# SPEAR: A SPECTRAL-DISENTANGLED MOE NEURAL OPERATOR WITH KNOWLEDGE-GUIDED EXPERT AGGREGATION FOR LARGE-SCALE PDE PRETRAINING

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Large-scale pre-training has improved the generalization of neural operators across diverse PDEs. However, existing PDE foundation models still face heterogeneous dynamics, where shared representations may cause knowledge interference and MoE architectures may introduce expert redundancy. We propose SPEAR, a spectral-disentangled MoE neural operator with knowledge-guided expert aggregation for large-scale PDE pre-training. SPEAR separates latent features into low- and high-frequency components, enabling shared modeling of transferable dynamics and specialized learning of PDE-specific patterns. To reduce redundancy, we further introduce a knowledge-guided aggregation strategy that measures expert similarity using dataset-specific knowledge and routing preferences, and consolidates similar experts. Experiments on twelve PDE datasets and multiple downstream benchmarks demonstrate strong performance in pre-training, fine-tuning, and transfer learning. Our aggregation strategy further reduces the number of experts by 50% while maintaining or improving prediction accuracy, balancing efficiency and generalization.

## 1 INTRODUCTION

Neural operators have emerged as a compelling alternative for solving PDEs by learning mappings between infinite-dimensional function spaces, achieving orders-of-magnitude speedups at inference time. However, existing neural operators are typically trained under single-task settings—specific PDE families with fixed parameter ranges—resulting in limited generalization to unseen equation types, parameter regimes, or boundary conditions. Inspired by large language models Vaswani et al. (2017) and vision foundation models Radford et al. (2021), recent efforts have introduced large-scale pre-training into the PDE domain to learn transferable physical representations. While pre-training enhances generalization, different PDEs exhibit substantially different underlying dynamics, and modeling multiple physical processes within a single shared parameter space may lead to knowledge conflicts and negative transfer.

To scale up modeling capacity while maintaining inference efficiency, recent work has introduced Mixture-of-Experts (MoE) architectures Shazeer et al. (2017) into neural operator design Wang et al. (2026); Sun et al. (2026). MoEs employ dynamic routing mechanisms that activate only a subset of expert subnetworks for each input, enabling massive parameter scaling without proportionally increasing per-sample computation. This paradigm shows promise in handling PDE heterogeneity: different experts can specialize in different equation types, physical regimes, or numerical patterns. However, MoE-based neural operators still face the following two critical challenges. As shown in Figure 1, we summarize the motivation and overview of SPEAR.

**Challenge 1: Spectral Coupling of PDEs**. Different PDE families differ in their physical behaviors. For example, elliptic equations describe steady-state equilibrium, with high-frequency features manifesting as smooth local extrema; hyperbolic equations describe wave propagation, with high-frequency features manifesting as sharp discontinuities; parabolic equations describe dissipative processes, with high-frequency features manifesting as decaying spikes. These differences are most pronounced in high-frequency local patterns, while low-frequency dynamics exhibit similarity across equation types. However, Fourier-based neural operators such as AFNO Guibas et al.



Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

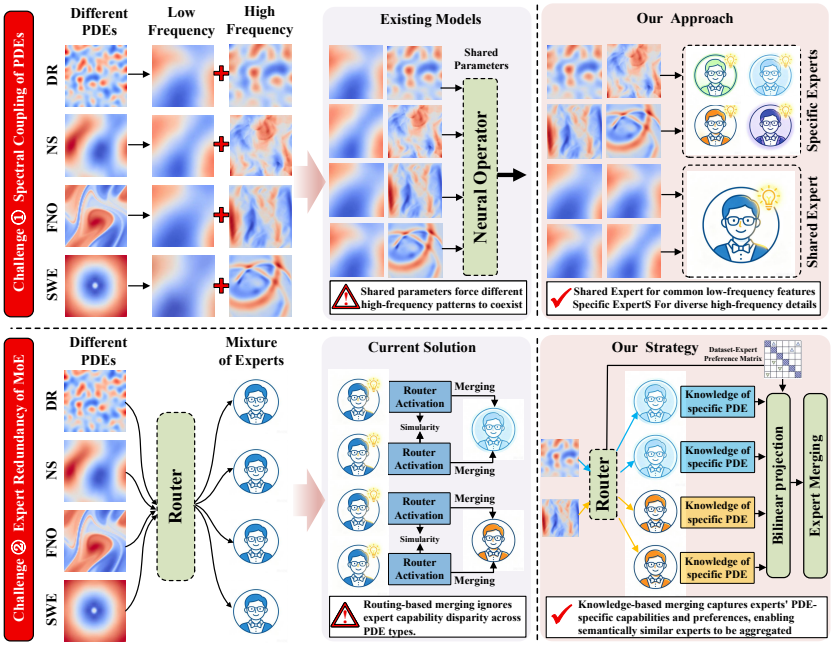



Figure 1: Motivation of SPEAR. The upper shows the advantage of frequency decoupling for separating shared and PDE-specific dynamics. The lower highlights knowledge-guided expert merging over routing-based merging.

(2021) couple all frequency components within shared parameters, forcing morphologically distinct high-frequency patterns to coexist and causing gradient interference. Existing remedies, including multi-scale operators Liu et al. (2024) and frequency enhancement modules Chen et al. (2025), have yet to break this spectral coupling constraint. Therefore, a mechanism that can separate shared dynamics from specific patterns according to frequency characteristics is urgently needed.

**Challenge 2: Expert Redundancy of MoE**. While MoE architectures enhance model capacity through specialization, this comes at a cost: as model scale grows, parameter redundancy proliferates, with many experts learning functionally similar representations while inference overhead increases. Existing expert merging methods Zhang et al. (2026); Li et al. (2026; 2024); Chen et al. (2024a) primarily consolidate similar experts based on routing statistics or expert outputs, rather than the task-specific knowledge semantics that experts acquire. This ”behavior-based” rather than ”knowledge-based” merging strategy lacks physical grounding. In PDE learning, this is particularly problematic: two experts that handle similar physical patterns may never be activated together due to different data distributions, and would thus be judged as ”dissimilar” and left unmerged.

It is worth noting that these two challenges are not independent. A key fact connects them: low-frequency dynamics are broadly transferable across PDE types and can be effectively captured by a shared backbone, while high-frequency patterns are PDE-specific and demand expert specialization. This observation naturally suggests a two-stage strategy: first, decouple frequencies to separate sharable low-frequency knowledge from PDE-specific high-frequency patterns, assigning the latter to a dedicated MoE module for expert-level specialization; then, consolidate the high-frequency experts to eliminate redundancy, ensuring that each preserved expert captures truly distinct physical knowledge. In other words: decouple to specialize, then aggregate to refine.

Building on this insight, in this paper, we present a **SP**ectral-disentangled MoE neural operator with knowledge-guided **E**xpert **A**gg**R**egation (**SPEAR**). Our framework introduces two tightly coupled innovations. **For challenge 1**, we propose a spectral-disentangled architecture that explicitly partitions the frequency spectrum. The low-frequency branch employs AFNO to capture global evolution patterns, while the high-frequency branch extracts fine-grained local structures via high-pass filtering and models them through an MoE module composed of shared convolutional backbones and LoRA adapters. Dynamic routing based on high-frequency statistics activates a part of experts to efficiently learn local dynamics across PDEs. **For challenge 2**, we design a knowledge-guided expert aggregation strategy that compresses and reorganizes functionally similar experts by incorporat-

ing physical correlations among PDE tasks and expert preference information. Rather than routing statistics, expert similarity is measured by the learned parameters for each PDE dataset. Then, a dataset-level knowledge matrix is constructed and fused via bilinear projection, yielding semantic similarities for expert merging. Our main contributions are summarized as follows:

1) We design a spectral-disentangled architecture that explicitly partitions the frequency spectrum: a shared backbone captures low-frequency dynamics across all PDE types, while a dedicated MoE module handles PDE-specific high-frequency patterns.

2) We propose a knowledge-guided expert aggregation strategy that computes similarities among experts based on their acquired physical knowledge during training and routing preference information, and merges them via bilinear projection and hierarchical clustering.

3) Extensive experiments on large-scale PDE pretraining datasets show that our method learns transferable representations, achieving strong performance across cross-equation pretraining, fine-tuning, and generalization to unseen PDEs with superior parameter efficiency.

## 2 Related Work

### 2.1 Neural Operators for PDE Solving

Neural operators learn mappings between infinite-dimensional function spaces Lu et al. (2021); Li et al. (2020). Various neural operators have been developed for PDE modeling. For example, DeepONet Lu et al. (2021) adopts a branch–trunk architecture, while the Fourier Neural Operator (FNO) Li et al. (2020) models global dependencies in the Fourier domain. More recently, methods such as MPP McCabe et al. (2023), DPOT Hao et al. (2024), Poseidon Herde et al. (2024), OmniArch Chen et al. (2024b), and UniSolver Zhou et al. (2025) have explored pre-training on diverse PDE datasets, advancing PDE foundation models. However, most existing methods adopt a unified architecture. As PDE datasets become larger and more diverse, differences in physical laws and dynamics may cause knowledge interference across heterogeneous tasks. Therefore, modeling diverse physical knowledge remains an important challenge in large-scale PDE pre-training.

### 2.2 Mixture-of-Experts Models

Mixture-of-Experts (MoE) expands model capacity through sparse expert activation and has been widely adopted in language and vision models. Representative works include Sparsely-Gated MoE Shazeer et al. (2017), Switch Transformer Fedus et al. (2022), and Mixtral Jiang et al. (2024). By activating only a small subset of experts for each input, MoE increases model capacity with limited additional computation. Recently, MoE has also been introduced into neural operator models. Methods such as MoE-POT Wang et al. (2026) and NESTOR Sun et al. (2026) use specialized experts to capture diverse physical knowledge and alleviate interference across heterogeneous PDE tasks. However, some experts may learn similar physical representations, leading to knowledge redundancy and inefficient parameter utilization. Therefore, reducing redundant experts while preserving model performance remains an important challenge in MoE-based PDE modeling.

### 2.3 Expert Merging

Expert merging is an emerging compression technique that identifies experts with similar functions or representations and integrates them through parameter fusion. Representative methods include SMoE Li et al. (2024) and HC-SMoE Chen et al. (2024a), which construct expert relationships from routing patterns or output features and then merge redundant experts. However, most existing methods lack explicit modeling of task semantics and domain knowledge, making it difficult to accurately characterize the knowledge learned by different experts.

In PDE modeling, MoE encourages functional specialization, while knowledge overlap may still exist among some experts. Incorporating physical knowledge into expert relationship modeling can therefore improve the identification of truly redundant experts and enable effective compression while preserving expert diversity, which is important for building efficient and scalable PDE foundation models.

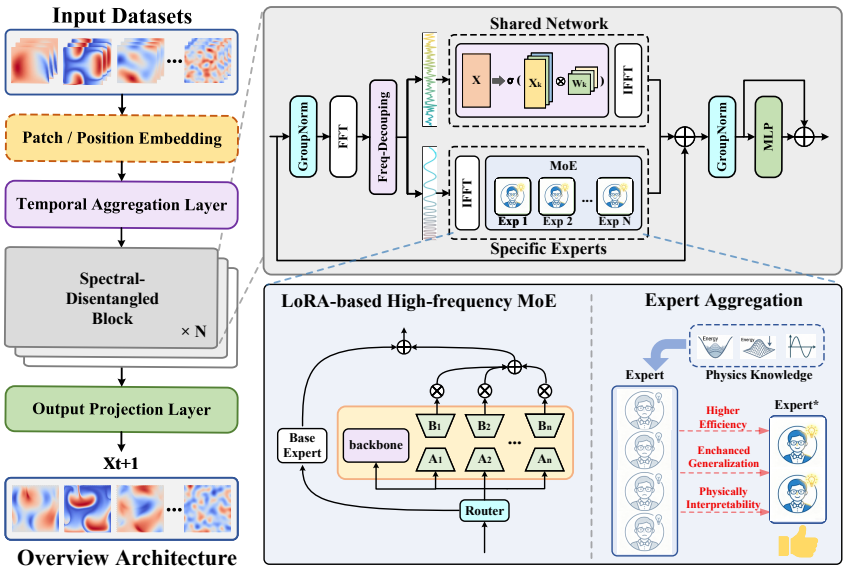



Figure 2: Overview architecture. The left panel shows the overall pipeline, the upper-right panel presents the Spectral-Disentangled block, and the lower-right panel illustrates the high-frequency MoE module with low-rank experts.

## 3 The Proposed Approach

We propose SPEAR, a spectral-disentangled MoE neural operator with knowledge-guided expert aggregation, as shown in Fig. 2. SPEAR separates shared low-frequency dynamics from PDE-specific high-frequency patterns, while the MoE module captures specialized local dynamics. Parameter-efficient design and expert aggregation further reduce redundancy.

### 3.1 Problem Definition

We consider parameterized PDEs over a spatial domain $\Omega \subset \mathbb{R}^n$ and a time interval $[0, T]$:

$$\frac{\partial \mathbf{u}}{\partial t} - \mathcal{H}\left(\mathbf{u}, \nabla \mathbf{u}, \nabla^2 \mathbf{u}, \ldots; \theta\right) = 0, s.t. \begin{cases} \mathbf{u}(\mathbf{x}, 0) = \mathbf{u}_0(\mathbf{x}), & \mathbf{x} \in \Omega, \\ \mathcal{B}[\mathbf{u}](\mathbf{x}, t) = g(\mathbf{x}, t), & (\mathbf{x}, t) \in \partial\Omega \times [0, T], \end{cases} \tag{1}$$

where $\mathbf{u}$ denotes the system state, $\mathcal{H}$ the governing operator, $\theta$ the physical parameters, and $\mathcal{B}$ the boundary operator. After time discretization, a neural operator $\mathcal{T}_w$ learns the temporal evolution as

$$\mathbf{u}_{t+1} = \mathcal{T}_w\left(\mathbf{u}_{t-K+1}, \ldots, \mathbf{u}_t\right), \tag{2}$$

where $w$ denotes the learnable parameters and $K$ is the number of input time steps. Following DPOT Hao et al. (2024), small random perturbations are added to the input sequence during training to reduce error accumulation and improve long-term prediction robustness.

### 3.2 Spectral-Disentangled Block

Given an input PDE sequence, we first perform patch-based input encoding and temporal aggregation Dosovitskiy et al. (2020); Hao et al. (2024) to obtain the latent representation $\mathbf{Y}$. Spatiotemporal physical fields contain smooth global structures and sharp local variations. Therefore, we introduce a spectral-disentangled dual-branch architecture that separates low- and high-frequency components. The low-frequency branch captures shared global dynamics, while the high-frequency branch models PDE-specific local patterns with MoE.

#### 3.2.1 Low-Frequency Global Branch

We employ AFNO Guibas et al. (2021) as the Shared Expert $\phi_{\text{shared}}$ for low-frequency global modeling. Given $\mathbf{Y}$, we first transform it into the frequency domain as $\hat{\mathbf{Y}} = \mathcal{F}(\mathbf{Y})$, where $\mathcal{F}$ and $\mathcal{F}^{-1}$ denote the 2D Fourier transform and its inverse. The first $K$ low-frequency modes are processed by

a two-layer block-wise complex-valued MLP: $\mathbf{Z}_{:K,:K} = \mathbf{W}_2^f \, \sigma\Big(\mathbf{W}_1^f \, \hat{\mathbf{Y}}_{:K,:K} + \mathbf{b}_1^f\Big) + \mathbf{b}_2^f$, where $\mathbf{W}_1^f, \mathbf{W}_2^f$ are block-wise complex-valued weights. The remaining modes are set to zero, and the result is transformed back to the spatial domain with a residual connection:

$$\mathbf{Y}_{\mathrm{lf}} = \phi_{\mathrm{shared}}(\mathbf{Y}) = \mathcal{F}^{-1}(\mathbf{Z}) + \mathbf{Y}. \tag{3}$$

#### 3.2.2 High-Frequency Local Branch

To capture sharp local variations, we apply a hard high-pass filter to $\mathbf{Y}$ in the frequency domain. Given $\hat{\mathbf{Y}} = \mathcal{F}(\mathbf{Y})$, a binary mask $\mathbf{M}$ removes the first $K$ low-frequency modes:

$$\mathbf{M}(i,j) = \begin{cases} 0, & i < K \text{ and } j < K, \\ 1, & \text{otherwise.} \end{cases} \tag{4}$$

The high-frequency feature is then obtained as $\mathbf{Y}_{\mathrm{hf}} = \mathcal{F}^{-1}(\hat{\mathbf{Y}} \odot \mathbf{M})$ and fed into the MoE module for modeling fine-grained local dynamics.

#### 3.2.3 Low-Rank Adapter-based High-frequency Mixture of Experts

High-frequency dynamics, such as small-scale vortices and shock boundaries, are challenging to model with a single architecture. We therefore introduce a low-rank adapter-based MoE with dynamic routing, consisting of $N$ non-shared experts and one shared expert, while the gate activates the Top-2 non-shared experts for each input. For brevity, let $\mathbf{H} \triangleq \mathbf{Y}_{\mathrm{hf}} \in \mathbb{R}^{B\times C\times H\times W}$ denote the high-frequency feature fed into this module.

**(1) Routing Strategy.** The router takes $\mathbf{H}$ as input. To emphasize local variations, we remove the spatial mean to obtain $\tilde{\mathbf{H}} = \mathbf{H} - \mathrm{Mean}_{H,W}(\mathbf{H})$. For the $b$-th sample, the routing feature combines the channel-wise mean and standard deviation:

$$\mathbf{g}_b = \Big[\mathrm{Mean}_{H,W}(\tilde{\mathbf{H}}_b);\ \mathrm{Std}_{H,W}(\tilde{\mathbf{H}}_b)\Big] \in \mathbb{R}^{2C}, \tag{5}$$

capturing both channel responses and local fluctuations.

Routing probabilities are computed as $\mathbf{p}_b = \mathrm{Softmax}(\mathbf{W}_r \mathbf{g}_b + \mathbf{b}_r)$, where $\mathbf{W}_r \in \mathbb{R}^{N\times 2C}$ and $\mathbf{b}_r \in \mathbb{R}^N$ are learnable parameters. Following Top-2 routing Shazeer et al. (2017), we select the two experts with the highest probabilities, denoted by index set $\mathcal{I}_b$, and normalize their weights as

$$\hat{p}_b^{(i)} = \alpha \, \frac{p_b^{(i)}}{\sum_{j\in\mathcal{I}_b} p_b^{(j)}}, \qquad i \in \mathcal{I}_b, \tag{6}$$

where $\alpha$ is the routing scale factor.

**(2) Base Expert.** To capture local details in high-frequency PDE features, we adopt depthwise separable convolution (DSC) Howard et al. (2017) as the shared backbone of all experts. It is an independent residual DSC block $\phi_{\mathrm{sh}}$ of the same form as Eq. (7):

$$\mathbf{y}_{\mathrm{sh}} = \phi_{\mathrm{sh}}(\mathbf{H}) = \mathbf{H} + \mathrm{PW}(\mathrm{DW}(\mathbf{H}))\,. \tag{7}$$

It provides stable base representations while reducing redundant learning among the specific experts.

**(3) Specific Experts (Non-Shared Experts).** The specific experts capture PDE-specific patterns by augmenting the shared backbone $\phi_{\mathrm{dsc}}$ with expert-specific low-rank adapters Hu et al. (2022). For the $i$-th expert, the adapter consists of a down-projection $\mathbf{B}_i \in \mathbb{R}^{r\times C}$ and an up-projection $\mathbf{A}_i \in \mathbb{R}^{C\times r}$ (both $1\times1$ convolutions), yielding

$$\mathbf{y}_i = \phi_{\mathrm{dsc}}(\mathbf{H}) + \mathbf{A}_i(\mathbf{B}_i(\mathbf{H}))\,, \qquad i = 1,\ldots,N. \tag{8}$$

The up-projection $\mathbf{A}_i$ is initialized to zero so that each specific expert initially equals $\phi_{\mathrm{dsc}}$, improving training stability.

#### 3.2.4 Frequency Branch Fusion

Through the Low-Rank Adapter-based High-frequency MoE, we obtain the high-frequency out $\mathbf{H}^{\mathrm{out}}$, which is then adaptively fused with the low-frequency output $\mathbf{Y}_{\mathrm{lf}}$ from the base expert via a pixel-wise gating module. Specifically, the gating weights are computed as $[\mathbf{G}_{\mathrm{lf}}, \mathbf{G}_{\mathrm{hf}}] =$

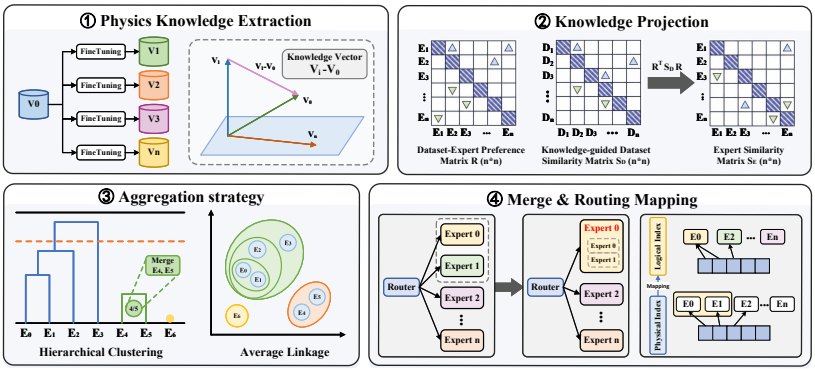



Figure 3: Knowledge-guided expert aggregation via bilinear projection and hierarchical clustering.

$\mathrm{Softmax}\big(\phi_g\big(\frac{1}{2}(\mathbf{Y}_{\mathrm{lf}} + \mathbf{H}^{\mathrm{out}})\big)\big)$, where $\phi_g$ is a lightweight $1\times1$ convolutional network. The fused output is $\mathbf{Y}_{\mathrm{out}} = \mathbf{G}_{\mathrm{lf}} \odot \mathbf{Y}_{\mathrm{lf}} + \mathbf{G}_{\mathrm{hf}} \odot \mathbf{H}^{\mathrm{out}}$.

### 3.3 Knowledge-Guided Expert Aggregation

Then, we propose a knowledge-guided expert aggregation strategy that groups experts learning similar high-frequency representations, reducing parameter and memory overhead while preserving expert specialization, as shown in Fig. 3.

#### 3.3.1 Expert Similarity

To measure expert similarity, we combine dataset-specific knowledge with routing preferences. For $D$ datasets and $N$ specific experts, the average routing probabilities form a dataset–expert matrix $\mathbf{R} \in \mathbb{R}^{D\times N}$, where $\mathbf{R}(d, i)$ denotes the mean routing probability of expert $i$ on dataset $d$.

Let $\mathbf{v}_0$ and $\mathbf{v}_d$ denote the pretrained and dataset-specific fine-tuned parameters, respectively. We represent dataset-specific knowledge as $\Delta\mathbf{v}_d = \mathbf{v}_d - \mathbf{v}_0$, and measure dataset similarity by cosine similarity: $\mathbf{S}_D(d_1, d_2) = \frac{\Delta\mathbf{v}_{d_1}^\top \Delta\mathbf{v}_{d_2}}{\|\Delta\mathbf{v}_{d_1}\|_2 \|\Delta\mathbf{v}_{d_2}\|_2}$, which is then projected into the expert space:

$$\mathbf{S}_E^{\mathrm{raw}} = \mathbf{R}^\top \mathbf{S}_D \mathbf{R} \in \mathbb{R}^{N\times N}. \tag{9}$$

To remove activation-scale effects, we normalize it as

$$\mathbf{S}_E(i, j) = \frac{\mathbf{S}_E^{\mathrm{raw}}(i, j)}{\sqrt{\mathbf{S}_E^{\mathrm{raw}}(i, i)\mathbf{S}_E^{\mathrm{raw}}(j, j)} + \varepsilon}, \tag{10}$$

where $\varepsilon$ ensures numerical stability. A larger $\mathbf{S}_E(i, j)$ indicates greater knowledge similarity between experts $i$ and $j$.

#### 3.3.2 Expert Aggregation Strategy

Based on $\mathbf{S}_E$, we perform threshold-constrained average-linkage hierarchical clustering Murtagh & Contreras (2012). Starting from each experts, the similarity between clusters $\mathcal{C}_a$ and $\mathcal{C}_b$ is defined as

$$\mathrm{Sim}(\mathcal{C}_a, \mathcal{C}_b) = \frac{1}{|\mathcal{C}_a|\,|\mathcal{C}_b|} \sum_{i\in\mathcal{C}_a} \sum_{j\in\mathcal{C}_b} \mathbf{S}_E(i, j). \tag{11}$$

The most similar clusters are iteratively merged until the maximum similarity falls below the threshold $\tau$. Within each cluster $\mathcal{C}_k$, experts are aggregated according to their global routing frequencies, with $\omega_i = f_i / \sum_{j\in\mathcal{C}_k} f_j$, where $f_i$ denotes the activation frequency of expert $i$. Given the adapter parameters $\boldsymbol{\theta}_i = \{\mathbf{A}_i, \mathbf{B}_i\}$, the aggregated expert is

$$\bar{\boldsymbol{\theta}}_k = \sum_{i\in\mathcal{C}_k} \omega_i \boldsymbol{\theta}_i. \tag{12}$$

The aggregated experts replace the original ones, and the routing mapping is updated accordingly for inference.

Table 1: The experiments cover four settings: small models, foundation-model pre-training, fine-tuning, and model merging. L2RE is used for evaluation. The best result in each setting is shown in **bold**, and the overall best in blue bold.

| | L2RE | Activated | FNO-$\nu$ | | | PDEBench CNS-($\eta$, $\zeta$), DR, SWE | | | | | | | | PDEArena | | CFDBench |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Model | Params | 1e-5 | 1e-4 | 1e-3 | 1,0.1 | 1,0.01 | Avg.(1) | 0.1,0.1 | 0.1,0.01 | Avg.(0.1) | DR | SWE | NS | NS-cond | - |
| Small model | FNO | 0.5M | **0.116** | 0.0922 | 0.0156 | 0.151 | 0.108 | 0.130 | 0.230 | 0.076 | 0.153 | 0.0321 | 0.0091 | 0.210 | 0.384 | 0.0274 |
| | UNet | 25M | 0.198 | 0.119 | 0.0245 | 0.334 | 0.291 | 0.313 | 0.569 | 0.357 | 0.463 | 0.0971 | 0.0521 | 0.102 | 0.337 | 0.209 |
| | FFNO | 1.3M | 0.121 | 0.0503 | 0.0099 | 0.0212 | 0.0520 | 0.0366 | 0.162 | 0.0452 | 0.104 | 0.0571 | 0.0116 | 0.0839 | 0.602 | 0.0071 |
| | GK-T | 1.6M | 0.134 | 0.0792 | **0.0098** | 0.0341 | **0.0377** | **0.0359** | 0.0274 | 0.0366 | 0.0320 | 0.0359 | 0.0069 | **0.0952** | 0.423 | 0.0105 |
| | GNOT | 1.8M | 0.157 | **0.0443** | 0.0125 | **0.0325** | 0.0420 | 0.0373 | **0.0228** | 0.0341 | 0.0285 | 0.0311 | **0.0068** | 0.172 | **0.325** | **0.0088** |
| | OFormer | 1.9M | 0.171 | 0.0645 | 0.0104 | 0.0417 | 0.0625 | 0.0521 | 0.0254 | **0.0205** | **0.0229** | **0.0192** | 0.0072 | 0.135 | 0.332 | 0.0102 |
| Pre-trained | MPP | 7M | - | - | - | - | - | 0.0442 | - | - | 0.0312 | **0.0168** | 0.0066 | - | - | - |
| | DPOT | 7M | 0.0976 | 0.0606 | 0.0095 | 0.0173 | 0.0397 | 0.0285 | 0.0132 | 0.0220 | 0.0176 | 0.0321 | 0.0056 | **0.125** | 0.384 | **0.0095** |
| | NESTOR | 13M | 0.1195 | 0.0951 | 0.0093 | 0.0167 | 0.0373 | 0.0270 | 0.0120 | 0.0202 | 0.0161 | 0.0308 | 0.0052 | 0.132 | 0.409 | 0.0112 |
| | Unisolver | 12M | **0.0600** | 0.0626 | 0.0148 | 0.0464 | 0.1172 | 0.0818 | 0.2510 | 0.1170 | 0.1840 | 0.7276 | 0.6851 | 0.131 | 0.565 | 0.0627 |
| | MoE-POT | 17M | 0.0965 | **0.0588** | 0.0090 | 0.0160 | 0.0395 | 0.0278 | 0.0126 | 0.0215 | 0.0171 | 0.0310 | 0.0053 | 0.134 | 0.367 | 0.0103 |
| | **Ours** | 11M | 0.0960 | 0.0770 | **0.0087** | **0.0120** | **0.0372** | **0.0246** | **0.0116** | **0.0184** | **0.0150** | 0.0403 | **0.0048** | 0.127 | **0.361** | 0.0115 |
| Fine-tune | DPOT-FT500 | 7M | 0.0520 | 0.0367 | 0.0058 | 0.0112 | 0.0195 | 0.0153 | 0.0174 | 0.0138 | 0.0156 | 0.0148 | **0.0024** | 0.0910 | 0.280 | 0.0039 |
| | NESTOR-FT500 | 13M | **0.0505** | 0.0217 | 0.0043 | 0.0094 | 0.0134 | 0.0114 | **0.0123** | 0.0083 | **0.0103** | **0.0117** | 0.0026 | 0.0683 | 0.285 | **0.0038** |
| | MoE-POT-FT500 | 17M | 0.0588 | 0.0352 | 0.0055 | 0.0108 | 0.0185 | 0.0147 | 0.0163 | 0.0131 | 0.0147 | 0.0140 | 0.0023 | 0.0985 | **0.263** | 0.0042 |
| | **Ours-FT500** | 11M | 0.0543 | **0.0197** | **0.0040** | **0.0090** | **0.0130** | **0.0110** | 0.0132 | **0.0082** | 0.0107 | 0.0136 | 0.0030 | **0.0637** | 0.291 | **0.0038** |
| Merge | **Ours-merge** | 11M | **0.0541** | **0.0192** | **0.0040** | **0.0090** | **0.0132** | **0.0111** | 0.0127 | **0.0079** | **0.0102** | **0.0130** | **0.0031** | **0.0637** | **0.279** | **0.0037** |
| | Ours-OPCM | 11M | 0.0670 | 0.0221 | 0.0051 | 0.0102 | 0.0140 | 0.0121 | 0.0132 | 0.0090 | 0.0111 | 0.0210 | 0.0051 | 0.0702 | 0.290 | 0.0041 |
| | Ours-SMOE | 11M | 0.0550 | 0.0200 | 0.0044 | 0.0094 | 0.0141 | 0.0118 | **0.0123** | 0.0084 | 0.0104 | 0.0135 | 0.0045 | 0.0640 | 0.293 | 0.0041 |

# 4 EXPERIMENTS

## 4.1 DATASETS AND BASELINES

**Datasets.** We use a mixed large-scale pre-training set of 12 PDE datasets with diverse equation types and parameter settings from FNO Li et al. (2020), PDEBench Takamoto et al. (2022), PDEArena Gupta & Brandstetter (2022), and CFDBench Luo et al. (2023).

**Baselines.** We compare against representative neural operators and PDE foundation models, including FNO Li et al. (2020), UNet Ronneberger et al. (2015), GK-T Cao (2021), GNOT Hao et al. (2023), OFormer Li et al. (2022), MPP McCabe et al. (2023), DPOT Hao et al. (2024), UniSolver Zhou et al. (2025), MoE-POT Wang et al. (2026), NESTOR Sun et al. (2026), and Poseidon Herde et al. (2024). UNet, GK-T, GNOT, and OFormer are trained separately on each dataset, while the others follow their original pre-training and transfer settings.

## 4.2 MAIN RESULTS

Table 1 compares our method across pre-training, fine-tuning, and expert merging. Overall, our method achieves the best performance on 8 of the 12 tasks.

**Pre-training Stage.** Our method achieves the best performance on 7 tasks, including 5 of the 6 PDEBench tasks, while maintaining low L2RE on most others. This demonstrates that the proposed MoE architecture effectively learns shared physical knowledge across heterogeneous PDEs with high parameter efficiency.

**Fine-tuning Stage.** We fine-tune the pre-trained model for 500 epochs on each downstream dataset. The resulting model achieves the best performance on 7 tasks. Compared with training from scratch, pre-training enables faster adaptation and higher prediction accuracy, indicating strong transferability across PDE tasks.

**Expert Merging.** We compress the pre-trained model by aggregating its 12 non-shared experts into 6 based on knowledge-guided similarities, and then fine-tune both the original and merged models on the 12 downstream datasets. Despite a 50% reduction in expert count, Ours-Merge maintains or improves performance on most tasks. For example, L2RE on DR decreases from 0.0136 to 0.0130, indicating nearly lossless compression and suggesting that expert aggregation can reduce redundancy and routing interference.

Overall, the pre-training and fine-tuning paradigm improves performance and adaptation across PDE tasks, while knowledge-guided expert aggregation enables efficient compression with minimal performance degradation, demonstrating strong cross-task generalization and parameter efficiency.

Table 2: Performance comparison of different methods under the new dataset split.

| Model | Activated | FNO-$\nu$ | | PDEBench | | | CFDBench |
|---|---|---|---|---|---|---|---|
| L2RE | Params | 1e-5 | 1e-3 | 0.1,0.01 | SWE | DR | - |
| FNO | 10M | 0.1910 | 0.0245 | 0.0859 | 11.0000 | 0.5300 | 0.0060 |
| DPOT | 7M | 0.1070 | 0.0155 | 0.0168 | 0.0063 | 0.0577 | 0.0067 |
| MoE-POT | 17M | 0.0682 | 0.0077 | **0.0105** | 0.0064 | 0.0411 | **0.0053** |
| NESTOR | 13M | 0.0674 | 0.0076 | 0.0159 | 0.0045 | 0.0184 | 0.0091 |
| Ours | 11M | **0.0582** | **0.0051** | 0.0165 | **0.0024** | **0.0145** | 0.0056 |

### 4.3 Experiments on New PDE Dataset Splits

To evaluate different pre-training data configurations, we select 6 subsets from 12 PDE datasets and compare our method with other pre-trained models (Table 2). Our method achieves the best results on 4 of 6 datasets and remains competitive on the others. Despite using only half the pre-training data, it learns transferable representations from heterogeneous PDEs and generalizes well to downstream tasks.

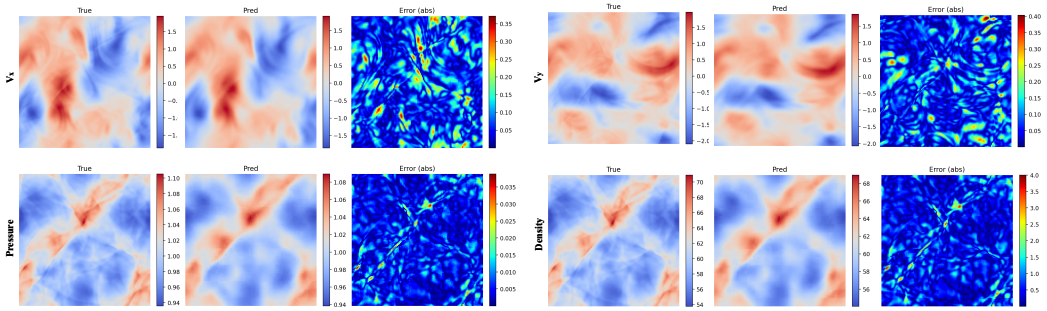
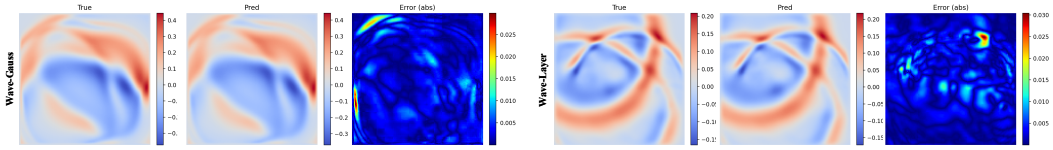



Figure 4: Final-step predictions for 2D high-resolution turbulence.




(a) Wave-Gauss. (b) Wave-Layer.

Figure 5: Final-step predictions for Wave-Layer and Wave-Gauss.

### 4.4 Downstream Task Experiments

Table 3: Comparison of transfer ability.

| Model | Activated Params | Wave-Layer | Wave-Gauss | Turb |
|---|---|---|---|---|
| FNO | 0.5M | - | - | 0.193 |
| MPP | 7M | - | - | 0.152 |
| DPOT | 7M | 0.0095 | 0.0075 | 0.135 |
| Poseidon | 21M | 0.29 | 0.29 | - |
| MoE-POT | 17M | 0.05 | 0.06 | **0.0621** |
| **Ours** | 11M | **0.0085** | **0.0066** | 0.0662 |

To evaluate generalization and transferability, we evaluate our model on a high-resolution 2D turbulence dataset and two unseen PDEs, Wave-Gauss and Wave-Layer. The results are reported in Table 3, Figures 4 and Figures 5. Our method achieves the best performance on both unseen PDEs. It obtains L2RE values of 0.0085 on Wave-Layer and 0.0066 on Wave-Gauss, also achieves the second-best result on high-resolution turbulence and outperforms most baselines. These results demonstrate that large-scale pre-training enables transferable representations and improves adaptation to unseen PDEs and complex downstream dynamics.

### 4.5 Effectiveness Analysis of Expert Aggregation

To evaluate the proposed aggregation strategy, we visualize the key matrices involved and compare downstream performance before and after aggregation.

Figure 6 (1) shows the dataset similarity matrix $\mathbf{S}_D$, computed from $\Delta\mathbf{v}_d = \mathbf{v}_d - \mathbf{v}_0$. Clear block structure is observed: datasets with similar physical mechanisms exhibit higher similarity, while

Table 5: Ablation experiments. "w/o" denotes the removal of the corresponding component.

| Method | 1e-5 | 1e-3 | 0.1, 0.01 | SWE | DR | CFDBench | Avg. $\mathcal{L}_2$ | Promotion |
|---|---|---|---|---|---|---|---|---|
| Ours | 0.0582 | 0.0051 | 0.0165 | 0.0024 | 0.0145 | 0.0056 | **0.0171** | - |
| w/o MoE | 0.0709 | 0.0054 | 0.0165 | 0.0031 | 0.0196 | 0.0079 | 0.0206 | 0.0035 |
| w/o High-Freq | 0.1065 | 0.0157 | 0.0170 | 0.0066 | 0.0573 | 0.0071 | 0.0350 | **0.0179** |
| w/o Aux Loss | 0.0694 | 0.0055 | 0.0154 | 0.0027 | 0.0185 | 0.0098 | 0.0202 | 0.0031 |
| w/o Orth Loss | 0.0692 | 0.0051 | 0.0168 | 0.0026 | 0.0189 | 0.0083 | 0.0202 | 0.0031 |

different equation types show weaker correlations, indicating that the learned knowledge captures dataset-specific characteristics.

Figure 6 (2) presents the dataset–expert preference matrix $\mathbf{R}$. The sparse and locally concentrated pattern indicates that different datasets activate distinct experts, confirming that routing promotes expert specialization. The knowledge-guided expert similarity matrix $\mathbf{S}_E$, obtained by combining $\mathbf{S}_D$ and $\mathbf{R}$, is shown in Figure 6 (3). High-similarity regions reflect knowledge overlap among experts, guiding the aggregation of redundant experts while preserving essential knowledge. As shown in Table 1, reducing the number of experts from 12 to 6 achieves 50% compression with negligible performance loss, and even improves performance on certain tasks. Overall, merging experts reduces redundancy while preserving generalization.

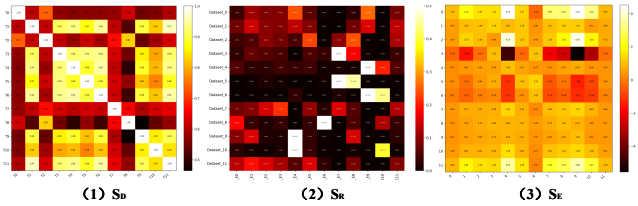



Figure 6: (1) dataset similarity matrix $\mathbf{S}_D$, (2) dataset–expert matrix $\mathbf{R}$, (3) expert similarity matrix $\mathbf{S}_E$.

### 4.6 Scaling Experiments

We pre-trained on 6 subsets of 12 PDE datasets and vary the number of non-shared experts while fixing the number of activated experts. As shown in Table 4, 12 non-shared experts achieve the best performance. Further increasing the expert scale does not always improve performance, as excessive experts complicate routing and may weaken expert collaboration. Therefore, an appropriate expert scale is essential for MoE modeling.

Table 4: The impact of the number of non-shared experts.

| Experts | 1e-5 | 1e-3 | 0.1, 0.01 | SWE | DR | CFDBench | Avg. $\mathcal{L}_2$ |
|---|---|---|---|---|---|---|---|
| 4 | 0.0666 | 0.0054 | 0.0162 | 0.0029 | 0.0193 | 0.0079 | 0.0197 |
| 8 | 0.0679 | 0.0049 | 0.0170 | 0.0029 | 0.0187 | 0.0083 | 0.0200 |
| 12 | 0.0582 | 0.0051 | 0.0165 | 0.0024 | 0.0145 | 0.0056 | **0.0171** |
| 16 | 0.0689 | 0.0048 | 0.0165 | 0.0040 | 0.0192 | 0.0077 | 0.0202 |
| 20 | 0.0700 | 0.0049 | 0.0170 | 0.0028 | 0.0197 | 0.0092 | 0.0206 |
| 24 | 0.0693 | 0.0049 | 0.0162 | 0.0031 | 0.0187 | 0.0082 | 0.0201 |

### 4.7 Ablation Studies

We conduct ablation studies on 6 PDE subtasks, with results reported in Table 5. **(1) w/o High-Freq.** Removing the high-frequency branch increases the average L2RE by 0.0179, causing the largest performance degradation among all variants. This confirms its critical importance in capturing fine-grained structures and sharp local variations. **(2) w/o MoE.** Removing the MoE architecture increases the average L2RE by 0.0035, demonstrating that dynamic expert routing effectively improves the modeling of heterogeneous PDE patterns. **(3) w/o Aux Loss.** Removing the auxiliary load-balancing loss increases the average L2RE by 0.0031. Although the impact is relatively modest, it promotes balanced expert utilization and stabilizes training. **(4) w/o Orth Loss.** Removing the orthogonality loss increases the average L2RE by 0.0031, indicating that it encourages diverse expert representations, reduces redundant feature learning, and promotes specialization among experts.

## 5 Conclusion

We presented SPEAR, a spectral-disentangled MoE neural operator with knowledge-guided expert aggregation for large-scale PDE pre-training. SPEAR separates shared low-frequency dynamics from PDE-specific high-frequency patterns and reduces expert redundancy through knowledge-guided aggregation while preserving performance. Extensive experiments demonstrate that SPEAR achieves strong pre-training, transferability, and generalization to unseen PDE systems, providing an efficient and scalable solution for PDE foundation models.

## AI USE STATEMENT

We used generative AI tools throughout this work: to refine the idea and the experimental design, to implement the method, to process and interpret results, to search related literature, to produce figures and format references, and to draft and polish sections of this paper. Tasks involving synthetic data generation and mathematical proofs are not applicable to this work. We have reviewed all AI-assisted work and take responsibility for the final content of this paper, including text, claims or artifacts produced with the aid of generative AI.

## ETHICS STATEMENT

Our work focuses on developing neural operators for large-scale PDE pre-training and expert aggregation, and poses no foreseeable ethical risks. The datasets used in our experiments consist of numerical simulations of physical systems and do not involve human subjects or personally identifiable information. Our study is intended to advance efficient and generalizable scientific computing methods.

## REPRODUCIBILITY STATEMENT

To facilitate reproducibility, we provide detailed descriptions of the model architecture, expert aggregation strategy, training settings, and dataset configurations in the main paper and appendix. Key hyperparameters and implementation details are also reported. The source code and additional experimental configurations will be provided in the supplementary materials.

## REFERENCES

Shuhao Cao. Choose a transformer: Fourier or galerkin. *Advances in Neural Information Processing Systems*, 34:24924–24940, 2021.

I Chen, Hsu-Shen Liu, Wei-Fang Sun, Chen-Hao Chao, Yen-Chang Hsu, Chun-Yi Lee, et al. Retraining-free merging of sparse moe via hierarchical clustering. *arXiv preprint arXiv:2410.08589*, 2024a.

Tianyu Chen, Haoyi Zhou, Ying Li, Hao Wang, Chonghan Gao, Rongye Shi, Shanghang Zhang, and Jianxin Li. Omniarch: Building foundation model for scientific computing. *arXiv preprint arXiv:2402.16014*, 2024b.

Tianyu Chen, Haoyi Zhou, Ying Li, Hao Wang, Zhenzhe Zhang, Tianchen Zhu, Shanghang Zhang, and Jianxin Li. Freqmoe: dynamic frequency enhancement for neural pde solvers. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence*, 2025.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022.

John Guibas, Morteza Mardani, Zongyi Li, Andrew Tao, Anima Anandkumar, and Bryan Catanzaro. Adaptive fourier neural operators: Efficient token mixers for transformers. *arXiv preprint arXiv:2111.13587*, 2021.

Jayesh K Gupta and Johannes Brandstetter. Towards multi-spatiotemporal-scale generalized pde modeling. *arXiv preprint arXiv:2209.15616*, 2022.

Zhongkai Hao, Zhengyi Wang, Hang Su, Chengyang Ying, Yinpeng Dong, Songming Liu, Ze Cheng, Jian Song, and Jun Zhu. Gnot: A general neural operator transformer for operator learning. In *International conference on machine learning*, pp. 12556–12569. PMLR, 2023.

Zhongkai Hao, Chang Su, Songming Liu, Julius Berner, Chengyang Ying, Hang Su, Anima Anandkumar, Jian Song, and Jun Zhu. Dpot: Auto-regressive denoising operator transformer for large-scale pde pre-training. *arXiv preprint arXiv:2403.03542*, 2024.

Maximilian Herde, Bogdan Raonić, Tobias Rohner, Roger Käppeli, Roberto Molinaro, Emmanuel De Bezenac, and Siddhartha Mishra. Poseidon: Efficient foundation models for pdes. *Advances in Neural Information Processing Systems*, 37:72525–72624, 2024.

Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Liang Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *Iclr*, 1(2):3, 2022.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.

Lujun Li, Zhu Qiyuan, Jiacheng Wang, Wei Li, Hao Gu, Sirui Han, and Yike Guo. Sub-moe: Efficient mixture-of-expert llms compression via subspace expert merging. In *AAAI 2026*, 2026.

Pingzhi Li, Zhenyu Zhang, Prateek Yadav, Yi-Lin Sung, Yu Cheng, Mohit Bansal, and Tianlong Chen. Merge, then compress: Demystify efficient smoe with hints from its routing policy. In *International Conference on Learning Representations*, volume 2024, pp. 14234–14256, 2024.

Zijie Li, Kazem Meidani, and Amir Barati Farimani. Transformer for partial differential equations' operator learning. *arXiv preprint arXiv:2205.13671*, 2022.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. *arXiv preprint arXiv:2010.08895*, 2020.

Xinliang Liu, Bo Xu, Shuhao Cao, and Lei Zhang. Mitigating spectral bias for the multiscale operator learning. *Journal of Computational Physics*, 506:112944, 2024. ISSN 0021-9991.

Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. *Nature machine intelligence*, 3(3):218–229, 2021.

Yining Luo, Yingfa Chen, and Zhen Zhang. Cfdbench: A large-scale benchmark for machine learning methods in fluid dynamics. *arXiv preprint arXiv:2310.05963*, 2023.

Michael McCabe, Bruno Régaldo-Saint Blancard, Liam Holden Parker, Ruben Ohana, Miles Cranmer, Alberto Bietti, Michael Eickenberg, Siavash Golkar, Geraud Krawezik, Francois Lanusse, et al. Multiple physics pretraining for physical surrogate models. *arXiv preprint arXiv:2310.02994*, 2023.

Fionn Murtagh and Pedro Contreras. Algorithms for hierarchical clustering: an overview. *Wiley interdisciplinary reviews: data mining and knowledge discovery*, 2(1):86–97, 2012.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241. Springer, 2015.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

Dengdi Sun, Xiaoya Zhou, Xiao Wang, Hao Si, Wanli Lyu, Jin Tang, and Bin Luo. Nestor: A nested moe-based neural operator for large-scale pde pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6147–6156, 2026.

Makoto Takamoto, Timothy Praditia, Raphael Leiteritz, Daniel MacKinlay, Francesco Alesiani, Dirk Pflüger, and Mathias Niepert. Pdebench: An extensive benchmark for scientific machine learning. *Advances in neural information processing systems*, 35:1596–1611, 2022.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Hong Wang, Haiyang Xin, Jie Wang, Xuanze Yang, Fei Zha, Yan Jiang, et al. Mixture-of-experts operator transformer for large-scale pde pre-training. *Advances in Neural Information Processing Systems*, 38:31498–31527, 2026.

Dengming Zhang, Xiaowen Ma, Zhen-Liang Ni, Zhenkai Wu, Han Shu, Xin Jiang, and Xinghao Chen. Expert merging: Model merging with unsupervised expert alignment and importance-guided layer chunking. In *The Fourteenth International Conference on Learning Representations*, 2026.

Hang Zhou, Yuezhou Ma, Haixu Wu, Haowen Wang, and Mingsheng Long. Unisolver: Pde-conditional transformers towards universal neural pde solvers. In *Proceedings of the International Conference on Machine Learning*, 2025.

# A Appendix

## A.1 Input Encoding and Time Aggregation

Input a PDE sequence $\mathbf{x} \in \mathbb{R}^{B\times H\times W\times T\times C}$, we concatenate it with the coordinate grid and reshape it as $\widetilde{\mathbf{x}} \in \mathbb{R}^{(BT)\times(C+3)\times H\times W}$. Following ViT patch embedding Dosovitskiy et al. (2020), we obtain

$$\mathbf{X} = \mathrm{PatchEmbed}(\widetilde{\mathbf{x}}) + \mathbf{E}_{\mathrm{pos}} \in \mathbb{R}^{(BT)\times D\times H'\times W'}, \tag{13}$$

where $H' = H/P_H$, $W' = W/P_W$, and $\mathbf{E}_{\mathrm{pos}}$ is the learnable positional embedding.

After reshaping $\mathbf{X}$ to $B \times H' \times W' \times T \times D$, aggregate temporal information Hao et al. (2024) as

$$\mathbf{Y}_{b,h,w} = \sum_{t=1}^{T} \mathbf{X}_{b,h,w,t}\mathbf{P}_t, \tag{14}$$

where $\mathbf{P}_t \in \mathbb{R}^{D\times D}$ is the learnable projection for time step $t$, yielding $\mathbf{Y} \in \mathbb{R}^{B\times H'\times W'\times D}$.

## A.2 Loss Functions

**(1) Load Balancing Loss** To promote balanced expert utilization, we adopt the load balancing loss Shazeer et al. (2017), defined as $\mathcal{L}_{\mathrm{aux}} = N\sum_{i=1}^{N} p_i f_i$, where $N$ is the number of non-shared experts, and $p_i$ and $f_i$ denote the mean routing probability and activation frequency of expert $i$, respectively.

**(2) Main Task Loss** We use the relative $L_2$ reconstruction error (L2RE) Li et al. (2020) as the main objective, defined as $\mathcal{L}_{\mathrm{L2RE}} = \frac{\|\hat{\mathbf{u}}_i^{(c)} - \mathbf{u}_i^{(c)}\|_2}{\|\mathbf{u}_i^{(c)}\|_2}$, where $\mathbf{u}_i^{(c)}$ and $\hat{\mathbf{u}}_i^{(c)}$ denote the ground-truth and predicted fields of the $i$-th sample on channel $c$.

**(3) Orthogonality Loss** To encourage expert diversity, we minimize the pairwise cosine similarity between the flattened up-projection parameters $\mathbf{A}_i$ of non-shared experts, using $\mathcal{L}_{\mathrm{orth}} = \frac{1}{|\mathcal{P}|}\sum_{(i,j)\in\mathcal{P}}\left(\frac{\mathbf{A}_i^\top \mathbf{A}_j}{\|\mathbf{A}_i\|_2\|\mathbf{A}_j\|_2}\right)^2$, where $\mathcal{P} = \{(i,j) \mid 1 \le i < j \le N\}$ denotes all unordered expert pairs.

**(4) Overall Loss** The overall objective is $\mathcal{L}_{\mathrm{total}} = \mathcal{L}_{\mathrm{L2RE}} + \lambda_1\mathcal{L}_{\mathrm{aux}} + \lambda_2\mathcal{L}_{\mathrm{orth}}$. We set $\lambda_1 = 0.001$, while $\lambda_2$ decays from $0.1$ to $0.01$ over the first 400 epochs and reaches $0$ at epoch 450.

Table 6: Comparison of activation parameters, total parameters, and activation rates of different models.

| | DPOT | MoE-POT | NESTOR | Ours (-LoRA) | Ours |
|---|---|---|---|---|---|
| Activated Parameters | 7.5M | 17M | 13M | 11.34M | 11.05M |
| Total Parameters | 7.5M | 30M | 83M | 22.13M | 13.36M |
| Activation Ratio | 100% | 56.67% | 15.66% | 51.24% | 82.71% |

### A.3 Efficiency Analysis

We compared the model parameter efficiency of traditional single-network, standard MoE, and LoRA-style MoE architectures, as shown in Table 6.

Traditional single-network architectures (such as DPOT) have all parameters involved in inference (100% activation). Increasing their model capacity typically relies on expanding the network size, leading to a linear increase in computational cost with the number of parameters. In contrast, the MoE architecture, by selectively activating some experts, effectively controls computational cost while significantly expanding model capacity.

Furthermore, under the same configuration (12 non-shared experts, Top-2 activation), we compared the expert designs of LoRA-style and standard MoE. The results show that the total number of parameters in the standard MoE model is almost twice that of the LoRA-style MoE, and this parameter gap widens further with the increase in the number of experts.

In summary, the LoRA-style MoE architecture, while maintaining strong scalability, significantly reduces model parameter redundancy and effectively alleviates memory pressure, providing an efficient model expansion scheme for large-scale PDE pre-training.

As shown in Table 7, expert merging substantially reduces the storage overhead of the model. For the four MoE layers, the parameters of the non-shared experts decrease from 1.573M to 0.786M, corresponding to a reduction of approximately 50.0%. Meanwhile, the checkpoint size decreases from 138 MB to 47 MB, reducing the storage footprint by 65.9%.

It is worth noting that our experts are implemented with lightweight low-rank adapters, and therefore account for only a relatively small portion of the overall model parameters. Consequently, the absolute parameter reduction reported here is relatively modest. For conventional MoE architectures with full-parameter experts, the benefit of expert merging is expected to become more pronounced as the number of experts and stacked MoE layers increases, since expert-related parameters grow with both dimensions. Thus, expert merging provides greater potential storage savings for larger and more deeply stacked MoE models.

Table 7: Efficiency comparison before and after expert merging.

| **Metric** | **Before** | **After** | **Reduction** |
|---|---|---|---|
| Expert Params. (4 layers) | 1.573M | 0.786M | 50.0% |
| Checkpoint Size | 138 MB | 47 MB | 65.9% |

### A.4 Data Preprocessing and Sampling

**Data Padding and Masking.** Different PDE datasets vary in resolution, number of variables, and geometric configurations. If we directly sample from the raw data, the resulting batch will have large variations in size, leading to unbalanced training loads and reduced efficiency in modern multi-GPU training. Here, we adopt the padding and masking strategy from DPOT. First, we select a fixed resolution $H = 128$, which matches a considerable portion of the datasets. Datasets with lower resolutions are upsampled to $H$ via interpolation, while those with higher resolutions are randomly downsampled or interpolated to $H$. Second, to unify the number of variables across different PDEs, we pad all datasets along the channel dimension (e.g., filling with ones) to match the maximum number of channels. For datasets with irregular geometries, an additional mask channel is introduced to encode the specific geometric configuration of each PDE instance.

Table 8: Model and training settings.

| Dim | Ratio | Layers | Blocks | Modes | Patch | Routed | Shared | Top-$k$ | Rank | Batch | LR | Epochs |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 512 | 1 | 4 | 4 | 32 | 8 | 12 | 1 | 2 | 32 | 32 | $1 \times 10^{-3}$ | 1000 |

**Balanced Data Sampling.** When training with multiple PDE datasets, differences among datasets can lead to unbalanced training progress and inefficiency. To address this issue, we adopt the sampling strategy from DPOT, which balances the sampling probabilities across datasets during training. Our goal is to ensure that each dataset is represented equally throughout the training process. Let $|D_k|$ denote the number of samples in the $k$-th dataset ($1 \leq k \leq K$), and assign a weight $w_k$ to each dataset to indicate its relative importance. Then, the sampling probability from dataset $D_k$ is defined as:

$$p_k = \frac{w_k}{K\,|D_k|\,\sum_k w_k}$$

We can observe that the sampling probability depends on the weight $w_k$ rather than the dataset size $|D_k|$, which helps mitigate gradient imbalance caused by dataset size disparities.

### A.5 Comparison with Expert Merging Strategies

To further evaluate the effectiveness of our knowledge-guided expert merging strategy, we construct two comparison variants, Ours-OPCM and Ours-SMoE. Ours-OPCM adapts the continuous model merging strategy of OPCM to the MoE module and progressively merges the non-shared experts. This strategy is originally designed for general model merging and does not explicitly consider the differences in the knowledge learned by individual experts. Ours-SMoE follows the routing-guided idea of SMoE and measures expert correlations based on the routing probabilities in our model for expert merging.

The results show that Ours-Merge outperforms Ours-OPCM on all 14 metrics, indicating that directly applying a generic continuous model merging strategy is insufficient to preserve the structured functionality of MoE experts. Compared with Ours-SMoE, Ours-Merge achieves lower errors on 13 out of 14 metrics, with only one metric being slightly worse. For example, the L2RE decreases from 0.293 to 0.279 on PDEArena NS-cond and from 0.0045 to 0.0031 on PDEBench SWE. These results suggest that routing preferences can capture certain correlations among experts, but routing information alone is insufficient to fully characterize the functional knowledge learned by different experts. Incorporating dataset-specific learned knowledge enables more accurate identification of functionally similar experts and better preserves model performance during expert compression. This further indicates that routing similarity is not equivalent to knowledge similarity, highlighting the importance of incorporating learned knowledge into expert relationship modeling for effective expert merging.

### A.6 Aggregation Cost

Our expert aggregation introduces only marginal additional computational overhead. We reuse the dataset similarity matrix $\mathbf{S}_D$ constructed in NESTOR, avoiding additional dataset-specific fine-tuning for its estimation. The dataset–expert preference matrix $\mathbf{R}$ is directly accumulated from routing statistics during pre-training and therefore requires no additional forward passes or optimization. Given $\mathbf{S}_D$ and $\mathbf{R}$, the raw expert similarity matrix is obtained by the bilinear projection $\mathbf{R}^\top \mathbf{S}_D \mathbf{R}$, followed by normalization and average-linkage hierarchical clustering. Experts within each cluster are then aggregated once by weighted averaging of their LoRA adapter parameters according to global routing frequencies, after which the routing mapping is updated accordingly. Thus, the entire aggregation process is performed offline without retraining and incurs only limited computational cost compared with large-scale PDE pre-training.

### A.7 Experimental Details

We pre-train our model on 8 NVIDIA RTX 4090 GPUs using the Adam optimizer with an initial learning rate of $1 \times 10^{-3}$ and a cyclic learning rate schedule, including 200 warm-up epochs. The model is trained for 1000 epochs with a batch size of 32. To alleviate the imbalance caused by

different dataset sizes, we assign dataset-specific training weights during multi-PDE pre-training. All datasets are processed at a spatial resolution of $128 \times 128$. The model uses 32 Fourier modes, a hidden dimension of 512, and 4 layers. The MoE module contains 12 non-shared experts and one shared expert, with Top-2 routing and a low-rank dimension of 32. Detailed configurations are provided in Table 8.

## B Detailed Information of Datasets

We list the configurations of the PDE datasets used for pre-training along with detailed descriptions of the governing partial differential equations:

**FNO-$v$**: This dataset focuses on the temporal evolution of the two-dimensional incompressible fluid vorticity field $w(x,t)$, where $(x,t) \in [0,1]^2 \times [0,T]$. The dynamics are governed by the two-dimensional Navier–Stokes equations in the vorticity–streamfunction formulation:

$$\partial_t w + u \cdot \nabla w = \nu \Delta w + f(x), \qquad \nabla \cdot u = 0, \tag{15}$$

where $u$ denotes the velocity field, $\nu$ is the viscosity coefficient, $\Delta$ represents the Laplace operator, and $f(x)$ denotes the external forcing term. By varying the viscosity $\nu$, the dataset provides fluid dynamics simulations under different flow regimes, enabling the study of how viscosity influences the evolution of vortex structures.

**PDEBench-CMS**: This dataset focuses on the numerical simulation of compressible fluid mechanics (CMS). The goal is to predict the temporal evolution of the velocity field $u(x,t)$, the pressure field $p(x,t)$, and the density field $\rho(x,t)$ over the spatio-temporal domain $(x,t) \in [0,1]^2 \times [0,1]$. The data are generated based on the governing equations of compressible fluid dynamics, which consist of the conservation of mass, momentum, and energy:

$$\partial_t \rho + \nabla \cdot (\rho u) = 0, \tag{16}$$

$$\rho\left(\partial_t u + u \cdot \nabla u\right) = -\nabla p + \eta \Delta u + \left(\zeta + \tfrac{\eta}{3}\right) \nabla(\nabla \cdot u), \tag{17}$$

$$\partial_t \left(\tfrac{3}{2} p + \tfrac{\rho u^2}{2}\right) = -\nabla \cdot \left[\left(\varepsilon + p + \tfrac{\rho u^2}{2}\right) u - u \cdot \sigma'\right], \tag{18}$$

where $\eta$ denotes the shear viscosity coefficient and $\zeta$ the bulk viscosity coefficient.$\varepsilon$ is the energy density and $\sigma'$ is the stress tensor.

**PDEBench-SWE**: The dataset is derived from PDEBench and focuses on the numerical simulation of the Shallow Water Equations (SWE). The objective is to predict the water depth field $h(x,t)$ over the spatiotemporal domain $(x,t) \in [-1,1]^2 \times [0,5]$. The SWE is a set of approximate governing equations widely used in ocean dynamics, flood modeling, and geomorphological evolution studies. The governing equations are given as follows:

$$\partial_t h + \nabla \cdot (hu) = 0, \tag{19}$$

$$\partial_t (hu) + \nabla \cdot \left(\tfrac{1}{2} h u^2 + \tfrac{1}{2} g r h^2\right) = -g r h \nabla b, \tag{20}$$

**PDEBench-DR**: The dataset is derived from PDEBench and focuses on the numerical simulation of diffusion–reaction (DR) systems. The objective is to predict the density field $u(x,t)$ over the spatiotemporal domain $(x,t) \in [-2.5, 2.5]^2 \times [0,1]$. The governing equation is given by:

$$\partial_t u = D \nabla^2 u + R(u), \tag{21}$$

where $D$ is the diffusion coefficient and $R(u)$ denotes the nonlinear reaction term.

**PDEArena**: The dataset is derived from PDEArena and focuses on the numerical simulation of incompressible Navier–Stokes (NS) flows. The objective is to predict the velocity field $u(x,t)$, pressure field $p(x,t)$, and density field $\rho(x,t)$ over the spatiotemporal domain $(x,t) \in [0,32]^2 \times [0,24]$.

The 2D incompressible Navier–Stokes equations are given by:

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} = -\nabla p + \nu \Delta \mathbf{u}, \tag{22}$$

$$\nabla \cdot \mathbf{u} = 0, \tag{23}$$

where $\mathbf{u} = (u, v)^\top$ is the velocity field, $p$ is the pressure, and $\nu$ is the kinematic viscosity.

NS-cond introduces additional physical conditions such as forcing fields $\mathbf{f}(\mathbf{x}, t)$ or spatially varying viscosity $\nu(\mathbf{x})$:

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} = -\nabla p + \nu(\mathbf{x})\Delta \mathbf{u} + \mathbf{f}(\mathbf{x}, t), \tag{24}$$

$$\nabla \cdot \mathbf{u} = 0. \tag{25}$$

Here, $\mathbf{f}(\mathbf{x}, t)$ denotes external forcing and $\nu(\mathbf{x})$ can vary spatially.

**CFDBench**: The dataset is derived from CFDBench and focuses on the numerical simulation of incompressible or weakly compressible flows in irregular geometries. The objective is to predict the velocity field $u(x, t)$ and the pressure field $p(x, t)$ over domains with complex boundaries. The governing equations are given as follows:

$$\partial_t(\rho u) + \nabla \cdot (\rho u^2) = -\nabla p + \nabla \cdot \mu(\nabla u + \nabla u^\top), \tag{26}$$

$$\nabla \cdot (\rho u) = 0, \tag{27}$$

where $\rho$ is the fluid density, $u$ is the velocity field, $p$ is the pressure, and $\mu$ denotes the viscosity coefficient.