LIMITING THE DEEP LEARNING

YIPING LU PEKING UNIVERSITY
DEEP LEARNING IS SUCCESSFUL, \textbf{BUT} …
LARGER THE BETTER?

Test error decreasing
LARGER THE BETTER?

Why Not Limiting To $\frac{p}{n} = +\infty$
TWO LIMITING

Input Layer  |  Hidden Layers  |  Output Layer

Depth Limiting
DEPTH LIMITING: ODE

[He et al. 2015] [E. 2017] [Haber et al. 2017] [Lu et al. 2017] [Chen et al. 2018]
BEYOND FINITE LAYER NEURAL NETWORK

Going into infinite layer

Differential Equation
Deep Limits of Residual Neural Networks
Matthew Thorpe¹ and Yves van Gennip²

¹Department of Applied Mathematics and Theoretical Physics,
University of Cambridge,
Cambridge, CB3 0WA, UK

²Delft Institute of Applied Mathematics,
Delft University of Technology,
2628 XE Delft, The Netherlands

March 2019

Abstract
Neural networks have been very successful in many applications; we often, however, lack a theoretical understanding of what the neural networks are actually learning. This problem emerges when trying to generalise to new data sets. The contribution of this paper is to show that, for the residual neural network model, the deep layer limit coincides with a parameter estimation problem for a nonlinear ordinary differential equation. In particular, while it is known that the residual neural network model is a discretisation of an ordinary differential equation, we show convergence in a variational sense. This implies that optimal parameters converge in the deep layer limit. This is a stronger statement than saying for a fixed parameter the residual neural network model converges (the latter does not in general imply the former). Our variational analysis provides a discrete-to-continuum $\Gamma$-convergence result for the objective function of the residual neural network training step to a variational problem constrained by a system of ordinary differential equations; this rigorously connects the discrete setting to a continuum problem.
TRADITIONAL WISDOM IN DEEP LEARNING

Milk + Coffee = ?

Lollipop + Diagram = ?
Neural Ordinary Differential Equations

Ricky T. Q. Chen*, Yulia Rubanova*, Jesse Bettencourt*, David Duvenaud
University of Toronto, Vector Institute
Toronto, Canada
ertgchen, rubanova, jessebett, duvenaud@cs.toronto.edu

Abstract

We introduce a new family of deep neural network models. Instead of specifying a discrete sequence of hidden layers, we parameterize the derivative of the hidden state using a neural network. The output of the network is computed using a black-box differential equation solver. These continuous-depth models have constant memory cost, adapt their evaluation strategy to each input, and can explicitly trade numerical precision for speed. We demonstrate these properties in continuous-depth residual networks and continuous-time latent variable models. We also construct continuous normalizing flows, a generative model that can train by maximum likelihood, without partitioning or ordering the data dimensions. For training, we show how to scalably backpropagate through any ODE solver, without access to its internal operations. This allows end-to-end training of ODEs within larger models.
BRIDGING **CONTROL** AND **LEARNING**

Feedback is the learning loss
BRIDGING **CONTROL** AND **LEARNING**

Feedback is the learning loss

Interpretability? PDE-Net ICML2018
BRIDGING **CONTROL AND LEARNING**

Feedback is the learning loss

Interpretability?  
- **PDE-Net** ICML2018
- **LM-ResNet** ICML2018
- **DURR** ICLR2019
- **Macaroon** submitted

Feedback is the learning loss

Neural Network

Interpretability?

PDE-Net ICML2018

A better model?

LM-ResNet ICML2018

New Algorithm?

YOPO Submitted


HOW DIFFERENTIAL EQUATION VIEW HELPS NEURAL NETWORK DESIGNING

PRINCIPLED NEURAL ARCHITECTURE DESIGN
NEURAL NETWORK AS SOLVING ODES

Dynamic System  Nueral Network

Continuous limit  Numerical Approximation

Table 1: In this table, we list a few popular deep networks, their associated ODEs and the numerical schemes that are connected to the architecture of the networks.

<table>
<thead>
<tr>
<th>Network</th>
<th>Related ODE</th>
<th>Numerical Scheme</th>
</tr>
</thead>
<tbody>
<tr>
<td>ResNet, ResNeXt, etc.</td>
<td>$u_t = f(u)$</td>
<td>Forward Euler scheme</td>
</tr>
<tr>
<td>PolyNet</td>
<td>$u_t = f(u)$</td>
<td>Approximation of backward Euler scheme</td>
</tr>
<tr>
<td>FractalNet</td>
<td>$u_t = f(u)$</td>
<td>Runge-Kutta scheme</td>
</tr>
<tr>
<td>RevNet</td>
<td>$\dot{X} = f_1(Y)$, $\dot{Y} = f_2(X)$</td>
<td>Forward Euler scheme</td>
</tr>
</tbody>
</table>

**WRN, ResNeXt, Inception-ResNet, PolyNet, SENet** etc…… : New scheme to Approximate the right hand side term

Why not change the way to discrete $u_t$?
MULTISTEP ARCHITECTURE?

\[ x_{n+1} = x_n + f(x_n) \]

\[ x_t = f(x) \]
MULTISTEP ARCHITECTURE?

**Linear Multi-step Scheme**

\[ x_t = f(x) \]

\[ x_{n+1} = (1 - k_n)x_n + k_nx_{n-1} + f(x_n) \]

**Linear Multi-step Residual Network**

\[ x_{n+1} = x_n + f(x_n) \]
Multistep Architecture?

Linear Multi-step Scheme

\[ x_t = f(x) \]

\[ x_{n+1} = (1 - k_n)x_n + k_n x_{n-1} + f(x_n) \]

Only One More Parameter

Linear Multi-step Residual Network

\[ x_{n+1} = x_n + f(x_n) \]
### Table 2: Linear Multi-step Resnet Test On Cifar

<table>
<thead>
<tr>
<th>Model</th>
<th>Layer</th>
<th>Accuracy</th>
<th>Params</th>
<th>Dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resnet</td>
<td>20</td>
<td>91.25</td>
<td>0.27M</td>
<td>Cifar10</td>
</tr>
<tr>
<td>Resnet</td>
<td>32</td>
<td>92.49</td>
<td>0.46M</td>
<td>Cifar10</td>
</tr>
<tr>
<td>Resnet</td>
<td>44</td>
<td>92.83</td>
<td>0.66M</td>
<td>Cifar10</td>
</tr>
<tr>
<td>Resnet</td>
<td>56</td>
<td>93.03</td>
<td>0.85M</td>
<td>Cifar10</td>
</tr>
<tr>
<td>Resnet</td>
<td>110</td>
<td>93.63</td>
<td>1.7M</td>
<td>Cifar10</td>
</tr>
<tr>
<td>LM-Resnet(Ours)</td>
<td>20</td>
<td>91.67</td>
<td>0.27M</td>
<td>Cifar10</td>
</tr>
<tr>
<td>LM-Resnet(Ours)</td>
<td>32</td>
<td>92.82</td>
<td>0.46M</td>
<td>Cifar10</td>
</tr>
<tr>
<td>LM-Resnet(Ours)</td>
<td>44</td>
<td>92.98</td>
<td>0.66M</td>
<td>Cifar10</td>
</tr>
<tr>
<td>LM-Resnet(Ours)</td>
<td>56</td>
<td>93.69</td>
<td>0.85M</td>
<td>Cifar10</td>
</tr>
<tr>
<td>EM-Resnet(Ours)</td>
<td>40</td>
<td>91.75</td>
<td>0.27M</td>
<td>Cifar10</td>
</tr>
</tbody>
</table>

### Table 3: Single-crop error rate on ImageNet (validation set)

<table>
<thead>
<tr>
<th>Model</th>
<th>Layer</th>
<th>top-1</th>
<th>top-5</th>
</tr>
</thead>
<tbody>
<tr>
<td>ResNet (He et al. (2015b))</td>
<td>50</td>
<td>24.7</td>
<td>7.8</td>
</tr>
<tr>
<td>ResNet (He et al. (2015b))</td>
<td>101</td>
<td>23.6</td>
<td>7.1</td>
</tr>
<tr>
<td>ResNet (He et al. (2015b))</td>
<td>152</td>
<td>23.0</td>
<td>6.7</td>
</tr>
<tr>
<td>LM-ResNet (Ours)</td>
<td>50, pre-act</td>
<td>23.8</td>
<td>7.0</td>
</tr>
<tr>
<td>LM-ResNet (Ours)</td>
<td>101, pre-act</td>
<td>22.6</td>
<td>6.4</td>
</tr>
</tbody>
</table>

#### Adjusted Models

- **ResNet**: 110 layers, accuracy 72.24%, 1.7M parameters, Cifar10 dataset.
- **ResNet**: 164 layers, accuracy 75.67%, 2.55M parameters, Cifar10 dataset.
- **ResNetXt**: 29 layers, accuracy 89.32%, 34.4M parameters, Cifar10 dataset.
- **ResNetXt**: 29 layers, accuracy 83.21%, 68.1M parameters, Cifar10 dataset.
- **LM-Resnet(Ours)**: 110 layers, accuracy 73.16%, 1.7M parameters, Cifar10 dataset.
- **LM-Resnet(Ours)**: 164 layers, accuracy 76.74%, 2.55M parameters, Cifar10 dataset.
- **LM-ResneXt(Ours)**: 29 layers, accuracy 82.51%, 34.4M parameters, Cifar10 dataset.
- **LM-ResneXt(Ours)**: 29 layers, accuracy 83.21%, 68.1M parameters, Cifar10 dataset.
PLOT THE MOMENTUM

Analysis by zero stability
\[ x_{n+1} = (1 - k_n)x_n + k_n x_{n-1} + \Delta t f(x_n) \]

\[ (1 + k_n) \dot{u} + \left( 1 - k_n \right) \frac{\Delta t}{2} \ddot{u}_n + o(\Delta t^3) = f(u) \]

[Su et al. 2016] [Dong et al. 2017]
Noise can avoid overfit?

\[ \dot{X}(t) = f(X(t), a(t)) + g(X(t), t)dB_t, X(0) = X_0 \]

The numerical scheme is only need to be **weak convergence**!

\[ E_{data}(loss(X(T))) \]
STOCHASTIC DEPTH AS AN EXAMPLE

\[ x_{n+1} = x_n + \eta_n f(x) = x_n + E\eta_n f(x_n) + (\eta_n - E\eta_n)f(x_n) \]

\[ \sqrt{p(t)(1-p(t))f(X) \odot [1_{N \times 1}, 0_{N,N-1}]dB_t} \]

Fig. 2. The linear decay of \( p_t \) illustrated on a ResNet with stochastic depth for \( p_0 = 1 \) and \( p_L = 0.5 \). Conceptually, we treat the input to the first ResBlock as \( H_0 \), which is always active.
Natural Language
**Idea:** Consider every *word* in a document as a *particle* in the n-body system.

Using a Neural Network to extract the feature of document!
TRANSFORMER AS A SPLITTING SCHEME

Attention Is All You Need

FFN: advection

Self-Attention: particle interaction
A BETTER SPLITTING SCHEME

True solution: \( u(t + \Delta t) = e^{\Delta t(A+B)}u(t) \)
Lie splitting: \( u_L(t + \Delta t) = e^{\Delta t A}e^{\Delta t B}u(t) \)
Strang splitting: \( u_S(t + \Delta t) = e^{\frac{1}{2}\Delta t A}e^{\Delta t B}e^{\frac{1}{2}\Delta t A}u(t) \)

(a) Original Transformer

(b) Oreo Layers
Table 1: Performance on the testsets of WMT14 En-De and IWSLT14 De-En tasks.

<table>
<thead>
<tr>
<th>Method</th>
<th>IWSLT14 De-En</th>
<th>WMT14 En-De</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>small</td>
<td>base</td>
</tr>
<tr>
<td>Transformer [3]</td>
<td>34.4</td>
<td>27.3</td>
</tr>
<tr>
<td>Relative Transformer [31]</td>
<td>/</td>
<td>26.8</td>
</tr>
<tr>
<td>Scaling NMT [32]</td>
<td>/</td>
<td>/</td>
</tr>
<tr>
<td>Dynamic Conv [33]</td>
<td>35.2</td>
<td>/</td>
</tr>
<tr>
<td><strong>Ours</strong></td>
<td><strong>35.43</strong></td>
<td><strong>28.91</strong></td>
</tr>
</tbody>
</table>

Table 2: The test results on the GLUE benchmark (except WNLI).

<table>
<thead>
<tr>
<th>Method</th>
<th>CoLA</th>
<th>SST-2</th>
<th>MRPC</th>
<th>STS-B</th>
<th>QQP</th>
<th>MNLI-m/mm</th>
<th>QNLI</th>
<th>RTE</th>
<th>GLUE</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Existing systems</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ELMo [8]</td>
<td>33.6</td>
<td>90.4</td>
<td>84.4/78.0</td>
<td>74.2/72.3</td>
<td>63.1/84.3</td>
<td>74.1/74.5</td>
<td>79.8</td>
<td>58.9</td>
<td>70.0</td>
</tr>
<tr>
<td>OpenAI GPT [35]</td>
<td>47.2</td>
<td>93.1</td>
<td>87.7/83.7</td>
<td>85.3/84.8</td>
<td>70.1/88.1</td>
<td>80.7/80.6</td>
<td>87.2</td>
<td>69.1</td>
<td>76.9</td>
</tr>
<tr>
<td>BERT_BASE [7]</td>
<td>52.1</td>
<td>93.5</td>
<td><strong>88.9/84.8</strong></td>
<td>87.1/85.8</td>
<td><strong>71.2/89.2</strong></td>
<td>84.6/83.4</td>
<td>90.5</td>
<td>66.4</td>
<td>78.3</td>
</tr>
<tr>
<td><strong>Our systems</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BERT_BASE (ours)</td>
<td>52.8</td>
<td>92.8</td>
<td>87.3/83.0</td>
<td>81.2/80.0</td>
<td>70.2/88.4</td>
<td>84.4/83.7</td>
<td>90.4</td>
<td>64.9</td>
<td>77.4</td>
</tr>
<tr>
<td><strong>Ours</strong></td>
<td><strong>57.6</strong></td>
<td><strong>94.0</strong></td>
<td><strong>88.4/84.4</strong></td>
<td><strong>87.5/86.3</strong></td>
<td><strong>70.8/89.0</strong></td>
<td><strong>85.4/84.5</strong></td>
<td><strong>91.6</strong></td>
<td><strong>70.5</strong></td>
<td><strong>79.7</strong></td>
</tr>
</tbody>
</table>
Computer Vision
ONE NOISE LEVEL ONE NET

Network 1 \( \sigma = 25 \)
ONE NOISE LEVEL ONE NET

Network 1  \( \sigma = 25 \)

Network 2  \( \sigma = 35 \)
WE WANT

One Model
WE ALSO WANT GENERALIZATION

BM3D

DnCNN
RETHINKING TRADITIONAL FILTERING APPROACH

Perona-Malik Equation

\[
\begin{align*}
\frac{\partial u}{\partial t} &= \text{div} \left( e(|\nabla u|^2) \nabla u \right) \text{ in } \Omega \times (0, T), \\
\frac{\partial u}{\partial N} &= 0 \text{ on } \partial \Omega \times (0, T), \\
u(0, x) &= u_0(x) \text{ in } \Omega,
\end{align*}
\]
RETHINKING TRADITIONAL FILTERING APPROACH

Perona-Malik Equation

\[
\begin{align*}
\frac{\partial u}{\partial t} &= \text{div} \left( c(|\nabla u|^2) \nabla u \right) \text{ in } \Omega \times (0, T), \\
\frac{\partial u}{\partial N} &= 0 \text{ on } \partial\Omega \times (0, T), \\
u(0, x) &= u_0(x) \text{ in } \Omega,
\end{align*}
\]
RETHINKING TRADITIONAL FILTERING APPROACH

Perona-Malik Equation

\[ \begin{aligned}
\frac{\partial u}{\partial t} &= \text{div} \left( e(|\nabla u|^2) \nabla u \right) \text{ in } \Omega \times (0, T), \\
\frac{\partial u}{\partial N} &= 0 \text{ on } \partial \Omega \times (0, T), \\
u(0, x) &= u_0(x) \text{ in } \Omega,
\end{aligned} \]
RETHINKING TRADITIONAL FILTERING APPROACH

Perona-Malik Equation

\[
\begin{align*}
\frac{\partial u}{\partial t} &= \text{div} \left( e(|\nabla u|^2) \nabla u \right) \text{ in } \Omega \times (0, T), \\
\frac{\partial u}{\partial N} &= 0 \text{ on } \partial \Omega \times (0, T), \\
u(0, x) &= u_0(x) \text{ in } \Omega,
\end{align*}
\]

Hello World

input

Noisy

Over-smooth

processing
MOVING ENDPOINT CONTROL VS FIXED ENDPOINT CONTROL

\[
\begin{align*}
\min_{w, \tau} & \quad L(X(\tau), y) + \int_{0}^{\tau} R(w(t), t) dt \\
\text{s.t.} & \quad \dot{X} = f(X(t), w(t)), t \in (0, \tau) \\
& \quad X(0) = x_0.
\end{align*}
\]

\[
\min_{\tau} \int_{0}^{\tau} R(w(t), t) dt \\
\text{s.t.} \quad \dot{X} = f(X(t), w(t)),
\]

\(\tau\) is the first time that dynamic meets \(X\).

\(\tau\) is the time
arrives the moon

Control dynamic:
Physic Law
Figure 9: Denoising results of images from BSD68 with extreme noise conditions ($\sigma = 95$).
Physics
PHYSICS DISCOVERY

Our Work
CONV FILTERS AS DIFFERENTIAL OPERATORS

**Proposition 2.1.** Let $q$ be a filter with sum rules of order $\alpha \in \mathbb{Z}_+^2$. Then for a smooth function $F(x)$ on $\mathbb{R}^2$, we have

$$\frac{1}{\varepsilon^{|\alpha|}} \sum_{k \in \mathbb{Z}^2} q[k] F(x + \varepsilon k) = C_\alpha \frac{\partial^\alpha}{\partial x^\alpha} F(x) + O(\varepsilon), \text{ as } \varepsilon \to 0,$$

where $C_\alpha$ is the constant defined by

$$C_\alpha = \frac{1}{\alpha!} \sum_{k \in \mathbb{Z}^2} k^\alpha q[k].$$

If, in addition, $q$ has total sum rules of order $K \setminus \{|\alpha| + 1\}$ for some $K > |\alpha|$, then

$$\frac{1}{\varepsilon^{|\alpha|}} \sum_{k \in \mathbb{Z}^2} q[k] F(x + \varepsilon k) = C_\alpha \frac{\partial^\alpha}{\partial x^\alpha} F(x) + O(\varepsilon^{K-|\alpha|}), \text{ as } \varepsilon \to 0.$$ 

(3) \hspace{5cm} (4)

$$\Delta u = u_{xx} + u_{yy}$$
\[ u_t = F(x, u, \nabla u, \nabla^2 u, \ldots), \quad x \in \Omega \subset \mathbb{R}^2, \quad t \in [0, T]. \]
- Constrain the function space
- Theoretical Recover Guarantee (Coming Soon)
- Symbolic Discovery

Table 1: PDE model identification.

<table>
<thead>
<tr>
<th>Correct PDE</th>
<th>Frozen-PDE-Net 2.0</th>
<th>PDE-Net 2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>$u_t = -uu_x - vv_y + 0.05(u_{xx} + u_{yy})$</td>
<td>$u_t = -0.906uu_x - 0.901vv_y + 0.033u_{xx} + 0.037u_{yy}$</td>
<td>$u_t = -0.986uu_x - 0.972vv_y + 0.054u_{xx} + 0.052u_{yy}$</td>
</tr>
<tr>
<td>$v_t = -uv_x - vv_y + 0.05(v_{xx} + v_{yy})$</td>
<td>$v_t = -0.907ev_y - 0.902uv_x + 0.039v_{xx} + 0.032v_{yy}$</td>
<td>$v_t = -0.984uv_x - 0.982vv_y + 0.055v_{xx} + 0.050v_{yy}$</td>
</tr>
</tbody>
</table>
ON GOING WORK

Shock Wave Architecture Search

Yufei Wang, Ziju Shen, Zichao Long, Bin Dong Learning to Solve Conservation Laws
HOW DIFFERENTIAL EQUATION VIEW HELPS OPTIMIZATION

CONSIDERING THE NEURAL NETWORK STRUCTURE
RELATED WORKS

- Maximal Principle [Li et al. 2018a] [Li et al. 2018b]
- Adjoin Method [Chen et al. 2018]
- Multigrid [Chang et al. 2017]
- Domain decomposition

- Our Work: ODE captures the composition structure of neural network!
TOWARDS DEEP LEARNING MODELS RESISTANT TO ADVERSARIAL ATTACKS

\[
\min_{\theta} \rho(\theta), \quad \text{where} \quad \rho(\theta) = \mathbb{E}_{(x,y) \sim D} \left[ \max_{\delta \in \mathcal{S}} L(\theta; x + \delta, y) \right]
\]

Problem:
- More capacity and stronger adversaries decrease transferability. Always 10 times wider
- PGD training is expansive!

Can adversarial training cheaper
TAKE NEURAL NETWORK ARCHITECTURE INTO CONSIDERATION

Previous Work

Adversary updater

Heavy gradient calculation

Black box
TAKE NEURAL NETWORK ARCHITECTURE INTO CONSIDERATION

Previous Work

Adversary updater

Heavy gradient calculation

Black box

YOPO

Adversary updater
\[
\min_{\theta} \max_{\|\eta\|_{\infty} \leq \epsilon} J(\theta, \eta) := \frac{1}{N} \sum_{i=1}^{N} \ell_i(x_{i,T}) + \frac{1}{N} \sum_{i=1}^{N} \sum_{t=0}^{T-1} R_t(x_{i,t}; \theta_t)
\]
subject to \( x_{i,1} = f_0(x_{i,0} + \eta_i; \theta_0), i = 1, 2, \ldots, N \)
\[
x_{i,t+1} = f_t(x_{i,t}, \theta_t), t = 1, 2, \ldots, T - 1
\]
DIFFERENTIAL GAME

\[
\min_{\theta} \max_{\|\eta\|_{\infty} \leq \epsilon} J(\theta, \eta) := \frac{1}{N} \sum_{i=1}^{N} \ell_i(x_{i,T}) + \frac{1}{N} \sum_{i=1}^{N} \sum_{t=0}^{T-1} R_t(x_{i,t}; \theta_t)
\]

subject to \( x_{i,1} = f_0(x_{i,0} + \eta; \theta_0), i = 1, 2, \ldots, N \)

\( x_{i,t+1} = f_t(x_{i,t}, \theta_t), t = 1, 2, \ldots, T - 1 \)

(2)
DIFFERENTIAL GAME

\[
\min_{\theta} \max_{\|\eta\|_\infty \leq \epsilon} J(\theta, \eta) := \frac{1}{N} \sum_{i=1}^{N} \ell_i(x_{i,T}) + \frac{1}{N} \sum_{i=1}^{N} \sum_{t=0}^{T-1} R_t(x_{i,t}; \theta_t)
\]

subject to:

\[
x_{i,1} = f_0(x_{i,0} + \eta_i; \theta_0), \quad i = 1, 2, \ldots, N
\]

\[
x_{i,t+1} = f_t(x_{i,t}, \theta_t), \quad t = 1, 2, \ldots, T - 1
\]
DECOUPLE TRAINING

- Synthetic gradients [Jaderberg et al. 2017]
- Lifted Neural Network [Askari et al. 2018] [Gu et al. 2018] [Li et al. 2019]
- Delayed Gradient [Huo et al. 2018]
- Block Coordinate Descent Approach [Lau et al. 2018]

- Can Control perspective helps us to understand decoupling?
- Our idea: Decouple the gradient back propagation with the adversary updating.
YOPO (YOU ONLY PROPAGATE ONCE)
YOPO (YOU ONLY PROPAGATE ONCE)
YOPO (YOU ONLY PROPAGATE ONCE)
WHY DECOUPLING

Theorem 1. (PMP for adversarial defense) There exists co-state processes $p_{s,t}^* = p_{s,t}^* : t = 0, \cdots , T$ such that the following holds for all $t \in [T]$ and $s \in [S]$:  

\begin{align}
  x_{s,t+1}^* &= \nabla_p H_t(x_{s,t}^*, p_{s,t+1}^*, \theta_t^*), \\
  p_{s,t}^* &= \nabla_x H_t(x_{s,t}^*, p_{s,t+1}^*, \theta_t^*), \\
  x_{s,0}^* &= x_{s,0} + \eta \\
  p_{s,T}^* &= -\frac{1}{S} \nabla \Phi(x_{s,T}^*)
\end{align}

At the same the the parameter of the first layer $\theta_0^*$ satisfies

\begin{align}
  \sum_{s=1}^{S} H_t(x_{s,0} + \hat{\eta}, p_{s,t+1}^*, \theta_0^*), \forall \theta \in \Theta_t \geq \sum_{s=1}^{S} H_0(x_{s,0}^*, p_{s,1}^*, \theta_0^*) \geq \sum_{s=1}^{S} H_0(x_{s,0}^*, p_{s,1}^*, \theta), \forall \theta \in \Theta_0, \|\hat{\eta}\|_\infty \leq \epsilon
\end{align}

and parameter of the other layers $\theta_t^*, t = 1, 2, \cdots , T$ will maximize the Hamiltonian functions

\begin{align}
  \sum_{s=1}^{S} H_t(x_{s,t}^*, p_{s,t+1}^*, \theta_t^*) \geq \sum_{s=1}^{S} H_t(x_{s,t}^*, p_{s,t+1}^*, \theta), \forall \theta \in \Theta_t
\end{align}
RESULT

<table>
<thead>
<tr>
<th>Training Methods</th>
<th>Clean Data</th>
<th>PGD-20 Attack</th>
<th>Training Time (mins)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Natural train</td>
<td>95.03%</td>
<td>0.00%</td>
<td>233</td>
</tr>
<tr>
<td>PGD-3 [24]</td>
<td>90.07%</td>
<td>39.18%</td>
<td>1134</td>
</tr>
<tr>
<td>PGD-5 [24]</td>
<td>89.65%</td>
<td>43.85%</td>
<td>1574</td>
</tr>
<tr>
<td>PGD-10 [24]</td>
<td>87.30%</td>
<td>47.04%</td>
<td>2713</td>
</tr>
<tr>
<td>Free-8 [28]</td>
<td>86.29%</td>
<td>47.00%</td>
<td>667</td>
</tr>
<tr>
<td>YOPO-3-5 (Ours)</td>
<td>87.27%</td>
<td>43.04%</td>
<td>299</td>
</tr>
<tr>
<td>YOPO-5-3 (Ours)</td>
<td>86.70%</td>
<td>47.98%</td>
<td>476</td>
</tr>
</tbody>
</table>

1 Code from https://github.com/ashafahi/free_adv_train.

Table 3: Results of Wide ResNet34 for CIFAR10.

(a) "Samll CNN" in [42] Result On MNIST
SO, WHAT IS INFINITE WIDE NEURAL NETWORK

Neural Network As particle system

Linearized Neural Network

Infinite wide neural network=Kernel Method

Radom Feature + Neural Tangent Kernel


SO, WHAT IS **INFINITE WIDE** NEURAL NETWORK

Neural Network As particle system

Linearized Neural Network

Infinite wide neural network = Kernel Method

Radom Feature + **Neural Tangent Kernel**

---


TWO DIFFERENTIAL EQUATION FOR DEEP LEARNING

Evolving Kernel

Neural ODE

Data Space

Feature Space

Nonlocal PDE

Classifier

Dog

Cat
WHAT FEATURE DOES NONLOCAL PDE CAPTURES
WHAT FEATURE DOES NONLOCAL PDE CAPTURES
WHAT FEATURE DOES NONLOCAL PDE CAPTURES

Harmonic Equation -> Dimensionality

[Osher et al.2016]
WHAT FEATURE DOES NONLOCAL PDE CAPTURES

Harmonic Equation -> Dimensionality

[Osher et al.2016]

Dimension optimization is not enough
WHAT FEATURE DOES NONLOCAL PDE CAPTURES

Harmonic Equation - Dimensionality

Osher et al. 2016

Dimension optimization is not enough

Biharmonic Equation - Curvature
Figure 4: Comparisons of success rates by WNLL, CURE and WeCURE on MNIST.
IMAGE INPAINTING

Weighted Nonlocal Laplacian
PSNR=23.51dB, SSIM=0.62

Weighted Curvature Regularization
PSNR=24.07dB, SSIM=0.68
<table>
<thead>
<tr>
<th>Images</th>
<th>C.man</th>
<th>House</th>
<th>Peppers</th>
<th>Starfish</th>
<th>Monarch</th>
<th>Airplane</th>
<th>Parrot</th>
<th>Lena</th>
<th>Barbara</th>
<th>Boat</th>
<th>Man</th>
<th>Couple</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sample Rate</td>
<td>10%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LDMM</td>
<td>0.2077</td>
<td>0.3406</td>
<td>0.4400</td>
<td>0.3850</td>
<td>0.4570</td>
<td>0.3338</td>
<td>0.4500</td>
<td>0.4508</td>
<td>0.4881</td>
<td>0.3121</td>
<td>0.3400</td>
<td>0.3389</td>
<td>0.3574</td>
</tr>
<tr>
<td>WNLL</td>
<td>0.3557</td>
<td>0.4236</td>
<td>0.5681</td>
<td>0.5415</td>
<td>0.5632</td>
<td>0.4352</td>
<td>0.5680</td>
<td>0.5316</td>
<td>0.6308</td>
<td>0.4383</td>
<td>0.4787</td>
<td>0.5123</td>
<td>0.5113</td>
</tr>
<tr>
<td>CURE</td>
<td>0.3591</td>
<td>0.4337</td>
<td>0.5849</td>
<td>0.5382</td>
<td>0.3577</td>
<td>0.4324</td>
<td>0.5733</td>
<td>0.5356</td>
<td>0.6392</td>
<td>0.4409</td>
<td>0.4817</td>
<td>0.5240</td>
<td>0.5164</td>
</tr>
<tr>
<td>WeCURE</td>
<td>0.3726</td>
<td>0.4397</td>
<td>0.6042</td>
<td>0.5721</td>
<td>0.6842</td>
<td>0.4448</td>
<td>0.5953</td>
<td>0.5402</td>
<td>0.6572</td>
<td>0.4628</td>
<td>0.5051</td>
<td>0.5476</td>
<td>0.5355</td>
</tr>
<tr>
<td>Sample Rate</td>
<td>15%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LDMM</td>
<td>0.3622</td>
<td>0.4288</td>
<td>0.5308</td>
<td>0.4848</td>
<td>0.5086</td>
<td>0.4252</td>
<td>0.5464</td>
<td>0.5382</td>
<td>0.6164</td>
<td>0.4187</td>
<td>0.4483</td>
<td>0.4619</td>
<td>0.4884</td>
</tr>
<tr>
<td>WNLL</td>
<td>0.4456</td>
<td>0.5053</td>
<td>0.6380</td>
<td>0.6196</td>
<td>0.7076</td>
<td>0.5052</td>
<td>0.6247</td>
<td>0.5031</td>
<td>0.6964</td>
<td>0.5130</td>
<td>0.5544</td>
<td>0.5911</td>
<td>0.5828</td>
</tr>
<tr>
<td>CURE</td>
<td>0.4464</td>
<td>0.5294</td>
<td>0.6610</td>
<td>0.6294</td>
<td>0.7299</td>
<td>0.5115</td>
<td>0.6435</td>
<td>0.5994</td>
<td>0.7068</td>
<td>0.5228</td>
<td>0.5637</td>
<td>0.6067</td>
<td>0.5959</td>
</tr>
<tr>
<td>WeCURE</td>
<td>0.4577</td>
<td>0.5459</td>
<td>0.6766</td>
<td>0.6658</td>
<td>0.7473</td>
<td>0.5273</td>
<td>0.6621</td>
<td>0.6102</td>
<td>0.7275</td>
<td>0.5462</td>
<td>0.5939</td>
<td>0.6308</td>
<td>0.6159</td>
</tr>
<tr>
<td>Sample Rate</td>
<td>20%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LDMM</td>
<td>0.4385</td>
<td>0.5148</td>
<td>0.5980</td>
<td>0.5783</td>
<td>0.6092</td>
<td>0.5003</td>
<td>0.6074</td>
<td>0.5097</td>
<td>0.6840</td>
<td>0.5003</td>
<td>0.5295</td>
<td>0.5501</td>
<td>0.5642</td>
</tr>
<tr>
<td>WNLL</td>
<td>0.4970</td>
<td>0.5735</td>
<td>0.6856</td>
<td>0.6691</td>
<td>0.7439</td>
<td>0.5684</td>
<td>0.6673</td>
<td>0.6376</td>
<td>0.7373</td>
<td>0.5722</td>
<td>0.6062</td>
<td>0.6364</td>
<td>0.6329</td>
</tr>
<tr>
<td>CURE</td>
<td>0.5063</td>
<td>0.6044</td>
<td>0.7051</td>
<td>0.6880</td>
<td>0.7687</td>
<td>0.5847</td>
<td>0.6850</td>
<td>0.6457</td>
<td>0.7515</td>
<td>0.5882</td>
<td>0.6203</td>
<td>0.6571</td>
<td>0.6505</td>
</tr>
<tr>
<td>WeCURE</td>
<td>0.5270</td>
<td>0.6167</td>
<td>0.7241</td>
<td>0.7214</td>
<td>0.7859</td>
<td>0.6009</td>
<td>0.7017</td>
<td>0.6570</td>
<td>0.7683</td>
<td>0.6093</td>
<td>0.6492</td>
<td>0.6806</td>
<td>0.6702</td>
</tr>
</tbody>
</table>

Table 4: The SSIM results of different methods on Set12 dataset with sampling rate 10%, 15% and 20%. The best results are indicated in red and are highlighted in bold. The second best results are indicated in blue and are highlighted by underline.
GOING INTO NEURAL!
LOW FREQUENCY FIRST

- Multigrid Method [Xu 1996]
- Inverse Scale Space [Scherzer et al. 2001] [Burfer et al. 2006]
- Ridge Regression [Smale et al. 2007] [Yao et al. 2007]
- Experiment In Neural Network [Rahaman et al. 2018] [Xu et al 2019]
TEACHER STUDENT OPTIMIZATION
TEACHER STUDENT OPTIMIZATION
TEACHER STUDENT OPTIMIZATION
NO EARLY STOPPING, NO DISTILLATION

Teacher test accuracy
NO EARLY STOPPING, NO DISTILLATION
NO EARLY STOPPING, NO DISTILLATION
NO EARLY STOPPING, NO DISTILLATION
NO EARLY STOPPING, NO DISTILLATION
NO EARLY STOPPING, NO DISTILLATION
**THEORY CAN GUARANTEE OUR METHOD!**

**Theorem 1.** Suppose the sequence $\alpha_t$ monotonically decreasing to 0 and we fix the learning rate $\eta = \frac{1}{2\sqrt{t}}$ for the gradient descent. Furthermore, we have two slow-decreasing conditions for $\alpha_t$

- $C_1 \triangleq \max\limits_{0 \leq t < T} 2\sqrt{t}(\alpha_t - \alpha_{t+1}) \leq \frac{c_{\text{low}}A}{2\sqrt{T} \cdot R_k}(1 - 2\rho)$,

- $\alpha_{T_1} \geq \max\left(\frac{c_{\text{low}}A}{2\sqrt{T} \cdot R_k}(1 - 2\rho), \frac{\mathbb{I} - \frac{3}{2} \rho}{2 - 2\rho}\right)$.

We choose the following label function $h(\cdot)$

$$h(x) = \begin{cases} 4 & |x| \leq \frac{1 - 2\rho}{4} \\ \frac{1 - 2\rho}{x} & |x| > \frac{1 - 2\rho}{4} \\ \text{sgn}(x) & |x| \leq \frac{1 - 2\rho}{4} \end{cases}$$

For the self-distillation algorithm, if the following two conditions for the radius $\epsilon$ and the width $k$ satisfied

$$\epsilon = \Omega\left(\frac{(1 - 2\rho)^2}{nT_2}\right)$$

$$k = \Omega\left(\frac{K}{c_{\text{low}} \cdot \Lambda(n \cdot T_2)} \cdot \left(\frac{1}{1 - 2\rho}\right)^8\right)$$

then with probability $1 - \delta$ we have:

$$\lim_{t \to \infty} \| f(W_t, X) - \hat{g} \|_2 = 0,$$

where $W_t$ is generated by the gradient descent algorithm minimizing $\mathcal{L}(W, X)$. 
 THEORY CAN GUARANTEE OUR METHOD!

Theorem 1. Suppose the sequence $\alpha_t$ monotonically decreasing to 0 and we fix the learning rate $\eta = \frac{1}{2nT}$ for the gradient descent. Furthermore, we have two slow-decreasing conditions for $\alpha_t$

- $C_1 \triangleq \max_{0 \leq t < T} 2\sqrt{\alpha_t(1 - \alpha_t + 1)} \leq \frac{c_{\text{mean}}A}{256K^2}(1 - 2\rho)$,
- $\alpha_{T_1} \geq \max\left(\frac{c_{\text{mean}}A}{256K^2}(1 - 2\rho), \frac{3}{2 - 2\rho}\right)$.

We choose the following label function $h(\cdot)$:

$$h(x) = \begin{cases} 
\frac{4}{1 - 2\rho} x & |x| \leq \frac{1 - 2\rho}{4} \\
\text{sgn}(x) & |x| > \frac{1 - 2\rho}{4}
\end{cases}$$

For the self-distillation algorithm, if the following two conditions for the radius $\epsilon$ and the width $k$ satisfied

$$\epsilon = \Omega\left(\frac{(1 - 2\rho)^2}{nT_2}\right)$$

$$k = \Omega\left(\frac{K}{c_{\text{mean}}A}(\frac{T_2}{n})^8\left(\frac{1}{1 - 2\rho}\right)^8\right)$$

then with probability $1 - \delta$ we have:

$$\lim_{t \to \infty} \|f(W_t, X) - \hat{y}\|_2 = 0,$$

where $W_t$ is generated by the gradient descent algorithm minimizing $\mathcal{L}(W, X)$. 

Ensure margin

Width enough neural network
We use a extremely large network for experiment!
FUTURE WORK: META-LEARNING

Algorithm 1 Self-distillation

Randomly initialize the network. $t = 0$

repeat

Fetch data $(x_1, y_1), \ldots, (x_n, y_n)$ from training set.

Set the label $\tilde{y}_t(i) = \alpha_t y(i) + (1 - \alpha_t) h(NN(x_i, \omega_t))$

Detach $\tilde{y}_t$ from the computational graph

Update $\omega_{t+1} = \omega_t - \eta \sum_{i=1}^{b} \nabla_{\omega} l(NN(x_i, \omega_t), \tilde{y}_t(i))$. $t = t + 1$

until training converged = 0

Aggregating AIR and GT Label

[Tanaka et al.2018] [Sun et al. 2018] [Han et al.2018] [Han et al.2019]
FUTURE WORK: META-LEARNING

Algorithm 1 Self-distillation

Randomly initialize the network. \( t = 0 \)

repeat
  Fetch data \((x_1, y_1), \ldots, (x_n, y_n)\) from training set.
  Set the label \( \hat{y}_t(i) = \alpha_t y(i) + (1 - \alpha_t) h(\mathcal{NN}(x_i, \omega_t)) \)
  Detach \( \hat{y}_t \) from the computational graph
  Update \( \omega_{t+1} = \omega_t - \eta \sum_{i=1}^{b} \nabla_{\omega} l(\mathcal{NN}(x_i, \omega_t), \hat{y}_t(i)) \).
  \( t = t + 1 \)
until training converged = 0

[Jetaka et al.2018] [Sun et al. 2018] [Han et al.2018] [Han et al.2019]

Supervise Signal: Validation set Robustness
NOISY LABEL LEADS TO ADVERSARIAL EXAMPLE?
NOISY LABEL LEADS TO ADVERSARIAL EXAMPLE?
NOISY LABEL LEADS TO ADVERSARIAL EXAMPLE?

** Initialization
** Good Lip Constant
NOISY LABEL LEADS TO ADVERSARIAL EXAMPLE?
NOISY LABEL LEADS TO ADVERSARIAL EXAMPLE?

- Clean label training
- Noisy label training
- Good Lip Constant
- Bad Lip Constant
RESEARCH INTEREST

- Geometry Of Data
- Control View Of Deep Learning
- Kernel Learning
- PDEs On Graphs
- Learning with limited data (noisy, semi-supervise, weak supervise)
- Computational Photography, Image Processing, Graphics.
THANK YOU AND QUESTIONS?


Yiping Lu, Di He, Zhuohan Li, Zhiqing Sun, Bin Dong, Tao Qin, Liwei Wang, Tie-yan Liu. OreNet: Understanding NLP from a neural ODE viewpoint. (Submitted)


Bin Dong, Jikai Hou, Yiping Lu, Zhihua Zhang. Distillation $\approx$ Early Stopping? Extracting Knowledge Utilizing Anisotropic Information Retrieval.(Submitted)

Acknowledgement

Bin Dong  Di He  Aoxiao Zhong  Xiaoshuai Zhang  Zhuohan Li  Jikai Hou  Tianyuan Zhang  Dinghuai Zhang  Zhiqing Sun  Zhanxing Zhu  Liwei Wang  Quanzheng Li  Zhihua Zhang

Always looking forward to cooperation opportunities
Contact: yplu@stanford.edu