An application of a PDE-based neural network with Bayesian optimization to regression

Hirotada Honda
Faculty of information and networking for innovation and design
Toyo university
Tokyo, Japan
honda.hirotada@iniad.org

***Abstract*—Application of a PDE-based neural network for regression is proposed. After reviewing the effectiveness of our model, we first confirmed the nonuniform learnability of the model in regression task. Subsequently, we applied it to regression tasks using Bayesian optimization. Numerical experiments show that our model exhibits a performance comparable to that of existing models.**

***Index Terms*—partial differential equation, Bayesian optimization, neural network.**

I. Introduction

Recently, differential equation-based neural networks (NN) have been actively discussed. Following the remarkable concept of neural ODEs [1], numerous models, including partial differential equation (PDE)-based neural networks, have been proposed. Although they are believed to perform well with an infinite number of layers, their theoretical and computational exploration is a topic of ongoing research. On top of this, the optimization procedures of nonlinear differential equation-based neural networks are often laborious to implement. Accompanied by remarkable developments in the optimization literature, represented by Bayesian optimization, we previously proposed the application of Bayesian optimization to ordinary differential equation (ODE)-based neural networks [2]. The effectiveness of PDE-based neural networks [3] have been theoretically proved [4]. There, the universal approximation property and th enonuniform leanability for binary classification tasks have been proven. In this paper, we propose the application of Bayesian optimization to PDE-based neural networks for regression tasks, and evaluate the effectiveness of our methodology through numerical experiments using popular datasets from the literature. We demonstrate that our model exhibits a performance comparable to that of existing methods. Furthermore, we show that the proposed method attains nonuniform learnability when applied to regression tasks with least squares at the output layer.

II. Terms and notations

A. General notations

We use the notation $I = (0, 1)$. Let $\mathcal{G}$ be an arbitrary open set in Euclidean space. With an arbitrary $T > 0$ we denote $\mathcal{G}_T \equiv \mathcal{G} \times (0, T)$ and $\partial\mathcal{G}$ denotes the boundary of $\mathcal{G}$. The notation $C^l(\mathcal{G})$ $(l \in \mathbb{N})$ denotes a set of functions defined on $\mathcal{G}$ with l-th order continuous derivatives. $L_2(\mathcal{G})$ denotes a set of square-integrable functions defined on $\mathcal{G}$, equipped with the norm $\|f\|_{L_2(\mathcal{G})} \equiv \left(\int_{\mathcal{G}} |f(x)|^2 \, \mathrm{d}x\right)^{\frac{1}{2}}$. For $u, v \in L_2(\mathcal{G})$, we denote their inner product in this space by $\langle u, v\rangle$. For $r \in \mathbb{N}$, we define the Sobolev spaces $H^r(\mathcal{G})$, which are spaces of functions $f(x)$, $x \in \mathcal{G}$, equipped with the norm $\|f\|^2_{H^r(\mathcal{G})} \equiv \sum_{|\alpha|\le r} \|D^\alpha f\|^2_{L_2(\mathcal{G})}$. Given $T > 0$, we use the notation $\mathcal{H}_T \equiv (0, T) \times I \times I$. Later, we will introduce two functions $w_0(x)$ and $w_1(t, x, y)$, which correspond to the weight parameters in the output and hidden layers of the NN. They are elements of $L_2(I)$ and $L_2(\mathcal{H}_T)$, respectively. For $p \in [i, +\infty]$, we define the Besov space by

$$B^{s,W}_{p\infty}(\mathbb{R}^d) = \Big\{ f \in L_p(\mathbb{R}^d) \big| \|f\|_{B^{s,W}_{p\infty}} < +\infty \quad p \in [1, +\infty] \Big\},$$

where

$$\|f\|_{B^{s,W}_{p\infty}} \equiv \big(\sum_{k\in\mathbb{Z}^d} \big|\langle f, \Phi_k\rangle\big|^p \big)^{1/p} + \sup_{l\ge 0} 2^{l(s+d/2-d/p)} \big(\sum_{k\in\mathbb{Z}^d, \tau\in\mathscr{T}} \big|\langle f, \Psi^\tau_{lk}\rangle\big|^p \big)^{1/p},$$

with $\{\Phi_k\}$ and $\{\Psi^\tau_{lk}\}$ being Wavelet basis in $L_2(\mathbb{R}^d)$ [5]. It is known that $H^1(\mathbb{R}^d) \subset B^{s,W}_{2\infty}(\mathbb{R}^d)$, for all $s \in \mathbb{N}$. The norms of vector and product spaces are defined in the usual manner.

B. Gaussian processes

$\mathcal{N}\big(\vec{\mu}, \Sigma\big)$ denotes the multivariate normal distribution, with $\vec{\mu}$ and Σ being the expectation vector and variance-covariance matrix, respectively. We define a Gaussian process [6] [7].

Definition II.1. *Let $\mathcal{X}$ be the subspace of $\mathbb{R}^d$. For a given function $f : \mathcal{X} \to \mathbb{R}$, an arbitrary $n \in \mathbb{N}$ and $\mathscr{E} \equiv (\vec{\xi}_1, \ldots, \vec{\xi}_n) \in \mathcal{X} \times \ldots \times \mathcal{X}$, if a vector-valued function, $\boldsymbol{f} = (f(\vec{\xi}_1), \ldots, f(\vec{\xi}_n))^\top \in \mathbb{R}^n$ satisfies $\boldsymbol{f} \sim \mathcal{N}\big(\vec{m}(\mathscr{E}), \Sigma(\mathscr{E})\big)$ with $\vec{m}(\mathscr{E}) = (m_0(\eta_1), \ldots, m_0(\eta_n))^\top \in \mathbb{R}^n$, and $\Sigma(\mathscr{E}) = \big[k(\eta_i, \eta_j)\big]^n_{i,j=1} \in \mathbb{R}^{n\times n}$ with some functions $m(\cdot) : \mathcal{X} \to \mathbb{R}$ and $k(\cdot, \cdot) : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, then, we say that f follows the Gaussian process [5], and denote by $\boldsymbol{f} \sim \mathrm{GP}\big(\vec{m}(\mathscr{E}), \Sigma(\mathscr{E})\big)$.*

In many cases, they employ a kernel function for $k(\cdot, \cdot)$ and a constant for m_0. Suppose we are provided with a dataset $\mathcal{D} = \{(\vec{\xi}_i, y_i)\}_{i=1}^N \subset \mathcal{X} \times \mathbb{R}$. Subsequently, the joint distribution of observations $\boldsymbol{y} = (y_1, \ldots, y_N)^\top$ and $\boldsymbol{y}^* = (y_1^*, \ldots, y_m^*)^\top$ that corresponds to a new element $\mathscr{E}^* = (\eta_1^*, \ldots, \eta_m^*)^\top$ is:

$$\begin{pmatrix} \boldsymbol{y} \\ \boldsymbol{y}^* \end{pmatrix} \sim \mathcal{N}\left[\begin{pmatrix} m_{0,N} \\ m_{0,m} \end{pmatrix}, \begin{pmatrix} \boldsymbol{K} + \sigma^2 \boldsymbol{I}_N & \boldsymbol{K}^* \\ \boldsymbol{K}^{*\top} & \boldsymbol{K}^{**} \end{pmatrix}\right],$$

where $\sigma^2 \boldsymbol{I}_N$ is the variance of the Gaussian noise of the measurement, $\boldsymbol{I}_N$ is the identity matrix of dimension N. $m_{0,N} = [m_0(\vec{\xi}_i)]_{i=1}^N \in \mathbb{R}^N$, $m_{0,m} = [m_0(\vec{\xi}_j^*)]_{j=1}^m \in \mathbb{R}^m$, $\boldsymbol{K} = [k(\vec{\xi}_i, \vec{\xi}_j)] \in \mathbb{R}^{N \times N}$, $\boldsymbol{K}^* = [k(\vec{\xi}_i, \vec{\xi}_j^*)] \in \mathbb{R}^{N \times m}$, and $\boldsymbol{K}^{**} = [k(\vec{\xi}_i^*, \vec{\xi}_j^*)] \in \mathbb{R}^{m \times m}$. Thus, the conditional distribution of $\boldsymbol{y}^*$ given $\mathcal{D}$ and $\mathscr{E}^*$ follows $\mathcal{N}(\hat{m}, \hat{\Sigma})$, where

$$\hat{m} = m_{0,m} + \boldsymbol{K}^{*\top}\left(\boldsymbol{K} + \sigma^2 \boldsymbol{I}_N\right)^{-1}(\boldsymbol{y} - m_{0,N}) \in \mathbb{R}^m,$$
$$\hat{\Sigma} = \boldsymbol{K}^{**} - \boldsymbol{K}^{*\top}\left(\boldsymbol{K} + \sigma^2 \boldsymbol{I}_N\right)^{-1}\boldsymbol{K}^* \in \mathbb{R}^{m \times m}.$$

C. Bayesian optimization

The Bayesian optimization can be explained in three ways. First, a Gaussian process is used, and the target loss function is assumed to follow the Gaussian process. The mean and variance of the loss function value are predicted using Gaussian process regression. Finally, the acquisition function, which is used as a measure of the preference for the explanatory variable value, is computed. In the experiments, we employed the acquisition function known as *Expectation of Improvement*(EI) [8] defined by $EI(\eta) \equiv \max\{0, f_n^* - f(\eta)\}$.

D. Leanability

Hereafter, we denote a probability space as $(\Omega, \mathscr{A}, P)$, where Ω is the sample space, $\mathscr{A}$, the σ-algebra with respect to probability measure P. We also denote a corresponding empirical measure as $P_m(B) = \frac{1}{m}\sum_{j=1}^m \delta_{\vec{\xi}_j}(B)$ for a Borel set B with $\delta(\cdot)$ being Dirac measure. Given a set $\mathcal{F}$ of integrable functions on Ω, let us define the notations

$$Pf = \int_\Omega f \, \mathrm{d}P, \quad \|P_m - P\|_{\mathscr{F}} \equiv \sup_{f \in \mathscr{F}} \sqrt{m}\big|P_m f - Pf\big|.$$

Definition II.2. *Given a probability space $(\Omega, \mathscr{A}, P)$ and a set of integrable real-valued functions $\mathscr{F}$, we say that $\mathscr{F}$ is a Glivenko-Cantelli class for P if and only if $\|P_m - P\|_{\mathscr{F}} \to 0$ $(m \to +\infty)$ holds almost uniformly.*

III. Related works

In this section, we discuss the studies concerning Bayesian optimization and differential equation-based neural network. Owing to its highly generalizable framework, Bayesian optimization [6] [9] has been widely applied in statistics and machine learning [10], including recommendation systems, experimental designs, environmental monitoring, and sensor networks [11].

Look and Kandemir [12] proposed applying the Bayesian approach to differential neural networks. Dandekar et al. [13] proposed the Bayesian neural ODE method, which employs an MCMC-based posterior distribution in contrast to the proposed method (Bayesian regression differs from Bayesian optimization). The advantages of the Bayesian optimization approach are as follows:

(i) The loss function can be a "black box" function, which is optimized by considering a surrogate quantity called the *acquisition function*. The differentiability, submodurality, or the convexity of the loss function is not required.
(ii) We can try challenging exploratory values using the acquisition function, which yields superior results with higher probability.

Therefore, the Bayesian optimization can be applied to our PDE-based neural network. This measure omits the tedious gradient-descent-based scheme and eases implementation.

IV. Our formulation

A. Overview

In this paper, we restrict ourselves to the regression task. We assume that there exits a data-generating probability distribution $\mathcal{D}$, from which the sample data, which comprises of the pairs of the input and output, say, $\{(\vec{\xi}_i, y_i)\}_{i=1}^m \subset \mathcal{X} \times \mathcal{Y}$, is i.i.d. drawn. We assume that $\mathcal{X} \subset \mathbb{R}^d$ and $\mathcal{Y} \subset \mathbb{R}$. Let us introduce: $u_0(x) = \sum_{j=1}^d \xi_j \chi_{I_j}$, with χ_{I_j} as the indicator functions of $I_j \equiv ((j-1)/d, j/d]$ $(j = 1, 2, \ldots, d)$. We formulate our PDE-based neural network as follows [4].

$$\begin{cases} u_t - \nu u_{xx} = \phi\Bigg(\displaystyle\int_I w_1(t,x,y) u(t,y) \, \mathrm{d}y \\ \qquad\qquad\qquad + \displaystyle\int_I w_1(t,x,y) \, \mathrm{d}y\Bigg) \text{ in } I_T, \\ u(0,x) = u_0(x) - 1 \equiv \tilde{u}_0 \quad \text{on } I, \\ u = 0 \quad \text{on } \partial I \quad \forall t \in (0,T). \end{cases} \tag{IV.1}$$

To (IV.1), the existence of temporally local and global solutions was obtained [4]. Hereafter, we will often denote the solution to (IV.1) as $u(t, x; w_1, \vec{\xi}, \nu)$ for clearly indicating its dependence on w_1, $\vec{\xi}$, and ν. Then, given the terminal moment T and diffusion coefficient ν, we define a hypothesis set. This set comprises functions on $\mathbb{R}^d$ realized by our model:

$$\mathscr{F}_T^{(\nu)} \equiv \left\{\vec{\xi} \longmapsto \int_I w_0(x) u(T, x; w_1 \vec{\xi}, \nu) \, \mathrm{d}x \;\middle|\; (w_0, w_1) \in \mathcal{W}_{ad}\right\}, \tag{IV.2}$$

where the admissible set [14] $\mathcal{W}_{ad}$ denotes a convex bounded set in $L_2(I) \times L_2(\mathcal{H}_T)$. In machine learning, actually, the parameter values are determined according to a dataset and an objective function. Given an objective function $l(\cdot)$ of the form: $l(h; \vec{\xi}, y) = \tilde{l}\left(\int w_0(x) u(T, x; w_1, \vec{\xi}, \nu) \mathrm{d}x, y\right)$, we consider the hypothesis set:

$$\mathscr{L}_T^{(\nu)} \equiv \tilde{l} \circ \mathscr{F}_T^{(\nu)} = \left\{ (\vec{\xi}, y) \longmapsto \tilde{l}\left(\int w_0(x) u(T, x; w_1, \vec{\xi}, \nu) \, \mathrm{d}x, y \right) \middle| (w_0, w_1) \in \mathcal{W}_{ad} \right\}. \tag{IV.3}$$

Then, we should aim at minimizing the risk over a loss function $l(\cdot)$: $L_D(h) = E_{z \sim \mathcal{D}}\Big[l(h; z)\Big]$, where $\mathcal{Z} \equiv \mathcal{X} \times \mathcal{Y}$, with $\mathcal{X}$ being a set of inputs. The notation $z \sim \mathcal{D}$ means that a random variable z is i.i.d drawn from $\mathcal{D}$. Similarly, we use the notation $S \sim \mathcal{D}^m$ to denote that a dataset S of sample size m is i.i.d drawn from $\mathcal{D}$. However, we usually do not know the actual distribution $\mathcal{D}$. For this reason, we usually try to minimize the surrogate quantity, which is called the "empirical risk": $L_S(h) \equiv \frac{1}{m}\sum_{i=1}^m l(h; \vec{\xi}_i, y_i)$, where $S = \{(\vec{\xi}_i, y_i)\}_{i=1}^m \subset \mathcal{Z}$ represents the training data drawn from the original unknown distribution $\mathcal{D}$. This framework is called *empirical risk minimization* (ERM). Utilizing the law of large numbers, $L_S(h)$ converges to the true risk as $m \to +\infty$ for each h. In summary, given a dataset $S \sim D^m$, we consider: $\min_{h \in \mathscr{F}_T^{(\nu)}} L_s(h)$, where $\mathscr{F}_T^{(\nu)}$ is defined in (IV.2) (see, Fig. 1 with r replaced by u). Because what we can modify is $(w_0, w_1) \in \mathcal{W}_{ad}$, this can be equivalently presented as:

$$\min_{(w_0, w_1) \in \mathcal{W}_{ad}} \frac{1}{m} \sum_{i=1}^m \tilde{l}\left(\int w_0(x) u(T, x; w_1, \vec{\xi}_i, \nu) \, \mathrm{d}x, y_i \right).$$

Hereafter, we shall clarify the objective function in this paper and validate the learnability of the problem above.

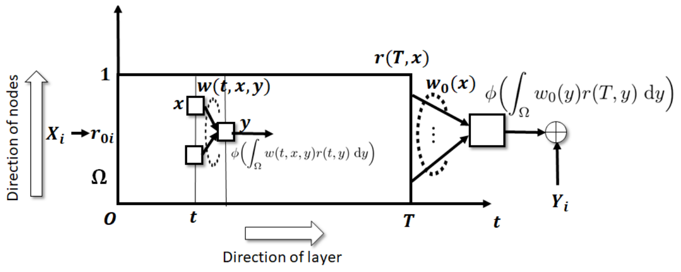
Direction of nodes
1
$X_i \to r_{0i}$
Ω
$w(t,x,y)$
x
y
$\phi\Big(\int_\Omega w(t,x,y) r(t,y)\,\mathrm{d}y\Big)$
$r(T,x)$
$w_0(x)$
$\phi\Big(\int_\Omega w_0(y) r(T,y)\,\mathrm{d}y\Big)$
Y_i
O
t
T
t
Direction of layer

Fig. 1. Overview of the proposed method

B. Proposed formulation in regression task

Because we focus on regression in this paper, we employ the least square function as $\tilde{l}$:

$$\tilde{l}\left(\int w_0(x) u(T, x; w_1, \vec{\xi}, \nu) \, \mathrm{d}x, y \right) = \left| y - \int w_0(x) u(T, x; w_1, \vec{\xi}, \nu) \, \mathrm{d}x \right|^2. \tag{IV.4}$$

Given $\vec{y} = [y_i]_{i=1}^m \in \mathbb{R}^m$, $u_i \equiv u(T, \cdot; w_1, \vec{\xi}_i, \nu)$, and introducing a notation $\vec{u} \equiv [u_i]_{i=1}^m$, this corresponds to finding the solution of $A_{\vec{u}} w_0 = \vec{y}$ with an operator $A_{\vec{u}} : L_2(I) \to \mathbb{R}^m$ defined by $A_{\vec{u}} w_0 = [\langle w_0, u_i \rangle]_{i=1}^m$. By introducing the pseudoinverse operator [15] $A_{\vec{u}}^\dagger = \sum_{i=1}^m y_i u(T, x; w_1, \vec{\xi}_i, \nu)$, we can equivalently write down as $w_0 = A_{\vec{u}}^\dagger \vec{y}$. Concretely, this problem can be presented as follows, which takes the form of the optimal control.

$$\min_{w_1 \in \mathcal{W}_{ad}^{(1)}} L_S^{(\varepsilon)}[h] \equiv \frac{1}{m} \sum_{i=1}^m \tilde{l}\big(\langle A_{\vec{u}}^\dagger \vec{y}, u(T; w_1, \vec{\xi}_i, \nu) \rangle, y_i \big)$$
$$\text{s.t.} \begin{cases} u_t - \nu u_{xx} = \phi\left(\displaystyle\int_I w_1(t, x, y) u(t, y) \, \mathrm{d}y + \int_I w_1(t, x, y) \, \mathrm{d}y \right) \text{ in } I_T, \\ u(0, x) = \tilde{u}_0 \quad \text{on } I, \\ u = 0 \quad \text{on } \partial I \quad \forall t \in (0, T), \end{cases} \tag{P}$$

where the admissible set $\mathcal{W}_{ad}^{(1)}$ denotes a convex bounded set in $L_2(\mathcal{H}_T)$.

C. Our approach

An approach to solve Problem (P) is to consider the gradient of the objective function and update the coefficient to minimize the loss function.

Another direction is a classical one in the optimal control theory; by applying the Pontryagin's maximum principle, we consider the corresponding Euler-Lagrange equation, and then Hamiltonian is minimized [14]. Another possible approach is to apply the Hamilton–Jacobi-Bellman equation, as employed by Han and Li [16] to derive the corresponding viscous solution. However, these approaches pose a significant burden on their implementation. Particularly, the implementation depends on each task; hence, the code requires drastic revisions depending on the purpose (classification, regression, etcetera). In this study, our PDE-based neural network above with Bayesian optimization is proposed, wherein the objective function distribution is traced to seek better parameter values using an acquisition function. This method does not require differentiability, submodularity, or convexity of the objective function, thereby facilitating simple implementation.

We implement the proposed method with Python using GpyOPT [17] and scikit-optimization [18] libraries for Bayesian optimization. The PDE was solved by Crank–Nicolson method. Because we discretize the time interval, we discretize $w_1(t, x, y)$ and $w_0(x)$ with N and L equivalent intervals $\triangle t$ and $\triangle x$, respectively. For $\phi(\cdot)$ in (P), we used a sigmoid function. In our implementation, the acquisition function is a function of: $w_0(t_i, x_j, y_k)$ $(i = 1, 2, \ldots, N, j, k = 1, 2, \ldots, L)$. Herein, we used EI as an acquisition function. At the output layer, we determine $w_0(x)$ so that the quantity in (IV.5) is minimized.

V. NONUNIFORM LEARNABILITY

Although our PDE-based neural network exhibits the universal approximation property [4], by using the concept of a structural risk minimization (SRM) scheme, we can still make it nonuniformly learnable [19]. A relaxation of the concept of learnability of this kind has also been applied to support vector machines [20]. To discuss this in more detail, we introduce some notations.

A. Learnability

To evaluate the 'goodness' of the training data, we define the following concept.

Definition V.1. *A training set S is called ε-representative with respect to the domain $\mathcal{Z} \equiv \mathcal{X} \times \mathcal{Y}$, hypothesis set $\mathscr{F}$, loss function $l(\cdot)$, and distribution $\mathcal{D}$ if the following holds: $\big|L_S(h) - L_{\mathcal{D}}(h)\big| \leq \varepsilon \quad \forall h \in \mathscr{F}$.*

To determine the conditions under which the ERM scheme works well, we need the following definition [19].

Definition V.2. *We say that a hypothesis set $\mathscr{F}$ possesses the uniform convergence property with respect to the domain $\mathcal{Z}$ and loss function $l(\cdot)$ if there exists a function $m_{\mathscr{F}}^{UC} : (0,1)^2 \to \mathbb{N}$, which is called the sample complexity, such that for each $\varepsilon, \delta \in (0,1)$ and for every probability distribution $\mathcal{D}$ over $\mathcal{Z}$, if S is a sample of $m \geq m_{\mathscr{F}}^{UC}(\varepsilon, \delta)$ elements that are drawn i.i.d. according to $\mathcal{D}$, then, with a probability of at least $1-\delta$, S is ε-representative.*

Our hypothesis set $\mathscr{F}_T^{(\nu)}$ does not satisfy the uniform convergence property itself. However, we can also consider a relaxed concept of learnability [19] .

Definition V.3. *A hypothesis set $\mathscr{F}$ is said to be nonuniformly learnable if there exists a learning algorithm A that associates a dataset S with a hypothesis $A(S) \in \mathscr{F}$ and a function $m_{\mathscr{F}} : (0,1)^2 \times \mathscr{F} \to \mathbb{N}$, such that for every $\varepsilon, \delta \in (0,1)$, and for every $h \in \mathscr{F}$, if $m \geq m_{\mathscr{F}}(\varepsilon, \delta, h)$ then for every distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$, with a probability of at least $1-\delta$ over the choice of $S \sim \mathcal{D}^m$, it is ensured that $L_{\mathcal{D}}(A(S)) \leq L_{\mathcal{D}}(h) + \varepsilon$.*

The following theorem [19] describes an important characterization of nonuniform learnability.

Theorem V.1. *Let $\mathscr{F}$ be a hypothesis set that can be written as a countable union of the individual hypothesis sets: $\mathscr{F} = \bigcup_{n\in\mathbb{N}} \mathscr{F}_n$, where each $\mathscr{F}_n$ exhibits a uniform convergence property. Then, $\mathscr{F}$ is nonuniformly learnable.*

Returning to our specific case, we can show the main theorem below.

Theorem V.2. *Let $\tilde{l}(\cdot)$ be Lipschitz continuous with respect to its arguments with L being its Lipschitz coefficient. Moreover, suppose $\mathcal{Z}$ is bounded. Then, the hypothesis set $\mathscr{F}_T^{(\nu)}$ defined in* (IV.2) *is nonuniformly learnable.*

B. Covering and bracketing numbers

Now, we define the concept of the covering number [7].

Definition V.4. *Let (X, d) be a metric space, and let $T \subset X$. We say that $T' \subset X$ is an ε-cover for T if, for all $x \in T$, there exists $y \in T'$ such that $d(x,y) \leq \varepsilon$. The ε-covering number of (X, d), denoted as $N(\varepsilon, T, d)$ is the size of the smallest ε-covering. The metric entropy ins the log covering number.*

Next, we define the bracketing number [7] [21].

Definition V.5. *Given two functions l and u, the bracket $[l, u]$ is the set of functions f which satisfy $l \leq f \leq u$. An ε-bracket is a bracket $[l, u]$ with $\|u - l\| < \varepsilon$. The bracketing number $N_{[]}(\varepsilon, \mathcal{F}, \|\cdot\|)$ is the minimum number of ε-brackets needed to cover $\mathcal{F}$.*

Later, we shall use the following theorem [21].

Theorem V.3. *let $\mathcal{F}$ be a class of measurable functions such that $N_{[]}(\varepsilon, \mathcal{F}, L_1(P)) < \infty$ for every $\varepsilon > 0$. Then, $\mathcal{F}$ forms a Glivenko-Cantelli class for P.*

C. *Proof of* Theorem V.2

To demonstrate Theorem V.2, we will introduce a sequence of hypothesis sets.

$$\mathscr{L}_T^{(\nu)}(n) \equiv \tilde{l} \circ \mathscr{F}_T^{(\nu)}(n)$$
$$\equiv \left\{ (\vec{\xi}, y) \longmapsto \tilde{l}\left(\int w_0(x) u(T, x; w_1, \vec{\xi}, \nu)\, \mathrm{d}x, y \right) \,\middle|\, \|w_1\|_{L_2(\mathcal{H}_T)} \leq n \right\} \; (n = 1, 2, \ldots).$$

Evidently, these sets form the following relationships.

$$\mathscr{F}_T^{(\nu)}(1) \subset \mathscr{F}_T^{(\nu)}(2) \subset \ldots,$$
$$\mathscr{F}_T^{(\nu)} = \bigcup_{n=1}^{\infty} \mathscr{F}_T^{(\nu)}(n) \quad \forall T, \nu > 0. \qquad \text{(V.1)}$$

Next, we demonstrate that each set $\mathscr{F}_T^{(\nu)}(n)$ in (V.1) satisfies uniform convergence property. Then, in virtue of Theorem V.1 , we can show the nonuniform learnability of $\mathscr{F}_T^{(\nu)}$. Let us present a known lemma [21] concerning the covering number $N(\cdot)$ and bracketing number $N_{[]}(\cdot)$.

Lemma V.1. *Let $\mathcal{F} = \{f_t \big| t \in \mathcal{T}\}$ be a class of functions defined on a set $\mathcal{X}$ satisfying Lipschitz continuity in the index parameter:*

$$\big|f_s(x) - f_t(x)\big| \leq d(s,t)F(x) \quad \forall x \in \mathcal{X},\ \forall s, t \in \mathcal{T}, \quad \text{(V.2)}$$

for some fixed function $F(\cdot)$, where $d(\cdot,\cdot)$ is a metric in the index space $\mathcal{T}$. Then, for any norm $\|\cdot\|$, $N_{[]}\big(2\varepsilon\|F\|, \mathcal{F}, \|\cdot\|\big) \leq N(\varepsilon, \mathcal{T}, d)$.

We also introduce the following lemma concerning the metric entropy of a set of functions.

Lemma V.2. *Let $M > 0$ and $B_M \equiv \{u \in H^1(I) \big| \|u\|_{H^1(I)} \leq M\}$. Then, B_M is relatively compact in $L_2(I)$ and satisfies*

$$\log N(\varepsilon, B_M, L_2(I)) \leq \frac{KM}{\varepsilon} \quad \forall \varepsilon > 0,$$

where K is a constant.

Proof. This lemma can be proven if we take $p = q = 2$ in Theorem 4.3.36 of [5], and note that the inclusion of function spaces $H^1(I) \subset B^{1,W}_{2\infty}(I)$ [5]. $\square$

Now, let us consider the set

$$\left\{ \vec{\xi} \longmapsto \tilde{l}\Big(\int_I w_0(x) u(T,x;w_1,\vec{\xi},\nu)\,\mathrm{d}x, y \Big) \right\}_{u(T,x;w_1,\vec{\xi},\nu)}$$

as $\mathcal{F}$, and the set $\{u(T,x;w_1,\vec{\xi},\nu)\}_{w_1}$ as $\mathcal{T}$ in Lemma V.1. Let us introduce $w_1^{(a)}, w_1^{(b)} \in L_2(\mathcal{H}_T)$, and a notation $\tilde{u}(t,x) \equiv u(t,x;w_1^{(a)},\vec{\xi},\nu) - u(t,x;w_1^{(b)},\vec{\xi},\nu)$. From the assumption, we have

$$\begin{aligned}
&\Big| \tilde{l}\Big(\int_I w_0^{(a)}(x) u(T,x;w_1^{(a)},\vec{\xi},\nu)\,\mathrm{d}x, y \Big) \\
&\qquad - \tilde{l}\Big(\int_I w_0(x)^{(b)} u(T,x;w_1^{(b)},\vec{\xi},\nu)\,\mathrm{d}x, y \Big) \Big| \\
&= \Big| \tilde{l}\Big(\int_I A^{\dagger}_{\vec{u}^{(a)}} \vec{y} u(T,x;w_1^{(a)},\vec{\xi},\nu)\,\mathrm{d}x, y \Big) \\
&\qquad - \tilde{l}\Big(\int_I A^{\dagger}_{\vec{u}^{(b)}} \vec{y} u(T,x;w_1^{(b)},\vec{\xi},\nu)\,\mathrm{d}x, y \Big) \Big| \\
&\leq L \Big| \int_I A^{\dagger}_{\vec{u}^{(a)}} \vec{y} u(T,x;w_1^{(a)},\vec{\xi},\nu)\,\mathrm{d}x \\
&\qquad - \int_I A^{\dagger}_{\vec{u}^{(b)}} \vec{y} u(T,x;w_1^{(b)},\vec{\xi},\nu)\,\mathrm{d}x \Big|.
\end{aligned}$$

The right-most-hand side is estimated above by

$$\begin{aligned}
&\Big| \int_I \Big(A^{\dagger}_{\vec{u}^{(a)}} \vec{y} - A^{\dagger}_{\vec{u}^{(b)}} \vec{y} \Big) u(T,x;w_1^{(a)},\vec{\xi},\nu)\,\mathrm{d}x \Big| \\
&\qquad + \Big| \int_I \Big(A^{\dagger}_{\vec{u}^{(a)}} \vec{y} - A^{\dagger}_{\vec{u}^{(b)}} \vec{y} \Big) u(T,x;w_1^{(b)},\vec{\xi},\nu)\,\mathrm{d}x \Big|
\end{aligned}$$

By noting that $A^{\dagger}_{\vec{u}^{(a)}} \vec{y} - A^{\dagger}_{\vec{u}^{(b)}} \vec{y} = \sum_{i=1}^m \tilde{u}(t,x;\vec{\xi_i}) y_i$, and using the boundedness of $\vec{y}$, we finally arrive at

$$\begin{aligned}
&\Big| \tilde{l}\Big(\int_I w_0^{(a)}(x) u(T,x;w_1^{(a)},\vec{\xi},\nu)\,\mathrm{d}x, y \Big) \\
&\qquad - \tilde{l}\Big(\int_I w_0^{(b)}(x) u(T,x;w_1^{(b)},\vec{\xi},\nu)\,\mathrm{d}x, y \Big) \Big| \\
&\qquad\qquad \leq c \|\tilde{u}(T)\|_{H^1} \|\vec{y}\|_{\mathbb{R}^m}.
\end{aligned}$$

But if we take $d(\cdot,\cdot)$ in Lemma V.1 as the norm of $H^1(I)$, this implies that we can apply Lemma V.1 to estimate $N_{[]}\big(2\varepsilon\|F\|, \mathcal{F}, \|\cdot\|\big)$ above.

Because the estimate of Lemma V.2 assures the finiteness of the right-hand side of Theorem V.3, this implies that $\mathcal{F}_T^{(\nu)}(n)$ satisfies the uniform convergence property for each n. In virtue of Theorem V.1, we complete the proof of Theorem V.2. $\square$

We have seen that under some conditions, $\mathscr{L}(n)$ forms a Glivenko-Cantelli class, and consequently, $\mathcal{F}_T^{(\nu)}(n)$ has a finite sample complexity, say, $m^{UC}_{\mathscr{F}_T^{(\nu)}(n)}(\epsilon,\delta)$. To examine nonuniform learnability of $\mathcal{F}_T^{(\nu)}$, let us consider

$$\varepsilon_n(m,\delta) = \min_{\varepsilon\in(0,1)} \big\{ m^{UC}_{\mathscr{F}_T^{(\nu)}(n)}(\epsilon,\delta) \leq m \big\}.$$

Then, it clearly holds that for each $n \in \mathbb{N}$,

$$\big| L_D(h) - L_S(h) \big| \leq \varepsilon_n(m,\delta) \quad \forall h \in \mathscr{F}_T^{(\nu)}(n).$$

In addition, if we consider a family of functions $w(n) : \mathbb{N} \to [0,1]$ that satisfies $\sum_{n=1}^{\infty} w(n) \leq 1$, we have an approach called *structural risk minimization* (SRM) [19]. The following result is known [19].

Theorem V.4. *Let $\mathscr{F}$ be a hypothesis class, such that $\mathscr{F} = \bigcup_n \mathscr{F}_n$, where each $\mathscr{F}_n$ has uniform convergence property with sample complexity $m^{UC}_{\mathscr{F}_n}$. Let $w : \mathbb{N} \to [0,1]$ be defined as $w(n) = 6/n^2\pi^2$. Then, $\mathscr{F}$ becomes nonuniformly learnable using the SRM scheme at a rate*

$$m^{NUC}_{\mathscr{F}}(\varepsilon,\delta,h) \leq m^{UC}_{\mathscr{F}_n}\Big(\frac{\varepsilon}{2}, \frac{6\delta}{(\pi n(h))^2} \Big).$$

Theorems V.2 to V.4, together with (V.1), guarantee that our PDE-based neural network has nonuniform learnability, and thus, can be used as a learner.

VI. Evaluation

We conducted some numerical experiments to evaluate the performance of our model using some datasets. Because the main focus of the present paper is the theoretical argument, this is the first example to check the effectiveness of our model.

1) Settings: In this experiment, we focused exclusively on regression. The proposed model was implemented using python 3.7 on a Windows Server 2019 (64bits), 12th Gen Intel (R) Core (TM) i7-12700, 2.11GHz, RAM 96.0GB.

In this experiment, we used the time difference $\triangle t = 5 \times 10^{-4}$ and a range of values for the number of temporal and spatial grids, denoted as N and L, respectively. At the output layer, we employed linear multiple regression scheme using statsmodels [22].

2) Datasets: Numerical simulations are conducted with 'California housing' [23] and 'Online News Popularity' [24] datasets, which are well-known benchmarks of regression. The first dataset contained 8 attributes of apartment houses in California, as well as the median value of the prices. The latter one contained 59 attributes of online news articles and the degree with which each article was shared. An overview of these datasets is presented in Table I. We standardized the whole dataset at first, and then split them into the training and test datasets. We employed 75% of the data for training. After the learning process using the training data, we measured the MSE (mean squared error) using the remaining test data. We also applied regressions with support vector machine (SVR), random forest (RFR), LightGBM, XGBoost, and linear multiple regression. In the multiple regression analysis, we added

TABLE I
OVERVIEW OF DATASETS

	California housing	Online News Popularity
Sample size (train)	15480	29733
Sample size (test)	5160	9911

no regularization terms. We applied the forward stepAIC to the variable selection process. We have also confirmed that the VIF (variance inflation factor) values are less than 10 over all selected explanatory variables, with which we have judged that there is no multicollinearity.

3) Results of experiments: Tables II and III show the MSE values of the training and test datasets (boldface indicates the best value for each indicator) under a range of combinations of parameters. The value of ν was set to 0.01. The performance of the proposed method was comparable to that of the existing methods (random forest regressor (RFR), support vector regressor (SVR) with RFB kernel, XGBoost, and LightGBM). It also seems that the proposed method does not overfit in the sense that the performance over the test datasets is not so deteriorated compared to that over the training dataset. Note that in the existing methods, we tuned the hyperparameters by using cross-validation and grid-search. From Tables II and III, we observe that the performance of our model varies depending on the values of $T(= \triangle t \times N)$.

TABLE II
RESULTS OF 'CALIFORNIA HOUSING' DATASET (AFTER STANDARDIZATION)

	(ν, N, L)	Training MSE / Test MSE
Proposed method	$(0.01, 2, 8)$	0.808 / 0.805
	$(0.01, 2, 24)$	0.336 / 0.360
	$(0.01, 2, 48)$	0.324 / 0.345
	$(0.01, 2, 72)$	0.314 / 0.321
	$(0.01, 5, 8)$	0.809 / 0.806
	$(0.01, 5, 24)$	0.338 / 0.346
	$(0.01, 10, 8)$	0.808 / 0.805
	$(0.01, 2, 16)$	0.758 / 0.788
	$(0.01, 30, 24)$	0.858 / 0.859
Existing methods	Multiple regression	0.521 / 0.541
	SVR	0.195 / 3.891
	RFR	**0.0258** / 0.469
	LightGBM	0.115 / 0.316
	XGBoost	0.0552 / **0.294**

VII. CONCLUSION

This study demonstrates the nonuniform learnability of our model when it is applied to a regression task. Moreover, we implemented our model on a computer and performed certain numerical experiments. It showed a comparable performance to that of the existing models, such as Rondom Forest, Support vector machine, LightGBM, and XGBoost. It was shown that the generalization performance could be adjusted by some parameters of the model. In the future, we will work on the extension of our method to multi-class classification problem. We will also consider the estimate on sample complexity, as well as more effective methods such as active learning. Regarding the possible future research issues, there is room for improvement in optimization procedure. We are planning to explore other optimization methods, such as Genetic Algorithm, Differential Evolution, LSHADE-SPACMA, and so forth. It is also important to discuss the PAC-Bayes perspective of the proposed model as well. Additionally, we intend to extend our PDE-based neural network to multidimensional Euclidean spaces. This is necessary when considering a GNN in which the elements are treated in the matrix form.

TABLE III
RESULTS OF 'ONLINE NEWS POPULARITY' DATASET (AFTER STANDARDIZATION)

	(ν, N, L)	Training MSE / Test MSE
Proposed method	$(0.01, 2, 8)$	0.858 / 0.860
	$(0.01, 2, 59)$	1.020 / **0.835**
	$(0.01, 5, 59)$	1.020 / 0.837
	$(0.01, 5, 118)$	1.013 / 0.844
	$(0.01, 5, 177)$	1.009 / 0.853
	$(0.01, 10, 59)$	1.020 / 0.837
Existing methods	Multiple regression	0.979 / 0.974
	SVR	0.591 / 0.947
	RFR	1.004 / 0.842
	LightGBM	**0.529** / 0.873
	XGBoost	0.815 / 0.841

REFERENCES

[1] Ricky T. Q. Chen, Y. Rubanova, J. Bettencourt and D. Duvenaud, Neural ordinary differential equations, Advances in Neural Information Processing Systems, **31** (2018).
[2] Honda, H., Sano,T., Nakamura,S., Ueno.M, Hanazawa,H. and Tuan,N.M.D., An ODE-based neural network with Bayesian optimization, JSIAM Letters 15 (2023), 101–104.
[3] H. Honda, On a partial differential equation based neural network, IEICE Communications Express, **10** (2021), 137–143.
[4] Honda, H. Universal approximation property of a continuous neural network based on a nonlinear diffusion equation. Adv Cont Discr Mod 2023, 43 (2023). 10.1186/s13662-023-03787-z
[5] E. Giné and R. Nickl, (2015). *Mathematical Foundations of Infinite-Dimensional Statistical Models.* Padstow Cornwall: Cambridge University Press.
[6] P. I. Frazier and W. Jialei, Bayesian Optimization for Materials Design, Eds. Lookman, Turab and Alexander, Francis J. and Rajan, Krishna, Information Science for Materials Discovery and Design, Springer International Publishing.
[7] R. M. Dudley (1999). *Uniform Central Limit Theorems.* Cambridge: Cambridge University Press.
[8] Jonus Mockus, Application of Bayesian approach to numerical methods of global and stochastic optimization, J. Global Optimization, **4** (1994), 347–365.
[9] P. I. Frazier, A Tutorial on Bayesian Optimization, arXiv (2018).
[10] B. Shahriari et al., Taking the Human Out of the Loop: A Review of Bayesian Optimization, Proceedings of the IEEE, **104** (2016), 148-175.
[11] R. Garnett et al., Bayesian optimization for sensor set selection, Proc. ACM/IEEE International Conference on Information Processing in Sensor Networks, (2010), 209–219.
[12] Andreas Look and Melih Kandemir, Differential Bayesian Neural Nets, ArXiv (2019), abs/1912.00796.
[13] R. Dandekar, et al., Bayesian Neural Ordinary Differential Equations, Advances in Neural Information Processing Systems, ArXiv (2022), abs/2012.07244.
[14] J. L. Lions (1971). *Optimal Control of Systems Governed by Partial Differential Equations.* Berlin, Heidelberg: Springer.
[15] M. Z. Nashed (1976). *Generalized Inverses and Applications.* Academic press, New York.

[16] W. E, J. Han and Q. Li, A mean-field optimal control formulation of deep learning, Res. Math. Sci., **6** (2019).
[17] GpyOpt, https://sheffieldml.github.io/GPyOpt/
[18] scikit-optimize, https://scikit-optimize.github.io/stable/
[19] S. Shai and B. Shai (2014). *Understanding Machine Learning.* Padstow Cornwall: Cambridge University Press.
[20] Burges, C.J. A Tutorial on Support Vector Machines for Pattern Recognition. Data Mining and Knowledge Discovery 2, 121–167 (1998).
[21] A. W. Vaart and J. A. Wellner, Weak Convergence and Empirical Processes, Springer New York, 1996.
[22] statsmodels, https://www.statsmodels.org/stable/index.html
[23] https://scikit-learn.org/stable/modules/generated/sklearn.datasets.fetch_california_housing.html
[24] Fernandes,Kelwin, Vinagre,Pedro, Cortez,Paulo, and Sernadela,Pedro. (2015). Online News Popularity. UCI Machine Learning Repository. https://doi.org/10.24432/C5NS3V.