Distillation of Foundation Models for Time-dependent PDEs

Daniel Musekamp[1]* **Boshra Ariguib**[1, 2] **Andrei Manolache**[1,3] **Mathias Niepert**[1]

[1]University of Stuttgart [2]University of Cologne [3]Bitdefender

Abstract

Foundation models for time-dependent partial differential equations (PDEs) are trained on large and diverse collections of physical systems and can generalize effectively to new downstream tasks. After fine-tuning on only a few trajectories from a target domain, they can achieve strong accuracy in low-data regimes. However, these models are typically large and computationally intensive, limiting their usefulness as fast surrogates for numerical solvers. We propose Teacher Rollout Extension (TREX), a knowledge distillation framework that transfers the predictive capability of a pretrained foundation model into a compact and efficient student. Starting from a fine-tuned teacher, TREX augments limited downstream data by generating long synthetic trajectories through teacher rollouts, optionally with periodic noise injection. This procedure samples from the teacher-induced rollout distribution without requiring explicit knowledge of the initial condition distribution, while exposing the student to long-horizon states and local recovery behavior around states encountered during autoregressive prediction. The student can further incorporate task-specific inductive biases, such as equivariance, that the teacher does not necessarily enforce. We evaluate TREX on multiple PDE benchmarks. The resulting students can match or surpass the teacher's accuracy while reducing the number of parameters by several orders of magnitude and achieving more than an order-of-magnitude speedup in inference.

1 Introduction

Spatiotemporal physical processes such as fluid dynamics or cell growth can be described using partial differential equations (PDEs). Hence, solving PDEs is integral to science and engineering, for example, in astrophysics, design optimization, or weather prediction. Since solving PDEs for large systems and nonlinear dynamics using numerical solvers can be slow, recent research has investigated applying neural networks as surrogate models to learn solutions to PDEs. Compared with classical solvers, such approaches are differentiable out of the box and can incorporate experimental data, thereby enabling the modeling of effects not included in the simulation model. One core problem with neural PDE surrogates is that they require a training dataset, which is often generated using an expensive numerical solver, thereby making the surrogate modeling potentially uneconomical.

As a remedy, foundation models for PDEs have been introduced (McCabe et al., 2024; Herde et al., 2024; McCabe et al., 2026). By pretraining a surrogate on a large set of diverse physics, foundation models can transfer knowledge to downstream tasks with fewer training samples. On the downside, their inference is computationally expensive, as they require more parameters to generalize well. Slow inference prohibits their use in applications that have real-time requirements, such as planning and control tasks in manufacturing or energy systems (Tian et al., 2025; Janssens & Meyers, 2024). Furthermore, the architectures used are mostly large transformers that lack inductive biases, since

*Corresponding author: daniel.musekamp@ki.uni-stuttgart.de. Code available at: `github.com/dmusekamp/trex`

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

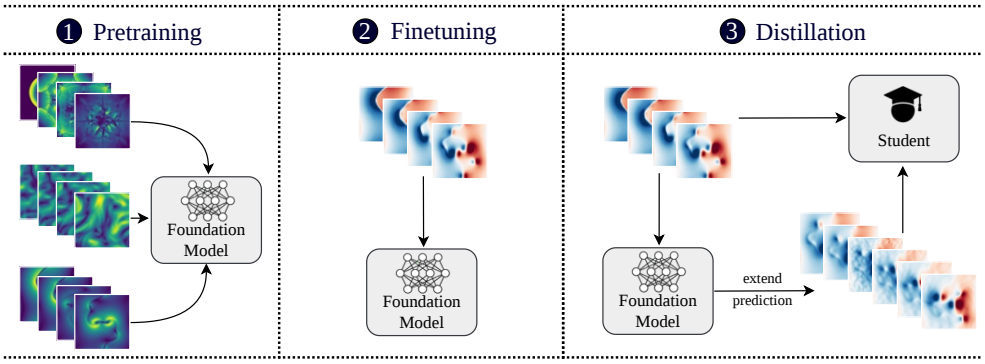

Figure 1: Foundation models are pretrained on a variety of PDE prediction tasks first (1), enabling them to adapt to a new task with only a few trajectories (2). To overcome their high inference time, we propose to distill the foundation model into a smaller and potentially equivariant student by extending the training trajectories using noisy rollouts of the teacher (3).

these may vary from problem to problem. For example, many PDEs are locally equivariant to rotations and translations, and globally as well if the boundary conditions are periodic. Including these equivariances in the architecture can make models more effective and data-efficient (Zhdanov et al., 2024; Helwig et al., 2023).

Therefore, we propose using knowledge distillation (KD) to reconcile the trade-off between fast inference and strong generalization (Fig. 1). Specifically, the pretrained foundation model is first fine-tuned on a small set of trajectories in the target domain. Afterward, the foundation model is used as the teacher to augment the available data with additional information. To our knowledge, we present the first systematic study of distilling general pretrained PDE foundation models into compact downstream surrogate models. Most prior work relies on specialized architectures or assumes full knowledge of the initial condition distribution (Zhang et al., 2025; Li et al., 2025). However, access to the downstream initial condition distribution is not always available. This is particularly relevant when data originates from experiments or partially observed systems, where only a limited set of trajectories is given. Additionally, such approaches are not applicable to models with multiple prior states as input, as the input needs to be a physically consistent state sequence.

Hence, we introduce Teacher Rollout Extension (**TREX**), a distillation technique that avoids the need to sample new initial conditions. Starting from the few available downstream trajectories, TREX generates long teacher rollouts and trains the student on the resulting teacher-labeled trajectories. This exposes the student to the teacher-induced rollout distribution, i.e., the distribution of states encountered during autoregressive prediction, rather than only the short trajectories observed in the downstream data. Periodic noise injection can further broaden local state-space coverage and teach the student how the teacher evolves from perturbed states. To transfer the teacher's knowledge to the student model, we add these teacher-generated trajectories to the student dataset and train on a mixture of ground truth and teacher-labeled transitions. To overcome the aforementioned lack of inductive biases in foundation models, we propose distilling the teacher into a student with task-appropriate inductive biases. For example, we use a tensorized FNO (Kossaifi et al., 2024) as the student. In addition to being equivariant to input translations, FNOs also exhibit inductive biases such as excluding high-frequency modes that are prone to noise. We investigate the effectiveness of our method through experiments with the Poseidon (Herde et al., 2024) and Walrus (McCabe et al., 2026) foundation models across a range of fluid dynamics tasks, demonstrating that TREX can achieve teacher accuracy while drastically reducing parameter count and inference time. Overall, our work provides the following contributions:

1. To our knowledge, we present the first systematic study of distilling general pretrained PDE foundation models into compact downstream surrogate models.

2. We introduce TREX, a novel distillation method that generates supervision data using noisy teacher rollouts, in some cases surpassing the teacher's accuracy.

3. We show that by distilling the foundation model into a small, lightweight student architecture with suitable inductive biases, we can reduce inference time by more than an order of magnitude while respecting physical properties such as equivariances.

2 Problem Definition

We consider time-dependent PDEs on a spatial domain $\mathcal{X}$ and time interval $[0, T]$, with solution $\mathbf{u} : [0, T] \times \mathcal{X} \to \mathbb{R}^{N_c}$:

$$\partial_t \mathbf{u} = F\left(t, \mathbf{x}, \mathbf{u}, \partial_{\mathbf{x}} \mathbf{u}, \partial_{\mathbf{x}\,\mathbf{x}} \mathbf{u}, \ldots\right), \qquad (t, \mathbf{x}) \in [0, T] \times \mathcal{X}, \tag{1}$$

$$\mathbf{u}(0, \mathbf{x}) = \mathbf{u}^0(\mathbf{x}), \qquad \mathbf{x} \in \mathcal{X}, \tag{2}$$

$$\mathcal{B}[\mathbf{u}](t, \mathbf{x}) = 0, \qquad (t, \mathbf{x}) \in [0, T] \times \partial\mathcal{X}. \tag{3}$$

Here, F defines the dynamics, $\mathbf{u}^0$ is the initial condition, and $\mathcal{B}$ specifies the boundary condition. A task corresponds to solving this initial-value problem for $\mathbf{u}^0 \sim p_0$. For learning, we use discretized solution trajectories $(\mathbf{u}^0, \mathbf{u}^1, \ldots, \mathbf{u}^{N_T-1})$, where each state $\mathbf{u}^i \in \mathbb{R}^{N_x \times N_c}$ contains the solution on N_x spatial grid points and N_c channels. A neural PDE surrogate then learns a discrete time-stepper $\mathcal{G}$ and is rolled out autoregressively $\mathbf{u}^{i+1} \approx \mathcal{G}(\mathbf{u}^i)$. For models that require multiple context frames, we treat the complete context window as the autoregressive state. Hence, each transition shifts this context by one frame and appends the newly predicted state.

3 Related Work

Knowledge distillation (Hinton et al., 2015; Gou et al., 2021) aims to provide the student with additional information from the teacher network. For classification tasks, the class logits can be transferred to the student (Ba & Caruana, 2014; Hinton et al., 2015). For general tasks, the student can be trained to have feature representations close to those of the teacher (Romero et al., 2015). Relational KD (Park et al., 2019) trains the student by comparing the relationship between the student's feature representations of different inputs and the teacher's, i.e., pushing the student to have the same distance between inputs as the teacher. Thus, the teacher and student do not need the same feature space representation. Similarly, Lin et al. (2025) introduce a region-aware attention mechanism that reconstructs student patches into representations more closely aligned with the teacher's view, even when the two models have substantially different architectures. The student can also be trained to match the teacher's derivatives. Srinivas & Fleuret (2018) train a student by matching the teacher's Jacobian matrix. Amin et al. (2025) apply a similar approach to distill a foundation model for molecules into a smaller student model by matching the teacher's energy-prediction Hessian. In addition to providing additional information about the encountered training examples, it is also possible to artificially generate new training instances using the teacher model. For example, Lopes et al. (2017) introduce data-free distillation, which aims to generate new samples that match the statistics of a dataset for which only those statistics are known.

Distillation has also been applied to PDEs or spatiotemporal predictions in general. Li et al. (2025) distill a special teacher architecture with branches for low- and high-frequency features into the student using separate losses for each branch. Wan et al. (2025) propose SINO, which first learns a fine-step surrogate from a small number of trajectories and subsequently distills it into a coarse-step FNO using a larger teacher-generated dataset. The synthetic trajectories are generated from newly sampled initial conditions, whereas TREX targets settings in which no such initial condition sampler is available and instead expands the available downstream trajectories through teacher rollouts. Zhang et al. (2025) present Omnifluids. Omnifluids first trains a high-resolution model on a PDE with varying parameters using a finite-difference loss for a known equation, similar to PINO (Li et al., 2024). The teacher is distilled into a lower-resolution model with the same architecture by matching the teacher's predictions on samples from the IC, assuming the teacher's parameter distribution is known. In contrast, we consider general foundation models pretrained on large datasets, and do not assume knowledge of the PDE or input distributions.

PDE Foundation Models. In recent years, a number of PDE foundation models have been presented. Most use large vision transformers as their backbone (McCabe et al., 2024; Liu et al., 2024; Herde et al., 2024; McCabe et al., 2026). The earliest example of a PDE foundation model for nonlinear

problems is MPP (McCabe et al., 2024), which also uses axial attention to reduce the computational load. Since then, model and dataset sizes have been increasing. Poseidon (Herde et al., 2024) is based on a SwinV2 (Liu et al., 2022a) vision transformer trained to predict the future state with the step size as a model input. Hence, it can be employed both for direct and autoregressive prediction. Walrus (McCabe et al., 2026) is trained on The Well data (Ohana et al., 2024) and can solve 2D and 3D tasks. While using a vision transformer, it also takes multiple time steps as inputs to infer the underlying physics from the observed trajectory segment. Zhou et al. (2025) present Unisolver, which provides the vision transformer backbone with additional information about the PDE, such as the domain geometry or the physical parameters. Most notably, the equation symbols are embedded using an LLM. In contrast, PDEformer-2 (Ye et al., 2025) uses a graph-based transformer, and DPOT (Hao et al., 2024) is based on Fourier attention layers. Shen et al. (2024) propose to use a pretrained LLM backbone to propagate in time. Additionally, they also embed additional inputs using an LLM similar to Unisolver. For more related work, see Appendix D.

4 Knowledge Distillation of PDE Foundation Models

We are given a small training set of trajectories $\mathbf{u}$ from a downstream target domain

$$\mathcal{D}_{\mathrm{GT}} = \{(\mathbf{u}_n^0, \ldots, \mathbf{u}_n^{N_T-1})\}_{n=1}^{N}, \tag{4}$$

where each trajectory consists of N_T states. Our goal is to build an accurate and efficient surrogate model by infusing knowledge from a foundation model (FM) used as the teacher $\mathcal{T}$, which autoregressively predicts the next state $\mathbf{z}^{t+1} = \mathcal{T}(\mathbf{z}^t)$. Since the FM was pretrained on a large set of PDE problems, it can provide a lower error on the downstream task than a model trained from scratch. In the first step, the FM is fine-tuned on the available downstream task data. Afterward, we distill the teacher's knowledge into the student model $\mathcal{S}$. A natural distillation strategy is to sample new initial conditions $\mathbf{u}_0 \sim p_0$, roll out the teacher, and train the student on the resulting teacher-labeled pairs (Zhang et al., 2025). However, in downstream applications, the initial condition distribution p_0 may be unknown, for example, for data coming from experimental sources. In other cases, it may be difficult to sample directly from this distribution. Lastly, for FMs that require multiple context frames, sampling an initial condition alone is insufficient to initialize the teacher, since generating a valid context requires additional simulator calls.

4.1 Knowledge Distillation using Teacher Rollout Extension

We propose *Teacher Rollout Extension* (TREX), a distillation procedure that trains the student not only on the available ground truth downstream trajectories but also on the state distribution induced by the fine-tuned teacher during autoregressive prediction. This distinction is important for PDE surrogates: at test time, the model is repeatedly applied to its own previous predictions, and therefore encounters states that need not lie exactly on the finite set of observed trajectories.

TREX targets this autoregressive state distribution directly. It generates additional training trajectories by rolling out the teacher from the available ground truth states. For simplicity, we state the following occupancy measure for the single-frame case used in our main Poseidon experiments. For multi-frame models, the same construction applies by replacing each state with its corresponding context window. Let $\delta_{\mathbf{u}_n^0}$ be the Dirac measure concentrated at $\mathbf{u}_n^0$, and let

$$\nu_0 = \frac{1}{N} \sum_{n=1}^{N} \delta_{\mathbf{u}_n^0} \tag{5}$$

be the empirical distribution over the available downstream initial states. A standard one-step training objective only supervises the model on transitions drawn from the short observed ground truth trajectories. In contrast, TREX constructs an extended teacher-induced state occupancy measure

$$\rho_{\mathcal{T}}^{K} = \frac{1}{K} \sum_{t=0}^{K-1} (\mathcal{T}^t)_{\#} \nu_0,$$

where $(\mathcal{T}^t)_{\#}\nu_0$ denotes the pushforward of ν_0 under t autoregressive teacher steps and K is the rollout horizon. Hence, the occupancy measure describes the state distribution induced by rolling

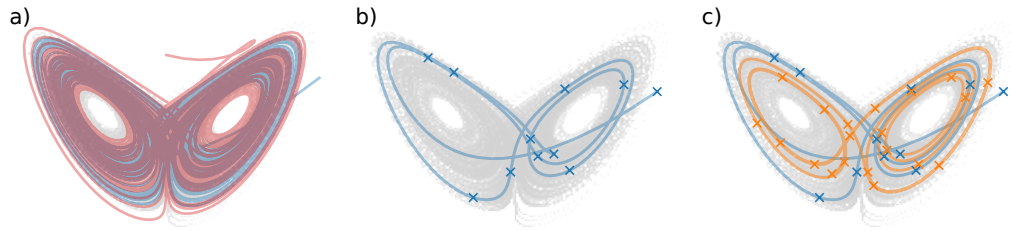

Figure 2: Long rollouts can increase state-space coverage in dynamical systems. Two trajectories of the Lorenz system (Lorenz, 1963) visit similar regions of state space over time (a). Starting from a short observed trajectory (b), extending the rollout generates additional states from the long-time trajectory distribution (c).

out the teacher from the selected set of ground truth samples. This deterministic occupancy measure motivates TREX by describing the additional states made available through teacher rollout extension. In practice, TREX further perturbs this rollout distribution through periodic noise injection, as described below.

Thus, TREX does not require access to the true initial condition distribution. Instead, it reuses the few ground truth trajectories as anchors and expands them through the dynamics learned by the foundation model. This perspective connects TREX to the long-time behavior of dynamical systems. Many dissipative PDEs and chaotic systems possess recurrent long-time structure, such as attractors or invariant measures, so that trajectories initialized from different states eventually visit similar regions of the state space. If the teacher dynamics admit an invariant measure $\mu_{\mathcal{T}}$ satisfying $\mathcal{T}_{\#}\,\mu_{\mathcal{T}} = \mu_{\mathcal{T}}$, then, under additional ergodicity or mixing assumptions, sufficiently long teacher rollouts may approach this long-time distribution. More generally, rollout extension increases coverage of states reachable under the teacher dynamics from the available downstream trajectories. The ergodic Lorenz attractor example in Fig. 2 illustrates this effect: although trajectories start from different initial conditions, their long rollouts repeatedly visit a shared region of the state space. TREX exploits the same principle for PDE foundation models by turning a small number of ground truth trajectories into a much broader set of teacher-labeled states.

4.2 Noisy Teacher Rollouts

Deterministic rollouts sample states on the teacher trajectory, but may provide limited coverage of nearby off-trajectory states. Additionally, it can be beneficial to include data points near typical rollout states in order to teach the student how the dynamics evolve from perturbed inputs. Hence, when appropriate, we inject noise periodically during the teacher rollout:

$$\tilde{\mathbf{z}}^t = \mathbf{z}^t + m_t \boldsymbol{\xi}^t, \qquad \mathbf{z}^{t+1} = \mathcal{T}\left(\tilde{\mathbf{z}}^t\right), \qquad m_t = \mathbf{1}\{t > 0 \ \wedge \ t \equiv 0 \mod T_{\text{Noise}}\}, \tag{6}$$

$$(\boldsymbol{\xi}^t)_c(\mathbf{x}) \sim \mathcal{N}(0, \sigma^2 s_c^2). \tag{7}$$

Here, $\boldsymbol{\xi}^t$ is Gaussian noise applied independently at each spatial location and channel, s_c denotes the global standard deviation of channel c computed over the dataset, and σ is a fixed scaling factor. The noise is not applied at every step, but at intervals of length T_{Noise}, so that we obtain unperturbed sub-trajectories for training and allow the teacher rollout to return toward states typical of its unperturbed rollout distribution after a perturbation. Adding noise during training is a common strategy for autoregressive neural PDE solvers (Sanchez-Gonzalez et al., 2018; Pfaff et al., 2020; Takamoto et al., 2023). In these approaches, the training inputs may be perturbed while the target trajectory remains unchanged. In TREX, by contrast, the teacher is explicitly evaluated after perturbation, so that the synthetic target reflects the teacher dynamics starting from the perturbed state.

By performing noisy rollouts from the initial conditions of the available ground truth trajectories, we obtain the dataset $\mathcal{D}_{\text{TREX}}$. It consists of teacher-generated sub-trajectories between consecutive noise injections. A segment starting at a noise-injection step t_i is of the form

$$\left(\tilde{\mathbf{z}}^{t_i}, \mathbf{z}^{t_i+1}, \ldots, \mathbf{z}^{t_i+T_{\text{Noise}}}\right).$$

These segments are subsequently split into sub-trajectories of the desired student rollout length.

The final student objective combines ground truth supervision with the teacher rollout distillation:

$$\mathcal{L}_{\mathcal{S}} = \mathcal{L}_{\mathrm{GT}} + \lambda_{\mathrm{TREX}} \mathcal{L}_{\mathrm{TREX}}.$$

For both ground truth and teacher-generated sub-trajectories, we train the student using a k-step autoregressive rollout loss. For TREX data,

$$\mathcal{L}_{\mathrm{TREX}} = \frac{1}{|\mathcal{D}_{\mathrm{TREX}}|} \sum_{(\tilde{\mathbf{z}}^i, \ldots, \mathbf{z}^{i+k}) \in \mathcal{D}_{\mathrm{TREX}}} \frac{1}{k} \sum_{t=i}^{i+k-1} \ell\left(\mathbf{z}^{t+1}, \mathcal{S}(\hat{\mathbf{u}}^t)\right), \quad \hat{\mathbf{u}}^i = \tilde{\mathbf{z}}^i, \quad \hat{\mathbf{u}}^{t+1} = \mathcal{S}(\hat{\mathbf{u}}^t).$$

The training sub-trajectories $\left(\mathbf{z}^i, \ldots, \mathbf{z}^{i+k}\right)$ are obtained by splitting the noise-free teacher-generated segments into sequences of the desired rollout length. $\mathcal{L}_{\mathrm{GT}}$ is defined analogously on sub-trajectories extracted from the available ground truth trajectories. As TREX merely provides additional data points, it could be combined with other methods, such as feature-based distillation techniques, in the future.

4.3 Enforcing Physical Constraints

Foundation models for PDEs are usually constructed using large transformer architectures (Sec. 3). This makes them flexible enough to model many different systems, but it also means that they often do not explicitly encode the symmetries of a specific downstream PDE. Once the downstream task is fixed, however, such a structure may be known and can be built into the student. We focus on equivariance as one such constraint. Let G be a group of transformations acting on the state space, and let $\tau_g u$ denote the transformed state for $g \in G$. A model $\mathcal{M}$ is equivariant to G if

$$\mathcal{M}(\tau_g u) = \tau_g \mathcal{M}(u), \qquad \forall g \in G.$$

For a PDE surrogate, this means that transforming the input state and then predicting the next state gives the same result as predicting first and then transforming the output. Many physical systems satisfy such symmetries. For example, on a periodic domain, the solution operator of a translation-invariant PDE satisfies $\Phi_{\Delta t}(\tau_a u) = \tau_a \Phi_{\Delta t}(u)$, where τ_a denotes a spatial shift. This provides a complementary benefit to distillation. The teacher contributes knowledge acquired during large-scale pretraining, while the student can impose task-specific structure known for the PDE. In low-data settings, this is particularly useful, as the student does not need to infer symmetry solely from the few available trajectories. The student can therefore learn from the teacher-generated data while restricting the learned transition map to functions that better match the downstream physics. Architectural constraints can also act as regularizers. For example, a student's architecture may suppress unresolved high-frequency components, enforce locality, preserve conservation laws, or respect boundary conditions. Such constraints can reduce noisy artifacts and improve rollout stability.

5 Experiments

For our main experiments, we use the Poseidon-L model (Herde et al., 2024) as the teacher. For simplicity, we always use the autoregressive evaluation of Poseidon (instead of the lead-time prediction mode). The teacher is distilled into a lightweight TFNO model (Kossaifi et al., 2024, 2025), which represents FNO's (Li et al., 2021) weight matrix using a Tucker factorization to reduce the number of parameters but also improve generalization through regularization effects. While the teacher is trained with the original n-to-n training protocol, the student is trained autoregressively with 2-step sub-trajectories. More model and training details are listed in Appendix F. For TREX, the teacher is rolled out for 100 steps starting from the initial condition, where noise is injected after every 10 steps. The relative noise standard deviation is set to $\sigma = 1$. As distillation baselines, we compare the original Poseidon-L teacher and the student trained only with the available ground truth data. Additionally, we add relational KD (Park et al., 2019) as a feature-based KD method, using Gaussian sketching to reduce the dimensionality to make the pairwise distances computationally feasible. Lastly, we add a baseline with full knowledge of the true initial condition distribution (IC-KD) similar to Zhang et al. (2025). We use a total batch size of 32 and adjust the number of epochs so that the total number of batch updates is equal between all methods (25k when using only ground truth). The training times are listed in the appendix (Tab. 7). TREX training is only slightly slower than pure ground truth training (e.g., increased training time from 272 to 291 min for 32 trajectories). For more details on the KD methods, see Appendix G.

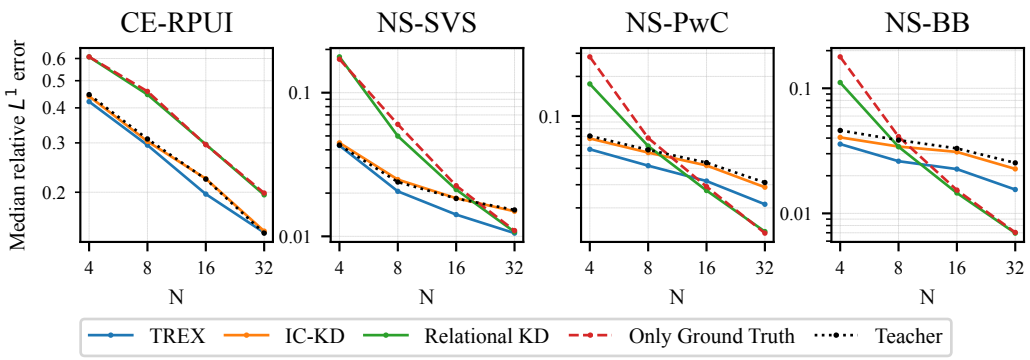

Figure 3: The median relative L^1 error for Poseidon downstream experiments. The distillation methods are compared to the teacher's accuracy over the number of trajectories in the training set N. TREX matches or surpasses the teacher across the selected downstream settings.

Table 1: Inference time, GPU memory, and number of parameters for the Poseidon-L teacher and the TFNO student. Inference and memory are computed for a batch size of 64.

Model	1-step		Full rollout		# Param.
	Time	Memory	Time	Memory	
Poseidon-L (lead-time)	270.9 ms	3569 MB	1908.2 ms	10325 MB	628.6M
Poseidon-L (autoreg.)	269.4 ms	3605 MB	1887.7 ms	3749 MB	628.6M
TFNO (student)	22.1 ms	1059 MB	152.1 ms	1171 MB	174.4K

As downstream tasks, we use compressible Euler and incompressible Navier-Stokes equations from Herde et al. (2024). Appendix E lists the used PDEs in detail. Since our goal is specifically to study whether useful knowledge from a foundation model can be transferred into a compact student, we focus on spatiotemporal Poseidon tasks for which the fine-tuned teacher outperforms the TFNO trained only on the available downstream data, as determined on the validation set. The trajectories contain 8 frames each. We report the original median relative L^1 error from Poseidon (Herde et al., 2024), which normalizes the L^1 loss for each quantity of interest (QOI) frame-wise:

$$L^1_{\text{QOI}}(\mathbf{u}, \hat{\mathbf{u}}) = \frac{\sum_{\mathbf{x}} \sum_{c_{\text{QOI}}} |\hat{\mathbf{u}}(t_{\text{last}}, \mathbf{x}, c_{\text{QOI}}) - \mathbf{u}(t_{\text{last}}, \mathbf{x}, c_{\text{QOI}})|}{\sum_{\mathbf{x}} \sum_{c_{\text{QOI}}} |\mathbf{u}(t_{\text{last}}, \mathbf{x}, c_{\text{QOI}})|}.$$

For completeness, we also provide the results over all time steps in Tab. 6. For the incompressible Navier-Stokes tasks, the QOI is the velocity vector. For compressible Euler (CE-RPUI), the mean of the median of all QOI errors is reported. We report the mean across three seeds for all main results, using different training trajectories for each seed as well.

5.1 Main Results

Fig. 3 shows the main results for the Poseidon distillation. TREX matches or surpasses its teacher network across the selected low-data settings, which we will further investigate in the next section. IC-KD is also able to match the teacher on all investigated settings; however, it requires the full IC distribution to be known, which is not the case for experimental settings and might also be expensive (for example, for NS-BB, the IC is actually an already evolved state (Appendix E)). While relational KD can also slightly improve the student, we were not able to match the teacher in settings where the teacher outperformed ground truth learning. A reason might be the difficulty of meaningful distance measures in the large-dimensional feature fields encountered in the spatial neural PDE solver latent space. Interestingly, we found that the TFNO is an exceptionally well-performing model for the considered tasks, even capable of outperforming the pretrained Poseidon in certain settings without KD. In these cases, TREX can perform worse than only using the ground truth since the added teacher-generated data is moving the student away from the better solution it would have found using only ground truth. For the other student models considered below, we did not find similar results. Thus, distillation should only be applied if the foundation model was found to generalize well to

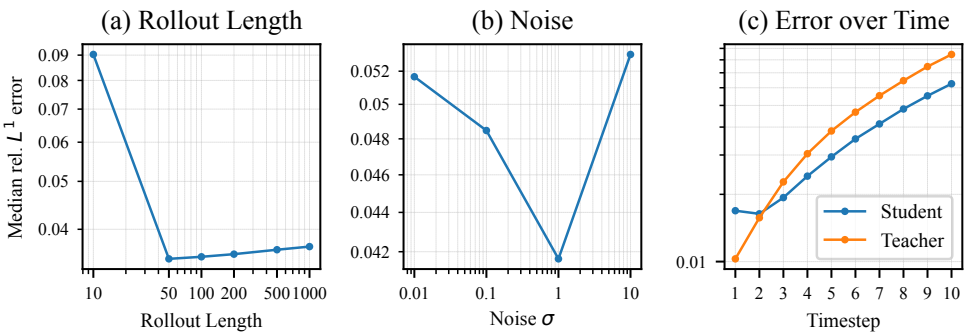

Figure 4: Ablation study on NS-SVS (four trajectories). The effect of the rollout is relatively constant over a wide range of hyperparameters (a). Injecting noise also reduces the error, particularly at relatively high noise amplitudes (b). In (c), the error over time is shown for NS-PwC (8 trajectories). Time steps $t > 7$ show temporal extrapolation beyond the training time horizon. While the TREX-trained student has a higher one-step error than the teacher, the student has a more stable rollout.

the task at hand using a held-out validation set. Fig. 4c) shows the error growth over time for the NS-PwC task, comparing the TFNO student with Poseidon. While the student has a larger error at the beginning, it has a more stable long-term rollout and has a slower error growth. Interestingly, this behavior is seen across multiple datasets (Fig. 9). While surpassing the teacher's accuracy, the student needs 3604 times fewer parameters (Tab. 1). The inference time is reduced by over an order of magnitude, and the memory consumption for a forward pass by a factor of 3.2 (compared to autoregressive Poseidon rollouts).

5.2 Ablations

To investigate why TREX can generate a student that outperforms its teacher, we conduct an ablation study. To this end, we select the NS-PwC-4 case and fully remove the ground truth trajectories, training the student only on teacher-generated data (Tab. 2). Even without the ground truth data, TREX still outperforms its teacher. Removing noise injections during the rollout reduces the gap, but not fully. The phenomenon that distillation can improve student performance has been observed in the general KD literature (Mobahi et al., 2020; Park et al., 2019). It can be explained by regularization effects, which can even occur when distilling the same model into the same architecture (Zhang et al., 2019; Mobahi et al., 2020).

Table 2: Ablation study on NS-PwC (four trajectories). Shown are the effects of removing the noise injection during the teacher rollout, as well as removing ground truth trajectories fully from the student training.

Variant	Median rel. L^1
TREX	0.056 ± 0.003
w/o GT	0.060 ± 0.002
w/o GT & w/o Noise	0.064 ± 0.006
Teacher	0.070 ± 0.011

Next to the core algorithm choices, we also investigate the influence of the included hyperparameters. Fig. 4a) shows the effect of the rollout length. The algorithm is quite robust to the choice of this hyperparameter. The performance only decreases slowly for very long rollouts. The effect of the strength of the injected noise is shown in Fig. 4b). Interestingly, choosing the noise quite high at a level of one time the channel-wise standard deviation s_c produces the best results. Fig. 8 shows an example of an extended noisy rollout (appendix). While the state appears very noisy after the noise injection, the teacher rollout effectively reduces the noise level in subsequent states. We also provide an experiment with other student architectures. We use a modern U-Net (Gupta & Brandstetter, 2023) and a standard FNO (Li et al., 2021). As shown in Fig. 5, TREX is also able to let other student architectures match the teacher's accuracy for all considered numbers of trajectories. While the TFNO was able to improve over the teacher using only ground truth for a higher number of trajectories in the main experiments (Fig. 3), this was not the case for the U-Net or even a standard FNO, and was only possible through distillation.

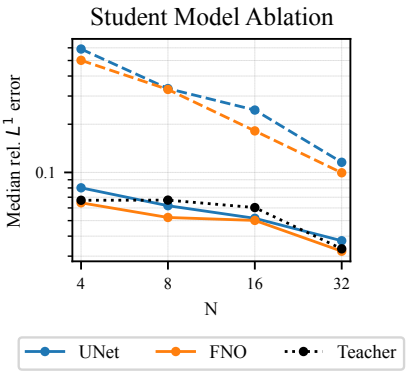

Figure 5: Ablation on the student architecture (NS-PwC). Shown are the results for a U-Net and FNO trained with TREX (full) and only ground truth data (dashed).

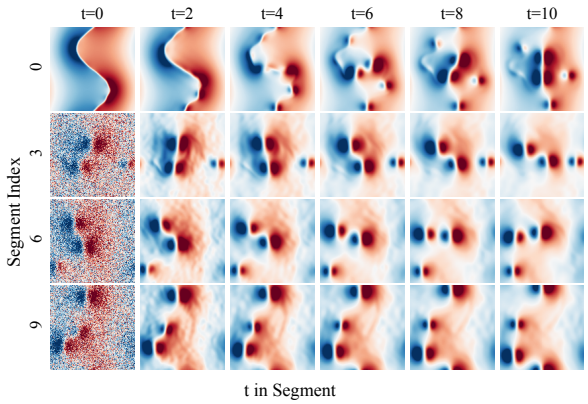

Figure 6: Example TREX rollout of the Poseidon teacher on the NS-SVS task (v_x channel). Each row represents a segment of the complete rollout, starting with the state to which the noise was applied.

Table 3: Distillation of Walrus (McCabe et al., 2026) on the Kolmogorov Flow dataset. TREX improves long-rollout accuracy while reducing inference time by more than an order of magnitude.

	$\text{VRMSE}_{\text{One-Step}}$	$\text{VRMSE}_{T\in[1:20]}$	$\text{VRMSE}_{T\in[21:60]}$	Inference Time in s
Walrus	**0.012**	0.214	0.713	21.425
Only GT	0.050	0.355	0.972	0.401
TREX	0.016	**0.123**	**0.686**	0.414

5.2.1 Teacher Model

We also experiment with another teacher model. Walrus (McCabe et al., 2026) is a larger vision transformer (1.3B parameters) that uses the last six states as context for next state prediction. This setup is crucially different since it does not allow IC-KD, as we would need to run the simulator to get the initial sequence to start the rollout. To also go beyond the rather short trajectories provided by the Poseidon datasets, we test distillation on the 2D Kolmogorov Flow dataset from Li et al. (2022) with a Reynolds number of 5000. The trajectories have 501 time steps, where the first 100 are discarded (Li et al., 2022). For the student, we select a higher-capacity TFNO with 64 modes and a feature dimension of 80, trained for only 10k epochs with a batch size of 64. Like the teacher, the student uses six input steps as context. For the ground-truth-only baseline, we found a single step as input to work better. We fine-tune Walrus on a single trajectory and report the VRMSE metric (App. Eq. (30)) for different rollout intervals. The student trained with TREX can also successfully distill the multi-step teacher foundation model (Tab. 3).

5.2.2 Equivariance

To investigate the effect of the explicit equivariance built into the student architecture, we test how the models behave under spatial shifts on the periodic grid. We apply random 2D rolls to the input fields, which corresponds to the translation symmetry of the underlying periodic domain. An equivariant model should produce the same prediction up to this same roll. We evaluate this in two ways: first, as a direct equivariance error, and second, through shifted autoregressive rollouts.

For the direct equivariance metric, let $a = (d_x, d_y)$ denote a sampled 2D periodic translation, and let τ_a be the corresponding shift operator. We compare the prediction on the shifted input with the shifted prediction on the original input, $\|f(\tau_a x) - \tau_a f(x)\|_1$, using the same median relative L^1 normalization as in the main evaluation. As shown in Tab. 4, the student has substantially lower equivariance error than the teacher across all datasets. This is expected from the student architecture, but it is still important: distillation transfers the teacher's predictive behavior into a model class that enforces a symmetry not explicitly respected by the teacher.

Table 4: Translation equivariance results. Left: median relative L^1 equivariance error, computed using the unperturbed predictions as ground truth. Right: final-step relative L^1 errors on original and shifted OOD rollouts.

	Equivariance error		Final-step rollout error			
Dataset	Teacher	Student	Poseidon	Poseidon shift	TFNO	TFNO shift
NS-BB	0.026 ± 0.013	0.000 ± 0.001	0.0651 ± 0.0286	0.0676 ± 0.0203	0.0504 ± 0.0080	0.0504 ± 0.0080
CE-RPUI	0.101 ± 0.006	0.001 ± 0.001	0.4189 ± 0.0767	0.4108 ± 0.0856	0.3549 ± 0.0344	0.3549 ± 0.0344
NS-PwC	0.023 ± 0.006	0.000 ± 0.001	0.1095 ± 0.0420	0.1041 ± 0.0385	0.0744 ± 0.0206	0.0744 ± 0.0206
NS-SVS	0.110 ± 0.005	0.000 ± 0.001	0.0820 ± 0.0083	0.2398 ± 0.0086	0.0755 ± 0.0072	0.0755 ± 0.0072

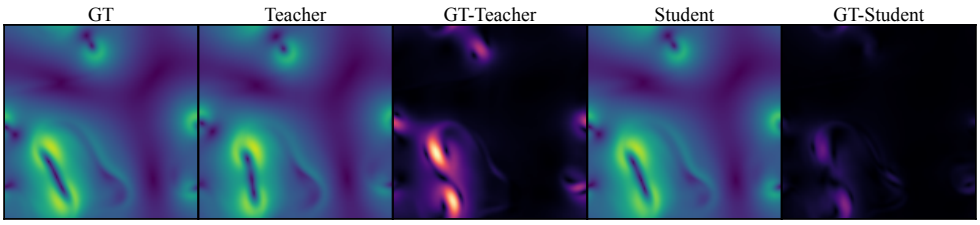

Figure 7: Final-step shifted-rollout comparison on NS-SVS. The teacher shows a visibly larger error with respect to the shifted ground truth (GT), while the student prediction remains much closer and the corresponding difference is substantially smaller.

We further evaluate the effect of equivariance in autoregressive rollouts for temporal extrapolation by applying the same spatial shift to the initial condition and comparing the final prediction to the shifted ground truth. The results in Tab. 4 show that the shifted rollout error of the student remains essentially unchanged, while the teacher can degrade substantially, most clearly on NS-SVS. Since NS-SVS also has the largest teacher equivariance error, we show a qualitative example in Fig. 7. In this case, the teacher produces visibly worse shifted rollouts, while the student remains consistent with the translated dynamics. We include the entire rollout trajectory in Appendix Fig. 10. These results suggest that equivariance can be a useful property of the student, especially in autoregressive simulations where small symmetry errors may accumulate over long rollouts. More broadly, it is encouraging that TREX can use a non-equivariant teacher to train a compact student that preserves the teacher's predictive behavior while enforcing the symmetry of the downstream task.

6 Conclusion

We introduce TREX, a simple and effective knowledge distillation framework for PDE foundation models. By extending limited downstream data through long and noisy teacher rollouts, TREX enables the student to match the teacher's long-horizon dynamics, thereby providing valuable additional information. Our results demonstrate that distillation in the foundation model regime is not only feasible but highly effective: compact student models can match the accuracy of large teachers while achieving substantial improvements in inference speed and memory efficiency. Additionally, the student can incorporate inductive biases to faithfully reproduce prior knowledge about the system, such as equivariances. Overall, TREX bridges the gap between the strong generalization capabilities of large foundation models and the efficiency requirements of practical surrogate modeling. In the future, TREX could be combined with other distillation techniques to also provide more information on the influence of other factors such as physical parameters to the student.

Acknowledgments and Disclosure of Funding

Funded by the German Federal Ministry of Research, Technology and Space (BMFTR) as part of the InnoPhase project (funding code: 02NUK078). Additionally, we acknowledge the support of Deutsche Forschungsgemeinschaft (DFG) under funding code 569019417 and the support of the Stuttgart Center for Simulation Science (SimTech). Andrei Manolache acknowledges funding by

the EU Horizon project ELIAS (No. 101120237). The authors thank the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for supporting Daniel Musekamp, Andrei Manolache, and Mathias Niepert.

References

Ishan Amin, Sanjeev Raja, and Aditi S Krishnapriyan. Towards fast, specialized machine learning force fields: Distilling foundation models via energy hessians. In *The Thirteenth International Conference on Learning Representations*, 2025.

Lei J Ba and Rich Caruana. Do deep nets really need to be deep? *Advances in neural information processing systems*, 27, 2014.

X Cheng, Y He, Y Yang, X Xue, S Cheng, D Giles, X Tang, and Y Hu. Learning chaos in a linear way. In *International Conference on Learning Representations*, pp. 55276–55318. ICLR, 2025.

Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. Knowledge distillation: A survey. *International journal of computer vision*, 129(6):1789–1819, 2021.

Jayesh K. Gupta and Johannes Brandstetter. Towards multi-spatiotemporal-scale generalized PDE modeling. *Trans. Mach. Learn. Res.*, 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. In *Proceedings of the 41st International Conference on Machine Learning*, pp. 17616–17635, 2024.

Jacob Helwig, Xuan Zhang, Cong Fu, Jerry Kurtin, Stephan Wojtowytsch, and Shuiwang Ji. Group equivariant fourier neural operators for partial differential equations. In *International Conference on Machine Learning*, pp. 12907–12930. PMLR, 2023.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

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.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

David Holzmüller, Viktor Zaverkin, Johannes Kästner, and Ingo Steinwart. A framework and benchmark for deep batch active learning for regression. *Journal of Machine Learning Research*, 24(164):1–81, 2023.

Nick Janssens and Johan Meyers. Towards real-time optimal control of wind farms using large-eddy simulations. *Wind Energy Science*, 9(1):65–95, 2024.

William B Johnson and Joram Lindenstrauss. Extensions of lipschitz mappings into a hilbert space. *Contemporary mathematics*, 26(189-206):1, 1984.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun (eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

Jean Kossaifi, Nikola Borislavov Kovachki, Kamyar Azizzadenesheli, and Anima Anandkumar. Multi-grid tensorized fourier neural operator for high- resolution pdes. *Trans. Mach. Learn. Res.*, 2024.

Jean Kossaifi, Nikola Kovachki, Zongyi Li, David Pitt, Miguel Liu-Schiaffini, Robert Joseph George, Boris Bonev, Kamyar Azizzadenesheli, Julius Berner, Valentin Duruisseaux, and Anima Anandkumar. A library for learning neural operators. *arXiv preprint arXiv:2412.10354*, 2025.

Yuqi Li, Chuanguang Yang, Hansheng Zeng, Zeyu Dong, Zhulin An, Yongjun Xu, Yingli Tian, and Hao Wu. Frequency-aligned knowledge distillation for lightweight spatiotemporal forecasting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 7262–7272, 2025.

Zongyi Li, Nikola Borislavov Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew M. Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*, 2021.

Zongyi Li, Miguel Liu-Schiaffini, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Learning chaotic dynamics in dissipative systems. *Advances in Neural Information Processing Systems*, 35:16768–16781, 2022.

Zongyi Li, Hongkai Zheng, Nikola Kovachki, David Jin, Haoxuan Chen, Burigede Liu, Kamyar Azizzadenesheli, and Anima Anandkumar. Physics-informed neural operator for learning partial differential equations. *ACM/IMS Journal of Data Science*, 1(3):1–27, 2024.

Jhe-Hao Lin, Yi Yao, Chan-Feng Hsu, Hong-Xia Xie, Hong-Han Shuai, and Wen-Huang Cheng. Perspective-aware teaching: Adapting knowledge for heterogeneous distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4178–4187, 2025.

Yuxuan Liu, Zecheng Zhang, and Hayden Schaeffer. PROSE: Predicting multiple operators and symbolic expressions using multimodal transformers. *Neural Networks*, 180:106707, 2024.

Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, et al. Swin transformer v2: Scaling up capacity and resolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 12009–12019, 2022a.

Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*. IEEE, 2022b.

Raphael Gontijo Lopes, Stefano Fenu, and Thad Starner. Data-free knowledge distillation for deep neural networks. *arXiv preprint arXiv:1710.07535*, 2017.

EN Lorenz. Deterministic nonperiodic flow. *J. Atmos. Sci.*, 20:130–141, 1963.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, 2019.

Michael McCabe, Bruno Régaldo-Saint Blancard, Liam Parker, Ruben Ohana, Miles Cranmer, Alberto Bietti, Michael Eickenberg, Siavash Golkar, Geraud Krawezik, Francois Lanusse, Mariel Pettee, Tiberiu Tesileanu, Kyunghyun Cho, and Shirley Ho. Multiple physics pretraining for spatiotemporal surrogate models. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Michael McCabe, Payel Mukhopadhyay, Tanya Marwah, Bruno Regaldo-Saint Blancard, Francois Rozet, Cristiana Diaconu, Lucas Meyer, Kaze WK Wong, Hadi Sotoudeh, Alberto Bietti, et al. Walrus: A cross-domain foundation model for continuum dynamics. *Forty-third International Conference on Machine Learning*, 2026.

Hossein Mobahi, Mehrdad Farajtabar, and Peter Bartlett. Self-distillation amplifies regularization in hilbert space. *Advances in Neural Information Processing Systems*, 33:3351–3361, 2020.

Ruben Ohana, Michael McCabe, Lucas Meyer, Rudy Morel, Fruzsina J Agocs, Miguel Beneitez, Marsha Berger, Blakesley Burkhart, Stuart B Dalziel, Drummond B Fielding, et al. The well: a large-scale collection of diverse physics simulations for machine learning. *Advances in Neural Information Processing Systems*, 37:44989–45037, 2024.

Wonpyo Park, Dongju Kim, Yan Lu, and Minsu Cho. Relational knowledge distillation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 3967–3976, 2019.

Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter Battaglia. Learning mesh-based simulation with graph networks. In *International conference on learning representations*, 2020.

Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. In Yoshua Bengio and Yann LeCun (eds.), *International Conference on Learning Representations*, 2015.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference*, pp. 234–241. Springer, 2015.

Alvaro Sanchez-Gonzalez, Nicolas Heess, Jost Tobias Springenberg, Josh Merel, Martin Riedmiller, Raia Hadsell, and Peter Battaglia. Graph networks as learnable physics engines for inference and control. In *International conference on machine learning*, pp. 4470–4479. PMLR, 2018.

Yair Schiff, Zhong Yi Wan, Jeffrey B Parker, Stephan Hoyer, Volodymyr Kuleshov, Fei Sha, and Leonardo Zepeda-Núñez. Dyslim: Dynamics stable learning by invariant measure for chaotic systems. In *Forty-first International Conference on Machine Learning*, 2024.

Junhong Shen, Tanya Marwah, and Ameet Talwalkar. UPS: efficiently building foundation models for PDE solving via cross-modal adaptation. *Trans. Mach. Learn. Res.*, 2024.

Suraj Srinivas and François Fleuret. Knowledge transfer with jacobian matching. In *International conference on machine learning*, pp. 4723–4731. PMLR, 2018.

Makoto Takamoto, Francesco Alesiani, and Mathias Niepert. Learning neural pde solvers with parameter-guided channel attention. In *International Conference on Machine Learning*, pp. 33448–33467. PMLR, 2023.

Mingxuan Tian, Haochen Mu, Tao Liu, Mengjiao Li, Donghong Ding, and Jianping Zhao. Physics-informed machine learning-based real-time long-horizon temperature fields prediction in metallic additive manufacturing. *Communications Engineering*, 4(1):168, 2025.

Han Wan, Rui Zhang, and Hao Sun. Spectral-inspired neural operator for data-efficient pde simulation in physics-agnostic regimes. *arXiv preprint arXiv:2505.21573*, 2025.

Zhanhong Ye, Zining Liu, Bingyang Wu, Hongjie Jiang, Leheng Chen, Minyan Zhang, Xiang Huang, Qinghe Meng Zou, Hongsheng Liu, Bin Dong, et al. Pdeformer-2: A versatile foundation model for two-dimensional partial differential equations. *arXiv preprint arXiv:2507.15409*, 2025.

Linfeng Zhang, Jiebo Song, Anni Gao, Jingwei Chen, Chenglong Bao, and Kaisheng Ma. Be your own teacher: Improve the performance of convolutional neural networks via self distillation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 3713–3722, 2019.

Rui Zhang, Qi Meng, Han Wan, Yang Liu, Zhi-Ming Ma, and Hao Sun. Omnifluids: Unified physics pre-trained modeling of fluid dynamics. *arXiv preprint arXiv:2506.10862*, 2025.

Maksim Zhdanov, David Ruhe, Maurice Weiler, Ana Lucic, Johannes Brandstetter, and Patrick Forré. Clifford-steerable convolutional neural networks. In *International Conference on Machine Learning*, pp. 61203–61228. PMLR, 2024.

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

A Limitations

While TREX provides an effective framework for distilling PDE foundation models, it comes with several limitations. First, TREX relies on the quality of the fine-tuned teacher. Since the method samples from the teacher-induced rollout distribution, any systematic bias or instability in the teacher's long-horizon dynamics may be transferred to the student. In particular, if the teacher converges to an incorrect or distorted attractor, TREX may reinforce this behavior. Although mixing teacher-generated data with ground truth trajectories mitigates this issue, it does not fully eliminate it. Second, rollout extension is most useful when long trajectories continue to explore informative regions of state space. Its benefit may therefore be limited for systems that rapidly collapse to a trivial fixed point, exhibit strongly transient dynamics, or contain multiple disconnected long-time regimes that are not represented by the available trajectories. Moreover, the Gaussian perturbations used by TREX are not guaranteed to satisfy all physical constraints of the underlying PDE and should therefore be viewed as a state-space augmentation strategy rather than as physically valid perturbations.

B Broader Impact

This work aims to advance machine learning methods for scientific simulation and time-dependent PDEs by making foundation-model-based surrogates more efficient and easier to deploy. Faster surrogate models could reduce the computational cost of simulation workflows and make such tools more accessible in scientific and engineering applications. We do not expect broad negative societal impacts from the method itself.

C Compute Resources

The experiments were performed on a cluster with 10 NVIDIA GeForce RTX 4090 GPUs and on a larger compute cluster with 32 NVIDIA A100 GPUs and 24 NVIDIA A40 GPUs. Each individual training run used a single GPU. The reported inference-time and memory measurements for Poseidon experiments were collected for a batch size of 64, as described in the main text.

D Continued Related Work

Several authors have investigated explicitly including attractor dynamics for neural PDE solvers. Li et al. (2022) include a loss term to enforce that the model always converges to the attractor on artificial data drawn uniformly from a fixed ball around zero. DySLIM (Schiff et al., 2024) enforces that the learned model follows the invariant measure by minimizing the maximum mean discrepancy between the data distribution and the learned attractor. Cheng et al. (2025) proposes a Poincaré flow neural network, which is trained by dividing the data into the contracting and measure-invariant phases that can occur in a dissipative chaotic system. Depending on the phase, a loss is applied to enforce the contracting or invariant behavior.

E Datasets

We mainly use the datasets provided by Herde et al. (2024). All datasets, besides Kolmogorov Flow, contain 8 time steps, which end at $T = 0.7$ and use periodic boundary conditions. The spatial resolution is 128x128 for a grid of $[0, 1] \times [0, 1]$. The Poseidon problems have 120 validation and 240 test trajectories.

E.1 NS-PwC

The Navier-Stokes piecewise constants task uses the incompressible Navier-Stokes equations:

$$\partial_t \mathbf{u} + (\mathbf{u} \cdot \nabla)\mathbf{u} + \nabla p = \nu \Delta \mathbf{u}, \nabla \cdot \mathbf{u} = 0. \tag{8}$$

For NS-PwC, the initial state is given as a set of constants on a grid of 10 cells in each dimension:

$$\omega_0(x, y) = c_{i,j} \quad \text{for } (x, y) \in [x_{i-1}, x_i] \times [y_{j-1}, y_j], \qquad i, j = 1, \ldots, p. \tag{9}$$

with $x_i = y_i = \frac{i}{p}$, for $i = 0, 1, 2, \ldots, p$ and the constant square values drawn from $c_{i,j} \sim \mathcal{U}_{[-1,1]}$. Here, ω_0 refers to the vorticity that can be transformed into the velocity in the x and y directions using the incompressibility condition.

E.2 NS-BB

In Fourier space, the Brownian bridge initial conditions are initialized as

$$W(x) = \sum_{|k|_\infty \leq N} \frac{1}{\|k\|_2^{3/2}} \sum_{m,n,\ell \in \{0,1\}} \alpha_k^{(mn\ell)} \, \mathrm{sc}_m(x) \mathrm{sc}_n(x) \mathrm{sc}_\ell(x), \tag{10}$$

where

$$\mathrm{sc}_i(x) = \begin{cases} \sin(x), & \text{for } i = 0, \\ \cos(x), & \text{for } i = 1 \end{cases}. \tag{11}$$

W(x) does not directly give u_0. Instead, it defines the state at t=-0.5, and the initial conditions are derived by evolving the Navier-Stokes equations until t=0. The random variables are sampled from $\alpha_k^{(mn\ell)} \sim \mathcal{U}_{[-1,1]}$.

E.3 NS-SVS

The next Navier-Stokes task uses Sinusoidal Vortex Sheet (SVS) initial conditions, which also use a vorticity formulation

$$\omega_0^\rho = \phi_\rho * \omega_0, \tag{12}$$

where ω_0 is defined as

$$\omega_0(x) = \delta(x - \Gamma) - \int_{\mathbb{T}^2} d\Gamma, \tag{13}$$

and

$$\phi_\rho(x) = \rho^{-2} \, \psi\left(\frac{\|x\|}{\rho}\right). \tag{14}$$

$\psi(r)$ is defined as

$$\psi(r) = \frac{80}{7\pi} \left[(r+1)_+^3 - 4\left(r + \frac{1}{2}\right)_+^3 + 6r_+^3 - 4\left(r - \frac{1}{2}\right)_+^3 + (r-1)_+^3 \right], \tag{15}$$

and Γ as

$$\Gamma = \left\{ (x, y) \in \mathbb{T}^2 \;\middle|\; y = \frac{1}{2} + 0.2 \sin(2\pi x) + \sum_{i=1}^{p} \alpha_i \sin\left(2\pi(x + \beta_i)\right) \right\}. \tag{16}$$

The variable parameters are drawn from

$$\alpha_i \sim \mathcal{U}_{[0,\, 0.003125]}, \quad \beta_i \sim \mathcal{U}_{[0,\, 1]}, \tag{17}$$

and ρ and p are fixed to $\rho = \frac{5}{128}$ and $p = 10$.

E.4 CE-RPUI

The Compressible Euler PDE is given as

$$u_t + \nabla \cdot F(u) = 0, \tag{18}$$

where u and $F(u)$ are defined as

$$u = \begin{bmatrix} \rho \\ \rho \mathbf{v} \\ E \end{bmatrix}, \quad F(u) = \begin{bmatrix} \rho \mathbf{v} \\ \rho \mathbf{v} \otimes \mathbf{v} + p\mathbf{I} \\ (E + p)\mathbf{v} \end{bmatrix}. \tag{19}$$

For the CE-RPUI problem, the four channels are drawn as constant values for each region of the input grid. The input division sets $D_{i,j}$ are computed using sinusoidal functions $\sigma_x(x, y)$ and $\sigma_y(x, y)$, which are defined as

$$\sigma_x(x,y) = \sum_{i,j=1}^{p} \alpha_{x,i,j} \sin\Big(2\pi\big(i+2p^2\big)x + \big(j+2p^2\big)y + \beta_{x,i,j}\Big), \tag{20}$$

and similarly

$$\sigma_y(x,y) = \sum_{i,j=1}^{p} \alpha_{y,i,j} \sin\Big(2\pi\big(i+2p^2\big)x + \big(j+2p^2\big)y + \beta_{y,i,j}\Big). \tag{21}$$

The coefficients $\alpha_{k,i,j}$ $\beta_{k,i,j}$ are drawn from uniform distributions:

$$\alpha_{k,i,j} \sim \mathcal{U}[-0.01, 0.01], \qquad \beta_{k,i,j} \sim \mathcal{U}[0,1]. \tag{22}$$

Afterward, the input space regions are defined as

$$D_{i,j} = \left\{ (x,y) \in \mathbb{T}^2 \;\middle|\; \begin{array}{l} x_{\min} \le \{x + \sigma_x(x,y) + 1\} < x_{\max}, \\ y_{\min} \le \{y + \sigma_y(x,y) + 1\} < y_{\max} \end{array} \right\}, \tag{23}$$

where $\{x\} := x - \lfloor |x| \rfloor \operatorname{sgn}(x)$ and

$$x_{\min} = \frac{i}{p+1}, \quad x_{\max} = \frac{i+1}{p+1}, \quad y_{\min} = \frac{j}{p+1}, \quad y_{\max} = \frac{j+1}{p+1}. \tag{24}$$

In the end, the initial condition is defined using the regions $D_{i,j}$

$$(\rho, v_x, v_y, p)\big|_{t=0} = (\rho_{i,j}, v^x_{i,j}, v^y_{i,j}, p_{i,j}) \quad \text{in } D_{i,j}, \tag{25}$$

where the region-wise parameters are drawn from

$$\rho_{i,j} \sim \mathcal{U}[1,3], \quad v^x_{i,j} \sim \mathcal{U}[-10,10], \quad v^y_{i,j} \sim \mathcal{U}[-10,10], \quad p_{i,j} \sim \mathcal{U}[5,7]. \tag{26}$$

E.5 Kolmogorov Flow

For the ablations, we use the Kolmogorov Flow dataset (Li et al., 2022) at a Reynolds number (Re) of 5000. The PDE can be described as

$$\frac{\partial u}{\partial t} = -u \cdot \nabla u - \nabla p + \frac{1}{Re}\Delta u + \sin(4y)\hat{x}, \qquad (x,y,t) \in [0,2\pi] \times [0,2\pi] \times [0,\infty) \tag{27}$$

$$\nabla \cdot u = 0, \qquad (x,y,t) \in [0,2\pi] \times [0,2\pi] \times [0,\infty) \tag{28}$$

$$u(\cdot, 0) = u_0, \qquad (x,y) \in [0,2\pi] \times [0,2\pi]. \tag{29}$$

We transform the PDE into velocity form so that it can be included in Walrus' channel mapping. The resolution is 128x128. The trajectories are 501 time steps long, where the first 100 are discarded. We use 48 trajectories for validation and test each.

F Model and Training Details

All student models are trained to predict the difference, i.e., $\hat{\mathbf{u}}^{t+1} = \mathcal{S}(\mathbf{u}^t) = \mathcal{M}(\mathbf{u}^t) + \mathbf{u}^t$. The learning rate is set to 10^{-3} initially and scaled down to 10^{-6} using cosine annealing. We use the standard Adam optimizer (Kingma & Ba, 2015) and a batch size of 32. For the main results, we report the test error for the model checkpoint with the best error on the validation set for the ground-truth-only and relational KD baselines, since we encountered overfitting for these methods. For TREX and IC-KD, we report the last model checkpoint to be fair to the teacher baseline, which also did not use early-stopping. We use the original training loss from Poseidon over the batch $\mathcal{B}$:

$$L^1_{\text{train}}(\theta) = \frac{1}{N_{\text{QOI}}} \sum_{\text{QOI}} \frac{\sum_{b\in\mathcal{B}} \sum_{t} \sum_{\mathbf{x}} \sum_{c_{\text{QOI}}} \left| \hat{\mathbf{u}}_{\theta,b}\left(t, \mathbf{x}, c_{\text{QOI}}\right) - \mathbf{u}_b\left(t, \mathbf{x}, c_{\text{QOI}}\right) \right|}{\sum_{b\in\mathcal{B}} \sum_{t} \sum_{\mathbf{x}} \sum_{c_{\text{QOI}}} \left| \mathbf{u}_b\left(t, \mathbf{x}, c_{\text{QOI}}\right) \right| + \varepsilon_{L_1}}.$$

TFNO. We use the Tucker factorized FNO (Kossaifi et al., 2024) version with the implementation provided by Kossaifi et al. (2025). The factorization reduces the number of parameters with a compression factor of 0.1. The network uses 32 modes per dimension, 32 hidden channels, three layers, and the GELU (Hendrycks & Gimpel, 2016) activation function.

U-Net. For the student model ablations, we use the modern U-Net (Ronneberger et al., 2015) as provided by Gupta & Brandstetter (2023). The U-Net has 4 levels of resolution, increasing the number of channels per level by 1, 2, 2, and 4, starting at 16 hidden channels after the embedding, resulting in 9,161,332 parameters. For U-Net, we found it advantageous to train with a 5-step error for the model ablation experiment.

FNO. We also use a traditional FNO (Li et al., 2021) without factorization for the model ablations. The model uses the same number of modes and channels as the TFNO, resulting in 5,042,836 parameters. The FNO uses 4 layers.

Poseidon. As the teacher, we use the Poseidon-L model (Herde et al., 2024). Poseidon-L was pretrained on six different datasets containing Navier-Stokes and compressible Euler equations with different sets of initial conditions. The model uses a SwinV2 transformer as its backbone, a multi-resolution vision transformer. The architecture operates on a latent representation with embedding dimension $C = 192$. For each hierarchical level i, the model employs $t_i = 8$ blocks, resulting in a total of 629M trainable parameters. The network consists of $L = 4$ levels, connected via $L - 1$ downsampling and upsampling operations to progressively refine spatial resolution. The number of attention heads per level is defined as:

$$[h_1, h_2, h_3, h_4] = [3, 6, 12, 24],$$

allowing increased representation capacity at coarser scales. In addition, each level incorporates two ConvNeXt layers (Liu et al., 2022b). The model operates on input patches of size $p = 4$ and a window size of $M = 16$.

Training is performed using the AdamW optimizer (Loshchilov & Hutter, 2019) with cosine learning rate decay. The main learning rate is initialized as $\eta_b = 5 \cdot 10^{-5}$. A weight decay of 10^{-6} is applied for regularization. The batch size is set to 16, and the model is trained for 200 epochs without early stopping. Gradient norms are clipped at a maximum value of 5 to ensure numerical stability.

Walrus. For Walrus (McCabe et al., 2026) fine-tuning, we used only 50k sample presentations to the model (batch size 4). We used global normalization and followed the original training procedure otherwise. For the distillation, we started 50 rollouts from different random sub-sequences in the trajectory of length 100. We found noise during Walrus rollout harmful, likely since adding noise to a multi-frame input does not just create a different state, but also suggests different physics to the model if it learned to infer it from its inputs. More importantly, it was crucial to deactivate the jitter added in Walrus inference, as this creates stochastic predictions and hence noisy labels. Walrus uses the VRMSE metric:

$$\mathrm{VRMSE}_{t_1:t_2}\big(\widehat{\mathbf{u}}, \mathbf{u}\big) = \frac{1}{t_2 - t_1 + 1} \sum_{t=t_1}^{t_2} \sqrt{\frac{\left\langle \left|\widehat{\mathbf{u}}_t - \mathbf{u}_t\right|^2 \right\rangle}{\left\langle \left|\mathbf{u}_t - \overline{\mathbf{u}}_t\right|^2 \right\rangle + \varepsilon}}. \tag{30}$$

We use the normalized mean absolute error as the loss function:

$$\mathcal{L}_{\mathrm{W}}(\mathbf{u}, \hat{\mathbf{u}}) = \frac{1}{T} \sum_{t=0}^{T-1} \left| \mathrm{Norm}_{\Delta}\big(\hat{\mathbf{u}}^{\,t+1} - \mathbf{u}^t\big) - \mathrm{Norm}_{\Delta}\big(\mathbf{u}^{t+1} - \mathbf{u}^t\big) \right|_1 .$$

The normalization is performed with the global mean and standard deviation of the transition differences.

G Knowledge Distillation Details

In this section, we provide additional results on the KD techniques used in this paper.

TREX For TREX, we use 100 rollout steps starting from the initial condition by default. The noise is applied after 10 steps, with a scaling factor of $\sigma = 1$. The TREX loss weight is set to 1 as well. The noisy rollouts are regenerated every 1000 epochs.

IC-KD For the IC-KD baseline, we use a buffer of teacher-generated trajectories that we regenerate every 10 epochs, with the size of the buffer determined by the number of ground truth trajectories available. The teacher-generated rollouts are then also divided into sub-trajectories for n-step training. Note that for the Brownian bridge task, the initial conditions are computationally intensive to compute because they are evolved from an earlier state.

$$\mathcal{L}_{\text{IC}} = \mathbb{E}_{\mathbf{u}_0 \sim p_0} \left[\frac{1}{T} \sum_{t=0}^{T-1} \ell \left(\mathcal{S}(\mathbf{u}^t), \mathcal{T}(\mathbf{u}^t) \right) \right], \qquad \mathbf{u}^{t+1} = \mathcal{T}(\mathbf{u}^t).$$

Relational KD Relational KD (Park et al., 2019) aims to make the distance between an input pair i, j from a mini-batch in the student feature space close to the distance in the feature space:

$$\mathcal{L}_{\text{RKD}} = \sum_{i<j} \ell_\delta \Big(d(\mathbf{t}_i, \mathbf{t}_j), d(\mathbf{s}_i, \mathbf{s}_j) \Big), \tag{31}$$

where $\mathbf{t}$ and $\mathbf{s}$ are the teacher and student features, respectively. The loss is the Huber distance

$$\ell_\delta(x, y) = \begin{cases} \frac{1}{2}(x-y)^2 & \text{if } |x-y| \leq 1 \\ |x-y| - \frac{1}{2} & \text{otherwise} \end{cases}. \tag{32}$$

Since the feature spaces of the spatial neural network are very large, we use Gaussian sketching, which uses random Gaussian matrices R to reduce the input features to a lower dimension for the teacher

$$\mathbf{t}_i = \mathbf{R}_T \tilde{\mathbf{t}}_i, \qquad (\mathbf{R}_T)_{ab} \sim \mathcal{N}\big(0, \tfrac{1}{k}\big), \qquad \mathbf{R}_T \in \mathbb{R}^{k \times d_T}, \tag{33}$$

and equivalently, the student model

$$\mathbf{s}_i = \mathbf{R}_S \tilde{\mathbf{s}}_i, \qquad (\mathbf{R}_S)_{ab} \sim \mathcal{N}\big(0, \tfrac{1}{k}\big), \qquad \mathbf{R}_S \in \mathbb{R}^{k \times d_S}. \tag{34}$$

It can be shown that the error in the distances $d(\mathbf{t}_i, \mathbf{t}_j)$ introduced by the Gaussian sketch is bounded (Holzmüller et al., 2023; Johnson & Lindenstrauss, 1984). We use $k = 512$ in all experiments.

The loss $\mathcal{L}_{\text{RKD}}$ is added to the main, ground truth loss with a scaling factor of 0.001. We use the last-layer features for both the student and the teacher.

H Further Results

Tab. 5 shows the main results for the median relative L^1 metric with the 95% confidence interval of the mean. For completeness, we also show the relative L^1 error for all time steps (Tab. 6). Additionally, we report the training durations in Tab. 7.

Table 5: Median rel. L^1 metric of the *last* time step. Entries show the mean with its 95% confidence interval calculated over three seeds.

Method	$N = 4$	$N = 8$	$N = 16$	$N = 32$
		CE-RPUI		
TREX	$\mathbf{0.421 \pm 0.015}$	$\mathbf{0.295 \pm 0.182}$	$\mathbf{0.197 \pm 0.038}$	0.144 ± 0.010
IC-KD	0.440 ± 0.077	0.304 ± 0.167	0.223 ± 0.105	0.145 ± 0.006
Relational KD	0.608 ± 0.099	0.446 ± 0.068	0.296 ± 0.031	0.196 ± 0.020
Only GT	0.608 ± 0.086	0.458 ± 0.092	0.296 ± 0.011	0.199 ± 0.018
Teacher	0.446 ± 0.074	0.310 ± 0.157	0.223 ± 0.108	0.143 ± 0.009
		NS-SVS		
TREX	$\mathbf{0.043 \pm 0.037}$	$\mathbf{0.021 \pm 0.005}$	$\mathbf{0.014 \pm 0.001}$	$\mathbf{0.011 \pm 0.002}$
IC-KD	0.045 ± 0.032	0.025 ± 0.005	0.018 ± 0.003	0.015 ± 0.004
Relational KD	0.178 ± 0.049	0.050 ± 0.020	0.021 ± 0.004	0.011 ± 0.001
Only GT	0.171 ± 0.053	0.060 ± 0.021	0.023 ± 0.005	0.011 ± 0.001
Teacher	0.043 ± 0.030	0.024 ± 0.005	0.018 ± 0.005	0.015 ± 0.003
		NS-PwC		
TREX	$\mathbf{0.056 \pm 0.003}$	$\mathbf{0.042 \pm 0.017}$	0.032 ± 0.012	0.021 ± 0.003
IC-KD	0.067 ± 0.012	0.053 ± 0.039	0.042 ± 0.038	0.029 ± 0.007
Relational KD	0.175 ± 0.014	0.059 ± 0.011	$\mathbf{0.027 \pm 0.003}$	0.013 ± 0.001
Only GT	0.282 ± 0.081	0.068 ± 0.011	0.029 ± 0.002	$\mathbf{0.013 \pm 0.001}$
Teacher	0.070 ± 0.011	0.055 ± 0.040	0.044 ± 0.035	0.031 ± 0.005
		NS-BB		
TREX	$\mathbf{0.036 \pm 0.007}$	$\mathbf{0.026 \pm 0.007}$	0.023 ± 0.006	0.016 ± 0.008
IC-KD	0.041 ± 0.010	0.034 ± 0.004	0.031 ± 0.005	0.023 ± 0.009
Relational KD	0.111 ± 0.022	0.034 ± 0.007	$\mathbf{0.015 \pm 0.001}$	$\mathbf{0.007 \pm 0.002}$
Only GT	0.178 ± 0.077	0.041 ± 0.015	0.015 ± 0.003	0.007 ± 0.001
Teacher	0.046 ± 0.010	0.039 ± 0.005	0.033 ± 0.003	0.025 ± 0.008

Table 6: Median rel. L^1 metric over *all* timesteps. Entries show the mean with its 95% confidence interval calculated over three seeds.

Method	$N = 4$	$N = 8$	$N = 16$	$N = 32$
		CE-RPUI		
TREX	$\mathbf{0.277 \pm 0.027}$	$\mathbf{0.182 \pm 0.065}$	$\mathbf{0.127 \pm 0.019}$	0.095 ± 0.008
IC-KD	0.281 ± 0.035	0.184 ± 0.080	0.135 ± 0.055	0.093 ± 0.005
Relational KD	0.471 ± 0.074	0.316 ± 0.042	0.197 ± 0.025	0.129 ± 0.014
Only GT	0.466 ± 0.067	0.327 ± 0.049	0.201 ± 0.011	0.130 ± 0.005
Teacher	0.282 ± 0.038	0.185 ± 0.078	0.129 ± 0.055	0.084 ± 0.005
		NS-SVS		
TREX	0.020 ± 0.014	$\mathbf{0.009 \pm 0.002}$	$\mathbf{0.006 \pm 0.001}$	0.004 ± 0.001
IC-KD	0.020 ± 0.009	0.011 ± 0.002	0.008 ± 0.001	0.006 ± 0.001
Relational KD	0.070 ± 0.023	0.019 ± 0.009	0.008 ± 0.002	0.004 ± 0.001
Only GT	0.068 ± 0.024	0.022 ± 0.009	0.008 ± 0.002	$\mathbf{0.004 \pm 0.001}$
Teacher	0.020 ± 0.008	0.011 ± 0.002	0.008 ± 0.001	0.006 ± 0.001
		NS-PwC		
TREX	$\mathbf{0.032 \pm 0.002}$	$\mathbf{0.024 \pm 0.007}$	0.019 ± 0.004	0.014 ± 0.002
IC-KD	0.037 ± 0.009	0.028 ± 0.016	0.023 ± 0.014	0.017 ± 0.003
Relational KD	0.105 ± 0.011	0.037 ± 0.007	$\mathbf{0.018 \pm 0.002}$	0.010 ± 0.001
Only GT	0.172 ± 0.048	0.043 ± 0.008	0.019 ± 0.002	$\mathbf{0.010 \pm 0.001}$
Teacher	0.039 ± 0.011	0.029 ± 0.016	0.024 ± 0.014	0.018 ± 0.002
		NS-BB		
TREX	$\mathbf{0.022 \pm 0.005}$	$\mathbf{0.016 \pm 0.005}$	0.013 ± 0.002	0.009 ± 0.003
IC-KD	0.023 ± 0.006	0.019 ± 0.004	0.017 ± 0.001	0.012 ± 0.004
Relational KD	0.067 ± 0.012	0.022 ± 0.004	$\mathbf{0.010 \pm 0.001}$	$\mathbf{0.005 \pm 0.001}$
Only GT	0.108 ± 0.050	0.026 ± 0.009	0.010 ± 0.002	0.005 ± 0.001
Teacher	0.026 ± 0.007	0.022 ± 0.002	0.018 ± 0.001	0.014 ± 0.004

Table 7: Average training duration in minutes on NS-PwC.

Method	$N = 4$	$N = 8$	$N = 16$	$N = 32$
TREX	47.1 ± 1.9	99.8 ± 3.2	147.4 ± 1.4	291.4 ± 4.6
IC-KD	85.2 ± 2.3	117.7 ± 4.5	193.5 ± 1.4	338.7 ± 6.1
Relational KD	38.2 ± 1.4	72.9 ± 2.7	143.1 ± 3.8	280.2 ± 9.8
Only Ground Truth	37.3 ± 0.4	70.2 ± 0.4	138.4 ± 4.3	271.5 ± 7.8

H.1 Example Rollout

Fig. 8 shows a full example rollout for the NS-SVS task including the noise applied to the teacher.

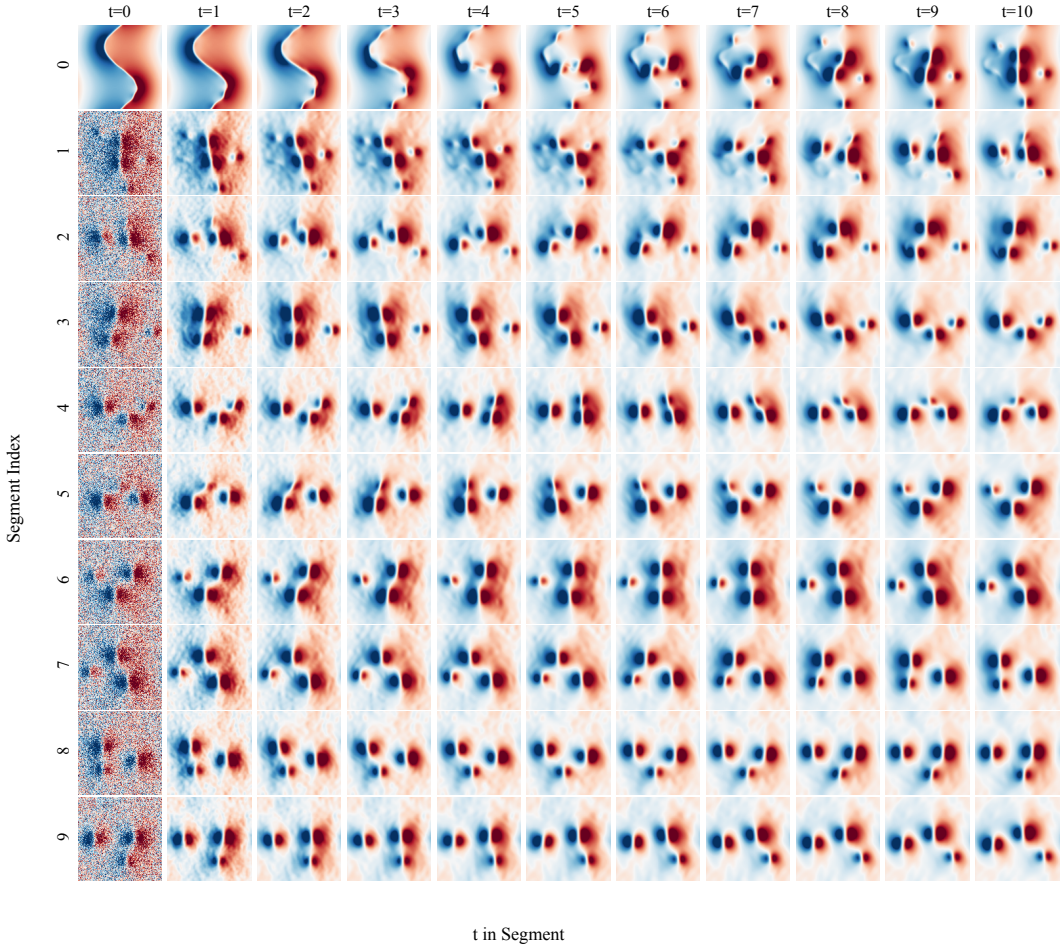

Figure 8: Example trajectory (velocity-x channel) generated using the teacher for $\sigma = 1$ on NS-SVS. After each sub-trajectory (row), the noise is applied.

H.2 Error Growth

In Fig. 9, the error growth for all four considered Poseidon tasks is depicted. While the one-step error is worse on average than the teacher's, the student often has a slower error growth.

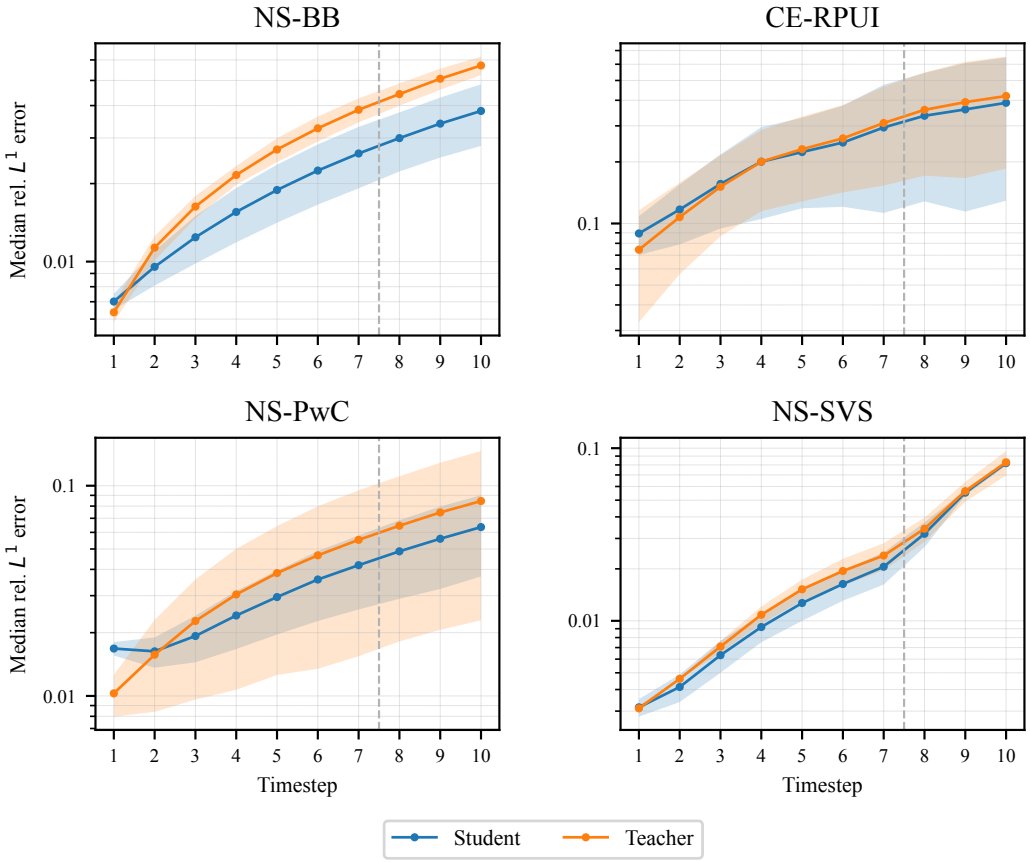

Figure 9: Error over time for the student trained with TREX (8 trajectories). The shaded area shows the 95% confidence interval of the mean (3 seeds). The dashed line shows the time horizon seen during training.

H.3 Equivariance

As a complement to the final-step comparison in Fig. 7, Fig. 10 shows the complete shifted rollout on NS-SVS. On the unshifted rollout, the teacher initially appears slightly closer to the ground truth than the student. After applying the spatial shift, however, the teacher prediction quickly degrades, starting around steps 3/4, while the student remains consistent throughout the rollout.

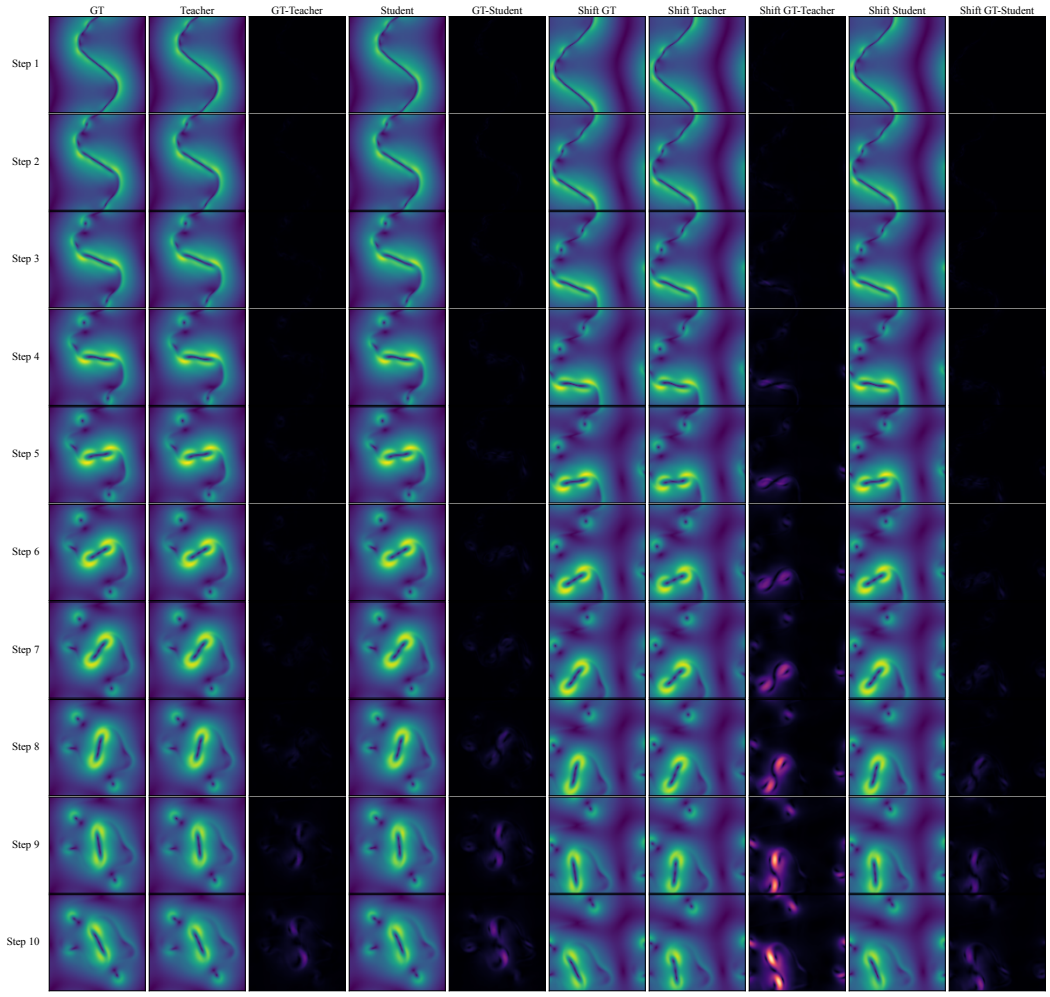

Figure 10: Full shifted-rollout comparison on NS-SVS. Each row shows one autoregressive step. The teacher accumulates larger errors under the spatial shift, while the equivariant student remains consistent with the shifted ground truth throughout the rollout.

H.4 Self-Distillation

We also perform a self-distillation experiment to further examine the influence of the teacher pretraining. We use the TFNO trained only on ground truth samples as the teacher for distillation into a freshly initialized model of the same architecture. The results in Tab. 8 show that, while self-distillation has a small positive regularizing effect, the main advantage comes from distilling the pretrained, higher-capacity foundation model.

Table 8: Self-distillation experiment using TREX on CE-RPUI (4 trajectories).

Method	Median rel. L^1
Only GT	0.608
TREX (self-distillation)	0.585
TREX (from Poseidon)	0.421

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main claims are stated in the abstract and introduction, and are supported by the experimental results. The paper claims that TREX distills PDE foundation models into smaller students, can match or sometimes surpass the teacher, improves inference efficiency, and can incorporate inductive biases such as equivariance. These claims are evaluated in the experiments.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper appendix includes a dedicated limitations section.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A]

Justification: The paper does not present formal theoretical results or theorems. The mathematical discussion is used to motivate the method and define the training objective.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The paper describes the teacher and student models, TREX training procedure, baselines, loss functions, rollout lengths, noise schedule, dataset sizes, evaluation metric, and number of seeds. Additional implementation and model details are provided in the appendix.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The experiments use existing public datasets and models. The code is available at `github.com/dmusekamp/trex`.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The paper specifies the training setup, baselines, model architecture, batch size, number of updates, TREX rollout length, noise frequency, noise scale, downstream tasks, dataset sizes, and evaluation metric. Further dataset and model details are provided in the appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

Justification: The main results are reported across three seeds, using different training trajectories for each seed. Tables report uncertainty values for the main experiments. Some auxiliary ablations are explicitly marked as single-seed where applicable. Additionally, experiments using Walrus were performed with a single seed due to computationally demanding fine-tuning.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The paper reports inference time, memory usage, parameter counts, training times and GPUs used.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research does not involve human subjects, private data, surveillance, or other high-risk data sources. To the best of our knowledge, the work conforms to the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.

- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Broader impacts are discussed in the Appendix.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The work does not release high-risk models or datasets.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: Authors are properly credited for the public models and data that are used in this work. TREX will be made publicly available under an MIT license.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: The paper does not introduce a new dataset or benchmark asset. The code will be made publicly available under an MIT license upon publication.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: We do not use crowdsourcing or human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: The paper does not involve human subjects, so IRB approval or equivalent review is not applicable.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: LLMs are not part of the presented methods.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.