不确定性量化下的物理信息深度学习综合库

B-PINNs 项目报告

作者: Giulia Mescolini, Luca Sosta 导师: Prof. Andrea Manzoni, Prof. Stefano Pag

2025 年 11 月 24 日

摘要

本项目开发了一个新的库，实现了贝叶斯物理信息神经网络（B-PINNs）方法的多种变体。该框架能够利用神经网络解决物理问题，同时考虑来自偏微分方程的信息并量化预测的不确定性。

为了实施这一策略，我们从头开始设计了一个模块化且灵活的管道。在设计中，我们一方面希望将 B-PINN 框架中涉及的网络任务的特性反映到类的层次结构中，另一方面旨在提出一种结构，能够在其中插入针对不同任务的各种功能。

在发布的版本中，我们提供了整个流程各个方面的不同变体，从数据管理到训练算法。其中，我们实现了 Adam、哈密顿蒙特卡洛 (Hamiltonian Monte Carlo, HMC)、变分推断 (Variational Inference, VI) 和斯坦变分梯度下降 (Stein Variational Gradient Descent, SVGD)。

该库的设计也预留了进一步扩展的空间，因为代表不同组件的模块被设计为由少量特定案例的行为定义，并且对功能没有严格的限制。这一点在实现具有不同例程和结构或不同问题的算法时得到了验证。

最后，我们展示了一系列结果，以突显该方法和库的主要特征：算法比较、在神经网络中引入 PDE 残差、高维度的可移植性以及通过微调模型参数可以获得的结果质量。

关键词： B-PINNs，不确定性量化，科学学习

目录

1 简介 (Introduction)

1.1 项目概述

机器学习技术在科学计算中的整合如今变得越来越流行。一个具有挑战性的场景是不确定性量化，神经网络的高效性有助于克服基于 PDE 求解器的传统方法的计算成本。

贝叶斯物理信息神经网络（B-PINNs）代表了一种解决此类问题的方法，这不仅归功于在学习过程中包含了微分方程的残差，还归功于能够重建网络输出分布而非单一预测的训练算法。

该方法是物理信息神经网络（PINNs）的演变，PINNs 是一种用于解决涉及偏微分方程问题的深度学习框架。B-PINNs 从贝叶斯统计理论中汲取灵感，增强了 PINNs 方法并使得引入不确定性量化成为可能。

在这个项目中，我们实现了一个用于贝叶斯物理信息机器学习的库，旨在实现模块化、灵活性并支持插件。通常只能找到应用于特定微分问题的特定贝叶斯训练方法的实现；因此，我们设定的目标是开发一个具有广泛功能的库。

1.2 报告结构

本报告介绍了拟议实现的理论基础：

- **第 2 章方法**：解释 B-PINN 的构建模块，从神经网络基础到 PINNs 和 BNNs 的理论。
- **第 3 章算法**：深入探讨实现的非确定性训练算法：HMC, VI, SVGD。
- **第 4 章代码概述**：介绍环境、库的选择以及面向对象编程的特性。
- **第 5 章源代码**：详细说明源代码结构。
- **第 6 章结果**：展示库在不同数据集和训练配置下的应用结果。
- **第 7 章结论**：总结项目并提出未来发展建议。

2 方法 (Methods)

2.1 神经网络概述

人工神经网络 (NN) 是受生物神经网络启发的计算系统。在深度学习框架下，它们能够处理复杂信息。主要结构包括输入层、隐藏层（特征学习）和输出层。

$$a_i^{(l)} = \phi\left(\sum_{j=1}^{N_{l-1}} w_{j,i} a_j^{(l-1)} + b_i^{(l)}\right) \tag{1}$$

常见的激活函数包括 Sigmoid, Tanh, ReLU, Leaky ReLU 和 Swish。本项目采用了全连接前馈神经网络。

2.1.1 反向传播与优化器

学习过程通过反向传播算法计算损失函数的梯度，并更新权重。常见的优化器包括：

- **梯度下降 (GD)**：沿着梯度的反方向更新。
- **随机梯度下降 (SGD)**：使用小批量 (batch) 数据进行更新。
- **Adam**：一种自适应学习率方法，结合了 AdaGrad 和 RMSProp 的优点。

2.2 物理信息神经网络 (PINNs)

PINNs 是训练用于解决受微分方程控制的监督学习任务的神经网络。

$$\mathcal{N}(u(x,t);\lambda) = f(x,t) \quad \text{in } \Omega \times [0,T] \tag{2}$$

PINN 的损失函数包含数据拟合项和 PDE 残差项：

$$LOSS = MSE_{data} + MSE_{residual} \tag{3}$$

2.3 贝叶斯神经网络 (BNNs)

BNNs 将贝叶斯推理与神经网络结合，引入不确定性量化。核心思想是用概率分布替换确定的网络参数 θ。根据贝叶斯定理：

$$p(\theta|D) \propto p(D|\theta)p(\theta) \tag{4}$$

其中 $p(\theta|D)$ 是后验分布，$p(D|\theta)$ 是似然，$p(\theta)$ 是先验。

2.4 贝叶斯物理信息神经网络 (B-PINNs)

B-PINNs 将 PINNs 集成到贝叶斯框架中。PDE 约束作为似然项的附加部分出现：

$$p(\theta|D,R) \propto p(D,R|\theta)p(\theta) \tag{5}$$

其中 R 代表 PDE 残差。

3 算法 (Algorithms)

3.1 哈密顿蒙特卡洛 (HMC)

HMC 是一种马尔可夫链蒙特卡洛 (MCMC) 方法，利用哈密顿动力学来生成符合目标分布的样本。它引入辅助动量变量 $\mathbf{r}$，构造哈密顿量 $H(\theta,\mathbf{r}) = U(\theta) + \frac{1}{2}\mathbf{r}^T M^{-1}\mathbf{r}$，其中 $U(\theta) = -\log(p(\theta|D))$。算法包含 Leap-Frog 积分步骤和 Metropolis-Hastings 接受-拒绝步骤。

3.2 变分推断 (VI)

VI 旨在通过在参数族中寻找最佳近似来逼近后验分布，从而将采样问题转化为确定性优化问题。目标是最小化 KL 散度 $D_{KL}(Q(\theta|\zeta)||P(\theta|D))$。

3.3 斯坦变分梯度下降 (SVGD)

SVGD 是一种基于粒子的变分推断方法。它使用一组粒子 $\theta_{i}{}_{i=1}^{N}$ 来近似目标分布，并通过最小化 KL 散度的函数梯度流来迭代更新粒子。更新规则涉及核函数 $k(\theta,\theta')$（如 RBF 核），使得粒子在向高概率区域移动的同时保持多样性（排斥力）。

4 代码概述 (Code Overview)

4.1 工作环境

代码基于 Python 3.10 开发，使用 ‘virtualenv‘ 管理环境。依赖库包括：

- **NumPy, SciPy**: 科学计算。
- **Matplotlib**: 可视化。
- **TensorFlow 2.9.1**: 深度学习框架和自动微分。

4.2 面向对象特性

代码广泛使用了 Python 的 OOP 特性，如继承（单继承和多继承）、抽象基类 (ABC)、数据类 (dataclass)、迭代器和属性装饰器 (@property)，以构建模块化和可扩展的架构。

4.3 仓库结构

- `config/`: 包含.json 配置文件。
- `data/`: 存储生成的数据集 (.npy)。
- `outs/`: 存储实验结果（日志、图表、参数）。
- `src/`: 源代码目录。

5 源代码 (Source Code)

5.1 主要模块

- **main.py**: 主执行脚本，负责参数处理、数据加载/生成、模型构建、训练、评估和绘图。
- **setup/**: 参数处理 (`Param`) 和数据生成 (`DataGenerator`)。
- **equations/**: 定义微分算子 (`Operators`) 和 PDE 问题（如 `Laplace`, `Oscillator`）。
- **networks/**: 定义网络架构。
  - `CoreNN`: 基础神经网络。
  - `PhysNN`: 引入物理信息。
  - `BayesNN`: 贝叶斯功能的封装。
  - `Theta`: 处理网络参数的代数运算。
- **algorithms/**: 实现训练算法。
  - `Algorithm`: 抽象基类。
  - `ADAM`, `HMC`, `VI`, `SVGD`: 具体实现。
- **postprocessing/**: 结果存储 (`Storage`) 和绘图 (`Plotter`)。

6 结果 (Results)

6.1 回归问题

使用 Adam, HMC, SVGD, VI 对简单函数（如 $u(x) = \cos(8x)$）进行回归。HMC 和 SVGD 提供了合理的置信区间。

6.2 阻尼谐振子问题

比较了标准 NN、PINN 和 B-PINN。在数据缺失的区域，物理信息的引入显著改善了预测，B-PINN 提供了不确定性估计。

6.3 高维域

展示了 2D 拉普拉斯问题的求解结果，验证了代码在多维情况下的可移植性。

6.4 HMC 展示

对 HMC 算法进行了深入的参数微调，展示了其在回归和谐振子问题上的高质量结果，特别是在噪声数据下的鲁棒性。

7 结论 (Conclusions)

本项目成功实现了一个用于贝叶斯物理信息深度学习的综合库。

- **模块化设计**：基于 OOP 的架构使得扩展新算法和新问题变得容易。
- **多样化算法**：实现了从确定性 (Adam) 到多种贝叶斯 (HMC, VI, SVGD) 的训练方法。
- **物理信息**：验证了 PDE 残差在数据稀缺情况下对模型性能的提升。
- **不确定性量化**：提供了对预测结果置信度的有效估计。

未来工作可以包括参数估计问题的扩展（逆问题）、更高级的超参数调优自动化以及对高维数据可视化的改进。

A 安装指南

```
# 创建虚拟环境
virtualenv venv
source venv/bin/activate

# 安装依赖
pip install -r requirements.txt
```