Coffee Talk #6

April 14, 2022

Ozgur Taylan Turan

DeepONet:Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators[1]

[1] L. Lu, P. Jin, and G. E. Karniadakis (Mar. 2021). “DeepONet: Learning Nonlinear Operators for Identifying Differential Equations Based on the Universal Approximation Theorem of Operators”. In: *Nature Machine Intelligence* 3.3, pp. 218–229. ISSN: 2522-5839. DOI: 10.1038/s42256-021-00302-5. arXiv: 1910.03193

Why this paper?

- Badly written, but has an interesting problem to tackle!

Preliminary Info

Function

$$f := \mathcal{V} \rightarrow \mathcal{V}$$

Operator

$$G := \mathcal{F} \rightarrow \mathcal{V}$$

$\mathcal{V}$: Vector Space

$\mathcal{F}$: Function Space

- $G(u) := s(x)$, where u is a function of x
- At a location y the operator evaluation is given by $G(u)(y) := s(y)$

Universal Approximation Theorems

Universal Approximation Theorem for Function

For every function g in M^r there is a compact subset K of R^r and an $f \in \Sigma^r(\Psi)$ such that for any $\varepsilon > 0$ we have $\mu(K) < 1 - \varepsilon$ and for every $x \in K$ we have $|f(x) - g(x)| < \varepsilon$, regardless of Ψ, r, or μ. □ [1]

Universal Approximation Theorem for Operator

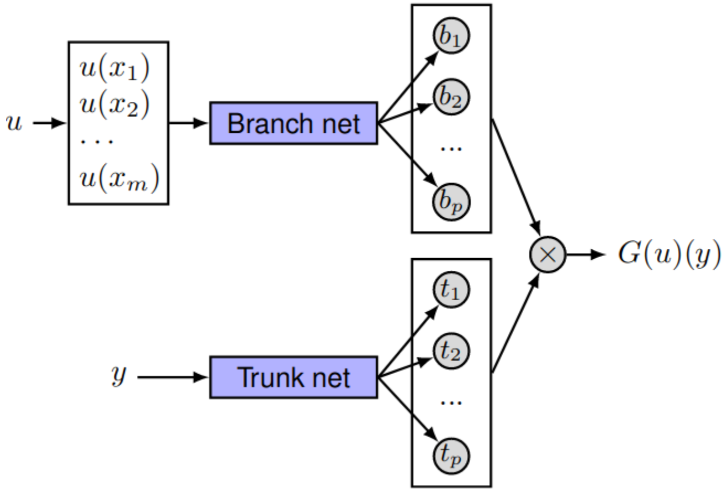
$$\left| G(u)(y) - \sum_{k=1}^{p} \sum_{i=1}^{n} c_i^k \sigma \left(\sum_{j=1}^{m} \xi_{ij}^k u(x_j) + \theta_i^k \right) \sigma(w_k \cdot y + \zeta_k) \right| < \epsilon$$ [2]

---

[1] K. Hornik, M. Stinchcombe, and H. White (Jan. 1989). "Multilayer Feedforward Networks Are Universal Approximators". In: *Neural Networks* 2.5, pp. 359–366. ISSN: 08936080. DOI: 10.1016/0893-6080(89)90020-8

[2] T. Chen and H. Chen (July 1995). "Universal Approximation to Nonlinear Operators by Neural Networks with Arbitrary Activation Functions and Its Application to Dynamical Systems". In: *IEEE Transactions on Neural Networks* 6.4, pp. 911–917. ISSN: 1941-0093. DOI: 10.1109/72.392253

What does it look like?



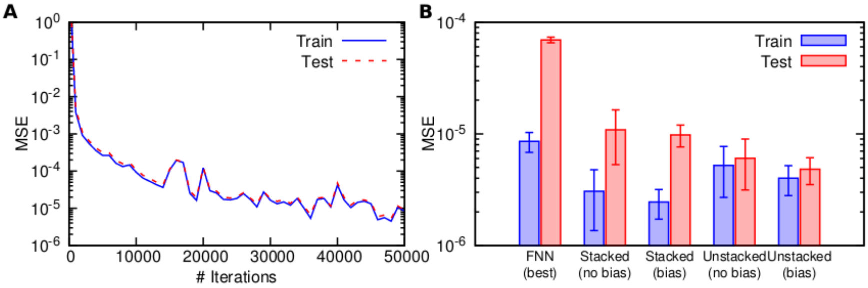
What does it look like?

Try to find the following operator:
$G := u(x) \to s(x) = \int_0^x u(\tau)d\tau$

Procedure:

- **Sample** $(u_k)_{k=1}^p$ **from Gaussian Random Field at locations** $(x_i)_{i=1}^m$
- For every function obtain some of the exact outputs at y of function $G(u)$
- Train with the given structure and selected hyper-parameters

Training Curve



No Results?

Please, check the paper for extensive examples!

Why is this a cool idea?

- Solving ODEs/SODEs can be challenging...
- Previous data usage for fast prediction for dynamical systems
- Combination of pre-trained DeepONets with the PINNs allow faster adaptations for unknown functions...

THANKS!