What's Wrong with End-to-End Learning for Phase Retrieval?

Ju Sun

Computer Sci & Eng, University of Minnesota

Jan 23, 2024

Machine Learning for Scientific Imaging, EI'24

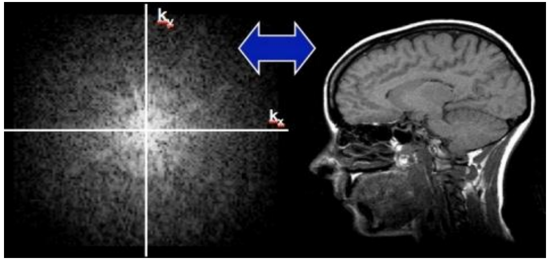
Inverse problems (IPs)

Most scientific IPs are **nonlinear**

Image denoising

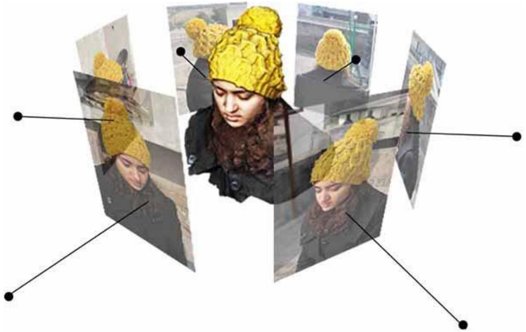
Object $\xrightarrow{f}$ **Observation**

$$\mathbf{x} \underset{f^{-1}}{\overset{f}{\rightleftarrows}} \mathbf{y}$$

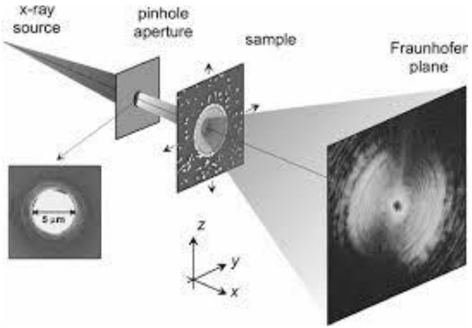


MRI reconstruction

Image super-resolution



3D reconstruction



Coherent diffraction imaging (CDI)

Traditional methods

Inverse problem: given $\mathbf{y} = f(\mathbf{x})$, recover $\mathbf{x}$

$$\min_{\mathbf{x}} \underbrace{\ell(\mathbf{y}, f(\mathbf{x}))}_{\text{data fitting}} + \lambda \underbrace{\mathrm{R}(\mathbf{x})}_{\text{regularizer}}$$

RegFit

Challenges:

1) Which ℓ? (e.g., unknown/compound noise)
2) Which R? (e.g., structures not amenable to math description)
3) Global optimization — esp. for nonlinear IPs
4) Convergence speed of iterative numerical methods

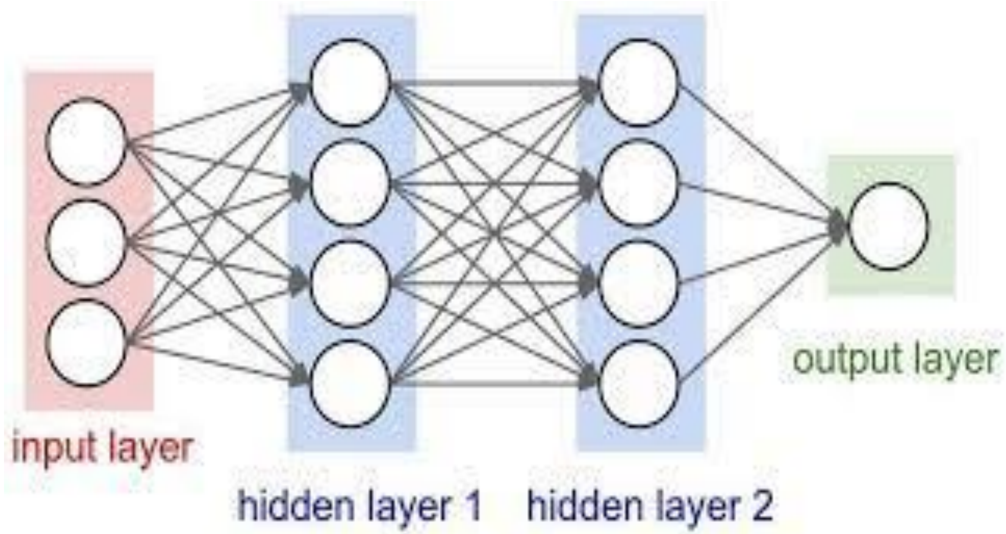
How has deep learning (DL) changed the story?

DL methods: the radical way

Inverse problem: given $\mathbf{y} = f(\mathbf{x})$, recover $\mathbf{x}$

Learn the f^{-1} with a training set $\{(\mathbf{y}_i, \mathbf{x}_i)\}$

$\mathbf{y}_i$



$\mathbf{x}_i$

DL methods: the middle way

Inverse problem: given $\mathbf{y} = f(\mathbf{x})$, recover $\mathbf{x}$

$$\min_{\mathbf{x}} \underbrace{\ell(\mathbf{y}, f(\mathbf{x}))}_{\text{data fitting}} + \lambda \underbrace{\mathrm{R}(\mathbf{x})}_{\text{regularizer}}$$

RegFit

Recipe: revamp numerical methods for RegFit with **pretrained/trainable DNNs**

DL methods: the middle way

Algorithm unrolling

$$\min_{\mathbf{x}} \underbrace{\ell(\mathbf{y},\, f(\mathbf{x}))}_{\text{data fitting}} + \lambda \underbrace{\mathrm{R}(\mathbf{x})}_{\text{regularizer}}$$

If R proximal friendly

$$\mathbf{x}^{k+1} = \mathcal{P}_R\left(\mathbf{x}^k - \eta\nabla^\top f(\mathbf{x}^k)\ell'(\mathbf{y},\, f(\mathbf{x}^k))\right)$$

Idea: make $\mathcal{P}_R$ trainable, using $\{(\mathbf{x}_i\,,\,\mathbf{y}_i)\}$

E.g.,

$$\ell(\mathbf{y},\, f(\mathbf{x})) = \|\mathbf{y} - \mathbf{A}\,\mathbf{x}\|_2^2$$

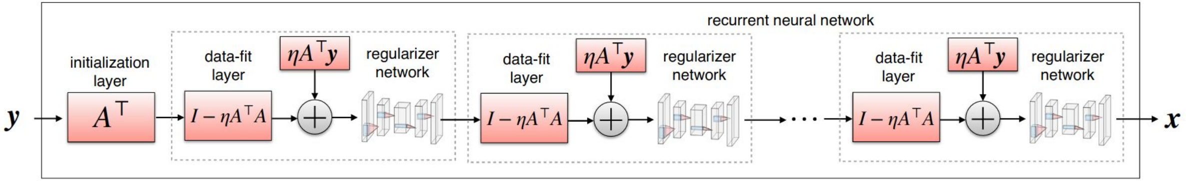


Fig credit: Deep Learning Techniques for Inverse Problems in Imaging https://arxiv.org/abs/2005.06001

DL methods: the middle way

$$\min_{\mathbf{x}} \underbrace{\ell(\mathbf{y},\, f(\mathbf{x}))}_{\text{data fitting}} + \lambda \underbrace{\mathrm{R}(\mathbf{x})}_{\text{regularizer}}$$

Using $\{\mathbf{x}_i\}$ only

Plug-and-Play

$$\mathbf{x}^{k+1} = \mathcal{P}_R\left(\mathbf{x}^k - \eta \nabla^\top f(\mathbf{x}^k)\ell'(\mathbf{y},\, f(\mathbf{x}^k))\right)$$

E.g. replace $\mathcal{P}_R$ with pretrained denoiser

Deep generative models

Pretraining: $\mathbf{x}_i \approx G_\theta(\mathbf{z}_i) \;\forall i$

Deployment: $\min_{\mathbf{z}} \ell(\mathbf{y},\, f \circ G_\theta(\mathbf{z})) + \lambda R \circ G_\theta(\mathbf{z})$

DL methods: a survey

Deep Learning Techniques for Inverse Problems in Imaging

Gregory Ongie[*], Ajil Jalal[†], Christopher A. Metzler[‡]
Richard G. Baraniuk[§], Alexandros G. Dimakis[¶], Rebecca Willett[‖]

April 2020

Abstract

Recent work in machine learning shows that deep neural networks can be used to solve a wide variety of inverse problems arising in computational imaging. We explore the central prevailing themes of this emerging area and present a taxonomy that can be used to categorize different problems and reconstruction methods. Our taxonomy is organized along two central axes: (1) whether or not a forward model is known and to what extent it is used in training and testing, and (2) whether or not the learning is supervised or unsupervised, i.e., whether or not the training relies on access to matched ground truth image and measurement pairs. We also discuss the tradeoffs associated with these different reconstruction approaches, caveats and common failure modes, plus open problems and avenues for future work.

Focuses on **linear** inverse problems, i.e., f linear

https://arxiv.org/abs/2005.06001

See also: Model-based deep learning
https://arxiv.org/abs/2012.08405

DL methods: the **economic (radical)** way

Deep image prior (DIP) $\mathbf{x} \approx G_\theta(\mathbf{z})$ G_θ (and $\mathbf{z}$) trainable

$$\min_{\mathbf{x}} \underbrace{\ell(\mathbf{y}, f(\mathbf{x}))}_{\text{data fitting}} + \lambda \underbrace{\mathrm{R}(\mathbf{x})}_{\text{regularizer}}$$

$$\Downarrow$$

$$\min_{\theta} \ell(\mathbf{y}, f \circ G_\theta(\mathbf{z})) + \lambda R \circ G_\theta(\mathbf{z})$$

No extra training data!

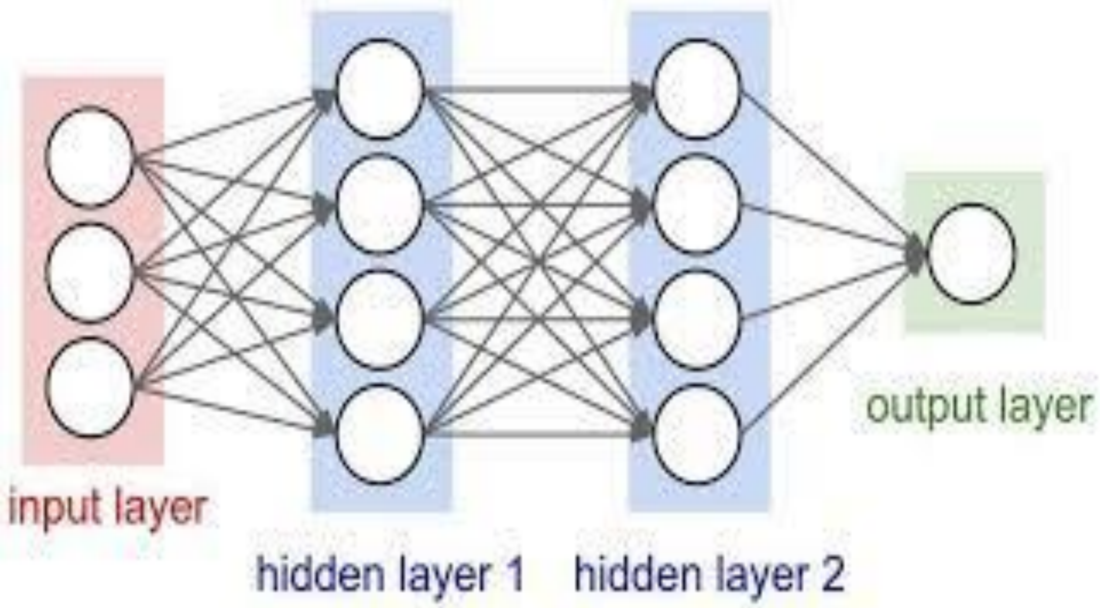
Ulyanov et al. **Deep image prior**. IJCV'20. https://arxiv.org/abs/1711.10925

In other words, deep reparametrization

In the same vein, **neural implicit representations** (PINNs in applied math)

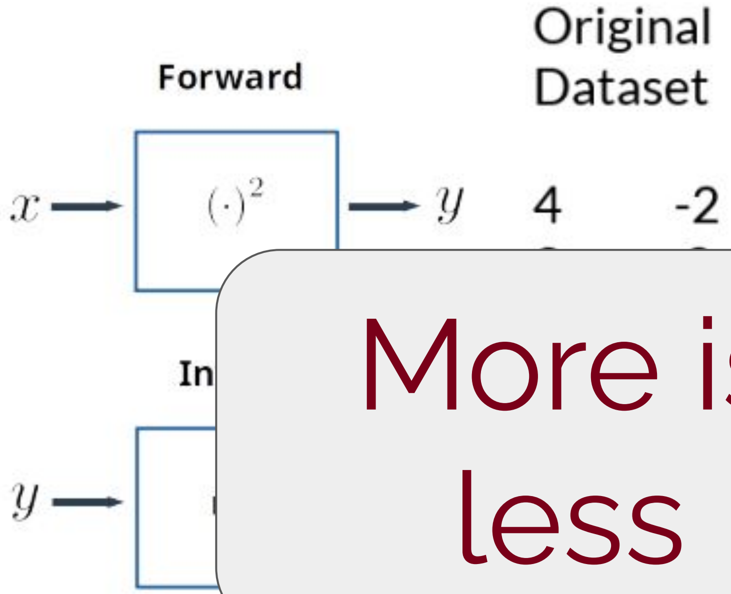
Focus here: **end-to-end methods**

$\mathbf{y}_i$

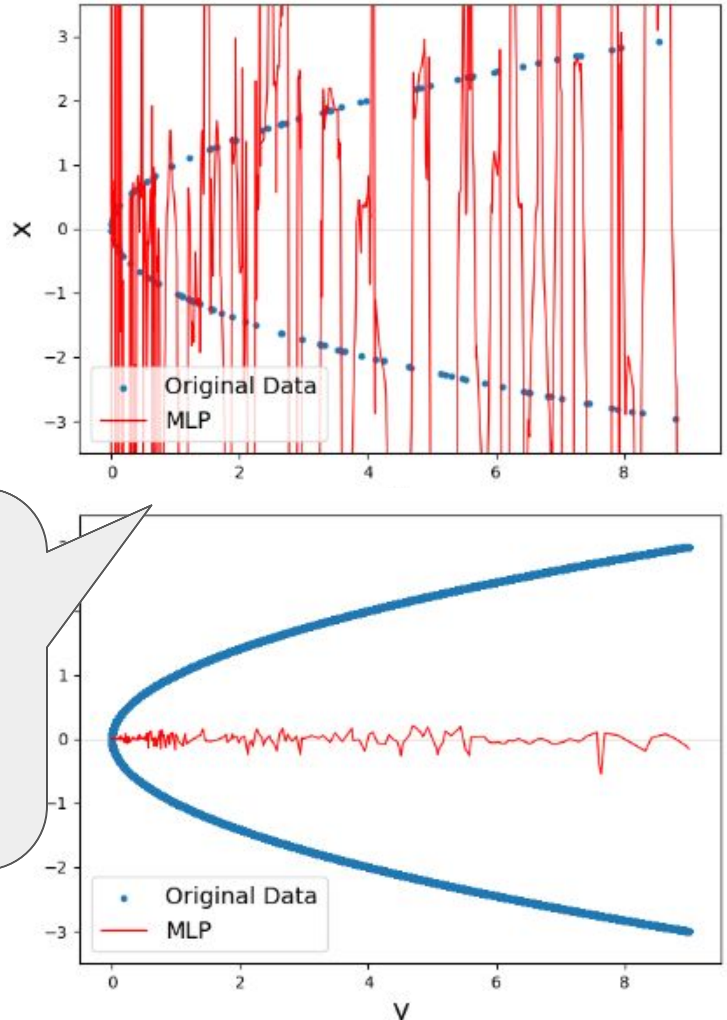


$\mathbf{x}_i$

A striking experiment



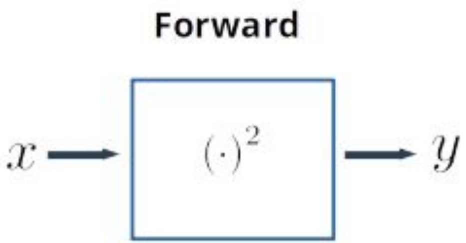
More is less



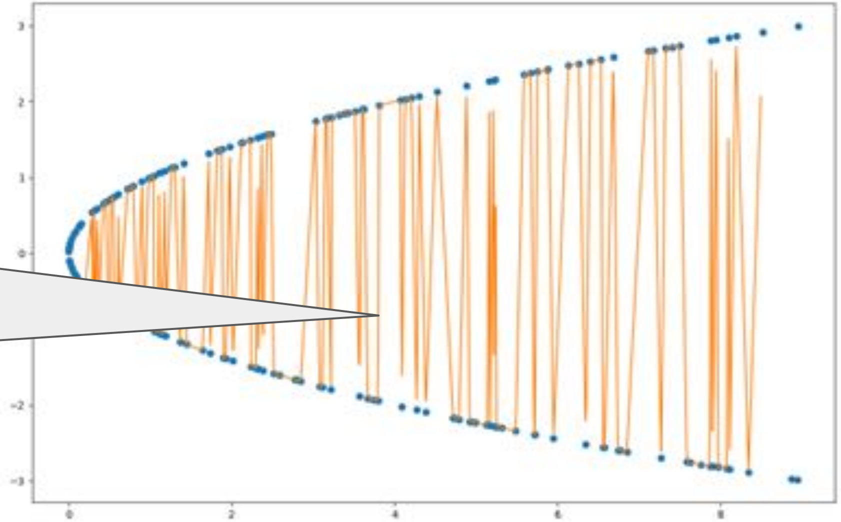
Why “more is less” here?

Forward symmetry: $\{+\sqrt{y}, -\sqrt{y}\} \leftrightarrow y$

Implies: on dense training set, very close y’s can mapped to very far aways x’s different by signs

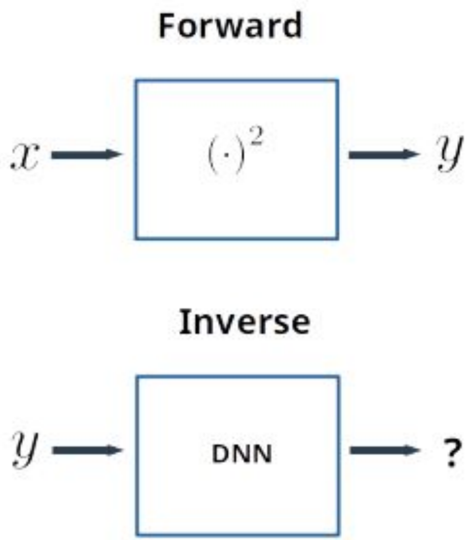


Original Dataset	
4	-2
9	3
16	-4



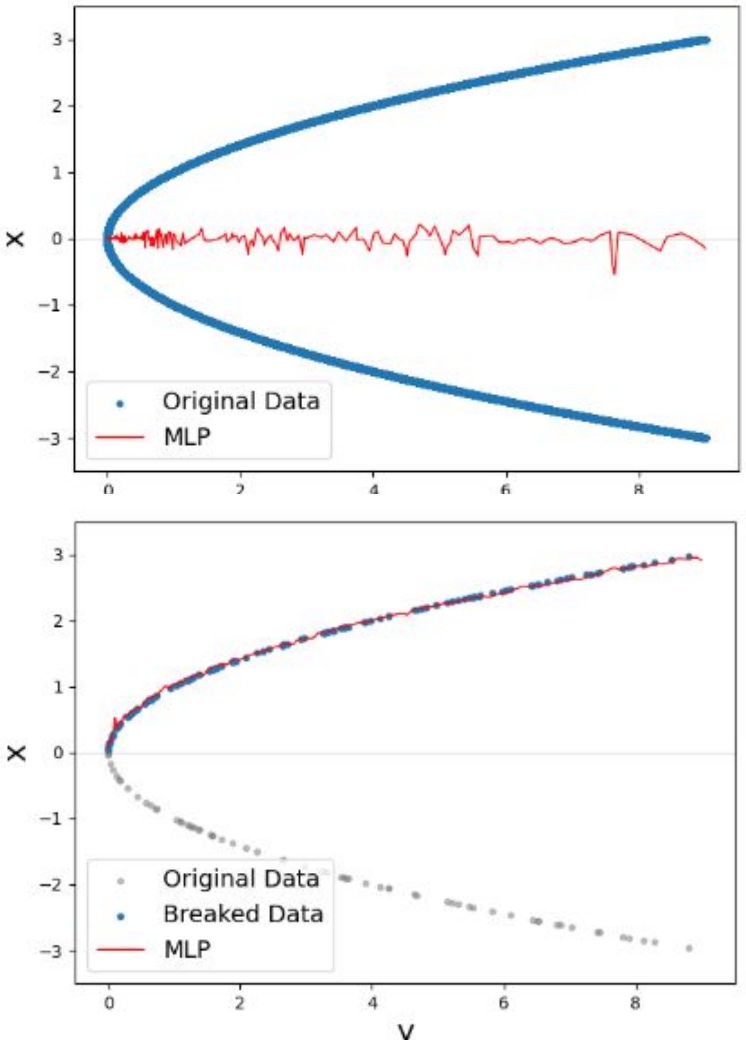
Highly oscillatory target function to learn by DNNs—difficult

Remedy: symmetry breaking



Original Dataset

4	-2
9	3
16	-4
100	10
4	2
64	-8
36	6



Fix all signs to be positive

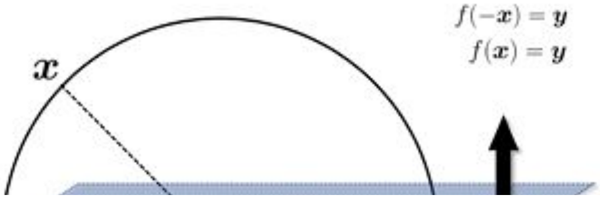
A slightly more complicated example

$$\mathbf{y} = |\mathbf{A}\mathbf{x}|^2 \quad \mathbf{A} : \text{iid Gaussian}$$

(Gaussian phase retrieval)

Forward symmetry: global sign

$$\mathbf{y} = |\mathbf{A}\mathbf{x}|^2 = |\mathbf{A}(-\mathbf{x})|^2$$

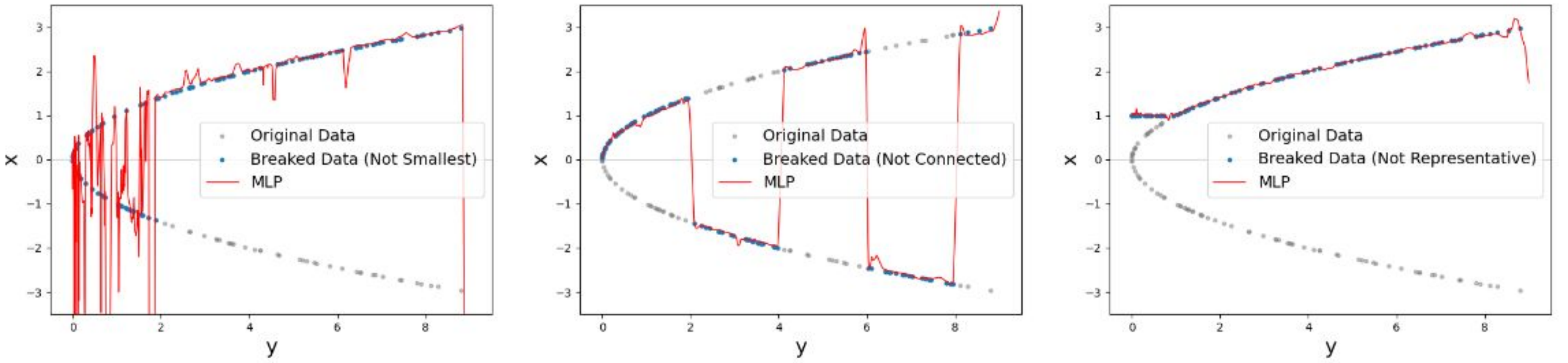


		After Symmetry Breaking		Before Symmetry Breaking	
Dim	Sample	DNN	K-NN	DNN	K-NN
5	2e4	4.08	11.82	85.37	68.26
	5e4	2.20	9.41	90.51	66.58
	1e5	1.30	7.98	96.66	66.18
	1e6	0.37	4.71	122.71	65.08

More is more **More is less**

Symmetry-breaking principle

Symmetry breaking: a preprocessing step on the training set

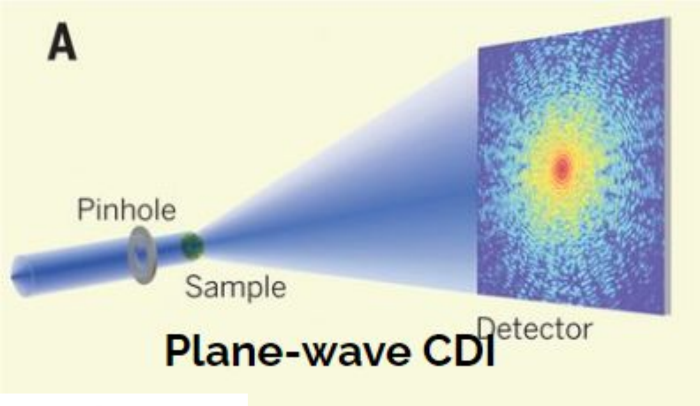


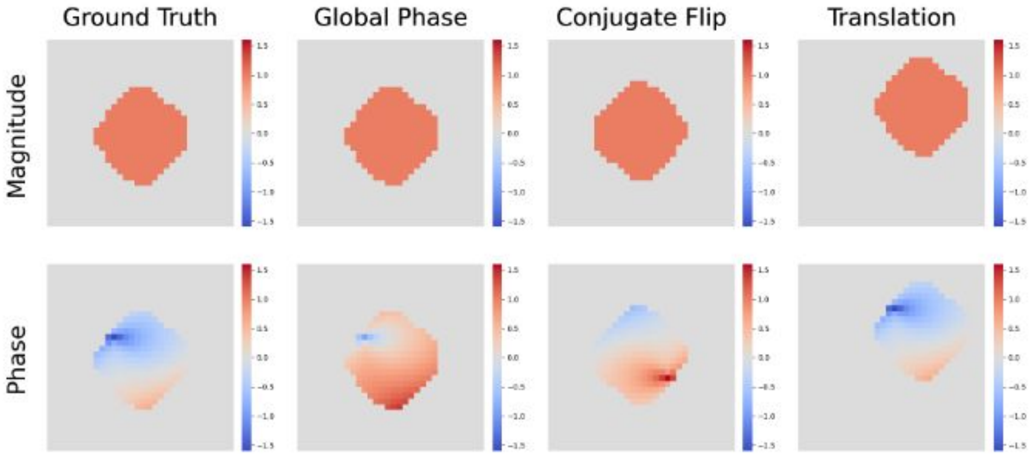
Finding the smallest, connected, representative set

A realistic example: far-field phase retrieval

plane-wave CDI (Fraunhofer PR)

$$\boldsymbol{Y} = |\mathcal{F}(\boldsymbol{X})|^2$$





3 symmetries

Algorithm 2 Procedure of symmetry breaking for FPR

Input: Forward mapping $\hat{f}(\boldsymbol{X}) \doteq |\mathscr{F}(\boldsymbol{X})|^2$ and randomly sam-
pled input data points $D = \{\boldsymbol{X}_j\} \subset \mathbb{C}^{N_1 \times N_2}$

Output: Symmetry breaking training dataset

Centering the nonzero content inside $\boldsymbol{X}_j$'s in th
put space. This heuristically breaks 2D translati

Taking the oversampled Fourier transform of $\boldsymbol{X}$

Decompose each element of $\mathscr{X}$ into polar form:
$\boldsymbol{\rho}(k_1,k_2)e^{i\boldsymbol{\theta}(k_1,k_2)}$ with the phase $\boldsymbol{\Omega}(k_1,k_2) = e^{i}$

Breaking symmetry in the phase domain with
gorithm $\boldsymbol{\Phi}$ described below.

while $\boldsymbol{X}_j \in D$ **do** $\boldsymbol{\Phi}$ as the following:

    Performing global phase transfer to make
    $\boldsymbol{\Omega}(k_1,k_2) \leftarrow \overline{\boldsymbol{\Omega}(1,1)}\boldsymbol{\Omega}(k_1,k_2)$

    if $\boldsymbol{\Omega}(1,2) \in \mathbb{S}_+$ **then**

        $\boldsymbol{\Omega}(k_1,k_2) \leftarrow \boldsymbol{\Omega}(k_1,k_2)$

    else if $\boldsymbol{\Omega}(1,2) \notin \mathbb{S}_+$ **then**

        $\boldsymbol{\Omega}(k_1,k_2) \leftarrow \overline{\boldsymbol{\Omega}(k_1,k_2)}$

    end if

end while

Applying forward mapping $\hat{f}$ on each point $\boldsymbol{X}_j$ and form a new training set $\{(|\mathscr{F}(\boldsymbol{X}_j)|^2, \boldsymbol{X}_j)\}$, which is a symmetry breaking set.

On training set

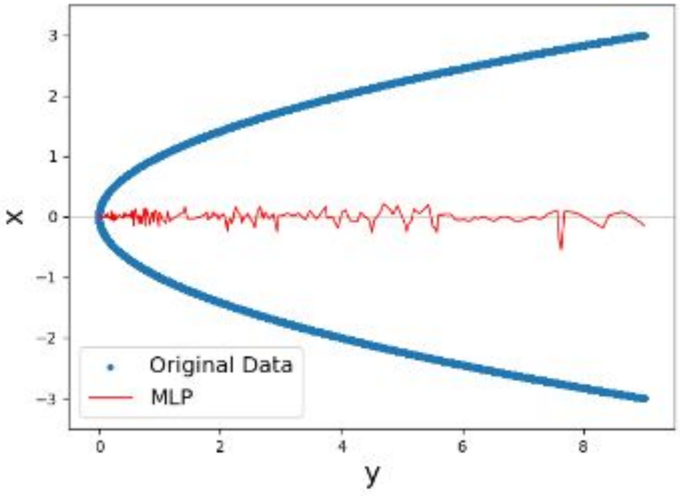
	UNet		SiSPRNet	
Sample	**Before**	**After**	**Before**	**After**
500	0.1341	0.0219	0.1525	0.0270
1000	0.0665	0.0236	0.1474	0.0212
1500	0.0672	0.0215	0.0701	0.0199
5000	0.0360	0.0187	0.0463	0.0147

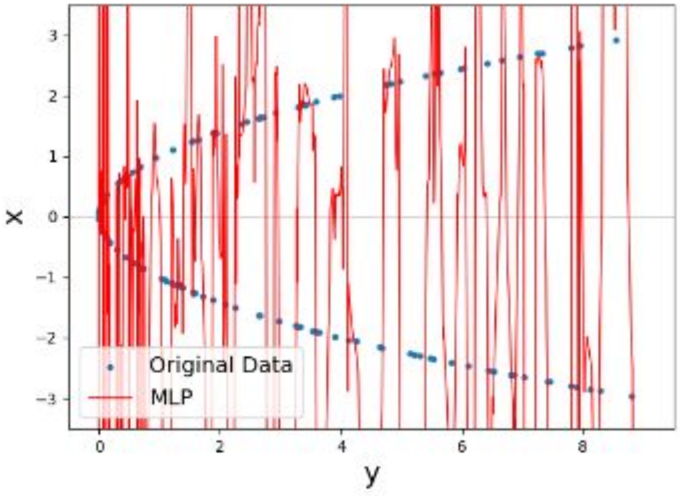
On test set

	UNet		SiSPRNet	
Sample	**Before**	**After**	**Before**	**After**
500	0.1210	0.0448	0.1412	0.0426
1000	0.1057	0.0460	0.1389	0.0357
1500	0.1056	0.0385	0.0733	0.0336
5000	0.0580	0.0326	0.0568	0.0255

More is <u>not</u> less

Why we don't quite feel the pain?





In practice, we're in data-sparse regime

But gain in performance is real and substantial

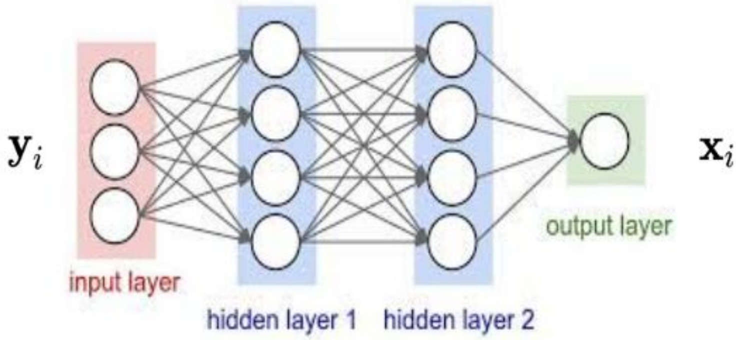
On training set

	UNet		SiSPRNet	
Sample	**Before**	**After**	**Before**	**After**
500	0.1341	0.0219	0.1525	0.0270
1000	0.0665	0.0236	0.1474	0.0212
1500	0.0672	0.0215	0.0701	0.0199
5000	0.0360	0.0187	0.0463	0.0147

On test set

	UNet		SiSPRNet	
Sample	**Before**	**After**	**Before**	**After**
500	0.1210	0.0448	0.1412	0.0426
1000	0.1057	0.0460	0.1389	0.0357
1500	0.1056	0.0385	0.0733	0.0336
5000	0.0580	0.0326	0.0568	0.0255

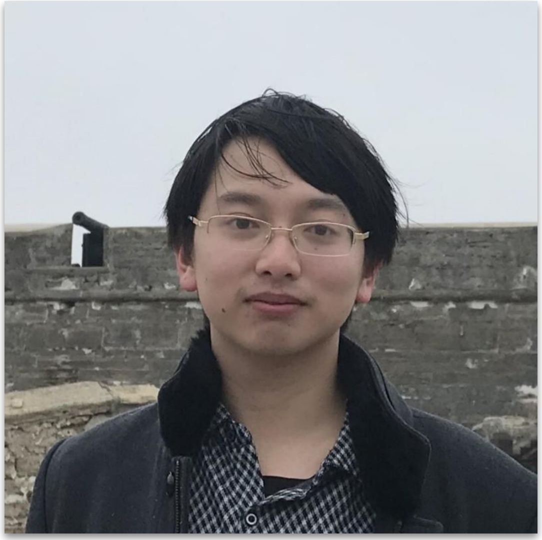
Focus here: **end-to-end methods**



Takeaway

- Forward symmetries in nonlinear IPs can **hurt/ruin** end-to-end DL methods
- Symmetry-breaking always benefits no matter data-rich or data-poor

Thanks



Wenjie Zhang
(CS&E, UMN)

Yuxiang Wan
(CS&E, UMN)

Dr. Zhong Zhuang
(Physics, UCLA)