

极限学习机和一类深度学习方法的对比较研究

徐翊铭, 范馨月

贵州大学数学与统计学院, 贵州 贵阳

收稿日期: 2022年11月21日; 录用日期: 2022年12月15日; 发布日期: 2022年12月23日

摘要

本文针对神经网络在微分方程求解中的先验估计问题, 将极限学习机(Extreme Learning Machine, ELM)嵌入粒子群优化算法和一类使用梯度更新的深度学习方法进行比较, 将他们应用于一类微分方程求解并进行对比。与使用梯度信息优化损失函数的深度学习方法不同, ELM使用Moore-Penrose广义逆替代了梯度信息更新权重以达到损失函数最小化。本文通过实验证明了ELM对于方程的区间没有要求, 训练时间短, 但劣势是若解是非连续的, 对间断点类型有要求等, 而此类深度学习算法加深了网络层数, 优势是学习到更抽象的特征, 且解可以是非连续的, 劣势对方程区间有较严格要求, 否则梯度消失, 并通过实验结果进行了验证, 且训练时间长此类深度学习算法有时会伴随着一些剪枝算法压缩网络。本文的结论可以用于先验的算法选择, 提高求解效率。

关键词

物理信息神经网络, 极限学习机, 神经网络, 深度学习, 积分 - 微分方程, 粒子群优化算法

A Comparative Study of Extreme Learning Machine and a Class of Deep Learning Methods

Yiming Xu, Xinyue Fan

School of Mathematics and Statistics, Guizhou University, Guiyang Guizhou

Received: Nov. 21st, 2022; accepted: Dec. 15th, 2022; published: Dec. 23rd, 2022

Abstract

In this paper, aiming at the priori estimation problem of neural network in differential equation solving, the Extreme Learning Machine (ELM) was embedded in particle swarm optimization algo-

rithm and a class of deep learning algorithms using gradient update are compared, and they are applied to a class of differential equation solving and compared. Unlike the deep learning algorithms using gradient information to optimize the loss function, ELM used Moore-Penrose generalized inverse instead of gradient information to update the weight to minimize the loss function. This paper proved through experiments that ELM has no requirements for the interval of the equation, and the training time is short, but the disadvantages are that if the solution is discontinuous, it has requirements for the type of breakpoints, etc. This kind of deep learning algorithm deepens the number of network layers, and the advantages are learning more abstract features, and the solution can be discontinuous. While the disadvantages are strict requirements for the interval of the equation, otherwise the gradient is disappear, which is verified by the experimental results, and the training time is long. Such deep learning algorithms sometimes are accompanied by some pruning algorithms to compress the network. The conclusion of this paper can be used for prior algorithm selection to improve the efficiency of solution.

Keywords

Physical-Information Neural Network, Extreme Learning Machine, Neural Network, Deep Learning, Integro-Differential Equation, Particle Swarm Optimization Algorithm

Copyright © 2022 by author(s) and Hans Publishers Inc.

This work is licensed under the Creative Commons Attribution International License (CC BY 4.0).

<http://creativecommons.org/licenses/by/4.0/>



Open Access

1. 引言

近年来,随着计算能力的提高与人工智能的飞速发展,神经网络被广泛应用于各个领域,包括用于数值求解各类微分方程[1] [2] [3]。与有限差分法、有限元法等经典的数值方法不同,神经网络在高维偏微分方程数值求解问题上逐渐被使用。但目前没有任何一个理论,可以做一个先验判断,选择怎样的网络结构能达到良好的求解效果,这将会成为该领域重要的研究内容之一。解此类问题的神经网络又可以粗略地分为深度学习[4] [5] [6]和单隐层网络[7] [8]。

Huangguangbin 教授团队提出了极限学习机算法(ELM) [9],此算法表现出求解各种微分方程优异能力[10] [11],它是基于一种离线的单隐层前馈神经网络,输出权重通过 Moore-Penrose 广义逆运算获得。该算法不仅训练速度快,避免了深度学习的梯度消失以及爆炸问题从而得到广泛应用。但对方程的解光滑性要求较高,需要解是连续的。

而深度学习大多根据梯度信息来更新权重,优化损失函数以此来求解方程,以美国布朗大学 Karniadakis 教授团队开发出物理信息神经网络为代表。此方法凭借着优秀的性能被广泛地应用于各类微分方程问题,他们通过将蕴藏在小样本数据中包含由物理定律等信息编码到深度网络中,提出了一种新的深度学习算法——内嵌物理机理神经网络[12] [13] (Physics-informed neural networks, PINNs)。所以 PINNs 通常受到数据中包含的物理定律的约束,可以通过这些物理规律建模成时间相关的非线性偏微分方程。这样就可以利用 PINNs 发展出一类新的偏微分方程数值求解方法。该网络也可以有效地求解各类微分方程[14] [15]。

近年来,一些研究人员将两种神经网络算法进行比对[16],得出一些结论帮助人们根据解的类型选择合适的网络结构。本文针对同一类方程,对比网络的深浅,证明了网络深浅的适用性,ELM 使用 Moore-Penrose 广义逆替代了梯度信息更新权重以达到损失函数最小化,PINNs 则直接使用梯度信息优化

权重使损失函数最小。虽然深度学习是神经网络发展的主流算法,但随着网络越深,所面临的问题变得越复杂,如:难以优化,训练时间长,内存空间大等。故本文对有连续解且求解区间范围较小的微分方程进行了理论证明以及实验,ELM 优于一类使用梯度更新权重的深度学习算法。

2. 相关工作

2.1. 物理信息神经网络

考虑积分 - 微分方程的解的问题((1)~(2)):

$$f\left(x; \frac{du}{dx}; \frac{d^2u}{dx^2}; \dots; \int_0^x u(x, q) dq; \lambda\right) = 0, x \in \Omega \quad (1)$$

$$B(u, x) = 0, x \in \partial\Omega \quad (2)$$

若积分项的参数为 0,那么((1)~(2))转换为微分方程,其中(2)为 Dirichlet、Neumann、Robin 或周期边界条件,对于时间相关问题,将时间视为的一个特殊分量,故包含时域。边值条件(2)可以简单地视为时空域上 Dirichlet 边界条件的一种特殊类型[13];若微分项的参数为 0,那么((1)~(2))转换为积分方程。

在 PINNs 中,首先构造一个神经网络,让输出 $\hat{u}(x; \theta)$ 作为解 $u(x)$ 的近似,让 $N^L(x): \mathbb{R}^{d_{in}} \rightarrow \mathbb{R}^{d_{out}}$ 是 L 层神经网络,神经网络解输入为 x ,并且输出一个与 $u(x)$ 维数相同的向量。其中, $\theta = \{W^l, b^l\}_{1 \leq l \leq L}$ 是神经网络中所有权重矩阵和偏置向量的集合。可以通过链式法则使用 AD [17]来求取微分,获得 $\hat{u}(x; \theta)$ 关于其输入 x 的导数。

其次,需要约束神经网络的输出 $\hat{u}(x; \theta)$ 以满足积分 - 微分方程和边界条件施加的物理条件。在训练中,我们取一些分散点(例如,域中随机分布的点或聚集点),分成两组训练点,内点 $T_f = \{x_1, x_2, \dots, x_N\}, T_f \in \Omega$,边界点 $T_b = \{x_1, x_2, \dots, x_M\}, T_b \in \partial\Omega$,为了衡量神经网络输出,即微分方程的解 $\hat{u}(x; \theta)$ 和约束之间的差异,定义损失函数为方程和边界条件残差 L2 范数加权和(采用均方误差(mean square error, MSE)) [13]:

$$L(\theta; T) = w_f L_f(\theta; T_f) + w_b L_b(\theta; T_b), \quad (3)$$

其中

$$L_f(\theta; T_f) = \frac{1}{|T_f|} \sum_{x \in T_f} \left\| f\left(x; \frac{d\hat{u}}{dx}; \frac{d^2\hat{u}}{dx^2}; \dots; \int_0^x \hat{u}(x, s) ds; \lambda\right) \right\|, \quad (4)$$

$$L_b(\theta; T_b) = \frac{1}{|T_b|} \sum_{x \in T_b} \|B(\hat{u}, x)\|_2^2. \quad (5)$$

通过最小化损失函数训练神经网络,找出最优参数,即

$$\theta^* = \arg \min L(\theta; \tau), \quad (6)$$

利用训练好的神经网络模型,可以求解方程在任意时间和位置解。

2.2. 改进的极限学习机

对于此积分 - 微分方程((7)~(8))

$$\psi'(x) = \frac{\lambda}{c} \psi(x) - \frac{\lambda}{c} \int_0^x \psi(x-t) dF(t) - \frac{\lambda}{c} [1 - F(x)], x \in [0, c] \quad (7)$$

$$\psi(0) = \frac{1}{1+\theta} \quad (8)$$

搭建三角神经网络(TNN) [17]对方程求解, 对于 M 个输入, 每个分量是 $x_j, j=1, \dots, M$, 有 $2N$ 个隐层神经元。基于 ELM 设计的神经网络输出如下(采用正弦三角函数作为激活函数, 即公式(9)~(11)):

$$\phi(x) = \begin{cases} \sin\left(\frac{k\pi}{q}x\right), & k \leq N, \\ \sin\left(\frac{k\pi}{q}x + \frac{\pi}{2}\right), & k > N, k = 1, 2, \dots, 2N. \end{cases} \quad (9)$$

构造试解:

$$\psi(x_j) = \sum_{i=1}^{2N} w_i \phi_i(a_j x_j + b_j), j = 1, \dots, M, \quad (10)$$

初值:

$$\psi(0) = \sum_{i=1}^{2N} w_i \phi_i(b_j) = \frac{1}{1+\theta} \quad (11)$$

将(9) (10)带入(7), 得到合并同类项得, 改写成矩阵形式为:

$$WH_1 = Y_1 \quad (12)$$

通常无法获得定积分 $\int_0^x \sum_{i=1}^{2N} \phi_i(a_j(x_j - x) + b_j) dF(x)$ 的原函数表达式, 因此我们使用复化辛普森求积公式来获得数值积分, 如下所示:

$$\begin{aligned} I &= \int_a^b f(x) \approx S_n = \frac{h}{6} \sum_{k=0}^{n-1} [f(x_k) + 4f(x_{k+1/2}) + f(x_{k+1})] \\ &= \frac{h}{6} \left[f(a) + 4 \sum_{k=0}^{n-1} f(x_{k+1/2}) + 2 \sum_{k=1}^{n-1} f(x_k) + f(b) \right] \end{aligned} \quad (13)$$

本文方程中只有一个初值条件。添加条件(11)最终得到(14):

$$WH = Y \quad (14)$$

其中 $H = [H_1, H_0]_{2N \times (M+1)}$, $W = [w_i]_{1 \times 2N}$, $Y = [Y_1, Y_0]_{(M+1) \times 1}^T$, W 为隐藏层的权重矩阵, Y 为输出, 即是每个输入点的数值解。

ELM 采用最小二乘法直接计算 W , 使其误差最小化, 代替传统梯度下降算法的重复迭代, 计算公式如下:

$$W^* = \arg \min_W \|HW - Y\|^2 = H^+ Y, \quad (15)$$

其中, H^+ 是矩阵 H 的 Moore-Penrose 广义逆, $H^+ = (H^T H)^{-1} H^T$ 。

注意到 q 是一个需要确定的参数, 有必要确定函数中的三角激活函数参数 q , 该 q 值的选择直接影响解的精度, 在 Zhou [18] 的论文中没有给定 q 的选取方法, 本文使用经常被用于选取最优参数的粒子群优化算法[19], 本文称之为粒子-三角神经网络(PSO-TNN)。我们使用粒子群优化算法最小化 MSE(也可以选择相同类型的评估误差指标)来选取 q , 从而实现三角激活函数的优化。

2.3. 理论证明

定理 1: 对于物理信息神经网络

i) 若选用 sigmoid/tanh 激活函数, 对于此积分微分方程((7)~(8)), 权重更新梯度为 0 向量。

ii) 若选用 relu 激活函数, 对于此积分微分方程((7)~(8)), 定义域是恒大于 0 的, 由 $\text{relu}(x) = \max\{x, 0\}$, 仅对输入点做线性加权, 方程的解一般为非线性, 若用线性拟合非线性误差会很大。

i) 证明:

进行预定义, 由于 $\lim_{x \rightarrow \infty} \text{sigmoid}(x) = 1$, 机器精度只达到 10^{-16} , 故定义若有 $|a-d| \leq 10^{-16}$, 那么认为 $a = d$ 。

对于积分微分方程(7-8), 积分项 $\int_0^x \psi(x-t) dF(t)$ 采用 $n=20$ 的 Gaussian-Legendre 求积式公式如下:

$$\int_0^x \psi(x-t) dF(t) \approx \sum_{i=1}^n w_i \psi(x-t(x_i)) F'(t(x_i)), \quad (16)$$

其中 $t(x_i)$ 是 Legendre 多项式的零点, $x \in [0, c]$, 采用物理信息神经网络求解该方程, 取点 N 个, $x_i, i=1, \dots, N$, 初始化方式将权重矩阵 W 置为 1, 偏置向量 \vec{b} 置为 0 向量。

网络结构为一层隐藏层(为了证明方便, 与多层隐藏层网络结构相似), 神经网络解为

$$y_i = \sum_{j=1}^M \psi(w_{ij} x_i + b_j)。$$

首先, 将区间分为 $[0, a]$ 和 $[a, c]$, 其中 $a = \inf\{a_1, |\text{sigmoid}(a_1) - 1| \leq 10^{-16} \cap a_1 \in [0, c]\}$, 且 $c \rightarrow \infty$ 。故 $\forall x \in [a, c]$, 都有 $\text{sigmoid}(x) = 1$ 。

根据[13]将损失函数写出来有

$$\text{MSE}(\vec{x}, W, \vec{b}) = \text{MSE}_c(0, W, \vec{b}) + \text{MSE}_r(\vec{x}, W, \vec{b}) \quad (17)$$

其中,

$$\begin{aligned} \text{MSE}_r(\vec{x}, W, \vec{b}) &= \frac{1}{N} \sum_{i=1}^N \left| \psi'(x_i) - \frac{\lambda}{c} \psi(x_i) + \frac{\lambda}{c} \int_0^{x_i} \psi(x_i-t) dF(t) + \frac{\lambda}{c} [1 - F(x_i)] \right|^2 \\ \text{MSE}_c(0, W, \vec{b}) &= \left| \psi(0) - \frac{1}{1+\theta} \right|^2 \end{aligned} \quad (18)$$

根据损失函数(17)~(18), 写出梯度如下

$$g_{t-1}^{ik} = \frac{\partial \text{MSE}(\vec{x}, W, \vec{b})}{\partial w_{ik}} = \frac{2}{N} \left| x \psi(x_i) (1 - \psi(x_i))^2 - \frac{\lambda}{c} \psi(x_i)^2 (1 - \psi(x_i)) \right|, \quad (19)$$

其中 i 和 k 代表第 k 层的第 i 个神经元, $t-1$ 代表第 $t-1$ 次更新的梯度, 现有 s 个点 $x_{s_i} \in [a, c]$, $i=1, \dots, s$, 且 $W = 1, \vec{b} = 0$, 有 $\psi(x_{s_i}) = 1$, 代回(19)得到 $g_{t-1}^{ik} = 0$, 使用 ADAM 优化算法[20], 参数更新如下:

$$w_t = w_{t-1} - \frac{\eta}{\sqrt{\hat{v}_t} + \varepsilon} \hat{m}_t \quad (20)$$

其中 $\hat{m}_t = \frac{m_t}{1 - \beta_1}$, $\hat{v}_t = \frac{v_t}{1 - \beta_2}$; $m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_{t-1}$, $v_t = g_0^2$, $v_t = \beta_2 v_{t-1} + (1 - \beta_2) (g_{t-1})^2$; 通常 $\beta_1 = 0.9$,

$\beta_2 = 0.999$, $\varepsilon = 10^{-8}$, η 为步长, 可取 $10^{-1} \sim 10^{-3}$ 。再将 $g_{t-1}^k = 0$ 带入(20), 得到 $w_t = w_{t-1}$, 即梯度消失。难以优化, 权重更新停止, 导致了物理信息神经网络解的误差很大。

定理 2: 若微分方程的解 $y(x)$ 是间断的, 以间断点为界写成分段函数, 误差由分段函数的子函数距离空间衡量。

证明: 采用三角激活函数, 构造微分方程试解 $\psi(x_j) = \sum_{i=1}^{2N} w_i \phi_i(a_j x_j + b_j), j = 1, \dots, M$ 。其中微分方程解空间的基为 $\varphi_1, \varphi_2, \dots, \varphi_n$, 因三角函数具有正交性, 它的基可由三角函数构成。可将不连续函数写为分段函数。那么由分段点, 写为两个及两个以上函数, 以两个为例进行证明:

$$\text{将解 } y(x) \text{ 改写为 } y(x) = \begin{cases} L(x), & x > a \\ l(x), & x \leq a \end{cases}。$$

1) 若 $L(x)$ 和 $l(x)$ 都为有界函数。那么分别用三角函数作为基函数 $L(x) = \sum_{i=0}^{\infty} \alpha_i \phi_i(x)$ 和

$l(x) = \sum_{k=0}^{\infty} \beta_k \phi_k(x)$, 其中 $\|\alpha_i\| < \infty$, $\|\beta_k\| < \infty$, $i, k = 1, \dots, \infty$ 。那么误差

$$\begin{aligned} \text{erro}(x) &= \left\| \sum_{i=0}^{\infty} \alpha_i \phi_i(x) - \sum_{k=0}^{\infty} \beta_k \phi_k(x) \right\| \\ &\leq \sum_{i=0}^{\infty} \|\alpha_i \phi_i(x) - \beta_i \phi_i(x)\| \\ &= \sum_{i=0}^{\infty} \|(\alpha_i - \beta_i) \phi_i(x)\| \\ &\leq \sum_{i=0}^{\infty} \max_i (\alpha_i - \beta_i) \|\phi_i(x)\| \\ &\leq \max_i (\alpha_i - \beta_i) \\ &\leq C \end{aligned}$$

2) 若 $L(x)$ 和 $l(x)$ 至少有一个不存在或者无界函数, 那么至少有一个不能使用三角函数作基函数表示。

综上, 若这几个函数空间皆为有界函数, 那么误差是有限数。若有一方为 ∞ 或不存在, 那么误差是无限或不存在。

推论 1: 若解函数的间断点为第一类不连续点和第三类间断点的在间断点有定义的情况, 那么误差为有限的, 可以使用 ELM 求解方程; 若为第二类间断点和第三类间断点的在间断点处无定义的情况, 那么误差是无限或不存在, 不可使用 ELM 求解方程。

推论 2: 对于区间是 $[a, b]$ 的微分方程。

i) 当使用 sigmoid/tanh 激活函数, $|b| \gg 10$ 或 $|a| \gg 10$ 时, 且解是连续的不适于使用梯度信息优化的深度学习的使用求解微分方程。

ii) 当使用 relu 激活函数, 当 $ab > 0$ 或 $|a| \gg |b|$ 或 $|b| \gg |a|$ 时, 且解是连续的不适于使用梯度信息优化的深度学习的求解微分方程。

iii) 当解是连续的, ELM 适用于求解此类微分方程。

3. 实验测评

1) 这里以数值实验来验证的定理 1 i)。

2) 使用 PSO-TNN 和 TNN 以及 PINNs, 求解索赔额服从 Pareto 分布的经典风险模型的破产概率问题, 此问题被证明为以下积分 - 微分方程[21] [22]:

$$\begin{cases} \psi'(x) = \frac{\lambda}{c}\psi(x) - \frac{\lambda}{c}\int_0^x \psi(x-t)dF(t) - \frac{\lambda}{c}[1-F(x)], x \in [0, 25000] \\ \psi(0) = \frac{1}{1+\theta} \end{cases}$$

其中 $c = 600, \lambda = 1, \beta = 3, \theta = 0.2, \gamma = 1000, F(x) = 1 - \frac{\gamma^\beta}{\gamma + x}$, 该问题没有解析解, 但是已知一些点的解, 如表 1。

Table 1. Exact solutions of known points
表 1. 已知准确解的点

x	$\psi(x)$
100	0.807
500	0.724
1000	0.645
5000	0.313
10,000	0.148
25,000	0.025

采用神经元个数 $N = 12$ 的 PSO-TNN 和 TNN 以及神经元个数 $N = 50$ 和层数 $L = 3$ 的 PINNs 求解该方程, 样本集为, $x_1 = 0, x_2 = 100, x_{i+2} = \frac{25000i}{50}, i = 1, \dots, 50$ 数值实验结果如下:

表 2 和图 1 描述了此方程的精确解、PSO-TNN 和 TNN 以及 PINNs 的数值解及其相应的误差。由表 2 中数据可知, 对于已知点, PSO-TNN 的 MSE 为 $8.91703e-08$, 由[17]中的 TNN 获得的 MSE 约为 $7.1372e-07$, 而由 PINNs 获得的 MSE 约为 $2.2680e-01$ 。对于未知点, PSO-TNN 和 TNN 的数值解非常相似。因此, PSO-TNN 的解非常接近精确解。从训练时间上, PINNs 时间为 1293.0747 s, 而 PSO-TNN 仅需要 2.3057240 s。故对于定义域区间范围较大的方程, 从训练时间还是误差, ELM 都优于深度学习。

若对于此方程的区间进行截断, 只取样本集为, $x \in [0, 100]$, 数值解如表三。由表 2 可知, 当方程自变量的区间右端点的值较小时, PINNs 可以把方程的解学习得较好, MSE 约为 $2.3297e-07$, 可以较好地近似该方程的解。

Table 2. Error and time of three algorithms in different intervals under Pareto distribution
表 2. Pareto 分布下不同区间三种算法的误差以及时间

x	PSO-TNN (12*1)		PINNs (50*3)		TNN (12*1)	
	MSE	t/s	MSE	t/s	MSE	t/s
[0, 100]	-	-	$2.3297e-07$	1166.9992	-	-
[0, 25,000]	$8.91703e-08$	2.3057240	$2.2680e-01$	1293.0747	$7.1372e-07$	2.1427120

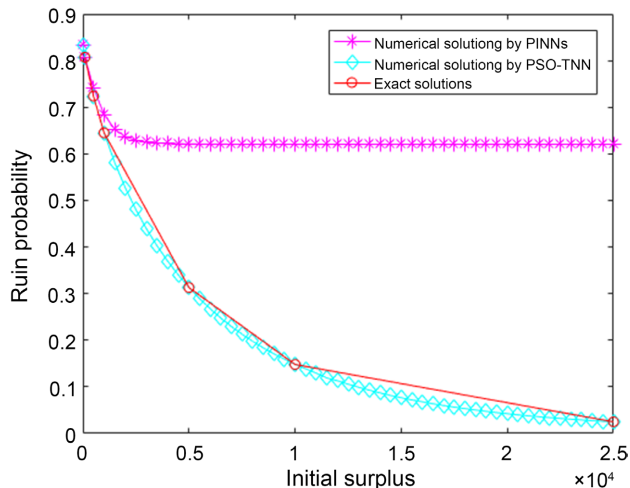


Figure 1. Numerical and analytical solutions by PINNs, PSO-TNN in (1)
图 1. (1)的 PINNs, PSO-TNN 数值解以及准确解

2) 使用 PSO-TNN 和 TNN 以及 PINNs, 求解索赔额服从指数分布下的经典风险模型的破产概率问题, 此问题被证明为以下积分 - 微分方程[21] [22]:

$$\begin{cases} \psi'(x) = \frac{\lambda}{c}\psi(x) - \frac{\lambda}{c}\int\psi(x-t)dF(t) - \frac{\lambda}{c}[1-F(x)], x \in [0,10] \\ \psi(0) = \frac{1}{1+\theta} \end{cases}$$

其中 $c=3, \lambda=4, \beta=2, \theta = \frac{c\beta}{\lambda} - 1, F(x) = 1 - e^{-\beta x}$ 时, 该问题的解析解为 $\psi(x) = \frac{1}{1+\theta} e^{-\frac{\theta\beta}{1+\theta}x}$, 先采用神经元个数 $N=12$ 的 PSO-TNN 和 TNN 以及神经元个数 $N=50$ 和层数 $L=3$ 的 PINNs 求解该方程, 其后增加神经个数再进行比对, 样本为: $x_i = \frac{10}{20}i, i=0, \dots, 20$ 数值实验结果如下:

表 3 和图 2 描述了此方程的精确解、PSO-TNN 和 TNN 以及 PINNs 的数值解及其相应误差。由表 3 中数据可知, PSO-TNN 的 MSE 为 $1.6967e-10$, 由[17]中的 TNN 获得的 MSE 约为 $4.1665e-06$, 而由 PINNs 获得的均方误差约为 $6.8827e-08$ 。虽然 PINN 同样优于原始 TNN, 但训练时间远长于 PSO-TNN 和 TNN。虽然 PINN 同样优于原始 TNN, 但训练时间远长于 PSO-TNN 和 TNN。显然, PSO-TNN 优于 TNN 和 PINNs。PSO-TNN 的解可以非常接近精确解。从训练时间上, PINNs 时间 1124.6291 s, 而 PSO-TNN 仅需要 0.0406790 s。故对于定义域区间范围较大的方程, 从训练时间还是误差, ELM 都优于深度学习。与(1)进行比较, 发现同类型的积分 - 微分方程, 当区间不同, 对于求解方法的网络结构是大相径庭的。

Table 3. Error and time of three algorithms under exponential distribution
表 3. 指数分布下的三种算法误差和时间

PSO-TNN		TNN		PINNs			
$N*L$	MSE	t/s	MSE	t/s	$N*L$	MSE	T/s
12*1	$1.6967e-10$	0.0392967	$4.1665e-06$	0.0406790	50*3	$6.8827e-08$	1124.6291
200*1	$6.1304e-09$	1.2220447	$4.2635e-06$	1.0881098	100*3		

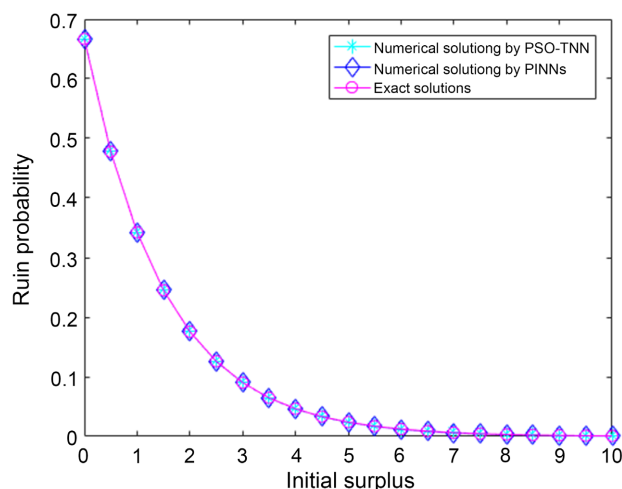


Figure 2. Numerical and analytical solutions by PINNs, PSO-TNN in (2)
图 2. (2)的 PINNs, PSO-TNN 数值解以及准确解

4. 结束语

本文以经典风险模型下的积分 - 微分方程为背景, 分别使用了改进的极限学习机和使用梯度信息优化的深度学习的最具代表性算法之一 PINNs 进行求解比较研讨, 通过理论证明了积分 - 微分方程(7)~(8), 当区间为 $[0, b], b \gg 10$ 时, 选用 sigmoid/tanh 激活函数, 使梯度为 0, 不适合使用 PINNs 求解方程; 选用 relu 激活函数, 用线性的试解去拟合非线性的真实解, 导致误差很大。再将结论推广到了一类使用梯度更新的深度学习算法和微分方程上, 又证明了解为不连续函数会由间断点的类型决定是否适用 ELM。又通过实验验证了上面的理论, 实验表明当 $|a| < 10$ 且 $|b| < 10$ 时, 使用 PSO-ELM 和 PINNs 的误差相近, 但是若有时间需求可选用 PSO-ELM。本文对于两种方法进行对比探究, 可以根据方程类型直接选定求解算法, 避免了逐个验证, 节省了计算资源和时间。[23]中表现出了 PINNs 的对拥有间断点的微分方程良好的近似能力, 未来的发展可以改进 PINNs 使它适用于推论 2 中且解是非连续的方程。

基金项目

贵州省科技计划项目(黔科合平台人才[2020]5016), 贵州大学一流课程培育项目(XJG2021040); 贵州大学教改项目(XJG2021027); 贵州大学研究生创新人才计划项目。

参考文献

- [1] Milligen, B.V., Tribaldos, V. and Jiménez, J.A. (1995) Neural Network Differential Equation and Plasma Equilibrium Solver. *Physical Review Letters*, **75**, 3594-3597. <https://doi.org/10.1103/PhysRevLett.75.3594>
- [2] Kim, S., Ji, W., Deng, S., Ma, Y. and Rackauckas, C. (2021) Stiff Neural Ordinary Differential Equations. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, **31**, Article No. 093122. <https://doi.org/10.1063/5.0060697>
- [3] Khoo, Y., Lu, J. and Ying, L. (2021) Solving Parametric PDE Problems with Artificial Neural Networks. *European Journal of Applied Mathematics*, **32**, 421-435. <https://doi.org/10.1017/S0956792520000182>
- [4] Chaharborj, S.S. and Yaghoub, M. (2017) Study of Fractional Order Integro-Differential Equations by Using Chebyshev Neural Network. *Journal of Mathematics and Statistics*, **13**, 1-13. <https://doi.org/10.3844/jmssp.2017.1.13>
- [5] Liu, Y., Kutz, J.N. and Brunton, S.L. (2022) Hierarchical Deep Learning of Multiscale Differential Equation Time-Steppers. *Philosophical Transactions of the Royal Society A*, **380**, Article ID: 20210200. <https://doi.org/10.1098/rsta.2021.0200>
- [6] Mo, Y., Ling, L. and Zeng, D. (2022) Data-Driven Vector Soliton Solutions of Coupled Nonlinear Schrödinger Equation Using a Deep Learning Algorithm. *Physics Letters A*, **421**, Article ID: 127739.

- <https://doi.org/10.1016/j.physleta.2021.127739>
- [7] Yang, Y., Hou, M., Sun, H., *et al.* (2020) Neural Network Algorithm Based on Legendre Improved Extreme Learning Machine for Solving Elliptic Partial Differential Equations. *Soft Computing*, **24**, 1083-1096. <https://doi.org/10.1007/s00500-019-03944-1>
- [8] Lu, Y., Chen, G., Yin, Q., Sun, H. and Hou, M. (2020) Solving the Ruin Probabilities of Some Risk Models with Legendre Neural Network Algorithm. *Digital Signal Processing*, **99**, Article ID: 102634. <https://doi.org/10.1016/j.dsp.2019.102634>
- [9] Huang, G.B., Zhu, Q.Y. and Siew, C.K. (2006) Extreme Learning Machine: Theory and Applications. *Neurocomputing*, **70**, 489-501. <https://doi.org/10.1016/j.neucom.2005.12.126>
- [10] Yang, Y., Hou, M., Luo, J. and Tian, Z. (2020) Numerical Solution of Several Kinds of Differential Equations Using Block Neural Network Method with Improved Extreme Learning Machine Algorithm. *Journal of Intelligent & Fuzzy Systems*, **38**, 3445-3461. <https://doi.org/10.3233/JIFS-190406>
- [11] Chen, Y., Yi, C., Xie, X., Hou, M. and Cheng, Y. (2020) Solution of Ruin Probability for Continuous Time Model Based on Block Trigonometric Exponential Neural Network. *Symmetry*, **12**, Article No. 876. <https://doi.org/10.3390/sym12060876>
- [12] Raissi, M., Perdikaris, P. and Karniadakis, G.E. (2017) Physics Informed Deep Learning (Part I): Data-Driven Solutions of Nonlinear Partial Differential Equations. ArXiv: 1711.10561.
- [13] Raissi, M., Perdikaris, P. and Karniadakis, G.E. (2017) Physics Informed Deep Learning (Part II): Data-Driven Discovery of Nonlinear Partial Differential Equations. ArXiv: 1711.10566.
- [14] Lu, L., Meng, X., Mao, Z. and Karniadakis, G.E. (2021) DeepXDE: A Deep Learning Library for Solving Differential Equations. *SIAM Review*, **63**, 208-228. <https://doi.org/10.1137/19M1274067>
- [15] Wang, L. and Yan, Z. (2021) Data-Driven Rogue Waves and Parameter Discovery in the Defocusing Nonlinear Schrödinger Equation with a Potential Using the PINN Deep Learning. *Physics Letters A*, **404**, Article ID: 127408. <https://doi.org/10.1016/j.physleta.2021.127408>
- [16] Chen, J., Du, R. and Wu, K. (2020) A Comparison Study of Deep Galerkin Method and Deep Ritz Method for Elliptic Problems with Different Boundary Conditions. *Communications in Mathematical Research*, **36**, 354-376. <https://doi.org/10.4208/cmr.2020-0051>
- [17] Baydin, A.G., Pearlmutter, B.A., Radul, A.A. and Siskind, J.M. (2015) Automatic Differentiation in Machine Learning: A Survey.
- [18] Zhou, T., Liu, X., Hou, M. and Liu, C. (2019) Numerical Solution for Ruin Probability of Continuous Time Model Based on Neural Network Algorithm. *Neurocomputing*, **331**, 67-76. <https://doi.org/10.1016/j.neucom.2018.08.020>
- [19] Sengupta, S., Basak, S., Peters, R.A. (2018) Particle Swarm Optimization: A Survey of Historical and Recent Developments with Hybridization Perspectives. *Machine Learning and Knowledge Extraction*, **1**, 157-191. <https://doi.org/10.3390/make1010010>
- [20] Kingma, D.P. and Ba, J. (2014) Adam: A Method for Stochastic Optimization. *Computer Science*.
- [21] You, H., Guo, J. and Jiang, J. (2019) Interval Estimation of the Ruin Probability in the Classical Compound Poisson Risk Model. *Computational Statistics & Data Analysis*, **144**, Article ID: 106890. <https://doi.org/10.1016/j.csda.2019.106890>
- [22] Wang, G. (2001) A Decomposition of the Ruin Probability for the Risk Process Perturbed by Diffusion. *Insurance: Mathematics and Economics*, **28**, 49-59. [https://doi.org/10.1016/S0167-6687\(00\)00065-2](https://doi.org/10.1016/S0167-6687(00)00065-2)
- [23] Mao, Z., Jagtap, A.D. and Karniadakis, G.E. (2020) Physics-Informed Neural Networks for High-Speed Flows. *Computer Methods in Applied Mechanics and Engineering*, **360**, Article ID: 112789. <https://doi.org/10.1016/j.cma.2019.112789>