Introduction to Operator Learning

CASML 2024 Preconference Workshop – Day 2

Divij Ghose

with

Diya Nag Chaudhury

AIREX Lab and Zenteiq.ai

Session Overview

Introduction to Operator Learning

An overview of this session

What are Operators?

- What are operators in the context of PDEs
- How can we train neural networks to learn operators

How do we train Deep Operator Networks?

- DeepONet architecture
- Training a DeepONet

Introduction to Operator Learning

An overview of this session

What are Fourier Neural Operators?

- Architecture of Fourier Neural Operators
- Training an FNO

Hands-on with FNOs

What are Operators?

Introduction to Operators

What is an Operator?

- An operator is a mapping between a space of functions to another space of functions

$$D : f \mapsto D(f)$$

- Here, D(u) is a function which can be sampled at discrete points,

$$D(f) : y \in \mathbb{R}^d \mapsto D(f)(x) \in \mathbb{R}$$

- Operators arise naturally in differential equations

$$-\Delta u(x) = f(x), \quad \text{in} \quad \Omega \subseteq \mathbb{R}^2, \qquad \leftrightharpoons \qquad D : f \mapsto u$$

The universal approximation theorem

How can neural networks learn operators?

- For an arbitrary accuracy, a sufficiently large neural network can approximate any non-linear continuous operator

$$\left| D(f)(y) - \sum_{k=1}^{p} \sum_{i=1}^{n} c_i^k \sigma \left(\sum_{j=1}^{m} \alpha_{ij}^k f(x_j) + \theta_i^k \right) \sigma \left(w_k \cdot y + b_k \right) \right| < \varepsilon$$

$f(x_i)$ Input function

y Points at which output is sampled

$\sigma(\cdot)$ Activation function

The universal approximation theorem

How can neural networks learn operators?

- For an arbitrary accuracy, a sufficiently large neural network can approximate any non-linear continuous operator

$$\left| D(f)(y) - \sum_{k=1}^{p} \sum_{i=1}^{n} c_i^k \sigma \left(\sum_{j=1}^{m} \alpha_{ij}^k f(x_j) + \theta_i^k \right) \sigma \left(w_k \cdot y + b_k \right) \right| < \varepsilon$$

Branch Network

Trunk Network

What are DeepONets?

The DeepONet Architecture

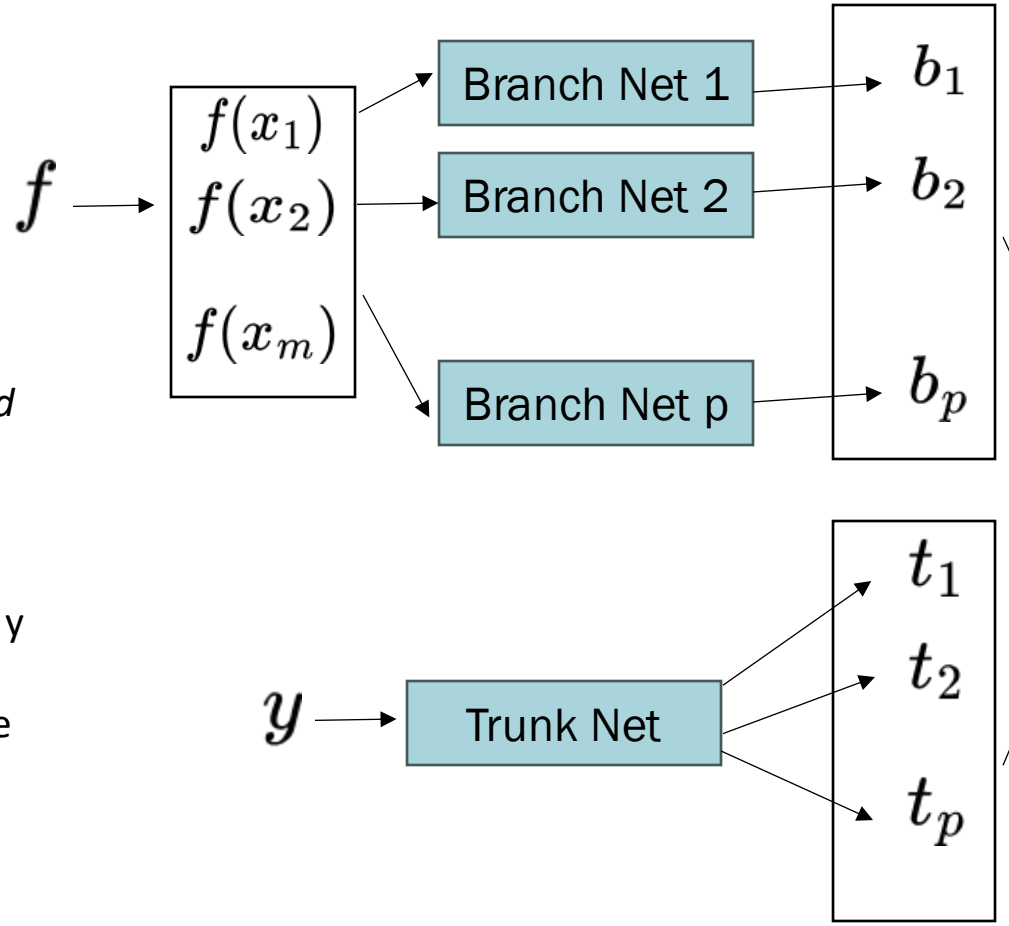
What are branch and Trunk Nets?

Branch Nets

- Branch Nets sample the input functions at fixed "sensors"
- Branch nets can be *stacked* or *unstacked*

Trunk Nets

- Trunk nets take the points y (at which the output function is sampled) as the input



Output

An inner product of the p-dimensional vectors from the branch and trunk nets gives the value of the output function at point y

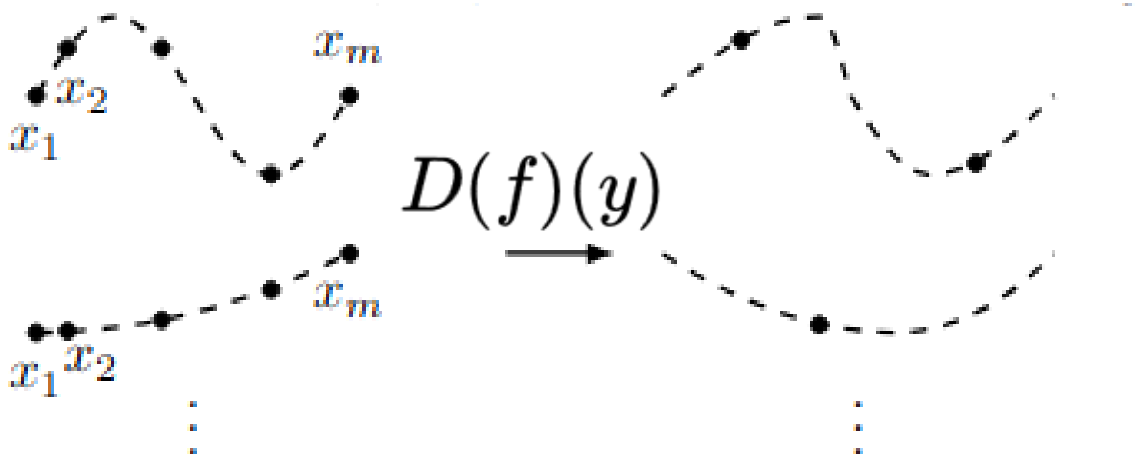
$$D(f)(y) \approx \sum_{k=1}^{p} b_k t_k$$

Input and output functions

How do we sample these functions?

Input function

- All functions sampled at the same "sensor" points



Output function

- Output functions can be sampled at different points via the trunk net

Source: DeepONet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of Operators, Lu Lu et al.

What are Fourier Neural Operators?

The FNO Architecture

What are the components of FNOs?

Components of FNOs

- A lifting layer, P
- Multiple sets of Fourier Layers
- A projection layer, Q

Why use Fourier layers?

- Fourier layers replace convolutional layers.
- The inputs and outputs of PDEs are (usually) continuous functions, more efficient to represent them in the Fourier space.

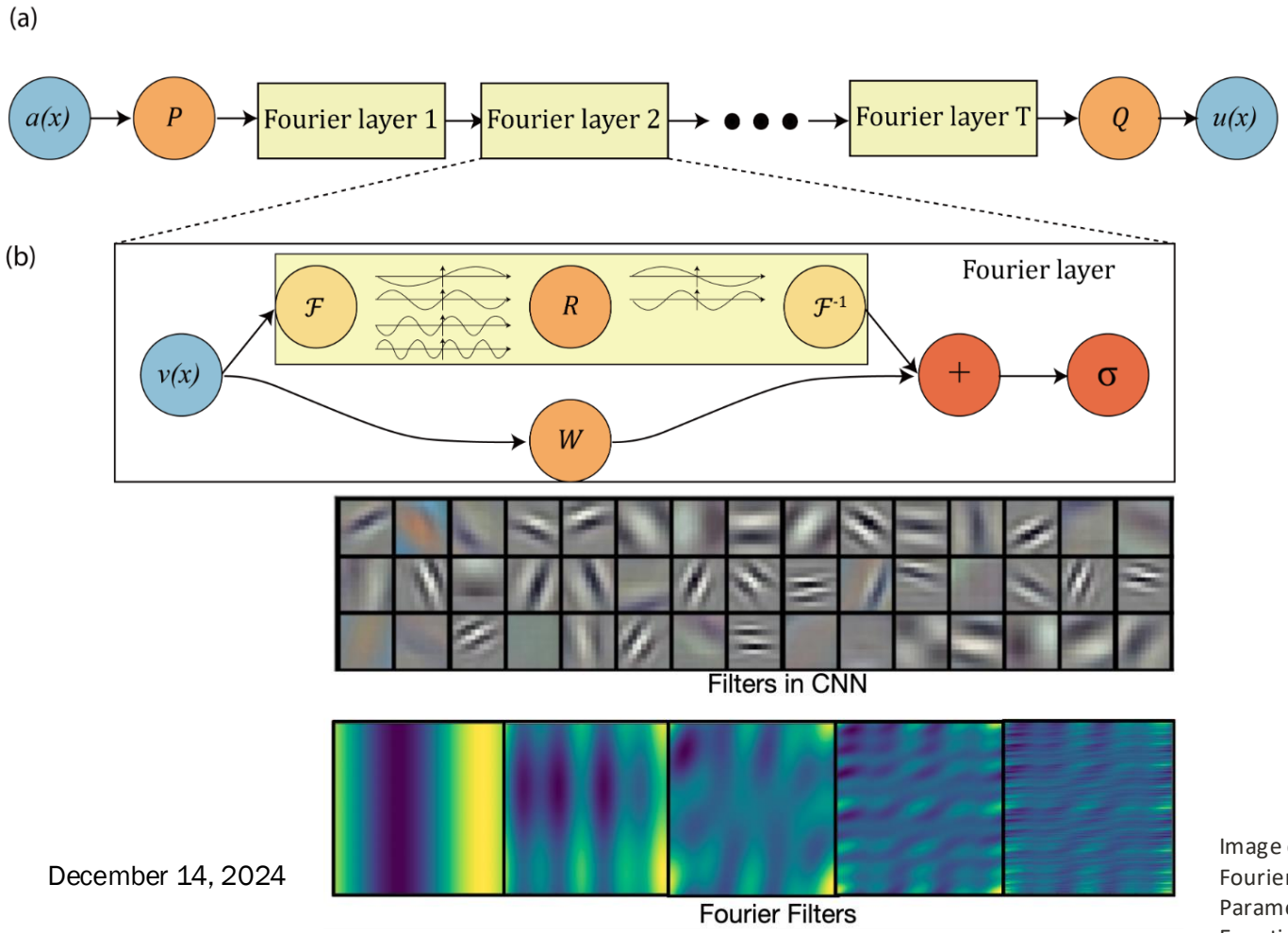
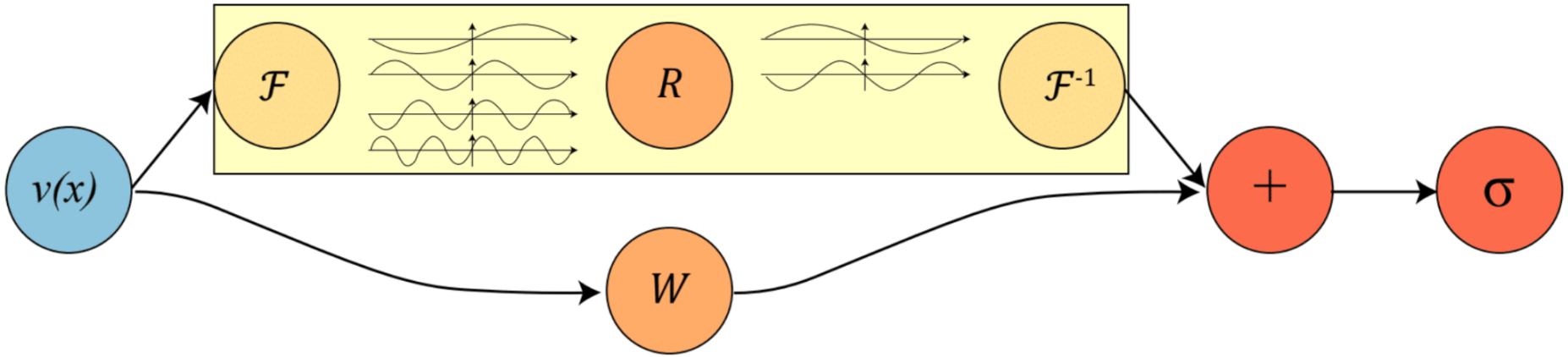


Image credit and reference: Fourier Neural Operator for Parametric Partial Differential Equations, Li et al.

The Fourier Layer

What are the components of a Fourier layer?



Operations in a Fourier Layer

- Transform the input into Fourier space
- Batched matrix multiplication for linear
- transformation
- Inverse Fourier transform back into real space
- Add point-wise linear transform
- Apply non-linear activation function

Image credit and reference: Fourier Neural Operator for Parametric Partial Differential Equations, Li et al.

Applications of FNOs

What can FNOs be used for?

Some use cases of FNOs

- Predict solution given a forcing function or spatially dependent parameter
- Predict solution at a time (t) given the initial condition
- Predict solution at a time (t+1) given the solution at time (t)

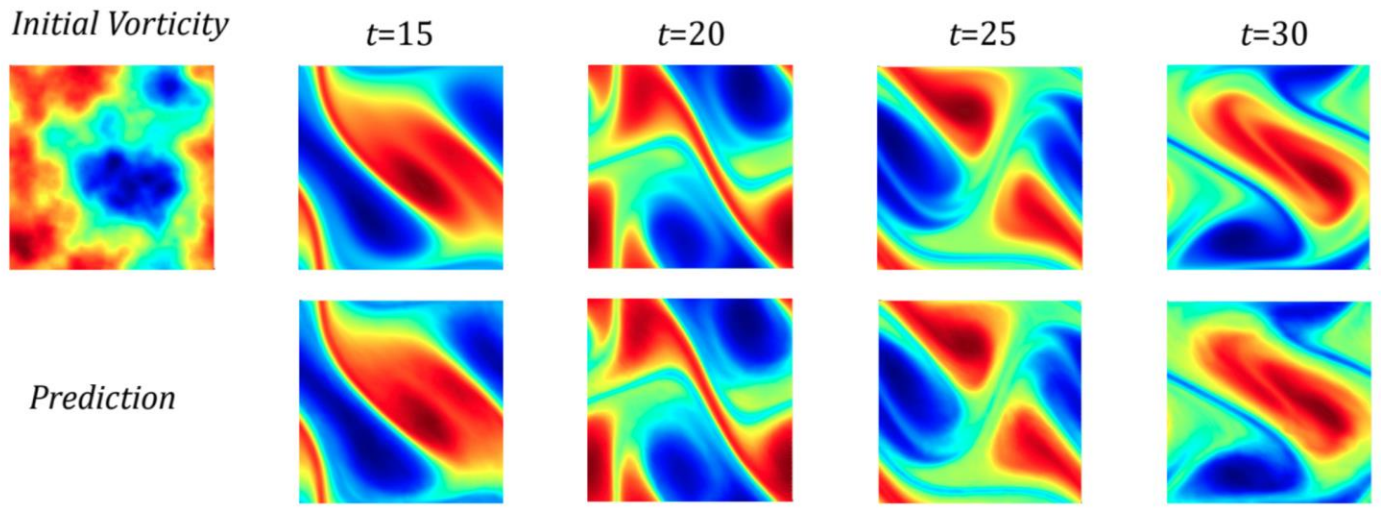


Image credit and reference: Fourier Neural Operator for Parametric Partial Differential Equations, Li et al.

Hands-on with FNOs follows

Q&A

Resources for DeepONets

- DeepXDE - https://deepxde.readthedocs.io
- NVIDIA Modulus - https://developer.nvidia.com/modulus

Resources for FNOs

- Neural Operators in PyTorch - https://neuraloperator.github.io
- Modulus - https://docs.nvidia.com/deeplearning/modulus/modulus-v2209/user_guide/theory/architectures.html#fourier-neural-operator

AIREX Lab

Divij's profile