King Fahd University of Petroleum and Minerals

College of Computing and Mathematics

Mathematics Department

Math 619-Project

Term 231

Pi-Deeponet Section

Project: Deep Learning Methods for Partial Differential Equations (PDEs)

Name	KFUPM ID
Hashim Al-Sadah	201578370
Abdulwahab Alghamdi	201734070
Hussain Al-Sinan	202205120

Instructor: Dr. Jamal Al-Smail

ABSTRACT

Physics Informed Deep Neural Operator (PI-DeepONet) is a new method of solving partial differential equations (PDEs) using deep learning. This method is based on the combination of the Physics Informed Neural Networks (PINNs) and the DeepONet. PINNs are a class of neural networks that are trained to solve PDEs by enforcing the PDEs as a loss function. DeepONet is a deep learning architecture that is used to solve PDEs by learning the solution operator. PI-DeepONet is a combination of these two methods, which allows for the solution of PDEs with less data and computational cost. This part will provide an overview of the PI-DeepONet method and its applications.

INTRODUCTION

In this section, we provide a concise overview of the DeepONet model architecture [1], focusing specifically on the learning of solution operators for parametric Partial Differential Equations (PDEs). The term "parametric PDEs" refers to PDE systems where certain parameters are permitted to vary within a specified range. These parameters could include factors such as the shape of the physical domain, initial or boundary conditions, coefficients (constant or variable), and source terms. To address such problems comprehensively, let (U, V, S) be a triplet of Banach spaces, and $N : U \times S \to V$ be a differential operator, either linear or nonlinear. We examine parametric PDEs of the form $N(u, s) = 0$ [1], where $u \in U$ represents the parameters (input functions), and $s \in S$ is the corresponding unknown solution to the PDE system. It is assumed that for any $u \in U$, there exists a unique solution $s = s(u) \in U$ to the equation $N(u, s) = 0$ (subject to appropriate initial and boundary conditions). Consequently, a solution operator $G : U \to U$ is defined as $G(u) = s(u)$. Following the original formulation by Lu et al. [1], we represent the solution map G using an unstacked DeepONet denoted as G_θ, where θ encompasses all trainable parameters of the DeepONet network. As depicted in Figure 1, the unstacked DeepONet comprises two separate neural networks, known as the "branch net" and "trunk net", respectively. The branch net takes u as input and produces a feature embedding $[b_1, b_2, ..., b_q]^T \in \mathbb{R}^q$ as output, where $u = [u(x_1), u(x_2), ..., u(x_m)]$ denotes a function $u \in U$ evaluated at a

set of fixed locations $\{x_i\}_{i=1}^m$. On the other hand, the trunk net accepts continuous coordinates y as inputs and generates a feature embedding $[t_1, t_2, ..., t_q]^T \in \mathbb{R}^q$ as output. The final output of the DeepONet is obtained by merging the outputs of the branch and trunk networks via a dot product. Specifically, the DeepONet prediction $G_\theta(u)(y)$ for a function u evaluated at y is expressed as:

$$G_\theta(u)(y) = \sum_{k=1}^{q} b_k(u(x_1), u(x_2), ..., u(x_m)) \cdot t_k(y), \tag{1}$$

where θ represents the collection of all trainable weight and bias parameters in the branch and trunk networks. These parameters are optimized by minimizing the mean square error loss given by:

$$L(\theta) = \frac{1}{NP} \sum_{i=1}^{N} \sum_{j=1}^{P} \left[\sum_{k=1}^{q} b_k(u^{(i)}(x_1), ..., u^{(i)}(x_m)) \cdot t_k(y_j^{(i)}) - G(u^{(i)})(y_j^{(i)}) \right]^2, \tag{2}$$

where $\{u^{(i)}\}_{i=1}^N$ denotes N separate input functions sampled from U. For each $u^{(i)}$, $\{y^{(i)}\}_{j=1}^P$ represents P locations in the domain of $G(u^{(i)})$, and $G(u^{(i)})(y_j^{(i)})$ is the corresponding output data evaluated at $y_j^{(i)}$. It's noteworthy that unlike the fixed sensor locations $\{x_i\}_{i=1}^m$, the locations of $\{y^{(i)}\}_{j=1}^P$ may vary for different i, thus enabling the construction of a continuous representation of the output functions $s \in S$.

References

[1] Sifan Wang, Hanwen Wang, and Paris Perdikaris. Learning the solution operator of parametric partial differential equations with physics-informed deeponets. *CoRR*, abs/2103.10974, 2021.