Deep Neural Networks Motivated by Partial Differential Equations

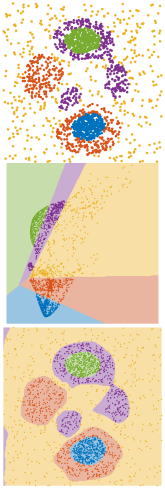
Machine Learning for Physics and the Physics of Learning

Los Angeles, September, 2019

Lars Ruthotto
Departments of Mathematics and Computer Science, Emory University
lruthotto@emory.edu
@lruthotto

Agenda: Deep Neural Networks motivated by PDEs

- **Deep Learning meets Optimal Control** (warmup)
- **Stability and Generalization** (skipped $\rightsquigarrow$ yesterday)
  - when is deep learning well-posed?
  - stabilizing the forward propagation
- **Convolutional Neural Networks motivated by PDEs**
  - parabolic CNNs: multiscale and multilevel schemes
  - hyberbolic CNNs: memory efficient + stable
  - IMEX-Net: Semi-implicit time integration
  - LeanResNet: reduced architectures
- **Conclusion and Summary**



Goals: Theoretical insight, mathematically sound architectures, competitive results.

E Haber, LR
Stable Architectures for DNNs.
Inverse Problems, 2017.

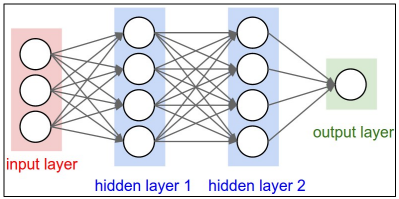
E Holtham et al.
Learning Across Scales.
AAAI, 2018.

B Chang et al.,
Reversible Architectures for Deep ResNNs.
AAAI, 2018.

LR, E Haber
Deep Neural Networks motivated by PDEs.
arXiv, 2018.

Deep Learning meets Optimal Control

Deep Learning Revolution (?)

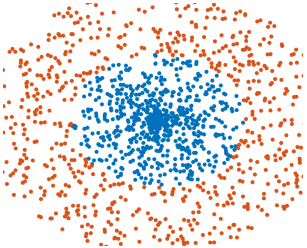


$$\begin{cases} \mathbf{Y}_{j+1} &= \sigma(\mathbf{K}_j\mathbf{Y}_j + \mathbf{b}_j) \\ \mathbf{Y}_{j+1} &= \mathbf{Y}_j + \sigma(\mathbf{K}_j\mathbf{Y}_j + \mathbf{b}_j) \\ \mathbf{Y}_{j+1} &= \mathbf{Y}_j + \sigma\left(\mathbf{K}_{j,2}\sigma(\mathbf{K}_{j,1}\mathbf{Y}_j + \mathbf{b}_{j,1}) + \mathbf{b}_{j,2}\right) \\ &\vdots \end{cases}$$

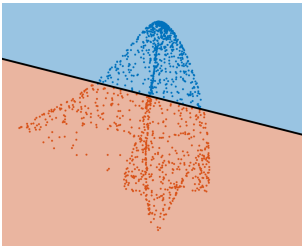
(Notation: $\mathbf{Y}_j$: features, $\mathbf{K}_j, \mathbf{b}_j$: weights, σ : activation)

- deep learning: use neural networks (from $\approx$ 1950's) with many hidden layers
- able to "learn" complicated patterns from data
- applications: image classification, face recognition, segmentation, driverless cars, ...
- recent success fueled by: massive data sets, computing power
- A few recent references:
  - **A radical new neural network design could overcome big challenges in AI**, MIT Tech Review '18
  - Data Scientist: Sexiest Job of the 21st Century, Harvard Business Rev '17

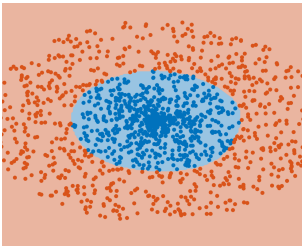
Optimal Control Framework for Deep Learning



training data, $\mathbf{Y}_0, \mathbf{C}$



prop. features, $\mathbf{Y}(T), \mathbf{C}$



classification result

Supervised Deep Learning Problem

Given training data, $\mathbf{Y}_0$, and labels, $\mathbf{C}$, find **network parameters** $\boldsymbol{\theta}$ and **classification weights** $\mathbf{W}$, $\boldsymbol{\mu}$ such that the DNN predicts the data-label relationship (and generalizes to new data), i.e., solve

$$\text{minimize}_{\boldsymbol{\theta},\mathbf{W},\boldsymbol{\mu}} \quad \text{loss}[g(\mathbf{W}+\boldsymbol{\mu}), \mathbf{C}] + \text{regularizer}[\boldsymbol{\theta}, \mathbf{W}, \boldsymbol{\mu}]$$

Deep Residual Neural Networks (simplified)

Award-winning forward propagation

$$\mathbf{Y}_{j+1} = \mathbf{Y}_j + h\mathbf{K}_{j,2}\sigma(\mathbf{K}_{j,1}\mathbf{Y}_j + \mathbf{b}_j), \quad \forall\, j = 0, 1, \ldots, N-1.$$

ResNet is forward Euler discretization of

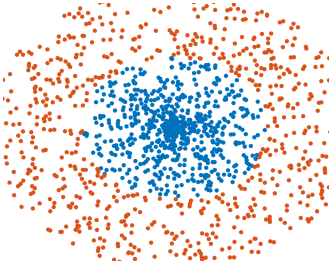
$$\partial_t \mathbf{y}(t) = \mathbf{K}_2(t)\sigma\left(\mathbf{K}_1(t)\mathbf{y}(t) + \mathbf{b}(t)\right), \qquad \mathbf{y}(0) = \mathbf{y}_0.$$

Notation: $\boldsymbol{\theta}(t) = (\mathbf{K}_1(t), \mathbf{K}_2(t), \mathbf{b}(t))$ and

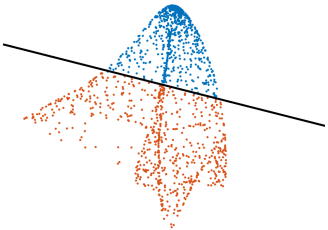
$$\partial_t \mathbf{y}(t) = f(\mathbf{y}, \boldsymbol{\theta}(t)), \quad \mathbf{y}(0) = \mathbf{y}_0$$

where $f(\mathbf{y}, \boldsymbol{\theta}) = \mathbf{K}_2(t)\sigma\left(\mathbf{K}_1(t)\mathbf{y}(t) + \mathbf{b}(t)\right).$

K. He, X. Zhang, S. Ren, and J. Sun
Deep residual learning for image recognition.
IEEE Conf. on CVPR, 770–778, 2016.



input features, $\mathbf{Y}_0$



propagated features, $\mathbf{Y}_N$

(Some) Related Work

DNNs as (stochastic) Dynamical Systems

- **Weinan E**, *Proposal on ML via Dynamical Systems*, Commun. Math. Stat., 5(1), 2017.
- **E Haber, LR**, *Stable Architectures for DNNs*, Inverse Problems, 2017.
- **Q. Li, L. Chen, C. Tai, Weinan E**, *Maximum Principle Based Algorithms*, arXiv, 2017.
- **B. Wang, B. Yuan, Z. Shi, S. Osher**, *ResNets Ensemble via the Feynman-Kac Formalism*, arXiv, 2018.

Numerical Time Integrators

- **Y. Lu, A. Zhong, Q. Li, B. Dong**, *Beyond Finite Layer DNNs*, arXiv, 2017.
- **B. Chang, L. Meng, E. Haber, LR, D. Begert, E. Holtham**, *Reversible architectures for DNNs*, AAAI, 2018.
- **T. Chen, Y. Rubanova, J. Bettencourt, D. Duvenaud**, *Neural ODEs*, NeurIPS, 2018.
- **E. Haber, K. Lensink, E. Treister, LR**, *IMEXnet: Forward Stable DNN*. ICML, 2019.

Optimal Control

- **S. Günther, LR, J.B. Schroder, E.C. Cyr, N.R. Gauger**, *Layer-parallel training of ResNets*, arXiv, 2018.
- **A. Gholami, K. Keutzer, G. Biros**, *ANODE: Unconditionally Accurate Memory-Efficient Gradients for Neural ODEs*, arXiv, 2019.
- **T. Zhang, Z. Yao, A. Gholami, K. Keutzer, J. Gonzalez, G. Biros, M. Mahoney**, *ANODEV2: A Coupled Neural ODE Evolution Framework*, arXiv, 2019.

PDE-motivated Approaches

- **E. Haber, LR, E. Holtham**, *Learning across scales - Multiscale CNNs*, AAAI, 2018.
- **LR, E. Haber**, *DNNs motivated by PDEs*, arXiv, 2018.

Convolutional Neural Networks meet PDEs

CNNs: Computer Science and Engineering Challenges

Computations

- note that networks have a widths and depth
- Toy example: width 16 , depth 20 (time steps).

Forward propagation: 5,120 2D-convs/image

Memory Consumption

- adjoint equations (backpropagation) need intermediate states (hidden features)
- Toy example (continued): training data is 5k images with 32×32 pixels.

Storage (just features): 130 GB (double) or 65 GB (single).

Architecture Design & Interpretation

- CNN should be easy to train and generalize well
- CNN should be difficult to fool (adversarial attacks)
- CNN should give uncertainties and explain reasoning

Pei et al., *DeepXplore*, 2017

paradigm shift needed for stable, well-posed, and efficient ML

Convolutions and PDEs

Let $\mathbf{y}$ be of 1D grid function $\mathbf{y} \leftrightarrow y$ (grid: n cells of width $h_x = 1/n$)

$$K(\theta)\mathbf{y} = [\theta_1 \ \theta_2 \ \theta_3] * \mathbf{y} = \left(\frac{\beta_1}{4}[1 \ \ 2 \ \ 1] + \frac{\beta_2}{2h_x}[-1 \ \ 0 \ \ 1] + \frac{\beta_3}{h_x^2}[-1 \ \ 2 \ -1] \right) * y,$$

where the coefficients $\beta_1, \beta_2, \beta_3$ satisfy

$$\begin{pmatrix} 1/4 & -1/(2h_x) & -1/h_x^2 \\ 1/2 & 0 & 2/h_x^2 \\ 1/4 & 1/(2h_x) & -1/h_x^2 \end{pmatrix} \begin{pmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \end{pmatrix} = \begin{pmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \end{pmatrix}.$$

In the limit $h_x \to 0$ this gives

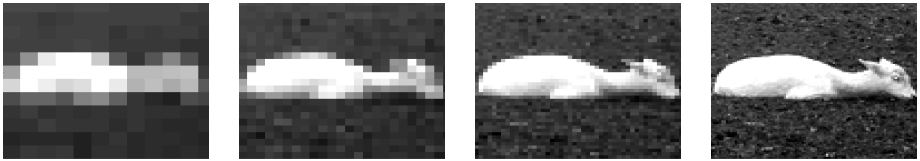
$$K(\theta(t)) = \beta_1(t) + \beta_2(t)\partial_x + \beta_3(t)\partial_x^2.$$

Take-aways:

- Convolution operator K is linear combination of differential operators
- CNN weights determine the PDE properties: order, type, stability,...
- Multiple channels $\rightsquigarrow$ coupled system of PDEs (one per channel)

Multi-Resolution Learning

level 1, 12×12 level 2, 24×24 level 3, 48×48 level 4, 96×96



Multi-Resolution Learning

Restrict the images n times
$\boldsymbol{\theta}^0, \mathbf{W}^0, \boldsymbol{\mu}^0 \leftarrow$ random initialization
for $j = 1 : n$ **do**
  optimize with data on level k starting from $\boldsymbol{\theta}^{j-1}, \mathbf{W}^{j-1}, \boldsymbol{\mu}^{k-1}$
  obtain $\boldsymbol{\theta}^*, \mathbf{W}^*, \boldsymbol{\mu}^*$
  $\boldsymbol{\theta}^j \leftarrow$ prolongate $\boldsymbol{\theta}^*$
  $\mathbf{W}^j \leftarrow$ prolongate $\mathbf{W}^*$

How to prolongate the kernels?

Galerkin Projection of Convolution Kernels

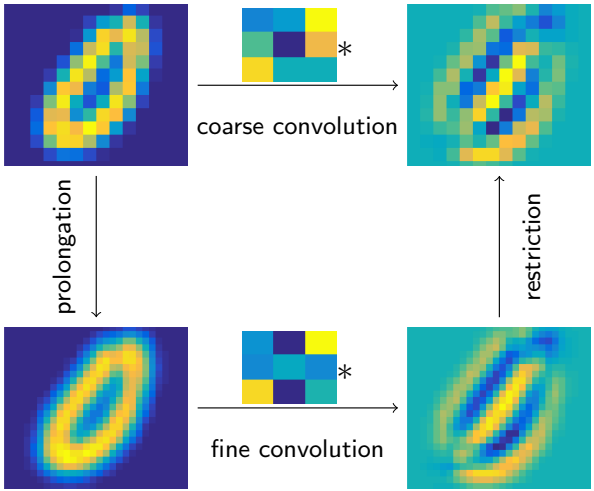
$$\mathbf{K}_H = \mathbf{R}\mathbf{K}_h\mathbf{P},$$

where

- $\mathbf{K}_h$ fine mesh operator (given)
- $\mathbf{R}$ restriction (e.g., averaging)
- $\mathbf{P}$ prolongation (e.g., interpolation)

Remarks:

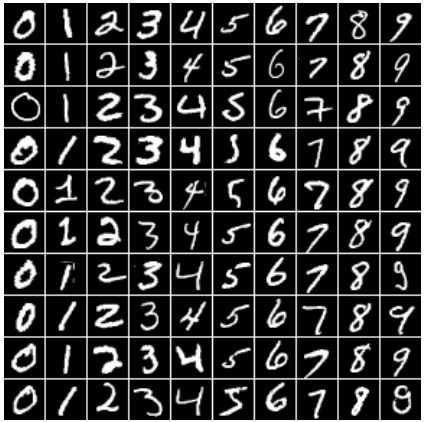
- Galerkin: $\mathbf{R} = \gamma\mathbf{P}^\top$
- Coarse $\rightarrow$ Fine: unique if kernel size constant.
- only small linear solve required



Example: Multiresolution Learning

data

- $60,000$ gray-scale images 18×18
- Residual Neural Network, width 6
- 2D convolution layers, fully connected
- $\tanh$ activation, softmax classifier



multilevel experiments

1. train on fine $\rightarrow$ classify coarse:

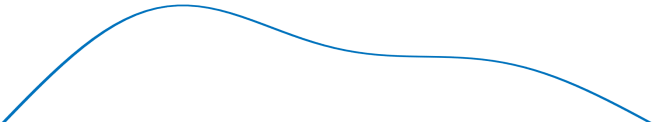
   84.1% (no restriction) vs. 94.9% (with restriction)

2. train on coarse $\rightarrow$ classify fine:

   61.0% (no prolongation) vs. 91.0% (with prolongation)

PDE-motivated Networks

Parabolic Residual Neural Networks



Recall the decay property of heat equation. Example:

$$\partial_t y(t,x) = \partial_{xx} y(t,x), \qquad \text{+ initial + boundary cond.}$$

Some consequences for learning:

- forward propagation is asymptotically stable
- theoretically, network is robust against perturbation of inputs (adversarial)
- learning problem ill-posed ($\rightsquigarrow$ do not integrate for too long)
- numerical methods for parabolic include multiscale, multilevel, ROM, ...

Parabolic CNN

In original Residual Net choose $\mathbf{K}_2 = -\mathbf{K}_1^\top = \mathbf{K}^\top$.
This gives parabolic PDE

$$\partial_t \mathbf{Y} = -\mathbf{K}(t)^\top \sigma(\mathbf{K}(t)\mathbf{Y} + \mathbf{b}(t)), \quad \mathbf{Y}(0) = \mathbf{Y}_0.$$

The Jacobian is

$$\mathbf{J}(t) = -\mathbf{K}(t)^\top \operatorname{diag}\left(\sigma'(\mathbf{K}(t)\mathbf{Y} + \mathbf{b}(t))\right) \mathbf{K}(t).$$

$\mathbf{J}$ is symmetric negative semidefinite ($\sigma' \geq 0$) $\rightsquigarrow$ stable if $\mathbf{K}, \mathbf{b}$ do not change too quickly. Use forward Euler discretization with h small enough

$$\mathbf{Y}_{j+1} = \mathbf{Y}_j - h\mathbf{K}_j^\top \sigma(\mathbf{K}_j\mathbf{Y} + \mathbf{b}_j), \quad j = 0, 1, \ldots, N-1$$

Similar to anisotropic diffusion (popular in image processing)

Y. Chen, T. Pock
Trainable Nonlinear Reaction Diffusion.
IEEE PAMI, 39(6), 1256–1272, 2017.

Theorem (LR, Haber[1] 2018)

If σ is monotonically non-decreasing, then the forward propagation is stable, i.e., there is a $M > 0$ (independent of T) such that

$$\|\mathbf{Y}(T) - \mathbf{Y}_\epsilon(T)\|_F \leq M\|\mathbf{Y}(0) - \mathbf{Y}_\epsilon(0)\|_F,$$

where $\mathbf{Y}$ and $\mathbf{Y}_\epsilon$ are solutions for different initial values.

For ease of notation, assume no bias. We show that

$$\partial_t\|\mathbf{Y}(t) - \mathbf{Y}_\epsilon(t)\|_F^2 \leq 0.$$

Integrating this over $[0, T]$ yields the stability result.
Why? Note that for all $t \in [0, T]$ taking derivative gives

$$\begin{aligned}\Big(-\mathbf{K}(t)^\top\sigma(\mathbf{K}(t)\mathbf{Y}) + \mathbf{K}(t)^\top\sigma(\mathbf{K}(t)\mathbf{Y}_\epsilon), \mathbf{Y} - \mathbf{Y}_\epsilon\Big)& \\ - \left(\sigma(\mathbf{K}(t)\mathbf{Y}) - \sigma(\mathbf{K}(t)\mathbf{Y}_\epsilon), \mathbf{K}(t)(\mathbf{Y} - \mathbf{Y}_\epsilon)\right) \quad &\leq 0.\end{aligned}$$

Where $(\cdot, \cdot)$ is inner product and we use $\sigma' \geq 0$.

[1] thanks to Martin Burger for pointing us to monotone operator theory

Hyperbolic Residual Neural Networks

Recall the reversibility of hyperbolic equations. Example:

$$\partial_{tt} y(t,x) = \partial_{xx} y(t,x), \qquad \text{+ initial + boundary cond.}$$

Similar property recently discovered for residual networks

$$\begin{aligned} \mathbf{y}_{k+1} &= \mathbf{y}_k + F(\mathbf{x}_k) \\ \mathbf{x}_{k+1} &= \mathbf{x}_k + G(\mathbf{y}_k) \end{aligned} \quad \longrightarrow \quad \begin{aligned} \mathbf{x}_{k-1} &= \mathbf{x}_k - G(\mathbf{y}_k) \\ \mathbf{y}_{k-1} &= \mathbf{y}_k - F(\mathbf{x}_k) \end{aligned}$$

useful for adjoint computations (backpropagation)

Reversible ResNets: ↓↓↓ memory ↑ computation


B.D. Nguyen, G.A. McMechan
Five ways to avoid storing source wavefield snapshots in 2D elastic prestack reverse time migration.
Geophysics, 2014.


A.N. Gomez, M. Ren, R. Urtasun, R.B. Grosse
The Reversible Residual Network: Backpropagation Without Storing Activations.
arXiv, 2017.

Hamiltonian CNN

Introducing auxiliary variable $\mathbf{Z}$, consider dynamics

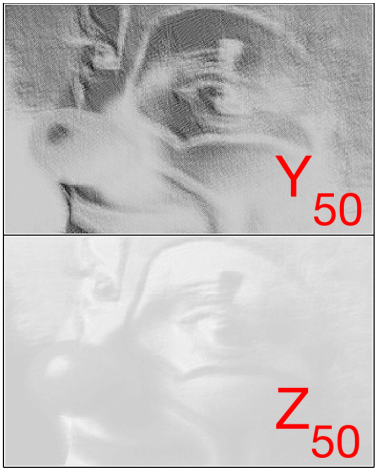
$$\partial_t \mathbf{Y}(t) = \mathbf{K}_1^T(t)\sigma(\mathbf{K}_1(t)\mathbf{Z}(t) + \mathbf{b}_1(t)),$$
$$\partial_t \mathbf{Z}(t) = -\mathbf{K}_2^T(t)\sigma(\mathbf{K}_2(t)\mathbf{Y}(t) + \mathbf{b}_2(t)).$$

In matrix form this is

$$\begin{pmatrix}\partial_t \mathbf{Y}\\ \partial_t \mathbf{Z}\end{pmatrix} = \begin{pmatrix}\mathbf{K}_1^T & 0\\ 0 & -\mathbf{K}_2^T\end{pmatrix}\sigma\Big(\begin{pmatrix}0 & \mathbf{K}_1\\ \mathbf{K}_2 & 0\end{pmatrix}\begin{pmatrix}\mathbf{Y}\\ \mathbf{Z}\end{pmatrix} + \begin{pmatrix}\mathbf{b}_1\\ \mathbf{b}_2\end{pmatrix}\Big).$$

(Can be shown that eigenvalues of Jacobian are all imaginary $\rightsquigarrow$ stability when $\mathbf{K}_1, \mathbf{K}_2, \mathbf{b}_1, \mathbf{b}_2$ change slowly in time) Discretize using Verlet method

$$\mathbf{Y}_{j+1} = \mathbf{Y}_j + h\mathbf{K}_{j1}^T\sigma(\mathbf{K}_{j1}\mathbf{Z}_j + \mathbf{b}_{j1}),$$
$$\mathbf{Z}_{j+1} = \mathbf{Z}_j - h\mathbf{K}_{j2}^T\sigma(\mathbf{K}_{j2}\mathbf{Y}_{j+1} + \mathbf{b}_{j2}).$$



Reversible Hamiltonian Neural Networks

Forward propagation in double-layer Hamiltonian

$$\mathbf{Y}_{j+1} = \mathbf{Y}_j + h\mathbf{K}_{j1}^T \sigma(\mathbf{K}_{j1}\mathbf{Z}_j + \mathbf{b}_{j1}),$$
$$\mathbf{Z}_{j+1} = \mathbf{Z}_j - h\mathbf{K}_{j2}^T \sigma(\mathbf{K}_{j2}\mathbf{Y}_{j+1} + \mathbf{b}_{j2}).$$

Recall: Antisymmetric structure gives stability (when parameters change slowly).

Clearly, given $\mathbf{Y}_N$ and $\mathbf{Z}_N$ dynamics can be computed backwards: For $j = N-1, N-2, \ldots, N$ do

$$\mathbf{Z}_j = \mathbf{Z}_{j+1} + h\mathbf{K}_{j2}^T \sigma(\mathbf{K}_{j2}\mathbf{Y}_{j+1} + \mathbf{b}_{j2})$$
$$\mathbf{Y}_j = \mathbf{Y}_{j+1} - h\mathbf{K}_{j1}^T \sigma(\mathbf{K}_{j1}\mathbf{Z}_j + \mathbf{b}_{j1}),$$

Possible to recompute weights, ↑50% computation costs, but large memory savings + stability

A. Mahendran, A Vedaldi
Understanding deep image representations by inverting them.
CVPR, 2015.

B. Chang, L Meng, E. Holtham, E. Haber, LR, D Begert
Reversible Architectures for Arbitrarily Deep ResNNs.
AAAI, 2018.

Second-Order Network

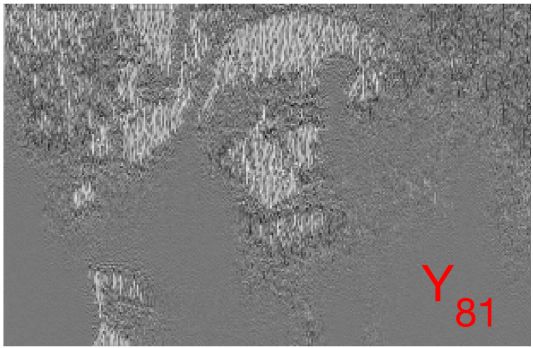
Consider second-order forward dynamics

$$\partial_{tt}\mathbf{Y} = -\mathbf{K}(t)^\top \sigma(\mathbf{K}(t)\mathbf{Y} + b(t))$$

And their Leapfrog discretization

$$\mathbf{Y}_{j+1} = 2\mathbf{Y}_j - \mathbf{Y}_{j-1} - h^2 \mathbf{K}_j^\top \sigma(\mathbf{K}_j\mathbf{Y} + b_j)$$

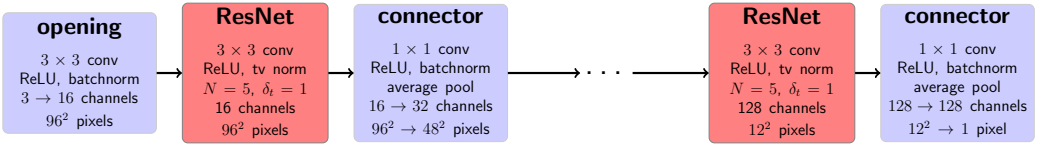
Similar to: Full Waveform Inversion, Ultrasound, . . .



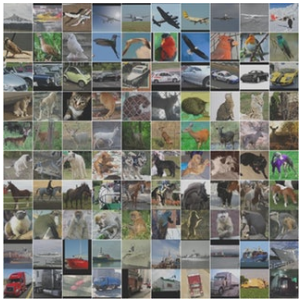
Note: architecture is reversible.

Numerical Results

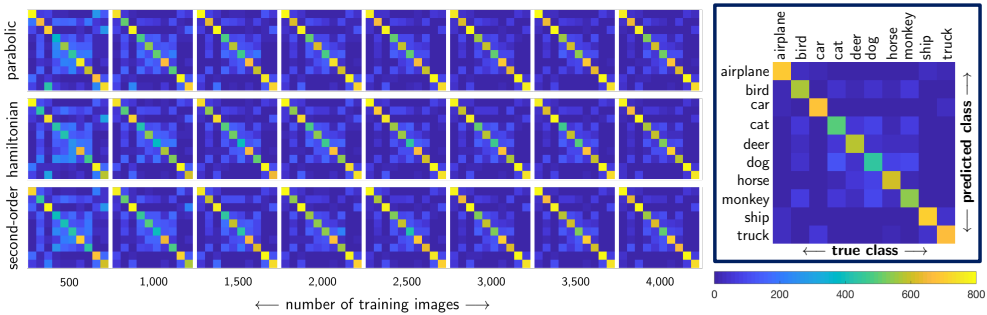
Example (STL-10): Comparison of Architectures



- ResNet $\in$ { Parabolic, Second-order, Hamiltionian }
- left to right: resolution $\downarrow$ number of channels $\uparrow$.
- STL-10: 5k training images, 8k test images, 10 classes
- training via stochastic gradient descent, momentum$= 0.9$
- regularization: smoothness in time (tv regularizer)

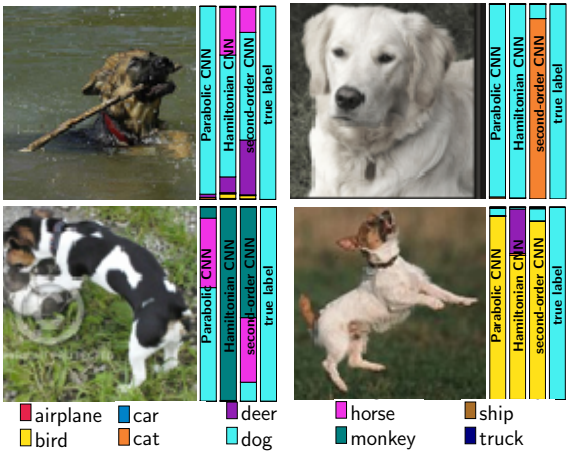


Example (STL-10): Increasing Training Data



Example (STL-10): Comparison of Architectures

	#weights	test accuracy
Parabolic	1.01M	80.9%
Hamiltonian	0.52M	80.4%
Hamiltonian	1.28M	85.5%
Second-order	1.01M	81.0%
Second-order	2.44M	84.6%



different architectures give overall competitive results

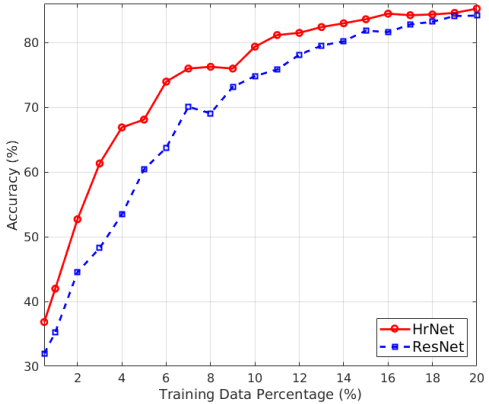
individual results may vary

B Chang et al.,
Reversible Architectures for Deep ResNNs.
AAAI, 2018.

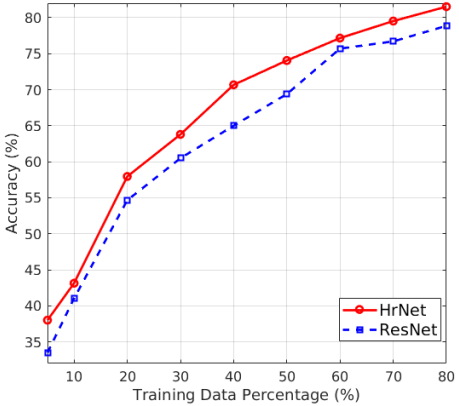
LR, E Haber
Deep Neural Networks motivated by PDEs.
arXiv, 2018.

Experiment (CIFAR-10 and STL-10): Stability and Generalization

Goal: Compare efficiency of reversible Hamiltonian CNN to ResNN



CIFAR10 (total: 50k images)

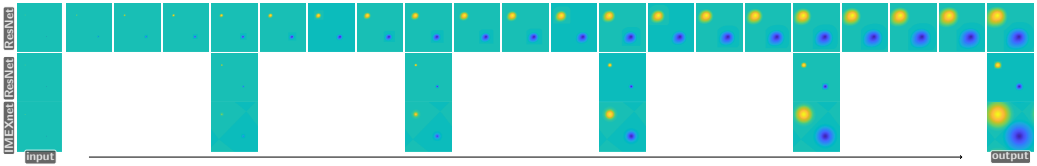


STL10 (total: 5k images)

stability leads to improved generalization

Semi-Implicit Networks

IMEXnet - Forward Stable DNN



Goals: Gain stability and address field-of-view problem with only few layers

$$\partial_t \mathbf{y}(t) = \underbrace{f(\mathbf{y}(t), \theta(t)) + \mathbf{L}\mathbf{y}(t)}_{\text{explicit}} - \underbrace{\mathbf{L}\mathbf{y}(t)}_{\text{implicit}},$$

for some $\mathbf{L}$ (symmetric positive definite and easy to invert).
Discretization leads to implicit-explicit forward propagation

$$\mathbf{Y}_{j+1} = (\mathbf{I} + h\mathbf{L}_j)^{-1}(\mathbf{Y}_j + h\mathbf{L}\mathbf{Y}_j + hf(\mathbf{Y}_j, \theta_j)),$$

C.B. Schönlieb, A. Bertozzi.
Unconditionally stable schemes for higher order inpainting.
Commun. Math. Sci., 9: 413–457, 2011.

E Haber, K Lensik, E. Treister, LR
IMEXnet - A Forward Stable Deep Neural Network.
ICML 2019.

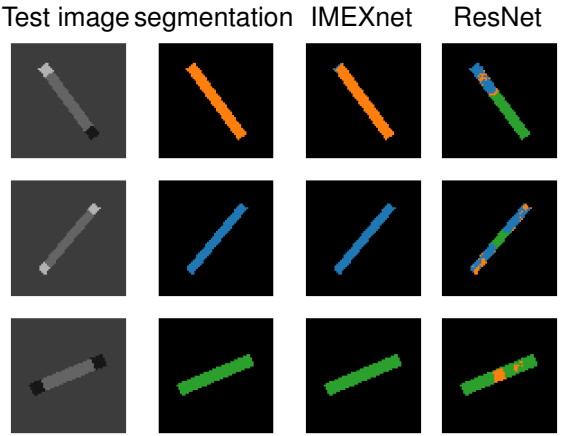
Example: Semantic Segmentation

Goal: Label each pixel in an image

Challenges:

- high-resolution output
- limited field of view
- synthetic Qtips data

E Haber, K Lensik, E. Treister, LR
IMEXnet - A Forward Stable Deep Neural Network.
ICML 2019.



Lean Convolutional Neural Networks

LeanResNet: Low-cost Convolutional Residual Networks

Commonly, convolutions couple all feature channels.

Example: Let $\mathbf{C}(\boldsymbol{\theta})$ be spatial convolution with stencil $\boldsymbol{\theta}$ and assume four inputs and outputs. Idea: Use sparser coupling patterns.

Next, reduce number of elements in stencil

$$\begin{bmatrix} \boldsymbol{\theta}_{1,1} & \boldsymbol{\theta}_{1,2} & \boldsymbol{\theta}_{1,3} \\ \boldsymbol{\theta}_{2,1} & \boldsymbol{\theta}_{2,2} & \boldsymbol{\theta}_{2,3} \\ \boldsymbol{\theta}_{3,1} & \boldsymbol{\theta}_{3,2} & \boldsymbol{\theta}_{3,3} \end{bmatrix} \longrightarrow \begin{bmatrix} & \boldsymbol{\theta}_{1,2} & \\ \boldsymbol{\theta}_{2,1} & \boldsymbol{\theta}_{2,2} & \boldsymbol{\theta}_{2,3} \\ & \boldsymbol{\theta}_{3,2} & \end{bmatrix}$$

Motivation: Reaction-diffusion PDEs.

Clearly, fewer parameters and flops, but how effective is it?

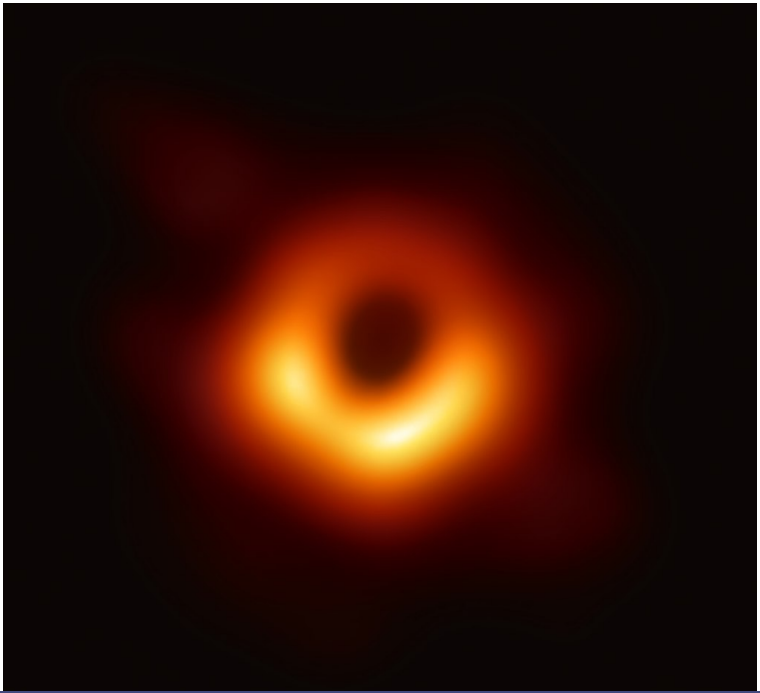
LeanResNet: Low-cost Convolutional Residual Networks

Architecture	Network	Params \ FLOPs[M]	Acc
		CIFAR10	
ResNet	Res24	4.3 \ 241	94.54%
MobileNetV2	Res24	0.5 \ 33	91.71%
ShuffleNetV2	0.5X	0.4 \ 42	91.56%
ShiftResNet	Res24	0.5 \ 31	92.50%
LeanResNet	Res24	0.5\ 29	**92.98**%
		CIFAR100	
ResNet	Res40	27.0 \ 1529	78.5%
MobileNetV2	Res40	3.1 \ 167	71.94%
ShuffleNetV2	1.5X	2.6 \ 375	74.2%
ShiftResNet	Res40	3.1 \ 201	74.2%
LeanResNet	Res40	2.9\ 179	**74.3**%
		TinyImageNet200	
ResNet	Res38	34.6 \ 6899	65.18%
MobileNetV2	1.4	4.7 \ 661	48.68%
ShuffleNetV2	2.0X	5.7 \ 740	58.39%
ShiftResNet	Res38	4.5 \ 793	61.82%
LeanResNet	Res38	4.6 \ 712	**62.58**%

J Ephrath, LR, E Haber, E Treister
LeanResNet: A Low-cost yet Effective Convolutional Residual Networks.
ICML Workshop on On-Device ML, 2019.

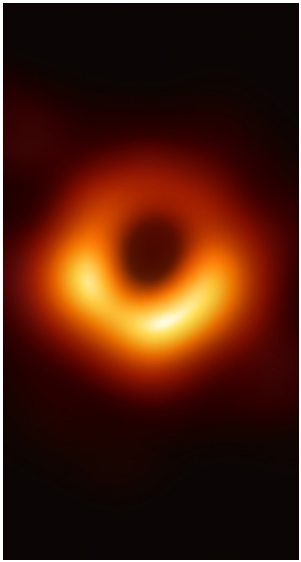
Conclusion

Some Perspectives



Analogies in PDE-Based Image Processing

- A few seminal works
  - optical flow (Horn&Schunck, 1981)
  - nonlinear diffusion for denoising (Perona&Malik, 1990, Weickert 2009)
  - nonlinear edge-preserving denoising (Rudin, Osher, Fatemi, 1992)
  - variational methods for image segmentation (Mumford&Shah, 1989)
- The common thread
  - model images as functions $I : \mathbb{R}^d \to \mathbb{R}^c$
    $\rightsquigarrow$ abstract from resolution
  - interpret operations on images as PDE operators
  - create insight by analyzing underlying PDE (existence, uniqueness, regularity)
  - use efficient PDE solvers for image processing



K Akiyama et al.
Imaging the Schwarzschild-radius-scale Structure of M87 with the Event Horizon Telescope using Sparse Modeling.
arXiv:1702.07361 [astro-ph.IM]

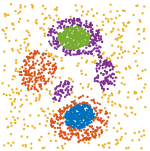
Σ: Deep Neural Networks motivated by PDEs

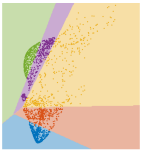
Optimal control formulation

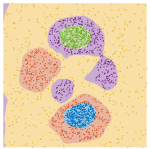
- new insights, theory, algorithms
- stability and impact on convergence/generalization

Properties of PDE-inspired CNNs

- parabolic: numerical stability, multiscale, . . .
- hyperbolic: reversible, preserve high-frequency features,. . .
- IMEX: stability, global coupling in feature space
- Lean: drastical reduction in #weights and flops
- all: performance matches/outperforms existing methods ($\rightsquigarrow$ Sensitivity analysis to select architecture)







Mission: Make CNNs stable, well-posed, and efficient enough to enable scientific machine learning

E Haber, LR
Stable Architectures for DNNs.
Inverse Problems, 2017.

E Holtham et al.
Learning Across Scales.
AAAI, 2018.

B Chang et al.,
Reversible Architectures for Deep ResNNs.
AAAI, 2018.

LR, E Haber
Deep Neural Networks motivated by PDEs.
arXiv, 2018.

Team and Acknowledgments


Lili Meng


Bo Chang


Elliot Holtham


Eldad Haber


Samy Wu Fung


Eran Treister


Keegan Lensik


Jacob Schroder


Eric Cyr


Stefanie Günther

Funding: DMS 1522599, DMS 1751636 US-Israel BSF 2018209

Other support: Donation of a TITAN X GPU