energies

MDPI

Article

Learning Flame Evolution Operator under Hybrid Darrieus Landau and Diffusive Thermal Instability

Rixin Yu [1,*], Erdzan Hodzic [2] and Karl-Johan Nogenmyr [3]

[1] Department of Energy Sciences, Lund University, 221 00 Lund, Sweden
[2] Department of Manufacturing Processes, RISE Research Institutes of Sweden, 553 22 Jonkoping, Sweden
[3] Siemens Energy AB, 612 31 Finspång, Sweden
* Correspondence: rixin.yu@energy.lth.se

Abstract: Recent advancements in the integration of artificial intelligence (AI) and machine learning (ML) with physical sciences have led to significant progress in addressing complex phenomena governed by nonlinear partial differential equations (PDEs). This paper explores the application of novel operator learning methodologies to unravel the intricate dynamics of flame instability, particularly focusing on hybrid instabilities arising from the coexistence of Darrieus–Landau (DL) and Diffusive–Thermal (DT) mechanisms. Training datasets encompass a wide range of parameter configurations, enabling the learning of parametric solution advancement operators using techniques such as parametric Fourier Neural Operator (pFNO) and parametric convolutional neural networks (pCNNs). Results demonstrate the efficacy of these methods in accurately predicting short-term and long-term flame evolution across diverse parameter regimes, capturing the characteristic behaviors of pure and blended instabilities. Comparative analyses reveal pFNO as the most accurate model for learning short-term solutions, while all models exhibit robust performance in capturing the nuanced dynamics of flame evolution. This research contributes to the development of robust modeling frameworks for understanding and controlling complex physical processes governed by nonlinear PDEs.

Keywords: machine learning; operator learning; convolutional neural network; fourier neural operator; partial differential equation; intrinsic flame instability

Citation: Yu, R.; Hodzic, E.; Nogenmyr, K.-J. Learning Flame Evolution Operator under Hybrid Darrieus Landau and Diffusive Thermal Instability. *Energies* **2024**, *17*, 3097. https://doi.org/10.3390/en17133097

Academic Editor: Albert Ratner

Received: 11 May 2024
Revised: 11 June 2024
Accepted: 20 June 2024
Published: 23 June 2024

1. Introduction

In recent years, the integration of artificial intelligence (AI) and machine learning (ML) with natural sciences and physical engineering has led to significant advancements, particularly in addressing the complexities of nonlinear partial differential equations (PDEs). These equations are fundamental in understanding various physical phenomena, ranging from turbulent fluid dynamics to complicated physico-chemical processes. Within the domain of nonlinear PDE systems lies a rich tapestry of intricate dynamics, including instabilities, multiscale interactions, and chaotic behaviors. To enhance predictive capabilities and design robust control strategies in engineering applications, computational methods are indispensable. These methods, often in the form of numerical solvers, enable the accurate simulation of PDE solutions across spatial and temporal domains. Implicit in these solvers is the concept of the functional mapping operator, which could iteratively advance the PDE solution functions in time, providing a pathway to explore the evolution of physical systems over extended durations. A distinctive class of machine learning methods has emerged, capable of learning and replicating the behavior of these PDE operators.

Recent advancements have seen the proliferation of operator learning methods, each offering unique insights and capabilities. Early efforts in this domain drew inspiration from deep convolutional neural networks (CNNs) [1–7], employing techniques from computer vision. These CNN-based approaches parameterize the PDE operator in a finite-dimensional

space, enabling the mapping of discrete functions onto image-like representations. Building upon this foundation, recent strides have witnessed the development of neural operator methods [8,9] capable of learning operators of infinite dimensionality. Notable examples include Deep Operator Network [10] and the Fourier Neural Operator (FNO) [11], both demonstrating remarkable proficiency across a diverse array of benchmark problems [12,13]. Furthermore, recent advancements have extended neural operators by amalgamating concepts from wavelet methods [14,15] and adapting approaches for complex domains [16].

In our recent investigations [17,18], we delved into the intricate dynamics of flame instability and nonlinear evolution, a canonical problem with profound implications for combustion science. Flames can undergo destabilization due to intrinsic instabilities, including the hydrodynamic Darrieus–Laudau (DL) mechanism [19,20] attributed to density gradients across a flame, and the Diffusive–Thermal (DT) mechanism [21,22] driven by heat and reactant diffusion disparities. Our previous work [17] primarily focused on DL flames, scrutinizing the evolution of unstable flame fronts within periodic channels of varying widths. Under DL instability, an initially planar flame morphs into a steady curved front; as the channel width increases, the curved front becomes sensitive to random noise, and small wrinkles start to emerge. At sufficiently large channels, DL flames give rise to complicated fractal fronts characterized by hierarchical cascading of cellular structures [23].

The nonlinear evolution of DT flame development can be modeled by the Michelson–Sivashinsky equation [24], while a more accurate but computationally expensive approach involves direct numerical simulation (DNS) of Navier–Stokes equations. Utilizing these two approaches to generate training datasets, our investigations [17] demonstrated that both CNNs and FNO could effectively capture the evolution of DL flames, with FNO exhibiting superior performance in modeling complex flame geometries over longer durations. Subsequently, we embarked on developing parameterized learning methodologies capable of encapsulating dynamics across diverse parameter regimes within a single network framework. Through the introduction of pCNNs and pFNO models [18], we demonstrated their efficacy in replicating the behavior of DL flames across varying channel widths. Additionally, our methods have shown success in learning the parametric solutions of the Kuramoto–Sivashinsky equation [25], which models unstable flame evolution due to the DT mechanism. However, a challenge remains in mitigating the tendency of these models to overestimate noise effects.

In this paper, we extend our research horizon to encompass the complexities arising from hybrid instabilities, specifically those arising from the coexistence of DL and DT mechanisms. These hybrid systems pose novel challenges, as they embody a rich spectrum of behaviors stemming from the interplay of distinct instability modes. Leveraging our recently developed operator learning methodologies, we aim to unravel the nuanced dynamics underlying such hybrid instabilities, shedding light on their short-term evolution and long-term statistical properties. Furthermore, our endeavor holds promise for the development of robust modeling frameworks capable of capturing the intricate dynamics of real-world flame evolution scenarios.

The paper is organized as follows: first, we describe the problem setup for learning PDE operators, followed by brief descriptions of the two parametric learning methods to be used in this work. These methods will be compared in the context of learning parametric-dependent solution time–advance operators for the Sivashinsky Equation [26], which models unstable front evolution due to hybrid mechanisms of flame instability. Finally, we provide a summary and conclusion.

2. Problem Setup for Learning PDE Operators

In this section, we delineate the problem setup for learning a parametric PDE operator, along with a description of recurrent training methods.

Consider a system governed by PDEs, typically involving multiple functions and mappings between them. Our focus here is on a parametric operator mapping, denoted as

$$\hat{\mathcal{G}} : \mathcal{V} \times \mathbb{R}^{d_\gamma} \to \mathcal{V}'; (v(x), \gamma) \mapsto v'(x') \tag{1}$$

where $\gamma \in \mathbb{R}^{d_\gamma}$ represents a set of parameters. The input function is $v(x)$, where $x \in \mathcal{D}$ resides in a functional space $\mathcal{V}(\mathcal{D}, \mathbb{R}^{d_v})$ with domain $\mathcal{D} \subset \mathbb{R}^d$ and co-domain $\mathbb{R}^{d_v}$ while the output function is $v'(x')$, where $x' \in \mathcal{D}'$ belongs to another functional space $\mathcal{V}'(\mathcal{D}', \mathbb{R}^{d'_v})$ with domain $\mathcal{D}' \subset \mathbb{R}^{d'}$ and co-domain $\mathbb{R}^{d'_v}$.

Our primary interest lies in the solution time advancement operator with parametric dependence, given by

$$\hat{\mathcal{G}} : (\phi(x, \bar{t}), \gamma) \mapsto \phi(x; \bar{t} + 1) \tag{2}$$

where $\phi(x; \bar{t})$ denotes the solution to a PDE under parameters γ, and $\bar{t} = t/\Delta_t$ represents normalized time with a small time increment Δ_t. For simplicity, we assume identical domain and co-domain for both input and output functions, i.e., $\mathcal{D}' = \mathcal{D}$, $\mathcal{V}' = \mathcal{V}$, $d' = d$, and $d'_v = d_v$, with periodic boundary conditions on $\mathcal{D}$.

To approximate the mapping $\hat{\mathcal{G}}$ using neural network methods, let Θ denote the space of trainable parameters in the network. A neural network can be defined as

$$\mathcal{G} : \mathcal{V} \times \mathbb{R}^{d_\gamma} \times \Theta \to \mathcal{V}' \text{ or equivalently } \mathcal{G}_\theta : \mathcal{V} \times \mathbb{R}^{d_\gamma} \to \mathcal{V}', \theta \in \Theta. \tag{3}$$

where Θ represents the space of network parameters. Training the neural network involves finding an optimal choice of parameters $\theta^* \in \Theta$ such that $\mathcal{G}_{\theta^*}$ approximates $\hat{\mathcal{G}}$.

Starting with an initial solution function $\phi(x; t_0)$ under fixed parameter values γ, the recurrent application of the operator $\mathcal{G}_{\theta,\gamma} := \mathcal{G}_\theta(\cdot, \gamma)$ can roll out predicted solutions of arbitrary length by iteratively updating the input function with its output from the previous prediction. Note that while the learned operator is expected to make accurate short-term predictions, its long-term prediction might be allowed to deviate if the ground truth PDE admits chaotic solutions. On the other hand, it is still desirable that the learned operator can reproduce the correct statistics in the long-term solutions.

Following previous studies [17,18], our training approach adopts a one-to-many setup where the recurrent network is trained to make multiple successive predictions from a single input function. Such a setup ensures numerical stability in the learned solution advancement operator, a crucial consideration highlighted in the prior work [17,18]. More specifically, let $\left\{ v_j, (\hat{\mathcal{G}}^1_{\gamma_i} v_j, \hat{\mathcal{G}}^2_{\gamma_i} v_j, ..., \hat{\mathcal{G}}^n_{\gamma_i} v_j) \right\}_{j=1,i=1}^{j=Z',i=Z''}$ be a total $(Z' \times Z'')$ number of training data arranged as input/output pairs in the 1-to-n manner, and an operator with a superscript n denotes its repeated application n times, e.g., $\hat{\mathcal{G}}^n_\gamma := \hat{\mathcal{G}}_\gamma \circ ... \circ \hat{\mathcal{G}}_\gamma$. Training a network $\mathcal{G}_\theta$ to approximate $\hat{\mathcal{G}}$ then becomes a minimization task

$$\min_{\theta \in \Theta} \mathbb{E}_{v \sim \chi', \gamma \sim \chi''} \left[C((\mathcal{G}^1_{\theta,\gamma} v, ..., \mathcal{G}^n_{\theta,\gamma} v), (\hat{\mathcal{G}}^1_\gamma v, , ..., \hat{\mathcal{G}}^n_\gamma v)) \right] \tag{4}$$

where $v \sim \chi'$ and $\gamma \sim \chi''$ are randomly drawn according to independent probability measures of χ' and χ'', respectively. The cost function $C : \mathcal{V}^n \times \mathcal{V}^n \to \mathbb{R}$ is set to the relative mean square (L2) error of $C(x, y) = ||x - y||_2 / ||y||_2$; here, $\mathcal{V}^n$ abbreviates the Cartesian product of n copies of $\mathcal{V}$.

3. Parametric Operator Learning Methods

In this section, we present a concise overview of two methods capable of learning the parametric operator $\hat{\mathcal{G}}$. Further details about these methods can be found in paper [18].

3.1. Parametric Convolutional Neural Network (pCNN)

The operator $\mathcal{G}_{\theta,\gamma}$ can be regarded as an image-to-image map when applied for the temporal advancement of discretized solutions. Deep Convolutional Neural Networks

(CNNs) have demonstrated effectiveness in image-learning tasks. The network architecture suitable for learning operators resembles a convolutional auto-encoder similar to that in U-Net [6] and ConvPDE [7]. This network comprises an encoder block and a decoder block, with input data passing through a series of transformations via convolutional layers. Additionally, the method incorporates side networks to handle additional parameter inputs. The pCNN model is outlined in Figure 1.

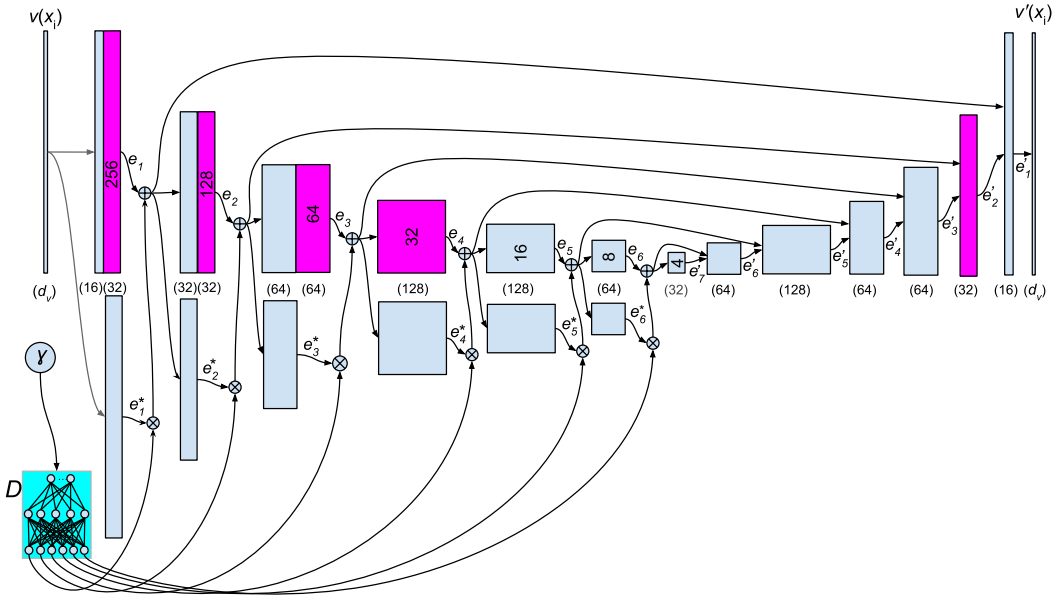

Figure 1. The parametric CNN model adopted in this study is demonstrated for input function $v(x_i)$ discretized at the 1D mesh of 256 points, with $L = 6$ levels of encoding and decoding. Standard convolution layers are represented by gray rectangles, while Inception layers are depicted in magenta. The output data channels c_l for each convolution layer are indicated within brackets.

Let e_0^+ denote the input function $v(x_j)$ represented on an x-mesh. The encoder block follows an iterative update procedure: $e_l^+ \mapsto (e_{l+1}, e_{l+1}^*) \mapsto e_{l+1}^+$. This iteration occurs over the level sequence $l = 0, 1, ..., L-1$. Denote the last decoding output as $e'_L = e_L^+$, a subsequent decoding procedure is applied $(e'_{l+1}, e_l^+) \mapsto e'_l$ through reversing level l.

Here, $e_l, e_l^*, e_l^+, e'_l \in \mathbb{R}^{c_l \times N_l}$ represent four data sequences, each with c_l channels and size of N_l. The data size is halved as $N_{l+1} = N_l/2$ for $l \geq 1$. The first-stage encoder contains two sub-maps of $e_l^+ \mapsto e_{l+1}$ and $e_l^+ \mapsto e_{l+1}^*$; both are implemented using vanilla-stacked convolution layers (with a filter size of 3, stride 1, periodic padding, and ReLU activation). Some layers are replaced by Inception layers for improved performance. Additionally, a size 2 max-pooling layer is prepended to halve the image size for $l \geq 1$. The second-stage encoder map is implemented as $e_{l+1}^+ = e_{l+1} + e_{l+1}^* \cdot D_l(\gamma)$. Here, D_l is a simple function (a two-layer perceptron) that converts the PDE parameters γ into a scaling ratio. The decoder update $(e'_{l+1}, e_l^+) \mapsto e'_l$ involves concatenating e'_{l+1} (but up-sample it to double its size) with e_l^+ along the channel dimension. The final output is obtained as $v'(x_j) = e'_1$.

3.2. Parametric Fourier Neural Operator (pFNO)

The parametric Fourier Neural Operator (pFNO) [18] was developed based on the original FNO method [11], wherein learning for the infinite-dimensional operator is achieved by parameterizing the integral kernel operators in Fourier Space. The pFNO adopts an architecture of maps-composition as $\mathcal{G} = \mathcal{Q} \circ \mathcal{H}_L \circ ... \circ \mathcal{H}_1 \circ \mathcal{P} \circ \mathcal{C}$, comprising a concatenation map $\mathcal{C}$, a lifting map $\mathcal{P}$, a sequence of hidden maps $\mathcal{H}_l$ for $l = 1, 2, \ldots, L$, and a projection map $\mathcal{Q}$.

The first map $\mathcal{C} : \mathcal{V} \times \mathbb{R}^{d_\gamma} \to \mathcal{V}^c(\mathcal{D}; \mathbb{R}^{d_v + d_\gamma}); (v(x), \gamma) \mapsto v^c(x)$ simply concatenates the parameters γ to the co-dimension of input function $v(x)$, yielding $v^c(x)$. The second

map $\mathcal{P} : \mathcal{V}^c \to \mathcal{V}^*; v^c(x) \mapsto \varepsilon_0(x)$, lifts the input to a higher-dimensional functional space $\mathcal{V}^* := \mathcal{V}^*(\mathcal{D}; \mathbb{R}^{d_\varepsilon})$ with $d_\varepsilon > d_v + d_\gamma$. The subsequent hidden maps $\mathcal{H}_l : \mathcal{V}^* \times \mathbb{R}^{d_\gamma} \to \mathcal{V}^* : (\varepsilon_{l-1}, \gamma) \mapsto \varepsilon_l$ act sequentially to update $\varepsilon_0 \mapsto \varepsilon_1 \mapsto \ldots \mapsto \varepsilon_L$ for all $\varepsilon_l \in \mathcal{V}^*$. Finally, the map $\mathcal{Q} : \mathcal{V}^* \to \mathcal{V}'; \varepsilon_L(x) \mapsto v'(x)$ projects back to low-dimension functional space, finally yielding $v'(x)$.

Both $\mathcal{P}$ and $\mathcal{Q}$ are implemented using simple multilayer perceptrons(MLP). The hidden maps $\mathcal{H}_{l+1}$ are implemented as parametric Fourier layers:

$$\varepsilon_{l+1} = \sigma\Big(W_l \varepsilon_l + b_l + \mathcal{F}^{-1}\{\mathfrak{R}_l^*(\mathcal{F}\{\varepsilon_l\}, \gamma)\}\Big) \tag{5}$$

where $W_l \in \mathbb{R}^{d_\varepsilon \times d_\varepsilon}$ and $b_l \in \mathbb{R}^{d_\varepsilon}$ are learnable weights and biases, respectively, and σ is a ReLU activation function. Here, $\mathcal{F}$ and $\mathcal{F}^{-1}$ represent the Fourier Transform and its inverse, respectively. The function $\mathfrak{R}_l^* : \mathbb{C}^{\kappa^{max} \times d_\varepsilon} \times \mathbb{R}^{d_\gamma} \to \mathbb{C}^{\kappa^{max} \times d_\varepsilon}$ acts on the truncated Fourier modes, transforming them as :

$$\mathfrak{R}_l^*(\mathcal{F}\{\varepsilon\}, \gamma)_{\kappa,i} = \sum_{j=1}^{d_\varepsilon} [(R_l)_{\kappa,i,j} + (R_l^*)_{\kappa,i,j} D_l^*(\gamma)_\kappa] \mathcal{F}\{\varepsilon\}_{\kappa,j},$$
$$\kappa = 0, 1, ..., \kappa^{max} \text{ and } i = 1, ..., d_\varepsilon \tag{6}$$

where $R_l, R_l^* \in \mathbb{C}^{\kappa^{max} \times d_\varepsilon \times d_\varepsilon}$ are two learnable weight tensors and $D_l^* : \mathbb{R}^{d_\gamma} \to \mathbb{R}^{\kappa^{max}}$ is a function converting the parameters γ into κ^{max}-number of scaling ratios. This function consists of a two-stage map $\gamma \mapsto D_l(\gamma) \mapsto D_l^*(\gamma)$, with $D_l(\gamma) \in \mathbb{R}^{N_D}$ outputting N_D scaling ratios and implemented as an MLP. The second map hierarchically redistributes these ratios across the wave numbers. In one dimension($d = 1$), the distribution map reads: $D_l^*(\gamma)_\kappa = D_l(\gamma)_i$ for $\kappa \in (\frac{\kappa^{max}}{2^{i+1}}, \frac{\kappa^{max}}{2^i}]$ at $i = 0, .., N_D - 2$, and, for $\kappa \in (0, \frac{\kappa^{max}}{2^{N_D-1}}]$ at $i = N_D - 1$.

One might observe that we can deactivate the second weight tensor R_l^* in Equation (6) by enforcing the map $D_l^*(\gamma)$ to output only zeros. This modification still enables learning of the parameter operator due to the concatenation map $\mathcal{C}$. Such a modified method can viewed as a simple tweak to the baseline method of FNO [11] and will be referred as pFNO* in a later section.

4. Numerical Experiments and Result Discussions

In this section, we employ the pFNO and pCNN methods to learn flame evolution under hybrid instabilities arising from both Darrieus–Landau (DL) [19,20] and Diffusive–Thermal (DT) [21,22] mechanisms. The dynamics of such unstable flame development are encapsulated by the Sivashinsky equation [26]. To facilitate parametric learning, we begin by reformulating the Sivashinsky equation, introducing two parameters that enable straightforward specification for blending the two instabilities and controlling the largest unstable wave numbers. By sampling across these parameters, we construct an extensive training dataset covering a range of relevant scenarios subjected to different DL/DT mixing. Subsequently, we present the results and compare the performance of the different methods in learning these hybrid instabilities.

4.1. Governing Equations

Consider modeling the unstable development of a statistically planar flame front. Let $\hat{t}$ denote time and $\hat{x}$ represent the spatial coordinate along the normal direction of flame propagation. Introduce a displacement function $\psi(\hat{x}, \hat{t}) : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ describing the stream-wise coordinate of a flame front undergoing intrinsic flame instabilities. Such evolution can be modeled by the Sivashinsky equation [26]:

$$\psi_{\hat{t}} + \frac{1}{2}(\psi_{\hat{x}})^2 = -4(1 + \mathrm{Le}^*)^2 \psi_{\hat{x}\hat{x}\hat{x}\hat{x}} - \mathrm{Le}^* \psi_{\hat{x}\hat{x}} + (1 - \Omega)\Gamma(\psi) \tag{7}$$

where $\Gamma : \psi \mapsto -\mathcal{H}(\psi_{\hat{x}})$ is a linear singular non-local operator defined using the Hilbert transform $\mathcal{H}$, or equivalently written as $\Gamma : \psi \mapsto \mathcal{F}^{-1}(|\kappa|\mathcal{F}_\kappa(\psi))$ using the spatial Fourier transform $\mathcal{F}_\kappa(\psi)$ and its inverse $\mathcal{F}^{-1}$.

In Equation (7), Ω is the density ratio between burned product and fresh reactant; Le^* is a ratio (positive or negative) depending on the Lewis number of deficient reactant and another critical Lewis number. Introduce three constants (a, b, c) for variable transformation on time $t = a^2\hat{t}$, space $x = b^{-2}\hat{x}$ and the displacement function $\psi(\hat{x}, \hat{t}) = c^2\phi(x, t)$, then Equation (7) can be rewritten as

$$\frac{1}{\tau}\phi_t + \frac{1}{2\beta^2}(\phi_x)^2 = -\frac{\mu}{\beta^4}\phi_{xxxx} - \frac{\nu}{\beta^2}\phi_{xx} + \frac{\rho}{\beta}\Gamma(\phi) \tag{8}$$

with $\beta = bc^{-1}$, $\nu = \mathrm{Le}^*/c^2$, $\rho = (1-\Omega)bc$, $\mu = 4(1+\mathrm{Le}^*)^2 b^{-2}c^{-4}$ and $\tau = a^{-2}b^{-2}$.

In this work, we consider the flame front solution $\phi(x, t)$ of Equation (8) in a channel domain subjected to periodic boundary condition, i.e., $x \in \mathcal{D} = (\pi, \pi]$. One might notice that Equation (8) admits a zero equilibrium solution being a flat flame (i.e., $\phi^*(x, t) = 0$); a perturbation analysis around this zero solution yields a linear dispersion relation

$$\frac{\omega(\kappa)}{\tau} = -\mu\left(\frac{\kappa}{\beta}\right)^4 + \nu\left(\frac{\kappa}{\beta}\right)^2 + \rho\left(\frac{\kappa}{\beta}\right), \ \forall\kappa = 0, 1, 2, \ldots \tag{9}$$

with the perturbed solution being $\phi(x, t) = \sum_\kappa \hat{\phi}_\kappa(t)e^{i\kappa x} + \hat{\phi}^*_\kappa(t)e^{-i\kappa x}$ (superscript * denotes complex conjugate) and the Fourier mode of perturbation evolving as $\hat{\phi}_\kappa(t) \approx \hat{\phi}_\kappa(0) \cdot e^{\omega(\kappa)\cdot t}$.

Equations (8) and (9) present a straightforward approach to hybridizing two flame instabilities of DT and DL mechanisms. This strategy is accomplished by specifying two parameters, ρ and β, while the remaining parameters (μ, ν, and τ) can be determined by additional constraints outlined below. Initially, the parameter ρ (in between 0 and 1) is defined to allow for the continuous blending of these two instabilities. When $\rho = 1$, the Sivashinsky Equation (8) yields a pure DL instability described by the Michelson–Sivashinsky (MS) equation [24]:

$$\frac{1}{\tau}\phi_t + \frac{1}{2\beta^2}(\phi_x)^2 = \frac{1}{\beta^2}\phi_{xx} + \frac{1}{\beta}\Gamma(\phi) \tag{10}$$

whereas, at the other end ($\rho = 0$), it recovers the pure DT instability as described by the Kuramoto–Sivashinsky (KS) equation [25]:

$$\frac{1}{\tau}\phi_t + \frac{1}{2\beta^2}(\phi_x)^2 = -\frac{1}{\beta^2}\phi_{xx} - \frac{1}{\beta^4}\phi_{xxxx}. \tag{11}$$

Secondly, the parameter β is determined as the largest value for which the dispersion relation of Equation (9) equals zero (i.e., $\omega(\beta) = 0$). This definition yields $\nu = \mu - \rho$. Consequently, we can prescribe β to establish the largest unstable wave number. To mitigate variability in the remaining parameters, a third constraint is imposed: the maximum value of $\omega(\kappa)$ over the interval $0 < \kappa < \beta$ must be $1/4$. Furthermore, $\tau = \rho\beta/10 + (1-\rho)$ is employed to better accommodate the timescales attributed to the various hybrid instabilities. This strategy allows for the determination of all remaining parameters given the values of ρ and β. This is illustrated in Figure 2, which presents dispersion relation plots and associated parameters.

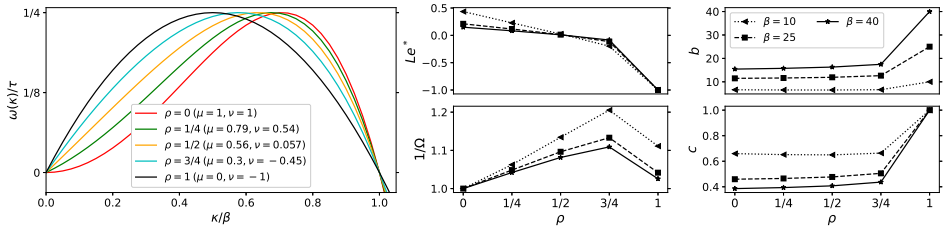

Figure 2. Dispersion relations (left) and relevant parameter values at prescribed values of ρ and β.

Before proceeding further, it may be worthwhile to mention a few well-known results. The KS Equation (11) is often utilized as a benchmark example for PDE learning studies and is renowned for exhibiting chaotic solutions at large β. On the other hand, the MS Equation (10), although less familiar outside the flame instability community, can be precisely solved using a pole decomposition technique [27], transforming it into a set of ODEs with finite freedoms. Moreover, at large β, the MS equation admits a stable solution in the form of a giant cusp front. However, at smaller β, the equation becomes susceptible to noise, resulting in unstable solutions characterized by persistent small wrinkles atop a giant cusp. Additional details about known theory can be found in references [23,28–35].

4.2. Training Dataset

Equation (8) is tackled using a pseudo-spectral approach combined with a Runge–Kutta (4,5) time integration method. All solutions are computed on a uniformly spaced 1D mesh consisting of 256 points. Training datasets are generated for a total of 15 parametric configuration tuples (ρ, β), formed as the Cartesian product of three values for β in the range $[10, 25, 40]$ and five values for ρ in the range $[0, 1/4, 1/2, 3/4, 1]$. For each of the fifteen parametric configurations, we generate 250 sequences of short-duration solutions, as well as a single sequence of long-duration solutions. Each short solution sequence spans a time duration of $0 \leq t \leq 75$ and contains 500 consecutive solutions separated by a time interval of $\Delta_t = 0.15$. Additionally, each sequence starts from random initial conditions $\phi_0(x)$ sampled from a uniform distribution over the range $[0, 0.03]$. The long sequence covers a time duration of $0 \leq t \leq 18,750$ and comprises 125,000 consecutive solutions outputted at the same interval Δ_t. A validation dataset is similarly created for all fifteen parameter tuples, but it contains only 10 percent of the data present in the training dataset.

4.3. Result Analysis

The training datasets described in the previous section are utilized to train parametric solution advancement operators, denoted as $\hat{\mathcal{G}}_{(\gamma)} : \phi(x;t) \mapsto \phi(x;t+\Delta_t)$ with $\gamma := (\rho, \beta)$ and $d_\gamma = 2$. As a reminder, one ending value of ρ = 0 enables the pure DT instability while the other ending value of $\rho = 1$ activates the pure DL instability.

In this study, three models—pFNO, pFNO*, and pCNN—described in Section 2 are employed to learn the two-parameter dependent operator $\hat{\mathcal{G}}_{(\rho,\beta)}$. As explained in the last paragraph in Section 2, pFNO* is a simple variant of the baseline FNO method [11] that includes the parameters in the co-domain of the input function. On the other hand, pCNN has shown poor performance in learning the full operator $\hat{\mathcal{G}}_{(\rho,\beta)}$, with a high training error exceeding 3 percent; see Table 1. Therefore, we resort to two slightly restricted models (pCNN10 and pCNN40), which learn the single parameter (ρ) dependent operators, $\hat{\mathcal{G}}_{(\rho,\beta=10)}$ and $\hat{\mathcal{G}}_{(\rho,\beta=40)}$, with each model being trained using one-third of the total dataset at ρ = 10 and 40, respectively.

Table 1. Relative $L2$ train/validation errors for all operator learning networks.

Model	Parameter Configurations (β, ρ)	Train $L2$	Valid. $L2$
pFNO*	$[10, 25, 40] \times [0, 1/4, 1/2, 3/4, 1]$	0.0098	0.010
pFNO	$[10, 25, 40] \times [0, 1/4, 1/2, 3/4, 1]$	0.0071	0.0073
pCNN	$[10, 25, 40] \times [0, 1/4, 1/2, 3/4, 1]$	0.036	0.037
pCNN10	$[10] \times [0, 1/4, 1/2, 3/4, 1]$	0.011	0.012
pCNN40	$[40] \times [0, 1/4, 1/2, 3/4, 1]$	0.022	0.022

The learned operator at given parameters is expected to make recurrent predictions of solutions over an extended period. The training for such operators aims not only for accurate short-term predictions but also for robust predictions of long-term solutions with statistics similar to the ground truth. As demonstrated in previous studies [17,18], achieving this involves organizing the training data in a 1-to-20 pair, as expressed in Equation (4), optimized for accurately predicting 20 successive steps of outputs from a single input over a range of parameter values.

Table 1 presents the relative training/validation errors for various models. The validation errors in Table 1 are consistent with those reported in our previous work [17,18]. Additional details on training and model hyper-parameters are provided in Appendix A.

Figure 3 compares two randomly initialized sequences of front displacements predicted by two models (pFNO* and pCNN10) against the reference solutions at $\rho = [0, 1/4, 1/2, 3/4, 1]$ and $\beta = 10$. A similar comparison for pFNO* and pCNN40 at $\beta = 40$ is shown in Figure 4. Additionally, Figures 5 and 6 depict similar comparisons for the predicted front slope (ϕ_x) at $\beta = 10$ and 40, respectively.

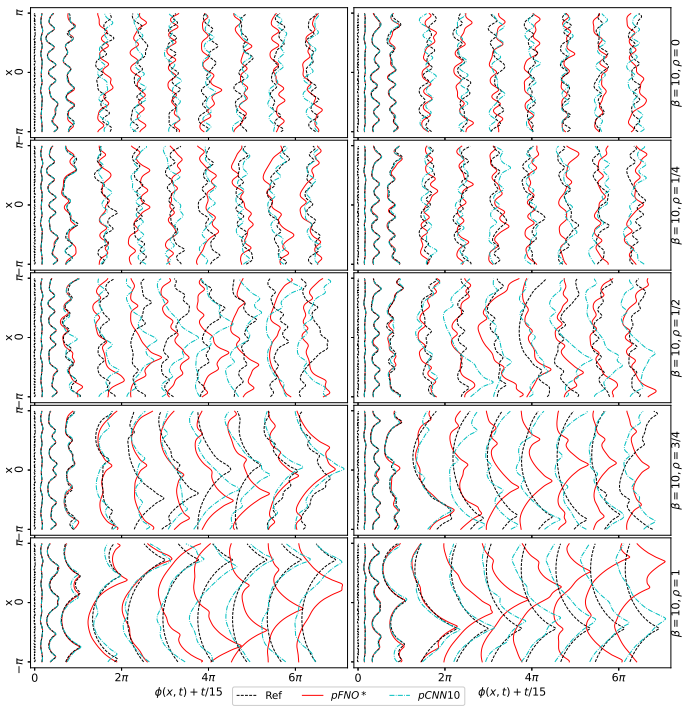

Figure 3. Long-term solutions of flame front displacement $\phi(x, t)$ at $\beta = 10$ and $\rho = [0, 1/4, 1/2, 3/4, 1]$ (from top to bottom row). Black reference solutions to Equation (8) obtained using high-order numerical methods are compared against predictions by the operator learning methods of pFNO* (red) and pCNN (cyan). The left and right columns correspond to two randomly initialized solution sequences, each showing eleven snapshots of $\phi(x, t)$ at $t/0.15 = 0$, 50, 125, 250, 500, 750, 1000, 1250, 1500, 1750 and 2000. A time shift ($t/15$) is added to the displayed fronts to avoid overlap.

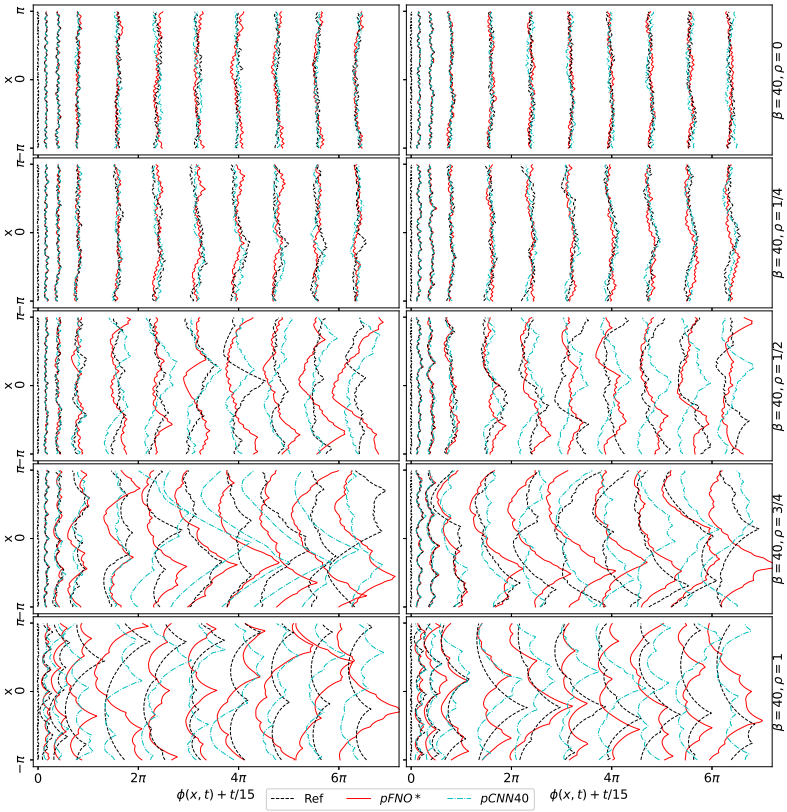

Figure 4. Comparison of flame front displacement predicted by pFNO* and pCNN40 at $\beta = 40$. Other details are the same as in Figure 3.

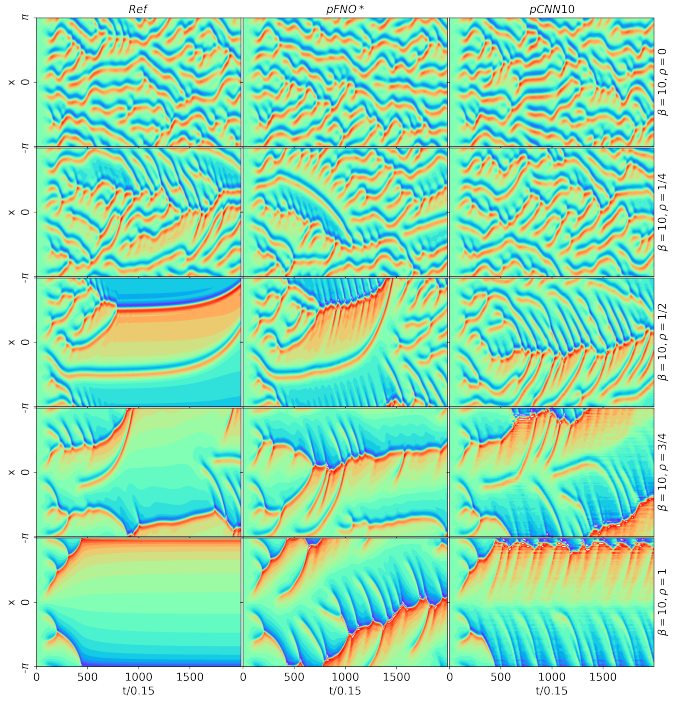

Figure 5. Comparison of front slope $\phi_x(x,t)$ calculated from a single instance reference solution (first column) of Equation (10) at $\beta = 10$ and $\rho = [0, 1/4, 1/2, 3/4, 1]$, against predictions by pFNO* and pCNN (last two columns). Rainbow color indicates values from negative to positive.

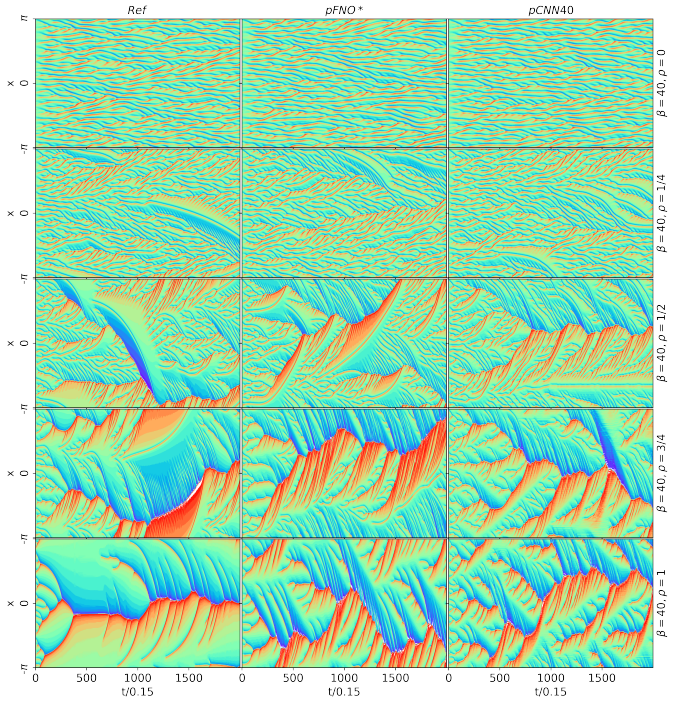

Figure 6. Comparison of front slopes predicted by pFNO* and pCNN40 at $\beta = 40$. Other details are the same as in Figure 5.

All relevant model predictions at all fifteen parametric configurations $(\beta, \rho) \in [10, 25, 40] \times [0, 1/4, 1/2, 3/4, 1]$ are compared in Figure 7 for the normalized total front length ($\int (\phi_x^2 + 1)^{1/2} dx/(2\pi)$), in Figure 8 for the model errors accumulated through recurrent predictions, and in Figure 9 for the long-term auto-correlation function. This auto-correlation function characterizes the long-term recurrently predicted solutions:

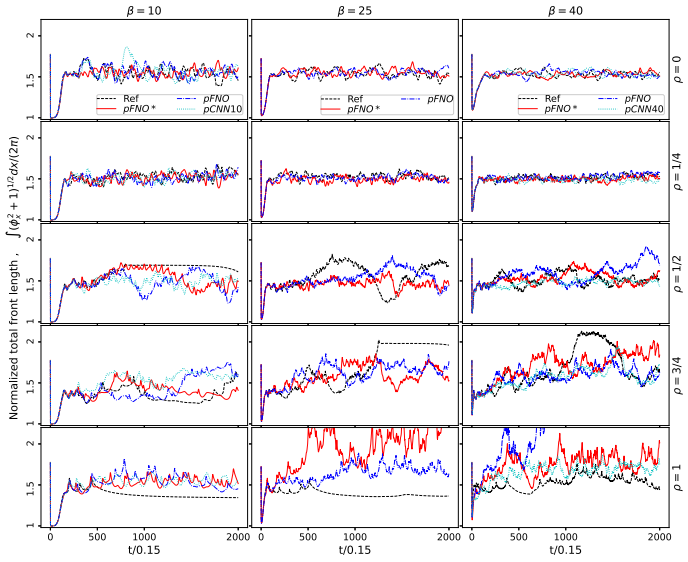

Figure 7. Normalized total front length comparison over 15 parametric configurations of $(\beta, \rho) = [10, 25, 40] \times [0, 1/4, 1/2, 3/4, 1]$. The black curve represents a single instance of reference solutions obtained from Equation (8). Predictions by pFNO* are shown in red, pFNO in blue, and pCNN10/pCNN40 in cyan.

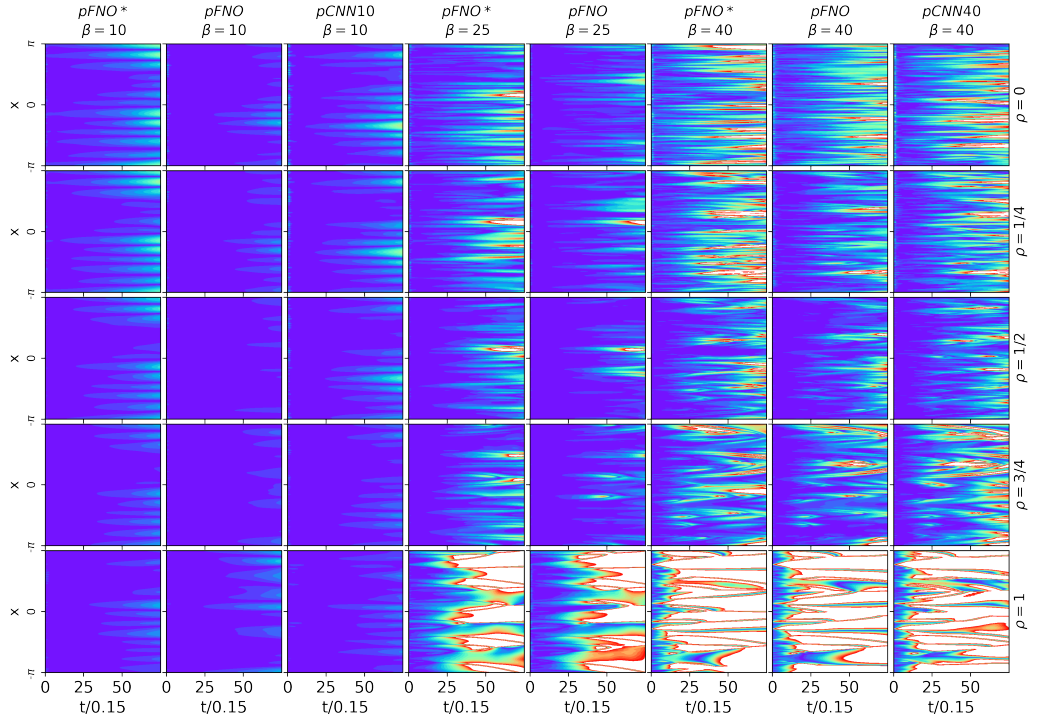

Figure 8. Time evolution of the relative $L2$ error between the reference solution of Equation (8) and the predicted solutions by pFNO*, pFNO, and pCNN10/pCNN40. The reference solutions are initialized with random conditions at 15 parametric configurations of $(\beta, \rho) = [10, 25, 40] \times [0, 1/4, 1/2, 3/4, 1]$. Rainbow colors from blue to red represent values ranging from 0 to 0.1; values above 0.1 are truncated and displayed as white.

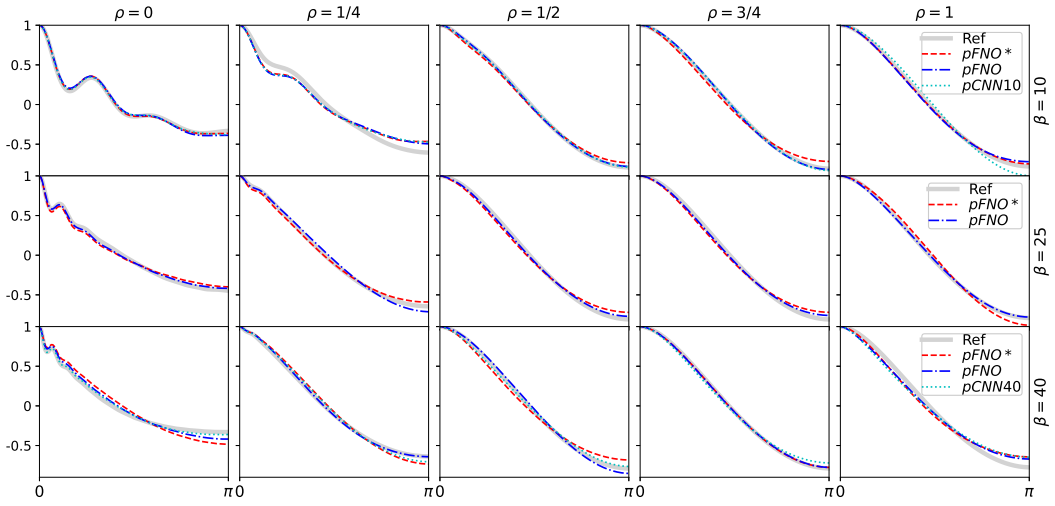

Figure 9. The auto-correlation functions (Equation (12)) calculated from the long-term reference solutions at 15 parametric configurations of $(\beta, \rho) = [10, 25, 40] \times [0, 1/4, 1/2, 3/4, 1]$ are compared against those computed from the long-term predictions learned using pFNO*, pFNO, and pCNN10/pCNN40.

$$\mathcal{R}(r) = \mathbb{E}\left(\int_{\mathcal{D}} \phi^*(x)\phi^*(x-r)dx \Big/ \int_{\mathcal{D}} \phi^*(x)\phi^*(x)dx \right). \tag{12}$$

where $\phi^*(x)$ denotes the predicted solutions obtained after a sufficiently long time. Numerical calculation for the expectation $\mathbb{E}$ in Equation (12) is implemented by averaging over seven randomly initialized sequences of model predictions for a time duration

$1000 < t/\Delta_t < 4000$. Moreover, for each of the learned models $\mathcal{G}_{\theta,(\beta,\rho)} \approx \hat{\mathcal{G}}_{(\beta,\rho)}$, we compute an approximated dispersion relation:

$$\omega'(\kappa) = \log(J(\kappa,\kappa))/\Delta_t \tag{13}$$

where J is the operator Jacobian

$$J(\kappa,\overline{\kappa}) = \frac{\partial}{\partial \epsilon_\kappa}\left|\mathcal{F}_{\overline{\kappa}}\{\mathcal{G}_{\theta,(\beta,\rho)}(2\epsilon_\kappa \cos(\kappa x))\}\right|, \quad \text{with } \kappa,\overline{\kappa} = 0,1,2,... \tag{14}$$

This Jacobian is computed using the automatic differential tool (e.g., torch.autograd. functional.jacobian in PyTorch version 3.10.4). Figure 10 compares the dispersion relations by all models with the reference ones. Additionally, Figure 10 shows one example of a learned operator Jacobian, which is clearly diagonal dominant.

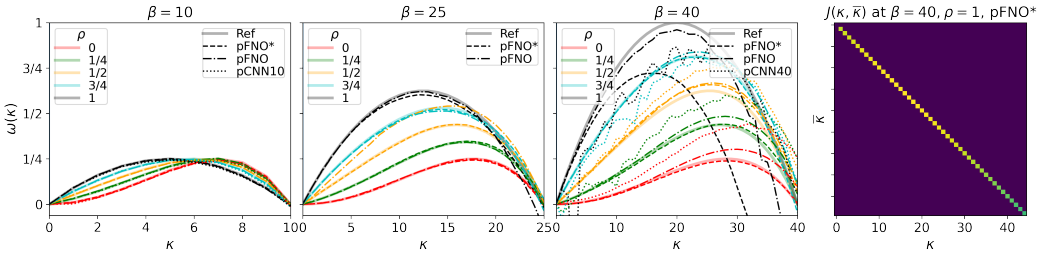

Figure 10. Left three columns: Comparison of reference dispersion relations (Equation (9), solid lines) with those computed for the learned operators of all models (Equation (13), non-solid lines), where line colors indicate different ρ. Right column: Illustration of a learned operator Jacobian (Equation (14)), with dark colors indicating small values.

4.4. Findings

Overall learning: Our study underscores the robust learning capabilities of pFNO and pCNN methodologies in capturing the nuanced dynamics of flame front evolution, modulated by varying DL and DT instabilities blends. Both pFNO and pFNO* demonstrate good performance in learning the full two-parameter front evolution operator $\hat{\mathcal{G}}_{\rho,\beta}$ modulated by a ρ-varying blends of DL/DT instabilities as well as by a β-varying size of the largest unstable wavelength. While pCNN encounters difficulty in learning the full operator, the method still performs well in learning different instabilities when being restricted for the single-parameter operators $\hat{\mathcal{G}}_{\rho,\beta=10}$ and $\hat{\mathcal{G}}_{\rho,\beta=40}$.

Short-term learning: Across the board, all learned models (pFNO, pFNO* and pCNN10/40) demonstrate good accuracy in short-term predictions, with training/validation errors below 2 percent (Table 1) and small accumulated errors (Figure 8). This precision extends to various metrics, including front displacement, front slope, and normalized front length (Figures 3–7 for $t \leq 50\Delta_t$), affirming the models' fidelity in capturing short-term dynamics. Moreover, pFNO demonstrates the smallest error and is the most accurate model for learning short-term solutions.

Long-term learning: Detailed analysis of reference solutions unveils distinct characteristics of isolated instabilities. At $\rho = 1$, DL fronts evolve toward a single giant cusp structure, either remaining stationary at small $\beta = 10$ (Figure 3) or exhibiting noise-induced wrinkles at larger $\beta = 40$ (Figure 4). Conversely, at $\rho = 0$, DT fronts adopt an overall flat shape interspersed with oscillatory wavy structures, with decreasing wavelength and amplitude as β increases from 10 to 40 (Figures 3 and 4). The slope plots for DT front evolution result in the typical zebra stripe pattern (Figures 5 and 6). Intermediate values of ρ showcase a gradual transition between these features, with the front structure blending wavy oscillations together with the cusp shape.

Long-term predictions by all learned models (pFNO*, pFNO, pCNN10/pCNN40) accurately replicate these characteristic behaviors across diverse parametric configurations,

encompassing pure DL and DT instabilities as well as blended scenarios. Quantitative comparisons through auto-correlation functions (Figure 9) and total front length (Figure 7) confirm the models' proficiency in capturing long-term solutions.

Learning challenges: However, a common challenge across both pFNO and pCNN models lies in over-predicting the impact of noise-induced wrinkles, particularly noticeable at small $\beta = 10$ (Figures 3 and 5). This tendency leads to an overestimation of the total front area, especially pronounced at lower β values of 10 and 25 (Figure 7 at $\rho = 1$). When learning for the hybrid DL and DT instabilities, excessive noisy wrinkles also show up in all the model predictions (at $\rho = 3/4$ in Figures 5 and 6), however, the issue becomes less discernible toward smaller values of ρ when DT instability plays a larger role, as also evident by the front length in Figure 7.

Extra finding: It is particularly interesting to point out that the two models, pFNO and pFNO*, learn well on the parametric-dependent linear dispersion relations, as seen in Figure 10. Except for a moderate level of mismatch at a few parameter conditions toward large ρ at 25 and 40, pFNO and pFNO* reproduce the relations quite accurately.

Such learning performance is impressive considering the fact that the data effective for learning these linear relations (i.e., the initial near-zero solutions) is just a tiny portion of the total dataset. For pCNN-based models, Figure 10 shows pCNN10 learns the dispersion quite accurately while pCNN40 learned relations show a more significant deviation than ones by pFNO.

5. Summary and Conclusions

This paper delves into the potential of machine learning (ML) for understanding and predicting the behavior of flames experiencing hybrid instabilities. These instabilities arise from the interplay of two key mechanisms: the Darrieus–Landau (DL) instability, driven by density gradients across a flame, and the Diffusive–Thermal (DT) instability, caused by heat and mass diffusion disparities.

The nonlinear development of unstable flames can be modeled by a well-known partial differential equation (PDE), specifically the Sivashinsky equation. By re-expressing the Sivashinsky equation, we introduce two parameters: ρ and β. These parameters control the blending of DT and DL instabilities, as well as the cutoff wavelength for unstable flame behavior.

Our learning problem focuses on understanding the PDE solution time advancement operator under different parameter combinations. This operator, when repeatedly applied with its input solution as the output from the previous iteration, yields a time sequence of solutions of arbitrary length. We employ two recently developed operator learning models: parameterized Fourier Neural Operators (pFNO) and Convolutional Neural Networks (pCNNs). Our findings demonstrate that both pFNO and pCNN models effectively capture the intricate flame dynamics under varying DT/DL instabilities (due to ρ variations). Specifically:

Short-Term Predictions: All learned models accurately predict short-term solutions and dispersion relations.

Long-Term Behavior: The models also reproduce correct statistics, quantified by autocorrelation functions and total front length.

pFNO Superiority: Notably, pFNO outperforms pCNN by allowing the learning of the full two-parameter operator, enabling variation in both ρ and β.

Challenges: However, both pCNN and pFNO tend to overestimate noise-induced wrinkles associated with DL instability, leading to inaccurate predictions of the total flame area, especially at lower instability levels.

In conclusion, this work showcases the potential of operator learning methodologies for analyzing complex flame dynamics arising from hybrid instabilities. While challenges persist, particularly related to noise overestimation, these methods offer assisting tools for understanding and predicting real-world flame behavior in combustion systems [36–43]. Realistic flame development can be influenced by various factors beyond the two intrinsic

flame instability mechanisms considered in this study. These additional factors include mechanisms such as thermoacoustic instabilities, Rayleigh–Taylor instabilities, and disturbances due to turbulent background flow. However, if the evolution of a realistic flame can be described by certain PDEs, it can still be viewed as a parametrized solution advancement operator. It is crucial to emphasize the importance of obtaining a high-quality training dataset on real flame evolution. Such datasets can be derived either from high-fidelity numerical simulations or sophisticated laser-diagnostic experiments. With this data, the flame evolution could potentially be learned by the parametric operator learning methods demonstrated in our work. Future research directions may involve incorporating additional physical mechanisms or exploring alternative learning architectures to further enhance the accuracy and robustness of these models.

Author Contributions: Conceptualization: R.Y., E.H., and K.-J.N.; methodology: R.Y., E.H., and K.-J.N.; software: R.Y.; validation: R.Y.; formal analysis: R.Y.; investigation: R.Y.; resources: R.Y., E.H., and K.-J.N.; data curation: R.Y., E.H., and K.-J.N.; writing—original draft preparation: R.Y.; writing—review and editing: R.Y., E.H., and K.-J.N.; visualization: R.Y.; supervision: R.Y.; project administration: R.Y., E.H., and K.-J.N.; funding acquisition: R.Y., E.H., and K.-J.N. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Swedish Research Council with grant number VR-2019-05648.

Data Availability Statement: The code and data that support the findings of this study are openly available at www.github.com/RixinYu/ML_paraFlame accessed on 11 May 2024.

Acknowledgments: The authors gratefully acknowledge the financial support from the Swedish Research Council (VR-2019-05648). The computations were enabled by resources provided by the National Academic Infrastructure for Supercomputing in Sweden (NAISS) at ALVIS and Tetralith, partially funded by the Swedish Research Council through grant agreement no. 2022-06725.

Conflicts of Interest: Author Karl-Johan Nogenmyr was employed by the Siemens Energy AB. The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Appendix A. Model Hyper-Parameters and Training Details

All models undergo training for 1000 epochs with a batch size of 1000, employing the Adam optimizer with a learning rate set at 0.0025 and a weight decay of 0.0001. A scheduler step size of 100 and a gamma value of 0.5 are applied for learning rate adjustment. To stabilize the training process, the maximum norm of gradients is clipped above 50.

Training for pFNO and pFNO* is conducted over the entire dataset using a single GPU (NVIDIA Tesla A40), taking approximately 28 and 47 h, respectively. In contrast, training pCNN10 and pCNN40 is performed on one-third of the training dataset, with both models requiring around 38 h using a single GPU. Conversely, training the pCNN model (to learn the full two-parameters operator) using the entire dataset takes 26 h utilizing four GPUs.

The pFNO and pFNO* networks are configured with $L = 4$ levels and $d_\varepsilon = 30$ channels, with two hyperparameters set: $\kappa_{\max} = 128$ and $N_\gamma = 5$. To reduce model size, all pFNO methods share most of the trainable parameters within a single parametric Fourier layer (Equation (5)) across all layers $l = 0, ..., L-1$, except for those used to parameterize the function $D_l(\gamma)$.

References

1. Guo, X.; Li, W.; Iorio, F. Convolutional neural networks for steady flow approximation. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016.
2. Zhu, Y.; Zabaras, N. Bayesian deep convolutional encoder–decoder networks for surrogate modeling and uncertainty quantification. *J. Comput. Phys.* **2018**, *366*, 415–447. [CrossRef]
3. Adler, J.; Öktem, O. Solving ill-posed inverse problems using iterative deep neural networks. *Inverse Probl.* **2017**, *33*, 124007. [CrossRef]
4. Bhatnagar, S.; Afshar, Y.; Pan, S.; Duraisamy, K.; Kaushik, S. Prediction of aerodynamic flow fields using convolutional neural networks. *Comput. Mech.* **2019**, *64*, 525–545. [CrossRef]

5. Khoo, Y.; Lu, J.; Ying, L. Solving parametric PDE problems with artificial neural networks. *Eur. J. Appl. Math.* **2021**, *32*, 421–435. [CrossRef]
6. Ronneberger, O.; Fischer, P.; Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Proceedings of the Medical Image Computing and Computer-Assisted Intervention—MICCAI 2015: 18th International Conference, Munich, Germany, 5–9 October 2015*; Proceedings, Part III 18; Springer: Cham, Switzerland, 2015.
7. Winovich, N.; Ramani, K.; Lin, G. ConvPDE-UQ: Convolutional neural networks with quantified uncertainty for heterogeneous elliptic partial differential equations on varied domains. *J. Comput. Phys.* **2019**, *394*, 263–279. [CrossRef]
8. Li, Z.; Kovachki, N.; Azizzadenesheli, K.; Liu, B.; Bhattacharya, K.; Stuart, A.; Anandkumar, A. Neural Operator: Graph Kernel Network for Partial Differential Equations. *arXiv* **2020**, arXiv:2003.03485.
9. Kovachki, N.; Li, Z.; Liu, B.; Azizzadenesheli, K.; Bhattacharya, K.; Stuart, A.; Anandkumar, A. Neural operator: Learning maps between function spaces. *arXiv* **2021**, arXiv:2108.08481.
10. Lu, L.; Jin, P.; Karniadakis, G. DeepONet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators. *arXiv* **2019**, arXiv:1910.03193.
11. Li, Z.; Kovachki, N.; Azizzadenesheli, K.; Liu, B.; Bhattacharya, K.; Stuart, A.; Anandkumar, A. Fourier neural operator for parametric partial differential equations. *arXiv* **2020**, arXiv:2010.08895.
12. Lu, L.; Jin, P.; Pang, G.; Zhang, Z.; Karniadakis, G. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nat. Mach. Intell.* **2021**, *3*, 218–229. [CrossRef]
13. Lu, L.; Meng, X.; Cai, S.; Mao, Z.; Goswami, S.; Zhang, Z.; Karniadakis, G. A comprehensive and fair comparison of two neural operators (with practical extensions) based on FAIR data. *Comput. Methods Appl. Mech. Eng.* **2022**, *393*, 114778. [CrossRef]
14. Gupta, G.; Xiao, X.; Bogdan, P. Multiwavelet-based operator learning for differential equations. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 24048–24062.
15. Tripura, T.; Chakraborty, S. Wavelet neural operator for solving parametric partial differential equations in computational mechanics problems. *Comput. Methods Appl. Mech. Eng.* **2023**, *404*, 115783. [CrossRef]
16. Chen, G.; Liu, X.; Li, Y.; Meng, Q.; Chen, L. Laplace neural operator for complex geometries. *arXiv* **2023**, arXiv:2302.08166.
17. Yu, R. Deep learning of nonlinear flame fronts development due to Darrieus–Landau instability. *APL Mach. Learn.* **2023**, *1*, 026106. [CrossRef]
18. Yu, R.; Hodzic, E. Parametric learning of time-advancement operators for unstable flame evolution. *Phys. Fluids* **2024**, *36*, 044109. [CrossRef]
19. Darrieus, G. Propagation d'un front de flamme. Unpublished work presented at La Technique Moderne. 1938.
20. Landau, L. On the theory of slow combustion. In *Dynamics of Curved Fronts*; Elsevier: Amsterdam, The Netherlands, 1988; pp. 403–411.
21. Zeldovich, Y. Theory of Combustion and Detonation of Gases. In *Selected Works of Yakov Borisovich Zeldovich, Volume I: Chemical Physics and Hydrodynamics*; Princeton University Press: Princeton, NJ, USA, 1944.
22. Sivashinsky, G. Diffusional-thermal theory of cellular flames. *Combust. Sci. Technol.* **1977**, *15*, 137–145. [CrossRef]
23. Yu, R.; Bai, X.S.; Bychkov, V. Fractal flame structure due to the hydrodynamic Darrieus-Landau instability. *Phys. Rev. E* **2015**, *92*, 063028. [CrossRef] [PubMed]
24. Michelson, D.M.; Sivashinsky, G.I. Nonlinear analysis of hydrodynamic instability in laminar flames—II. Numerical experiments. *Acta Astronaut.* **1977**, *4*, 1207–1221. [CrossRef]
25. Kuramoto, Y. Diffusion-induced chaos in reaction systems. *Prog. Theor. Phys. Suppl.* **1978**, *64*, 346–367. [CrossRef]
26. Sivashinsky, G.I. Nonlinear analysis of hydrodynamic instability in laminar flames—I. Derivation of basic equations. *Acta Astronaut.* **1977**, *4*, 1177–1206. [CrossRef]
27. Thual, O.; Frisch, U.; Hénon, M. Application of pole decomposition to an equation governing the dynamics of wrinkled flame fronts. *J. Phys.* **1985**, *46*, 1485–1494. [CrossRef]
28. Vaynblat, D.; Matalon, M. Stability of Pole Solutions for Planar Propagating Flames: I. Exact Eigenvalues and Eigenfunctions. *SIAM J. Appl. Math.* **2000**, *60*, 679–702. [CrossRef]
29. Vaynblat, D.; Matalon, M. Stability of Pole Solutions for Planar Propagating Flames: II. Properties of Eigenvalues/Eigenfunctions and Implications to Stability. *SIAM J. Appl. Math.* **2000**, *60*, 703–728. [CrossRef]
30. Olami, Z.; Galanti, B.; Kupervasser, O.; Procaccia, I. Random noise and pole dynamics in unstable front propagation. *Phys. Rev. E* **1997**, *55*, 2649. [CrossRef]
31. Denet, B. Stationary solutions and Neumann boundary conditions in the Sivashinsky equation. *Phys. Rev. E* **2006**, *74*, 036303. [CrossRef] [PubMed]
32. Kupervasser, O. *Pole Solutions for Flame Front Propagation*; Springer International Publishing: Cham, Switzerland, 2015.
33. Karlin, V. Cellular flames may exhibit a non-modal transient instability. *Proc. Combust. Inst.* **2002**, *29*, 1537–1542. [CrossRef]
34. Creta, F.; Lapenna, P.E.; Lamioni, R.; Fogla, N.; Matalon, M. Propagation of premixed flames in the presence of Darrieus–Landau and thermal diffusive instabilities. *Combust. Flame* **2020**, *216*, 256–270. [CrossRef]
35. Creta, F.; Fogla, N.; Matalon, M. Turbulent propagation of premixed flames in the presence of Darrieus–Landau instability. *Combust. Theory Model.* **2011**, *15*, 267–298. [CrossRef]
36. Hodzic, E.; Alenius, E.; Duwig, C.; Szasz, R.; Fuchs, L. A Large Eddy Simulation Study of Bluff Body Flame Dynamics Approaching Blow-Off. *Combust. Sci. Technol.* **2017**, *189*, 1107–1137. [CrossRef]

37. Hodzic, E.; Jangi, M.; Szasz, R.Z.; Bai, X.S. Large eddy simulation of bluff body flames close to blow-off using an Eulerian stochastic field method. *Combust. Flame* **2017**, *181*, 1–15. [CrossRef]
38. Hodzic, E.; Jangi, M.; Szasz, R.Z.; Duwig, C.; Geron, M.; Early, J.; Fuchs, L.; Bai, X.S. Large Eddy Simulation of Bluff-Body Flame Approaching Blow-Off: A Sensitivity Study. *Combust. Sci. Technol.* **2018**, *191*, 1815–1842. [CrossRef]
39. Yu, R.; Bai, X.S.; Lipatnikov, A.N. A direct numerical simulation study of interface propagation in homogeneous turbulence. *J. Fluid Mech.* **2015**, *772*, 127–164. [CrossRef]
40. Yu, J.; Yu, R.; Bai, X.; Sun, M.; Tan, J.G. Nonlinear evolution of 2D cellular lean hydrogen/air premixed flames with varying initial perturbations in the elevated pressure environment. *Int. J. Hydrogen Energy* **2017**, *42*, 3790–3803. [CrossRef]
41. Yu, R.; Nillson, T.; Bai, X.; Lipatnikov, A.N. Evolution of averaged local premixed flame thickness in a turbulent flow. *Combust. Flame* **2019**, *207*, 232–249. [CrossRef]
42. Yu, R.; Lipatnikov, A.N. Surface-averaged quantities in turbulent reacting flows and relevant evolution equations. *Phys. Rev. E* **2019**, *100*, 013107. [CrossRef]
43. Yu, R.; Nilsson, T.; Fureby, C.; Lipatnikov, A. Evolution equations for the decomposed components of displacement speed in a reactive scalar field. *J. Fluid Mech.* **2021**, *911*, A38. [CrossRef]