to scaling laws, displaying slopes of -1.21 for models that utilize Batch Normalization (BN) and -1.11 for those that do not. This difference in slope values indicates that the BN model is slightly more efficient with its parameters than the one without BN, suggesting a faster rate of decrease in test error.

The VGG implementations on ImageNet exhibit a clear training error. However, as long as the training error is comparably small to the test error, we can still apply Eq. (17) to obtain an estimate of the slope for smaller models. The values of α were found to be 0.551 for models without BN and 0.478 for those with BN. The slope for models without BN is -1.581, while the slope for models utilizing BN is -1.784. The values are lower but consistent with those obtained using the aforementioned least-squares method for three observations. Tables 3 and 4 present the prediction performance results, and the performance differences shown in the tables primarily result from the implementation of BN.

Model-Depth	Param. (M)	Test Error (%)	Pred. Test Error (%)	Relative Error (%)
VGG-13	133	30.1	x	x
VGG-16	138	28.4	x	x
VGG-19	144	27.6	26.55	3.8

Table 3: Predict top-1 test error of VGG family on the ImageNet database without Batch Normalization. The number of parameters and top-1 test errors were obtained from the original paper by Simonyan and Zisserman (2014) as well as various benchmark re-implementations, including those in PyTorch and TensorFlow Paszke et al. (2019); PyTorch Development Team (2025). There may be slight variations depending on the training setup, such as data augmentation techniques and learning rate schedules.

Model-Depth	Param. (M)	Test Error (%)	Pred. Test Error (%)	Relative Error (%)
VGG-13-BN	133	28.4	x	x
VGG-16-BN	138	26.6	x	x
VGG-19-BN	144	25.8	24.66	4.4

Table 4: Predict the top-1 test error for the VGG family of models on the ImageNet database when using Batch Normalization Paszke et al. (2019); PyTorch Development Team (2025). The original VGG paper did not incorporate BN, which was introduced later by Ioffe and Szegedy (2015). The numbers in the table may vary slightly due to the training setup, including factors like data augmentation techniques and learning rate schedules.

5 Analysis of primary assumptions with random balls

Our earlier analysis in Lemma 2 relies on two primary assumptions. We carefully assess the conditions needed to validate these assumptions and explore their connections to the concentration results that underpin the generalization error bounds. In particular, we address the probability that random balls of a fixed radius densely cover the unit d-ball. Furthermore, we derive the concentration bound necessary for the histogram approximation assumption.

We use the random process $X(\gamma)$ to generate centers for balls of radius γ, one at a time. These centers are drawn independently based on the density function $\mathcal{P}$. We denote S as the set of centers generated by this process, and $X_{min}(S)$ as the smallest subset of S that forms a covering of the input domain with balls of radius γ.

The first lemma below discusses the probability of a sequence covering the unit d-ball randomly for the first time, while the following lemma focuses on a concentration inequality to approximate the unknown density function using histogram approximation. The proofs of the first lemma and the second lemma are provided in Appendices D and E, respectively.

Lemma 3. *Assume the density function $\mathcal{P} \geq k > 0$ for some k over $B_d(\underline{0}, 1)$. The probability of the initial $N \geq 1/(c\gamma^d)$ samples in $X(\gamma)$ forming a cover of the unit d-ball is*

$$P(|X_{\min}| \leq N) \geq 1 - \frac{1}{cN\gamma^d}, \tag{18}$$

for some constant c satisfying $P(\mathbf{x}, B_d(\mathbf{x}, \gamma)) \geq c\gamma^d$ for any $\mathbf{x} \in B_d(\underline{0}, 1)$ (see Eq. (7)). ■

Let $R(S)$ denote the collection of balls with centers in $X_{min}(S)$ and radius γ. We next address the concentration inequality to approximate the following term with the histogram for $p \in R(S)$:

$$\int_p \mathcal{I}(\mathbf{x} \in p) P(\mathbf{x})\, d\mathbf{x}, \tag{19}$$

where $\mathcal{I}(\mathbf{x} \in p)$, taking value 1 or 0, is an indicator function for $\mathbf{x} \in p$. In our previous analysis, $\int_p \mathcal{I}(\mathbf{x} \in p) P(\mathbf{x}) d\mathbf{x}$ is replaced with $\frac{n_p}{N}$, the portion of N training points assigned to a ball falling within the ball (though the balls can overlap, each training point is assigned to exactly one ball).

Lemma 4. *Consider process $X(\gamma)$. Denote S contains a sequence of N points with $|X_{min}(S)| \leq N$. The inequality for the density function $\mathcal{P}$ approximated by histogram is*

$$P(|\int_p \mathcal{I}(\mathbf{x} \in p)\mathcal{P}(\mathbf{x})d\mathbf{x} - \frac{n_p}{N}| \geq t_p) \leq 2\exp(-2Nt_p^2). \tag{20}$$

If t_p goes to zero at a rate slower than N goes to infinity, then in the limit as $N \to \infty$, we have $\int_p \mathcal{I}(\mathbf{x} \in p)\mathcal{P}(\mathbf{x})\, d\mathbf{x} = \frac{n_p}{N}$. ■

Proposition 2. *Assume $\mathcal{P} \geq k > 0$ for some k over $B_d(\underline{0}, 1)$. Consider the random ball model $X(\gamma)$, the assumptions of γ-covering and the histogram approximation hold for any $\gamma > 0$, when $N \to \infty$.* ■

Proof. Using $P(A, B) = P(A)P(B|A)$ where $A = |X_{min}| \leq N$ denotes the γ-covering of the input domain and $B = |\int_p \mathcal{I}(\mathbf{x} \in p)\mathcal{P}(\mathbf{x})d\mathbf{x} - \frac{n_p}{N}| < t_p$ for any $t_p > 0$ and p denotes a region of $R(S)$.

According to Lemma 3, $P(|X_{min}| \leq N) \geq 1$, as $N \to \infty$. Hence, the assumption of γ-covering is affirmative. In addition, Lemma 4 asserts that the Hoeffding's inequality as described in Eq. (20) indicates that for any region and any $t_p > 0$ and $N \to \infty$,

$$P(|\int_p \mathcal{I}(\mathbf{x} \in p)\mathcal{P}(\mathbf{x})d\mathbf{x} - \frac{n_p}{N}| < t_p \big| \, |X_{min}| \leq N) > 1. \tag{21}$$

This means that $\int_p \mathcal{I}(\mathbf{x} \in p)\mathcal{P}(\mathbf{x})d\mathbf{x} = \frac{n_p}{N}$. Hence $P(A, B) = 1$ as $N \to \infty$. Q.E.D.

6 Conclusions

Our analysis of scaling laws for classification networks focuses on three key considerations: the covering radius of training-point balls, the exact histogram approximation of the regional average of the unknown density function, and the relationship between ball radius and network parameters. We demonstrate that the scaling laws for sample and network sizes depend on different factors. The law governing sample size depends on the effective or intrinsic dimension of data manifolds, independent of network models and learning methods. On the other hand, we introduce the concept of box-module dimension to quantify the complexity of a network in generating a covering for a partition shape within a specified radius. By applying the conditions that characterize this dimension, we can derive a law that relates it to the size of the network. This law is model-dependent, and we can utilize it to estimate the resources needed to increase the model's depth and to adjust parameter settings in order to achieve the desired level of accuracy.

In light of these findings, we propose the following question: Can we extend our analysis to explore the behavior of networks in a space defined by the dimensions of generalization error, the number of parameters, and the number of training points, within a Probably Approximately Correct (PAC) framework? By doing so, we may uncover additional patterns or principles that govern network performance, such as the conditions under which the scaling laws cease to apply, thereby deepening our understanding of the complex behaviors observed empirically.

References

Alessio Ansuini, Alessandro Laio, Jakob H Macke, and Davide Zoccolan. Intrinsic dimension of data representations in deep neural networks. *Advances in Neural Information Processing Systems*, 32, 2019.

Yasaman Bahri, Ethan Dyer, Jared Kaplan, Jaehoon Lee, and Utkarsh Sharma. Explaining neural scaling laws. *Proceedings of the National Academy of Sciences*, 121(27):e2311878121, 2024.

Peter L Bartlett, Philip M Long, Gábor Lugosi, and Alexander Tsigler. Benign overfitting in linear regression. *Proceedings of the National Academy of Sciences*, 117(48):30063–30070, 2020.

Riccardo Benedetti, Jean-Jacques Risler, et al. *Real algebraic and semialgebraic sets*. Hermann, 1991.

Devansh Bisla, Apoorva Nandini Saridena, and Anna Choromanska. A theoretical-empirical approach to estimating sample complexity of dnns. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3270–3280, 2021.

Stephen P Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

Leshem Choshen, Yang Zhang, and Jacob Andreas. A hitchhiker's guide to scaling law estimation, 2024. *URL https://arxiv. org/abs/2410.11840*, 2024.

Corinna Cortes, Lawrence D Jackel, Sara Solla, Vladimir Vapnik, and John Denker. Learning curves: Asymptotic values and rate of convergence. *Advances in neural information processing systems*, 6, 1993.

Hugo Cui, Bruno Loureiro, Florent Krzakala, and Lenka Zdeborová. Generalization error rates in kernel regression: The crossover from the noiseless to noisy regime. *Advances in Neural Information Processing Systems*, 34:10131–10143, 2021.

Alan M Frieze and Shang-Hua Teng. On the complexity of computing the diameter of a polytope. *computational complexity*, 4(3):207–219, 1994.

Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric algorithms and combinatorial optimization*, volume 2. Springer Science & Business Media, 2012.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Joel Hestness, Sharan Narang, Newsha Ardalani, Gregory Diamos, Heewoo Jun, Hassan Kianinejad, Md Mostofa Ali Patwary, Yang Yang, and Yanqi Zhou. Deep learning scaling is predictable, empirically. *arXiv preprint arXiv:1712.00409*, 2017.

Marcus Hutter. Learning curve theory. *arXiv preprint arXiv:2102.04074*, 2021.

Wen-Liang Hwang and Shih-Shuo Tung. Analysis of function approximation and stability of general dnns in directed acyclic graphs using un-rectifying analysis. *Electronics (Invited paper)*, 12(18):3858, 2023.

Abbas Jafar and Myungho Lee. High-speed hyperparameter optimization for deep resnet models in image recognition. *Cluster Computing*, 26(5):2605–2613, 2023.

Elizaveta Levina and Peter Bickel. Maximum likelihood estimation of intrinsic dimension. *Advances in neural information processing systems*, 17, 2004.

Houyi Li, Wenzhen Zheng, Qiufeng Wang, Hanshan Zhang, Zili Wang, Shijie Xuyang, Yuantao Fan, Zhenyu Ding, Haoying Wang, Ning Ding, et al. Predictable scale: Part i, step law–optimal hyperparameter scaling law in large language model pretraining, 2025. *URL https://arxiv. org/abs/2503.04715*, 2025.

Alexander Maloney, Daniel A Roberts, and James Sully. A solvable model of neural scaling laws. *arXiv preprint arXiv:2210.16859*, 2022.

Curtis G Northcutt, Anish Athalye, and Jonas Mueller. Pervasive label errors in test sets destabilize machine learning benchmarks. *arXiv preprint arXiv:2103.14749*, 2021.

Junhyung Park, Patrick Bloebaum, and Shiva Prasad Kasiviswanathan. A classical view on benign overfitting: The role of sample size. *arXiv preprint arXiv:2505.11621*, 2025.

Razvan Pascanu, Guido Montufar, and Yoshua Bengio. On the number of response regions of deep feed forward networks with piece-wise linear activations. *arXiv preprint arXiv:1312.6098*, 2013.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chillemi, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Fowler, et al. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pages 8024–8035, 2019.

PyTorch Development Team. TorchVision models and pre-trained weights. `https://pytorch.org/vision/stable/models.html`, 2025. Accessed: 21 Oct. 2025.

Jonathan S Rosenfeld, Amir Rosenfeld, Yonatan Belinkov, and Nir Shavit. A constructive prediction of the generalization error across scales. *arXiv preprint arXiv:1909.12673*, 2019.

Jonathan S Rosenfeld, Jonathan Frankle, Michael Carbin, and Nir Shavit. On the predictability of pruning across scales. In *International conference on machine learning*, pages 9075–9083. PMLR, 2021.

Utkarsh Sharma and Jared Kaplan. Scaling laws from the data manifold dimension. *Journal of Machine Learning Research*, 23(9):1–34, 2022.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Ross Wightman, Hugo Touvron, and Hervé Jégou. Resnet strikes back: An improved training procedure in timm. *arXiv preprint arXiv:2110.00476*, 2021.

Yizhe Xiong, Xiansheng Chen, Xin Ye, Hui Chen, Zijia Lin, Haoran Lian, Zhenpeng Su, Wei Huang, Jianwei Niu, Jungong Han, et al. Temporal scaling law for large language models. *arXiv preprint arXiv:2404.17785*, 2024.

A Proof of Lemma 2

One-hot classifiers are piecewise constant functions, assigning inputs to a single class i within the range of $\{1, \cdots, l\}$. The standard coordinate basis $\mathbf{e}_i$ represents class i in a one-hot coding system. To compute the 0/1 error function, we can use the formula below: for any $f(\mathbf{y}), f(\mathbf{z}) \in \{\mathbf{e}_1, \cdots, \mathbf{e}_l\}$,

$$g(f(\mathbf{y}), f(\mathbf{z})) = \frac{1}{\sqrt{2}}\|f(\mathbf{y}) - f(\mathbf{z})\|_2 = \begin{cases} 1 \text{ if } f(\mathbf{y}) \neq f(\mathbf{z}) \\ 0 \text{ otherwise.} \end{cases} \tag{22}$$

For a given classifier f, a ball exists around any point $\mathbf{x}$ in the input domain, where all the points inside this ball have the same class as that of $\mathbf{x}$. If $\mathbf{x}$ lies on the classification boundary of f, the radius of the ball is zero. We use the notation $\eta_f(\mathbf{x}) \geq 0$ to denote the maximum radius of the ball centered at $\mathbf{x}$ such that $f(\mathbf{y}) = f(\mathbf{x})$ for all $\mathbf{y}$ belonging to $B_d(\mathbf{x}, \eta_f(\mathbf{x}))$. Therefore, the class of f at any point $\mathbf{y}$ is the same as that of $\mathbf{x}$ when the distance between $\mathbf{x}$ and $\mathbf{y}$ is less than or equal to $\eta_f(\mathbf{x})$. Hence, we have

$$h_f(\mathbf{x}; \|\mathbf{x} - \mathbf{y}\|_2) = g(f(\mathbf{x}), f(\mathbf{y})) = 0 \text{ if } \|\mathbf{x} - \mathbf{y}\|_2 \leq \eta_f(\mathbf{x}). \tag{23}$$

We can bound h_f with the following radius function:

$$\hat{h}_f(\mathbf{x}; \|\mathbf{x} - \mathbf{y}\|_2) = \begin{cases} 1 \text{ if } \|\mathbf{x} - \mathbf{y}\|_2 > \eta_f(\mathbf{x}) \\ 0 \text{ otherwise.} \end{cases} \quad \geq h_f(\mathbf{x}; \|\mathbf{x} - \mathbf{y}\|_2). \tag{24}$$

$\hat{h}_f(\mathbf{x}; \|\mathbf{x} - \mathbf{y}\|_2)$ is a non-decreasing function of $\|\mathbf{x} - \mathbf{y}\|_2$ for any function f. In the case where $\mathbf{x}$ is on the classification boundary, $\hat{h}_f(\mathbf{x}; \|\mathbf{x} - \mathbf{y}\|_2)$ will always be equal to 1 for any value of $\mathbf{y} \neq \mathbf{x}$.

The classification network $\mathcal{M}$ is responsible for learning the classifier function f by utilizing N training data pairs $(\mathbf{x}_i, f(\mathbf{x}_i))$. The function of $\mathcal{M}$ is piece-wise constant, dividing the input domain into non-overlapping regions and assigning each region a specific class. The collection of all these regions is referred to as R_s. Each region in R_s is associated with two geometric parameters related to balls: the smallest enclosing radius and the largest inscribed radius. The smallest enclosing radius that encompasses all regions in R_s and the largest inscribed radius for all regions in R_s are denoted as γ_s and β_l, respectively.

The training points are β_l-covering distributed means that every region of R_s must have at least one training point (see Lemma 1). For any input point $\mathbf{x}$ located in region p, let $\mathbf{x}_p$ represent a training point within

that region. The distance between $\mathbf{x}$ and $\mathbf{x}_p$ will always be less than or equal to $2\gamma_s$, since the radius of the enclosing ball for the region is γ_s. To calculate the error between $f(\mathbf{x})$ and $\mathcal{M}(\mathbf{x})$ with respect to $\mathbf{x}_p$, we can use the following equation:

$$\begin{aligned} g(f(\mathbf{x}), \mathcal{M}(\mathbf{x})) &= \frac{1}{\sqrt{2}}(\|f(\mathbf{x}) - \mathcal{M}(\mathbf{x}) + f(\mathbf{x}_p) - f(\mathbf{x}_p) + \mathcal{M}(\mathbf{x}_p) - \mathcal{M}(\mathbf{x}_p)\|_2) \\ &\leq \frac{1}{\sqrt{2}}(\|f(\mathbf{x}) - f(\mathbf{x}_p)\|_2 + \|f(\mathbf{x}_p) - \mathcal{M}(\mathbf{x}_p)\|_2 + \|\mathcal{M}(\mathbf{x}_p) - \mathcal{M}(\mathbf{x})\|_2) \\ &\leq g(f(\mathbf{x}), f(\mathbf{x}_p)) + \varepsilon_p. \end{aligned} \tag{25}$$

Here, we define $\frac{1}{\sqrt{2}}\|f(\mathbf{x}_p) - \mathcal{M}(\mathbf{x}_p)\|_2$ as $g(f(\mathbf{x}_p), \mathcal{M}(\mathbf{x}_p)) = \varepsilon_p$. As points in the same region of P_s of $\mathcal{M}$ have the same class, the term $\|\mathcal{M}(\mathbf{x}_p) - \mathcal{M}(\mathbf{x})\|_2$ is zero.

Denote $\{\mathbf{x}_{p,i}\}_{i=1}^{n_p}$ as the set of n_p training points in region p. By applying Eq. (25) to each training point in the region and then computing the average, we can obtain the following result:

$$\begin{aligned} g(f(\mathbf{x}), \mathcal{M}(\mathbf{x})) &\leq \frac{1}{n_p}\sum_{i=1}^{n_p} g(f(\mathbf{x}), f(\mathbf{x}_{p,i})) + \bar{\varepsilon}_p \\ &\leq \max_i g(f(\mathbf{x}), f(\mathbf{x}_{p,i})) + \bar{\varepsilon}_p. \end{aligned} \tag{26}$$

To assess the expected error for probability distribution $\mathcal{P}$, we use Eq. (26) for each point $\mathbf{x}$ in a region p, and then sum over all regions of R_s to obtain:

$$\begin{aligned} \int_{B_d(\underline{0},1)} g(f(\mathbf{x}), \mathcal{M}(\mathbf{x}))\mathcal{P}(\mathbf{x})\, d\mathbf{x} &= \sum_{p\in R_s}\int_p g(f(\mathbf{x}), \mathcal{M}(\mathbf{x}))I(\mathbf{x}\in p)P(\mathbf{x})\, d\mathbf{x} \\ &\leq \sum_{p\in R_s}\int_p \max_i g(f(\mathbf{x}), f(\mathbf{x}_{p,i}))\mathcal{I}(\mathbf{x}\in p)\mathcal{P}(\mathbf{x})d\mathbf{x} \\ &+ \sum_{p\in R_s}\bar{\varepsilon}_p\int_p \mathcal{I}(\mathbf{x}\in p)\mathcal{P}(\mathbf{x})\, d\mathbf{x}. \end{aligned} \tag{27}$$

In the equation, $\bar{\varepsilon}_p$ represents the mean of the prediction error, which is a constant in region p, and the equality holds because R_s partitions the input domain. The left side indicates the generalization error, while the right comprises two integrals. The first integral is the maximum value of the function g calculated with respect to all training points $\mathbf{x}_{p,i}$ in region p. The second term in Eq. (27) can be further derived using the assumption of histogram approximation to relate this term to the average of all training points:

$$\begin{aligned} \sum_{p\in P_s}\bar{\varepsilon}_p\int_p \mathcal{I}(\mathbf{x}\in p)\mathcal{P}(\mathbf{x})\, d\mathbf{x} &= \sum_{p\in P_s}\bar{\varepsilon}_p\frac{n_p}{N} = \frac{1}{N}\sum_{p\in P_s}\sum_{i=1}^{n_p}\|f(\mathbf{x}_{p,i}) - \mathcal{M}(\mathbf{x}_{p,i})\|_2 \\ &= \frac{1}{N}\sum_{i=1}^{N}\|f(\mathbf{x}_i) - \mathcal{M}(\mathbf{x}_i)\|_2. \end{aligned} \tag{28}$$

Thus, Eq. (27) can be expressed as:

$$\begin{aligned} \int_{B_d(\underline{0},1)} g(f(\mathbf{x}), \mathcal{M}(\mathbf{x}))\mathcal{P}(\mathbf{x})\, d\mathbf{x} &\leq \sum_{p\in R_s}\int_p \max_i g(f(\mathbf{x}), f(\mathbf{x}_{p,i}))\mathcal{I}(\mathbf{x}\in p)\mathcal{P}(\mathbf{x})d\mathbf{x} \\ &+ \frac{1}{N\sqrt{2}}\sum_i\|f(\mathbf{x}_i) - \mathcal{M}(\mathbf{x}_i)\|_2 \\ &= \sum_{p\in R_s}\int_p \max_i g(f(\mathbf{x}), f(\mathbf{x}_{p,i}))\mathcal{I}(\mathbf{x}\in p)\mathcal{P}(\mathbf{x})d\mathbf{x} \\ &+ \frac{1}{N}\sum_i g(f(\mathbf{x}_i), \mathcal{M}(\mathbf{x}_i)). \end{aligned} \tag{29}$$

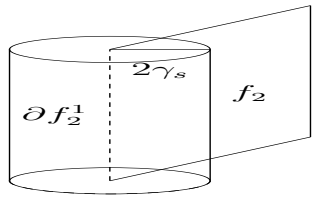

Figure 3: The boundary of closed surface f_2 consists of four sub-components under stratification, and its dimension is 2. The central axis (represented as a dashed line segment) of the cylinder in $\mathbb{R}^3$ is the first part of the boundary of f_2, denoted as ∂f_2^1. This axis has a dimension of 1, while the radius of the cylinder is $2\gamma_s$. The volume of the cylinder can be expressed as $|\partial f_2^1|(2\gamma_s)^2\pi$.

We can utilize two facts to calculate an upper bound for the first term in Eq. (29). The first is that $\|\mathbf{x} - \mathbf{x}_{p,i}\|_2 \leq 2\gamma_s$, and the second is that $\hat{h}_f$ is a non-decreasing function, as per Eqs. (24) and (23). By using these facts, we obtain the following inequality:

$$\begin{aligned}\sum_{p\in R_s}\int_p \max_i g(f(\mathbf{x}), f(\mathbf{x}_{p,i}))\mathcal{I}(\mathbf{x}\in p)\mathcal{P}(\mathbf{x})d\mathbf{x} &\leq \sum_{p\in R_s}\int_p \max_i \hat{h}_f(\mathbf{x}; \|\mathbf{x}-\mathbf{x}_{p,i}\|_2)\mathcal{I}(\mathbf{x}\in p)\mathcal{P}(\mathbf{x})d\mathbf{x} \\ &\leq \int \hat{h}_f(\mathbf{x}; 2\gamma_s)\mathcal{P}(\mathbf{x})d\mathbf{x} \\ &\leq \int 1([2\gamma_s \geq \eta_f(\mathbf{x})])\mathcal{P}(\mathbf{x})d\mathbf{x}. \end{aligned} \tag{30}$$

Here, $1([2\gamma_s \geq \eta_f(\mathbf{x})])$ is an indicator function for $2\gamma_s \geq \eta_f(\mathbf{x})$, and the last inequality is a result of Eq. (24). For function f, the value of $\eta_f(\mathbf{x})$ remains fixed. Moreover, the function $1([2\gamma_s \geq \eta_f(\mathbf{x})])$ is an increasing function of γ_s. Therefore, a network $\mathcal{M}$ with a lower value of γ_s can lead to a reduced generalization bound.

(i) The probability mass that contributes to $\int 1([2\gamma_s \geq \eta_f(\mathbf{x})])\mathcal{P}(\mathbf{x})d\mathbf{x}$, which can generate generalization error, can be bounded. The points satisfying $2\gamma_s \geq \eta_f(\mathbf{x})$ are within a distance no greater than $2\gamma_s$ from the classification boundary of f, denoted by ∂f, as the schematic diagram illustrated in Figure 1(b). The boundary can be decomposed as the union of semi-algebraic subsets as follows:

$$\partial f = \bigcup_{k=0}^{d-1} f_k = \bigcup_{k=0}^{d-1}\bigcup_i f_k^i,$$

where f_k^i is a k-dimensional semi-algebraic set. We define the $2\gamma_s$-neighborhood of ∂f in ℓ_2-norm as

$$A(2\gamma_s, \partial f) = \{\mathbf{x}\in B_d(\underline{0}, 1) \mid \mathrm{dist}(\mathbf{x}, \partial f) \leq 2\gamma_s\} = \bigcup_{k=0}^{d-1} A(2\gamma_s, f_k) \tag{31}$$

and the $2\gamma_s$-neighborhood of $f_k = \bigcup_i f_k^i$ is defined as

$$A(2\gamma_s, f_k) = \bigcup_i A(2\gamma_s, f_k^i) = \bigcup_i \{\mathbf{x}\in B_d(\underline{0}, 1) \mid \mathrm{dist}(\mathbf{x}, f_k^i) \leq 2\gamma_s\}.$$

The volume of $A(2\gamma, f_k)$ can be bounded in accordance with the union bound and $\dim f_k^j = k$ as

$$\begin{aligned}|A(2\gamma_s, f_k)| &\leq \sum_i |A(2\gamma_s, f_k^i)| \leq (2\gamma_s)^{d-k}|B_{d-k}(\underline{0}, 1)|\sum_i |f_k^i| \\ &= (2\gamma_s)^{d-k}|B_{d-k}(\underline{0}, 1)||f_k|. \end{aligned} \tag{32}$$

See the illustrative example on calculating $|A(2\gamma_s, f_2)|$ in Figure 3. In the above, we utilize the fact that $|A(2\gamma_s, f_k^i)|$ equals $|B_{d-k}(\underline{0}, 2\gamma_s)||f_k^i|$, where $|B_{d-k}(\underline{0}, 2\gamma_s)| = (2\gamma_s)^{d-k}|B_{d-k}(\underline{0}, 1)|$.

By applying the union bound to Eq. (31) and then to Eq. (32), we derive a bound for the volume of the $2\gamma_s$-neighborhood of ∂f:

$$|A(2\gamma_s, \partial f)| \leq \sum_{k=0}^{d-1} |A(2\gamma_s, f_k)| \leq \sum_{k=0}^{d-1} (2\gamma_s)^{d-k} |B_{d-k}(\underline{0}, 1)||f_k|.$$

Consequently,

$$\int 1([2\gamma_s \geq \eta_f(\mathbf{x})])\mathcal{P}(\mathbf{x})d\mathbf{x} = c_{\mathcal{P}}|A(2\gamma_s, \partial f)| \leq c_{\mathcal{P}} \sum_{k=0}^{d-1} (2\gamma_s)^{d-k} |B_{d-k}(\underline{0}, 1)||f_k|. \tag{33}$$

The first equality bounds the probability mass by the volume of the points over the $2\gamma_s$-neighborhood of ∂f, where $c_p \in (0, 1)$ represents the average density of $\mathcal{P}$ distributed across the neighborhood.

(ii) When γ_s is small and $|f_{d-1}| \neq 0$, the sum $\sum_{k=0}^{d-1} (2\gamma_s)^{d-k} |B_{d-k}(\underline{0}, 1)||f_k|$ in equation (33) can be approximated by the leading term $(2\gamma_s)|B_1(\underline{0}, 1)||f_{d-1}| = 4\gamma_s|f_{d-1}|$ with $|B_1(\underline{0}, 1)| = 2$ (the length of a segment).

B Box-module dimensions of shallow and deep networks

We develop a DNN focusing on increasing depth, in contrast to the wider configurations discussed by Sharma and Kaplan (2022).

Assume that the training points are distributed within a unit cube $[0, 1]^{d_e}$, where d_e denotes the effective dimension. The unit cube can be structured as a grid composed of hypercubes, each with a side length of $s < 1$. In this configuration, the number of hypercubes, denoted as N_s, is given by $(s)^{-d_e}$. Sharma and Kaplan (2022) suggest that this quantity is proportional to the number of parameters, $\#S_s$, in a network that is necessary to achieve the grid structure.

So, if the side length of hypercubes in the grid becomes $2s < 1$, the number of hypercubes becomes $N_{2s} = (2s)^{-d_e}$, which is proportional to the number of parameters $\#S_{2s}$ of a network to realize the grid.

Following this reasoning, when reducing the side length from $2s$ to s, the number of parameters required for the networks is given by $\#S_s = c(s)^{-d_e} = 2^{d_e}\#S_{2s}$. This relationship illustrates that the number of parameters for a new network increases by a factor of 2^{d_e} compared to the original network when the side length of the original configuration is halved.

This exponential increase in parameter counts is attributed to the widening of the network while keeping the depth fixed. Hence, the parameters used to derive the grid with a side length of $2s$ are not applicable to derive those for the grid with a side length of s. We thus assert that the networks described in Sharma and Kaplan (2022) for achieving the grid configuration are shallow, which consistently increases the number of parameters by widening a layer.

Alternatively, it is possible to construct the exact grid configuration with side length s from two configurations with side length $2s$, by translating one configuration s units to obtain the other. If we can build a network for a grid of side length $2s$ and then incorporate this translation to derive the other, we can significantly reduce the number of parameters required for constructing the configuration with side length s.

Suppose we have a sub-network $\mathcal{N}^1_{2s}$ that consists of $\frac{1}{2s}$ parallel and equal distance hyperplanes with respect to the first component of the coordinate system of d_e components. From those hyperplanes, we can generate $\frac{1}{s}$ parallel and equal distance hyperplanes by adding a layer to $\mathcal{N}^1_{2s}$ as follows:

$$\mathcal{N}^1_s = \text{ReLU} \begin{bmatrix} \mathcal{N}^1_{2s} \\ \mathcal{N}^1_{2s} + \begin{bmatrix} -s \\ 0 \\ \vdots \\ 0 \end{bmatrix} \end{bmatrix} = \text{ReLU } \mathbf{A}_s \mathcal{N}^1_{2s}. \tag{34}$$

The top affine mapping in $\mathbf{A}_s$ is the identity matrix $\mathbf{I}$, which generates the same hyperplanes as those in $\mathcal{N}_{2s}^1$. The bottom mapping involves translating $-s$, resulting in another set of hyperplanes, each of which is s units smaller than the corresponding one in $\mathcal{N}_{2s}^1$. The ReLU layer maintains the output within the range $[0,1]^{d_e}$ in the first quadrant of $\mathbb{R}^{d_e}$.

By concatenating the outputs of $\mathcal{N}_s^i$ for $i = 1, \cdots, d_e$, we obtain the network that generates the grid configuration of side length s:

$$\mathcal{N}_s = \begin{bmatrix} \mathcal{N}_{2s}^1 \\ \vdots \\ \mathcal{N}_{2s}^{d_e} \end{bmatrix} = \mathrm{ReLU}\left(\mathrm{diag}(\mathbf{A}_s, \cdots, \mathbf{A}_s)\right)\mathcal{N}_{2s}, \tag{35}$$

where $\mathrm{diag}(\mathbf{A}_s, \cdots, \mathbf{A}_s)$ is the block diagonal matrix with each block $\mathbf{A}_s$. The domain of $\mathcal{N}_s$ is the grid of side length s. Figure 4 shows how the network $\mathcal{N}_s$ is constructed within the domain $[0,1]^2$, following Eqs. (34) and (35).

This construction shows that adding a layer increases the number of parameters linearly with respect to d_e. This relationship is described by the equation $\#W_s = \#W_{2s} + d_e c \leq (1+\alpha)\#W_{2s}$, where c denotes the number of parameters for $\mathbf{A}_s$, $\alpha > 0$, and $\#W_{is}$ denote the number of parameters for $\mathcal{N}_{is}$ for $i = 1$ and 2. This linear increase in parameter counts stands in sharp contrast to the exponential growth observed in the network of Sharma and Kaplan (2022).

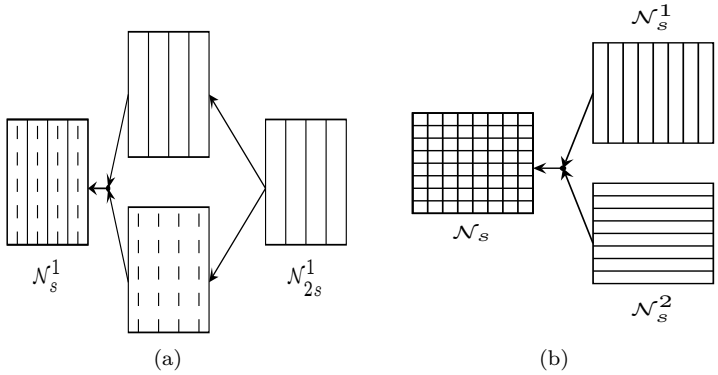

Figure 4: Schematic diagrams illustrate how the network $\mathcal{N}_s$ is formed by adding an additional layer to $\mathcal{N}_{2s}$, with its domain in $[0,1]^2$. Subfigure (a) depicts the configurations along the first coordinate, transitioning from $\mathcal{N}_{2s}^1$ to $\mathcal{N}_s^1$. Here, the distance between the parallel hyperplanes in $\mathcal{N}_{2s}^1$ is $2s$, while the distance for the hyperplanes in $\mathcal{N}_s^1$ is s. The middle top subfigure corresponds to $\mathcal{N}_{2s}^1$, whereas the middle bottom subfigure shows the configuration of $\mathcal{N}_{2s}^1 + [-s\ 0]^\top$. The configuration of $\mathcal{N}_s^1$ is obtained by concatenation those in $\mathcal{N}_{2s}^1$ and $\mathcal{N}_{2s}^1 + [-s\ 0]^\top$, as defined by Eq. (34). Subfigure (b) illustrates how the configurations of $\mathcal{N}_s^1$ and $\mathcal{N}_s^2$ are concatenated to obtain the configuration of $\mathcal{N}_s$, as defined by Eq. (35). The domain partition of $\mathcal{N}_s$ corresponds to the Cartesian product of the domain partitions of $\mathcal{N}_s^1$ and $\mathcal{N}_s^2$, as stated by Hwang and Tung (2023).

Next, we compare the mean absolute test errors. Assume training points form a γ_s-covering of the unit cube. Let $\mathbf{x}$ and $\mathbf{x}_i$ be two points within a hypercube of side length s, with $\|\mathbf{x} - \mathbf{x}_i\| \leq \gamma_s$. Hence, $\gamma_s \leq \sqrt{d_e}s$, where $\sqrt{d_e}s$ represents the maximum distance between two points inside a hypercube of side length s.

Define $f : \mathbb{R}^{d_e} \to \mathbb{R}$ as the target function that has a Lipschitz constant K_f. Hence, $\|\nabla f(\mathbf{x})\| \leq K_f$. To approximate network output $\mathcal{N}_s(\mathbf{x})$, we can use Taylor's expansion of $f(\mathbf{x})$ around $\mathbf{x}_i$:

$$\mathcal{N}_s(\mathbf{x}) \approx f(\mathbf{x}_i) + \nabla f(\mathbf{x}_i)^\top(\mathbf{x} - \mathbf{x}_i).$$

This approximation leads us to the bound:

$$|\mathcal{N}_s(\mathbf{x}) - f(\mathbf{x})| \leq |f(\mathbf{x}_i) - f(\mathbf{x})| + K_f\|\mathbf{x}_i - \mathbf{x}\| \leq 2K_f\gamma_s.$$

Consequently, if we let $2^l s = 1$, the test error can be bounded by:

$$\int_0^1 d^{d_e}\mathbf{x}|f(\mathbf{x}) - \mathcal{N}_s(\mathbf{x})| \leq 2K_f\gamma_s \leq 2K_f\sqrt{d}_e s = 2K_f\sqrt{d}_e(1/2^l). \tag{36}$$

By unfolding the recurrence equation $\#W_s \leq (1+\alpha)\#W_{2s}$ l times and given that $\#W_1 = 1$, we can determine the number of parameters for network $\mathcal{N}_s$ as follows:

$$\#W_s = (1+\alpha)^l \#W_1 = (1+\alpha)^l. \tag{37}$$

If we substitute $2^l = (1+\alpha)^{l/\log_2(1+\alpha)}$ into Eq. (37), we obtain: $2^l = \#W_s^{1/\log_2(1+\alpha)}$. Then, if we substitute this expression for 2^l into Eq. (36), the test error for the network $\mathcal{N}_s$ is bounded by:

$$\frac{2K_f\sqrt{d}_e}{\#W_s^{1/\log_2(1+\alpha)}}. \tag{38}$$

The absolute value of the slope in the log-log plot is $1/\log_2(1+\alpha)$.

In contrast, using the recurrence for the network of Sharma and Kaplan (2022): $\#S_s = 2^{d_e}\#S_{2s}$ and let $\#S_1 = 1$, we obtain $\#S_s = 2^{ld_e}\#S_1 = 2^{ld_e}$. Substituting $2^l = (\#S_s)^{1/d_e}$ into Eq. (36), the test error is bounded by

$$\frac{2K_f\sqrt{d}_e}{\#S_s^{1/d_e}}. \tag{39}$$

The slope's absolute value is $1/d_e$, as noted in Sharma and Kaplan (2022).

Additionally, we argue that $\#S_s \geq \#W_s$ because the wide network includes additional parameters without accounting for the values of the existing ones in previous depths. Consequently, based on Eqs. (38) and (39), we establish a comparison between box-module dimensions, resulting in the inequality:

$$\log_2(1+\alpha) \leq d_e.$$

C Proof of Proposition 1

Because the two networks $[I+\rho M_L]\circ[I+\rho M_{L-1}]\circ\cdots\circ[I+\rho M_1]$ and $[\rho M_L]\circ[\rho M_{L-1}]\circ\cdots\circ[\rho M_1]$ share the same input domain partition Hwang and Tung (2023), we can simplify our notation for convenience. If our focus is on the input domain partition, we define:

$$\mathcal{M}_L^0 = [\rho M_L]\circ[\rho M_{L-1}]\circ\cdots\circ[\rho M_1].$$

We will construct $\mathcal{M}_l^0 = [\rho M_l]\cdots[\rho M_1]$ layer by layer, ensuring that it satisfies the condition in Eq. (12) with a constant value of α maintained throughout the network's depth.

The function of $\mathcal{M}_L^0$ is piece-wise continuous. Pascanu et al. (2013) previously discussed the machine's capability for function approximations, focusing on the number of regions generated for function approximation. If P_l^0 is the finest input domain partition of $\mathcal{M}_l^0$, where each partitioning region of P_l^0 contains no subregions, then $|P_l^0|$ represents the size of P_l^0. When considering P_l^0, each partitioning region is a polytope, an affine linear mapping domain. Additionally, we note that P_{l+1}^0 is a refined partition of P_l^0, meaning that every region in P_{l+1}^0 is within a region in P_l^0. This refinement of partitioning regions in P_l^0 sets up a tree-like partitioning of P_l^0, where each region in P_{l+1}^0 has a unique parent region in P_l^0.

In the realm of P_L^0, every partitioning region forms a polytope, essentially the intersection of half-planes. The "diameter" of a region denotes the longest distance between two points within the region. The maximum diameter among all polytopes in a partition is equivalent to the diameter of the smallest enclosing ball of the partition. Calculating the diameter of a polytope according to Frieze and Teng (1994) can be intricate when dealing with polytopes of arbitrary dimension and facet number. Here, we are developing a tractable

algorithm to partition a polytope and reduce the diameters of the resulting sub-polytopes to a constant fraction of the original value.

The algorithm begins by dividing a polytope p in P_L^0 via a cutting hyperplane passing through an anchor point in p into two sub-polytopes. This method is then applied recursively to each sub-polytope. As a result, polytope p is the root of a tree-like structure, with intermediate and leaf nodes representing the sub-polytopes. The sub-polytopes at the leaf nodes that refine p will have a diameter smaller than half that of the root polytope. Consequently, the collection of sub-polytopes at the leaf nodes of all trees forms the partition P_{L+1}^0, with the diameter to be less than half that of P_L^0.

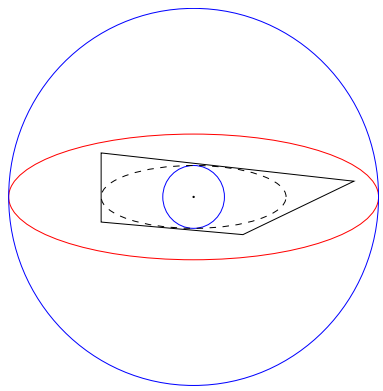

Figure 5: The dashed ellipsoid represents the maximum inscribed ellipsoid of polygon p. As per Eq. (41), the primary axis of the ellipsoid has a radius of η, while the minor axis has a radius of $\tilde{\eta}$, and their ratio is given by $\tilde{\eta}/\eta \geq \zeta > 0$. The center of the ellipsoid corresponds to the highlighted anchor point $\mathbf{a}^p$ of the polygon. The outer ellipsoid, shown in red and expanded by a factor of two at the boundary of the maximum inscribed ellipsoid, encloses the polygon. Enclosed within the polygon is a blue inner circle with a radius of $\tilde{\eta}$, and the polygon itself is enclosed by a blue outer circle with a radius of 2η; hence, $B_2(\mathbf{a}^p, \tilde{\eta}) \subseteq p \subseteq B_2(\mathbf{a}^p, 2\eta)$.

The anchor point is the center of the maximum inscribed ellipsoid within the polytope. By Boyd and Vandenberghe (2004), we can determine this point using convex optimization methods. Additionally, we can utilize properties that approximate the volume of a polytope from both inside and outside at the anchor point, as described in Grötschel et al. (2012) to bound the radius of the smallest enclosing ball of a polytope in P_{L+1}^0. Let $\mathbf{a}^p \in \mathbb{R}^d$ be the anchor point of the polytope p. Any hyperplane passing through $\mathbf{a}^p$ divides the polytope into two sub-polytopes, with q being one of them. The volume-reduction property shows a reduction in volume between p and q, as demonstrated by the inequality

$$\mathrm{vol}(q) \leq \left(1 - \frac{1}{d}\right) \mathrm{vol}(p). \tag{40}$$

Additionally, if polytope p is convex (or its convex hull if it's not), then the volume of p can be effectively approximated from the outside by an ellipsoid enlarging a constant factor of the inner maximum inscribed ball of p; specifically, the outer ellipsoid of p represents the boundary of the inner ellipsoid, expanded by a factor d around the anchor point.

Let $\eta > 0$ represent the radius of the principal axis of the inner maximum volume inscribed ellipsoid of p, and $\tilde{\eta} > 0$ denote the smallest radius of this ellipsoid. Assume the relationship $\tilde{\eta}/\eta \geq \zeta > 0$ holds, where ζ is a predetermined value. Then, we can express the approximation of polytope volume from inside and outside by balls in the following inequality:

$$\zeta^d \eta^d \,\mathrm{vol}\, B_d(\underline{0}, 1) \leq \tilde{\eta}^d \,\mathrm{vol}\, B_d(\underline{0}, 1) \leq \mathrm{vol}(p) \leq (d\eta)^d \,\mathrm{vol}\, B_d(\underline{0}, 1). \tag{41}$$

The first inequality follows the condition $\tilde{\eta}/\eta \geq \zeta$. The second inequality approximates the volume of p from the inside using a ball of radius $\tilde{\eta}$, applying the formula $\mathrm{vol} B_d(\underline{0}, r) = r^d \mathrm{vol} B_d(\underline{0}, 1)$. The final inequality

provides an approximation from the outside, where $d\eta$ represents the radius of an enclosing ball around p. Figure 5 offers a schematic illustration of Eq. (41).

Let's consider sub-polytopes formed by cutting $p \in P_L^0$ with a hyperplane passing the anchor point of p. These sub-polytopes are children of the root polytope p in a binary branch. This process can be repeated for each sub-polytope, creating a binary tree of depth k, with at most 2^k leaves, each leaf sub-polytope connected to a path rooted at p. We can label the sub-polytope in a path at depth one as q_1, the sub-polytope in the path at depth two as q_2, and so on, as shown in Figure 6, which provides a schematic description of a path of sub-polytopes in the tree and their hyperplane cuttings. To ensure a child's sub-polytope has a non-empty volume, we assume that all eigenvalues of the largest inscribed ellipsoids of any sub-polytope q_i are greater than zero, and the ratio of the smallest radius to the largest radius of a maximum inscribed ellipsoid is not less than ζ. Let's denote η_k and $\tilde{\eta}_k$ as the largest and smallest radii of the principal axes of the maximum volume inscribed ellipsoid of q_k. Using Eqs. (40) and (41), we obtain the inequality

$$\zeta^d \eta_k^d \text{vol } B_d(\underline{0}, 1) \leq \text{vol}(q_k) \leq (1 - \frac{1}{d})^k \text{vol } (p). \tag{42}$$

As we increase the depth k, the volume of q_k decreases to zero. This implies that the sequence $\{\eta_k\}$ decreases to zero as k increases, satisfying the inequality

$$\eta_k \leq \left[\frac{(1 - \frac{1}{d})^k \text{vol}(p)}{\zeta^d \text{vol } B_d(\underline{0}, 1)} \right]^{\frac{1}{d}}. \tag{43}$$

Thus, for each path from root polytope p to leaf sub-polytope q_k of the binary tree, we can find a sufficiently large k such that $(d\eta_k) \leq \frac{\gamma_p}{2}$, where γ_p is the radius of the smallest enclosing ball of p, and q_k is enclosed by a ball of radius $d\eta_k$. The number of hyperplane cuts required to create the path from p to q_k is equal to k, the depth of the path, with each hyperplane cut needing $d+1$ parameters. Based on the construction, a binary tree of depth $k-1$ has less than $2k$ nodes. To reduce the enclosing radius of p to half of the value for all paths in the tree, less than $2k(d+1)$ parameters are needed.

Note that each region in P_l^0 has a specific depth for its own binary tree. We can define $\omega - 1$ as the maximum depth for the binary tree corresponding to any polytope in P_l^0 for $l \leq L$. Using ω, the number of polytopes in P_{l+1}^0 is at most $|P_{l+1}^0| \leq 2^\omega |P_l^0| \leq 2^{l\omega}$ with $|P_0^0| = 1$. The number of parameters to generate P_{l+1}^0 from P_l^0 is bounded by $(d+1)2^{l\omega}$.

The number of parameters in $\mathcal{M}_{l+1}^0$, denoted as $\#W_{l+1}$, is equal to the number of parameters in $\mathcal{M}_l^0$, denoted as $\#W_l$, plus the number of parameters required to obtain P_{l+1}^0 from P_l^0. By using the induction hypothesis $\#W_l \leq C(d+1)2^{l\omega}$ for some $C \geq 1$, we can obtain the following inequality by setting $\alpha \geq 1$:

$$\#W_{l+1} \leq \#W_l + (d+1)2^{l\omega} \leq \#W_l + C(d+1)2^{l\omega} \leq 2\#W_l \leq (1+\alpha)\#W_l.$$

To complete the construction, we need to translate all cutting hyperplanes of the trees into a new layer. By induction, suppose we obtain $\mathcal{M}_l^0$. We denote the restriction of $\mathcal{M}_l^0$ on the polytope $p \in \mathcal{P}_l^0$ as $\mathcal{M}_l^0|p$. Let $q \in \mathcal{P}_{l+1}^0$ represent a sub-polytope of p. We denote $\mathcal{M}_{l+1}^0|q = \rho \mathbf{A}_{q,p} \mathcal{M}_l^0|p$ the equations define polytope q, where $\rho \mathbf{A}_{q,p}$ is the affine mapping derived from the sequence of cutting hyperplanes used to construct q from p. In the derivation, we can replace ρ by a diagonal matrix containing entries 0 or 1. The values in this matrix are determined by the values of $\mathcal{M}_l^0|p$. The overall affine linear mapping M_{l+1} is then constructed by stacking the arrays of $\mathbf{A}_{q,p}$ generated from each polytope p in $\mathcal{P}_l^0$.

D

Proof of Lemma 3

If the center of a ball b with radius γ is located outside the ball $B_d(\mathbf{x}, \gamma)$, then the point $\mathbf{x}$ cannot be contained within the ball b. We can conclude that the probability of a randomly chosen ball not containing $\mathbf{x}$ is given by $1 - P(\mathbf{x}; B_d(\mathbf{x}, \gamma))$, where $P(\mathbf{x}; B_d(\mathbf{x}, \gamma))$ represents the probability of $\mathbf{x}$ being located inside the ball $B_d(\mathbf{x}, \gamma)$.

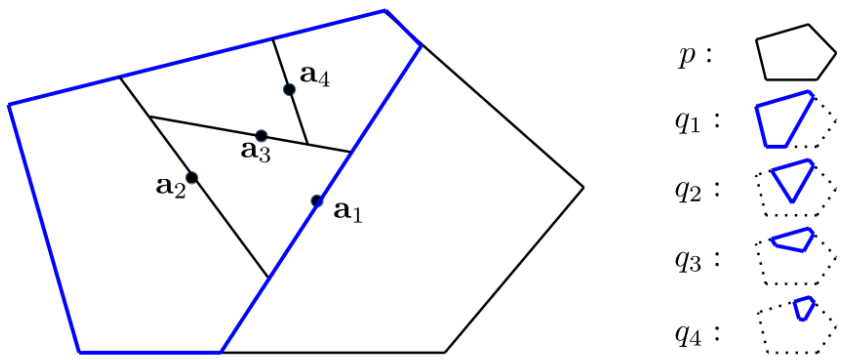

Figure 6: The convex polygon p can be partitioned into sub-polytopes using sequences of hyperplane cuttings, leading to a binary tree structure with p at the root. The root p is divided into two sub-polygons by a hyperplane passing through the anchor point $\mathbf{a}_1$, the center of the maximal inscribed ball of p. The sub-polygon q_1, colored in blue, is further divided using anchor points $\mathbf{a}_2$, $\mathbf{a}_3$, and $\mathbf{a}_4$, corresponding to the centers of the maximal inscribed balls for sub-polygons q_2, q_3, and q_4, respectively. It is worth noting that the volume of any sub-polygon retains a constant fraction of its parent polygon, as defined by Eq. (40).

The probability of $\mathbf{x}$ being covered by the l-th random ball but not by the first $l-1$ balls by process $X(\gamma)$ is given by $P(\mathbf{x}; B_d(\mathbf{x}, \gamma))(1 - P(\mathbf{x}; B_d(\mathbf{x}, \gamma)))^{l-1}$. Hence, the expected number of random balls required to cover $\mathbf{x}$ for the first time is

$$\begin{aligned}
\bar{l}(\mathbf{x}) &= \sum_{l=1}^{\infty} l P(\mathbf{x}; B_d(\mathbf{x}, \gamma))(1 - P(\mathbf{x}; B_d(\mathbf{x}, \gamma)))^{l-1} \\
&= P(\mathbf{x}; B_d(\mathbf{x}, \gamma)) \frac{\mathrm{d} \sum_{l=1}^{\infty} [1 - P(\mathbf{x}; B_d(\mathbf{x}, \gamma))]^l}{\mathrm{d}[1 - P(\mathbf{x}; B_d(\mathbf{x}, \gamma))]} \\
&= \frac{1}{P(\mathbf{x}; B_d(\mathbf{x}, \gamma))} \leq \frac{1}{c\gamma^d}.
\end{aligned} \tag{44}$$

In the above calculation, we utilized the formula $\sum_{i=0}^{\infty} x^i = \frac{1}{1-x}$ for $x \in (0, 1)$, and $P(\mathbf{x}, B_d(\mathbf{x}, \gamma)) \geq c\gamma^d$ for all $\mathbf{x} \in B_d(\underline{0}, 1)$, which is a consequence of $\mathcal{P} \geq k > 0$ over $B_d(\underline{0}, 1)$.

The mean of the first time to cover the unit d-ball by random balls is the integral of the product of $\bar{l}(\mathbf{x})$ and $\mathcal{P}(\mathbf{x})$ over the unit d-ball, i.e.,

$$\mathcal{E}(|X_{min}|) = \int \bar{l}(\mathbf{x}) \mathcal{P}(\mathbf{x}) d\mathbf{x} \leq \frac{1}{c\gamma^d}. \tag{45}$$

We now estimate the average number of samples drawn independently according to $\mathcal{P}$ and the process $X(\gamma)$ needed to cover the unit d-ball for the first time. Leveraging the Markov inequality, we can derive an upper bound for the probability that the initial N samples generated by $X(\gamma)$ cannot cover the unit d-ball as follows:

$$P(|X_{min}| > N) \leq \frac{\mathcal{E}(|X_{min}|)}{N} \leq \frac{1}{cN\gamma^d}. \tag{46}$$

Hence, the probability of the initial N samples forming a covering is $P(|X_{min}| \leq N) \geq 1 - \frac{1}{cN\gamma^d}$.

E

Proof of Lemma 4

The histogram mass in region $p \in R(S)$ is the average observation number in the sequence within the region, given by

$$Z_p^m(S) = \frac{1}{N} \sum_{i=1}^{N} \mathcal{I}(X_i \in p) = \frac{n_p}{N}, \tag{47}$$

where $\mathcal{I}$ is the indicator function and n_p is the number of training points in bin p. Approximating with Hoeffding's inequality, we obtain Eq. (20), which is

$$P(|\int_p \mathcal{I}(\mathbf{x} \in p)\mathcal{P}(\mathbf{x})d\mathbf{x} - \frac{n_p}{N}| \geq t_p) \leq 2\exp(-2Nt_p^2). \tag{48}$$

For any $t_p > 0$, as $N \to \infty$, the expression $1 - 2\exp(-2Nt_p^2)$ approaches 1. Consequently, for any sequence $t_p \to 0$ slower than $N \to 1$, $1 - 2\exp(-2Nt_p^2)$ will also approach 1. Hence, $\int_p \mathcal{I}(\mathbf{x} \in p)\mathcal{P}(\mathbf{x})\, d\mathbf{x} = \frac{n_p}{N}$ in the limit as $N \to \infty$.