ETH zürich

FNO based Foundational Models for Phase Field Dynamics

Benedict Armstrong
benedict.armstrong@inf.ethz.ch

January 2025

Introduction

This study implements a variant of Fourier Neural Operators (FNO) [1] to solve the one-dimensional Allen-Cahn equation. The Allen-Cahn equation is a fundamental partial differential equation (PDE) used to model phase separation processes in materials science. A primary challenge in this task is ensuring that the trained model can generalize to unseen initial conditions, a critical requirement for robust neural operator-based PDE solvers.

The FNO architecture is designed to learn mappings between function spaces, making it particularly well-suited for solving PDEs. Unlike traditional numerical solvers, FNOs operate in the frequency domain, enabling efficient learning of complex spatial and temporal dependencies and invariance to the mesh. The implementation of the model is structured across two primary files: lib/model.py, which defines the neural network architecture, and lib/layers.py, which contains the necessary layer definitions.

This work evaluates the performance of the trained model in predicting the solution of the Allen-Cahn equation for various initial con tions, including those not seen during traini The results highlight the model's ability to ge eralize and provide insight into the applicabilit of neural operators for PDE-driven problems.

Data Generation

For this task, we were asked to generate our own data for training and testing. The data was generated by solving the *Allen-Cahn* equation using the `scipy.solve_ivp` solver and the `RK45` method implemented in `lib/allen_cahn.py`. Due to the exponential dependence of the convergence behavior on the ε parameter the model was train on a range of ε values. I used the suggested values of $\varepsilon = 0.1, 0.05, 0.02$ and $N_{\text{train}} = 400$ total samples. The `data.ipynb` notebook contains an interactive visualization of the generated data for different ε values and initial conditions. Solutions for lower epsilon values converge very quickly and sampling the same 5 equidistant timesteps for all ε values doesn't capture the trajectories well. To mitigate this I first scaled the time dimension by ε and then (only for training data) sampled 5 random timesteps from a non uniform distribution in $[0, 1]$. The distribution was generated using the following code snippet and is shown in Figure 1.

```
t_train = np.logspace(0, 1, 30, base=3)
t_train = (t - 1) / t[-1]
```

Figure 1: Evaluation points for ● training / ✘ testing

This ensures we use only the 5 temporal snapshots for training and testing and that the snapshots capture a greater range of the solution dynamics.

Model

As mentioned above I used a very similar model to the one used for the *TFNO* task only being adapted to accept tensors with last dimension 2 and include the ε in the time embedding.

Training

The data was split into a training and test set with a ratio of $80:20$ and trained for 100 epochs. The model was trained using the `Adam` optimizer and the `CosineAnnealingWarmRestarts` scheduler. The full parameters used to train the model are listed in Table 3. The model achieved an average relative L_2 error of 0.019 on the entire test set, Table 1 shows the errors and some examples are plotted in Figure 4 We can also see in Table 1 that the model performs similarly well on all tested types of initial conditions.

Dataset (IC Type), Δt			rel. L_2 err.
Test	(All), $[0.25, 0.5, 0.75, 1.]$	$\Delta t =$	0.019
Test (All), $\Delta t = 1.00$			0.020
Test (Fourier), $\Delta t = 1.00$			0.024
Test (GMM), $\Delta t = 1.00$			0.015
Test (Piecewise), $\Delta t = 1.00$			0.021

Table 1: Predictions on test dataset at different Δt

Results

After training, the model was evaluated on a set of out of distribution (OOD) data. The model achieved a relative L_2 error of 0.028. Examples of the model's predictions on the OOD data can be found in the `eval_ood.ipynb` notebook and in Figure 5 The model performs moderately well on the OOD data as shown in Table 2.

Dataset, Δt	Average rel. L_2 err.
OOD, All Δt	0.0287
OOD, $\Delta t = 1.00$	0.2189

Table 2: Predictions on OOD dataset at different Δt

Generalization to unseen ε values & initial conditions

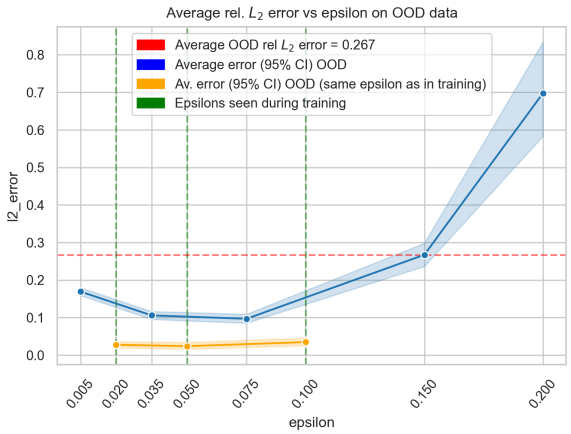

Figure 2: L_2 error on OOD data vs. ε for the Allen-Cahn model

Figure 2 shows well how the model fails to extrapolate to high unseen epsilons while performing reasonably well on interpolated and lower epsilon values. The orange line shows the error when only the initial conditions are different (higher frequency and sharper transitions) which yields good results. Figure 5 shows some of the model's predictions on OOD data.

High Frequency Initial Conditions and low ε

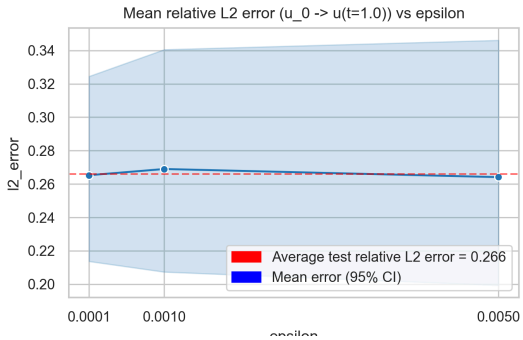

Figure 3: L_2 error on OOD (low ε) data vs. ε for the Allen-Cahn model

The model was also tested on super low ε values such as $\varepsilon = 0.0001, 0.001, 0.005$ showing a sig-

nificant decrease in performance. Figure 6 shows some examples of high frequency initial conditions, which seem The model's predictions on the test data for different ε values can be found in the `eval_ood_hf_low_e.ipynb` notebook.

Conclusion & Improvements

The model performs well on the test data and reasonably well on OOD data for the seen, interpolated and lower ε values. The model fails to generalize to larger extrapolated ε values. I also tried fine-tuning the model on the OOD data but only got moderate or no improvements in the relative L_2 error.

Proof for the stability of the *Allen-Cahn* equation

A More detailed results including accounting for discretization error and further discussion can be found in [2, Chapter 6].

Proof:

With $c_f = \sup_{s\in B_1(0)} |f'(s)|$, we have

$$|f(x) - f(y)| =\leq c_f\ |x-y|$$

for all $x, y \in \mathbb{R}$. The difference $\delta = u - \tilde{u}$ satisfies

$$(\partial_t, v) + (\nabla\delta, \nabla v) = -\varepsilon^2 (f(u) - f(\tilde{u}), v)$$

for almost every $t \in (0,T)$ and all $v \in H_0^1(\Omega)$. We use the test function $v = \delta$ and get

$$\begin{aligned}\frac{1}{2}\frac{d}{dt}\ \|\delta\|^2 + \|\nabla\delta\|^2 &\leq c_f\varepsilon^{-2}\ \|\delta\|^2 \\ &\leq c_f\varepsilon^{-2}\ \|\delta\|^2 + \frac{1}{2}(\|\delta\ \|^2 + \|\nabla\delta\|^2) \\ &\leq \frac{1}{2}\left(1 + 2c_f\varepsilon^{-2}\right)\ \|\delta\|^2 + \frac{1}{2}\ \|\nabla\delta\|^2\end{aligned}$$

Which absorbs the $\|\nabla\delta\|^2$ term. Integrating over $(0,T')$ we get

$$\|\delta(T')\|^2 + \int_0^{T'} \|\nabla\delta\|^2\,\mathrm{d}t \leq \|\delta(0)\|^2 + \left(1 + 2c_f\varepsilon^{-2}\right)\int_0^{T'} \|\delta\|^2\,\mathrm{d}t$$

We define $A = \|\delta(0)\|^2$, $b = \left(1 + 2c_f\varepsilon^{-2}\right)$ and

$$y(t) = \|\delta(t)\|^2 + \int_0^t \|\nabla\delta\|^2\,\mathrm{d}s$$

and apply the Gronwall lemma [2, Proposition 6.2] for nonnegative functions $y \in C([0,T])$

$$y(T') \leq A + \int_0^{T'} b(t)y(t)\,\mathrm{d}t$$

with a nonnegative function $b \in L^1(0,T')$. We have

$$y(T') \leq A\exp\left(\int_0^T b(t)\,\mathrm{d}t\right)$$

which proves the claim:

$$\|\delta(t)\|^2 + \int_0^t \|\nabla\delta\|^2\,\mathrm{d}s \leq \|\delta(0)\|^2 \exp\left(T + T2c_f\varepsilon^{-2}\right)$$

■

Even for moderately large ε values such as $\varepsilon \approx 10^{-1}$ the exponential factor which depends on ε^{-2} makes this error estimate only moderately useful.

Bibliography

[1] Z. Li *et al.*, "Fourier Neural Operator for Parametric Partial Differential Equations." [Online]. Available: https://arxiv.org/abs/2010.08895

[2] S. Bartels, *Numerical Methods for Nonlinear Partial Differential Equations*. Springer, 2015.

Appendix

Parameter	Value
Epochs	100
Batch size	1024
Optimizer	Adam
weight_decay	1e-5
Learning rate	0.0005
Scheduler	CosineAnnealingWarmRestarts
eta_min	1e-6
T_0	20
Loss	Relative L_2 error
Modes	16
Width	64
Fourier Layers	4

Table 3: Parameters used for training Allen-Cahn FNO model

Figure 4: Predictions on test data for the Allen Cahn TFNO model

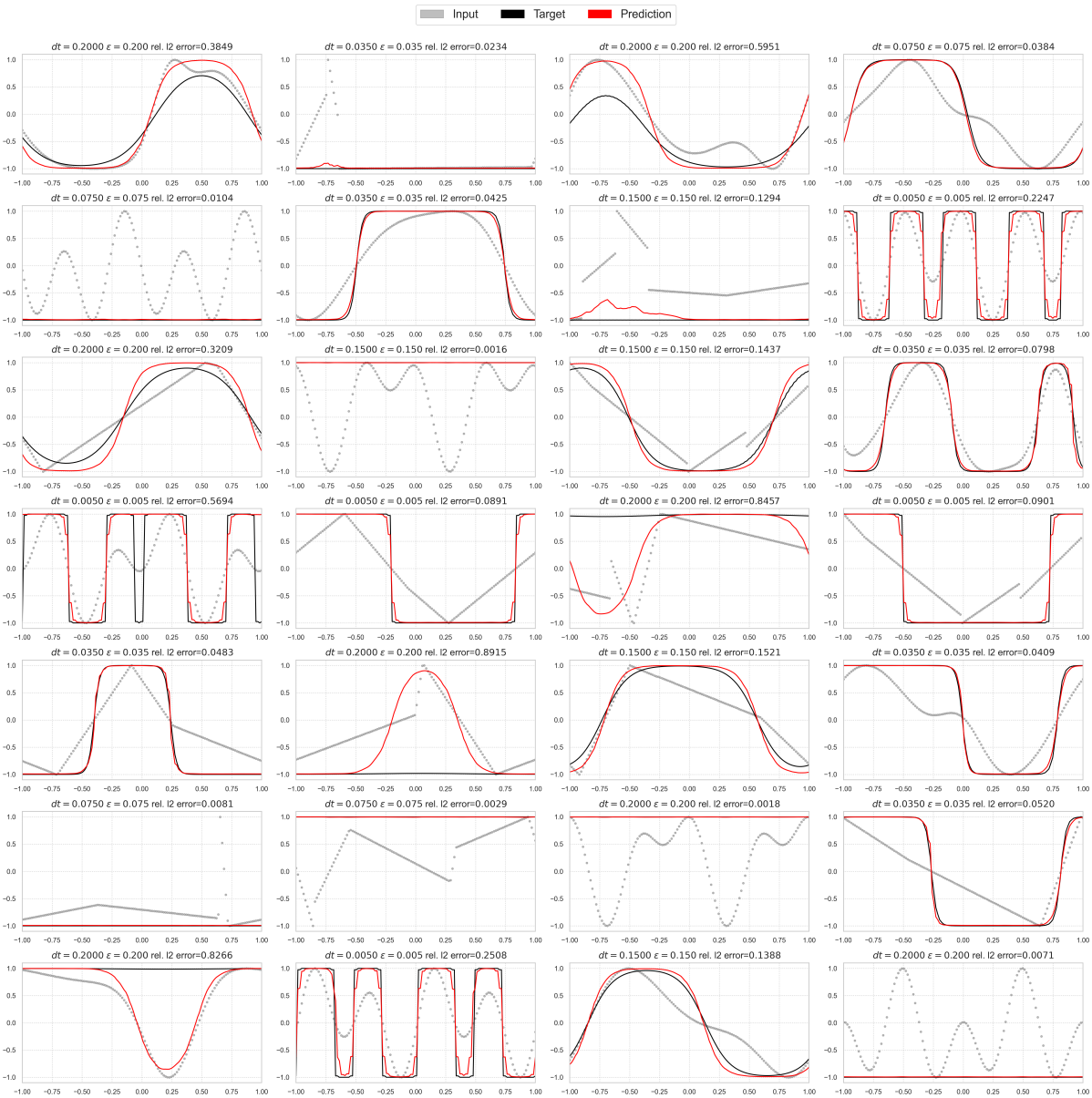

Figure 5: Predictions on OOD data for the Allen Cahn TFNO model

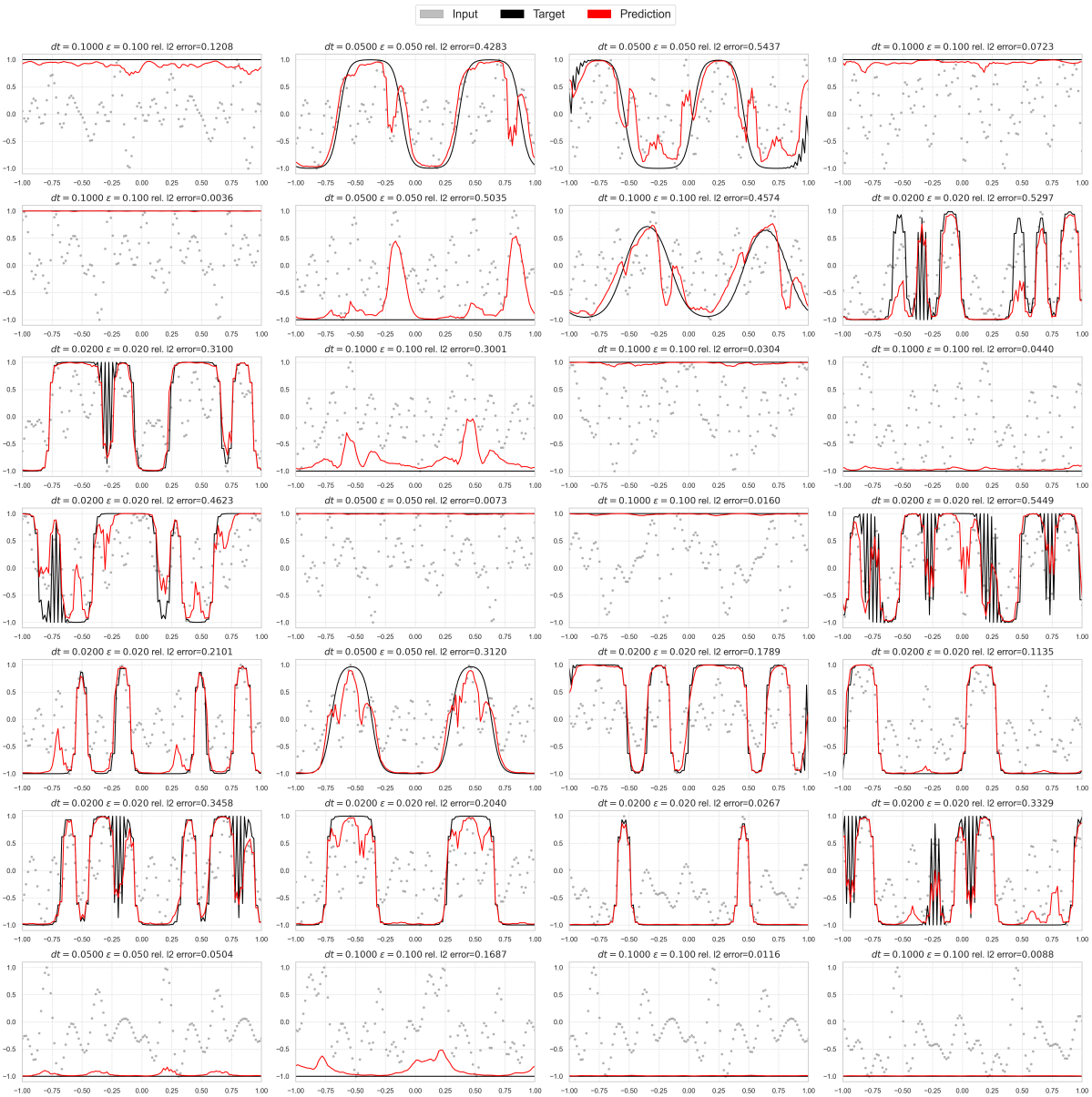

Figure 6: Predictions on OOD (High Frequency IC) data for the Allen Cahn TFNO model

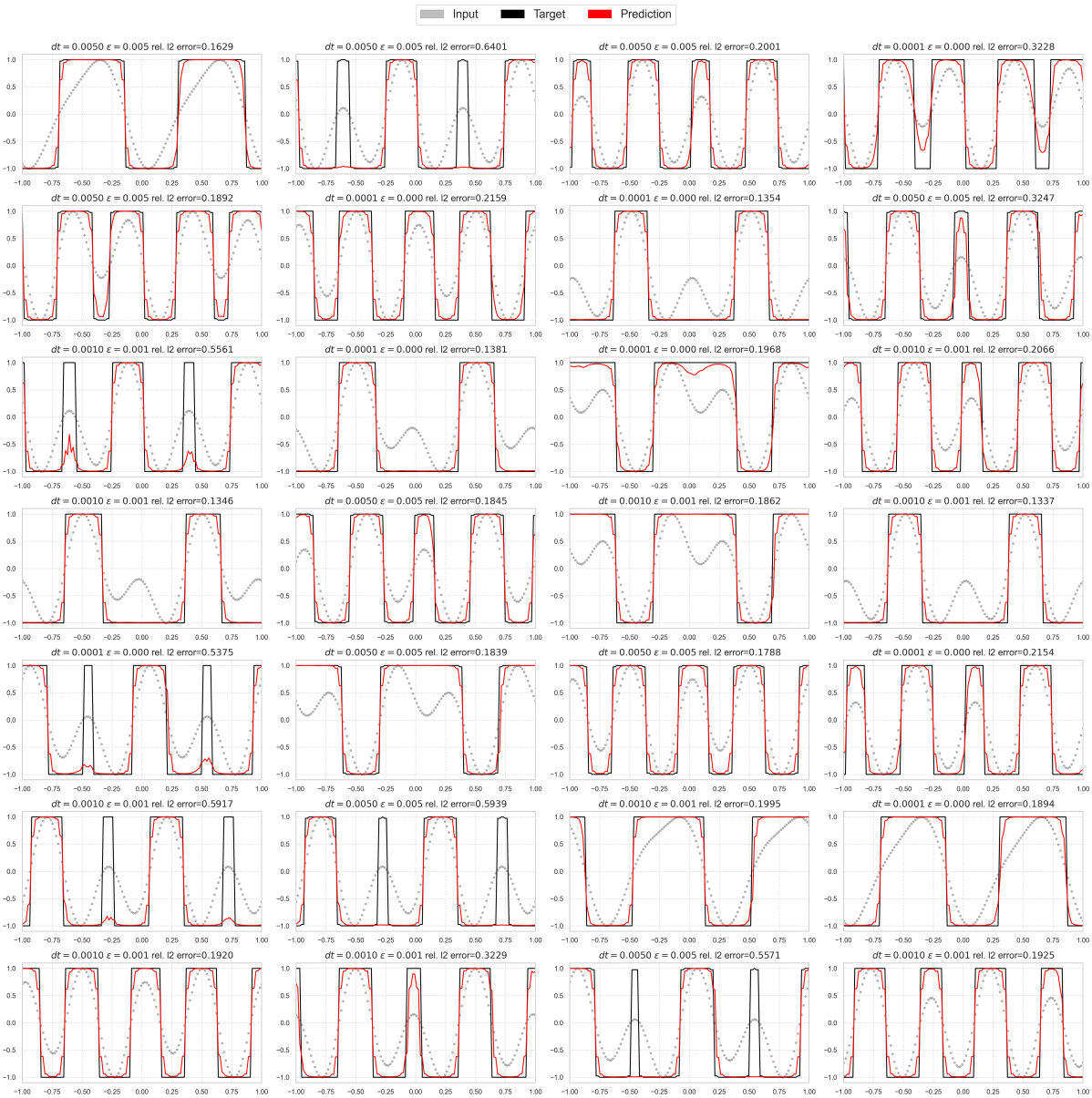

Figure 7: Predictions on OOD (Low ε) data for the Allen Cahn TFNO model