Functional Analysis of Operator Learning: Quantitative Error Bounds in Sobolev Spaces

Nicole Hao

May 16, 2025

1 Introduction

Operator learning refers to the task of approximating mappings between infinite-dimensional function spaces, such as those arising from solutions of partial differential equations (PDEs) parameterized by initial conditions or coefficients.

Formally, we seek to learn an operator

$$\mathcal{G} : \mathcal{X} \to \mathcal{Y}$$

, where $\mathcal{X}$ and $\mathcal{Y}$ are typically Banach or Hilbert spaces of functions, such as Sobolev spaces.

This learning paradigm/scheme has gained attention in the scientific machine learning filed due to its ability to bypass traditional numerical solvers and directly model complex solution operators from data. Neural operator architectures, such as DeepONet [1] and Fourier Neural Operators [**?**], have demonstrated remarkable empirical performance on benchmark PDEs. However, their theoretical underpinnings, especially in the context of Sobolev function spaces, are not well understood.

From the perspective of functional analysis, learning an operator between Sobolev spaces $\mathcal{X} = H^s(D)$ and $\mathcal{Y} = H^t(D')$ raises important questions about approximation theory, continuity, and generalization in high-dimensional regimes. Although neural operators like DeepONet [1] show empirical success, their theoretical foundations in Sobolev spaces remain relatively underdeveloped.

This project connects the studies on functional analysis, specifically Sobolev spaces, and operator learning. Through this project, I aim to contribute to the general understanding on this topic by

1. **Establishing approximation bounds** for deep neural networks mapping between Sobolev spaces $\mathcal{X} = H^s(D) \to \mathcal{Y} = H^t(D')$, with explicit dependence on domain dimension and smoothness as a continuation and specification of the universal approximation theorem.

2. **Validating the theory established numerically** using PDE solution operator learning (See conclusions and future work for the code for numerical experiments, results will be updated in the Github repo)

This goal has implications to the scientific machine learning fields. This is because many physical systems, especially those governed by partial differential equations (PDEs), are naturally described by mappings between functions.

For example, solving a PDE often amounts to computing a solution function $u(x,t)$ given an initial or boundary condition $u_0(x)$. Such mappings are not pointwise but involve entire functions as both input and output, making them *operators*.

Without approximation bounds, we have no rigorous proof/evidence that a neural network can actually learn the target operator to any desired accuracy. These bounds tell us that approximation is possible using neural networks in principle. Also, I believe that this approximation can be used as a measure of how difficult it is to achieve a given level of accuracy (perhaps through network size or depth).

To (again, rigorously) model such mappings, we specifically consider **Sobolev spaces**. I chose Sobolev spaces for several reasons. Firstly, and in my opinion the most important reason, is that Sobolev space include weak derivatives. Many solutions to PDEs (like shock waves in fluid dynamics) may not be classically differentiable. Sobolev spaces $H^s(D)$ accommodate such functions by allowing *weak derivatives*, which makes them ideal for modeling both classical and weak solutions. Secondly, Sobolev spaces H^s are *Hilbert spaces*, meaning they have inner products. This provides powerful tools from functional analysis, such as orthonormal bases, projection theorems, and spectral theory. And finally, the Rellich–Kondrachov compactness theorem, bounded subsets of H^s embed compactly into lower-order Sobolev or continuous spaces. This makes it mathematically possible to approximate infinite-dimensional mappings using finite-dimensional neural networks. I will dedicate an entire section later on the Rellich–Kondrachov compactness theorem, and how it helps us achieve the goal of establishing approximation bounds for DNNs mappings.

2 Literature Review

This project was mainly inspired by **A Mathematical Analysis of Neural Operator Behaviors (Le & Dik, 2024)** and can be seen as an extension of the paper, but on the operator learning problem. Here's a list of other papers I read to set up this project (not ranked by relevance):

Deep Operator Networks (DeepONet) [1] introduced one of the first architectures for learning nonlinear operators from data. Based on the universal approximation theorem for operators, DeepONet consists of a branch network for encoding input functions and a trunk network for output evaluation, showing strong performance on a variety of differential equations.

Fourier Neural Operators (FNO) [?] this paper showed a different approach by parameterizing integral kernels directly in Fourier space. The focus of this method was on achieving fast and accurate operator learning (for parametric PDEs like turbulent flow simulations).

A Mathematical Guide to Operator Learning [?] a comprehensive review of neural operator theory and practice. It covers key architectures, approximation theory, and insights from infinite-dimensional analysis.

3 Sobolev Spaces and Operator Learning

3.1 Problem Setup

Let $D \subset \mathbb{R}^d$ be a bounded Lipschitz domain. For $s \in \mathbb{N}$, the Sobolev space $H^s(D)$ consists of square-integrable functions with weak derivatives up to order s also square-integrable:

$$H^s(D) := \left\{ f \in L^2(D) \;\middle|\; \partial^\alpha f \in L^2(D),\ \forall |\alpha| \leq s \right\},$$

with the norm

$$\|f\|_{H^s(D)} := \left(\sum_{|\alpha| \leq s} \int_D |\partial^\alpha f(x)|^2 \, dx \right)^{1/2}.$$

Let $\mathcal{X} = H^s(D)$ and $\mathcal{Y} = H^t(D')$, where $D' \subset \mathbb{R}^{d'}$ is another bounded Lipschitz domain and $t \in \mathbb{N}$.

Given a nonlinear operator $\mathcal{G} : \mathcal{X} \to \mathcal{Y}$, the current objective, as defined in the introduction, is to approximate $\mathcal{G}$ using a deep neural network $\mathcal{G}_\theta$, where θ represents the parameters, such that the approximation is uniformly accurate on a compact set $\mathcal{K} \subset \mathcal{X}$.

$$\sup_{f \in \mathcal{K}} \|\mathcal{G}(f) - \mathcal{G}_\theta(f)\|_{H^t(D')} < \varepsilon.$$

This formulation is a classic formulation of *operator learning* in function spaces, where the goal is to learn a map between infinite-dimensional spaces with controlled approximation error. In this project, I will be using finite-dimensional NNs.

3.2 Compactness via Rellich–Kondrachov Theorem

As mentioned previously, an important reason why I decided to choose Sobolev spaces is because of the finite-dimensional approximability of $\mathcal{G}$ hinges on the compactness of Sobolev embeddings.

Rellich–Kondrachov Theorem: Let $D \subset \mathbb{R}^d$ be a bounded Lipschitz domain. If $s > t + d/2$, then the embedding $H^s(D) \hookrightarrow H^t(D)$ is compact.

Proof Sketch. Let $\{f_n\} \subset H^s(D)$ be bounded. By the Banach–Alaoglu theorem, it has a weakly convergent subsequence in H^s. The Rellich–Kondrachov theorem guarantees strong convergence in $H^t(D)$, hence precompactness. We will use this theorem later in 2 proofs (see nd 5). □

This compactness implies that for any compact $\mathcal{K} \subset H^s$, the image under $\mathcal{G}$ can be uniformly approximated in H^t by finite-dimensional projections, a key step in neural operator approximation.

4 Reformulation of the Universal Approximation in Sobolev Norms

We now formalize a universal approximation result for operator learning in the Sobolev setting. I will begin with a formal approximation result. Instead of a grid-based proof like theorem 3.10 from A Mathematical Analysis of Neural Operator Behaviors by Le and Dik, I tried to contribute by reformulating it using functional projection and basis expansion (I also believe that this is a better exercise given the focus of the course on functional spaces and their properties).

Let $\mathcal{G} : H^s(D) \to H^t(D')$ be a continuous nonlinear operator, and let $\mathcal{K} \subset H^s(D)$ be compact.

Then for any $\varepsilon > 0$, there exists a ReLU neural network $\mathcal{G}_\theta$ such that:

$$\sup_{f \in \mathcal{K}} \|\mathcal{G}(f) - \mathcal{G}_\theta(f)\|_{H^t(D')} < \varepsilon.$$

Proof. We construct the approximation in three steps:

Step 1: Finite-dimensional projection. We let $\{\phi_k\}_{k=1}^{\infty} \subset H^s(D)$ be an orthonormal basis. Define a projection operator $P_N f$ as

$$P_N f = \sum_{k=1}^{N} \langle f, \phi_k \rangle \phi_k.$$

Since $\mathcal{K} \subset H^s(D)$ is compact and $P_N f \to f$ in H^s, we also have (by continuity of $\mathcal{G}$) that

$$\mathcal{G}(P_N f) \to \mathcal{G}(f)$$

in H^t uniformly on $\mathcal{K}$

Step 2: To reduce to finite-dimensional learning, we let

$$\mathbf{c}_N(f) = (\langle f, \phi_1 \rangle, \ldots, \langle f, \phi_N \rangle) \in \mathbb{R}^N.$$

Then $\mathcal{G} \circ P_N$ can be viewed as a map

$$\mathbb{R}^N \to H^t(D').$$

Define ψ_k as a basis for $H^t(D')$, and define

$$\mathcal{G}_N(f) := \sum_{j=1}^{M} g_j(\mathbf{c}_N(f))\psi_j$$

for suitable continuous g_j.]

Step 3: Since $\mathbf{c}_N(\mathcal{K}) \subset \mathbb{R}^N$ is compact and g_j are continuous, we may approximate each g_j uniformly on this set by a ReLU network $g_{j,\theta}$, using the universal approximation theorem in finite dimensions.

Define:

$$\mathcal{G}_\theta(f) := \sum_{j=1}^{M} g_{j,\theta}(\mathbf{c}_N(f))\psi_j.$$

Then for $f \in \mathcal{K}$:

$$\|\mathcal{G}(f) - \mathcal{G}_\theta(f)\|_{H^t} \leq \|\mathcal{G}(f) - \mathcal{G}(P_N f)\|_{H^t} + \|\mathcal{G}(P_N f) - \mathcal{G}_\theta(f)\|_{H^t}.$$

So, if we choose N large enough that the first term is less than $\varepsilon/2$, and approximate g_j well enough to make the second term $< \varepsilon/2$.

Hence the total error is less than ε, uniformly on $\mathcal{K}$. This concludes the proof. □

In this section, I used the compactness of the embedding $H^s \hookrightarrow H^t$ (using the Rellich–Kondrachov Theorem) and the universal approximation property of neural networks in finite-dimensional spaces. We essentially used projection to reduce the infinite-dimensional operator learning problem to a function approximation that can be evaluated on finite-dimensional neural networks.

5 Quantitative Approximation Bound for Operator Learning

Now that we have explained the universal approximation in sobolev norms, we can build on it and derive an original theorem that gives an **explicit bound** on the neural network complexity required to **approximate a nonlinear operator between Sobolev spaces**.

Quantitative Approximation Bound: Let $\mathcal{G} : H^s(D) \to H^t(D')$ be a continuous operator between Sobolev spaces, where $D, D' \subset \mathbb{R}^d$ are bounded Lipschitz domains and $s > d/2$. Let $\mathcal{K} \subset H^s(D)$ be compact. Then for any $\varepsilon > 0$, there exists a ReLU neural network operator $\mathcal{G}_\theta$ with $\mathcal{O}(\varepsilon^{-d/s})$ parameters such that

$$\sup_{f \in \mathcal{K}} \|\mathcal{G}(f) - \mathcal{G}_\theta(f)\|_{H^t(D')} < \varepsilon.$$

Proof. Let $\{\phi_k\} \subset H^s(D)$ be an orthonormal basis, and define the projection

$$P_N f := \sum_{k=1}^{N} \langle f, \phi_k \rangle \phi_k.$$

By the Rellich–Kondrachov theorem and continuity of $\mathcal{G}$, there exists N such that

$$\sup_{f \in \mathcal{K}} \|\mathcal{G}(f) - \mathcal{G}(P_N f)\|_{H^t(D')} < \varepsilon/2.$$

Let

$$\mathbf{c}(f) := (\langle f, \phi_1 \rangle, \ldots, \langle f, \phi_N \rangle) \in \mathbb{R}^N$$

, and define $\mathcal{G}(P_N f) =: F(\mathbf{c}(f)) \in H^t(D')$. Write F in terms of a basis $\{\psi_j\} \subset H^t(D')$ as

$$F(\mathbf{c}) = \sum_{j=1}^{M} g_j(\mathbf{c}) \psi_j.$$

Each g_j is continuous on the compact set $\mathbf{c}(\mathcal{K}) \subset \mathbb{R}^N$, so we can approximate it uniformly by a ReLU neural network $g_{j,\theta}$ such that

$$\sup_{\mathbf{c} \in \mathbf{c}(\mathcal{K})} |g_j(\mathbf{c}) - g_{j,\theta}(\mathbf{c})| < \frac{\varepsilon}{2M}.$$

Define the neural network operator:

$$\mathcal{G}_\theta(f) := \sum_{j=1}^{M} g_{j,\theta}(\mathbf{c}(f)) \psi_j.$$

Then for all $f \in \mathcal{K}$,

$$\|\mathcal{G}(f) - \mathcal{G}_\theta(f)\|_{H^t} \leq \|\mathcal{G}(f) - \mathcal{G}(P_N f)\|_{H^t} + \|\mathcal{G}(P_N f) - \mathcal{G}_\theta(f)\|_{H^t} < \varepsilon.$$

Finally, from classical results on approximation by neural networks in $\mathbb{R}^N$, approximating Lipschitz functions g_j to accuracy δ requires $\mathcal{O}(\delta^{-N})$ parameters. Since the projection error scales like $\mathcal{O}(N^{-s/d})$, we choose $N = \mathcal{O}(\varepsilon^{-d/s})$. This yields the stated bound. □

6 Conclusions and Furture Work

To validate our approximation theorem, I tried training a Fourier Neural Operator (FNO) to learn the solution operator of the 1D viscous Burgers' equation:

$$\partial_t u(x,t) + u(x,t)\partial_x u(x,t) = \nu \partial_{xx} u(x,t), \quad x \in [0,1],\ t \in [0,1],$$

with periodic boundary conditions and viscosity $\nu > 0$. The operator of interest maps the initial condition $u_0 \in H^1([0,1])$ to the solution at final time $u(\cdot, 1) \in H^1([0,1])$:

$$\mathcal{G} : u_0 \mapsto u(\cdot, 1).$$

Then the plan was to approximate $\mathcal{G}$ using an FNO trained on data pairs $(u_0, u(\cdot, 1))$ generated by a spectral PDE solver. Each u_0 is sampled from a distribution of smooth functions with finite H^1-norm. The FNO consists of four Fourier layers and is trained with the Adam optimizer over 100 epochs.

However, due to the time limit for the project, (and a very hectic finals season), I decided to include further numerical results and the accompanying code in this GitHub repository, which will continue to be updated.

I believe that these results should empirically validate our theoretical approximation result as well as further validating the original Theorem 3.10 (Universality of Neural Operators for PDE Solvers) from the 2024 paper A Mathematical Analysis of Neural Operator Behaviors. The low H^1-error indicates that the neural network is learning not only function values but also their derivatives. This justifies that it approximates the target operator in the Sobolev norm $\|\cdot\|_{H^1}$.

In the theoretical analysis portion of the report, I showed that any continuous operator $\mathcal{G} : H^s(D) \to H^t(D')$ can be uniformly approximated on compact sets by a finite-dimensional neural network. The final experimental results should provide empirical evidence of this statement by showing the following

- A neural operator (FNO) achieves uniformly small approximation error across a compact set of initial functions $u_0 \in H^1([0, 1])$.
- The errors are measured in the same H^1 norm used in the theorem, confirming alignment between theory and practice.
- The compactness of the data (bounded H^1-norm and smooth initial conditions) justifies application of the Rellich–Kondrachov-based projection argument used in the proof.

A Definitions and Theorems

Note: The appendix is for myself mostly as an reminder of formal definitions. It helps with synergizing/systemizing all the concepts we've learned/touched on throughout the course.

A.1 Lipschitz Domain

A domain $D \subset \mathbb{R}^d$ is called a *Lipschitz domain* if, near every point on its boundary, it can be locally represented as the region above the graph of a Lipschitz continuous function. That is, for every $x_0 \in \partial D$, there exists a neighborhood U of x_0 and a Lipschitz function $\varphi : \mathbb{R}^{d-1} \to \mathbb{R}$ such that (after a coordinate change):

$$D \cap U = \{x = (x', x_d) \in U \mid x_d > \varphi(x')\}.$$

A.2 Sobolev Space

The Sobolev space $H^s(D)$ consists of functions $f \in L^2(D)$ such that all weak partial derivatives $\partial^\alpha f \in L^2(D)$ for $|\alpha| \leq s$. These spaces are Hilbert spaces equipped with the norm:

$$\|f\|_{H^s(D)} := \left(\sum_{|\alpha| \leq s} \int_D |\partial^\alpha f(x)|^2 \, dx \right)^{1/2}.$$

A.3 Weak Derivative

Let $f \in L^1_{\text{loc}}(D)$, where $D \subset \mathbb{R}^d$ is open. We say that $g \in L^1_{\text{loc}}(D)$ is the *weak derivative* of f with respect to x_i if:

$$\int_D f(x) \, \partial_i \varphi(x) \, dx = - \int_D g(x) \, \varphi(x) \, dx \quad \text{for all } \varphi \in C_c^\infty(D).$$

In this case, we write $\partial_i f = g$ in the weak sense.

More generally, for a multi-index $\alpha \in \mathbb{N}^d$, f has weak derivative $\partial^\alpha f \in L^1_{\text{loc}}(D)$ if:

$$\int_D f(x) \, \partial^\alpha \varphi(x) \, dx = (-1)^{|\alpha|} \int_D \partial^\alpha f(x) \, \varphi(x) \, dx \quad \forall \varphi \in C_c^\infty(D).$$

Remarks: Weak derivatives generalize classical derivatives to functions that may not be differentiable in the usual sense. The space $H^s(D)$ is defined using these weak derivatives, allowing for the inclusion of solutions to PDEs that are not classically smooth.

A.4 Square-Integrable Function

A function $f : D \to \mathbb{R}$ is called *square-integrable* if:

$$\int_D |f(x)|^2 \, dx < \infty.$$

The space of such functions is denoted $L^2(D)$, a Hilbert space with inner product $\langle f, g \rangle = \int_D f(x) g(x) \, dx$.

B Appendix: Fourier Neural Operators (FNOs)

The Fourier Neural Operator (FNO), introduced by Li et al. [**?**], is a deep learning architecture designed to learn mappings between infinite-dimensional function spaces—especially solution operators of parametric partial differential equations (PDEs).

Unlike traditional neural networks that act on finite-dimensional vectors, FNOs learn operators of the form:

$$\mathcal{G} : f(x) \mapsto u(x), \quad f \in \mathcal{X}, \ u \in \mathcal{Y},$$

where $\mathcal{X}, \mathcal{Y}$ are typically subsets of $L^2(D)$ or $H^s(D)$ over a spatial domain $D \subset \mathbb{R}^d$. The central innovation of FNO is to parameterize the action of the operator in the **Fourier**

domain, allowing it to efficiently capture long-range dependencies and smooth functional structure.

FNO layers consist of:

- A **Fourier transform** to move the function into frequency space,
- A learned, **diagonal multiplier** (analogous to a convolution kernel) that acts on each frequency mode,
- An **inverse Fourier transform** to return to the spatial domain,
- Pointwise non-linearities and potential skip connections.

Mathematical Structure of an FNO Layer

Let $v : D \to \mathbb{R}^C$ be a function with C channels. An FNO layer updates v as follows:

$$\mathrm{FNO}(v)(x) = \mathcal{F}^{-1}\left(R(\hat{v})\right)(x) + W(v(x)),$$

where:

- $\hat{v} = \mathcal{F}(v)$ is the Fourier transform,
- R is a learned transformation applied mode-wise (typically a complex-valued linear layer on each frequency),
- W is a learned pointwise linear transformation.

The number of retained modes k is typically truncated, introducing an implicit low-pass filter that stabilizes training and improves generalization.

References

[1] Lu, L. et al. (2021). Learning nonlinear operators via DeepONet. *Nature Machine Intelligence*.

[2] Jacot, A. et al. (2018). Neural Tangent Kernel. *NeurIPS*.

[3] Adams, R. (2003). Sobolev Spaces. *Academic Press*.