ExtremONet: Extreme-Learning-based Neural Operator for identifying dynamical systems.

Jari Beysen[1], Floriano Tori[1]

1. Background

Learning dynamical systems is labor intensive, different variations require re-training of models.

Neural Operators learn mappings between function spaces, allowing them to generalize across trajectories with (slightly) different underlying mechanics.

DeepONets was the first Neural operator developed; it requires two separate Neural Networks: one for a functional input, one for a vectorial input.

Training DeepONets is slow and requires a lot of data; not computationally efficient nor ecologically responsible.

Extreme Learning Machines are random 1-layer NNs that are trained via Ridge Regression, and therefore in 1 step.

2. Contributions

We developed the **ExtremONet**: an Extreme-Learning-Based Neural Operator.

- Trainable in 1 step (~15 steps for the regularization term in Ridge Regression); **100-10 000x faster to train** than a DeepONet for similar performance.
- **10x less generalization error** due to regularization term and reduced amount of parameters.
- **Out-of-Distribution Generalization!**
- Some theoretical justification (Approximation Theorem in future work).

3. Experiments

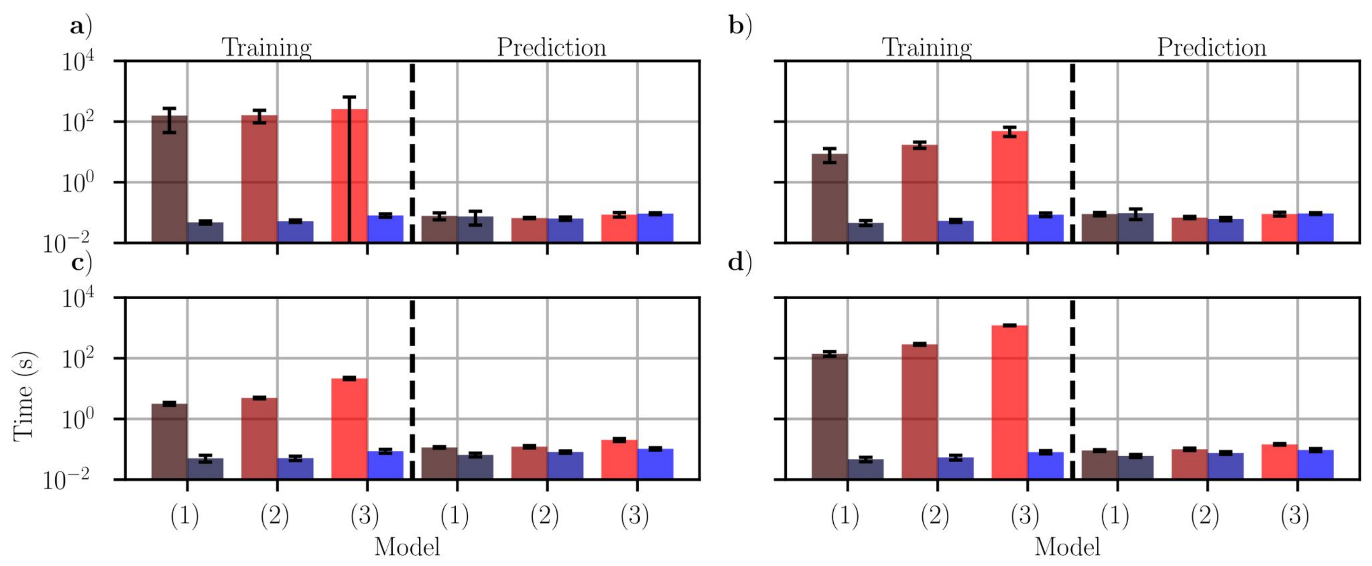

Fig.: Train- and prediction mean and standard deviation time of three DeepONets (red) and three ExtremONets (blue) (30 repeats). Model sizes are small (1), medium (2), and large (3) (also indicated by bar colour saturation). a) Lorenz-63 system. b) Gravity pendulum ODE with GRF (t). c) Parametrized Sine-Gordon PDE. d) Diffusion PDE with GRF (x).

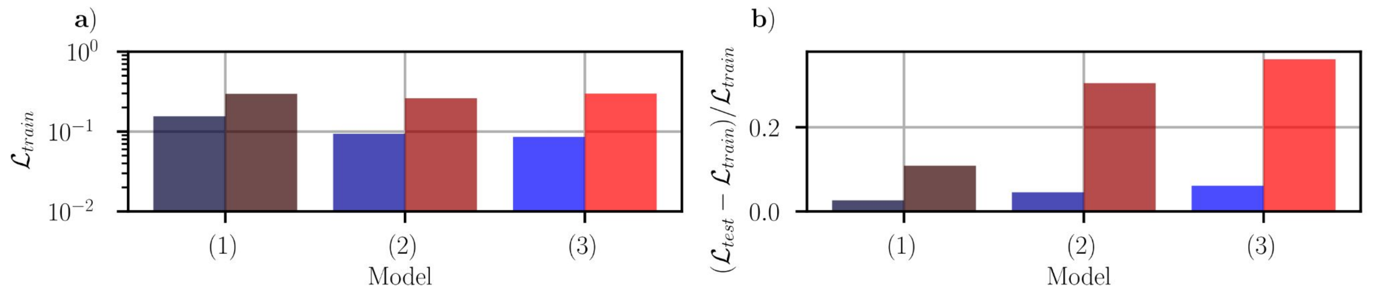

Fig.: a) Mean of the training error. b) Mean of the relative generalization error. DeepONet (red) and the ExtremONet (blue) for the increasing model sizes; model sizes are small (1), medium (2), and large (3) (also indicated by bar colour saturation).

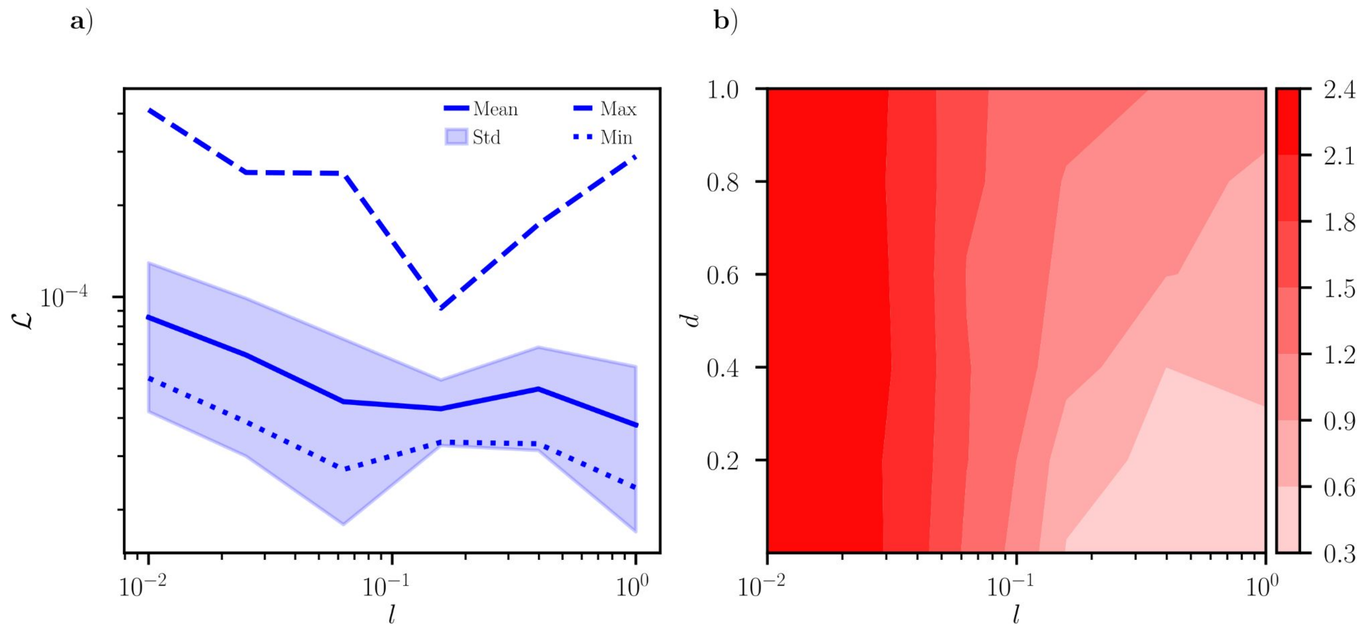

Fig.: ExtremONet in phase space trained on the diffusion with spatial GRF dataset; 30 repeats. a) Train loss statistics vs. GRF length scale l ∈ [0.01, 1]; y-axis is log10-NMSE. b) log10 test-train error ratio for varying GRF length scales (x-axis) and drift d ∈ [0, 1] (y-axis).

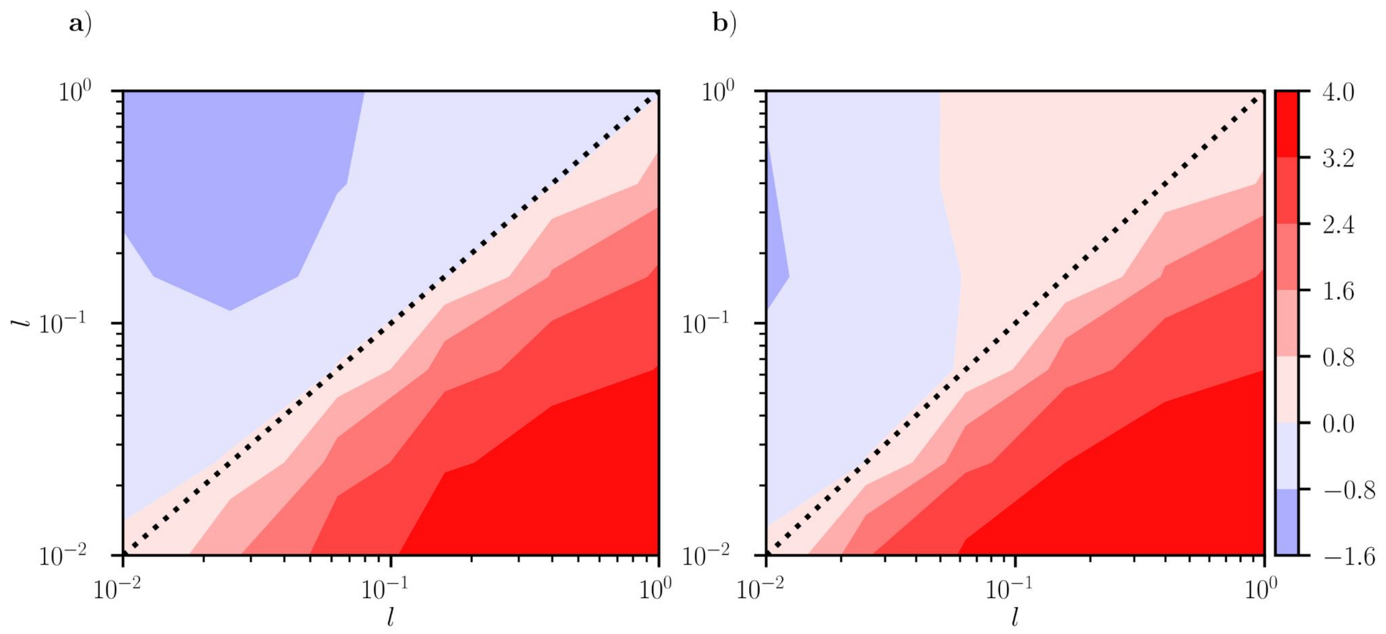

Fig.: ExtremONet out-of-function-space drift: log10 test-train error ratio vs. GRF length scale. x-axis: training scale, y-axis: test scale; 30 repeats. a) Pendulum ODE with GRF (t). b) Diffusion PDE with GRF (x).

Theoretical contributions

Rank-1 UATO: *For compact sets* $K_1 \subset X$, $K_2 \subset \mathbb{R}^d$, $V \subset C(K_1)$, *a non-linear operator* $G: V \to C(K_2)$, *the following inequality:*

$$\inf_{\alpha}\left|G\left(u(\cdot)\right)(t) - \sum_{k=1}^{m'} \alpha^{(R)}_{k(W)} \psi\left(\sum_{l=1}^{r}\alpha^{(B)}_{kl(W)} u(s_l) + \alpha^{(B)}_{k(b)}\right)\phi\left(\alpha^{(T)}_{k(W)} t + \alpha^{(T)}_{k(b)}\right)\right| \le$$

$$\inf_{\theta}\left|G\left(u(\cdot)\right)(t) - \sum_{k=1}^{p}\sum_{j=1}^{m} \theta^{i(R)}_{j(W)} \psi\left(\sum_{l=1}^{r}\alpha^{i(B)}_{jl(W)} u(s_l) + \alpha^{i(B)}_{k(b)}\right)\phi\left(\alpha^{(T)}_{j(W)} t + \alpha^{(T)}_{j(b)}\right)\right|$$

holds; where $m' = pm$, $\psi(\cdot)$, $\phi(\cdot)$, *features* s, *parameters* θ, *and integer* r *are defined as in the UATO; an overparametrization of the UATO can be reduced to a form that lets* $\alpha^{(R)}$ *be rank-2 tensors (compatible with Linear Regression).*

4. Dataset & OOD generalization

4 systems - 2 ODEs: Lorenz-63 and driven pendulum , 2 PDEs: Sine-Gordon: Sine-Gordon and Diffusion. First ones are parametrized, second ones are driven by Gaussian Random Fields (GRFs)

In-function-space dynamical drift:
Dynamics evolve within the same function space sampled during training. The trajectory data is no longer IID.

Out-of-function-space sampling:
Dynamics of the test set are not sampled from
the training distribution. The input function and trajectory data are no longer IID.

ExtremONet resilient against IFSDD when in phase space; generalizes from complex to less complex in the case of OOFSS.

5. Take-Aways

The **ExtremONet** is a **DeepONet alternative** that, on small datasets, Is faster to train, Generalizes better, Can generalize out of distribution (from complex to less complex), can function as self-correcting RNN in the case of IFSDD, and Is more data-efficient.

ELMs have **limitations**:

- No (effective) multi-layer possibilities
- Might not be able to learn too complex problems
- ELMs have no approximation guarantee.

[1] Data Analytics Laboratory, Vrije Universiteit Brussel
jari.beysen@gmail.com