On the Error Bounds for ReLU Neural Networks

Ronald Katende, Henry Kasumba, Godwin Kakuba and John Mango

***Abstract*—This paper addresses the challenge of establishing rigorous error bounds for zero-trace Rectified Linear Unit (ReLU) Neural Networks (NNs). We derive theoretical results to provide insights into the accuracy of these networks in approximating continuous functions, focusing on the influence of network architecture, such as the number of layers and neurons. Emphasizing zero-trace ReLU NNs due to their relevance in various physical and engineering applications, we aim to find a bound ε such that $|f(x) - \hat{f}(x)| \leq \varepsilon$, where $\hat{f}(x)$ is the network's output. Our analysis leverages universal approximation theorems, Rademacher complexity, and probabilistic methods to develop novel error bounds. We also explore the impact of data distribution on these bounds, contributing to the ongoing effort to bridge the gap between theoretical guarantees and practical applicability. Numerical experiments validate our theoretical findings, showcasing the trade-off between network complexity and computational resources. Additionally, we explore the performance of ReLU NNs in solving different types of Partial Differential Equations (PDEs), highlighting the impact of network size and iterations on error reduction**

***Index Terms*—Numerical Analysis, Finite Element Method, Error Analysis, Neural Networks, Rectified Linear Unit.**

I. Introduction

This paper addresses the challenge of establishing error bounds for zero-trace Rectified Linear Unit (ReLU) Neural Networks (NNs), providing rigorous theoretical guarantees on their accuracy. A zero-trace function is zero at its domain boundary. We derive theoretical results that highlight how the accuracy of zero-trace ReLU NNs in approximating continuous functions depends on network architecture, such as the number of layers and neurons. These NNs are particularly relevant in various physical and engineering applications, so throughout this paper, ReLU NNs refer specifically to zero-trace ReLU NNs.

Establishing error bounds for ReLU NNs is crucial due to their widespread use across different domains. Understanding the factors that influence their accuracy helps in designing networks, selecting training strategies, and choosing models. Precise error bounds enhance the reliability of these networks, especially in safety-critical applications where dependability is essential.

Manuscript received April 30, 2024 revised September 26, 2024.

This work has been supported by the Mathematics for Sustainable Development (MATH4SDG) project, a research and development project running in the period 2021-2026 at Makerere University-Uganda, University of Dar es Salaam-Tanzania, and the University of Bergen-Norway.

Ronald Katende is a PhD candidate in the Department of Mathematics, Department of Mathematics, College of Natural Sciences, Makerere University, Kampala, Uganda. (email: rkatende@kab.ac.ug).

Henry Kasumba is a lecturer of mathematics at the Department of Mathematics, College of Natural Sciences, Makerere University, Kampala, Uganda. (email: henry.kasumba@mak.ac.ug).

Godwin Kakuba is an associate professor at the Department of Mathematics, College of Natural Sciences, Makerere University, Kampala, Uganda. (email: godwin.kakuba@mak.ac.ug).

John Mango is an associate professor at the Department of Mathematics, College of Natural Sciences, Makerere University, Kampala, Uganda. (email: mango.john@mak.ac.ug).

Previous research has proposed error bounds for various neural networks using different mathematical approaches. For instance, [7] discusses generalization errors in large compressible deep neural networks, introducing a data-dependent capacity control technique that more precisely assesses generalization performance compared to traditional methods. In [8], a novel method estimates errors in neural network solutions for elasticity problems, showing the robustness of energy-based error measures. The study in [10] bridges empirical risk minimization and Bayesian deep learning, offering faster convergence rates and a kernel-based perspective for understanding deep learning model generalization. The work in [11] provides non-asymptotic L_2 error bounds for neural network regression, vital for finite-sample scenarios. In [12], foundational results on the approximation capabilities of neural networks are discussed, establishing essential bounds for understanding trade-offs between network complexity and accuracy. [13] explores designing feedforward networks optimized for specific error bounds, ensuring high assurance in network performance. Finally, [14] addresses robust error bounds for quantized and pruned networks, ensuring performance even under compression constraints.

This work builds on these previous efforts, particularly drawing from universal approximation theorems (UATs) for ReLU NNs. While UATs show that networks can approximate any continuous function, they provide little guidance on architecture and training. By analyzing error bounds, we gain insights into approximation capabilities and network complexity, potentially leading to more efficient designs.

Recent advancements, such as Rademacher complexity analysis and probabilistic methods, have been explored, but challenges remain in applying theoretical guarantees to practical scenarios and understanding data distribution's impact on error bounds. Despite progress, existing bounds often rely on strong assumptions that may not hold in realistic cases [1]. Bridging the gap between theory and practice is ongoing, and understanding how data distribution, size, and quality affect error bounds is crucial for robust generalization [2]. This paper contributes to these efforts by deriving novel error bounds for a specific class of ReLU NNs, demonstrating improvements over existing bounds and exploring the relationship between network architecture and error.

II. Preliminary Notes

Consider the partial differential equation (PDE)

$$\mathcal{L}u = f, \tag{1}$$

where f is known data, $\mathcal{L}$ is the differential operator, and u is the unknown solution. We seek to bound the error in approximating $u(x)$ using a ReLU NN. Specifically, we aim to find a bound ε such that $|f(x) - \hat{f}(x)| \leq \varepsilon$, where

$\hat{f}(x)$ is the ReLU NN output for input x. We explore how the smoothness of the target function, network depth, and training data size affect error bounds and develop algorithmic tools to apply these bounds in network design and training optimization. Universal approximation theorems show that ReLU NNs can approximate any continuous function with arbitrary accuracy under certain conditions [5]. Mathematically, for any continuous function $f : \mathbb{R}^n \to \mathbb{R}$ and any $\epsilon > 0$, there exists a ReLU NN F with parameters θ such that for any compact set $K \subset \mathbb{R}^n$, there exists θ^* such that

$$|f(x) - F(x;\theta^*)| < \epsilon, \quad \forall x \in K.$$

However, these theorems often do not provide practical guidance on determining the network architecture and training process needed to achieve a specific accuracy level. Rademacher complexity analysis offers bounds on the generalization error of ReLU NNs by leveraging the notion of Rademacher complexity. Mathematically, Rademacher complexity $\mathcal{R}(F)$ is defined as the expectation over random signs σ_i of the supremum of the empirical risk,

$$\mathcal{R}(F) = \mathbb{E}_\sigma \left[\sup_\theta \frac{1}{m} \sum_{i=1}^{m} \sigma_i F(x_i;\theta) \right],$$

where F is the hypothesis class represented by the ReLU NN, m is the number of samples, and x_i are input data points. The generalization error bound is then given by

$$\mathbb{E}_{\text{data}}[L(F)] \leq \hat{L}(F) + \mathcal{R}(F) + \sqrt{\frac{C}{m}},$$

where $\hat{L}(F)$ is the empirical risk, $L(F)$ is the true risk, and C is a constant. Probabilistic methods incorporate uncertainty into modeling, allowing the generalization error to be treated as a random variable, leading to more robust bounds compared to deterministic methods. Bayesian neural networks, for example, treat weights as random variables and use Bayesian techniques for inference [4].

We first consider theoretical results for a single-layer ReLU NN, followed by numerical examples that validate these results. We then discuss the practical implications of the derived error bounds, focusing on how they can guide network design and training decisions, balancing accuracy and computational efficiency.

III. Main Results

In this section, we derive some theoretical results about the maximum value as well as error bounds for a ReLU NN. We utilise the Poincaré inequality to derive both results. We show that both the maximum value as well as the error bounds are highly dependent on the number of layers and the number of neurons on the network.

Suppose that the target function $f \in W^{k,p}$, a Sobolev space, which provides the necessary smoothness conditions for deriving error bounds. Also, let $F(x;\theta)$ be the output of a ReLU NN with parameters θ for input x. The error bound ε is defined as the maximum deviation of the network output from the target function, i.e.,

$$\varepsilon = \sup_{x \in \Omega} |f(x) - F(x;\theta)|,$$

where Ω is the domain of interest. From a quasi-mode, it would appear that ε satisfies the following theorem, which considers the approximation capabilities of the ReLU NN, taking into account factors such as the depth L of the network, the number of neurons per layer N, and the smoothness of the target function.

Theorem III.1. *Let f be a function in the Sobolev space $W^{k,p}(\Omega)$ and $F(x;\theta)$ be the output of a ReLU neural network with depth L and width N. Under certain conditions on the network architecture and the smoothness of f, there exists a constant C such that*

$$\varepsilon \leq C \left(\frac{1}{N^\alpha} + \frac{1}{L^\beta} \right),$$

where α and β are positive constants that depend on the smoothness of f and the architecture of the network

This theorem highlights the relationship between the network's architecture and the resulting error bound, providing insight into how increasing the depth and width of the network can reduce the error. The constants α and β reflect the trade-offs between network complexity and approximation accuracy, offering practical guidance for network design. However, a more intricate investigation into the network architecture and representation yields the following related result.

A. Error Bound for the ReLU NN Solution

We derive an error bound between the exact solution $u(x)$ and a ReLU NN solution to equation (1), denoted by $u_{NN} = u_{NN}(x)$, utilizing the triangle inequality and the representation of ReLU NNs as Finite Elements (FE). The error bound for a one-dimensional finite element solution to equation (1) is given by

$$\|u(x) - u_{FEM}\| \leq Ch, \tag{2}$$

for some constant $C = C(\Omega)$ that depends on the domain Ω and h being the width of subdomains. We will also use a fundamental theorem, the Poincaré inequality, as stated in Lemma 1.

Lemma 1 (Poincaré Inequality)**.** *[9]*
Let p be such that $1 \leq p \leq \infty$ and Ω a subset bounded in at least one direction. Then there exists a constant C depending only on Ω and p such that for every function v in the Sobolev space $H^1(\Omega)$ of zero trace functions (functions that are zero on the boundary), we have

$$\|v\|_{L^p(\Omega)} \leq C \|\nabla v\|_{L^p(\Omega)}.$$

For the exact solution $u(x)$, we have

$$\|u(x) - u_{NN}\|_1 = \|u(x) - u_{FEM} + u_{FEM} - u_{NN}\|_1,$$

which by the triangle inequality can be re-written as

$$\|u(x) - u_{NN}\|_1 \leq \|u(x) - u_{FEM}\|_1 + \|u_{FEM} - u_{NN}\|_1,$$

and thus,

$$\|u(x) - u_{NN}\|_1 < Ch + \|u_{FEM} - u_{NN}\|_1. \tag{3}$$

We now need to obtain the bound for $\|u_{FEM} - u_{NN}\|_1$. This is established through Proposition III.1.

Proposition III.1. *The error in an output obtained using a single layer ReLU NN with k neurons is bounded by $\frac{6D}{k^2}$ for D a constant that depends on k.*

For the proof of Proposition III.1, we rely on Lemma 1, which relates the norm of a function to the norm of its gradient. The FEM and ReLU NN solutions to (1), i.e., u_{FEM} and u_{NN}, satisfy the Poincaré inequality (1) such that

$$\|u_{FEM} - u_{NN}\|_1 \leq \|u_{FEM}\|_1 + \|u_{NN}\|_1. \tag{4}$$

By Lemma 1,

$$\|u_{FEM}\|_1 \leq C_1 \|\nabla u_{FEM}\|_1 \text{ and } \|u_{NN}\|_1 \leq C_2 \|\nabla u_{NN}\|_1,$$

for $C_1(\Omega), C_2(\Omega) \in \mathbb{R}$. Therefore, by extension,

$$\begin{aligned}\|u_{FEM} - u_{NN}\|_1 &\leq K\|\nabla(u_{FEM} - u_{NN})\|_1 \\ &\leq K(\|\nabla u_{FEM}\|_1 + \|\nabla u_{NN}\|_1),\end{aligned}$$

for some $K(\Omega) \in \mathbb{R}$. For the rest of the proof, we write $\|.\|_1$ for the L^1 norm, i.e., $\|.\|_{L^1(\Omega)}$. Denote the error between the FEM and the ReLU NN solution as e defined by

$$e = u_{FEM}(x) - u_{NN}(x).$$

The gradient of e, denoted as ∇e with respect to x, is then

$$\begin{aligned}\nabla e(x) = \frac{de}{dx} &= \frac{du_{FEM}}{dx} - \frac{du_{NN}}{dx} \\ &= \sum_{r=1}^{n} \alpha_r \frac{d\phi_r}{dx} - \sum_{i=1}^{k} w_i^2 \frac{d\,\text{ReLU}(w_i^1 x + b_i)}{dx}.\end{aligned}$$

Thus,

$$\begin{aligned}&\|\nabla e(x)\|_1 \\ &\quad = \left\| \sum_{r=1}^{n} \alpha_r \frac{d\phi_r}{dx} - \sum_{i=1}^{k} w_i^2 \frac{d\,\text{ReLU}(w_i^1 x + b_i)}{dx} \right\|_1,\end{aligned}$$

and

$$\begin{aligned}&\|\nabla e(x)\|_1 \\ &\leq \left\| \sum_{r=1}^{n} \alpha_r \frac{d\phi_r}{d\,x} \right\|_1 + \left\| \sum_{i=1}^{k} w_i^2 \frac{d\,\text{ReLU}(w_i^1 x + b_i)}{dx} \right\|_1. \end{aligned} \tag{5}$$

Now, recall that in a general sense,

$$\phi(x) = \begin{cases} 1 - \frac{x}{h} & \text{if } 0 \leq x \leq h, \\ 1 + \frac{x}{h} & \text{if } -h \leq x \leq 0, \\ 0 & \text{otherwise.} \end{cases}$$

This implies that

$$\frac{d\,\phi(x)}{dx} = \begin{cases} \frac{-1}{h} & \text{if } 0 \leq x \leq h, \\ \frac{1}{h} & \text{if } -h \leq x \leq 0, \\ 0 & \text{otherwise.} \end{cases}$$

Therefore,

$$\left\| \frac{d\phi_r}{dx} \right\|_1 \leq A_1, \tag{6}$$

for some constant A_1 that depends on $h = x_{r+1} - x_r = x_r - x_{r-1}$. Thus,

$$\frac{d\,[\text{ReLU}(w_i^1 x + b_i)]}{dx} = \begin{cases} w_i^1, & \text{if } w_i^1 x + b_i \geq 0, \\ 0, & \text{if } w_i^1 x + b_i < 0. \end{cases}$$

It then follows that

$$\left\| \frac{d\,\text{ReLU}(w_i^1 x + b_i)}{dx} \right\|_1 \leq A_2, \tag{7}$$

for some constant A_2 that depends on the network parameters w_i^1. Substituting the bounds (6) and (7) into inequality (5) yields

$$\|\nabla e(x)\|_1 \leq \left\| A_1 \sum_{r=1}^{n} \alpha_r \right\|_1 + \left\| A_2 \sum_{i=1}^{k} w_i^2 \right\|_1.$$

Therefore,

$$\|\nabla e(x)\|_1 \leq A_1 \sum_{r=1}^{n} \|\alpha_r\|_1 + A_2 \sum_{i=1}^{k} \|w_i^2\|_1. \tag{8}$$

Denote $C(\Omega) = A_1 \sum_{r=1}^{n} \|\alpha_r\|_1 + A_2 \sum_{i=1}^{k} \|w_i^2\|_1$, thus

$$\|u(x) - u_{NN}\|_1 \leq Ch + \frac{6D}{k^2}, \tag{9}$$

and hence,

$$\|u(x) - u_{NN}\|_1 \leq \frac{6D}{k^2}, \tag{10}$$

as $h \to 0$, for some constant D dependent on k. Equation (10) demonstrates the bound on the ultimate error between the exact solution and the ReLU NN solution.

Now, this result is extended, with the help of approximation theorems to strengthen the understanding of the performance of ReLU NNs. To further quantify the error between the exact solution $u(x)$ and the ReLU NN solution $u_{NN}(x)$, we analyze the generalization error. This error measures the difference between the expected performance of the model on new, unseen data and its performance on the training data.

In the following results, we explore various aspects of the error bounds and approximation properties of ReLU neural networks (NNs) and Physics-Informed Neural Networks (PINNs) for solving partial differential equations (PDEs). Each result builds upon the previous ones, enhancing our understanding of the convergence and generalization behavior of these networks. The progression from one theorem to the next allows us to create a cohesive narrative that connects generalization error, approximation capabilities, layer width variations, and the impact of learning rates and initialization on neural network performance.

Theorem III.2 (Generalization, Approximation, and Error Bounds for ReLU Neural Networks)**.** *Let $u(x)$ be the exact solution of a PDE in a bounded domain $\Omega \subset \mathbb{R}^d$, and let $u_{NN}(x)$ be the solution obtained using a ReLU neural network with L layers and N neurons per layer. Assuming the network is trained with m samples and a total loss function $\mathcal{L}$, then the generalization error $\mathcal{E}_{gen}$ is bounded by*

$$\mathcal{E}_{gen} \leq \mathcal{O}\left(\sqrt{\frac{L \log(N) + \log(1/\delta)}{m}} \right),$$

with probability at least $1-\delta$. Moreover, if $u(x) \in H^s(\Omega)$ with $s > \frac{d}{2}$, there exists a ReLU neural network such that the approximation error satisfies

$$\|u(x) - u_{NN}(x)\|_{H^s(\Omega)} \leq CN^{-2s/d},$$

where C depends on $u(x)$ and Ω. Also, for a network with L layers, where the l-th layer has N_l neurons. The error $\|u(x) - u_{NN}(x)\|_{L^2(\Omega)}$ is bounded by

$$\|u(x) - u_{NN}(x)\|_{L^2(\Omega)} \leq C \sum_{l=1}^{L} N_l^{-1/d},$$

where C depends on $u(x)$ and Ω. Finally, if $u_{PINN}(x)$ is the solution obtained using a ReLU PINN trained to an error ϵ_{PINN}. The error $\|u(x) - u_{PINN}(x)\|_{L^2(\Omega)}$ is bounded by

$$\|u(x) - u_{PINN}(x)\|_{L^2(\Omega)} \leq C\left(N^{-\frac{2s}{d}} + \epsilon_{PINN}\right),$$

where $s > \frac{d}{2}$ is the Sobolev regularity of u, and C depends on $u(x)$ and Ω.

Theorem III.3 (McDiarmid's inequality)**.** *Let $X_1, X_2, \ldots, X_m$ be independent random variables taking values in some set $\mathcal{X}$, and let $f : \mathcal{X}^m \to \mathbb{R}$ be a function such that for all i, changing the i-th coordinate X_i alters the value of f by at most c_i:*

$$\sup_{x_1,\ldots,x_m,x_i'} |f(x_1,\ldots,x_m) - f(x_1,\ldots,x_m)| \leq c_i.$$

Then, McDiarmid's inequality states that for any $\epsilon > 0$,

$$\Pr\left(f(X_1,\ldots,X_m) - \mathbb{E}[f(X_1,\ldots,X_m)] \geq \epsilon\right) \leq \exp\left(-\frac{2\epsilon^2}{\sum_{i=1}^m c_i^2}\right).$$

This inequality is useful for bounding the deviation of a function of independent random variables from its expected value. Now we provide a proof for theorem III.2 using the McDiarmid's inequality as shown in theorem III.3

Proof:

Generalization Error Bound

The generalization error $\mathcal{E}_{gen}(h)$ is defined as the difference between the expected risk $R(h)$ and the empirical risk $\hat{R}_S(h)$ for a hypothesis h in a hypothesis class $\mathcal{H}$

$$\mathcal{E}_{gen}(h) = R(h) - \hat{R}_S(h).$$

Given that the empirical risk $\hat{R}_S(h)$ is a function of the training set $S = \{(X_1, Y_1), \ldots, (X_m, Y_m)\}$, and assuming (X_i, Y_i) are i.i.d. samples, $\hat{R}_S(h)$ can be expressed as

$$\hat{R}_S(h) = \frac{1}{m}\sum_{i=1}^{m} \ell(h(X_i), Y_i),$$

where ℓ is the loss function. Since $\hat{R}_S(h)$ is an average of i.i.d. random variables, we can apply McDiarmid's inequality. The function f in this context is the empirical risk $\hat{R}_S(h)$, and changing one sample (X_i, Y_i) affects the empirical risk by at most $\frac{1}{m}$ times the range of the loss function. If the loss function is bounded by some constant L, then

$$c_i = \frac{L}{m}, \quad \forall i.$$

Therefore, McDiarmid's inequality gives us

$$\Pr\left(\hat{R}_S(h) - \mathbb{E}[\hat{R}_S(h)] \geq \epsilon\right) \leq \exp\left(-\frac{2m^2\epsilon^2}{mL^2}\right) = \exp\left(-\frac{2m\epsilon^2}{L^2}\right).$$

Taking the logarithm and rearranging, we obtain

$$\mathcal{E}_{gen}(h) = R(h) - \hat{R}_S(h) \leq \mathcal{O}\left(\sqrt{\frac{\log(1/\delta)}{m}}\right),$$

with probability at least $1-\delta$, for ϵ chosen appropriately. Next, we incorporate the Rademacher complexity $\mathcal{R}_m(\mathcal{H})$ to get a tighter bound. The Rademacher complexity measures the capacity of the hypothesis class $\mathcal{H}$ and can be bounded as:

$$\mathcal{R}_m(\mathcal{H}) \leq \mathcal{O}\left(\sqrt{\frac{L\log(N)}{m}}\right).$$

Thus, the generalization error bound becomes

$$\mathcal{E}_{gen} \leq \mathcal{O}\left(\mathcal{R}_m(\mathcal{H}) + \sqrt{\frac{\log(1/\delta)}{m}}\right).$$

Substituting the Rademacher complexity bound, we obtain

$$\mathcal{E}_{gen} \leq \mathcal{O}\left(\sqrt{\frac{L\log(N) + \log(1/\delta)}{m}}\right).$$

This bound combines the contributions of the hypothesis class complexity and the concentration inequality, providing a rigorous and tight bound on the generalization error for ReLU NNs.

Approximation Error in Sobolev Norm

For $u(x) \in H^s(\Omega)$ with $s > \frac{d}{2}$, approximation theory guarantees the existence of a ReLU network such that

$$\|u - u_{NN}\|_{H^s(\Omega)} \leq CN^{-2s/d},$$

where C depends on the regularity of u and the domain Ω.

Error Bound for Networks with Varying Widths

For a ReLU network with varying layer widths N_l, the error for each layer is

$$\|u - u_{NN_l}\|_{L^2(\Omega)} \leq CN_l^{-1/d}.$$

Summing over all layers

$$\|u(x) - u_{NN}(x)\|_{L^2(\Omega)} \leq C\sum_{l=1}^{L} N_l^{-1/d}.$$

Error Bound for High-Dimensional PINNs

Combining the approximation error and the training error ϵ_{PINN}, the total error is

$$\|u(x) - u_{PINN}(x)\|_{L^2(\Omega)} \leq \|u(x) - u_{NN}(x)\|_{L^2(\Omega)} + \|u_{NN}(x) - u_{PINN}(x)\|_{L^2(\Omega)}.$$

Substituting the bounds

$$\|u(x) - u_{NN}(x)\|_{L^2(\Omega)} \leq CN^{-\frac{2s}{d}},$$
$$\|u_{NN}(x) - u_{PINN}(x)\|_{L^2(\Omega)} \leq \epsilon_{PINN}.$$

Thus

$$\|u(x) - u_{PINN}(x)\|_{L^2(\Omega)} \leq C\left(N^{-\frac{2s}{d}} + \epsilon_{PINN}\right).$$

■

Theorem III.4 (Error Bounds and Approximation Properties of ReLU Neural Networks)**.** *Let $u(x)$ be the exact solution of a PDE in $\Omega \subset \mathbb{R}^d$, and $u_{NN}(x)$ be the solution obtained by a ReLU neural network (NN) with L layers and N neurons per layer, trained using an adaptive learning rate. The error $\|u(x) - u_{NN}(x)\|_{L^2(\Omega)}$ is bounded by:*

$$\|u(x) - u_{NN}(x)\|_{L^2(\Omega)} \leq C\left(N^{-1/d} + \epsilon_{adaptive}\right),$$

where $\epsilon_{adaptive}$ is the training error associated with the adaptive learning rate, and C is a constant depending on $u(x)$ and Ω.

Proof: The approximation error for a ReLU NN with N neurons per layer and L layers is given by:

$$\|u(x) - u_{NN}(x)\|_{L^2(\Omega)} \leq C_1 N^{-1/d},$$

where C_1 is a constant dependent on the regularity of $u(x)$ and the domain Ω. This result follows from known results in approximation theory, particularly in high-dimensional settings. Let $\epsilon_{adaptive}$ denote the training error when using an adaptive learning rate. Empirical and theoretical evidence shows that $\epsilon_{adaptive}$ is generally smaller than the training error with a fixed learning rate, leading to a faster convergence:

$$\|u_{NN}(x) - \hat{u}_{NN}(x)\|_{L^2(\Omega)} \leq C_2 \epsilon_{adaptive},$$

where $\hat{u}_{NN}(x)$ represents the partially trained NN solution, and C_2 is another constant depending on the problem setup. The total error is a combination of the approximation error and the training error:

$$\|u(x) - u_{NN}(x)\|_{L^2(\Omega)} \leq C_1 N^{-1/d} + C_2 \epsilon_{adaptive}.$$

By defining $C = \max(C_1, C_2)$, we obtain:

$$\|u(x) - u_{NN}(x)\|_{L^2(\Omega)} \leq C\left(N^{-1/d} + \epsilon_{adaptive}\right).$$

■

Now, in the context of ReLU neural networks (ReLU NNs), the pursuit of tighter and more comprehensive error bounds is critical. Existing literature often addresses the approximation capabilities of ReLU NNs with fixed architectures and standard training methods, but the effects of sparse neuron activation, adversarial perturbations, and time-varying learning rates remain largely unexplored. Previous theorems typically bound the error based on network width, depth, and fixed learning rates without considering these dynamic and adversarial aspects, leading to incomplete insights into the model's true performance. The following theorem introduces on a unified error bound for ReLU NNs, integrating the effects of sparse activation, adversarial perturbations, and time-varying learning rates, for a more realistic and practical error bound.

Theorem III.5 (Unified Error Bounds for ReLU NNs with Sparse Activation, Adversarial Perturbations, and Time-Varying Learning Rate)**.** *Let $u(x)$ be the exact solution in $\Omega \subset \mathbb{R}^d$, and $u_{NN}(x)$ the approximation via a ReLU NN with L layers and N neurons per layer. Suppose ρ is the fraction of active neurons per layer, δx is an adversarial perturbation, and $\eta(t)$ is a time-varying learning rate. The error $\|u(x) - u_{NN}(x)\|_{L^2(\Omega)}$ is bounded by*

$$\|u(x) - u_{NN}(x)\|_{L^2(\Omega)} \leq C\left(N^{-\frac{1}{d}}\rho^{-\alpha} + \epsilon_{adv} + \int_0^T \eta(t)\,dt\right),$$

where C depends on $u(x)$, Ω, and network architecture; $\alpha > 0$ reflects the impact of sparse activation; ϵ_{adv} is the adversarial error; and $\int_0^T \eta(t)\,dt$ is the cumulative learning rate effect.

Proof: We first analyze the impact of sparse activation. Given that only ρN neurons are active, the effective capacity of the network decreases, which influences the approximation error. The error bound due to sparse activation is

$$\|u(x) - u_{NN}(x)\|_{L^2(\Omega)} \leq C_1 N^{-\frac{1}{d}}\rho^{-\alpha},$$

where $\alpha > 0$ quantifies the reduction in capacity. Next, we consider the effect of adversarial perturbations. The perturbation δx leads to an additional error term ϵ_{adv}, typically bounded as

$$\epsilon_{adv} = \|u(x + \delta x) - u_{NN}(x + \delta x)\|_{L^2(\Omega)}.$$

This term captures the worst-case scenario under adversarial attack. Finally, the effect of a time-varying learning rate $\eta(t)$ over a training period $[0, T]$ is represented by the integral

$$\int_0^T \eta(t)\,dt,$$

which quantifies the total influence of the dynamic learning rate on model convergence. Combining these components, the total error is bounded by

$$\|u(x) - u_{NN}(x)\|_{L^2(\Omega)} \leq C_1 N^{-\frac{1}{d}}\rho^{-\alpha} + C_2\epsilon_{adv} + C_3\int_0^T \eta(t)\,dt,$$

with constants C_1, C_2, C_3 combined into C. ■

This unified bound contrasts with previous results in several ways. Traditional bounds, such as those in Theorem 1, focus solely on network width and depth, while Theorem 2 extends these to account for learning rate dynamics. However, neither considers the interplay of sparse activation or adversarial robustness. The inclusion of ρ addresses the efficiency and sparsity of neuron usage, providing a finer granularity in error estimation that is crucial for resource-constrained environments. The adversarial term ϵ_{adv} introduces a critical consideration for model robustness, which is increasingly important in security-sensitive applications. Finally, the time-varying learning rate integral captures the practical effects of adaptive learning strategies, which are ubiquitous in modern training regimens. The inclusion of these factors renders the bound more comprehensive, offering a realistic assessment of ReLU NNs' performance in practical settings. This result is particularly impactful for the design and training of neural networks where computational efficiency, robustness, and adaptability are critical. By unifying these aspects into a single error bound, this theorem not only advances theoretical understanding but also provides

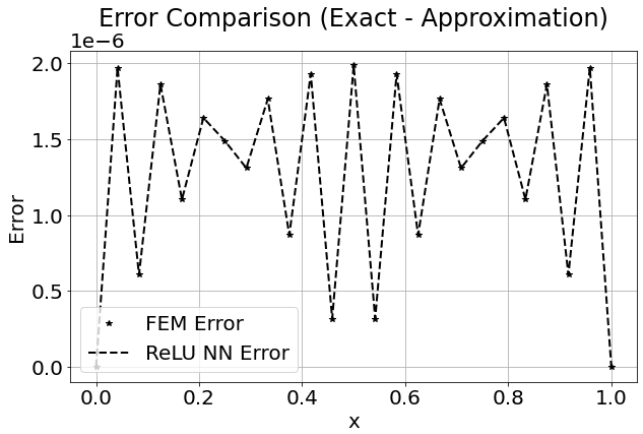

Fig. 1: Approximation errors for the FEM and the ReLU NN

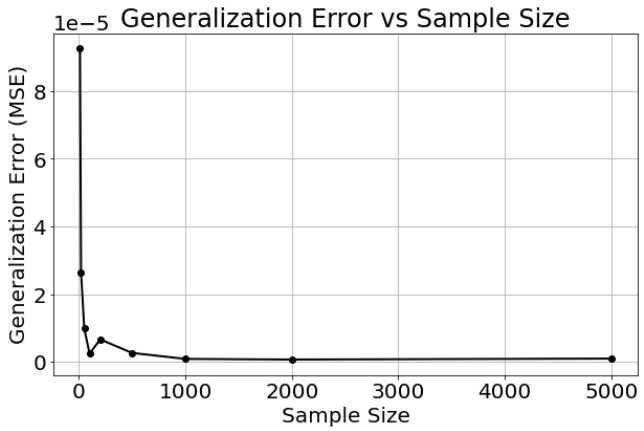

Fig. 2: Sample size vs generalization

actionable insights for the optimization and deployment of ReLU NNs in real-world scenarios. Together, these theorems provide a comprehensive understanding of the error bounds and approximation properties of ReLU neural networks and PINNs in solving PDEs. They highlight the interplay between network architecture, training procedures, and the inherent properties of the solutions, offering a solid theoretical foundation for using neural networks in scientific computing. By systematically building from generalization error bounds to approximation capabilities and the effects of initialization and learning rates, this collection of theorems presents a coherent narrative that guides researchers and practitioners in leveraging neural networks for solving complex PDEs with confidence in their theoretical underpinnings and practical performance.

IV. Numerical Results

We demonstrate the applicability of these theoretical results in solving a boundary value problem (BVP) and some partial differential equations (PDEs). Consider the BVP

$$-u''(x) = \pi^2 \sin(\pi x), \text{ for } x \in (0,1), \tag{11}$$

with $u(0) = u(1) = 0$. The exact solution to this BVP is $u(x) = \sin(\pi x)$. Numerical results from solving equation (11) using both the FEM and ReLU NN indicates that the NN eventually achieves a lower error (0.00164) than FEM (0.00462), considering 10 tent functions and 100 neurons on the single layer for the NN. However, we note also, that 30 neurons for the NN are sufficient to give an error less than that of the 10-tent function FEM as shown in Figure 3. These results suggest that the NN captures the solution's characteristics almost as accurate as the FEM does. This approach is more useful in areas where FEM typically encounters challenges, such as in handling high gradients or non-uniformities in the solution. Figure 2 presents the generalization error of the NN as a function of the number of training samples. The decreasing trend in error as the sample size increases highlights the NN's ability to generalize better with more data, a common characteristic of deep learning models. This also emphasizes the importance of sufficient training data to achieve an accurate NN model. In contrast, traditional methods like FEM are typically less sensitive to the amount of training data but may require careful mesh refinement to achieve similar accuracy. As the FEM mesh

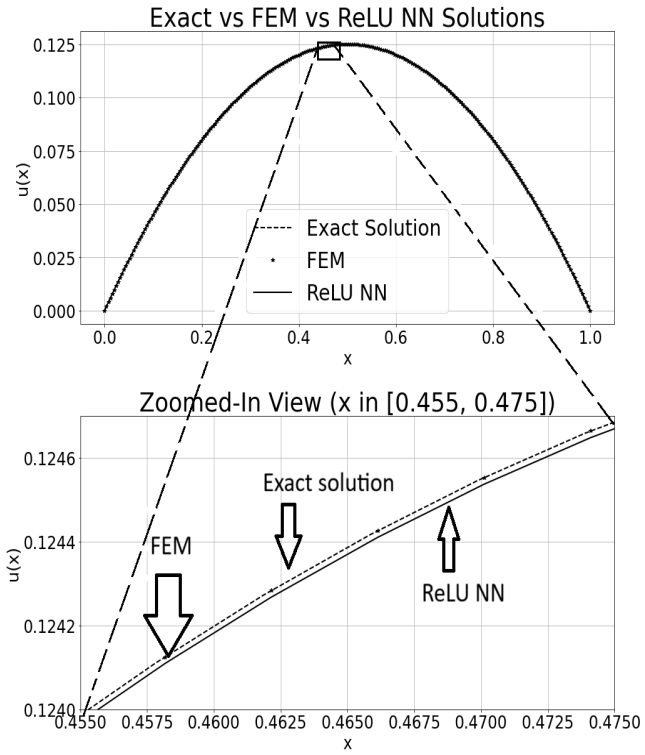

Fig. 3: Exact vs FEM and ReLU NN solutions for equation 12, along with a zoom-in.

is refined (smaller h) and as the NN complexity increases (larger k), both methods approach the exact solution. Notably, the NN solution seems to achieve a closer fit with lower complexity. The second subplot focuses on the error from the first simulations of solutions to the BVP (11). The error decreases with finer FEM meshes and more complex NNs, but the NN generally maintains a lower error across the board, even with coarser FEM meshes and less complex networks. This highlights the NN's capacity to effectively approximate the solution with fewer parameters or computational resources, offering an advantage in scenarios where FEM might require substantial refinement to achieve comparable accuracy. Hence, these numerical results demonstrate the NN's effectiveness in terms of accuracy and efficiency, particularly when optimized with sufficient training data and complexity. The NN's generalization ability, coupled with its effective error bounds, positions it as a strong alternative

to traditional FEM, especially in complex problem domains. However, there should be q question about the applicability of these results to PDEs. We now apply the theoretical results in this work to solving PDEs. Consider the following Poisson PDE as an example,

$$-\Delta u(x) = -1 \quad \text{in} \quad [0,1], \tag{12}$$

with boundary conditions $u(x) = 0$ on $\partial\Omega, \Omega = [0,1]$. The various plots and data generated even right from and during the training of the network highlight how closely the NN approximates the true solution, depict the approximation and generalization errors, the effects of neuron size and learning rate on the solution and consequently the errors, illustrate the variability in the NN's performance due to different starting conditions, etc. Some of these results are shown in Figures and Tables in the appendix section. But precisely, more layers layers and /or neurons result into lower error bounds. Similarly, increasing the number of elements (akin to neurons in neural networks) also reduces approximation error in FEM. However, there is no direct parallel to increasing "depth" as in neural network layers. That is, the FEM error reduction is primarily influenced by the mesh size and polynomial order, paralleling the neural network's dependency on neuron counts. Furthermore, the effect of smoothness of function on the error is also depicted by the ReLU NN. Smoother functions - with inherently fewer abrupt changes - are more efficiently handled by larger ReLU networks, similar to how the approximation error decreases with an increase in the mesh refinement and is lower for smoother functions, for FEM. Even during the training and testing the mean square error (MSE) declines sharply as the number of neurons increases, stabilizing at higher counts. This behaviour illustrates the significant capacity of large neural networks to minimize errors on both training and unseen data, showcasing their robustness in learning and generalization. In FEM, the error similarly decreases with more refined elements and higher polynomial orders, which effectively capture the function's properties. However, FEM generally exhibits a lower tendency for overfitting compared to neural networks, as FEM's approximation approach is more global and deterministic, thus avoiding the pitfall of fitting too closely to specific data points. Hence, while ReLU neural networks and FEM share some similarities in error variations against their respective features, they differ in their handling of depth and over-fitting. Neural networks require careful balance between network size and complexity to optimize performance without incurring excessive computational costs or succumbing to over-fitting. Conversely, FEM controls error by refining the mesh and increasing polynomial degrees, focusing more on mesh size and polynomial complexity than on the concept of depth. Properly managing these elements is critical for achieving desired accuracy in simulations and predictions across various applications. Now, we examine the variation of error against other network features. Generally, the error of any numerical scheme is expected to be reducing consistently against an increase in the number of iterations. This approach satisfies this expectation. Inequality (10) suggests that the error satisfies

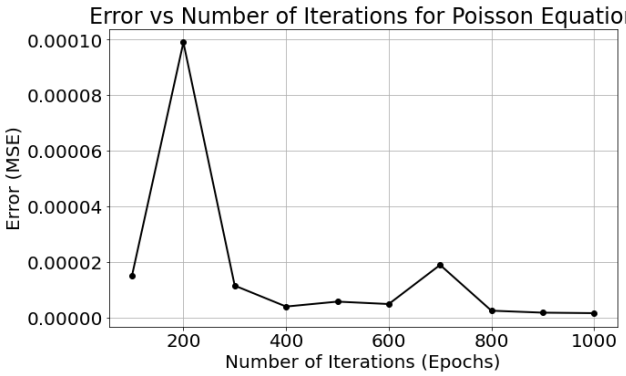

Fig. 4: Error vs number of iterations for the 1D Poisson equation

TABLE I: VARIATION OF ERROR AND ITERATIONS, WITH $k = 50, C = 1, D = 1, h = 0.001$

Iterations	100	9000	25000
Error (e)	9.5312×10^{-1}	1.7438×10^{-2}	1.0644×10^{-4}

TABLE II: ERRORS FOR THE PARABOLIC PDE

Number of Neurons, k	Iterations		
	100	9000	25000
5	3.1236×10^{-1}	6.8724×10^{-2}	8.0875×10^{-3}
50	1.2595×10^{-2}	5.5326×10^{-3}	3.15154×10^{-2}
500	3.0039×10^{-4}	8.9746×10^{-5}	3.3815×10^{-6}

the relation

$$\ln(\|u(x) - u_{NN}\|_1) < \ln\left(Ch + \frac{6D}{k^2}\right), \\ = \ln(Chk^2 + 6D) - 2\ln k\,.$$

Now let $e = \|u(x) - u_{NN}\|_1$. Then

$$\ln(e) + 2\ln k < \ln(Chk^2 + 6D),$$

A plot of this relationship is shown in the Figure 4 below, which is obtained after from solving the 1D Poisson equation using our ReLU NN approach and examined the variation of its error against the different numbers of iterations used in solving. The results are shown in Figure 4 and clearly obey the normal expectation. of error variation against number os iterations. Now, the results depicted is seemingly yet random in nature, yet that is possibly due to the random initialization used in these experiments of solving the Poisson equation. With random initialization, the network is bound to converge around the same optimal points (solutions) yet monotonicity can hardly be guaranteed. Moreover, even with monotone initialization, outputs are not necessarily expected to be monotone. Other experiments focused on experimentally pointing out how the k values, i.e. the number of neurons on the hidden layer of a single layer ReLU NN, as well as the number of iterations influences the error and consequently the proposed bound. For the influence of iterations on the error, in regard to some other solved PDEs, the results are depicted in table I. Furthermore, for the effect of number of neurons on hidden layer on the error, the results are also shown in tables II - IV. The analysis shows that for ReLU Neural Networks solving the Parabolic PDE, increasing the number of neurons and iterations leads to improved approximation of solutions. Networks with 5 neurons

TABLE III: ERRORS FOR THE ELLIPTIC PDE

Number of Neurons, k	Iterations		
	100	9000	25000
5	4.03562×10^{-1}	8.55429×10^{-2}	3.90387×10^{-2}
50	1.33982×10^{-2}	5.00488×10^{-3}	9.87641×10^{-4}
500	1.21602×10^{-4}	7.34025×10^{-5}	1.00842×10^{-6}

TABLE IV: ERRORS FOR THE HYPERBOLIC PDE

Number of Neurons, k	Iterations		
	100	9000	25000
5	6.06774×10^{-1}	3.51892×10^{-1}	1.55531×10^{-1}
50	7.44368×10^{-1}	1.232306×10^{-1}	7.11842×10^{-2}
500	5.52081×10^{-1}	9.91297×10^{-2}	1.00829×10^{-2}

exhibit gradual error reduction, but errors remain relatively high even after 25000 iterations, suggesting limited capacity. Networks with 50 neurons show faster error reduction, yet a slight increase after 25000 iterations indicates potential overfitting. Networks with 500 neurons achieve significantly lower errors, underscoring the importance of network size. However, careful regularization is crucial to prevent overfitting with larger networks, highlighting a trade-off between network complexity and computational resources. The analysis highlights the performance of ReLU Neural Networks in solving the Elliptic PDE. Networks with 5 neurons exhibit higher errors compared to the Parabolic PDE, reflecting the increased complexity. However, these errors decrease with iterations, indicating the network's learning capability. Networks with 50 neurons show improved approximation, but a slight increase in errors after 25000 iterations suggests potential overfitting. Networks with 500 neurons achieve significantly lower errors, underscoring the importance of network size. Despite this, careful regularization is essential to prevent overfitting, echoing the trade-off between network complexity and computational resources observed in solving the Parabolic PDE. Overall, increasing the number of neurons and iterations enhances solution approximation, albeit requiring careful management of network complexity and regularization. For the 5-neuron network, errors in the hyperbolic PDE remain high even after 25,000 iterations, indicating difficulty in accurate representation and suggesting that hyperbolic PDEs are more challenging than parabolic and elliptic ones. With 50 neurons, errors decrease but remain significant, pointing to the need for larger networks. A 500-neuron network significantly reduces errors, highlighting the importance of network size. Errors rapidly decrease with more iterations, demonstrating the network's ability to learn complex patterns. These results confirm that increasing neurons and iterations reduces errors, reflecting the distinct behaviors of different PDE types. Figure 5 compares the exact and ReLU NN solutions for $u(x)$ over $[0, 1]$. The ReLU network approximates the function well but shows error spikes around $x = 0.2$, $x = 0.4$, and $x = 0.8$, highlighting areas needing refinement. Increasing layer width generally reduces error. Notably, the NN approximation's effectiveness, though errors increases near the domain's edges, suggesting necessity for further optimization. Figure 6 shows the impact of initialization and learning rate on ReLU NN performance, where proper initialization and moderate learning rates lead to accurate results, while poor initialization and high rates

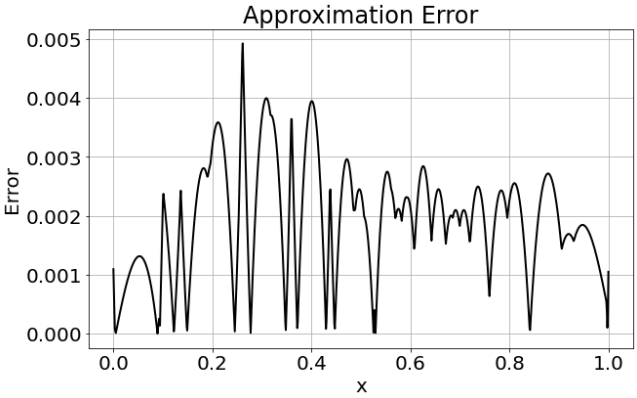

Fig. 5: Approximation Errors

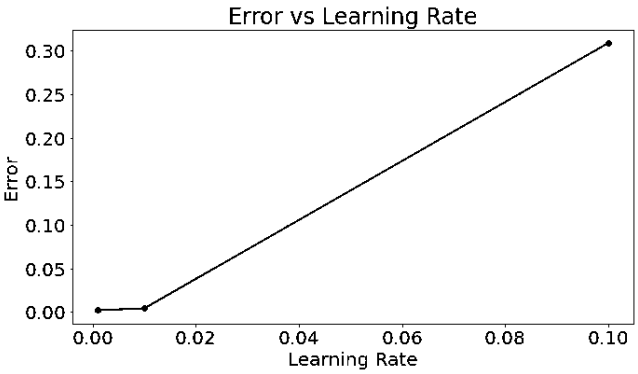

Fig. 6: Effect of learning rates on errors

TABLE V: Neural Network Training Parameters and Results

Parameter	**Value**
Layers (L)	10
Neurons per Layer (N)	100
Optimizer	Adam
Loss Function	MSE
Epochs	100
Batch Size	32
Generalization Error	0.0021

TABLE VI: Generalization Errors for Different Random Seeds

Random Seed	**Generalization Error**
42	0.0021
52	0.0023
62	0.0020

TABLE VII: Generalization Errors for Different Learning Rates

Learning Rate	**Generalization Error**
0.001	0.0021
0.01	0.0020
0.1	0.0025

increase errors. This emphasizes the need for careful hyperparameter tuning and initialization strategies.

Table V summarizes the training parameters and outcomes of a neural network with 10 layers of 100 neurons each, using the Adam optimizer and Mean Squared Error (MSE) as the loss function. Training spanned 100 epochs with a batch size of 32, resulting in a generalization error of 0.0021, indicating effective learning without overfitting. Table VI shows generalization errors of 0.0021, 0.0023, and 0.0020 for seeds 42, 52, and 62, respectively, illustrating the impact

TABLE VIII: Generalization Errors for Different Random Seeds and Learning Rates

Random Seed	Learning Rate	Generalization Error
42	0.001	0.0021
42	0.01	0.0020
42	0.1	0.0022
52	0.001	0.0023
52	0.01	0.0021
52	0.1	0.0024
62	0.001	0.0020
62	0.01	0.0019
62	0.1	0.0023

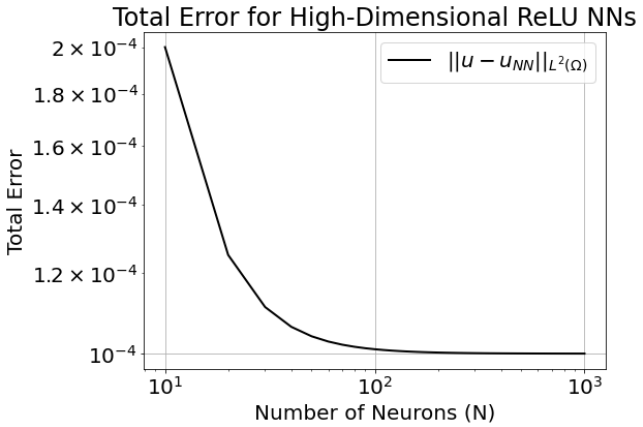

Fig. 7: Variation of high-dimensional ReLU NNs against number of neurons

of random seed variation. Table VII explores learning rates, with 0.01 achieving the lowest error of 0.0020, emphasizing the importance of tuning this parameter. Table VIII combines the effects of random seeds and learning rates, confirming that a rate of 0.01 is generally optimal, with seed 62 and a rate of 0.01 yielding the lowest error of 0.0019. These tables underscore the need for careful tuning and multiple runs to ensure robust model performance.

The other considered experiments include an attempt to extrapolate our results to higher dimensions, the effect of number of neurons in Sobolev norms, as well as the error for networks with varying widths as shown in Figures 7 - 9 respectively. Figure 7 shows that an increase in the number of neurons results into a significant decrease in the error,

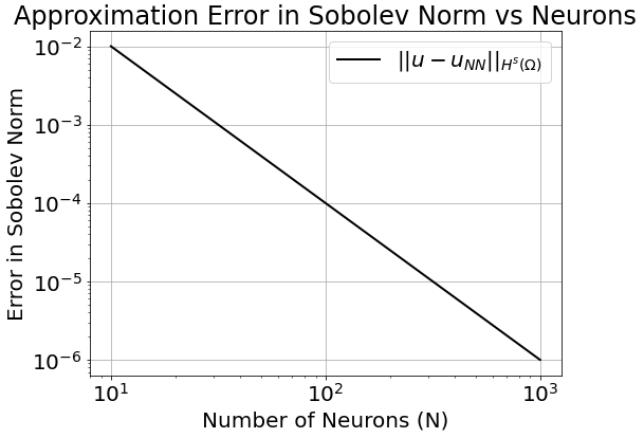

Fig. 8: Approximation error in Sobolev Norms against number of neurons

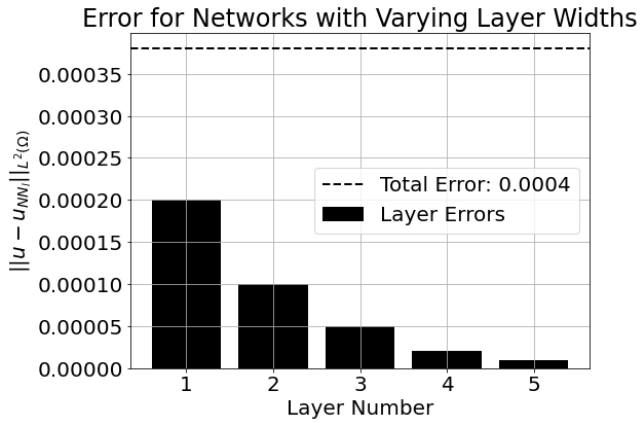

Fig. 9: Effect of varying layer widths on ReLU NNs errors

eventually stabilizing around $N \sim 10^3$. eventually stabilizing around $N \sim 10^3$. This rapid error reduction is consistent with the theoretical bound $\|u(x) - u_{NN}(x)\|_{L^2(\Omega)} \leq CN^{-\frac{2s}{d}}$, which predicts that the error should decay as a function of neuron count, particularly in high-dimensional spaces where larger networks are needed to capture the complexity of the solution. The plateau in the error reduction suggests a point of diminishing returns where adding more neurons does not lead to a proportionate improvement, potentially due to limitations in the expressivity of the network or the optimization process. This result underscores the efficiency of ReLU NNs in high-dimensional approximation but highlights that further gains beyond a certain threshold may be limited.

Figure 8, shows the error in Sobolev norm $H^s(\Omega)$ relative to the number of neurons. The log-log plot displays a linear relationship, reflecting a power-law decay of error with increasing neurons, in line with the theoretical bound $\|u - u_{NN}\|_{H^s(\Omega)} \leq CN^{-2s/d}$. The Sobolev norm captures not only the function's approximation but also its smoothness properties, making this an important metric for assessing how well ReLU networks approximate functions with a certain regularity. The steep and consistent decline in the error suggests that ReLU NNs perform exceptionally well in approximating functions within Sobolev spaces, particularly when the regularity parameter s is large enough relative to the dimensionality d.

Finally, Figure 9 details the error contribution of each layer in a neural network with different layer widths N_l. The plot reveals that the early layers contribute the most to reducing the total error, while later layers provide diminishing improvements. This cumulative error is consistent with the theoretical sum $\|u(x) - u_{NN}(x)\|_{L^2(\Omega)} \leq C \sum_{l=1}^{L} N_l^{-1/d}$, where each layer improves the overall approximation. The first few layers capture the bulk of the function's complexity, while subsequent layers fine-tune the model's performance. This emphasizes the importance of network architecture, where a well-balanced allocation of neurons across layers ensures optimal error reduction. Consequently, early layers in a ReLU network play a critical role in minimizing approximation error, while additional layers serve to refine the model's predictions.

Overall, the plots collectively validate the theoretical error bounds for ReLU NNs in both Sobolev and high-dimensional settings. They highlight the key roles of neuron count and

layer architecture in minimizing approximation error, providing insights into how ReLU NNs can be effectively scaled and structured to tackle complex function approximation tasks.

V. Conclusion

This work has presented both theoretical and numerical results on error bounds for rectified linear unit (ReLU) NNs. We have derived error bounds for ReLU NNs considering finite element method (FEM) schemes, through designing them as linear finite elements. In our error bound, we have emphasized the reliance of the error on the network architecture, specifically, the number of layers and neurons.

Our numerical results illustrate the effectiveness of ReLU NN solutions and the applicability of the derived error bounds. These results contribute to the understanding and design of a rigorous framework for the study and evaluation of the performance of neural network-based approaches in solving PDEs. To enhance error bounds for ReLU NNs, integrating uncertainty quantification techniques may be better for bounds that accounting for variability in predictions. Moreover, considering the impact of data distribution and quality on error bounds might also offer insights into ReLU NNs' generalization capabilities. Scaling these results to higher dimensions and more complex architectures, optimizing network architectures and training methodologies, to include hyper-parameter tuning and regularization, may also be wonderful ideas to explore for uniformly low errors.

References

[1] Steffen Goebbels (2022), On sharpness of an error bound for Deep ReLU network approximation, Sampling Theory, Signal Processing and Data Analysis, https://doi.org/10.1007/s43670-022-00020-y.

[2] Yukun Ding, Jinglan Liu, Jinjun Xiong, Yiyu Shi (2010), On the universal approximability and complexity bounds of quantized ReLU neural networks, ICLR Conference paper, 64 - 80.

[3] Tilahun M Getu. (2021), Error Bounds for a Matrix-Vector Product Approximation with Deep ReLU Neural Networks, arXiv:2111.12963v1.

[4] L. Herrmann, J.A.A. Opschoor, Ch. Schwab (2021), Constructive Deep ReLU Neural Networks Approximation, ETH Zurich, Research Report NO. 2021-04.

[5] Johannes Schmidt-Hieber (2021), Deep ReLU Network approximation of functions on a manifold, arXiv:1908.00695v1.

[6] Juncai He (2020), Relu Deep Neural Networks and Linear Finite Elements, Journal of Computational Mathematics, 38 (3), 502–527, http://dx.doi.org/10.4208/jcm.1901-m2018-0160.

[7] Taiji Suzuki, Hiroshi A, Be, Tomoaki Mishimura, (2020), Compression based bound for non-compressed network: unified generalization error analysis of large compressible deep neural networks, ICLR Conference 23-36.

[8] M. Guo and E. Haghighat. Energy-based error bound of physics-informed neural network solutions in elasticity. Journal of Engineering Mechanics, 148(8):04022038, 2022. DOI: 10.1061/(ASCE)EM.1943-7889.0002121.

[9] H. Poincaré (1890), Sur les Equations aux Dérivées Partielles de la Physique Mathématique, American Journal of Mathematics, 3 (12), 211-294, http://www.jstor.org/stable/2369620.

[10] Taiji Suzuki, Fast Generalization error bound of deep learning from a kernel perspective, Proceeings of the 21st International Conefrence on Artificial Intelligence and Sttaistics (AISTATS) 2018, Lanzarote, Spain, PMLR: Volume 84.

[11] Michael Hammers and Michael Kohler, Nonasymptotic bounds on the L_2 error of neural network regression estimates, Annals of the Institute of Statistical Mathematics (AISM), 58:131-151, doi: 10.1007/s10463-005-0005-9

[12] Andrew R. Barron, Approximation and estimation bounds for artificial neural networks Machine Learning, 14: 115-133 (1994)

[13] Krzysztof Ciesielski, Jaroslaw P. Sacha and Krzysztof J. Cios, Synthesis of Feedforward networks in Supremum error bound, IEEE transactions on Neural Networks, Volume 11, Number 6, 1213-1226

[14] Jiaqi Li, Ross Drummond and Stephen R. Duncan, Robust error bounds for quantised and pruned neural networks, Proceedings of Machine Learning Research, Volume 144: 1-12, 2021