DeepONet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators

Lu Lu

joint work with P. Jin & G. Karniadakis

Division of Applied Mathematics, Brown University

Joint Mathematics Meetings
January 15, 2020

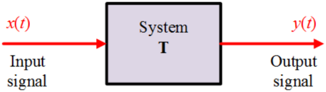
From function to operator

- Function: $\mathbb{R}^{d_1} \to \mathbb{R}^{d_2}$

  e.g., image classification:  $\mapsto 5$

- Operator: function (∞-dim) $\mapsto$ function (∞-dim)
  e.g., derivative (local): $x(t) \mapsto x'(t)$
  e.g., integral (global): $x(t) \mapsto \int K(s,t)x(s)ds$

  e.g., dynamic system:



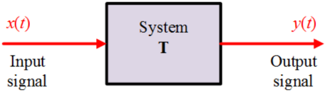
From function to operator

- Function: $\mathbb{R}^{d_1} \to \mathbb{R}^{d_2}$

  e.g., image classification:  $\mapsto 5$

- Operator: function (∞-dim) $\mapsto$ function (∞-dim)
  e.g., derivative (local): $x(t) \mapsto x'(t)$
  e.g., integral (global): $x(t) \mapsto \int K(s,t)x(s)ds$

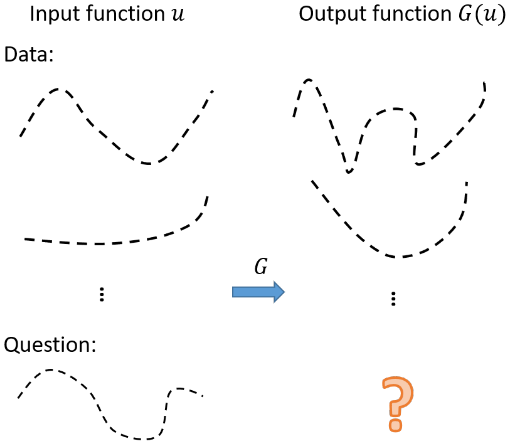
  e.g., dynamic system:



  $\Rightarrow$ Can we learn operators via neural networks?
  $\Rightarrow$ How?

Problem setup

$G : u \mapsto G(u)$
$G(u) : y \in \mathbb{R}^d \mapsto G(u)(y) \in \mathbb{R}$



Universal Approximation Theorem for Operator

$G : u \mapsto G(u)$
$G(u) : y \in \mathbb{R}^d \mapsto G(u)(y) \in \mathbb{R}$

Theorem (Chen & Chen, 1995)

Suppose that σ is a continuous non-polynomial function, X is a Banach Space, $K_1 \subset X$, $K_2 \subset \mathbb{R}^d$ are two compact sets in X and $\mathbb{R}^d$, respectively, V is a compact set in $C(K_1)$, G is a continuous operator, which maps V into $C(K_2)$. Then for any $\epsilon > 0$, there are positive integers n, p, m, constants $c_i^k, \xi_{ij}^k, \theta_i^k, \zeta_k \in \mathbb{R}$, $w_k \in \mathbb{R}^d$, $x_j \in K_1$, $i = 1, \ldots, n$, $k = 1, \ldots, p$, $j = 1, \ldots, m$, such that

$$\left| G(u)(y) - \sum_{k=1}^{p} \sum_{i=1}^{n} c_i^k \sigma \left(\sum_{j=1}^{m} \xi_{ij}^k u(x_j) + \theta_i^k \right) \sigma(w_k \cdot y + \zeta_k) \right| < \epsilon$$

holds for all $u \in V$ and $y \in K_2$.

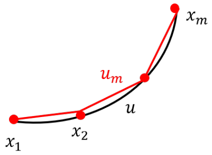
BROWN

Convergence w.r.t. the number of sensors

Consider $G : u(x) \mapsto \boldsymbol{s}(x)$ $(x \in [0,1])$ by ODE system

$$\frac{d}{dx}\boldsymbol{s}(x) = \boldsymbol{g}(\boldsymbol{s}(x), u(x), x), \quad \boldsymbol{s}(0) = \boldsymbol{s_0}$$

$$\forall u \in V \Rightarrow u_m \in V_m$$



Let $\kappa(m, V) := \sup_{u \in V} \max_{x \in [0,1]} |u(x) - u_m(x)|$

e.g., Gaussian process with kernel $e^{-\frac{\|x_1 - x_2\|^2}{2l^2}}$: $\kappa(m, V) \sim \frac{1}{m^2 l^2}$

Theorem (Lu et al., 2019; informal)

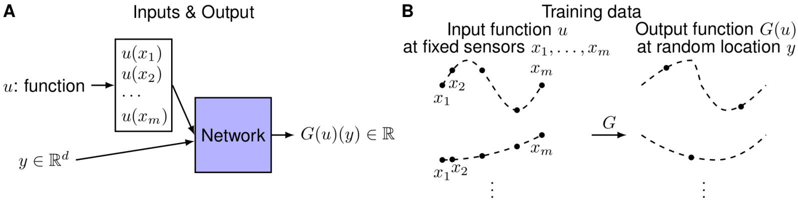
There exists a constant C, such that for any y,

$$\sup_{u \in V} \|G(u)(y) - NN(u(x_1), \dots, u(x_m), y)\|_2 < C\kappa(m, V).$$

Problem setup

$G : u \mapsto G(u)$
$G(u) : y \in \mathbb{R}^d \mapsto G(u)(y) \in \mathbb{R}$

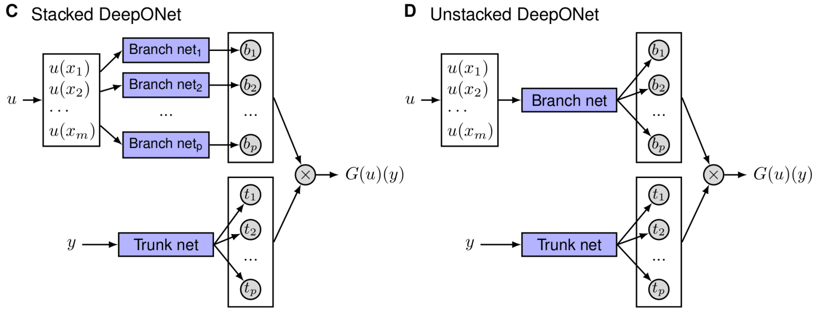
- Inputs: u at sensors $\{x_1, x_2, \ldots, x_m\}$, $y \in \mathbb{R}^d$
- Output: $G(u)(y) \in \mathbb{R}$



Next, how to design the network?

Lu et al., arXiv:1910.03193, 2019

Deep operator network (DeepONet)



$$G(u)(y) \approx \sum_{k=1}^{p} b_k(u) \cdot t_k(y)$$

Ideas:

- Prior knowledge: u and y are independent
- $G(u)(y)$: a function of y conditioning on u
  - $t_k(y)$: basis functions of y
  - $b_k(u)$: u-dependent coefficients

Lu et al., arXiv:1910.03193, 2019

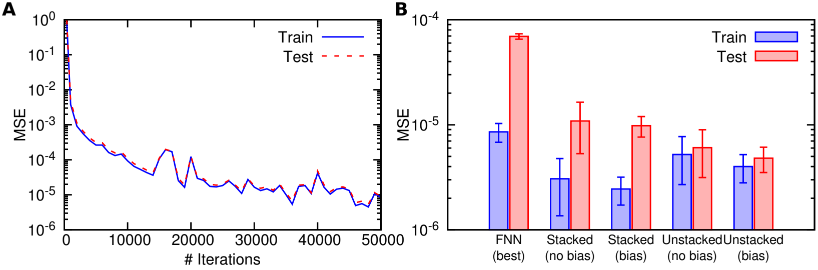
A simple ODE case

$$\frac{ds(t)}{dt} = u(t), \quad t \in [0,1],$$

with an initial condition $s(0) = 0$.

$$G : u(x) \mapsto s(y) = \int_0^y u(\tau) d\tau$$

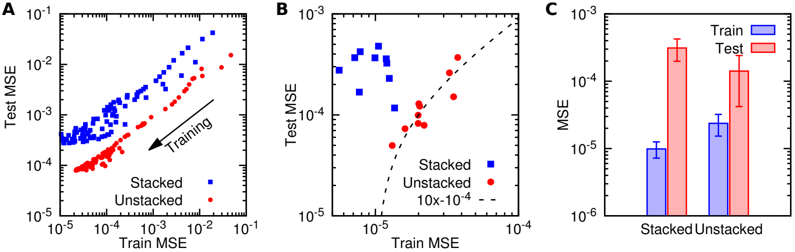
Very small generalization error!



Lu et al., arXiv:1910.03193, 2019

A nonlinear ODE case

$$\frac{ds(x)}{dx} = -s^2(x) + u(x)$$



Linear correlation between training and test errors

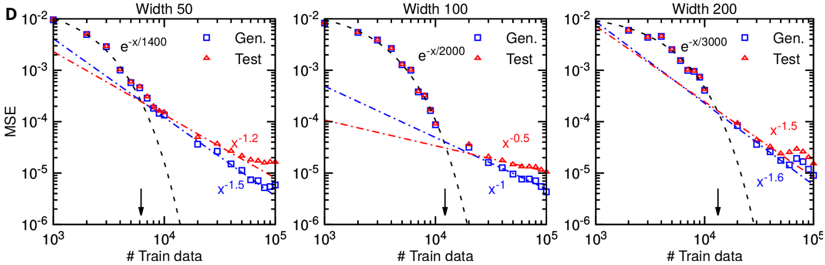
- A: in one training process
- B: across multiple runs (random dataset and network initialization)

Lu et al., arXiv:1910.03193, 2019

Gravity pendulum with an external force $u(t)$

$$\frac{ds_1}{dt} = s_2, \quad \frac{ds_2}{dt} = -k\sin s_1 + u(t)$$

$G : u(x) \mapsto \mathbf{s}(x)$



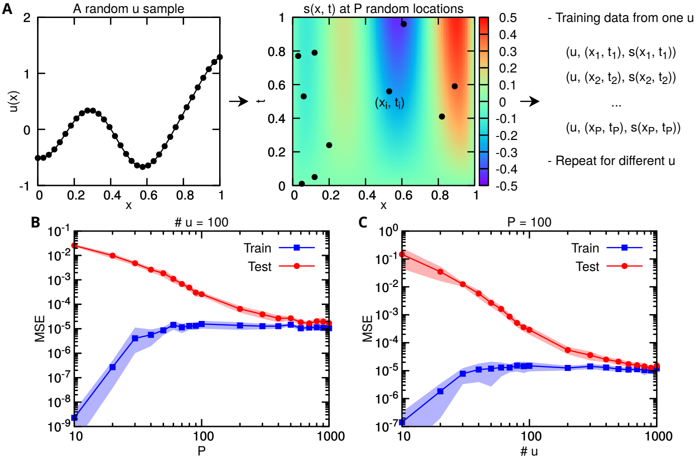
Test/generalization error:

- small dataset: exponential convergence
- large dataset: polynomial rates
- smaller network has earlier transition point

Lu et al., arXiv:1910.03193, 2019

Diffusion-reaction system

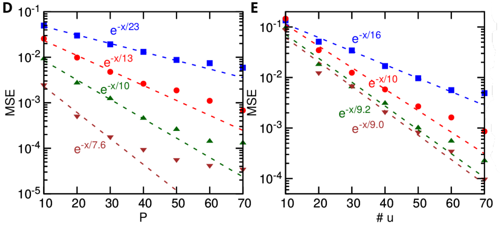
$$\frac{\partial s}{\partial t} = D\frac{\partial^2 s}{\partial x^2} + ks^2 + u(x), \quad x \in [0,1], t \in [0,1]$$



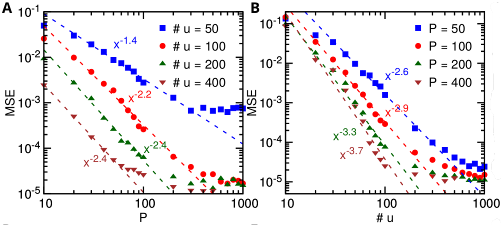
Lu et al., arXiv:1910.03193, 2019

Diffusion-reaction system

Exponential convergence



Polynomial convergence



Lu et al., arXiv:1910.03193, 2019

Advection-diffusion system

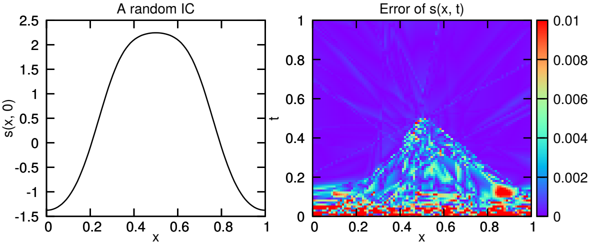
$$\frac{\partial s}{\partial t} + \frac{\partial s}{\partial x} - \frac{\partial^2 s}{\partial x^2} = 0 \quad \text{or} \quad \frac{\partial s}{\partial t} - 1.5({}^{RL}_{-\infty}D^{1.5}_x)s = 0$$

$x \in [0, 1]$, $t \in [0, 1]$, periodic BC

$G : u(x) = s(x, 0) \mapsto s(x, t)$

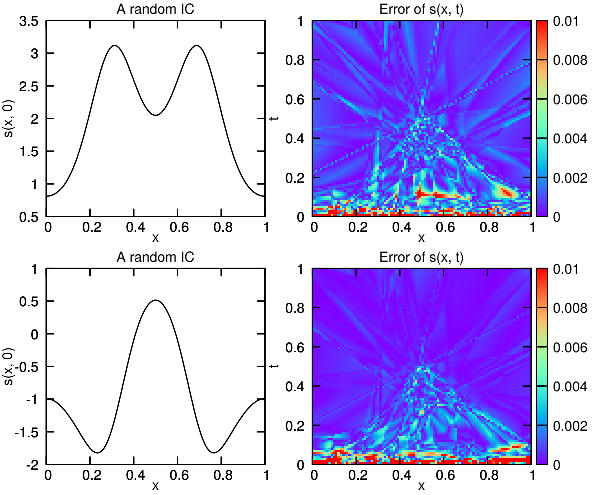
- 100 u sensors
- training data: # IC = 1000, 100 random points of $s(x, t)$ for each IC

Prediction for a new random IC:



Advection-diffusion system

More predictions:



Summary

- Number of sensors, $\kappa(m, V)$
- DeepONet
  - 1D ODE (linear, nonlinear), gravity pendulum, diffusion-reaction system (nonlinear), advection-diffusion system
  - Small generalization error
  - Exponential/polynomial error convergence

- Lu, Jin, & Karniadakis, arXiv:1910.03193, 2019.

BROWN