基于深度神经算子框架的柴油发动机气体流动动力学实时预测

Varun Kumar[a]，Somdatta Goswami[b]，Daniel Smith[c]，George Em Karniadakis[a,b,*]

[a]布朗大学工程学院，美国罗得岛州普罗维登斯[b]布朗大学应用数学系，美国罗得岛州普罗维登斯[c]康明斯公司，美国印第安纳州哥伦布

arXiv:2304.00567v2 [eess.SP] 6 Jul 2023

摘要

我们开发了一种数据驱动的深度神经算子框架，用于近似柴油发动机的多个输出状态，并以合理的精度生成实时预测。随着排放标准日趋严格，能够分析系统行为的快速、准确模型已成为系统开发的必要条件。内燃机运行过程中涉及的快速瞬态过程，使得为此类系统开发准确的基于物理的模型十分困难。作为基于物理的模型的替代方案，我们开发了一种基于算子的回归模型（DeepONet），以发动机运行条件为输入变量，学习平均值燃气流动发动机模型的相关输出状态。我们采用平均值模型作为比较基准，并使用Simulink进行仿真。所开发的方法需要利用输出状态的初始条件，才能预测时间域内准确的序列。为此，我们在所提出的框架中嵌入了序列到序列方法。通过将模型的预测输出与Simulink模型生成的真实值进行比较，评估模型的准确性。观察到的最大 $\mathcal{L}_2$ 相对误差约为6.5%。我们在模拟噪声条件下评估了DeepONet模型的敏感性，结果显示该模型对噪声的敏感性较低。我们还使用均值集成方法进一步评估模型预测的不确定性。在（$\mu + 2\sigma$）边界处发现的最坏情况误差为12%。所提出的框架能够实时预测输出状态，并支持对复杂输入输出算子映射进行数据驱动学习。因此，该模型可应用于初始开发阶段，此时可能尚无准确模型可用。

关键词：柴油发动机，神经网络，非线性动力学，算子学习，不确定性量化

1. 引言

与汽油发动机相比，柴油发动机具有更高的峰值扭矩和热效率，因此广泛应用于重型领域。然而，由于这类发动机会排放危害健康的氮氧化物（NO_x）和颗粒物，因此受到严格的排放控制限值约束。为确保符合相关法规标准，通常需要在最佳运行条件与发动机排放之间进行权衡。在此背景下，通过基于模型的工程方法理解内燃机的运行，已成为产品

*通讯作者。电子邮件：george_karniadakis@brown.edu

开发中的一种常用方法。发动机制造商不断寻求通过调整基本运行设置来提升性能的方法，以改善其中任一性能指标。解析模型已广泛用于仿真内燃机的行为，并且已有多种商业软件包可用于对其行为进行建模 [1–3]。文献 [4, 5], 提出的柴油发动机气体流动仿真平均值模型基于歧管充排气概念。这些模型通过对发动机瞬态行为进行近似来仿真发动机行为，并已广泛用于控制设计和故障诊断 [6–9]。尽管发动机仿真模型的预测能力已显著提升，但人们已认识到，基于数值模型的柴油发动机控制复杂度会随独立变量数量的增加而上升，因此需要采用多维、灵活且自适应的扩展模块。

近年来，数据驱动和物理驱动的代理建模取得了显著进展，在模拟复杂系统行为方面成效卓著 [10, 11]。这些模型依靠经验数据，通过分析方法构建真实系统行为的表征。深度学习技术凭借其利用标注数据集对复杂非线性关系进行建模的固有能力，处于数据驱动建模范式的前沿 [12]。此外，这些代理模型能够快速预测，这对于现场部署至关重要。然而，深度学习算法处理真实数据的能力仍面临额外挑战，因为真实数据往往伴有噪声，而这给内燃机建模带来了困难。

本研究旨在采用适当的深度学习技术，开发一种稳健高效的代理模型，以模拟柴油发动机的运行。具体而言，我们开发了一种深度算子网络（下文称为 DeepONet），用于预测柴油发动机的气体流动动力学。神经网络的输入是来自现场的传感器测量数据，而真实值则通过一个平均值发动机 Simulink模型进行模拟[1]。所开发的代理模型考虑了构成内燃机完整分析模型的各项参数彼此独立这一特性。此外，我们还对模型不确定性开展了全面研究。以下列出了本研究的主要贡献：

- 展示了深度神经算子（DeepOnet）在预测七种输出状态方面的应用（进气歧管压力 P_{im}、排气歧管压力 P_{em}、残余气体分数x_r、进气过程结束时进气阀关闭后的温度 T_1、涡轮轴转速 ω_t、废气再循环执行器信号 $\tilde{u}_{egr}$以及可变几何涡轮增压器执行器信号 $\tilde{u}_{vgt}$），在 1，000 秒预测窗口内的最大相对 $\mathcal{L}2$ 误差为$\approx 6.5\%$。我们使用现有的 Simulink模型作为真实值数据生成器来训练 DeepONet 模型，然后对未见输入样本预测输出状态。

- 能够在不到一秒的时间内生成实时预测，只需提供一组输入即可。DeepONet 训练完成后，使用训练好的模型生成预测所需的时间极短。对于此类系统的实际应用而言，能够即时生成准确预测具有显著优势。

- 展示一种用于学习柴油发动机输入与输出状态之间函数映射的 DeepONet 架构示例。该映射基于

[1]由 Johan Wahlström 和 Lars Eriksson 所属的 Vehicular Systems 提供的软件包。http://www.fs.isy.liu.se/Software

将发动机转速、燃油供给、EGR阀位置和VGT阀位置数据作为 DeepONet 模型的输入，而无需掌握控制方程的具体知识；这些方程有时可能并不准确已知。

- 识别所提出的 DeepONet 模型在输入含噪时的不确定性，并确定此类条件下的误差。随着噪声水平增加，代理模型的预测误差也随之增大，但增幅仍在可接受范围内。输入中添加噪声后，输出状态 P_{em} 的最大 $\mathcal{L}_2$误差为 $\approx 7\%$。

- 通过在神经网络架构中使用 Dropout，确定模型估计的不确定性。我们采用基于集成的方法，量化 DeepONet 模型预测中存在的最大不确定性。结果发现，在距集成均值 2σ 个标准差处，最大相对 $\mathcal{L}_2$ 误差为 12.6%，比使用集成均值时报告的 P_{em} 误差约高 6%。

本文其余部分安排如下。在第 2 节中，简要介绍用于生成深度算子网络训练真实值的Simulink模型，并讨论为满足当前工作的目标而对 Simulink模型所作的少量修改。在第 3 节中，我们简要概述 DeepONet架构，并展示所开发的用于柴油发动机建模的代理模型。第 4 节介绍所开展实验的详细内容，以及添加噪声对预测精度的影响。第 5 节展示通过在分支网络中使用 Dropout 得到的模型不确定性结果。最后，在第 7 节中，我们简要总结观察结果，并报告当前方法的已知局限性。

2. 柴油发动机的数值仿真

柴油发动机内空气与废气的快速交换以及能量传递，使得建立用于分析的代表性模型颇具挑战。为简化模型，研究人员提出了基于歧管容积排空与充气过程的平均值模型（[4, 5]）。Wahlström 和 Ericsson 提出的发动机模型 [13] 便是此类简化模型之一。该模型基于歧管、废气再循环阀和涡轮增压器内的气体流动动态，也是本研究生成仿真数据的依据。本节简要介绍该模型的输入和输出，以便说明。图 1展示了与 Wahlström 模型相关的各个组件示意图。该模型的输入和输出是通过专用传感器在一段时间内记录的信号。该模型的输入信号可定义为输入向量：

$$\text{inputs} : [n_e,\ u_\delta,\ u_{egr},\ u_{vgt}],$$

其中，n_e 表示发动机转速，u_δ 表示每个循环喷入气缸的燃油量，u_{egr} 和u_{vgt}表示在发动机标定过程中通过经验确定的废气再循环阀和可变几何涡轮增压器阀开度。该模型产生的输出状态定义为输出向量：

$$\text{output states} : [P_{im},\ P_{em},\ X_{O_{im}},\ X_{O_{em}},\ \omega_t,\ \tilde{u}_{egr1},\ \tilde{u}_{egr2},\ \tilde{u}_{vgt}],$$

其中，P_{im} 为进气歧管压力，P_{em} 表示排气歧管压力，$X_{O_{im}}$表示进气歧管中的氧气质量分数，$X_{O_{em}}$ 表示排气歧管中的氧气质量分数，$\tilde{u}_{egr1,\ 2}$ 表示废气再循环执行器动态特性，$\tilde{u}_{vgt}$ 表示可变几何涡轮增压器阀执行器动态特性。通过在 Simulink 中使用传统数值求解器求解相应的常微分方程（ODE），可获得这八个输出状态。由 Simulink 生成的输出数据

模型在本研究中用作真实值。有兴趣的读者可参阅 [13] ，了解输入与输出状态之间的关系以及 Simulink 模型的其他参数详情[14]。

为实现第1节讨论的当前研究目标，我们修改了原始 Simulink 模型，以提取所需的输出状态。这些修改是为了使本研究能够适用于未来的参数辨识任务。我们在 Simulink模型中添加了额外模块，以提取必要的输出状态。修改后的模型提取的新输出状态表示为

$$\text{new output states} : [P_{im},\ P_{em},\ x_r,\ T_1,\ \omega_t,\ \tilde{u}_{egr},\ \tilde{u}_{vgt}],$$

其中， x_r 表示残余气体分数， T_1 表示进气冲程结束、进气阀关闭时的温度， $\tilde{u}_{egr}$ 表示通过组合 $\tilde{u}_{egr1}$ 和 $\tilde{u}_{egr2}$ 得到的废气再循环阀执行器输出（见 [13]中的公式39）。输入数据采集自试验台上的一台6缸重型卡车发动机。生成 Simulink模型输出数据所需的参数采用自 [13]。我们采用2 Hz的采样频率生成真实值解，用于评估代理模型的准确性。

Wahlström 和 Ericsson（WE）发动机模型采用一组参数，这些参数由对发动机实验室测量值进行最小二乘拟合得到。因此，要将该模型用于不同的发动机，必须确定这些参数。例如，废气再循环阀的 Simulink模型采用系数c_{egr1}，c_{egr2}，c_{egr3}， 和 Π_{egropt}，这些系数通过经验曲线拟合确定。因此，要仿真并将WE模型用于其他应用，还需根据解析关系确定参数，这需要额外工作。在优化循环中求解反问题以计算这些参数，会增加预测延迟。代理模型能够以较低成本近似高保真仿真，在保持解准确性的同时显著节省计算量。 2019 [15]，介绍的深度神经算子（ DeepONet）已被有效用作复杂物理问题的代理模型，例如断裂力学 [16]，、气泡动力学 [17]，和电对流 [18] 等。基于算子的模型的一项重要优势是能够根据数据学习输入与输出之间的非线性函数映射。此外，预训练 DeepONet 的预测时间不到一秒，这对于现场应用中的实时预测至关重要。DeepONet具有灵活性，能够推广到不同的运行条件，例如不同的环境温度和压力；这是它相较于传统求解器的另一项优势。在下一节中，我们将展示所提出的算子回归模型，该模型用于将输入条件映射到输出状态。

3. 柴油发动机建模的算子回归

深度算子网络能够从数据中学习非线性算子，适用于难以确定基于物理的模型或需要获得泛化结果的场景。柴油发动机具有高频动态过程，因此适合采用基于算子的神经网络。DeepONet 的理念源自算子通用逼近定理 [19]，，该定理指出，单隐藏层神经网络能够准确逼近任意线性或非线性的连续函数或算子。在开始学习参数微分方程的解算子之前，我们必须先区分函数回归与算子回归。函数回归中的解

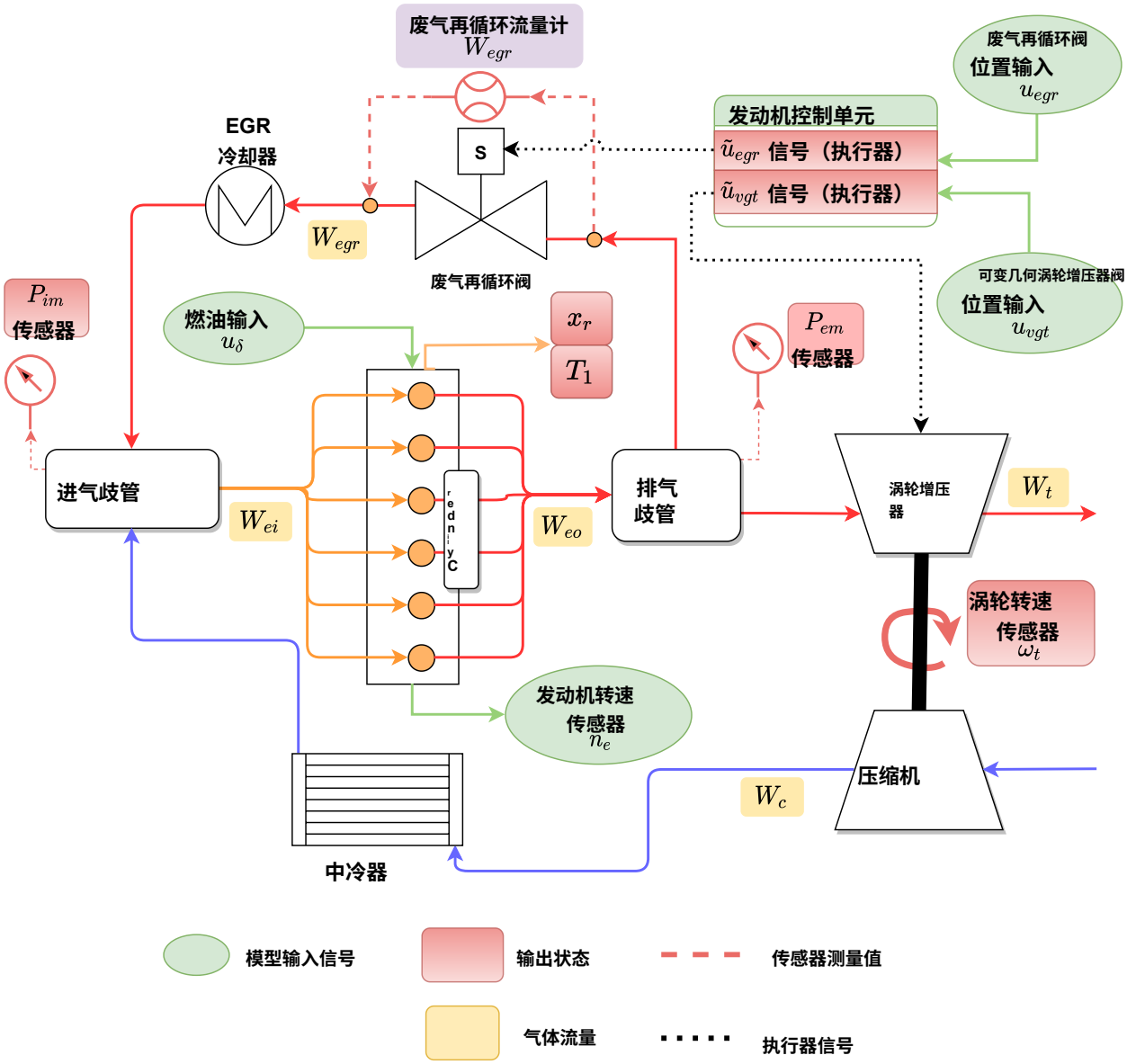

图 1：柴油机平均值模型中各子系统示意图（修改自 [13]）。在此示意图中，n_e、u_δ、u_{egr} 和 u_{vgt} 分别表示发动机转速、每个循环喷入气缸的燃油量、废气再循环阀和涡轮增压器阀门执行器接收的阀门位置信号。带有 W 的项表示流经该部件的质量流量，其中 W_{ei} 表示进入气缸的气体流量，W_{eo} 表示流入排气歧管的废气流量，W_{egr} 表示废气再循环气体流量，而 W_c 表示从压缩机流入进气歧管的新鲜空气流量。

方法将解参数化为有限维欧几里得空间之间的神经网络：$\mathcal{F}:\mathbb{R}^{d_1}\to\mathbb{R}^{d_1}$，其中 d_1 为离散化点的数量。而在算子回归中，则使用算子将一个函数映射为另一个函数。基于这一理念，本节首先介绍 DeepONet 的传统架构，随后介绍本文提出的 DeepONet 专用架构。

3.1. 深度算子网络（*DeepONet*）

DeepONet 由两个深度神经网络（DNN）组成： $\mathcal{N}_1$ （通常称为分支网络）对输入函数进行编码， $\boldsymbol{u}$ 在 m 个固定位置（通常称为传感器）处进行采样； $\mathcal{N}_2$ （称为主干网络）将解的评估位置作为输入， $\boldsymbol{y}$ （主干网络）。在广义设置下，只要输入函数在 m 个传感器位置处进行了离散化，分支网络的输入就可以采用物理域、初始条件或边界条件、常数或变量系数、源项等形式。对于输入函数的规则间隔离散化，可以使用卷积神经网络（CNN）作为分支网络；而对于稀疏表示，则可以考虑使用前馈神经网络（FNN），甚至可以针对序列数据使用循环神经网络（RNN）。在本研究中，我们使用 FNN 表示主干网络和分支网络。

为理解深度神经网络的理论基础，我们考虑一个具有 L 个隐藏层的神经网络，其中第 0 层表示输入层，第 $(L+1)$ 层表示输出层；进入某一层 l 上的一个神经元 i^{th} 的加权输入 $\boldsymbol{z}_i^l$ 是权重 $\boldsymbol{W}_{ij}^l$ 和偏置 $\boldsymbol{b}_j^{l-1}$ 的函数，可表示为

$$\boldsymbol{z}_i^l = \mathcal{R}_{l-1}\left(\sum_{j=1}^{m_{l-1}}\left(\boldsymbol{W}_{ij}^l(\boldsymbol{z}_j^{l-1}) + \boldsymbol{b}_j^l\right)\right), \tag{1}$$

其中， m_{l-1} 是第 $l-1$ 层中的神经元数量， $\mathcal{R}_{l-1}(\cdot)$ 表示第 l 层的激活函数。基于上述概念，计算输出 $\boldsymbol{Y}_L$ 的前馈过程表示如下：

$$\begin{aligned}
\boldsymbol{Y}^L &= \mathcal{R}_L(\boldsymbol{W}^{L+1}\boldsymbol{z}^L + \boldsymbol{b}^L)\\
\boldsymbol{z}^L &= \mathcal{R}_{L-1}\left(\boldsymbol{W}^L\boldsymbol{z}^{L-1} + \boldsymbol{b}^L\right)\\
\boldsymbol{z}^{L-1} &= \mathcal{R}_{L-2}\left(\boldsymbol{W}^{L-1}\boldsymbol{z}^{L-2} + \boldsymbol{b}^{L-1}\right)\\
&\vdots\\
\boldsymbol{z}^1 &= \mathcal{R}_0\left(\boldsymbol{W}^1\boldsymbol{x} + \boldsymbol{b}^1\right),
\end{aligned} \tag{2}$$

其中， $\boldsymbol{x}$ 是神经网络的输入。公式 2 可编码为压缩形式 $\boldsymbol{Y} = \mathcal{N}(x;\ \boldsymbol{\theta})$，其中 $\boldsymbol{\theta} = (\boldsymbol{W}, \boldsymbol{b})$ 包含神经网络 $\mathcal{N}$ 的所有权重和偏置。对于 DeepONet， $\mathcal{N}_1$ 以函数作为输入，以表示 $\boldsymbol{U} = \{\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_N\}$ 个样本的输入实现，这些输入实现在 n_{sen} 个传感器位置处离散化为 $\boldsymbol{u}_i = \{u_i(\boldsymbol{x}_1)\ u_i(\boldsymbol{x}_2)\ .....,\ u_i(\boldsymbol{x}_{n_{sen}})\}$ 和 $i \in [1, N]$；而 $\mathcal{N}_2$ 将位置 $\boldsymbol{y} = \{\boldsymbol{y}_1, \boldsymbol{y}_2, \cdots, \boldsymbol{y}_p\} = \{(\hat{}x_1, \hat{}y_1)、(\hat{}x_2, \hat{}y_2)\ .....、(\hat{}x_p, \hat{}y_p)$ 作为输入，以评估解算子，其中 $\hat{}x_i$ 和 $\hat{}y_i$ 分别表示点 $\boldsymbol{y}_i$ 的 x 和 y 坐标。设分支神经网络包含 l_{br} 个隐藏层，其中第 $(l_{br}+1)^{\text{th}}$ 层为输出层，由 q 个神经元组成。对于分支网络中的输入函数 $\boldsymbol{u}_i$ ，该网络返回一个嵌入于 $[b_1, b_2, \ldots, b_q]^{\mathrm{T}}$ 中的特征作为输出。前馈分支神经网络的输出 $\boldsymbol{z}_{br}^{l_{br}+1}$ 表示为

$$\begin{aligned}
\boldsymbol{z}_{br}^{l_{br}+1} &= [b_1, b_2, \ldots, b_q]^{\mathrm{T}}\\
&= \mathcal{R}_{br}\left(\boldsymbol{W}^{l_{br}}\boldsymbol{z}^{l_{br}} + \boldsymbol{b}^{l_{br}+1}\right),
\end{aligned} \tag{3}$$

其中，$\mathcal{R}_{br}(\cdot)$表示分支网络的非线性激活函数，$\boldsymbol{z}^{l_{br}} = f_{br}(u_i(\boldsymbol{x}_1)\ u_i(\boldsymbol{x}_2)\ldots, u_i(\boldsymbol{x}_m))$，其中 $f_{br}(\cdot)$表示一个分支网络函数。类似地，考虑一个包含 l_{tr} 个隐藏层的主干网络，其中第（$l_{tr}+1$）层为输出层

由 q 个神经元组成。主干网络输出一个特征嵌入 $[t_1, t_2, \ldots, t_q]^{\mathrm{T}}$。主干网络的输出可表示为

$$\begin{aligned} \boldsymbol{z}_{tr}^{l_{tr}+1} &= [t_1, t_2, \ldots, t_q]^{\mathrm{T}} \\ &= \mathcal{R}_{tr}\left(\boldsymbol{W}^{l_{tr}} \boldsymbol{z}^{l_{tr}} + \boldsymbol{b}^{l_{tr}+1}\right), \end{aligned} \tag{4}$$

其中，$\mathcal{R}_{tr}(\cdot)$ 表示主干网络的非线性激活函数，$\boldsymbol{z}^{l_{tr}-1} = f_{tr}(\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots\ldots, \boldsymbol{y}_p)$。关键在于，我们将一个新的算子 $\mathcal{G}_{\boldsymbol{\theta}}$ 揭示为一种神经网络，该网络能够从未见过的含噪输入中推断感兴趣量。训练这两个神经网络以学习解算子，使得

$$\mathcal{G}_{\boldsymbol{\theta}} : \boldsymbol{u}_i \to \mathcal{G}_{\boldsymbol{\theta}}(\boldsymbol{u}_i), \ \forall \ i = \{1, 2, 3, \ldots, N\}. \tag{5}$$

对于单个输入函数 $\boldsymbol{u}_i$，在任意坐标 $\boldsymbol{y}$ 处计算得到的 DeepONet 预测值 $\mathcal{G}_{\boldsymbol{\theta}}(\boldsymbol{u})$ 可表示为

$$\begin{aligned} \mathcal{G}_{\boldsymbol{\theta}}(\boldsymbol{u}_i)(\boldsymbol{y}) &= \sum_{k=1}^{q} \left(\mathcal{R}_{br}(\boldsymbol{W}_k^{l_{br}} \boldsymbol{z}_k^{l_{br}-1} + \boldsymbol{b}_k^{l_{br}}) \cdot \mathcal{R}_{tr}(\boldsymbol{W}_k^{l_{tr}} \boldsymbol{z}_k^{l_{tr}-1} + \boldsymbol{b}_k^{l_{tr}})\right) \\ &= \sum_{k=1}^{q} b_k(u_i(\boldsymbol{x}_1), u_i(\boldsymbol{x}_2), \ldots, u_i(\boldsymbol{x}_m)) \cdot t_k(\boldsymbol{y}). \end{aligned} \tag{6}$$

DeepONet 需要大量成对的输入输出观测标注数据集，但它提供了一种简单直观的模型架构，训练速度快，从而能够连续表示与分辨率无关的目标输出函数。通常，通过最小化损失函数来获得 DeepONet 的可训练参数 $\boldsymbol{\theta}$ （见 公式 6）。文献中常用的损失函数包括 L_1 损失函数和 L_2 损失函数，其定义为

$$\begin{aligned} \mathcal{L}_1 &= \sum_{i=1}^{n} \sum_{j=1}^{p} \left|\mathcal{G}(\boldsymbol{u}_i)(\boldsymbol{y}_j) - \mathcal{G}_{\boldsymbol{\theta}}(\boldsymbol{u}_i)(\boldsymbol{y}_j)\right| \\ \mathcal{L}_2 &= \sum_{i=1}^{n} \sum_{j=1}^{p} \left(\mathcal{G}(\boldsymbol{u}_i(\boldsymbol{y}_j) - \mathcal{G}_{\boldsymbol{\theta}}(\boldsymbol{u}_i)(\boldsymbol{y}_j)\right)^2, \end{aligned} \tag{7}$$

其中，$\mathcal{G}_{\boldsymbol{\theta}}(\boldsymbol{u}_i)(\boldsymbol{y}_j)$ 是 DeepONet 得到的预测值，$\mathcal{G}(\boldsymbol{u}_i)(\boldsymbol{y}_j)$ 是目标值。

接下来，我们介绍一种用于柴油发动机的 DeepONet 算法，并基于现有的带标签数据集计算深度神经网络的权重和偏置。

3.2. 柴油发动机的 *DeepONet*代理模型

通常，神经算子在设计时会将单个函数作为分支网络的输入。然而，在为柴油发动机设计代理模型时，我们需要考虑四个输入函数，n_e、u_δ、u_{egr}和u_{vgt}。传统 DeepONet架构对输入空间的限制，使我们无法学习定义在多个输入空间上的广泛有用算子。为此，我们在所提出的代理模型中采用了 DeepONet的多输入算子架构 [20, 21] 。本研究采用所提出的架构来近似柴油发动机的输出状态，如图2所示。

为准备 DeepONet 的训练数据，我们将从发动机试验台采集的四个输入的时序信号n_e，u_δ，u_{egr}，u_{vgt}按每个信号每十个时间点分段，从而将基于点的信号转换为基于特征的表示，以增强神经网络的学习能力

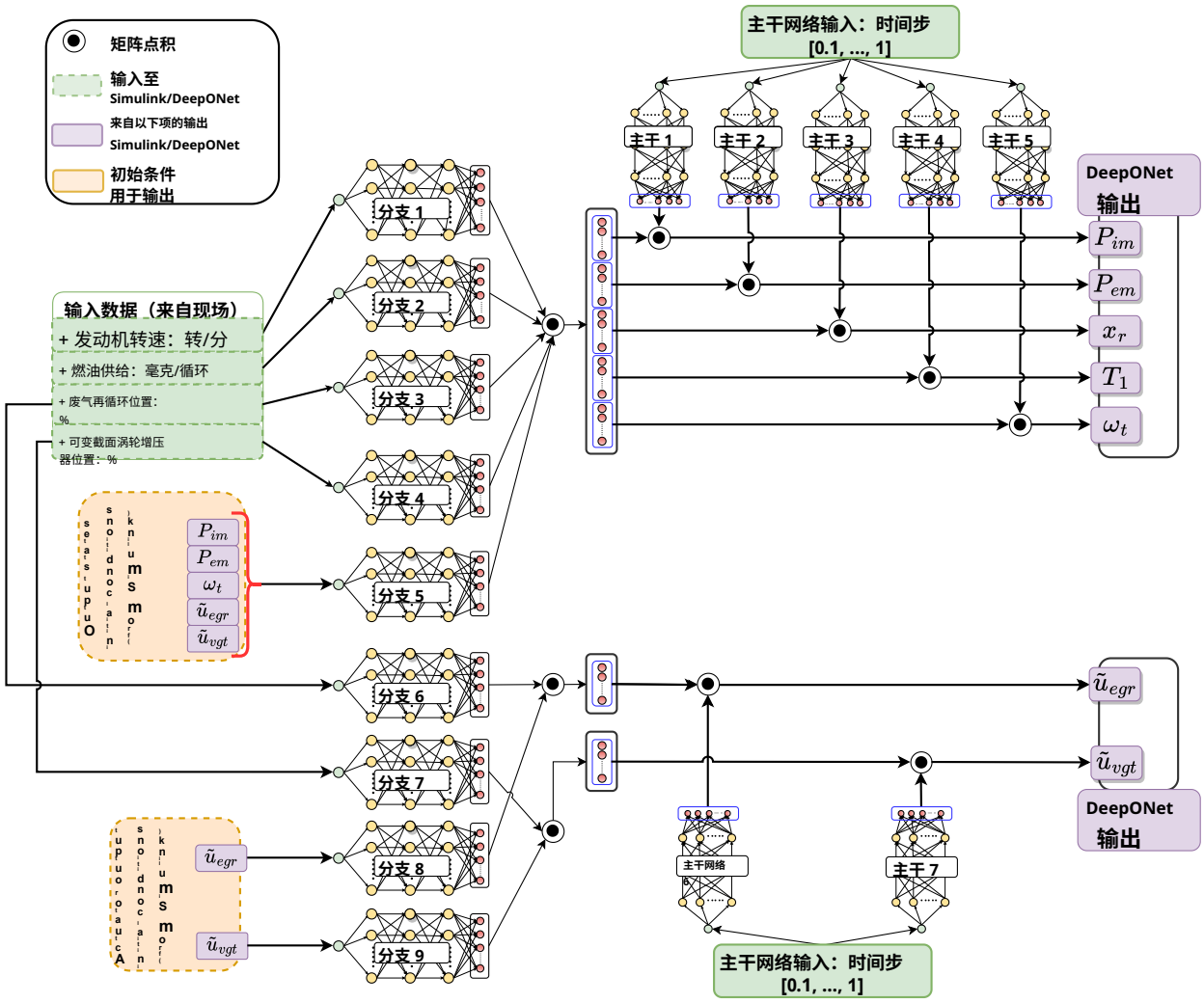

图 2：用于近似柴油发动机输出状态的多输入 DeepONet 架构。模型的输入置于绿色框中，输出置于紫色框中。该架构采用九个分支网络生成函数映射，七个主干网络则用于确定每个输出状态在各个时间点的基函数。由于 $\tilde{u}_{egr}$ 和 $\tilde{u}_{vgt}$ 严格依赖于 u_{egr} 和 u_{vgt}，因此为它们使用了独立的分支。为帮助解收敛，还将输出状态的初始条件作为额外输入提供。

过程。四个输入信号分别对应一个分支网络（图1 – 4 中2的分支1 – 4），用于学习特征，并充当专用的函数逼近器。除了这四个输入外，还以初始条件的形式向 DeepONet 提供另一组输入，这些初始条件从分支5中的输出预测提取而来。这类似于为求解微分方程提供初始条件。此处，每个训练信号的初始条件均从第一个输入点提取。图3 展示了从输出中提取初始条件的数据提取过程。请注意，初始条件值仅用于可在现场测量的输出状态，即 P_{im}，P_{em}，ω_t，$\tilde{u}_{egr}$和 $\tilde{u}_{vgt}$（见图1）。将初始条件作为输入有助于约束算子学习过程，这与求解微分方程的过程类似。分支网络 $1-5$ 和主干网络 $1-5$的输出嵌入的点积映射到五个输出状态：P_{im}，P_{em}，x_r，T_1和 ω_t。其函数表示为

输入 u_egr 和 u_vgt 分别由分支网络 6 和 7 生成；这些网络与分支网络 8 和 9 相结合。分支网络 8 和 9 包含有关 $\tilde{u}_{egr}$ 和$\tilde{u}_{vgt}$初始条件的信息，而主干网络 6 和 7 的相应输出则近似表示输出状态，$\tilde{u}_{egr}$ 和 $\tilde{u}_{vgt}$。由于四个输入中只有 u_{egr} 和 u_{vgt} 与状态 $\tilde{u}_{egr}$ 和 $\tilde{u}_{vgt}$ 相关，因此使用了独立的分支来生成这些状态。感兴趣的读者可参阅文献 [13]中的方程 $39-41$ 和 57，了解更多详情。理解输出状态与影响它们的输入之间的关联性十分重要，因为这有助于网络学习过程，并帮助建立输入与输出之间恰当的映射。这里需要强调的是，鉴于 DeepONet架构的灵活性，可根据需要调整所提出的设计，以纳入更多输入。例如，如果需要扩展网络以纳入新的输入和/或输出状态，则可修改设计，增加一个分支和主干。我们利用 [13] 中所述相关方程之间的关联性，将相应输入映射到输出状态。

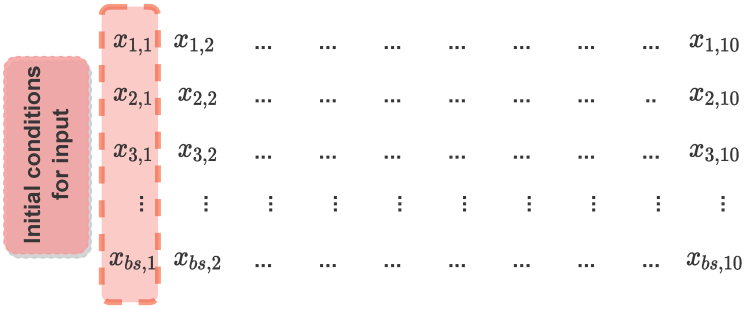

图 3：从输出场中提取初始条件，作为 DeepONet 的输入。这些初始条件仅针对可测量的输出提供 P_{im}，P_{em}，ω_t，$\tilde{u}_{egr}$，以及 $\tilde{u}_{vgt}$。

3.3. 数据生成

为了训练深度神经网络，我们使用第 2 节所述的 Simulink模型生成输出状态的真实值。图4 展示了利用 Simulink 生成的数据训练代理模型的方案。DeepONet 的预测值与 Simulink 的输出之间的均方误差用于反向传播和神经网络训练。在本研究中，Simulink模型使用从[13]获得的现有参数值进行仿真。

如前所述，初始条件取自带标签的输出数据集。然而，在实际系统中，输出状态的初始条件可能不可用；或者，人们可能希望将测量传感器的输出与代理模型给出的预期输出进行比较，以识别系统行为中的差异。为应对这一挑战，我们将该问题构建为序列到序列学习。第一个测试信号 $\mathcal{S}^1_{\text{test}}$ 的初始条件取自最后一个训练信号最后一个时间点的输出。随后，第 t 个测试信号 $\mathcal{S}^t_{\text{test}}$ 从信号 $\mathcal{S}^{(t-1)}$ 测试 的最后一个时间点的预测输出中获取初始条件；该输出是使用所提出的代理模型进行预测得到的。

数据划分：通过选取一段连续时间，将输入数据集和真实值数据集划分为训练集和测试集。完整数据集的总时长约为 15 小时，数据是在一段时间内使用同一台发动机采集的。从中选取一段连续的 1000 秒数据作为测试数据。测试数据片段经过仔细挑选，以确保其在训练集中也有代表性。由于数据集中怠速工况的数据不足，测试集未纳入发动机怠速等较少出现的工况

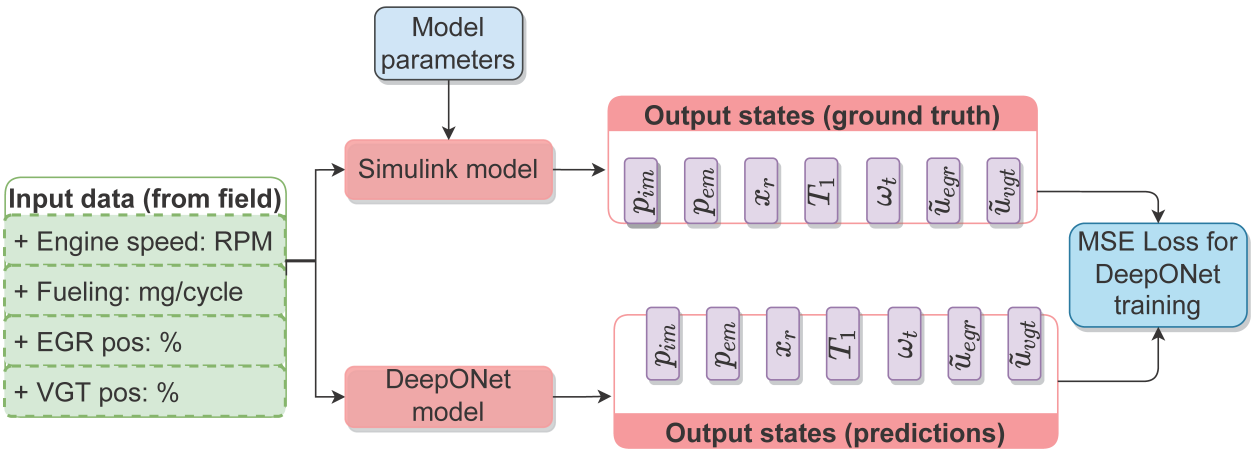

图 4：DeepONet 模型概览，以及将其与 Simulink模型 配合使用作为真实值生成器的方式。

供模型学习的数据点。需要注意的是，在生成输入数据时，应考虑所有可能遇到的发动机工况。使用丰富的输入空间进行训练，能够使 DeepONet 神经网络做出泛化预测。首先将输入数据转换为信号序列，窗口大小为 10（对应 5 秒的数据片段）。此转换旨在为神经网络模型提供更多特征，以增强学习效果。窗口大小通过尝试不同取值并结合实验结果进行启发式选择。观察发现，窗口大小增大时，模型精度会下降。随后将全部四个输入转换为信号序列，再输入 DeepONet。

4. 实验结果

本节介绍并讨论实验结果。除了使用干净输入数据进行的实验，我们还展示了含噪输入和输出下的结果，以说明对于类似问题，DeepONet 预测的敏感性。

4.1. DeepONet 预测结果

图2 所示的 DeepONet 代理模型的架构详情见表1。权重优化采用带学习率调度器的 Adam优化器。初始学习率为 $1e-3$，直至前5,000 个训练轮次，随后降至 $5e-4$，直至训练轮次达到 10,000。此后，使用恒定学习率 $1e-4$，直至训练在20,000 个训练轮次时结束。还在生成输入函数表示的分支网络中采用了 Dropout 作为网络正则化器。Dropout 丢弃率通过启发式方法进行调整。除了对神经网络进行正则化外，Dropout 还可用于模型不确定性估计 [22]，，我们将在以下章节中给出不确定性估计结果。模型在 NVIDIA A40 GPU 上训练，训练时间约为三小时。训练期间还使用了逐点自适应权重对每个时间点进行正则化 [23, 24]。表 $\mathcal{L}_2 2$（第一行）展示了在1000 秒预测时间窗口内七个输出状态的误差值。

表1：用于柴油发动机建模的DeepONet架构详情

神经网络	深度	宽度	激活函数		丢弃率
			隐藏层	最终层	
分支 $1-2$	3	128	Swish	ReLU	0.20
分支 $3-4$	3	128	Swish	ReLU	0.10
分支 $6-7$	2	128	Swish	线性层	0.05
分支 $8-9$	3	128	Swish	线性层	−−
主干网络	3	128	Swish	Sigmoid	−−

作为一个代表性案例，图5 展示了测试数据子集中第一个60 秒时间窗口内 DeepONet 预测值与真实值的对比。输出状态 P_{em}的误差率最高，在1000秒评估窗口内达到 6.5%。执行器信号状态 $\tilde{u}_{egr}$ 和 $\tilde{u}_{vgt}$ 的预测值与真实值吻合良好。这是预期之中的，因为根据其驱动常微分方程，它们的函数关系更容易映射。 P_{em} 表示排气歧管中的压力以及气缸内的燃烧过程。状态 x_r表示燃烧室内的残余气体分数，是确定废气状态的重要参数，尤其会影响其温度，进而影响压力。在我们的架构中，未将 x_r 的初始条件作为输入，因为该状态无法在现场直接测量。对 x_r 的估计误差可能会对状态 P_{em} 的误差产生显著影响，因为神经网络是联合训练的，从而导致 P_{em}的预测误差较高。状态P_{em} 误差较高也可能表明，需要识别更多输入特征，以帮助增强该状态的算子学习过程。图6展示了模型训练期间七个状态的误差传播。

4.2. 含噪数据的结果

与所有真实系统一样，柴油发动机的传感器信号都伴有难以确定或准确测量的噪声。为了解含噪输入对模型预测的影响，我们模拟了三种噪声条件：1%、2% 和 3% 的加性白高斯噪声。所选噪声水平基于真实系统中产生 DeepONet 四个输入的传感器的预期噪声水平。我们将这种白噪声添加到输入数据中，并使用先前讨论过的、已在干净数据上预训练的 DeepONet 架构进行预测。我们在与干净数据验证数据集相同的时间段内使用含噪数据进行测试，以确保预测所用的数据块未在神经网络训练期间出现过。

表2 展示了使用干净输入与在输入测试数据集中加入高斯白噪声时的预测结果误差对比。两个输出状态， $\tilde{u}_{egr}$ 和 $\tilde{u}_{vgt}$ 受输入噪声影响最大。噪声对其他输出状态的影响很小。图7展示了真实值与 DeepONet 在输入测试数据带有3% 高斯白噪声时的预测结果对比。

我们还评估了输出数据受到噪声干扰时模型预测的敏感性。本研究旨在了解模型在使用本身含有噪声的数据集进行训练时，噪声对模型准确性的影响；大多数现场测量都存在这种情况。为此，我们在输出训练数据集中添加了幅值最高达 3% 的高斯白噪声。噪声水平上限设为 3%，这是根据现场测量输出状态的传感器使用经验确定的

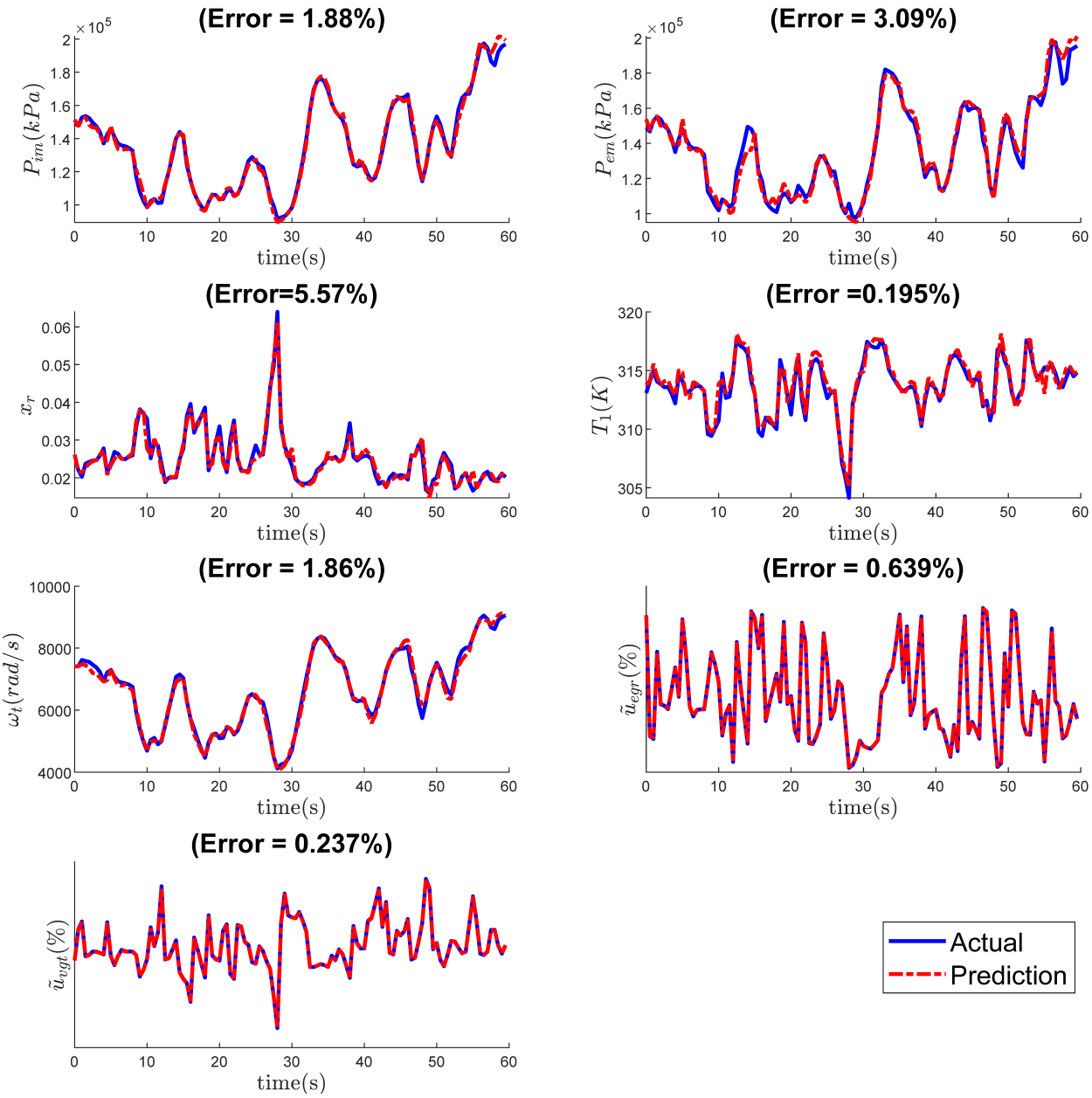

图 5：DeepONet 对七个输出状态的预测结果，以及与 Simulink 得到的真实值的对比。为便于查看，此处仅展示前 60 秒的数据子集。每个图上方标注的误差值是在这 60 秒时间窗口内计算得出的。出于保密考虑，不显示 $\tilde{u}_{egr}$ 和 $\tilde{u}_{vgt}$ 的纵轴刻度。

在本研究中，DeepONet 使用这些带噪标注数据进行训练，随后使用训练后的网络和干净的输入和输出数据集生成预测。表 3 对比了使用干净输出和带噪输出训练的 DeepONet 模型的误差。

4.3. 初始条件的序列到序列链式连接

除了使用来自真实值输出数据（由 Simulink 生成）的初始条件外，我们还评估了序列到序列链式连接初始条件对我们的

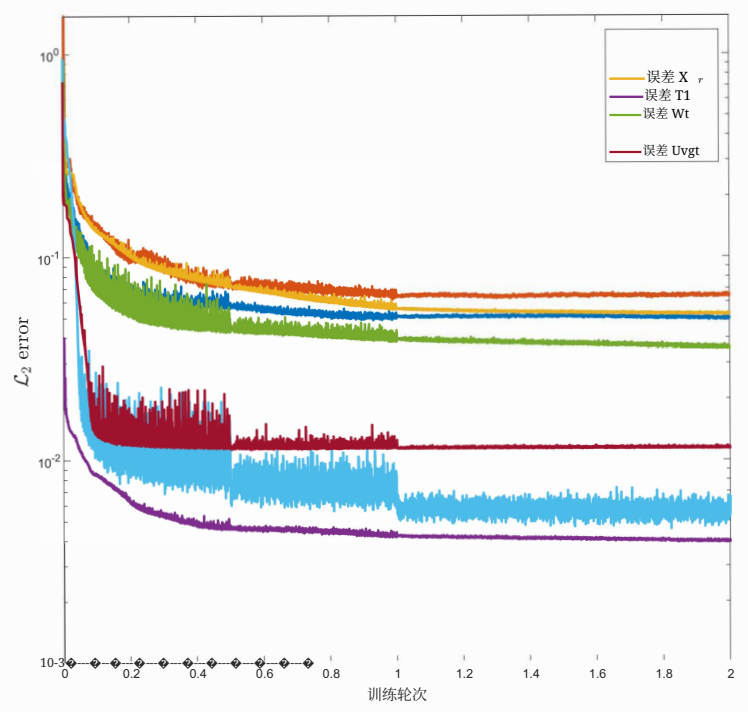

图 6：训练过程中七个输出状态的测试误差传播。各输出的误差传播程度有所不同，这可能是由各函数的算子映射复杂度不同所致。

表2：使用所提出的基于 DeepONet 的代理模型得到的七个预测输出状态的 $\mathcal{L}_2$ 误差值。误差结果涵盖 1000 秒的测试时间窗口。此处还展示了无噪声和不同噪声条件下的误差比较。

噪声水平	**$\mathcal{L}_2$ 误差（%）**						
	P_{im}	P_{em}	x_r	T_1	ω_T	$\tilde{u}_{egr}$	$\tilde{u}_{vgt}$
无噪声	4.96	6.44	5.22	0.39	3.56	0.59	1.15
1%	5.09	6.64	5.26	0.40	3.66	1.54	1.86
2%	5.15	6.75	5.30	0.41	3.71	2.64	3.03
3%	5.17	6.94	5.56	0.41	3.89	4.14	4.35

表 3：使用干净与含噪声的输出训练数据集训练的模型之间的 $\mathcal{L}_2$ 误差比较。对于在带标签的训练数据中添加噪声后训练的神经网络，其相对误差略有增加。此处的预测使用无噪声测试数据集生成。

噪声水平	**$\mathcal{L}_2$ 误差（%）**						
	P_{im}	P_{em}	x_r	T_1	ω_T	$\tilde{u}_{egr}$	$\tilde{u}_{vgt}$
无噪声	4.96	6.44	5.22	0.39	3.56	0.59	1.15
3%	4.97	7.14	5.47	0.85	5.17	0.67	1.17

数据训练信号。这在实际应用中很重要，因为真实值数据可能无法获取，也可能因数据记录系统存在系统性故障而出错。在这里，我们将前一个信号序列的 DeepONet 预测输出用作下一个信号序列的初始条件。此过程会应用于整个测试数据集，并且完整

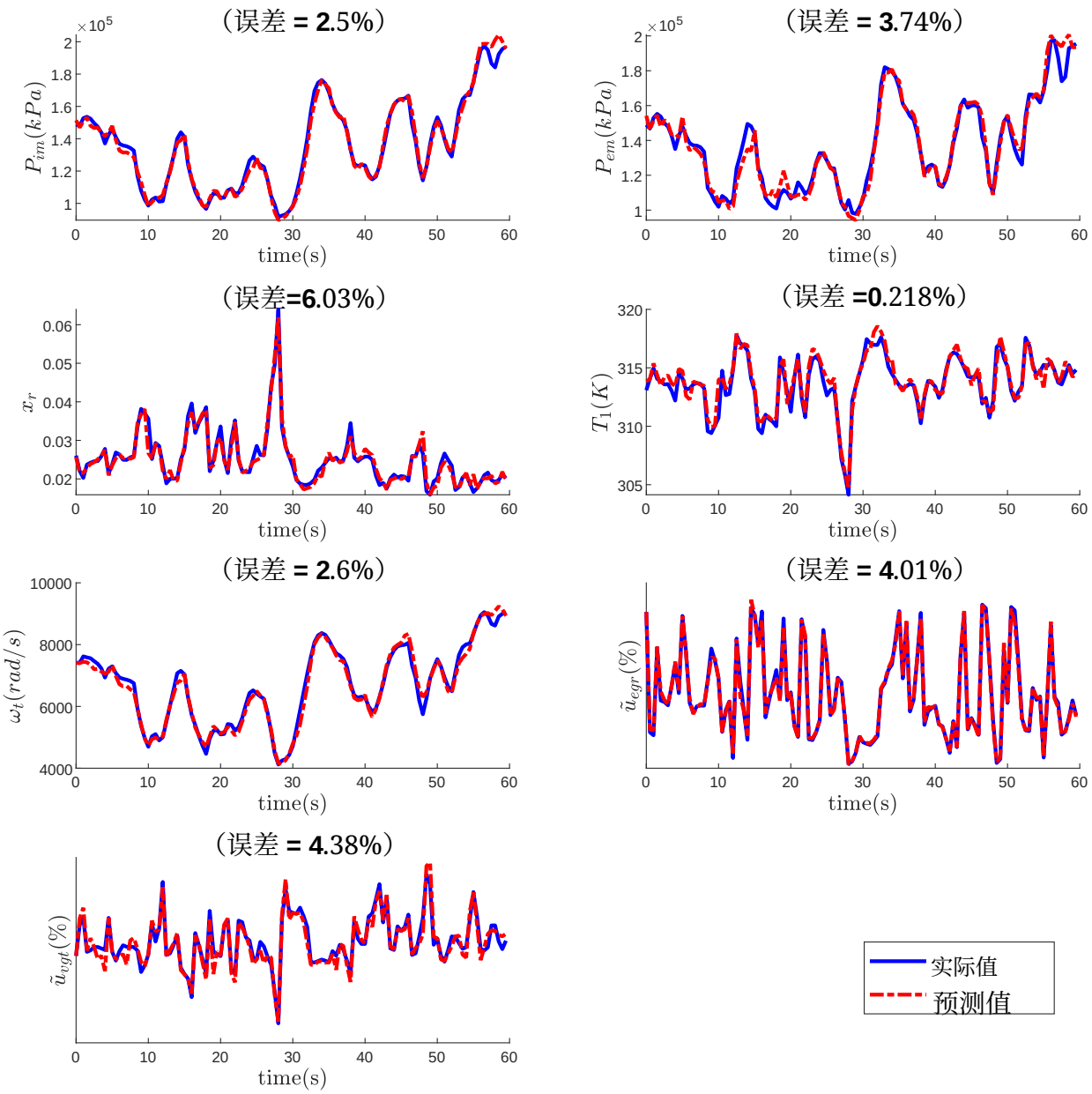

图7：向测试输入数据子集添加 3% 高斯白噪声后的 DeepONet 预测结果。预测误差随着噪声水平的增加而逐渐增大，其中执行器预测的增幅最大，这可能是由于其相应分支网络所采用的丢弃率较低。

的预测结果通过每次处理一个信号序列获得。然而，这会导致预测时间更长（≈ 30秒，本文中如此），并且随着预测序列变长，预测误差累积也会增大。图 8 展示了序列到序列初始条件使用方案与使用真实值数据中的初始条件两种方案的累积误差对比。表 4对比了这两种初始条件方案在预测过程中的相对误差。

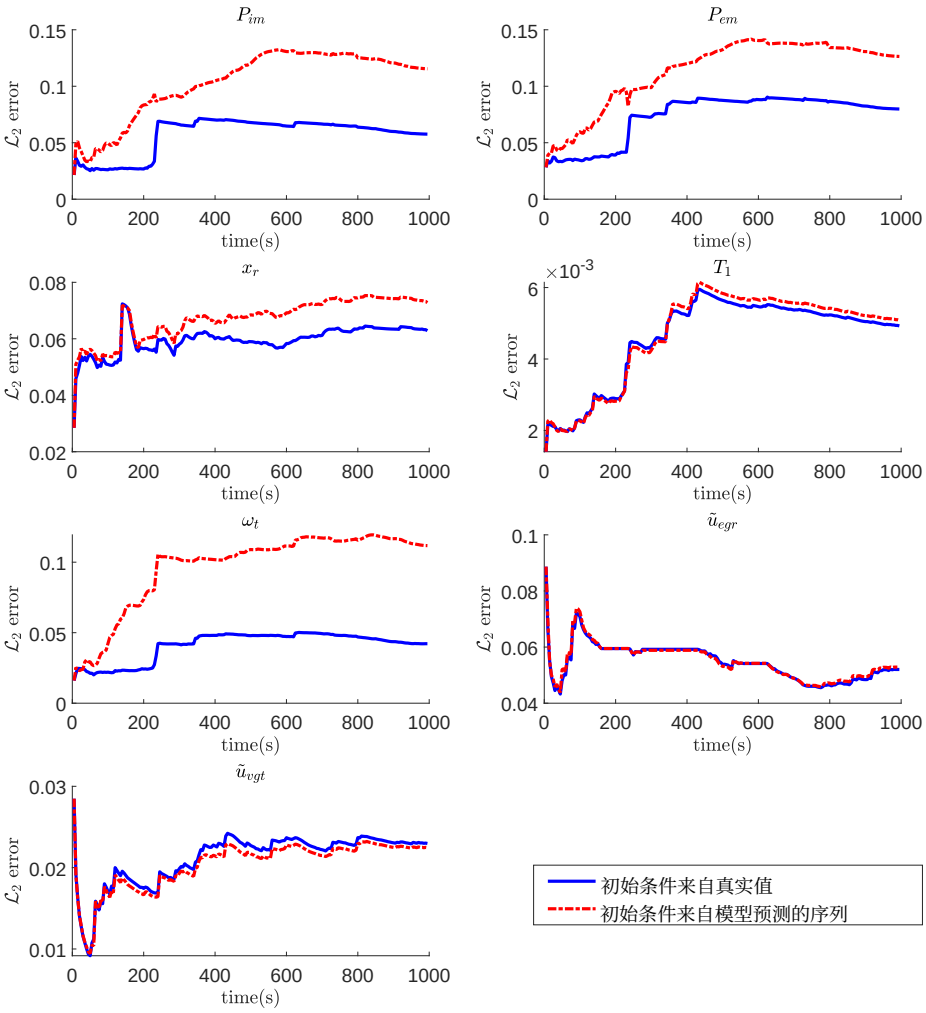

图8：比较采用序列到序列方案生成初始条件与采用来自真实值的信号序列初始条件时的累积误差。采用序列到序列方案时，误差累积更高，这是因为 DeepONet 的模型预测存在预测不准确性，将其用作下一个信号序列的初始条件会导致误差累积。对于状态 P_{im}、 P_{em} 和 ω_t ，累积误差最高，这可能是因为 DeepONet 模型本身的误差值较高。

表4： $\mathcal{L}_2$ 比较以真实值为初始条件的预测与采用序列到序列方案生成初始条件的预测误差。

初始条件来源	**$\mathcal{L}_2$ 误差（%）**						
	P_{im}	P_{em}	x_r	T_1	ω_T	$\tilde{u}_{egr}$	$\tilde{u}_{vgt}$
真实值	4.96	6.44	5.22	0.39	3.56	0.59	1.15
序列到序列	11.54	12.64	7.28	0.85	11.17	5.29	2.25

5. 模型不确定性估计

神经网络经过训练，学会一组特定的权重和偏置以预测输出后，便是确定性的，相同的输入会产生相同的输出。模型不确定性涉及模型权重估计中可能存在的

潜在差异；了解不确定性的程度有助于增强对模型预测的信心。在深度学习研究中，神经网络的概率建模已被用作不确定性估计工具 [25–27]。这些方法根植于贝叶斯概率框架。传统的贝叶斯建模方法包括蒙特卡洛 Dropout [22, 28], 哈密顿蒙特卡洛 [29–31], 反向传播贝叶斯 [32–34]等；全面综述请参见 [27]。在本研究中，我们使用 dropout 来理解模型不确定性，原因如下：

- Dropout 在 DeepONet 分支层中用作正则化器。
- Dropout 为模型不确定性估计提供了一种更快捷的方法。变分推断等其他方法在我们的这类复杂架构上可能难以训练。此外，结合我们的传感器数据设定先验分布和后验分布 [35] 也会带来额外的复杂性。

我们通过以下步骤展示模型不确定性：

1. 使用输入数据（无噪声）训练 DeepONet 模型，并在分支网络中使用蒙特卡洛 Dropout，架构如表 1 所示。2. 在数据的测试部分生成预测。每次进行预测时，由于 Dropout 层赋予分支网络随机性，得到的输出状态预测结果各不相同。3. 收集 100 个输出并计算集成均值，以表示平均预测结果， μ。计算每个点在所有这些预测结果中的标准差， σ 。4. 围绕集成均值创建不确定性区域，其范围为 $\mu \pm 2\sigma$

图 9 展示了集成均值的对比结果，该集成均值通过使用基于 Dropout 的 DeepONet 模型生成 100 次预测计算得到。每个点的标准差根据这 100 次预测计算得出，以生成围绕该集成均值的不确定性带。表 5 展示了 1000 秒测试子集的总误差，其中集成均值由基于 Dropout 的随机 DeepONet 生成的 100 次预测计算得到。使用集成均值估计的七个输出状态的相对 $\mathcal{L}_2$ 误差百分比高于确定性预测误差。这是由于使用 Dropout 的分支层具有随机性，导致神经网络在每次进行预测时都会学习不同的映射。为展示该模型可能产生的最大预测误差，我们使用 $\pm 2\sigma$ 值与集成均值计算最坏情况预测边界，并根据目标预测估计误差。最坏情况误差值也列于表5中。在此最坏情况场景下，预测估计的最大可能误差约为 12.6% ，对应状态 P_{em}。

6. 局限性

本节讨论所提出的基于算子网络的代理模型的一些局限性。前文讨论的预测误差值（表 2 和 5）是根据测试中使用的1000 秒子集计算得出的总误差值。然而，在整个时间跨度内，误差值并不均匀，不同时间段的结果各不相同。图10 展示了测试数据集中接下来 60 秒时间窗口的预测结果，并与图 5 所示结果进行了比较。可以看到，与图 5所示结果相比，四个输出状态的误差值显著更高。这

表5：DeepONet 对输出状态的确定性预测与通过在分支网络中使用 Dropout 生成的预测集成之间的误差比较。集成均值的误差值表示：当对 100 个不同预测进行集成时，用于测试的 1000 秒时间序列的总误差。集成 $\mu + 2\sigma$ 的误差值表示当预测处于不确定性带边界时的最坏情况误差。

测试条件	$\mathcal{L}_2$ **误差（%）**						
	P_{im}	P_{em}	x_r	T_1	ω_t	$\tilde{u}_{egr}$	$\tilde{u}_{vgt}$
确定性预测	4.96	6.44	5.22	0.39	3.56	0.59	1.15
集成 μ	4.87	6.34	5.22	0.39	3.58	0.77	1.15
集成 $\mu + 2\sigma$	8.01	12.60	10.90	0.75	6.81	9.06	3.90

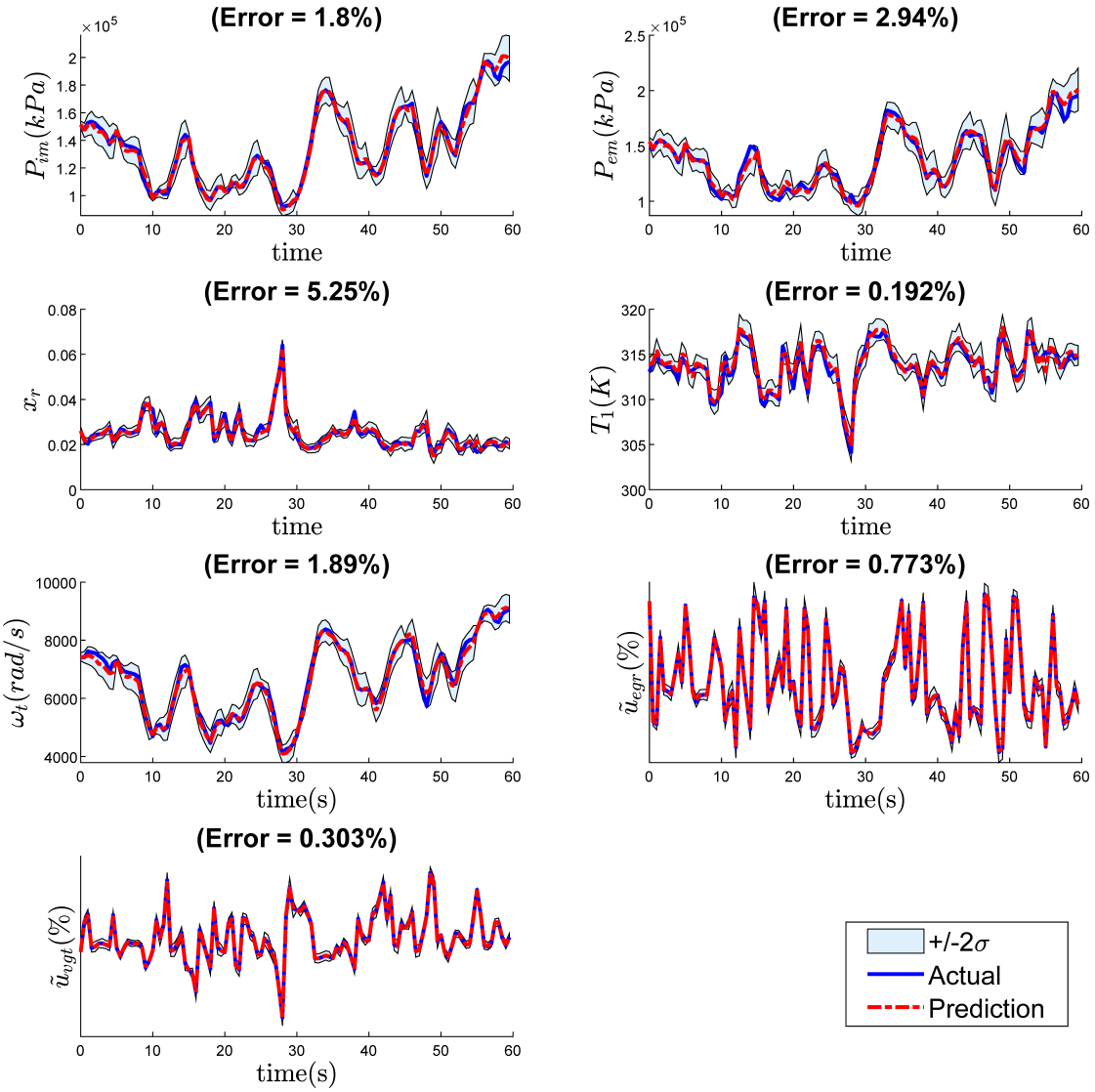

图9：DeepONet 预测结果与不确定性结果。蒙特卡洛丢弃网络在预测过程中抽取 100 个样本，生成集成均值，丢弃率如 1 所示。预测值为 100 个样本的均值，而这些样本的标准差所表示的不确定性则显示为集成均值周围的 $\pm 2\sigma$ 区域。

这一现象可能归因于输入数据，这些数据是在通用测试条件下从试验台采集的。这里使用的训练数据集可能仅限于预测某些运行条件。理想情况下，应使用经过标准化且包含发动机所有可能运行状态代表性信息的训练数据集进行训练和评估。

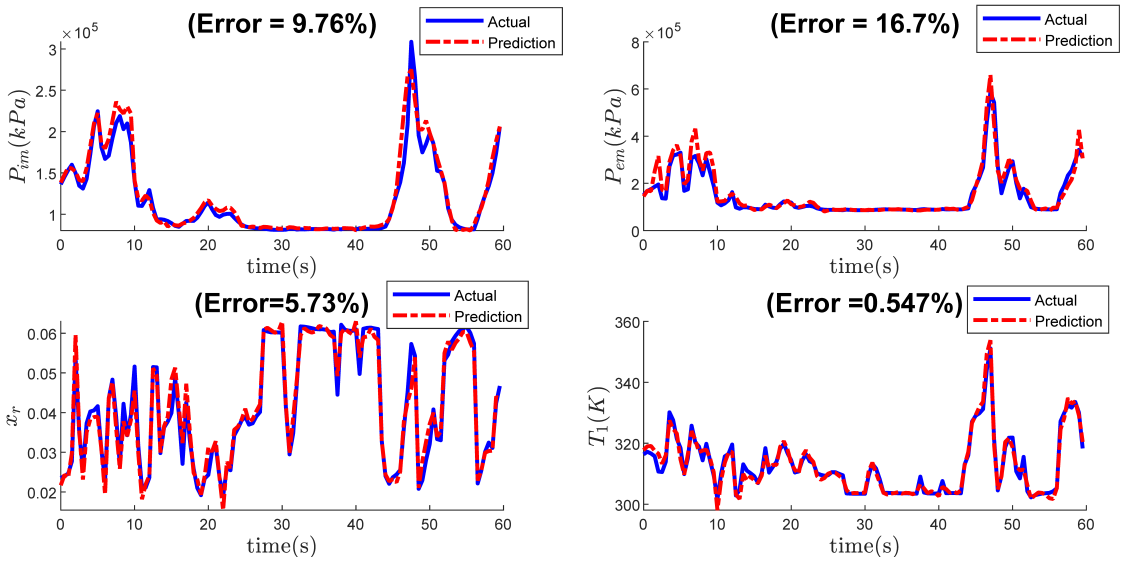

图10：DeepONet对测试数据集中另一时间段的预测结果，展示了在本研究所用训练数据集条件下当前模型的局限性。本例中观察到较高的误差百分比，这可能是由于训练数据集中缺少具有代表性的数据点。

当前DeepONet架构可以扩展到其他发动机型号。然而，这需要使用从每台发动机单独采集的数据训练DeepONet模型，工作量和耗时都很大。我们也认识到，利用现场传感器测量数据采集真实值来训练新模型可能颇具挑战，并且可能需要扩展输入特征以实现算子学习。这是我们将继续探索的研究方向，也是本研究未来目标的一部分。

当前的 DeepONet 模型使用 [13]提供的参数集进行了训练。为了生成数据，需要确定测试发动机使用平均值模型所需的参数值。如 [36]所述，可以使用 PINN 框架进行参数辨识。未来，我们将继续探索将算子网络与基于 PINN 的方法相结合的可能性。

7. 总结

本文提出了一种基于深度算子的神经网络模型，用于预测柴油发动机平均值气流模型的输出状态。该模型使用四个输入信号：发动机转速（n_e）、燃油供给（u_δ）、EGR阀位置（u_{egr}）和VGT阀位置（u_{vgt}），以生成到七个输出状态的连续算子映射。输出状态的真实值由 [14]在线提供的Simulink模型生成。DeepONet模型训练完成后，使用此前未见的测试数据预测这七个输出状态。其准确性

和鲁棒性分别通过相对于真实值的 $\mathcal{L}_2$ 相对误差以及噪声条件下的准确性进行评估。我们的研究结果总结如下：

1. 基于深度算子的神经网络模型能够学习算子，将柴油发动机平均值气流模型的输入转换为所需输出。DeepONet 预测的输出状态在 1000 秒测试时间窗口内表现出较高的精度（见表 2）。观测到的最大相对误差为 6。4% ，对应输出状态 P_{em}。该状态高度依赖排气歧管动力学，例如排气歧管温度，因此可能需要增加输入特征以提高精度。我们还从图 5中观察到，根据特征形状，状态 ω_t、 P_{im} 和 P_{em} 彼此相关。这是涡轮增压器总成与进气和排气系统耦合的结果。

2. DeepONet 在使用模拟噪声数据测试输入和输出时，表现出良好的泛化精度。在使用无噪声输入和带标签数据训练的模型进行测试时，向输入添加 3% 白噪声会使预测误差略有增加（见表2）。除了向输入添加噪声外，我们还研究了带噪声标签数据的影响：使用模拟噪声标签训练神经网络，然后使用无噪声数据进行测试。实验中同样观察到预测误差略有增加（见表 3）。我们的神经网络模型对噪声不太敏感，这可能归因于网络架构中采用了 Dropout，使神经网络能够为回归任务学习更具泛化能力的解。

3. 为了使我们的架构能够应用于真实场景（在这些场景中，真实值可能不可用或不可信），我们提出了初始条件的序列到序列连接。将当前信号序列的最后一个输出预测用作下一个信号序列的初始条件和预测值。然而，与使用输出状态的真实值（若可用）作为模型初始条件的情况相比，此过程会导致更高的累积误差（见表4）。从图8中可以看出，累积误差会随预测窗口而变化，这表明模型的准确度取决于预测窗口的位置。我们将在局限性部分6中对此进行更详细的讨论。

4. 我们通过结合 Dropout 层的集成均值方法来确定 DeepONet 模型的不确定性。在 $\mu + 2\sigma$ 极限处发现，最坏情况下的相对误差为 12。6% 而相对于集成均值的相对误差与确定性模型的误差处于相同范围内（见表 5）。

作为这项工作的延续，我们正在探索如何扩展 DeepONet 模型，使其能够泛化到不同的运行条件。运行环境条件、温度和压力都可能导致发动机产生不同的响应。为此，可以添加一个新的分支输入，并使用通过改变 Simulink模型中的环境温度和压力参数生成的仿真数据进行训练。

致谢：

本研究使用了布朗大学计算与可视化中心提供的计算资源和服务。

声明

- **资助：**本研究由康明斯公司资助。

- **利益冲突：**作者 Varun Kumar 和 Somdatta Goswami 声明不存在任何经济利益。作者 George Em Karniadakis 获得了康明斯公司的研究支持。

- **所用数据的伦理审批与知情同意：**不适用。

- **数据可用性与获取方式：**本研究期间生成和/或分析的数据集，可在论文发表后，经康明斯公司许可，向通讯作者提出合理请求获取。

- **作者贡献：**Varun Kumar 负责数据生成、数据处理、机器学习模型设计、编程、结果解读及材料准备。Somdatta Goswami 负责提供算子网络设计方面的专业指导、材料准备及手稿审阅。Daniel Smith 为数据生成、问题设定及手稿审阅提供了必要指导。George Karniadakis 对手稿及本研究采用的方法提出了关键反馈意见。

参考文献

[1] AVL，AVL Boost 发动机仿真，https://www.avl.com/boost，访问日期：2022-08-08。[2] Gamma Technologies，GT Power 发动机仿真，https://www.gtisoft.com/gt-power/，访问日期：2022-08-08。[3]Ricardo Inc，WAVE 1D 仿真，https://software.ricardo.com/products/wave，访问日期：2022-08-08。[4] E. Hendricks，《大型涡轮增压二冲程柴油发动机的紧凑而全面的模型》，SAE Transactions（1986）820–834。[5]N. Watson，用于电子控制系统开发的动态涡轮增压柴油发动机仿真器，1984。[6] F. Kimmich、A. Schwarte、R. Isermann，采用基于信号和过程模型的方法对现代柴油发动机进行故障检测，Control Engineering Practice 13 (2)（2005）189–203。[7] H. Wu、X. Wang、R. Winsor、K. Baumgard，采用 GT-Power 一维详细模型的柴油发动机平均值建模，技术报告，SAE 技术论文（2011）。[8] C. Svard、M. Nyberg，将具有混合因果性的计算序列用于汽车系统故障诊断的残差生成器，《IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans》40 (6)（2010）1310–1328。[9] Z. Han、R. D. Reitz，使用 RNG 模型对内燃机进行湍流建模 κ-ε，Combustion Science and Technology 106 (4-6)（1995）267–295。

[10] S. Goswami、C. Anitescu、T. Rabczuk，采用深度能量最小化进行自适应四阶相场分析，理论与应用断裂力学 107（2020）102527。[11] S. Goswami、M. Yin、Y. Yu、G. E. Karniadakis，一种用于预测准脆性材料裂纹路径的物理信息变分 DeepONet，计算力学与工程方法 391（2022）114587。[12] Y. LeCun、Y. Bengio、G. Hinton，《深度学习》，《自然》521 (7553)（2015）436–444。[13] J. Wahlström、L. Eriksson，通过优化模型参数捕捉非线性系统动力学，对配备可变几何涡轮增压器和废气再循环的柴油发动机进行建模，机械工程师学会会刊 D辑：汽车工程杂志 225 (7)（2011）960–986。[14] J. B. Dabney、T. L. Harman，《精通 Simulink》，第230卷，培生／普伦蒂斯·霍尔，上萨德尔河镇，2004年。[15]L. Lu、P. Jin、G. Pang、Z. Zhang、G. E. Karniadakis，基于算子的通用逼近定理，通过 DeepONet 学习非线性算子，《自然·机器智能》3 (3)（2021）218–229。[16] S. Goswami、A. Bora、Y. Yu、G. E. Karniadakis，《物理信息神经算子》，arXiv 预印本 arXiv:2207.05748（2022）。[17] C. Lin、Z. Li、L. Lu、S. Cai、M. Maxey、G. E. Karniadakis，用于预测多尺度气泡生长动力学的算子学习，《计算物理学杂志》154 (10)（2021）104118。[18]S. Cai、Z. Wang、L. Lu、T. A. Zaki、G. E. Karniadakis，DeepM&Mnet：基于神经网络算子逼近推断电对流多物理场，《计算物理学杂志》436（2021）110296。[19] T. Chen、H. Chen，具有任意激活函数的神经网络对非线性算子的通用逼近及其在动力系统中的应用，《IEEE神经网络汇刊》6 (4)（1995）911–917。[20] P. Jin、S. Meng、L. Lu，MIONet：通过张量积学习多输入算子，arXiv 预印本 arXiv:2202.06137（2022）。[21] S. Goswami、D. S. Li、B. V. Rego、M. Latorre、J. D. Humphrey、G. E. Karniadakis，通过神经算子学习促成主动脉瘤的异质性机械生物学损伤因素，arXiv 预印本 arXiv:2205.03780（2022）。[22] Y. Gal、Z. Ghahramani，Dropout 作为贝叶斯近似：表示深度学习中的模型不确定性，载于：国际机器学习会议，PMLR，2016年，第1050–1059页。[23]L. McClenny、U. Braga-Neto，采用软注意力机制的自适应物理信息神经网络，arXiv 预印本 arXiv:2009.04544（2020）。[24]K. Kontolati、S. Goswami、M. D. Shields、G. E. Karniadakis，过度参数化对基于流形的代理模型和深度神经算子的影响，arXiv 预印本 arXiv:2203.05071（2022）。

[25]D. J. MacKay，反向传播网络的实用贝叶斯框架，《神经计算》4 (3) (1992) 448–472。[26] L. V. Jospin，H. Laga，F. Boussaid，W. Buntine，M. Bennamoun，动手学习贝叶斯神经网络——面向深度学习用户的教程，IEEE 计算智能杂志 17 (2) (2022年) 29–48。[27]A. F. Psaros，X. Meng，Z. Zou，L. Guo，G. E. Karniadakis，科学机器学习中的不确定性量化：方法、指标与比较，arXiv预印本 arXiv:2201.07766 (2022年)。[28] N. Srivastava，G. Hinton，A. Krizhevsky，I. Sutskever，R. Salakhutdinov，Dropout：一种防止神经网络过拟合的简单方法，机器学习研究期刊 15 (1) (2014) 1929–1958。[29] W. K. Hastings，使用马尔可夫链的蒙特卡罗采样方法及其应用，牛津大学出版社，1970年。[30] R. Bardenet，A. Doucet，C. C. Holmes，关于高维数据的马尔可夫链蒙特卡罗方法，《机器学习研究期刊》18 (47) (2017)。[31] R. M. Neal 等，基于哈密顿动力学的马尔可夫链蒙特卡罗，《马尔可夫链蒙特卡罗手册》2 (11) (2011年) 2。[32] C. Blundell，J. Cornebise，K. Kavukcuoglu，D. Wierstra，神经网络中的权重不确定性，载于：国际机器学习会议，PMLR，2015年，第1613–1622页。[33] J. M. Hernández-Lobato，R. Adams，用于贝叶斯神经网络可扩展学习的概率反向传播，载于：国际机器学习会议，PMLR，2015年，第1861–1869页。[34] D. M. Blei，A. Kucukelbir，J. D. McAuliffe，变分推断：统计学家综述，《美国统计学会杂志》112 (518) (2017) 859–877。[35] X. Meng，L. Yang，Z. Mao，J. del Águila Ferrandis，G. E. Karniadakis，从数据和物理学中学习函数先验与后验分布，《计算物理学杂志》457 (2022年) 111073。[36] M. Raissi，P. Perdikaris，G. E. Karniadakis，物理信息神经网络：用于求解涉及非线性偏微分方程的正问题和反问题的深度学习框架，《计算物理学杂志》378 (2019) 686–707。