# Neural operators

ARTICLES
https://doi.org/10.1038/s42256-021-00302-5
nature machine intelligence
Check for updates

## Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators

Lu Lu[1], Pengzhan Jin[2,3], Guofei Pang[2], Zhongqiang Zhang[4] and George Em Karniadakis[2]✉

Published as a conference paper at ICLR 2021

## Fourier Neural Operator for Parametric Partial Differential Equations

**Zongyi Li**
zongyili@caltech.edu

**Nikola Kovachki**
nkovachki@caltech.edu

**Kamyar Azizzadenesheli**
kamyar@purdue.edu

**Burigede Liu**
bgl@caltech.edu

**Kaushik Bhattacharya**
bhatta@caltech.edu

**Andrew Stuart**
astuart@caltech.edu

**Anima Anandkumar**
anima@caltech.edu

arXiv:2010.08895v3 [cs.LG] 17 May 2021

PINNs: $\Delta u = f$ on $\Omega$ $\quad u = u_0$ on $\partial\Omega$

find $u$ $\longrightarrow$ $u(x) = u_\theta(x)$

NN

$x \to \boxed{NN} \to u_\theta(x)$

Neural ops: solve PDE as parametrized by $f, u_0$

$f, u_0 \longrightarrow \boxed{NN} \longrightarrow u$

## Operators

map function → function $\qquad u, v : \mathbb{R} \to \mathbb{R}$

$$u(x) \longmapsto \begin{matrix} v(y) \\ v(x) \end{matrix}$$

e.g.

$$\frac{d}{dx} : \quad x^2 \longmapsto 2x$$

$$\int : \quad \cos x \longmapsto \sin x$$

PDE : $\quad f \longmapsto$ solution of $\;\Delta v = f \quad v = 0$ on $\partial\Omega$
$\qquad\qquad\qquad\quad v$

## Neural ops

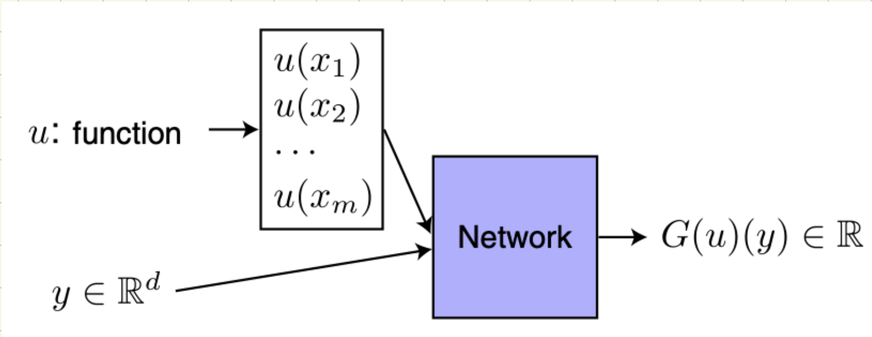
$$u \longrightarrow \underset{NN}{\boxed{G}} \longrightarrow G(u)$$

$$G(u)(y) \in \mathbb{R}$$

$$u \longrightarrow \boxed{NN} \longrightarrow G(u) = v$$

$$\begin{tikzcd} u \arrow[dr] & & \\ & \boxed{NN} \arrow[r] & G(u)(y) \in \mathbb{R} \\ y \in \mathbb{R} \arrow[ur] & & \end{tikzcd}$$

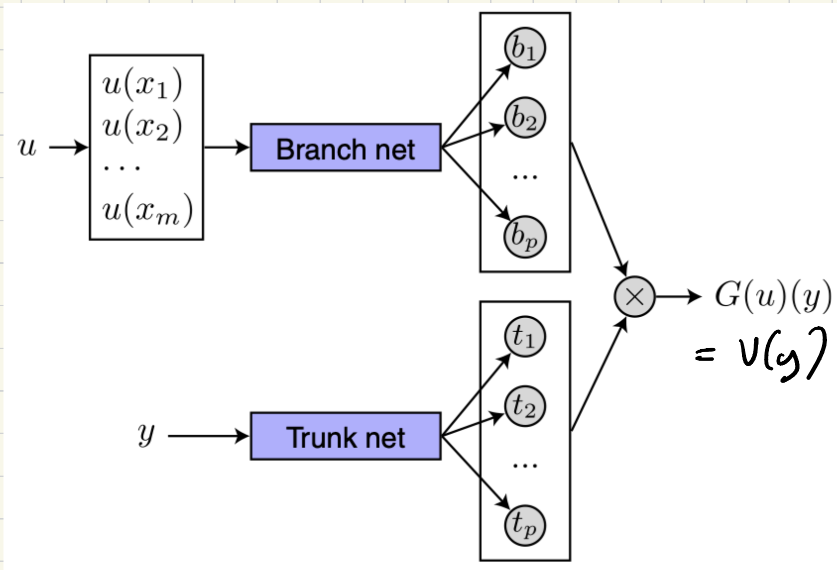
$$\begin{tikzcd} \begin{bmatrix} u(x_1) \\ u(x_2) \\ \vdots \\ u(x_n) \end{bmatrix} \in \mathbb{R}^n \arrow[dr] & & \\ & \boxed{NN} \arrow[r] & G(u)(y) \in \mathbb{R} \\ y \in \mathbb{R} \arrow[ur] & & \end{tikzcd}$$




$$u \longrightarrow \boxed{NN} \longrightarrow v = G(u)$$

# DeepONets

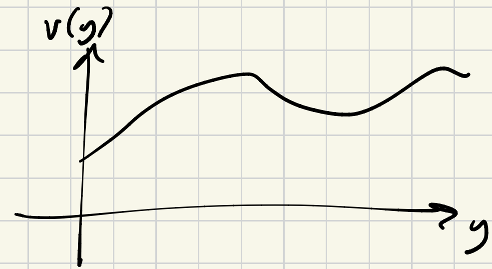
p = # sensors

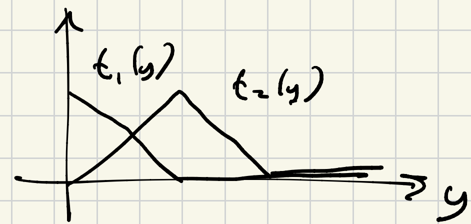



coeffs

$$v(y) = \sum_{i=1}^{p} b_i(u)\, t_i(y)$$

basis on output



**Theorem 1 (Universal Approximation Theorem for Operator).** *Suppose that $\sigma$ 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 nonlinear continuous operator, which maps $V$ into $C(K_2)$. Then for any $\epsilon > 0$, there are positive integers $n$, $p$ and $m$, constants $c_i^k$, $\xi_{ij}^k$, $\theta_i^k$, $\zeta_k \in \mathbb{R}$, $w_k \in \mathbb{R}^d$, $x_j \in K_1$, $i = 1, \ldots, n$, $k = 1, \ldots, p$ and $j = 1, \ldots, m$, such that*

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

*holds for all $u \in V$ and $y \in K_2$. Here, $C(K)$ is the Banach space of all continuous functions defined on $K$ with norm $\| f\|_{C(K)} = \max_{x \in K} |f(x)|$.*

# Training a DeepONet

$$
\begin{tikzcd}
u \arrow[r] & \begin{bmatrix} u(x_1) \\ \vdots \\ u(x_m) \end{bmatrix} \arrow[dr] & & \\
& & \boxed{NN} \arrow[r] & G(u)(y) = v(y) \\
& y \arrow[ur] & &
\end{tikzcd}
$$

for now, we use supervised training

$\longrightarrow$ generate many triples $\{(u_i, y_i, v_i)\}_{i=1}^{N}$

$\longrightarrow$ train the network to output $v_i$ when $u_i, y_i$ input

## Learning $\frac{d}{dx}$

$\Omega = [0,1]$

choose many inputs $u(x)$ $\sin x$, $\cos x$ $\sin \pi x$

choose evaluation points $u(x_1), \ldots, u(x_p)$ $u = [\sin(0.1\pi), \sin(0.2\pi), \ldots, \sin(0.9\pi)]$

choose $y$ (randomly in domain)

$y = 0.372$

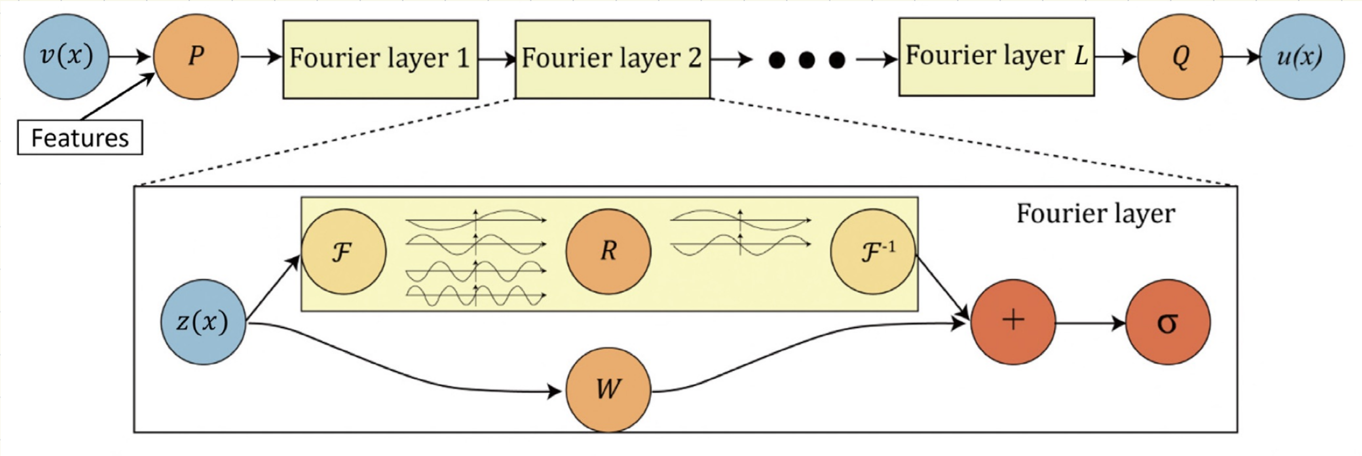
evaluate $\frac{d}{dx} u(y)$

$v = \pi \cos(\pi \cdot 0.372)$

$\Delta v = u$ pick output $v$

calculate input $u$

# Fourier Neural Operators (FNO)




$$u_{k+1}(x) = \sigma\left(W u_k(x) + \mathcal{F}^{-1}\left(R \cdot \mathcal{F}(u_k)\right)\right)$$