Ensemble and Mixture-of-Experts DeepONets For Operator Learning

Ramansh Sharma, Varun Shankar

Kahlert School of Computing, University of Utah

The University of Manchester, April 21, 2025

Outline

- Operator Learning
- Deep Operator Network (DeepONet)
- Ensemble DeepONet
- Results
- Conclusion

Operator Learning

- Let $\mathcal{U}$ and $\mathcal{V}$ be two separable function spaces.
- G: $\mathcal{U}\ to\ \mathcal{V}$, is the general (potentially nonlinear) operator we are interested in learning.
- Data; $\{(u_i, v_i)\},\ i=1,...,N$, where $u_i \in \mathcal{U}$ are the input functions, and $v_i \in \mathcal{V}$ are the output functions.
- The approximation $\hat{G}:\mathcal{U} \times \Theta\ to\ \mathcal{V}$, where the parameters Θ are picked to minimize $|G - \hat{G}|$

Operator Learning

Examples:

- Derivative: $u(t) \to u'(t)$
- Laplacian: $u(x, y) \to u_{xx} + u_{yy}$
- Integral transform: $u(x, y) \to \int u(t)\, K(x, t)\, dt$

Operator Learning

Examples:

- Derivative: G: $u(t) \to u'(t)$
- Laplacian: G: $u(x, y) \to u_{xx} + u_{yy}$
- Integral transform: G: $u(x, y) \to \int u(t)\, K(x, t)\, dt$

Deep Operator Network (DeepONet)

DeepONets consist of two neural networks,

- **Branch**: Nonlinear encoding of the input functions; $\boldsymbol{\beta} : \mathbb{R}^{N_x} \to \mathbb{R}^p$
- **Trunk**: Nonlinear basis for the output functions; $\boldsymbol{\tau} : \mathbb{R}^{d_v} \to \mathbb{R}^p$

The DeepONet can be viewed as an p-dimensional inner product between the branch and the trunk:

$$\hat{G}(u)(y) = \langle \boldsymbol{\tau}(y), \boldsymbol{\beta}(u) \rangle + b_0$$

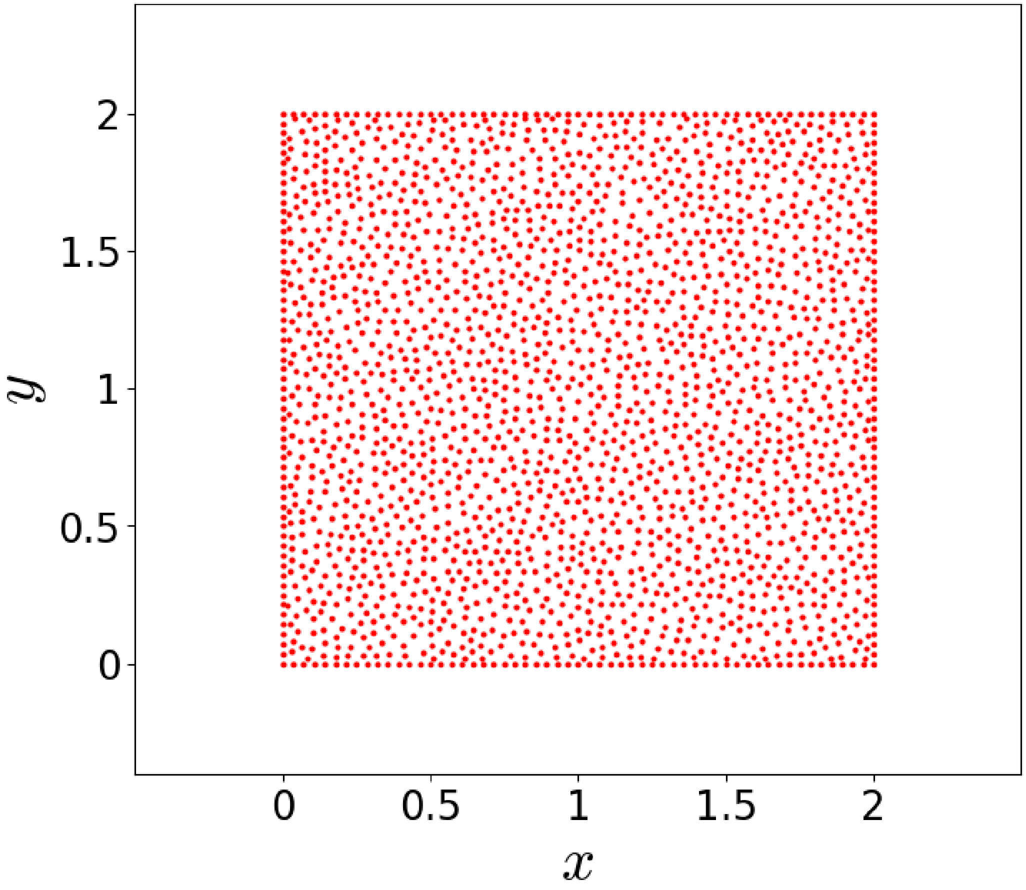
such that $\|v_i(y) - \hat{G}(u_i)(y)\|_2^2$ is minimized for all training function pairs.

Partition-of-Unity Mixture-of-Experts (PoU-MoE) Trunk

Partition-of-Unity Mixture-of-Experts (PoU-MoE) Trunk

The PoU-MoE trunk is motivated by the partition-of-unity approximation.

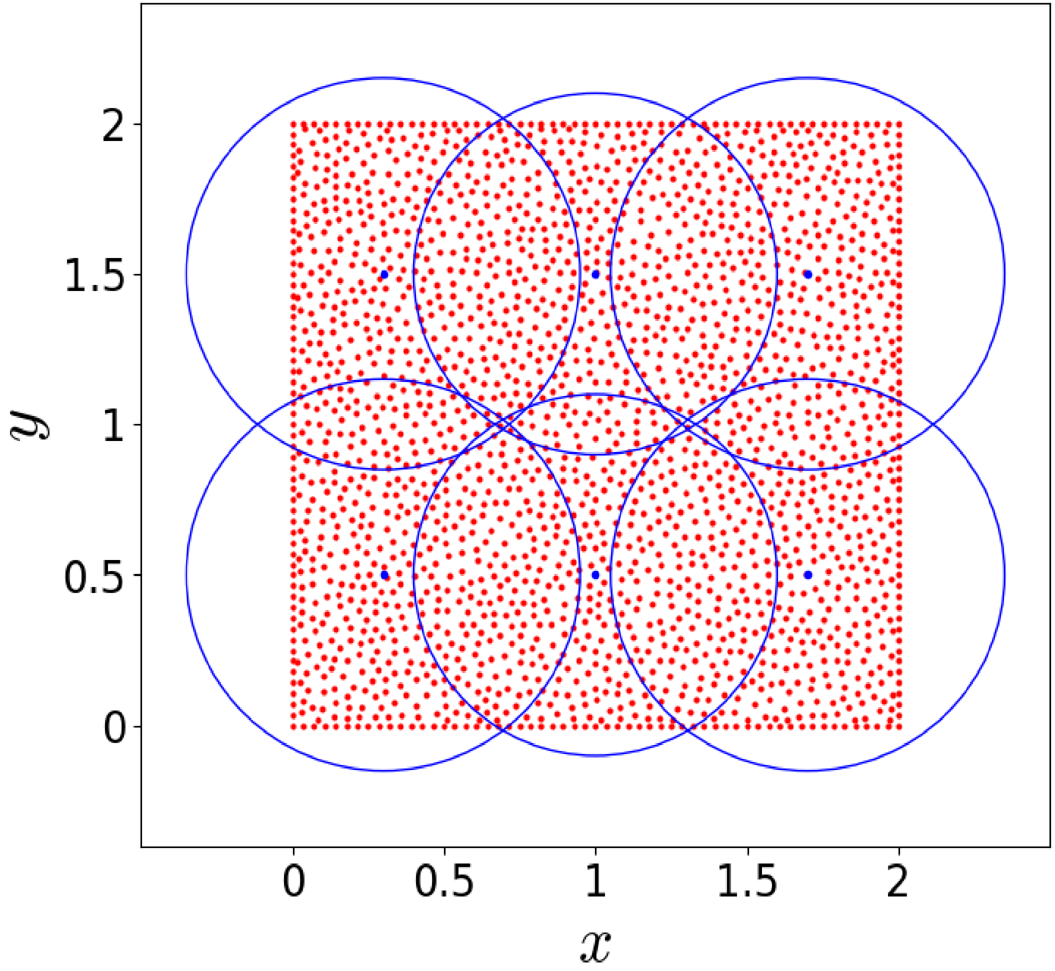
We partition the output function domain Ω into P overlapping spherical patches; $\Omega_k, k=1,...,P$.



Partition-of-Unity Mixture-of-Experts (PoU-MoE) Trunk

The PoU-MoE trunk is motivated by the partition-of-unity approximation.

We partition the output function domain Ω into P overlapping spherical patches; $\Omega_k, k=1,...,P$.



Partition-of-Unity Mixture-of-Experts (PoU-MoE) Trunk

- The key idea is to train a separate trunk network on each patch.
- Then, *blend* them together to produce one global trunk (to be used in the **ensemble**).
- The PoU-MoE trunk is written as,

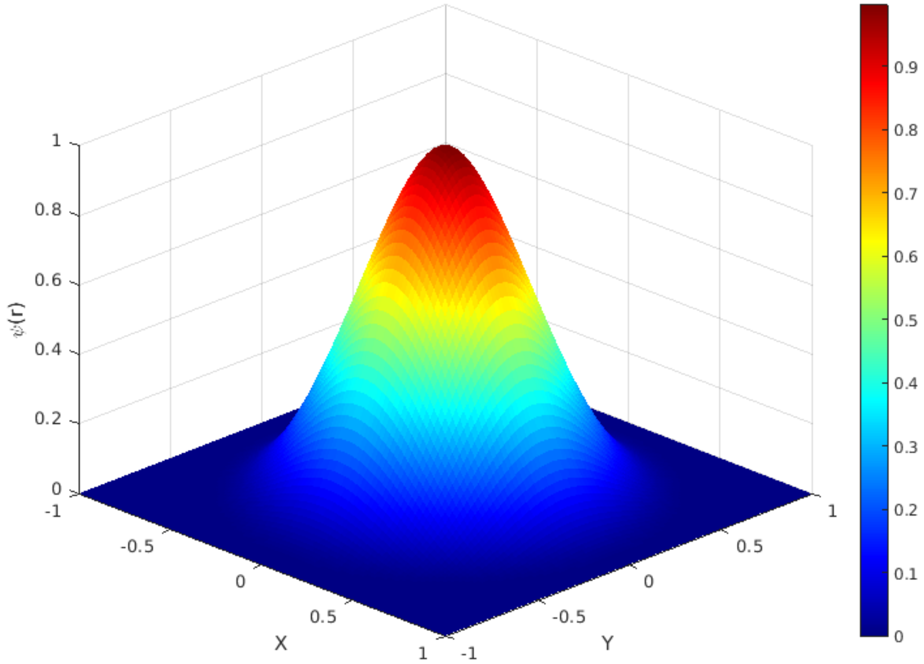
$$\boldsymbol{\tau}_{\text{PU}}(y) = \sum_{k=1}^{P} w_k(y)\boldsymbol{\tau}_k(y),$$

where the weight functions w_k are the compactly supported $\mathbb{C}^2\left(\mathbb{R}^3\right)$ Wendland kernel.

Partition-of-Unity Mixture-of-Experts (PoU-MoE) Trunk

Scaled and shifted Wendland kernel on a patch Ω_k is given by

$$\psi_k(y, y^c) = \psi_k\left(\frac{\|y - y_k^c\|}{\rho}\right) = \psi_k(r) \quad = (1-r)_+^4(4r+1).$$



Partition-of-Unity Mixture-of-Experts (PoU-MoE) Trunk

Scaled and shifted Wendland kernel on a patch Ω_k is given by

$$\psi_k(y, y^c) = \psi_k\left(\frac{\|y - y_k^c\|}{\rho}\right) = \psi_k(r) \quad = (1-r)_+^4(4r+1).$$

The weight functions w_k are then given by,

$$w_k(y) \quad = \frac{\psi_k(y)}{\sum_j \psi_j(y)}, \; k, j = 1, \ldots, P,$$

With the condition, $\sum_k w_k(y) = 1.$

Partition-of-Unity Mixture-of-Experts (PoU-MoE) Trunk

Each patch's trunk $\boldsymbol{\tau}_k$ can be viewed as a **spatially local "expert"**.

Partition-of-Unity Mixture-of-Experts (PoU-MoE) Trunk

Each patch's trunk $\boldsymbol{\tau}_k$ can be viewed as a **spatially local "expert"**.

Properties of $\boldsymbol{\tau}_{\text{PU}}$

- Is sparse in its experts $\boldsymbol{\tau}_k$.
- Constitutes a global set of basis functions.
- Is a universal approximator.

Proper Orthogonal Decomposition (POD) Trunk

The POD trunk uses the output functions' eigenvectors as a set of **global** basis functions.

$$\boldsymbol{\tau}_{\mathrm{POD}}(y) = \begin{bmatrix} \phi_1(y) & \phi_2(y) & \dots & \phi_p(y) \end{bmatrix},$$

In this work, we also use a **"Modified-POD"** trunk that includes the mean function ϕ_0 in the set of basis functions.

$$\boldsymbol{\tau}_{\mathrm{Modified\text{-}POD}}(y) = \begin{bmatrix} \phi_0(y) & \phi_1(y) & \dots & \phi_{p-1}(y) \end{bmatrix}.$$

Ensemble DeepONet

Ensemble DeepONet

Goal: Use both local and global basis functions in the DeepONet.

Ensemble DeepONet

Goal: Use both local and global basis functions in the DeepONet.

We propose the **ensemble trunk** which uses multiple types of basis functions.

Example, given three trunk networks, $\boldsymbol{\tau}_1, \boldsymbol{\tau}_2, \boldsymbol{\tau}_3$.

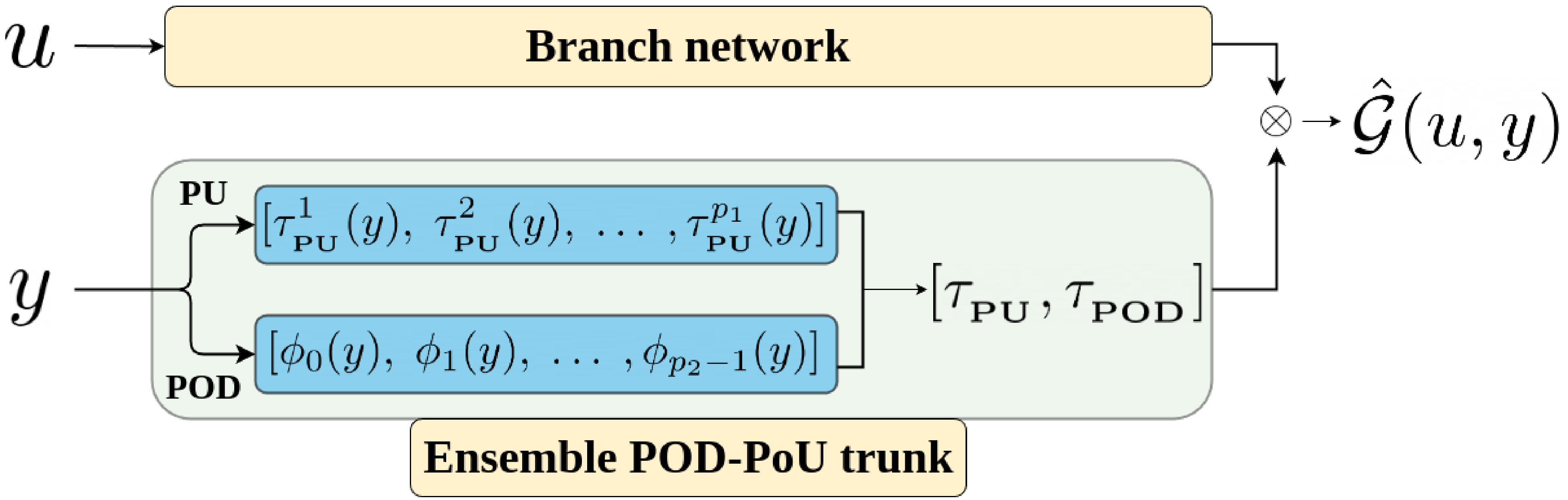
$$\hat{G}(u, y) = \left\langle \underbrace{[\boldsymbol{\tau}_1(y), \boldsymbol{\tau}_2(y), \boldsymbol{\tau}_3(y)]}_{\text{Ensemble trunk}}, \hat{\boldsymbol{\beta}}(u) \right\rangle + b_0,$$

where,

$$\boldsymbol{\tau}_1 : \mathbb{R}^{d_v} \to \mathbb{R}^{p_1}, \boldsymbol{\tau}_2 : \mathbb{R}^{d_v} \to \mathbb{R}^{p_2}, \boldsymbol{\tau}_3 : \mathbb{R}^{d_v} \to \mathbb{R}^{p_3}$$

$$\boldsymbol{\beta} : \mathbb{R}^{N_x} \to \mathbb{R}^{p_1+p_2+p_3}$$

Ensemble DeepONet



Ensemble architectures

What makes a good ensemble trunk?

Ensemble architectures

What makes a good ensemble trunk?

Combine trunks with different properties?

Ensemble architectures

What makes a good ensemble trunk?

- **Vanilla-POD**: Adding POD modes.
- **Vanilla-PoU**: Adding spatial locality (PoU-MoE).
- **POD-PoU**: Both POD global modes and PoU-MoE local expertise.
- **Vanilla-POD-PoU**: Adding a vanilla trunk (extra trainable parameters) to a POD-PoU ensemble.

Ensemble architectures

What makes a good ensemble trunk?

- **Vanilla-POD**: Adding POD modes.
- **Vanilla-PoU**: Adding spatial locality (PoU-MoE).
- **POD-PoU**: Both POD global modes and PoU-MoE local expertise.
- **Vanilla-POD-PoU**: Adding a vanilla trunk (extra trainable parameters) to a POD-PoU ensemble.
- $(P+1)$**-Vanilla**: Simple overparametrization. We use $P+1$ vanilla trunks in this model, where P is the number of PoU-MoE patches.

2D Reaction-Diffusion

$$
\begin{aligned}
\frac{\partial c}{\partial t} &= k_{\text{on}}\,(R - c)\,c_{\text{amb}} - k_{\text{off}}\,c + \nu \Delta c, \;\; y \in \Omega, \;\; t \in T, \\
\nu \frac{\partial c}{\partial n} &= 0, \;\; y \in \partial \Omega, \\
c(y,0) &\sim \mathcal{U}(0,1).
\end{aligned}
$$

$c_{\text{amb}}(y,t)$ is a background source of the chemical, k_{on} and k_{off} are constants.

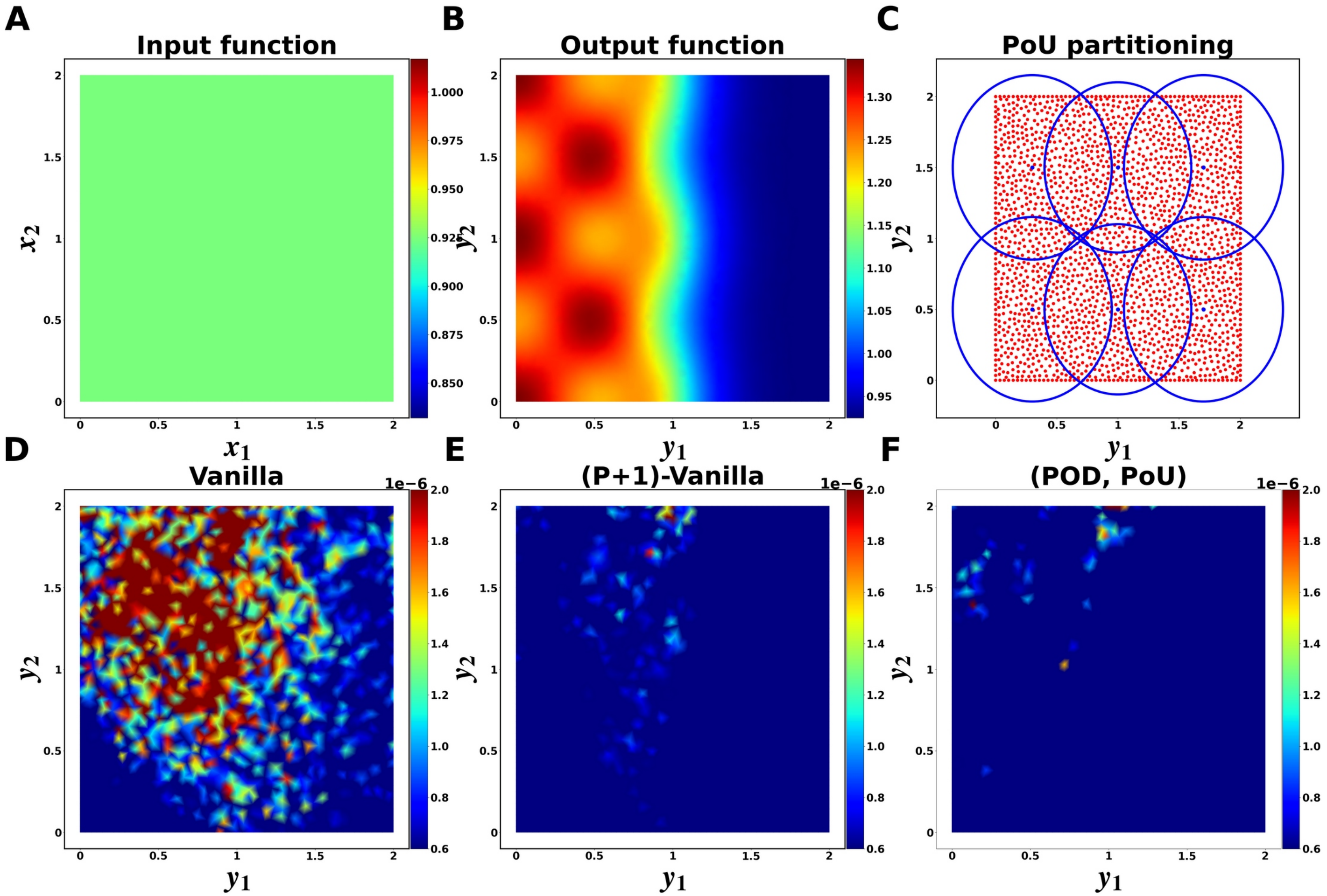
$\Omega = [0,2]^2$ and $T = [0, 0.5]$.

k_{on} and k_{off} chosen to introduce a sharp spatial discontinuity in the solution at $y_1 = 1$.

$$
k_{\text{on}} = \begin{cases} 2, & y_1 \leq 1.0, \\ 0, & \text{otherwise} \end{cases}, \qquad k_{\text{off}} = \begin{cases} 0.2, & y_1 \leq 1.0, \\ 0, & \text{otherwise} \end{cases}.
$$

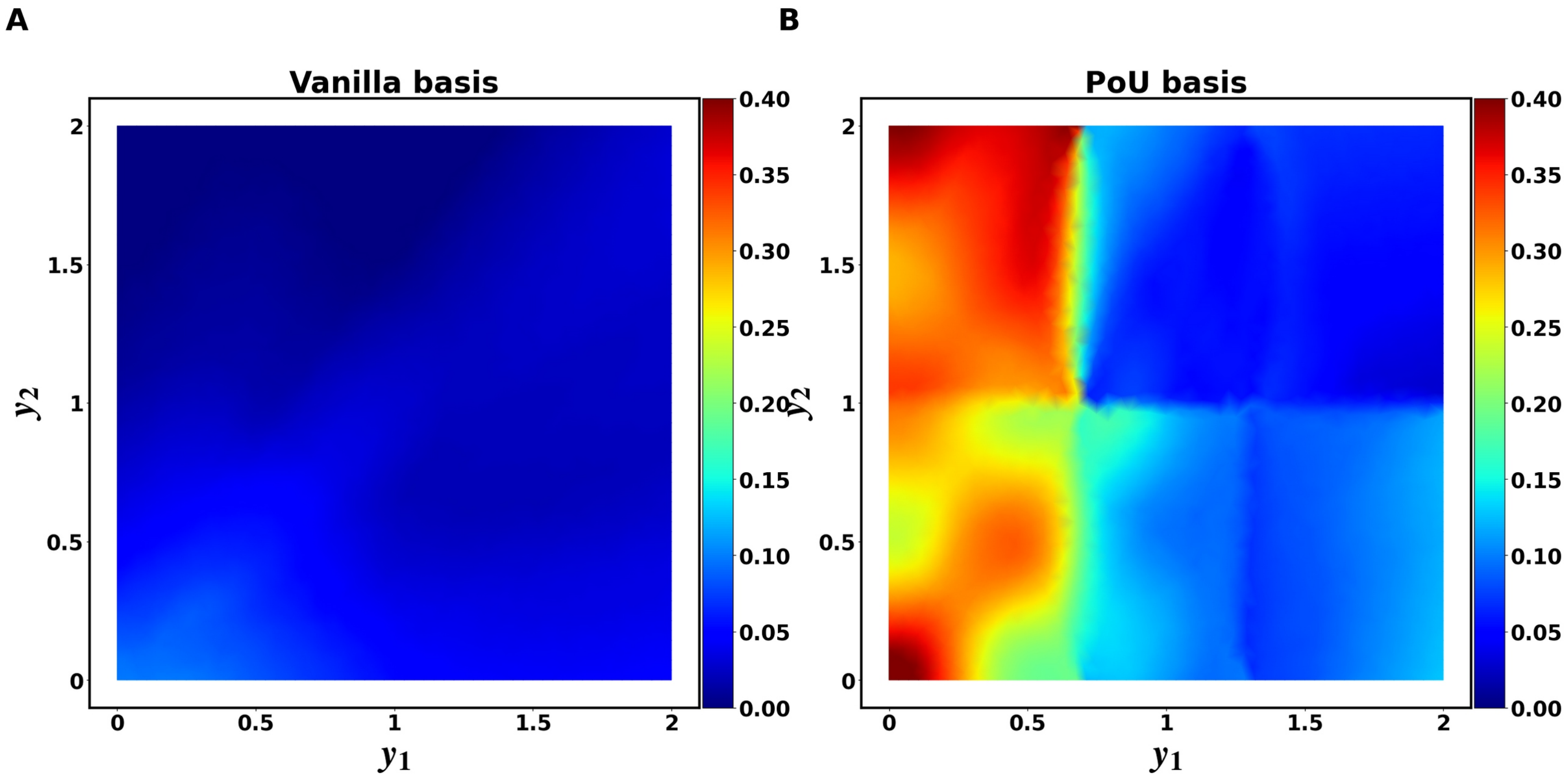
Goal: Learn the solution operator $G : c(y,0) \rightarrow c(y,0.5)$.

2D Reaction-Diffusion



	Vanilla	$(P+1)$-Vanilla	(POD, PoU)
Relative l_2 error	0.144 ± 0.01	0.0644 ± 0.02	$\mathbf{0.0539 \pm 4e-5}$

Spatial Locality



- Basis functions corresponding to the largest branch coefficients, i.e., the most “important” basis functions.
- The PoU basis spatially varies significantly more than the vanilla basis.
- The PoU-MoE trunk learns spatially local features, which improves accuracy.

3D Variable-Coefficient Reaction-Diffusion

$$\frac{\partial c}{\partial t} = k_{\text{on}}\left(R - c\right) c_{\text{amb}} - k_{\text{off}}\, c + \nabla \cdot \left(K(y) \nabla c\right), \; y \in \Omega, \; t \in T,$$
$$K(y)\frac{\partial c}{\partial n} = 0, y \in \partial\Omega,$$
$$c(y, 0) \sim \mathcal{U}(0, 1).$$

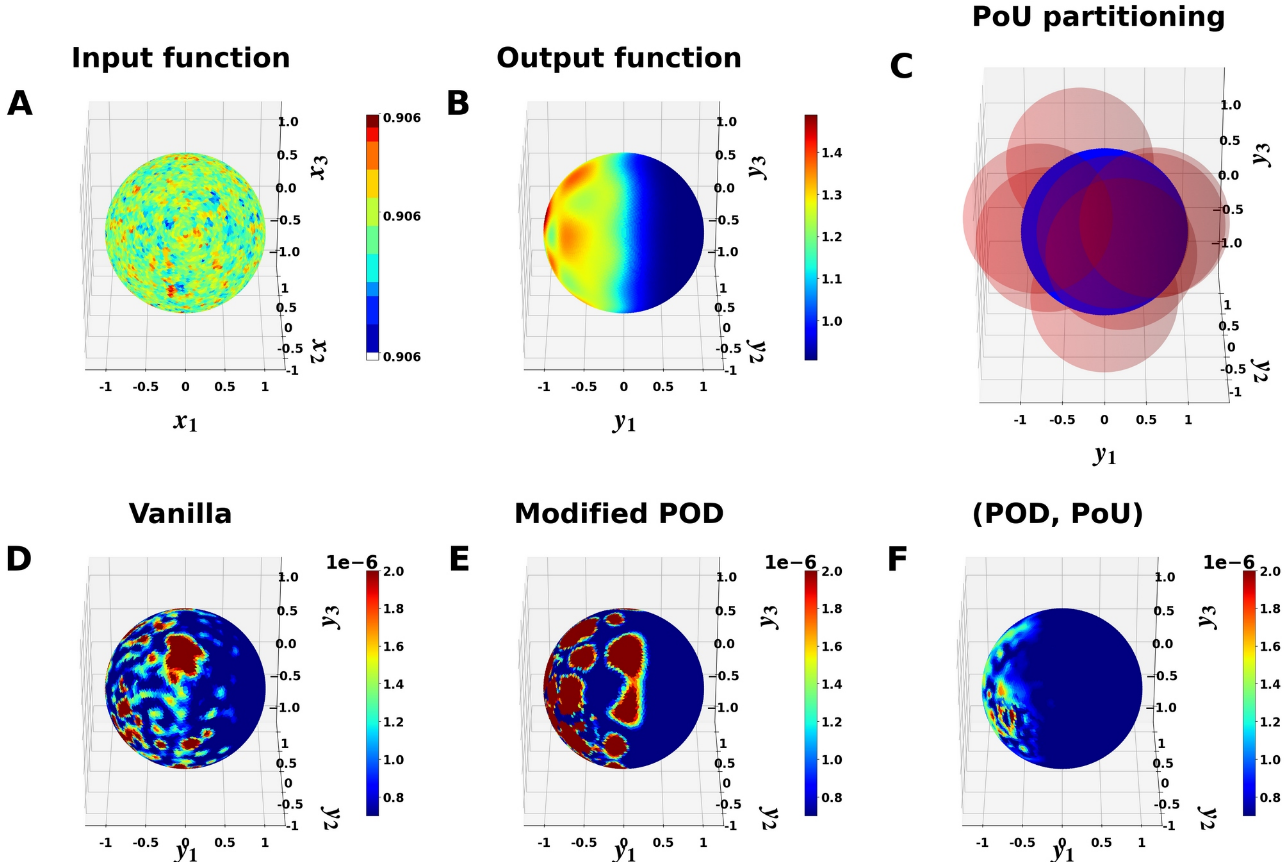
Ω was the unit ball, and $T = [0, 0.5]$.

Sharp point of discontinuity at $y_1 = 0$.

$K(y)$ was chosen to introduce steep gradients in the diffusion term.

Goal: Learn the solution operator $G : c(y,0) \to c(y,0.5)$.

3D Variable-Coefficient Reaction-Diffusion



	Vanilla	Modified-POD	(POD, PoU)
Relative l_2 error	0.127 ± 0.03	$0.155 \pm 4e-5$	$\mathbf{0.0576 \pm 0.05}$

2D Lid-driven Cavity Flow

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} = -\nabla \mathbf{p} + \nu \Delta \mathbf{u}, \quad \nabla \cdot \mathbf{u} = 0, \quad y \in \Omega, \quad t \in T,$$

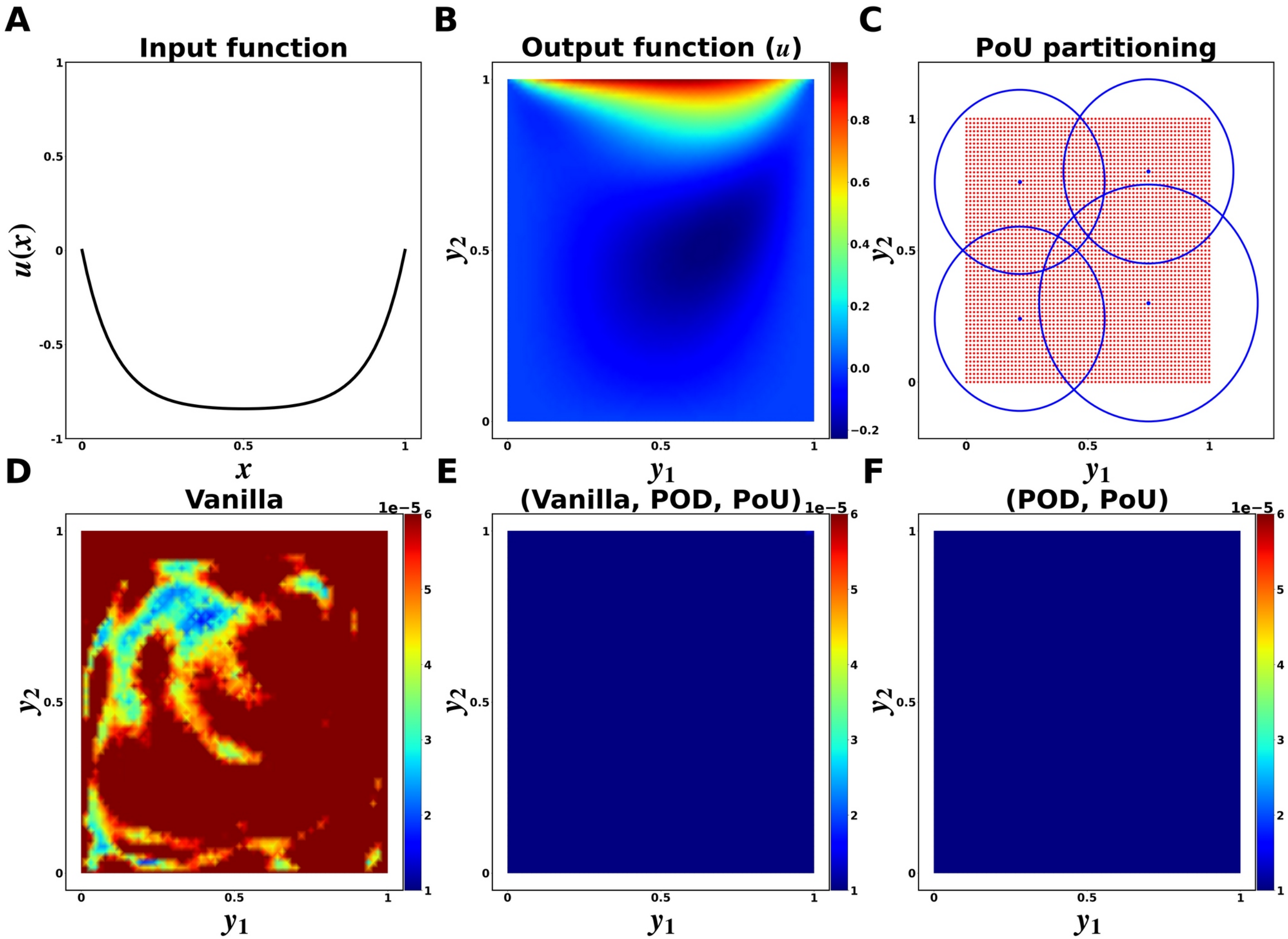
$$\mathbf{u} = \mathbf{u}_b,$$

Ω was set to $[0,1]^2$. The steady state boundary condition is

$$u_b = U\left(1 - \frac{\cosh\left(r(x - \frac{1}{2})\right)}{\cosh\left(\frac{r}{2}\right)}\right), \quad v_b = 0, r = 10.$$

Goal: Learn the solution operator $G : \mathbf{u}_b \to \mathbf{u}$.

2D Lid-driven Cavity Flow



	Vanilla	Vanilla-POD-PoU	(POD, PoU)
Relative l_2 error	5.53 ± 1.05	0.229 ± 0.01	$\mathbf{0.204 \pm 0.01}$

2D Darcy Flow

$$-\nabla \cdot (K(y)\, \nabla u(y)) = f(y), \;\; y \in \Omega,$$
$$u(y) \sim \mathcal{GP}\left(0, \mathcal{K}(y_1, y_1')\right),$$

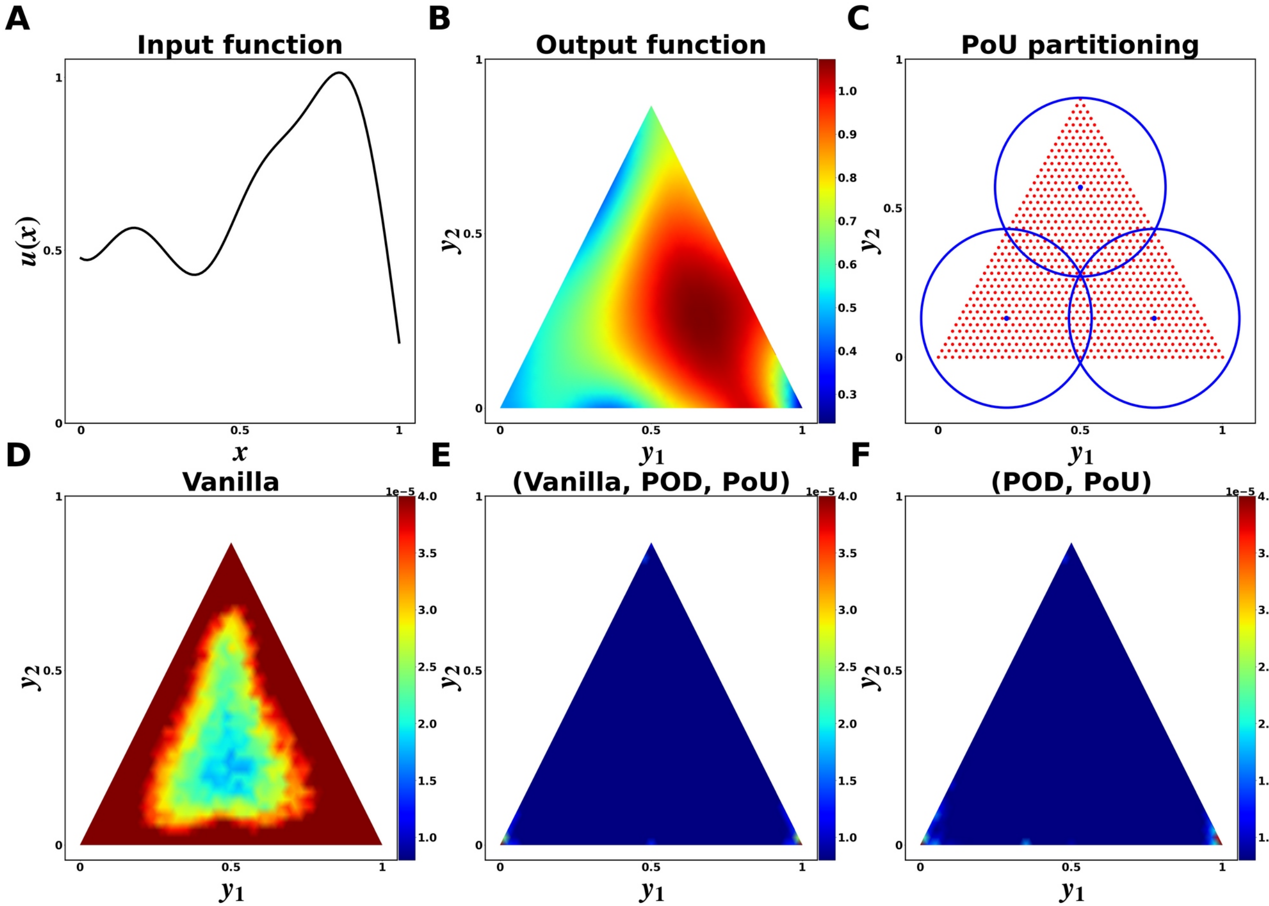
$K(y)$ is the permeability field.

$f(y) = -1.$

Ω was a triangular domain.

Goal: Learn the solution operator $G : u(y)|_{\partial\Omega} \to u(y)|_{\Omega}$.

2D Darcy Flow



	Vanilla	Vanilla-POD-PoU	(POD, PoU)
Relative l_2 error	0.857 ± 0.08	$\mathbf{0.187 \pm 0.02}$	0.204 ± 0.02

Insights

The big question - what makes a good ensemble trunk?

Insights

Trunk Choices	Darcy flow	Cavity flow	2D RD	3D RD
POD global modes	Yes	No	No	No
modified POD global modes	Yes	No	No	No
Adding POD global modes	Yes	Yes	Yes	No
Adding spatial locality	No	Yes	Yes	No
Only POD global modes + spatial locality	Yes	**Yes**	**Yes**	**Yes**
Only POD global modes + spatial locality + vanilla trunk	**Yes**	Yes	Yes	No
Adding excessive overparametrization	No	Yes	Yes	No

Yes/no refers to whether the strategy outpeforms a vanilla-DeepONet.

Conclusion

- The ensemble DeepONet, a method of enriching the basis functions of the DeepONet.
- The POD-PoU ensemble consistently beats the vanilla-DeepONet across all problems (**2-4x accuracy improvement**).
- Simple overparametrization $((P+1)$ -Vanilla DeepONet) is not enough and sometimes deteriorates accuracy; **a judicial combination of localized and global basis functions is vital**.
- The novel PoU-MoE trunk captures spatially local features.
- The PoU-MoE trunk brings expressivity in problems with steep gradients in either the input or output functions.

Future work

- Extend PoU-MoE to adaptive partitioning strategies (trainable patch centers and patch radii, trainable patch shape).
- Ensemble learning for other neural operators (FNO, GNO, etc.).

Thank You!

Ramansh Sharma and Varun Shankar. “**Ensemble and Mixture-of-Experts DeepONets for Operator Learning**”. Transactions on Machine Learning Research, March 2025. arxiv.org/abs/2405.11907.

3D Variable-Coefficient Reaction-Diffusion

K(y) was chosen to have steep gradients and defined as,

$$K(y) = B + \frac{C}{\tanh(A)}\left((A-3)\tanh(8y_1-5) - (A-15)\tanh(8y_1+5) + A\tanh(A)\right),$$

where A=9, B=0.0215, C=0.005.

Other results

Relative l_2 errors (as percentage) on the test dataset. RD stands for reaction-diffusion.

	Darcy flow	Cavity flow	2D RD	3D RD
Vanilla	0.857 ± 0.08	5.53 ± 1.05	0.144 ± 0.01	0.127 ± 0.03
POD	0.297 ± 0.01	$7.94 \pm 2e-5$	$5.06 \pm 8e-7$	9.40 ± 8
Modified-POD	0.300 ± 0.04	$7.93 \pm 2e-5$	$0.131 \pm 4e-5$	$0.155 \pm 4e-5$
(Vanilla, POD)	0.227 ± 0.03	0.310 ± 0.03	$0.0751 \pm 4e-5$	5.24 ± 10.4
$(P+1)$-Vanilla	1.19 ± 0.06	2.17 ± 0.3	0.0644 ± 0.02	5.25 ± 10.3
(Vanilla, PoU)	0.976 ± 0.03	1.06 ± 0.05	0.0946 ± 0.03	5.25 ± 10.3
(POD, PoU)	0.204 ± 0.02	$\mathbf{0.204 \pm 0.01}$	$\mathbf{0.0539 \pm 4e-5}$	$\mathbf{0.0576 \pm 0.05}$
(Vanilla, POD, PoU)	$\mathbf{0.187 \pm 0.02}$	0.229 ± 0.01	$0.0666 \pm 8e-5$	5.22 ± 10.4

Runtime results

Average time per training epoch in seconds. RD stands for reaction-diffusion.

	Darcy flow	Cavity flow	2D RD	3D RD
Vanilla	$8.93e-4$	$3.99e-4$	$2.97e-4$	$2.10e-4$
POD	$5.19e-4$	$2.46e-4$	$2.06e-4$	$1.22e-4$
Modified-POD	$6.86e-4$	$2.49e-4$	$2.08e-4$	$1.22e-4$
(Vanilla, POD)	$9.80e-4$	$3.92e-4$	$3.03e-4$	$2.32e-4$
$(P+1)$-Vanilla	$1.10e-3$	$8.51e-4$	$7.27e-4$	$9.45e-4$
Vanilla-PoU	$8.67e-4$	$9.52e-4$	$1.03e-3$	$1.39e-3$
POD-PoU	$6.74e-4$	$8.21e-4$	$9.24e-4$	$1.28e-3$
Vanilla-POD-PoU	$8.55e-4$	$9.48e-4$	$1.05e-3$	$1.43e-3$

Runtime results

Inference time on the test dataset in seconds. RD stands for reaction-diffusion.

	Darcy flow	Cavity flow	2D RD	3D RD
Vanilla	$1.66e-4$	$1.39e-4$	$1.32e-4$	$7.20e-5$
POD	$1.57e-4$	$1.12e-4$	$1.12e-4$	$6.42e-5$
Modified-POD	$1.34e-4$	$1.08e-4$	$9.94e-5$	$6.62e-5$
(Vanilla, POD)	$1.69e-4$	$1.33e-4$	$1.20e-4$	$7.76e-5$
$(P+1)$-Vanilla	$2.08e-4$	$2.12e-4$	$1.71e-4$	$1.48e-4$
Vanilla-PoU	$1.91e-4$	$2.42e-4$	$2.21e-4$	$2.37e-4$
POD-PoU	$1.63e-4$	$1.94e-4$	$1.96e-4$	$2.30e-4$
Vanilla-POD-PoU	$2.00e-4$	$2.18e-4$	$2.28e-4$	$2.41e-4$

Universal Approximation Theorem - PoU-MoE Trunk

Theorem

Let $\mathcal{G} : \mathcal{U} \to \mathcal{V}$ *be a continuous operator. Define* $\mathcal{G}^\dagger$ *as*
$\mathcal{G}^\dagger(u)(y) = \left\langle \beta(u;\theta_b), \sum_{j=1}^{P} w_j(y)\boldsymbol{\tau}_j(y;\theta_{\boldsymbol{\tau}_j}) \right\rangle + b_0$, *where* $\boldsymbol{\beta} : \mathbb{R}^{N_x} \times \Theta_{\boldsymbol{\beta}} \to \mathbb{R}^p$ *is a branch network embedding the input function u,* $\boldsymbol{\tau}_j : \mathbb{R}^{d_v} \times \Theta_{\boldsymbol{\tau}_j} \to \mathbb{R}^p$ *are trunk networks,* b_0 *is a bias, and* $w_j : \mathbb{R}^{d_v} \to \mathbb{R}$ *are compactly-supported, positive-definite weight functions that satisfy the partition of unity condition* $\sum_j w_j(y) = 1, j = 1, \ldots, P$*. Then* $\mathcal{G}^\dagger$ *can approximate* $\mathcal{G}$ *globally to any desired accuracy, i.e.,*

$$\mathcal{G}(u)(y) - \mathcal{G}^\dagger(u)(y)\|_{\mathcal{V}} \leq \epsilon,$$

where $\epsilon > 0$ *can be made arbitrarily small.*

Universal Approximation Theorem - PoU-MoE Trunk

Proof

$$
\begin{aligned}
\|\mathcal{G}(u)(y) - \mathcal{G}^{\dagger}(u)(y)\|_{\mathcal{V}} &= \left\| \mathcal{G}(u)(y) - \left\langle \boldsymbol{\beta}(u;\theta_b), \sum_{j=1}^{P} w_j(y) \boldsymbol{\tau}_j(y;\theta_{\boldsymbol{\tau}_j}) \right\rangle - b_0 \right\|_{\mathcal{V}}, \\
&= \left\| \underbrace{\left(\sum_{j=1}^{P} w_j(y)\right)}_{=1} \mathcal{G}(u)(y) - \left\langle \boldsymbol{\beta}(u;\theta_b), \sum_{j=1}^{P} w_j(y) \boldsymbol{\tau}_j(y;\theta_{\boldsymbol{\tau}_j}) \right\rangle \right. \\
&\qquad \left. - \underbrace{\left(\sum_{j=1}^{P} w_j(y)\right)}_{=1} b_0 \right\|_{\mathcal{V}}, \\
&= \left\| \sum_{j=1}^{P} w_j(y) \left(G(u)(y) - \left\langle \boldsymbol{\beta}(u;\theta_b), \boldsymbol{\tau}_j(y;\theta_{\boldsymbol{\tau}_j}) \right\rangle - b_0 \right) \right\|_{\mathcal{V}}, \\
&\leq \sum_{j=1}^{P} w_j(y) \| \mathcal{G}(u)(y) - \left\langle \boldsymbol{\beta}(u;\theta_b), \boldsymbol{\tau}_j(y;\theta_{\boldsymbol{\tau}_j}) \right\rangle - b_0 \|_{\mathcal{V}}.
\end{aligned}
$$

Universal Approximation Theorem - PoU-MoE Trunk

Given a branch network $\boldsymbol{\beta}$ that can approximate functionals to arbitrary accuracy, the (generalized) universal approximation theorem for operators automatically implies that a trunk network $\boldsymbol{\tau}_j$ (given sufficient capacity and proper training) can approximate the restriction of $\mathcal{G}$ to the support of $w_i(\mathbf{y})$ such that:

$$\|\mathcal{G}(u)(y) - \langle \boldsymbol{\beta}(u;\theta_b), \boldsymbol{\tau}_j(y;\theta_{\boldsymbol{\tau}_j})\rangle - b_0\|_{\mathcal{V}} \leq \epsilon_j,$$

for all y in the support of w_j and any $\epsilon_j > 0$. Setting $\epsilon_j = \epsilon$, $j = 1, \ldots, P$, we obtain:

$$\|\mathcal{G}(u)(y) - \mathcal{G}^\dagger(u)(y)\|_{\mathcal{V}} \leq \epsilon \underbrace{\sum_{j=1}^{P} w_i(y)}_{=1},$$

$$\implies \|\mathcal{G}(u)(y) - \mathcal{G}^\dagger(u)(y)\|_{\mathcal{V}} \leq \epsilon.$$

where $\epsilon > 0$ can be made arbitrarily small. This completes the proof.

Universal Approximation Theorem - Ensemble Trunk

Theorem

Let $\mathcal{G} : \mathcal{U} \to \mathcal{V}$ *be a continuous operator. Define* $\hat{\mathcal{G}}$ *as* $\hat{\mathcal{G}}(u, y) = \left\langle \hat{\boldsymbol{\tau}}(y; \theta_{\boldsymbol{\tau}_1}; \theta_{\boldsymbol{\tau}_2}; \theta_{\boldsymbol{\tau}_3}), \hat{\boldsymbol{\beta}}(u; \theta_b) \right\rangle + b_0$, *where* $\hat{\boldsymbol{\beta}} : \mathbb{R}^{N_x} \times \Theta_{\hat{\boldsymbol{\beta}}} \to \mathbb{R}^{p_1+p_2+p_3}$ *is a branch network embedding the input function* u, b_0 *is the bias, and* $\hat{\boldsymbol{\tau}} : \mathbb{R}^{d_v} \times \Theta_{\hat{\boldsymbol{\tau}}_1} \times \Theta_{\hat{\boldsymbol{\tau}}_2} \times \Theta_{\hat{\boldsymbol{\tau}}_3} \to \mathbb{R}^{p_1+p_2+p_3}$ *is an ensemble trunk network. Then* $\hat{\mathcal{G}}$ *can approximate* $\mathcal{G}$ *globally to any desired accuracy, i.e.,*

$$\|\mathcal{G}(u)(y) - \hat{\mathcal{G}}(u)(y)\|_{\mathcal{V}} \leq \epsilon,$$

where $\epsilon > 0$ *can be made arbitrarily small.*

Proof.

This follows from the (generalized) universal approximation theorem[a] which holds for arbitrary branches and trunks.

Ensemble FNO

FNOs consist of a *lifting* operator, a *projection* operator, and intermediate Fourier layers consisting of kernel-based integral operators.

f_t denotes the intermediate function at the t^{th} Fourier layer. Then, f_{t+1} is given by

$$f_{t+1}(y) \quad = \sigma \left(\int_\Omega \mathcal{K}(x,y)\, f_t(x)\, dx \;+\; W\; f_t(y) \right),\; x \in \Omega,$$

where σ is an activation function, $\mathcal{K}$ is a matrix-valued kernel, and W is the pointwise

convolution.

This is a projection of $f_t(x)$ onto a set of *global* Fourier modes. Incorporating a set of localized basis functions in an ensemble FNO using the PoU-MoE formulation:

$$f_{t+1}(y) \quad = \sigma \left(\underbrace{\int_\Omega \mathcal{K}(x,y) f_t(x)\, dx}_{\text{Global basis}} + \underbrace{\sum_{k=1}^{P} w_k(y) \int_{\Omega_k} \mathcal{K}(x,y)\; f_t(x)|_{\Omega_k}\; dx}_{\text{Localized basis}} + W f_t(y) \right),$$

The PoU-MoE formulation now combines a set of *localized* integrals, each of which is a projection of f_t onto a local Fourier basis.