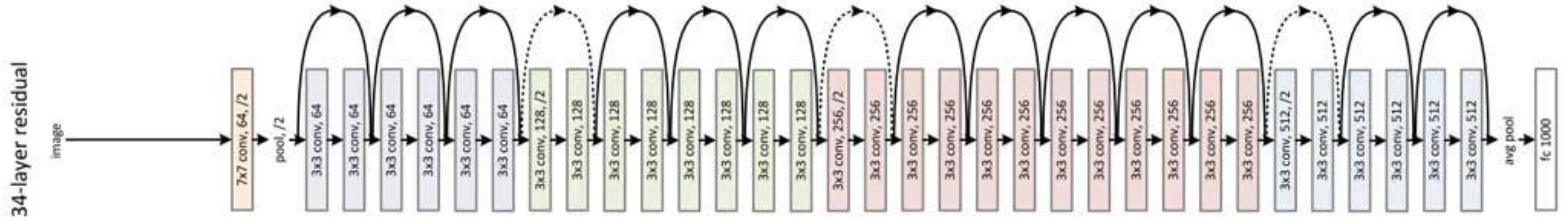
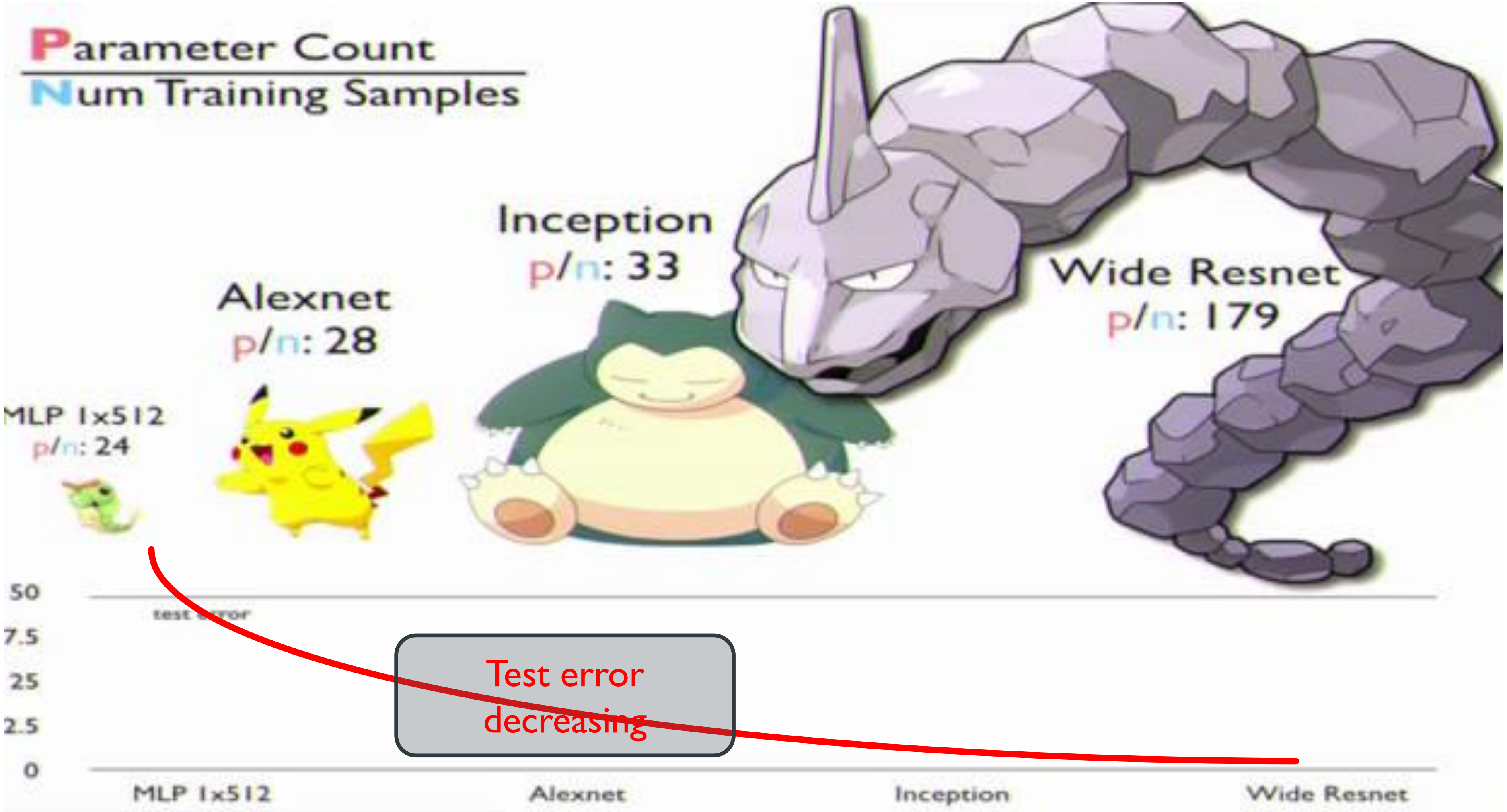
DYNAMIC SYSTEM VIEW OF DEEP LEARNING

YIPING LU PEKING UNIVERSITY

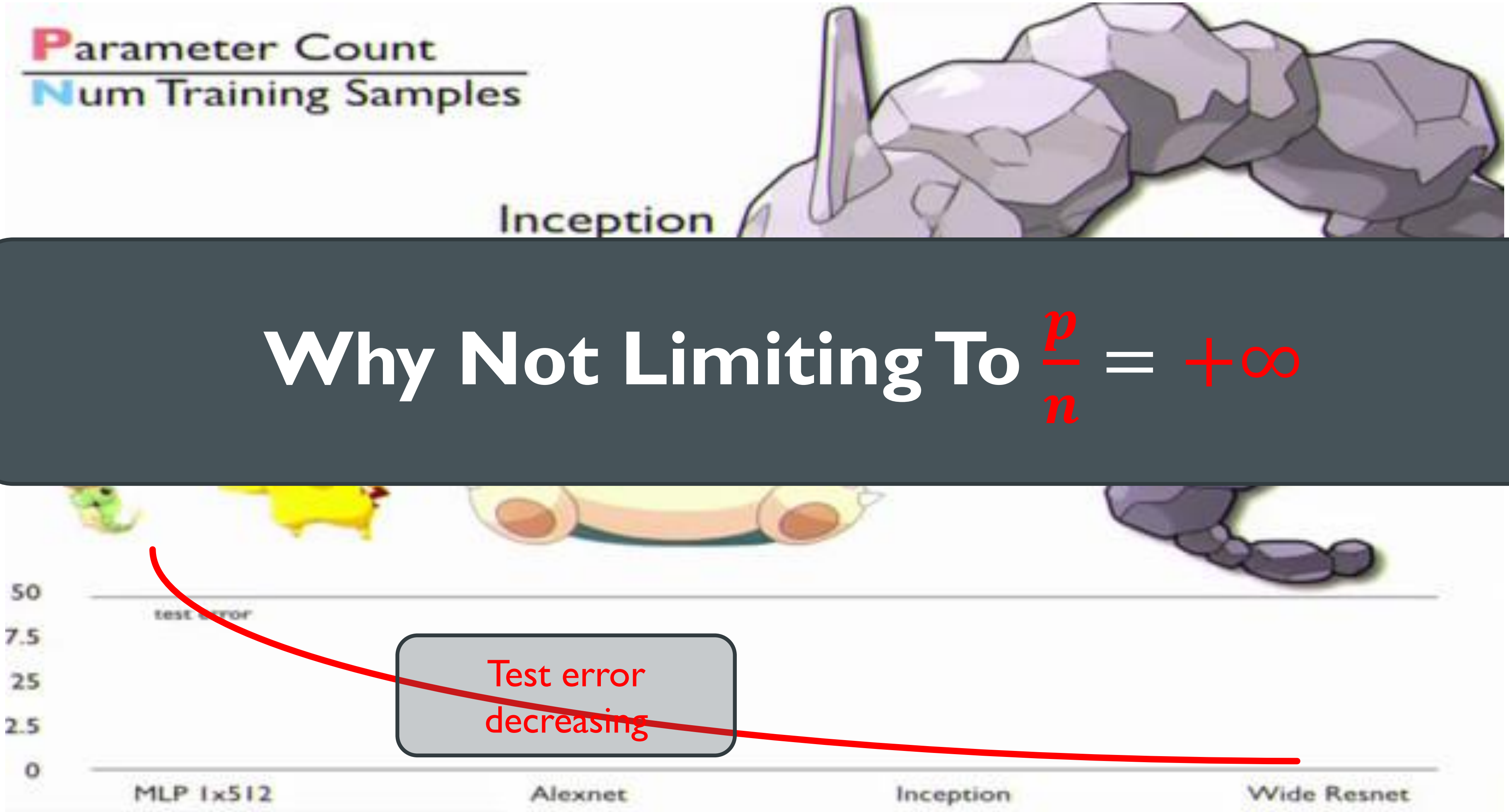


DEEP LEARNING IS SUCCESSFUL, BUT…

LARGER THE BETTER?

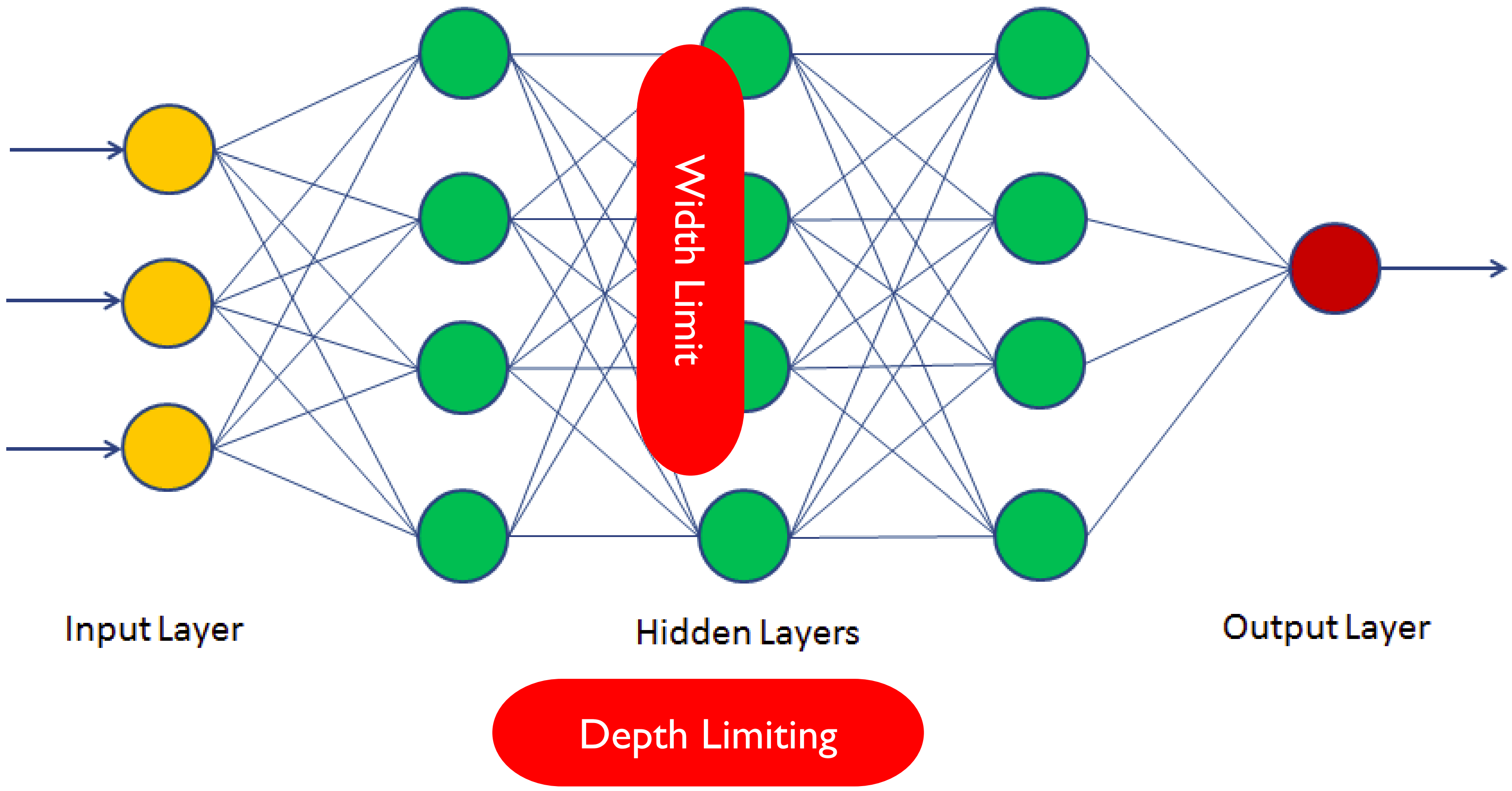


LARGER THE BETTER?

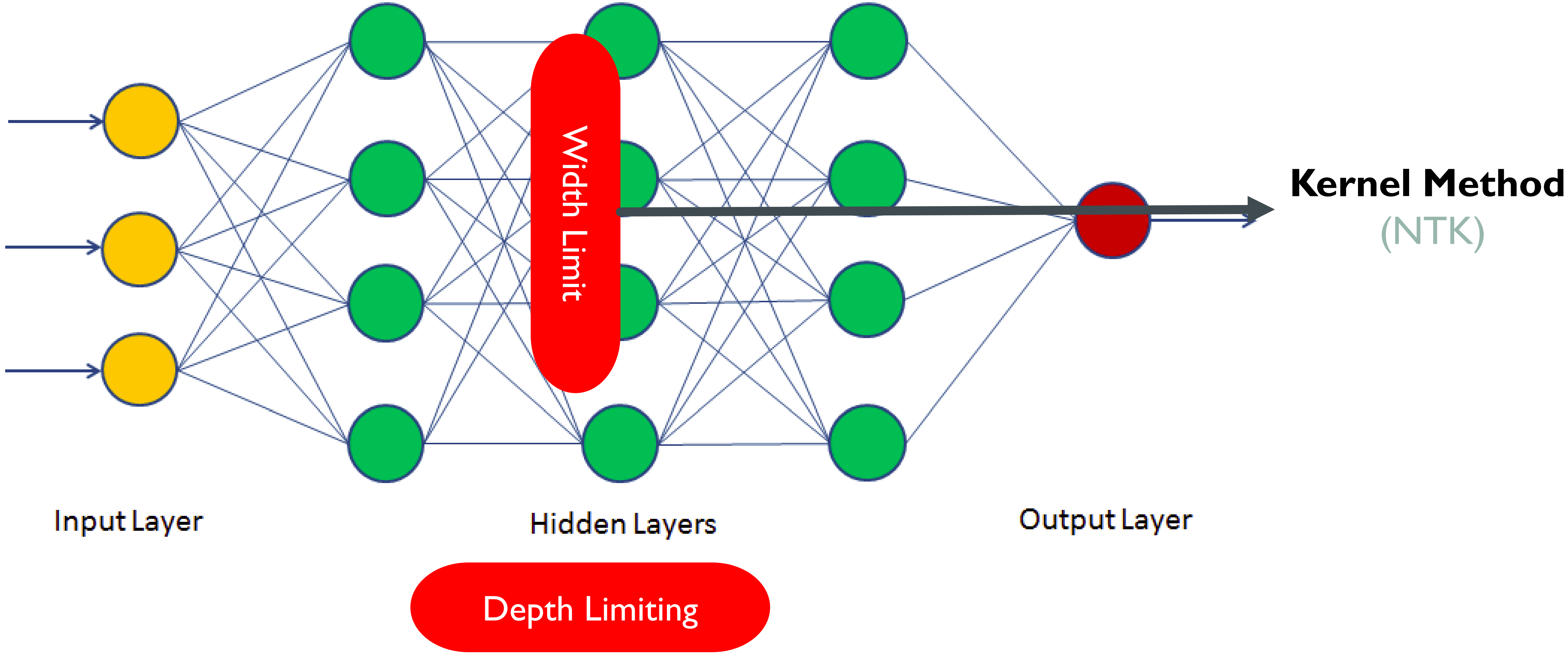


Why Not Limiting To $\frac{p}{n} = +\infty$

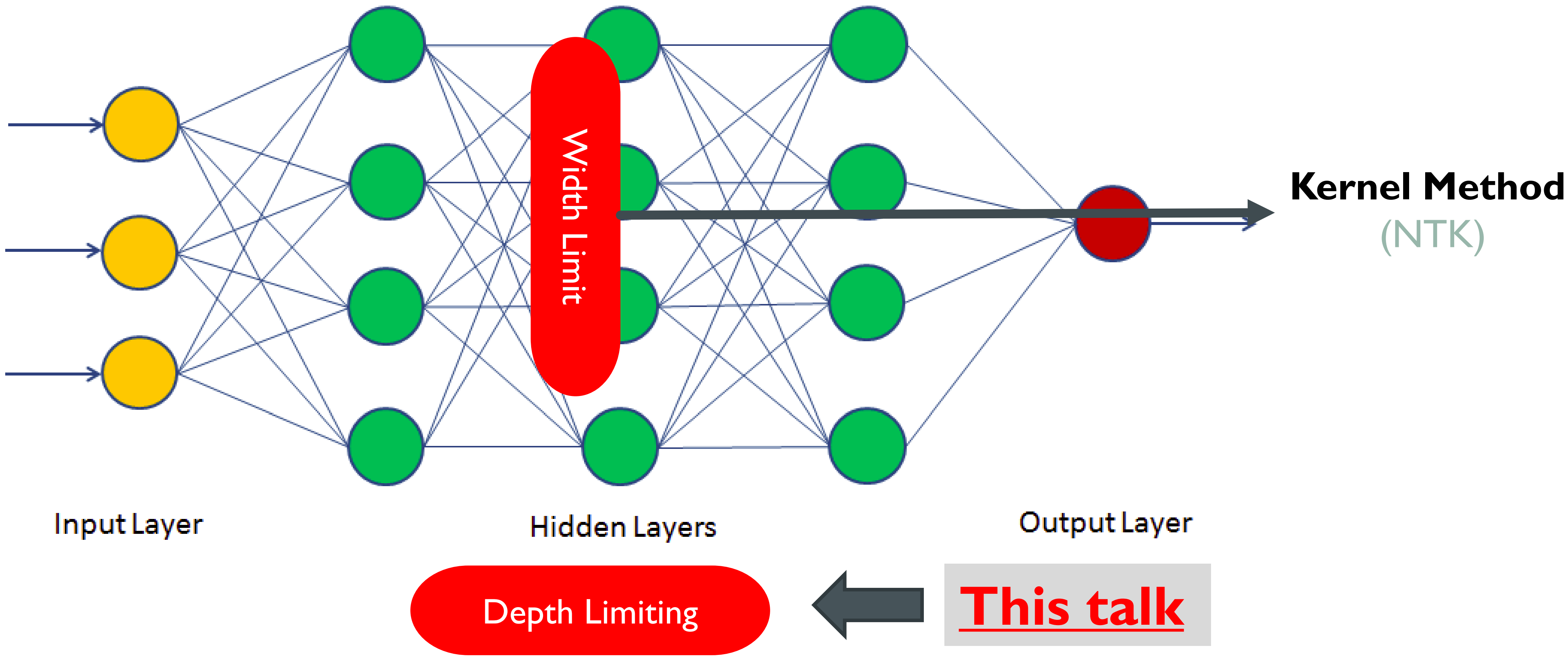
TWO LIMITING



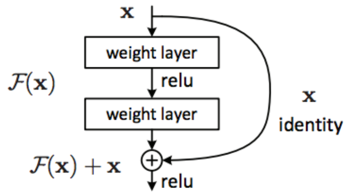
TWO LIMITING



TWO LIMITING



DEPTH LIMITING: ODE



First-order ODE $\Rightarrow$ Numerical solver: Euler's method $\Rightarrow$ ResNet

[He et al. 2015]

$$\frac{dx}{dt} = F(x,t),$$

$$x(0) = x_0,$$

$$x_{l+1} = x_l + \gamma F(x_l, t_l),$$

$$x_0 \doteq x(0), x_l = x(\gamma l), \ldots \gamma - step\ size$$

$$x_{l+1} = x_l + F(x_l, t_l),$$

[E. 2017] [Haber et al. 2017]
[Lu et al. 2017] [Chen et al. 2018]

ALSO THEORETICAL GUARANTEED

Deep Limits of Residual Neural Networks

Matthew Thorpe[1] and Yves van Gennip[2]

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

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

March 2019

Abstract

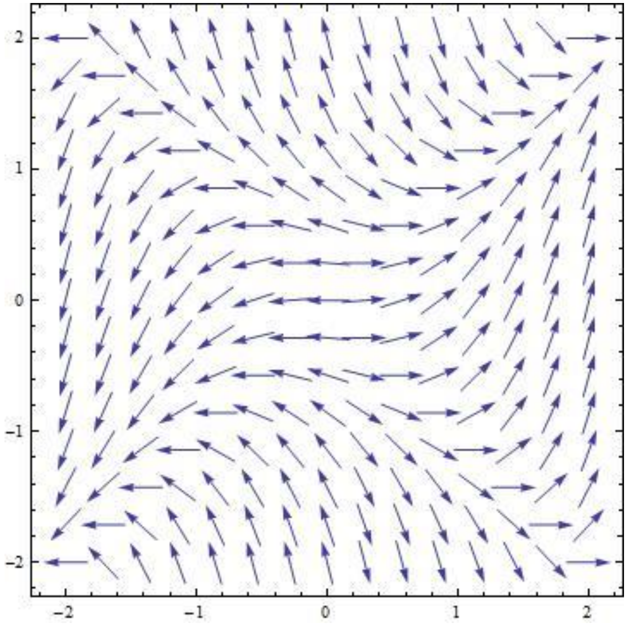
Neural networks have been very successful in many applications; we often, however, lack a theoretical understanding of what the neural networks are actually learning. This problem emerges when trying to generalise to new data sets. The contribution of this paper is to show that, for the residual neural network model, the deep layer limit coincides with a parameter estimation problem for a nonlinear ordinary differential equation. In particular, whilst 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 Γ-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.

Compactness yields convergence

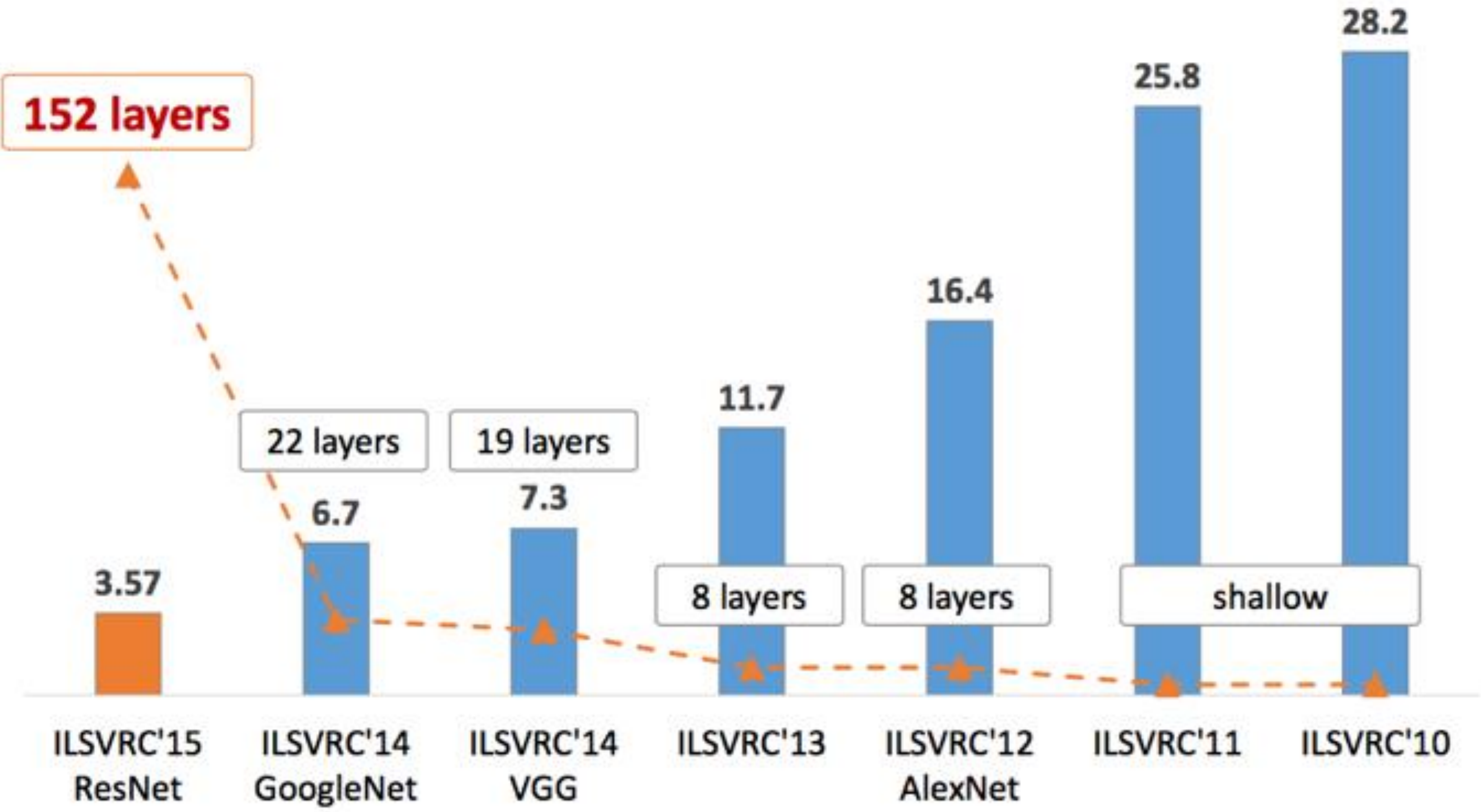
Gamma Convergence

BEYOND FINITE LAYER NEURAL NETWORK

Going into infinite layer



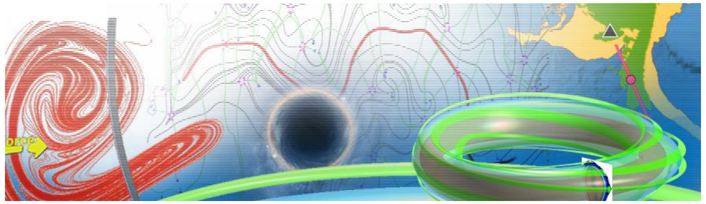
Differential Equation



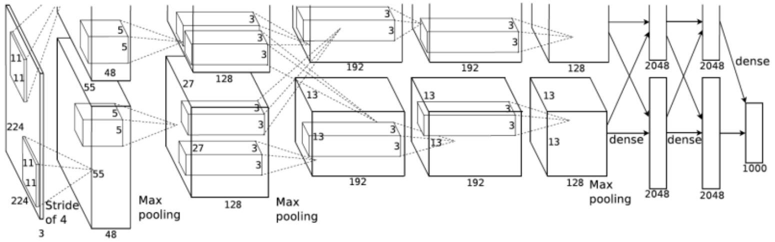
TRADITIONAL WISDOM IN DEEP LEARNING

+

=



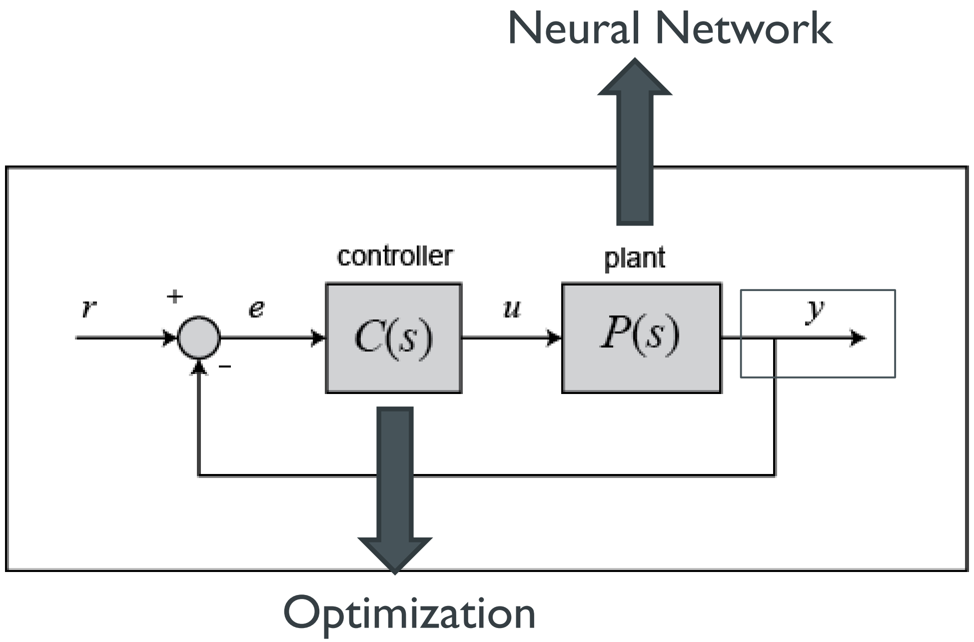
+



=

?

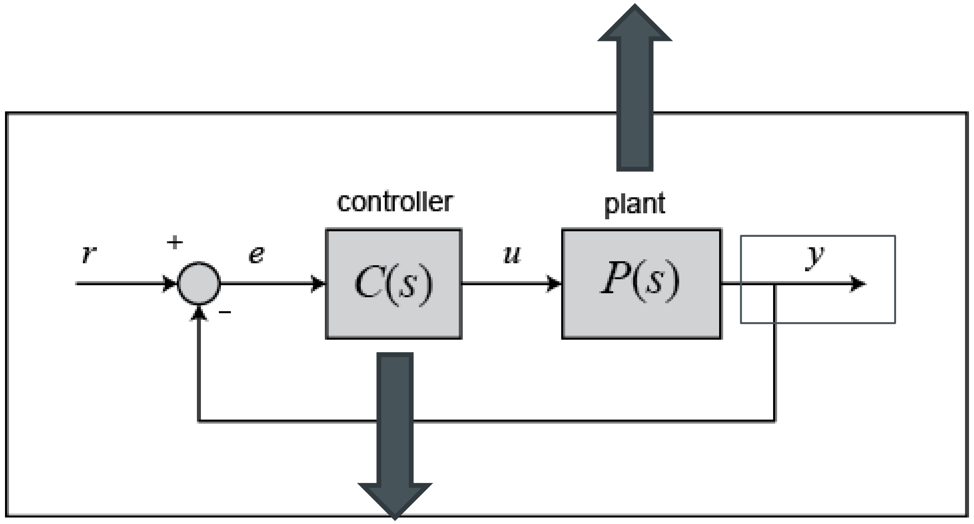
BRIDGING **CONTROL** AND **LEARNING**



Feedback is the learning loss

BRIDGING **CONTROL** AND **LEARNING**

Neural Network



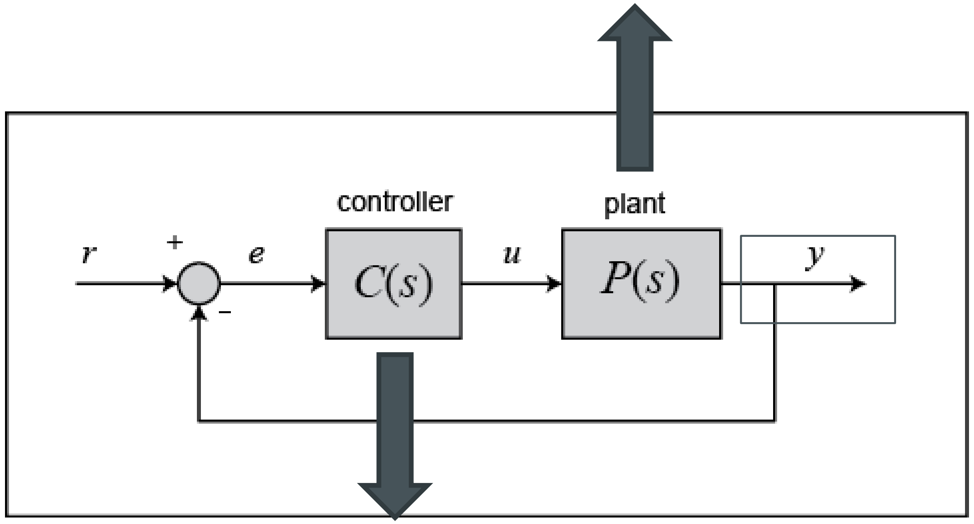
Optimization

Interpretability? PDE-Net ICML2018

Feedback is the learning loss

BRIDGING CONTROL AND LEARNING

Neural Network



Optimization

Interpretability? PDE-Net ICML2018

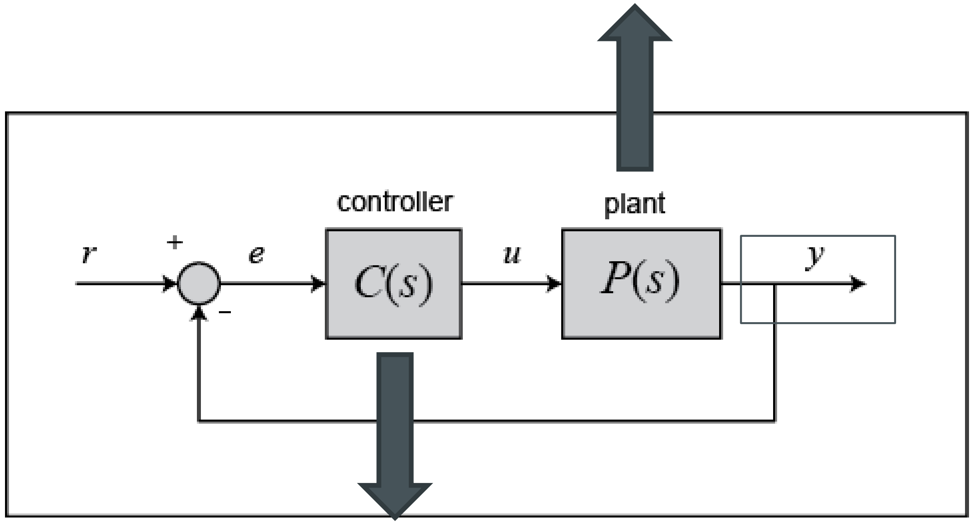
A better model? LM-ResNet ICML2018
DURR ICLR2019
Macaroon submitted

Feedback is the learning loss

Chang B, Meng L, Haber E, et al. Reversible architectures for arbitrarily deep residual neura networks[C]//Thirty-Second AAAI Conference on Artificial Intelligence. 2018.
Tao Y, Sun Q, Du Q, et al. Nonlocal Neural Networks, Nonlocal Diffusion and Nonlocal Modeling[C]//Advances in Neural Information Processing Systems. 2018: 496-506.

BRIDGING **CONTROL** AND **LEARNING**

Neural Network



Optimization

New Algorithm?

Interpretability? PDE-Net ICML2018

A better model? LM-ResNet ICML2018
DURR ICLR2019
Macaroon submitted

Feedback is the learning loss

Chang B, Meng L, Haber E, et al. Reversible architectures for arbitrarily deep residual neura networks[C]//Thirty-Second AAAI Conference on Artificial Intelligence. 2018.
Tao Y, Sun Q, Du Q, et al. Nonlocal Neural Networks, Nonlocal Diffusion and Nonlocal Modeling[C]//Advances in Neural Information Processing Systems. 2018: 496-506.

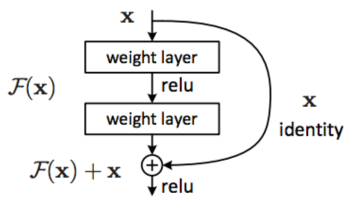
YOPO
Submitted

An W, Wang H, Sun Q, et al. A pid controller approach for stochastic optimization of deep networks[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018: 8522-8531.
Chen T Q, Rubanova Y, Bettencourt J, et al. Neural ordinary differential equations[C]//Advances in Neural Information Processing Systems. 2018: 6571-6583.
Li Q, Hao S. An optimal control approach to deep learning and applications to discrete-weight neural networks[J]. arXiv preprint arXiv:1803.01299, 2018.
Chang B, Meng L, Haber E, et al. Multi-level residual networks from dynamical systems view[J]. arXiv preprint arXiv:1710.10348, 2017.

HOW DIFFERENTIAL EQUATION VIEW HELPS DEEP LEARNING SYSTEM DESIGNING

PRINCIPLED NEURAL ARCHITECTURE DESIGN

DEPTH LIMITING: ODE



First-order ODE	⟹	Numerical solver: Euler's method	⟹	ResNet [He et al. 2015]
$\frac{dx}{dt} = F(x,t),$		$x_{l+1} = x_l + \gamma F(x_l, t_l),$		$x_{l+1} = x_l + F(x_l, t_l),$
$x(0) = x_0,$		$x_0 \doteq x(0), x_l = x(\gamma l), \ldots \gamma - step\ size$		[E. 2017] [Haber et al. 2017] [Lu et al. 2017] [Chen et al. 2018]

Numerical solver ⟹ Deep architecture

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.

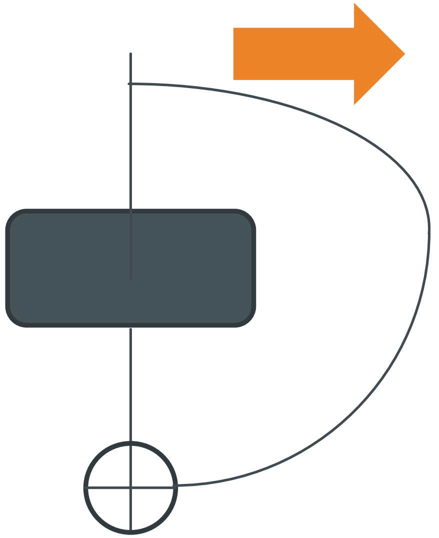
Network	Related ODE	Numerical Scheme
ResNet, ResNeXt, etc.	$u_t = f(u)$	Forward Euler scheme
PolyNet	$u_t = f(u)$	Approximation of backward Euler scheme
FractalNet	$u_t = f(u)$	Runge-Kutta scheme
RevNet	$\dot{X} = f_1(Y), \dot{Y} = f_2(X)$	Forward Euler scheme

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?

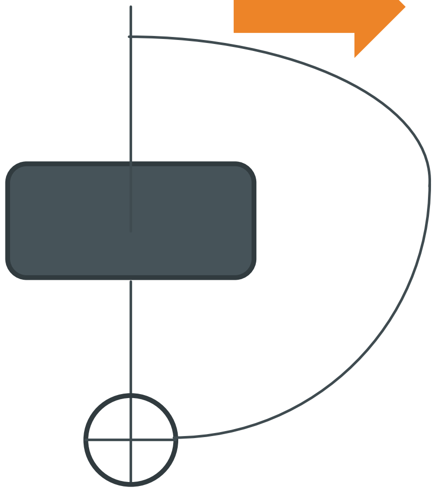
MULTISTEP ARCHITECTURE?



$$x_t = f(x)$$

$$x_{n+1} = x_n + f(x_n)$$

MULTISTEP ARCHITECTURE?

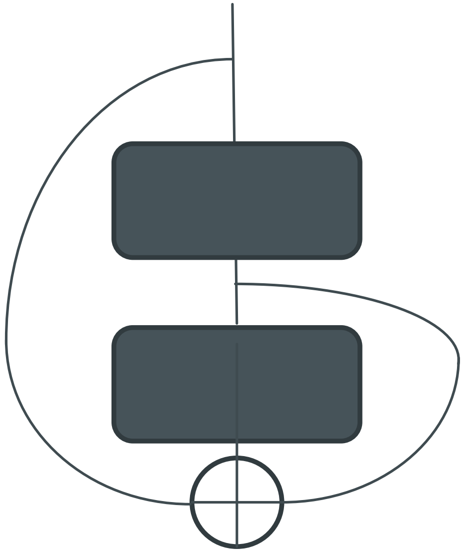


$$x_{n+1} = x_n + f(x_n)$$

$$x_t = f(x)$$

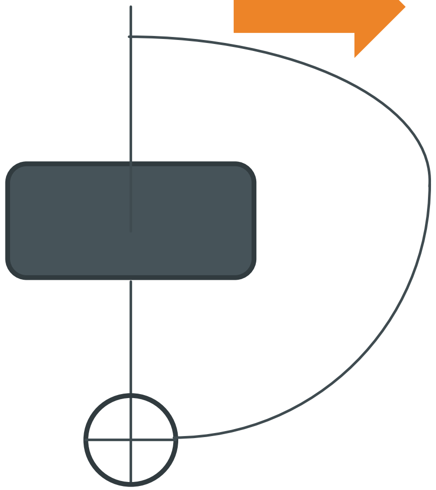
Linear Multi-step Scheme

$$x_{n+1} = (1 - k_n)x_n + k_n x_{n-1} + f(x_n)$$



Linear Multi-step Residual Network

MULTISTEP ARCHITECTURE?

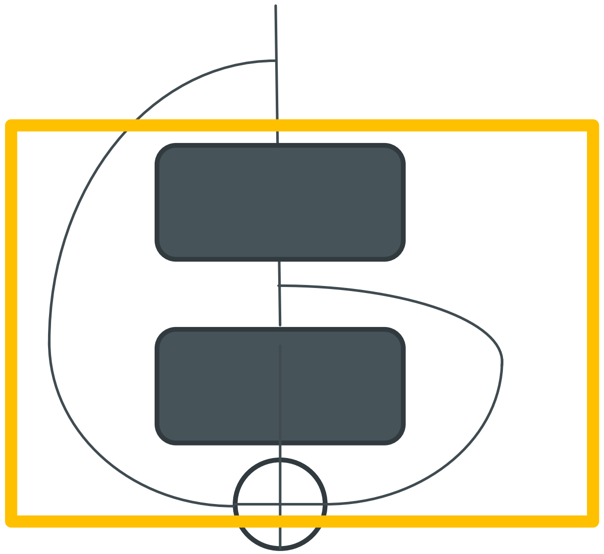


$$x_{n+1} = x_n + f(x_n)$$

$$x_t = f(x)$$

Linear Multi-step Scheme

$$x_{n+1} = (1 - k_n)x_n + k_n x_{n-1} + f(x_n)$$



Only One More Parameter

Linear Multi-step Residual Network

EXPERIMENT

Table 2: Linear Multi-step Resnet Test On Cifar

Model	Layer	Accuracy	Params	Dataset
Resnet	20	91.25	0.27M	Cifar10
Resnet	32	92.49	0.46M	Cifar10
Resnet	44	92.83	0.66M	Cifar10
Resnet	56	93.03	0.85M	Cifar10
Resnet	110	93.63	1.7M	Cifar10
LM-Resnet(Ours)	20	91.67	0.27M	Cifar10
LM- Resnet(Ours)	32	92.82	0.46M	Cifar10
LM- Resnet(Ours)	44	92.98	0.66M	Cifar10
LM- Resnet(Ours)	56	**93.69**	0.85M	Cifar10
EM- Resnet(Ours)	40	91.75	0.27M	Cifar10
Resnet	110	72.24	1.7M	Cifar100
Resnet	164	75.67	2.55M	Cifar100
Resnet	1202	77.29	18.88M	Cifar100
ResneXt	29(8×64d)	82.23	34.4M	Cifar100
ResneXt	29(16×64d)	82.69	68.1M	Cifar100
LM-Resnet(Ours)	110	73.16	1.7M	Cifar100
LM-Resnet(Ours)	164	76.74	2.55M	Cifar100
LM-ResneXt(Ours)	29(8×64d)	82.51	34.4M	Cifar100
LM-ResneXt(Ours)	29(16×64d)	**83.21**	68.1M	Cifar100

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

Model	Layer	top-1	top-5
ResNet (He et al. (2015b))	50	24.7	7.8
ResNet (He et al. (2015b))	101	23.6	7.1
ResNet (He et al. (2015b))	152	23.0	6.7
LM-ResNet (Ours)	50, pre-act	23.8	7.0
LM-ResNet (Ours)	101, pre-act	**22.6**	**6.4**

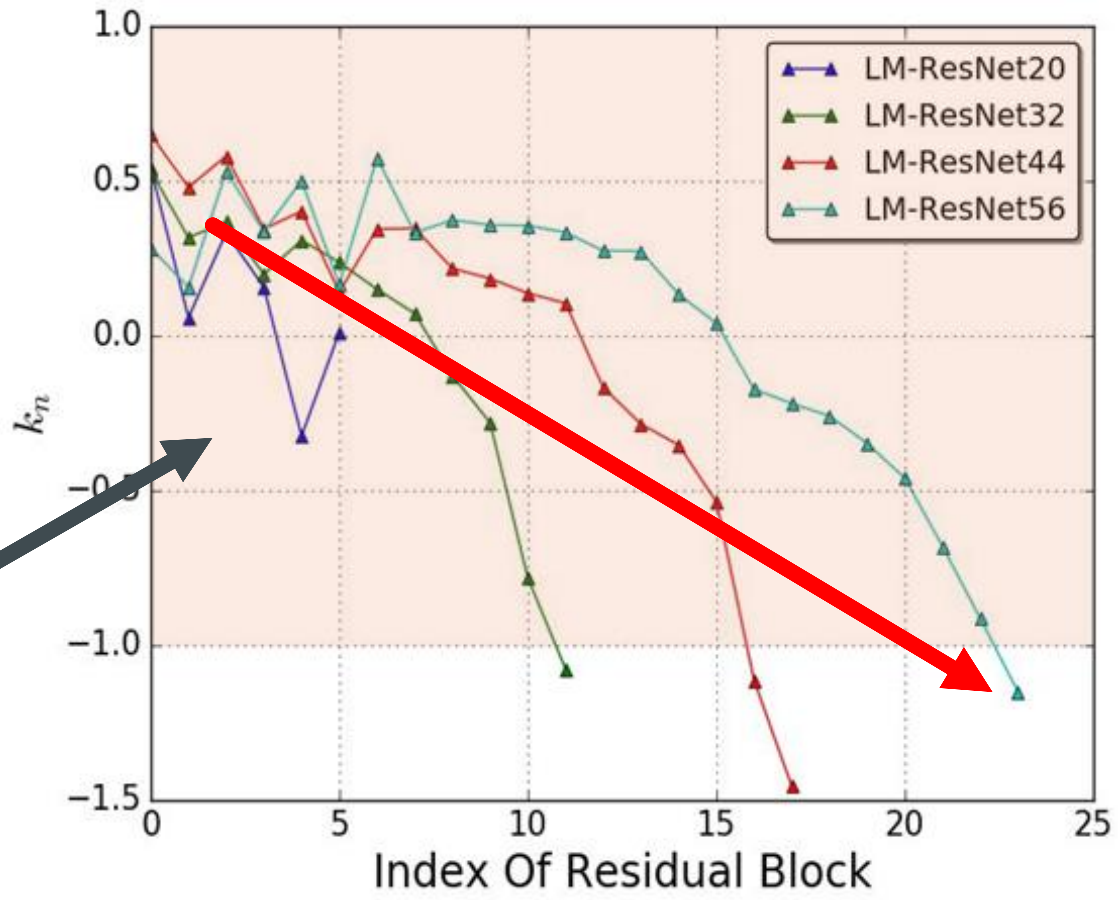
MODIFIED EQUATION VIEW

$$x_{n+1} = (1 - k_n)x_n + k_n x_{n-1} + \Delta t f(x_n)$$

Learn A Momentum

$$(1 + k_n)\,\dot{u} + \boxed{(1 - k_n)\frac{\Delta t}{2}\ddot{u}_n} + o(\Delta t^3) = f(u)$$

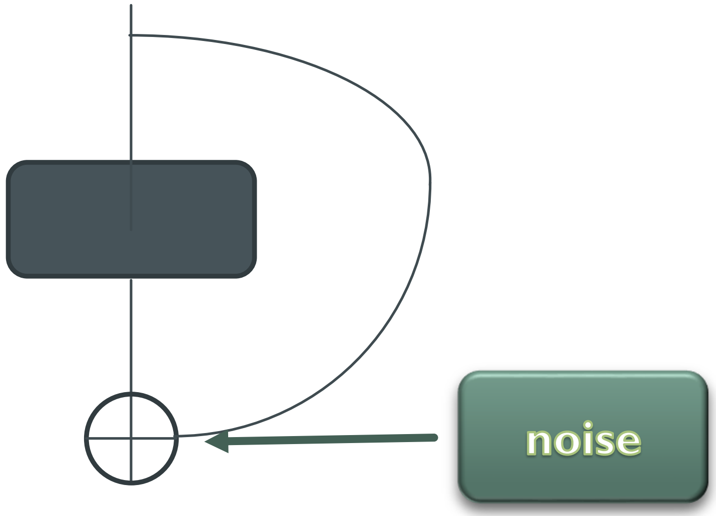
[Su et al. 2016] [Dong et al. 2017]



Analysis by *zero stability*

UNDERSTANDING DROPOUT

Noise can avoid overfit?



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

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

$\boldsymbol{E_{data}}(loss(X(T))$

STOCHASTIC DEPTH AS AN EXAMPLE

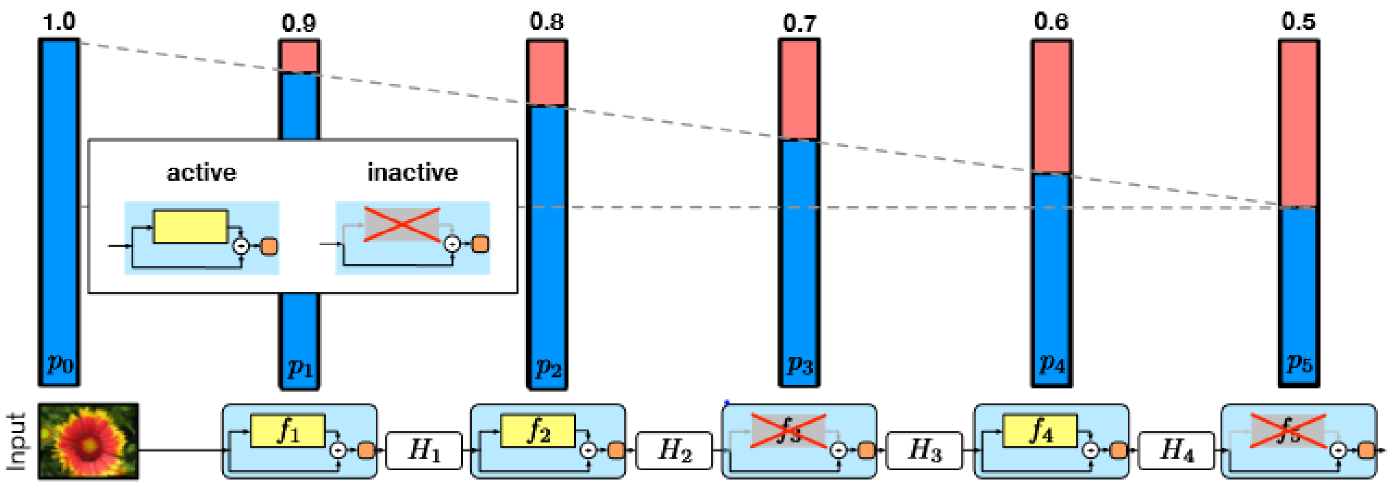


Fig. 2. The linear decay of p_ℓ illustrated on a ResNet with stochastic depth for $p_0 = 1$ and $p_L = 0.5$. Conceptually, we treat the input to the first ResBlock as H_0, which is always active.

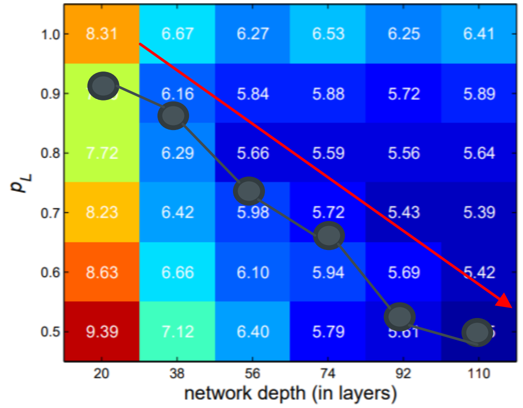
$$\boldsymbol{x_{n+1}} = \boldsymbol{x_n} + \boldsymbol{\eta_n f(x)}$$

$$= \boldsymbol{x_n} + \underbrace{\boldsymbol{E\eta_n f(x_n)}}_{\text{mean}} + \underbrace{\boxed{\boldsymbol{(\eta_n - E\eta_n) f(x_n)}}}_{\text{noise}}$$

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

$$\boxed{\text{We need } 1 - 2p_n = O(\sqrt{\Delta t})}$$

Hyper-parameter setting meets convergence condition



SOME RECENT WORK

arXiv.org > cs > arXiv:1812.00174

Computer Science > Machine Learning

Stochastic Training of Residual Networks: a Differential Equation Viewpoint

Qi Sun, Yunzhe Tao, Qiang Du

(Submitted on 1 Dec 2018)

arXiv.org > cs > arXiv:1906.02355

Computer Science > Machine Learning

Neural SDE: Stabilizing Neural ODE Networks with Stochastic Noise

Xuanqing Liu, Tesi Xiao, Si Si, Qin Cao, Sanjiv Kumar, Cho-Jui Hsieh

(Submitted on 5 Jun 2019)

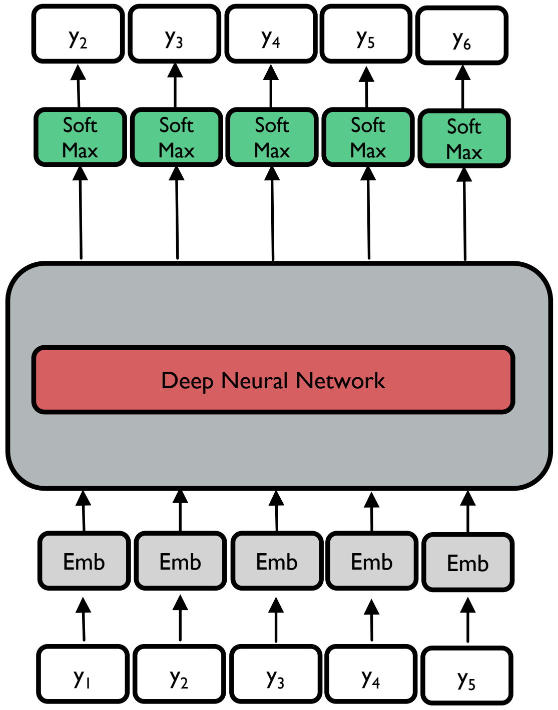
arXiv.org > cs > arXiv:1811.10745

Computer Science > Machine Learning

ResNets Ensemble via the Feynman-Kac Formalism to Improve Natural and Robust Accuracies

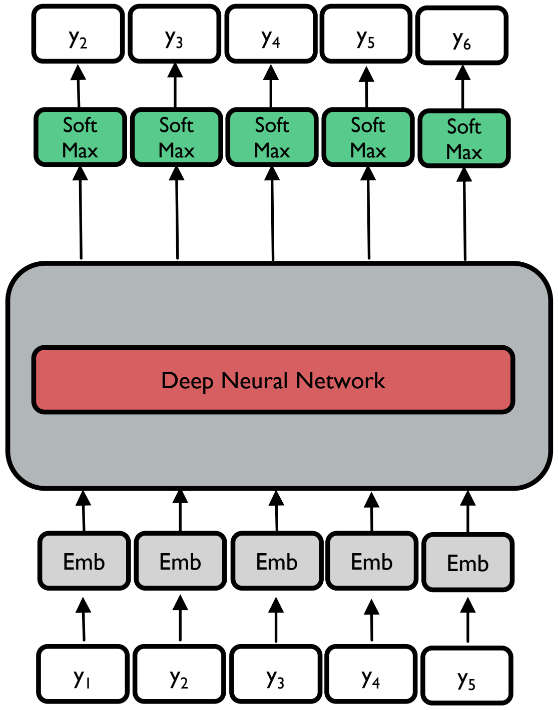
Bao Wang, Binjie Yuan, Zuoqiang Shi, Stanley J. Osher

UNDERSTANDING SEQUENCE TO SEQUENCE MODELING

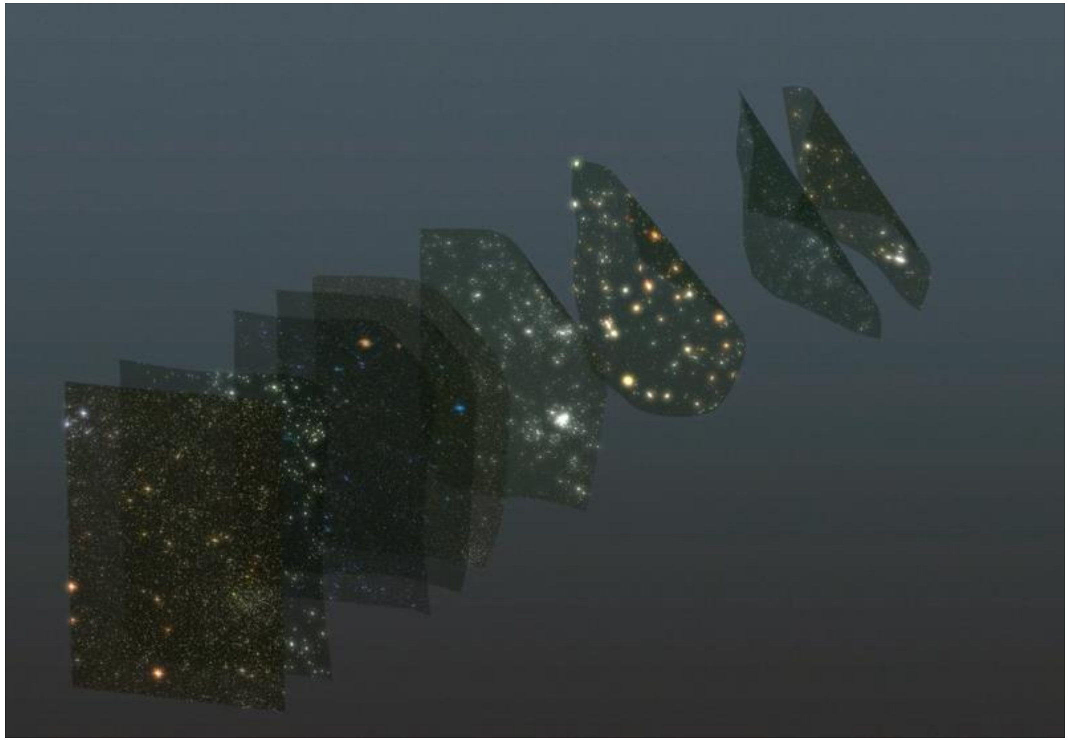


The input shape is different (sentences of different length)

UNDERSTANDING SEQUENCE TO SEQUENCE MODELING



The input shape is different
(sentences of different length)



Idea: Consider every **word** in a document as a **particle** in the n-body system.

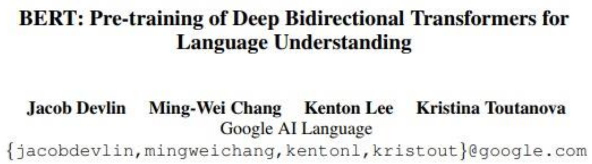
TRANSFORMER AS A SPLITTING SCHEME

Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com
Noam Shazeer* Google Brain noam@google.com
Niki Parmar* Google Research nikip@google.com
Jakob Uszkoreit* Google Research usz@google.com
Llion Jones* Google Research llion@google.com
Aidan N. Gomez*† University of Toronto aidan@cs.toronto.edu
Łukasz Kaiser* Google Brain lukaszkaiser@google.com
Illia Polosukhin*‡ illia.polosukhin@gmail.com

Abstract

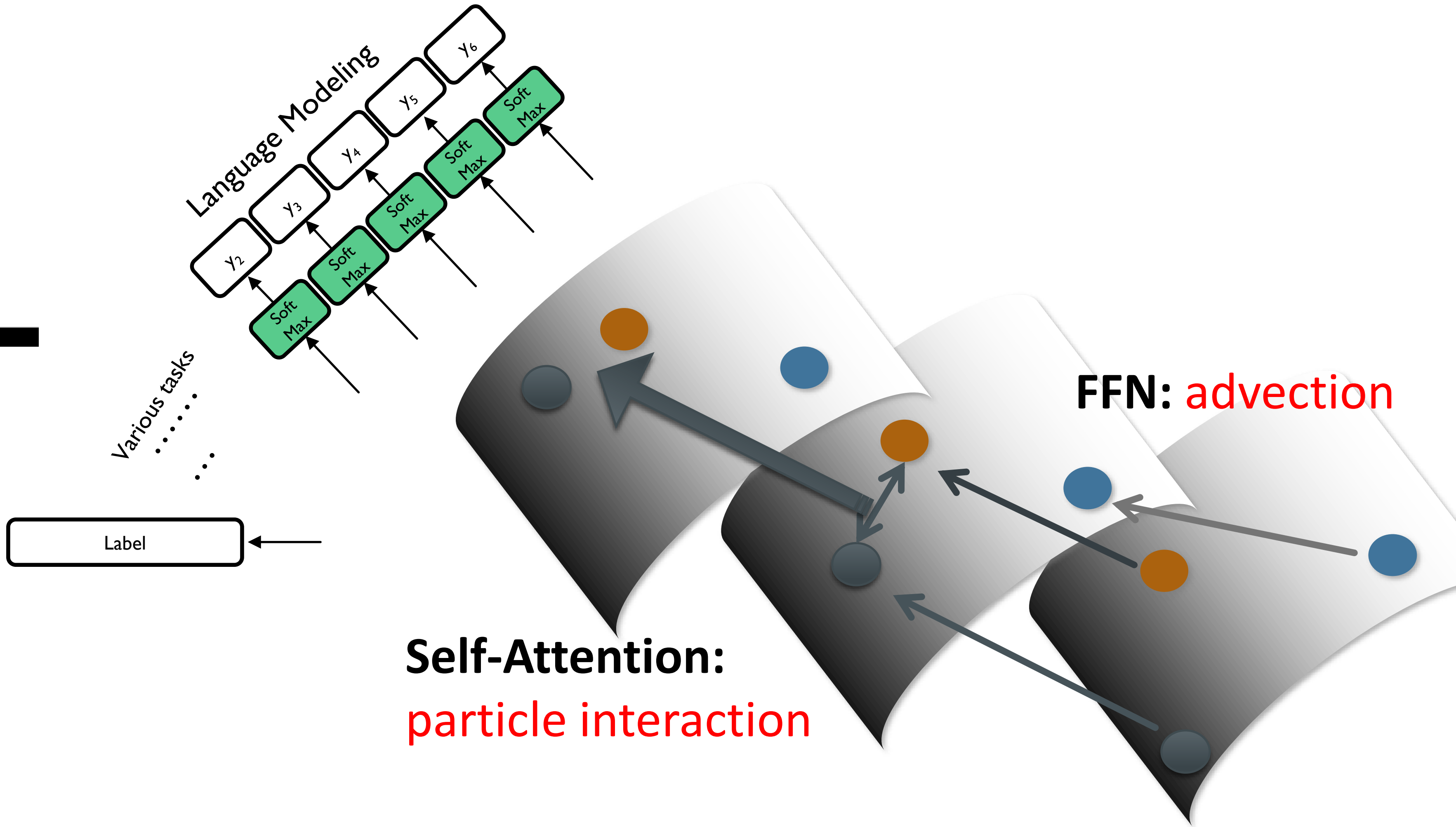
[Vaswani et al.2017]



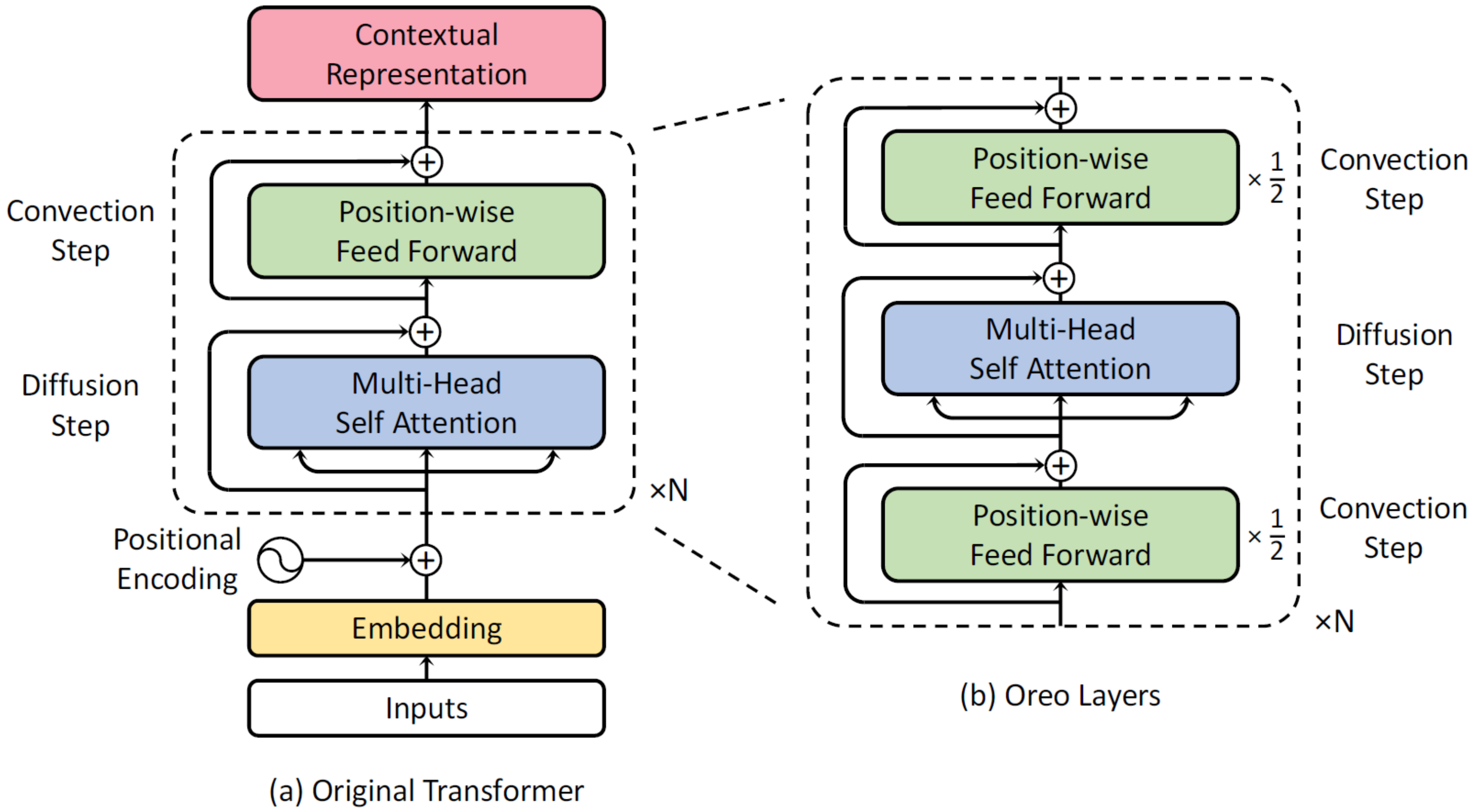
BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Jacob Devlin Ming-Wei Chang Kenton Lee Kristina Toutanova
Google AI Language
{jacobdevlin,mingweichang,kentonl,kristout}@google.com

[Devlin et al.2017]



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: $u_S(t+\Delta t) = e^{\frac{1}{2}\Delta t A}e^{\Delta t B}e^{\frac{1}{2}\Delta t A}u(t)$

RESULT

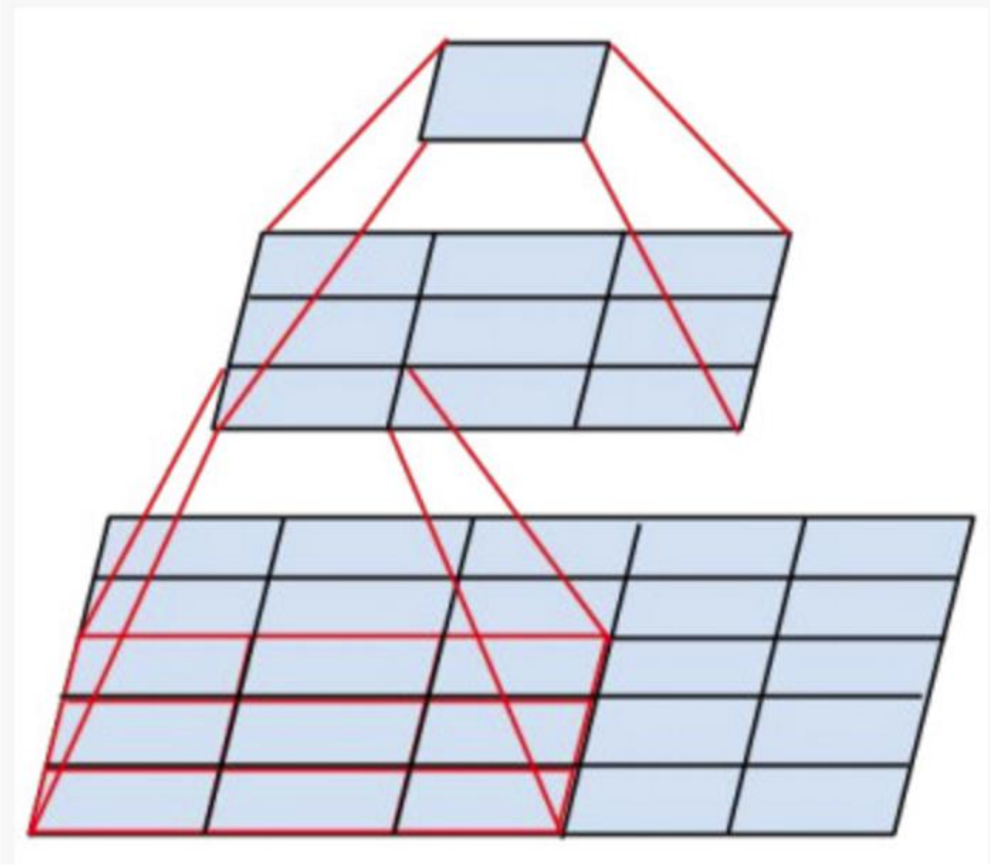
Table 1: Performance on the testsets of WMT14 En-De and IWSLT14 De-En tasks.

Method	IWSLT14 De-En small	WMT14 En-De base	WMT14 En-De big
Transformer [3]	34.4	27.3	28.4
Weighted Transformer [30]	/	28.4	28.9
Relative Transformer [31]	/	26.8	29.2
Universal Transformer [4]	/	28.9	/
Scaling NMT [32]	/	/	29.3
Dynamic Conv [33]	35.2	/	29.7
Ours	**35.43**	**28.91**	**30.22**

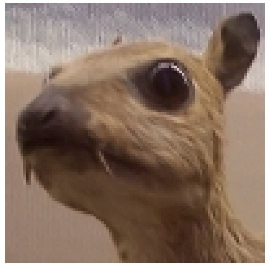
Table 2: The test results on the GLUE benchmark (except WNLI).

Method	CoLA	SST-2	MRPC	STS-B	QQP	MNLI-m/mm	QNLI	RTE	**GLUE**
Existinng systems									
ELMo [8]	33.6	90.4	84.4/78.0	74.2/72.3	63.1/84.3	74.1/74.5	79.8	58.9	70.0
OpenAI GPT [35]	47.2	93.1	87.7/83.7	85.3/84.8	70.1/88.1	80.7/80.6	87.2	69.1	76.9
$\text{BERT}_{\text{BASE}}$ [7]	52.1	93.5	**88.9/84.8**	87.1/85.8	**71.2/89.2**	84.6/83.4	90.5	66.4	78.3
Our systems									
$\text{BERT}_{\text{BASE}}$ (ours)	52.8	92.8	87.3/83.0	81.2/80.0	70.2/88.4	84.4/83.7	90.4	64.9	77.4
$\text{Ours}_{\text{BASE}}$	**57.6**	**94.0**	88.4/84.4	**87.5/86.3**	70.8/89.0	**85.4/84.5**	**91.6**	**70.5**	**79.7**

CONVOLUTION?



Input image



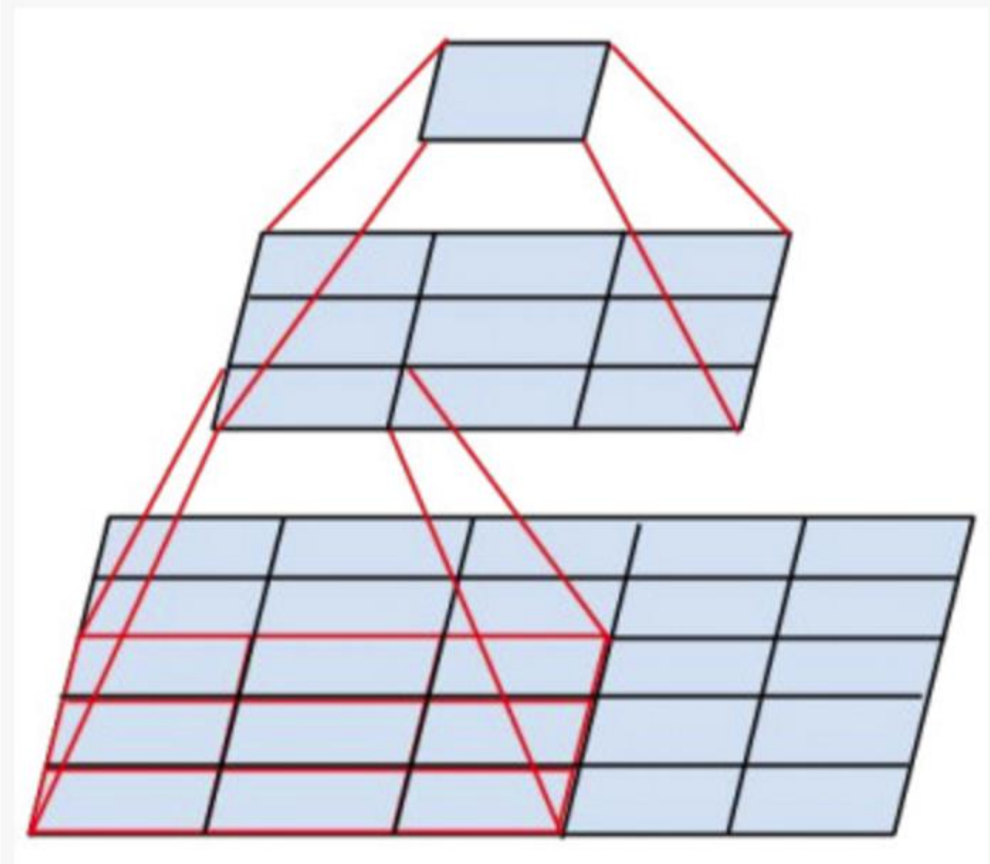
Convolution Kernel

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

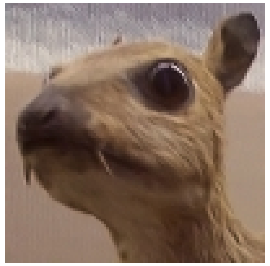
Feature map

How to interpret **Convolution**?

CONVOLUTION?



Input image



Convolution Kernel

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Feature map

How to interpret **Convolution**?

NN → ODE

CNN → PDE

Gradient

CONV FILTERS AS DIFFERENTIAL OPERATORS

Propositin 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*

$$\frac{1}{\varepsilon^{|\alpha|}} \sum_{k\in\mathbb{Z}^2} q[k]F(x+\varepsilon k) = C_\alpha \frac{\partial^\alpha}{\partial x^\alpha} F(x) + O(\varepsilon), \text{ as } \varepsilon \to 0, \tag{3}$$

where C_α 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 $K\backslash\{|\alpha|+1\}$ for some $K > |\alpha|$, then

$$\frac{1}{\varepsilon^{|\alpha|}} \sum_{k\in\mathbb{Z}^2} q[k]F(x+\varepsilon k) = C_\alpha \frac{\partial^\alpha}{\partial x^\alpha} F(x) + O(\varepsilon^{K-|\alpha|}), \text{ as } \varepsilon \to 0. \tag{4}$$

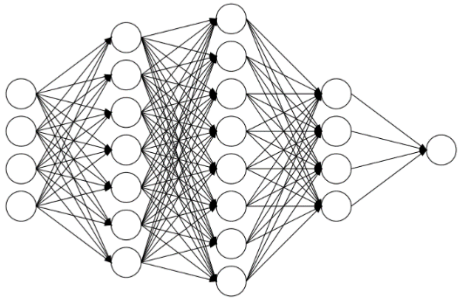
	1	
1	-4	1
	1	

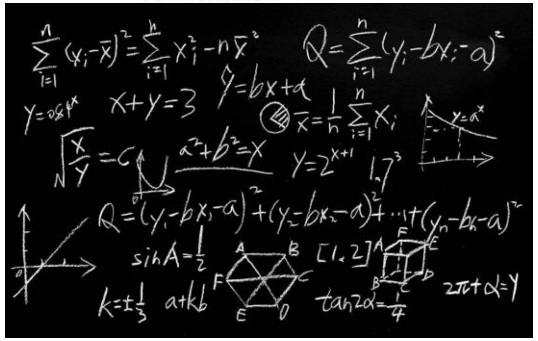
$$\Delta u = u_{xx} + u_{yy}$$

PHYSICS DISCOVERY

Our Work

现象　　规律　　法则





NEW MATH DISCOVERY

The Ramanujan Machine: Automatically Generated Conjectures on Fundamental Constants

Gal Raayoni[1], George Pisha[1], Yahel Manor[1], Uri Mendlovic[2], Doron Haviv[1], Yaron Hadad[1], and Ido Kaminer[1]

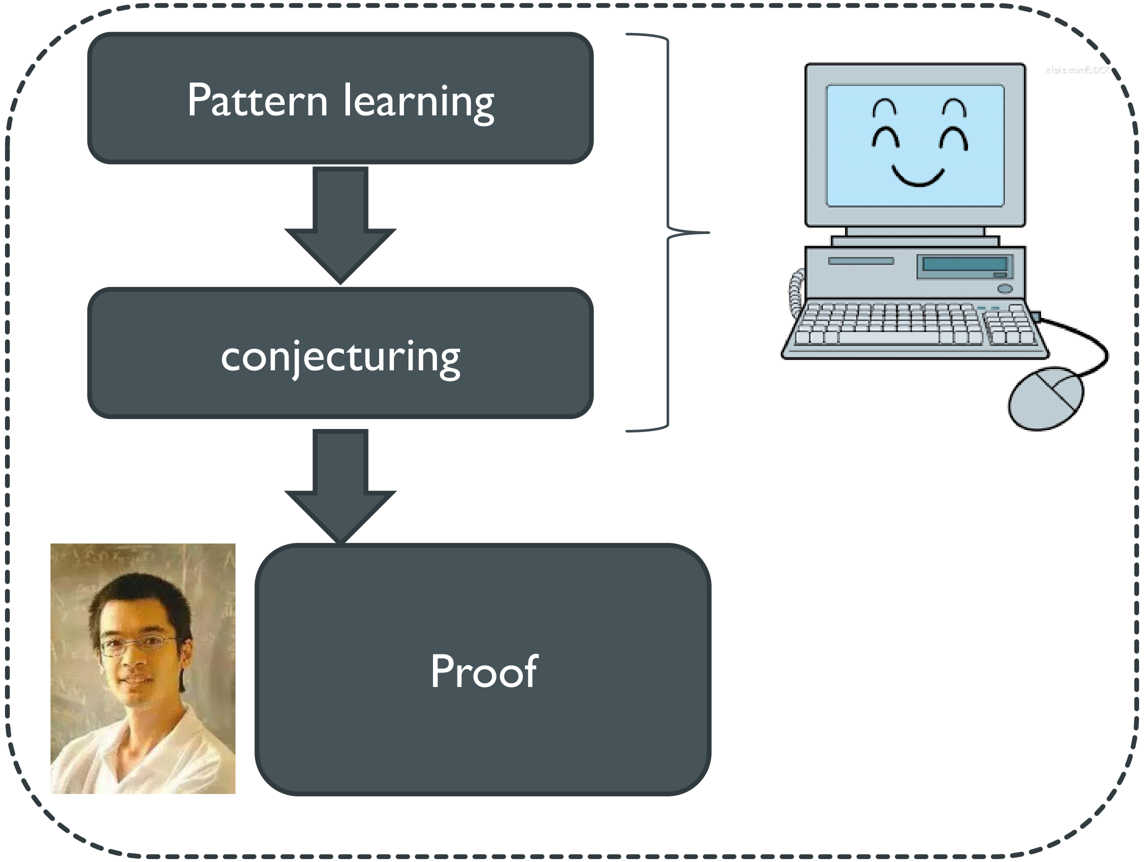
[1]Technion - Israel Institute of Technology, Haifa 3200003, Israel
[2]Google Inc., Tel Aviv 6789141, Israel

Abstract

Fundamental mathematical constants like e and π are ubiquitous in diverse fields of science, from abstract mathematics and geometry to physics, biology and chemistry. Nevertheless, for centuries new mathematical formulas relating fundamental constants have been scarce and usually discovered sporadically. In this paper we propose a novel and systematic approach that leverages algorithms for deriving new mathematical formulas for fundamental constants and help reveal their underlying structure. Our algorithms find dozens of well-known as well as previously unknown continued fraction representations of π, e, and the Riemann zeta function values. Two new conjectures produced by our algorithm, along with many others, are:

$$e = 3 + \cfrac{-1}{4 + \cfrac{-2}{5 + \cfrac{-3}{6 + \cfrac{-4}{7 + \ldots}}}} \quad , \quad \frac{4}{\pi - 2} = 3 + \cfrac{1 \cdot 3}{5 + \cfrac{2 \cdot 4}{7 + \cfrac{3 \cdot 5}{9 + \cfrac{4 \cdot 6}{11 + \ldots}}}}$$

arXiv:1907.00205



PDE-NET VERSION 1.0 AND 2.0

$$u_t = F(x, u, \nabla u, \nabla^2 u, \ldots), \quad x \in \Omega \subset \mathbb{R}^2, \quad t \in [0, T].$$

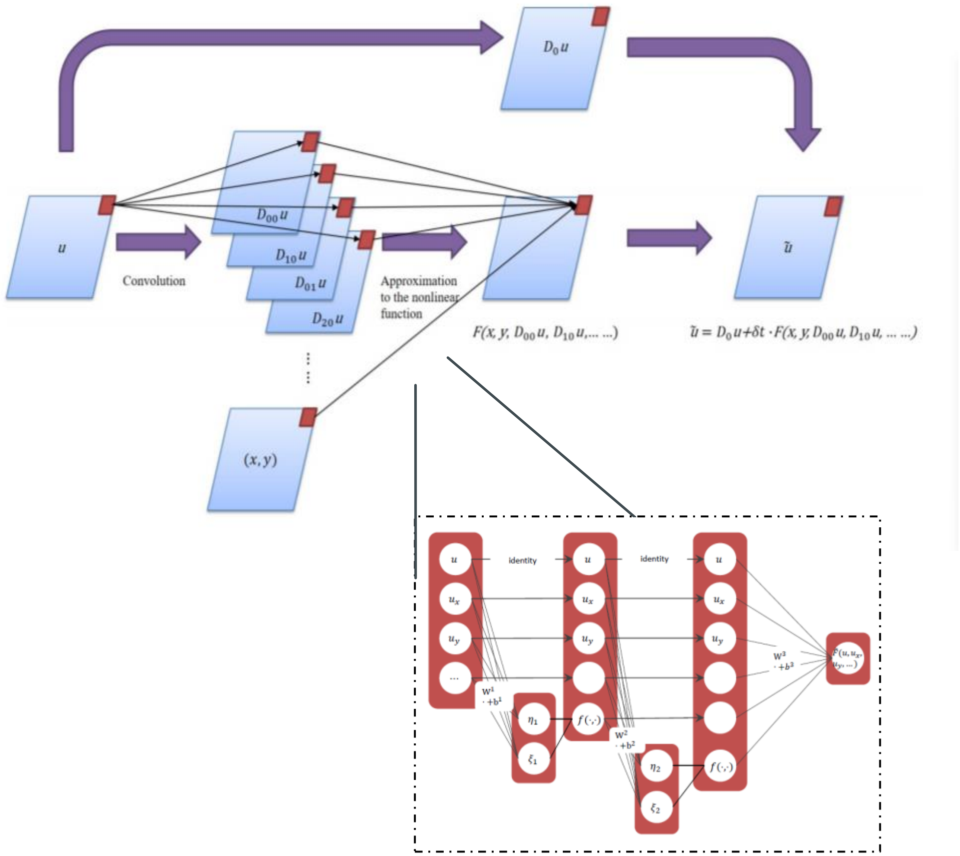
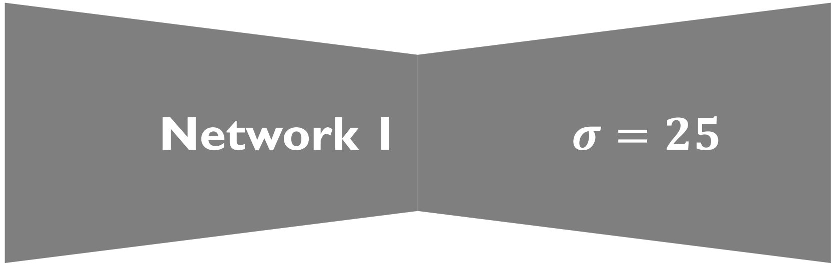


Table 1: PDE model identification.

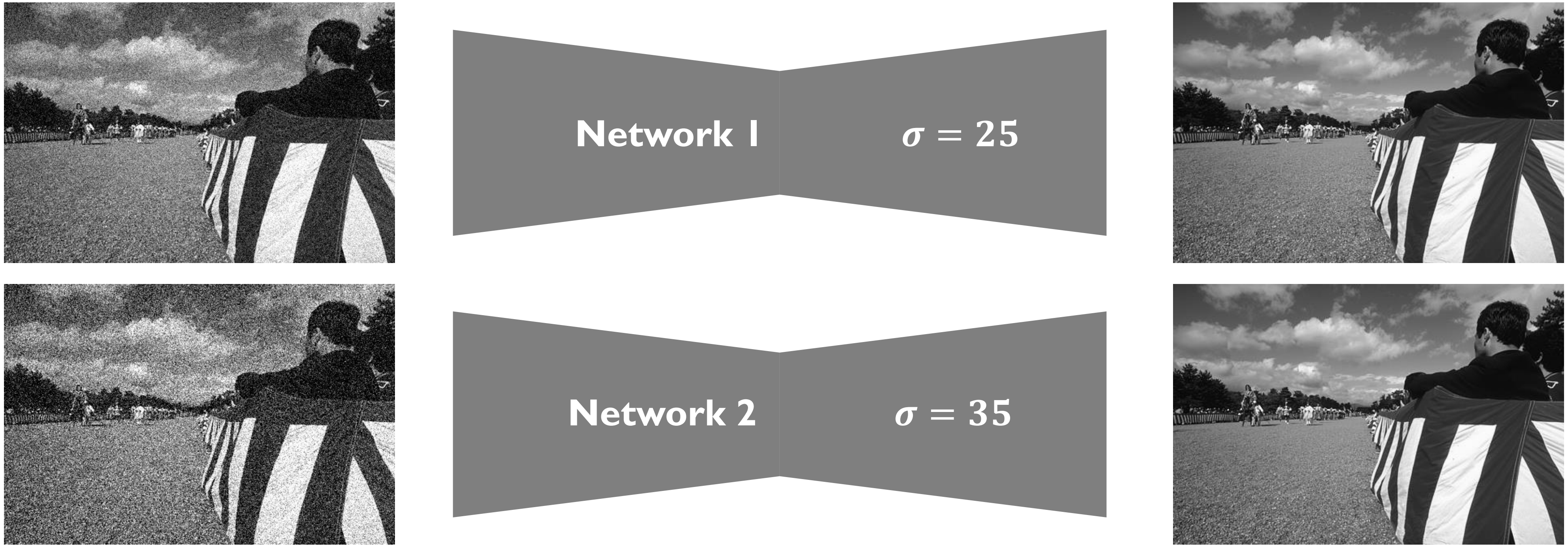
Correct PDE	$u_t = -uu_x - vu_y + 0.05(u_{xx} + u_{yy})$ $v_t = -uv_x - vv_y + 0.05(v_{xx} + v_{yy})$
Frozen-PDE-Net 2.0	$u_t = -0.906uu_x - 0.901vu_y + 0.033u_{xx} + 0.037u_{yy}$ $v_t = -0.907vv_y - 0.902uv_x + 0.039v_{xx} + 0.032v_{yy}$
PDE-Net 2.0	$u_t = -0.986uu_x - 0.972u_y v + 0.054u_{xx} + 0.052u_{yy}$ $v_t = -0.984uv_x - 0.982vv_y + 0.055v_{xx} + 0.050v_{yy}$

- Constrain the function space
- Theoretical Recover Guarantee(Coming Soon)
- Symbolic Discovery

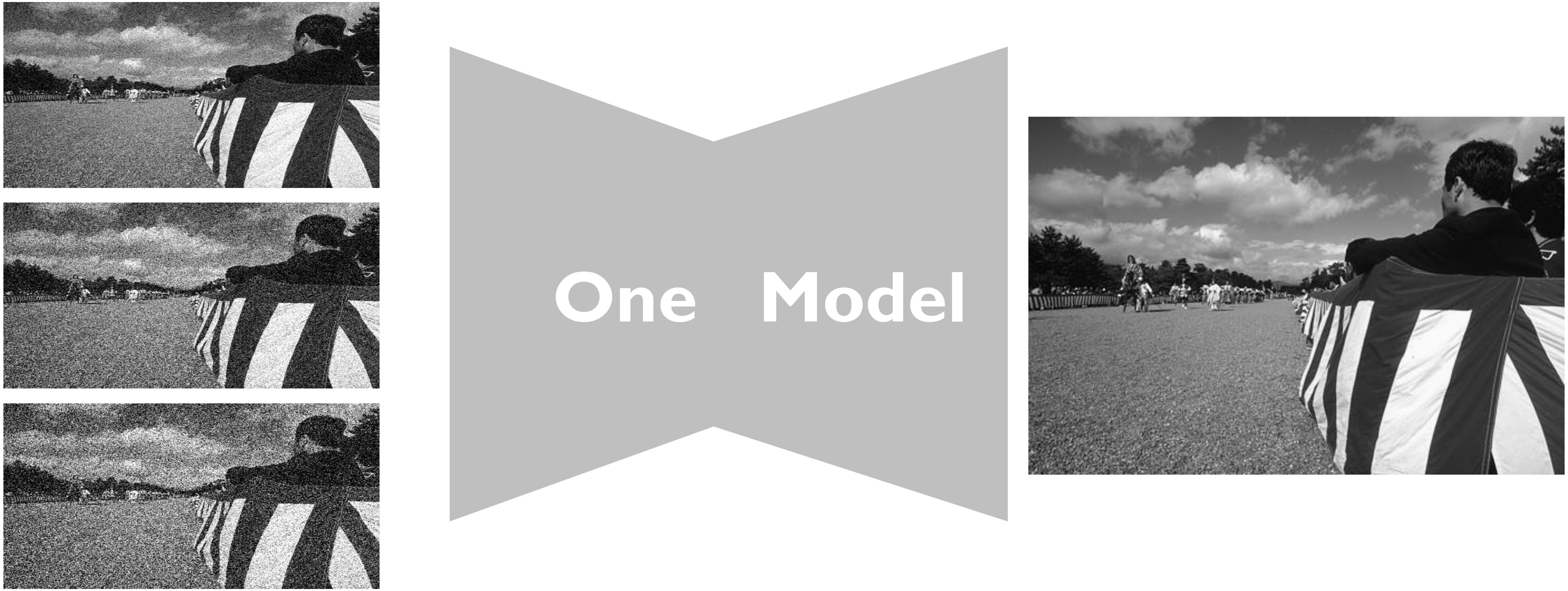
ONE NOISE LEVEL ONE NET



ONE NOISE LEVEL ONE NET



WE WANT



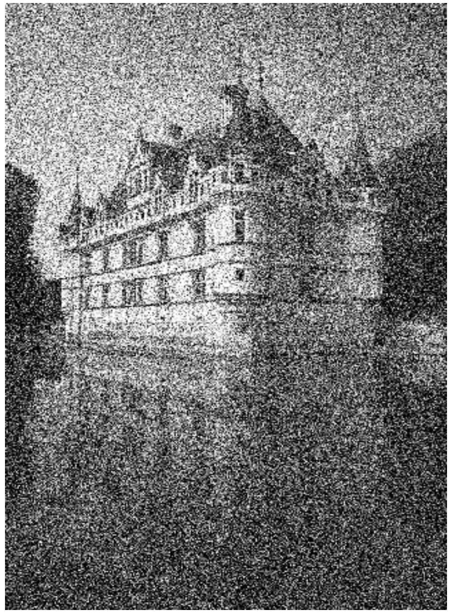
WE ALSO WANT GENERALIZATION

Train

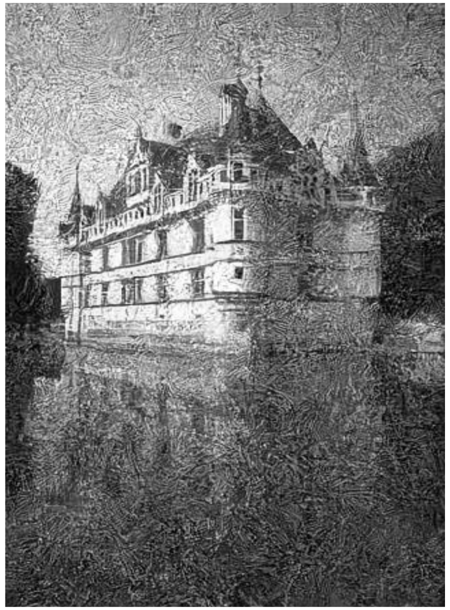
10dB Noise | 30dB Noise | 50dB Noise

Test

70dB Noise



BM3D



DnCNN

Traditional Method **VS** Deep Learning

RETHINKING TRADITIONAL FILTERING APPROACH

input

$$
\begin{cases}
\dfrac{\partial u}{\partial t} = \operatorname{div}\left(c(|\nabla u|^2)\ \nabla u \right) \quad \text{in} \quad \Omega \times (0,\mathrm{T}), \\
\dfrac{\partial u}{\partial N} = 0 \quad \text{on} \quad \partial\Omega \times (0,T), \\
u(0,x) = u_0(x) \quad \text{in} \quad \Omega,
\end{cases}
$$

Perona-Malik Equation

processing

RETHINKING TRADITIONAL FILTERING APPROACH

input

$$\begin{cases} \dfrac{\partial u}{\partial t} = \operatorname{div}\left(\boxed{c(|\nabla u|^2)}\ \nabla u \right) \ \text{ in } \ \Omega \times (0, \mathrm{T}), \\ \dfrac{\partial u}{\partial N} = 0 \ \text{ on } \ \partial\Omega \times (0, T), \\ u(0, x) = u_0(x) \ \text{ in } \ \Omega, \end{cases}$$

Perona-Malik Equation

Noisy

processing

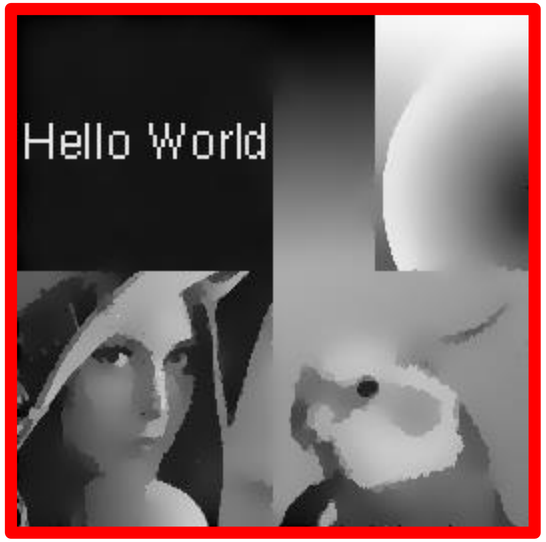
RETHINKING TRADITIONAL FILTERING APPROACH

input

$$\begin{cases} \dfrac{\partial u}{\partial t} = \operatorname{div}\left(\boxed{c(|\nabla u|^2)}\ \nabla u \right) \ \text{ in } \ \Omega \times (0, \mathrm{T}), \\ \dfrac{\partial u}{\partial N} = 0 \ \text{ on } \ \partial\Omega \times (0, T), \\ u(0, x) = u_0(x) \ \text{ in } \ \Omega, \end{cases}$$

Perona-Malik Equation

Noisy



processing

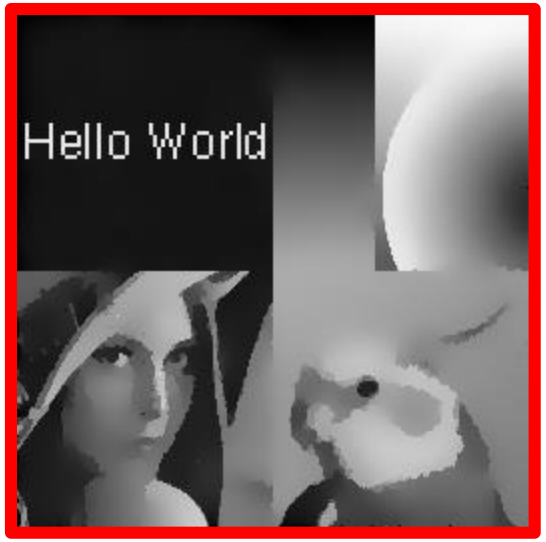
RETHINKING TRADITIONAL FILTERING APPROACH

input

$$\begin{cases} \dfrac{\partial u}{\partial t} = \operatorname{div}\left(\boxed{c(|\nabla u|^2)}\ \nabla u \right) \ \text{ in } \ \Omega \times (0, \mathrm{T}), \\ \dfrac{\partial u}{\partial N} = 0 \ \text{ on } \ \partial\Omega \times (0, T), \\ u(0, x) = u_0(x) \ \text{ in } \ \Omega, \end{cases}$$

Perona-Malik Equation

Noisy



Over-smooth

processing

MOVING ENDPOINT CONTROL VS FIXED ENDPOINT CONTROL

Supervised Loss

$$\min_{w} L\big(X(T)\big) + \int_0^{\tau} R(w(t),t)dt$$ Weight Decay

$$s.t.\dot{X} = f\big(X(t),w(t)\big),$$

Learn the weight

Deep Neural Network

MOVING ENDPOINT CONTROL VS FIXED ENDPOINT CONTROL

$$\boxed{\min_{w}} L\big(X(T)\big) + \int_0^{\tau} R(w(t),t)dt$$
$$s.t. \dot{X} = f\big(X(t), w(t)\big),$$

$$\longrightarrow$$

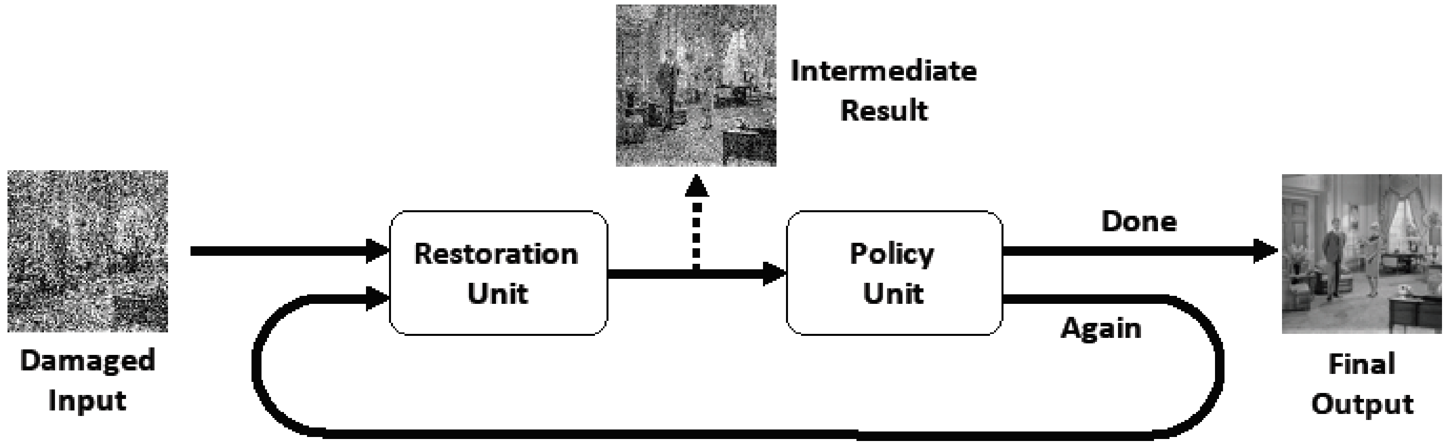
$$\boxed{\min_{w,\tau}} L\big(X(T)\big) + \int_0^{\tau} R(w(t),t)dt$$
$$s.t. \dot{X} = f\big(X(t), w(t)\big),$$

MOVING ENDPOINT CONTROL VS FIXED ENDPOINT CONTROL

$$\boxed{\min_{w}} L(X(T)) + \int_0^{\tau} R(w(t),t)dt \quad s.t. \dot{X} = f(X(t),w(t)),$$

$$\boxed{\min_{w,\tau}} L(X(T)) + \int_0^{\tau} R(w(t),t)dt \quad s.t. \dot{X} = f(X(t),w(t)),$$

Discretization

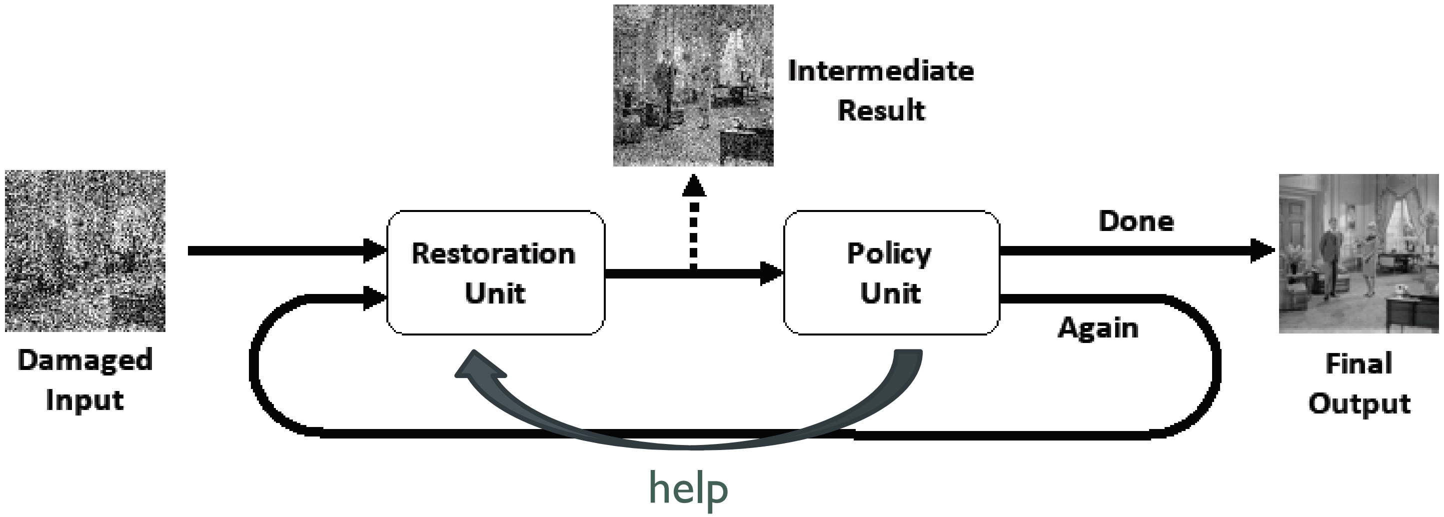


MOVING ENDPOINT CONTROL VS FIXED ENDPOINT CONTROL

$$\boxed{\min_{w}} L(X(T)) + \int_0^{\tau} R(w(t),t)dt$$
$$s.t. \dot{X} = f(X(t), w(t)),$$

$$\boxed{\min_{w,\tau}} L(X(T)) + \int_0^{\tau} R(w(t),t)dt$$
$$s.t. \dot{X} = f(X(t), w(t)),$$

Discretization



RESULT

Denoising

	BM3D	WNNM	DnCNN-B	UNLNet_5	DURR
$\sigma = 25$	28.55	28.73	29.16	28.96	29.16
$\sigma = 35$	27.07	27.28	27.66	27.50	27.72
$\sigma = 45$	25.99	26.26	26.62	26.48	26.71
$\sigma = 55$	25.26	25.49	25.80	25.64	25.91
$\sigma = 65$	24.69	24.51	23.40*	-	25.26*
$\sigma = 75$	22.63	22.71	18.73*	-	24.71*

JPEG

QF	JPEG	SA-DCT	AR-CNN	AR-CNN-B	DnCNN-3	DURR
10	27.77	28.65	28.98	28.53	29.40	29.23*
20	30.07	30.81	31.29	30.88	31.59	31.68
30	31.41	32.08	32.69	32.31	32.98	33.05
40	32.45	32.99	33.63	33.39	33.96	34.01*

GENERALIZE TO UNSEEN NOISE LEVEL

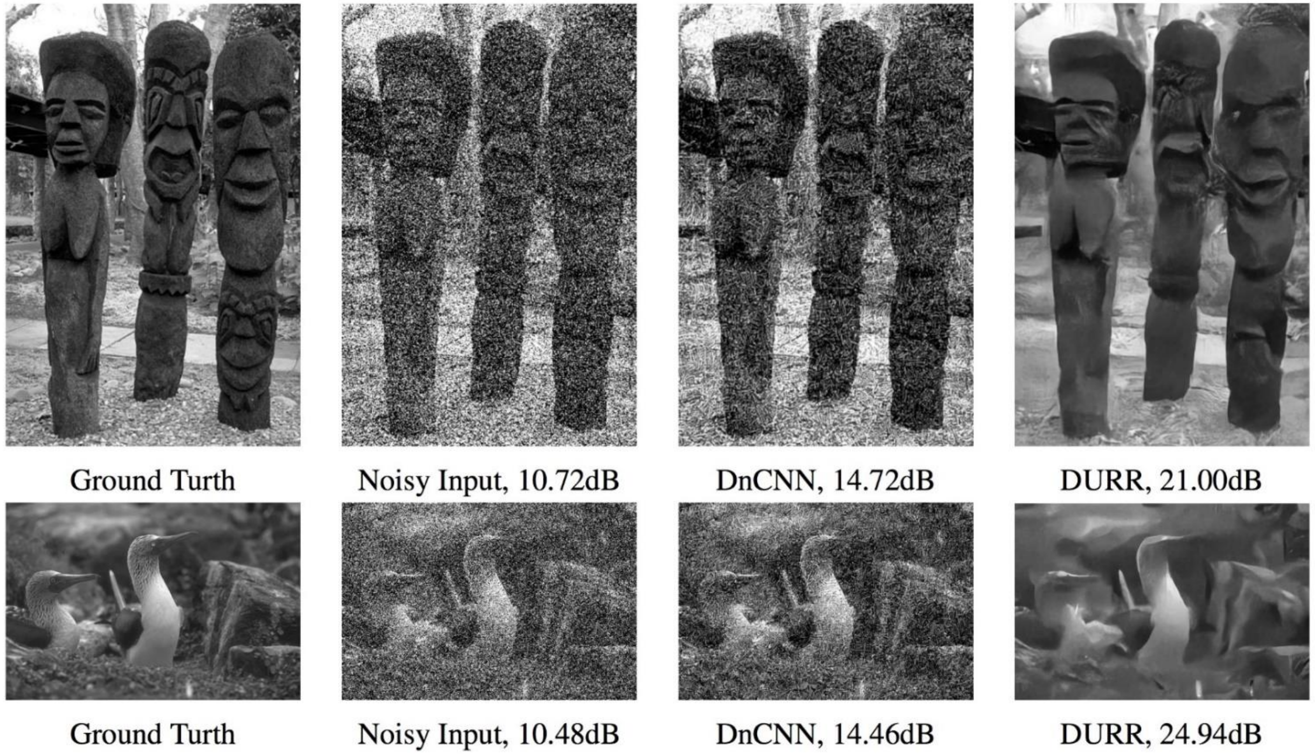
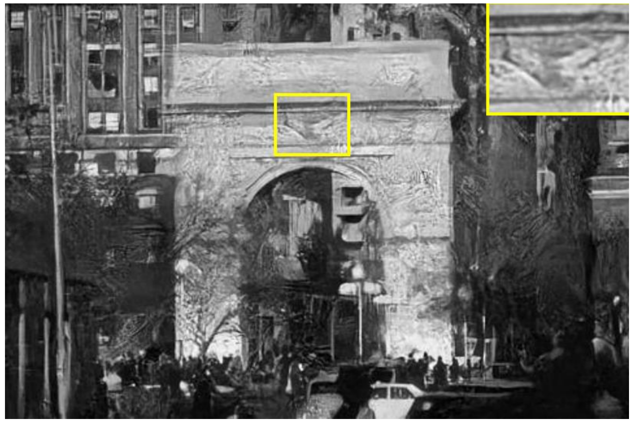
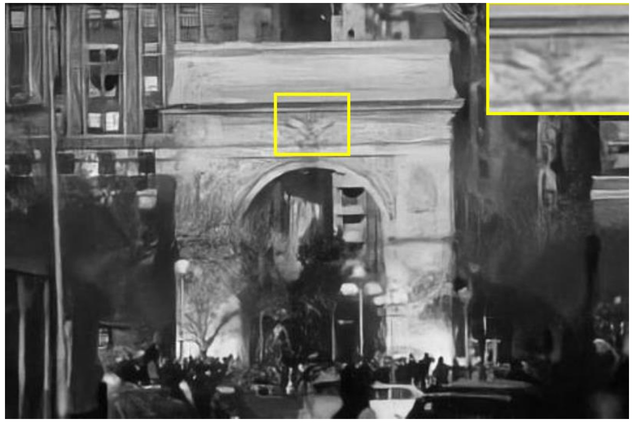


Figure 9: Denoising results of images from BSD68 with extreme noise conditions ($\sigma = 95$).



DnCNN



DURR

HOW DIFFERENTIAL EQUATION VIEW HELPS OPTIMIZATION ALGORITHM

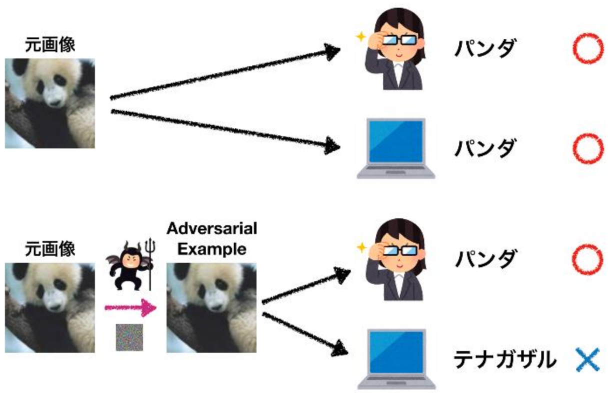
NEURAL NETWORK AWARE OPTIMIZATION METHODS

TOWARDS DEEP LEARNING MODELS RESISTANT TO ADVERSARIAL ATTACKS

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

Robust Optimization

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



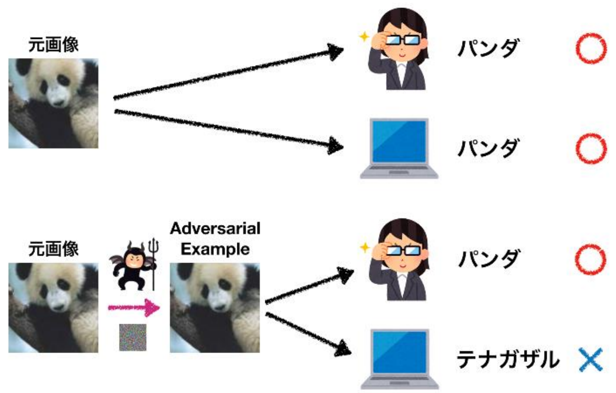
TOWARDS DEEP LEARNING MODELS RESISTANT TO ADVERSARIAL ATTACKS

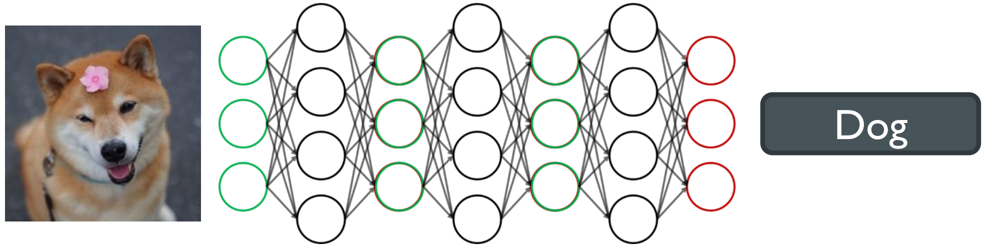
$$\min_{\theta} \rho(\theta), \quad \text{where} \quad \rho(\theta) = \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\max_{\delta\in\mathcal{S}} L(\theta, x+\delta, y)\right]$$

Robust Optimization

Problem:

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





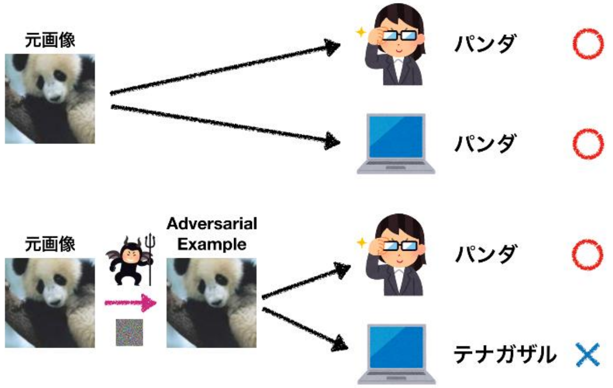
TOWARDS DEEP LEARNING MODELS RESISTANT TO ADVERSARIAL ATTACKS

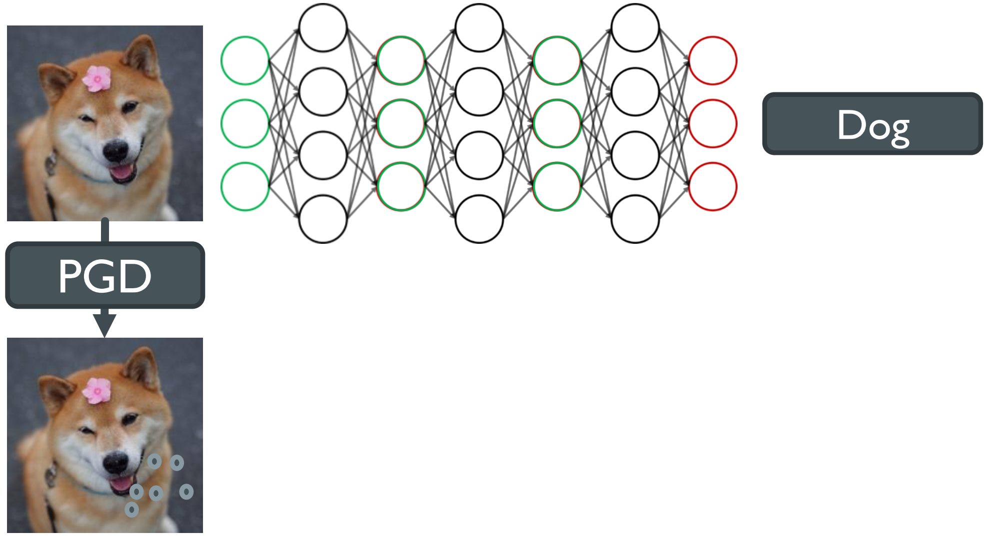
$$\min_{\theta} \rho(\theta), \quad \text{where} \quad \rho(\theta) = \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\max_{\delta\in\mathcal{S}} L(\theta, x+\delta, y)\right]$$

Robust Optimization

Problem:

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





Adversarial example

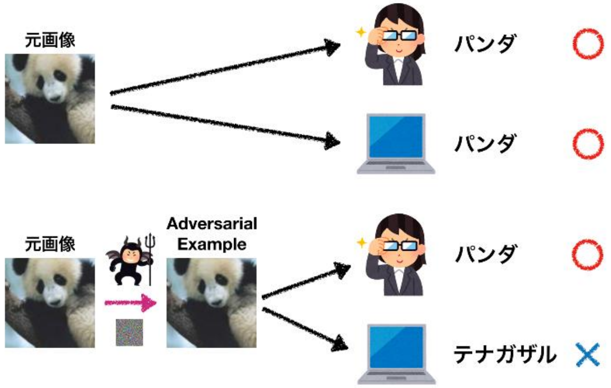
TOWARDS DEEP LEARNING MODELS RESISTANT TO ADVERSARIAL ATTACKS

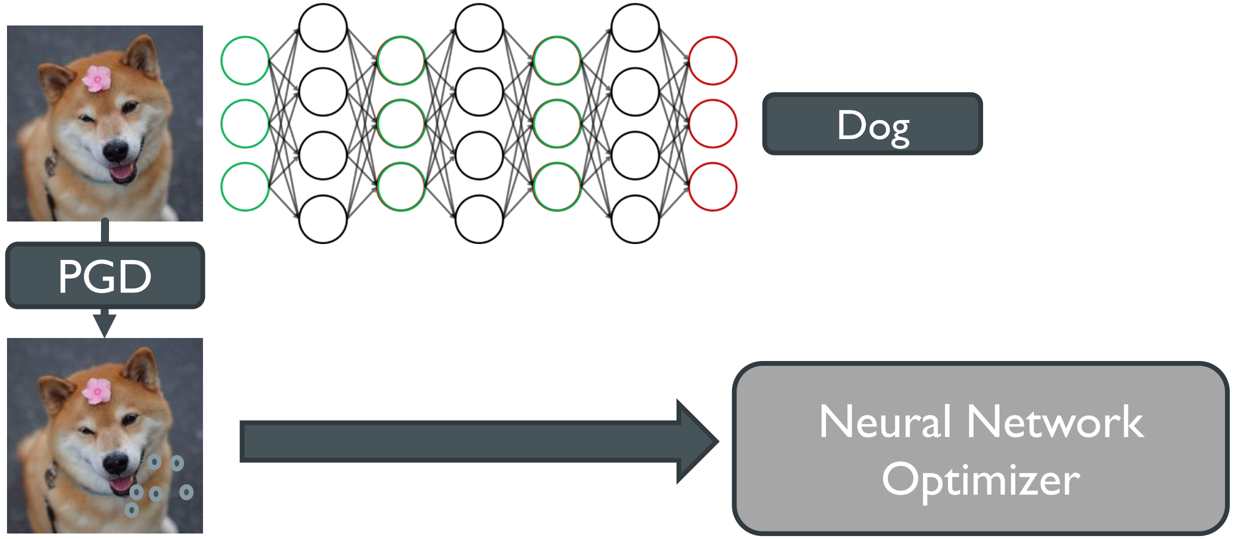
$$\min_{\theta} \rho(\theta), \quad \text{where} \quad \rho(\theta) = \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\max_{\delta\in\mathcal{S}} L(\theta, x+\delta, y)\right]$$

Robust Optimization

Problem:

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





Adversarial example

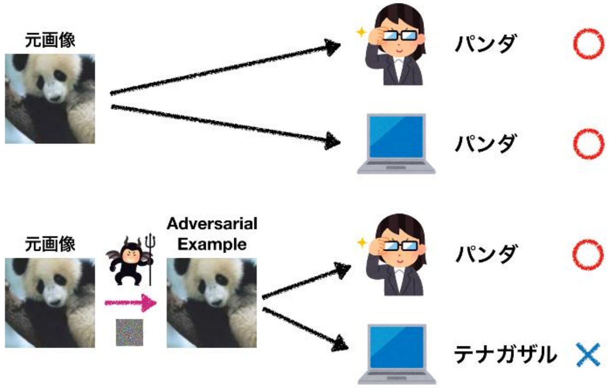
TOWARDS DEEP LEARNING MODELS RESISTANT TO ADVERSARIAL ATTACKS

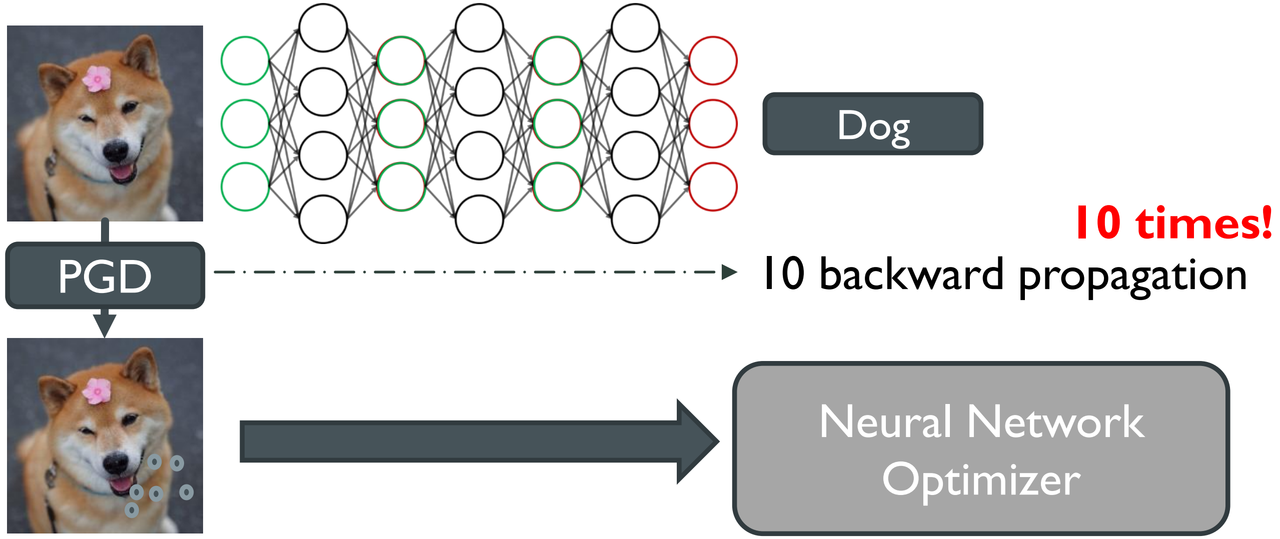
$$\min_{\theta} \rho(\theta), \quad \text{where} \quad \rho(\theta) = \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\max_{\delta\in\mathcal{S}} L(\theta, x+\delta, y)\right]$$

Robust Optimization

Problem:

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





Adversarial example

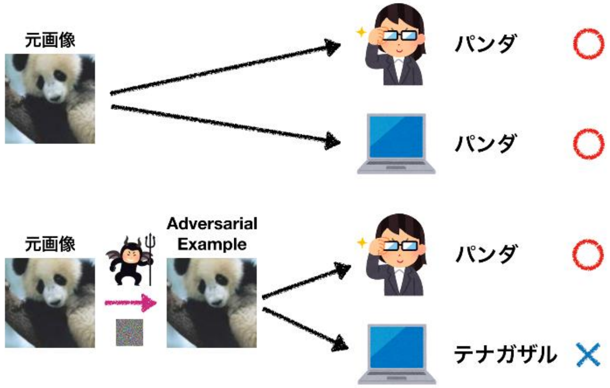
TOWARDS DEEP LEARNING MODELS RESISTANT TO ADVERSARIAL ATTACKS

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

Robust Optimization

Problem:

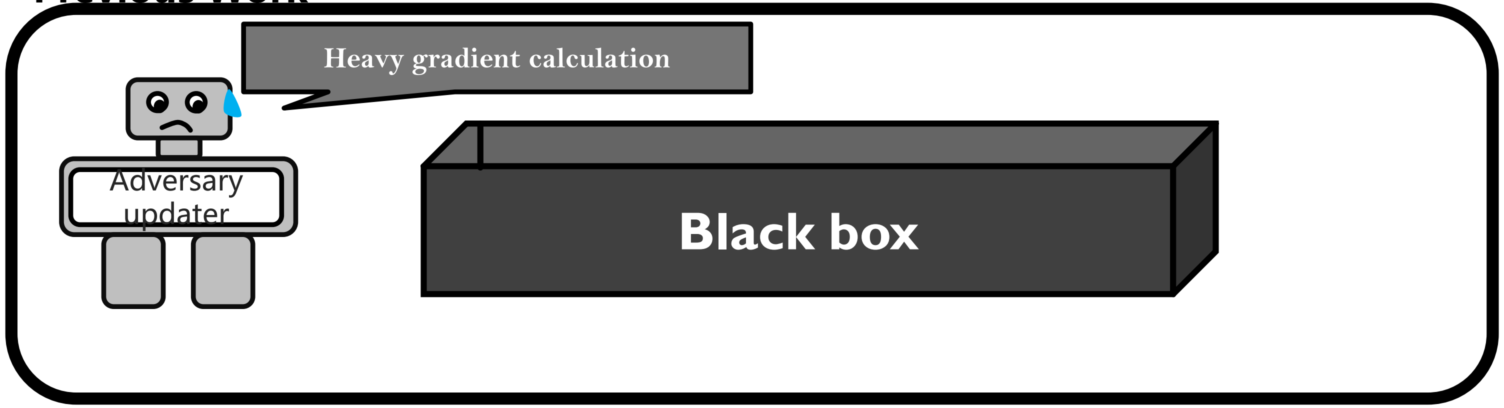
- More capacity and stronger adversaries decrease transferability. Always 10 times wider
- PGD training is expansive!



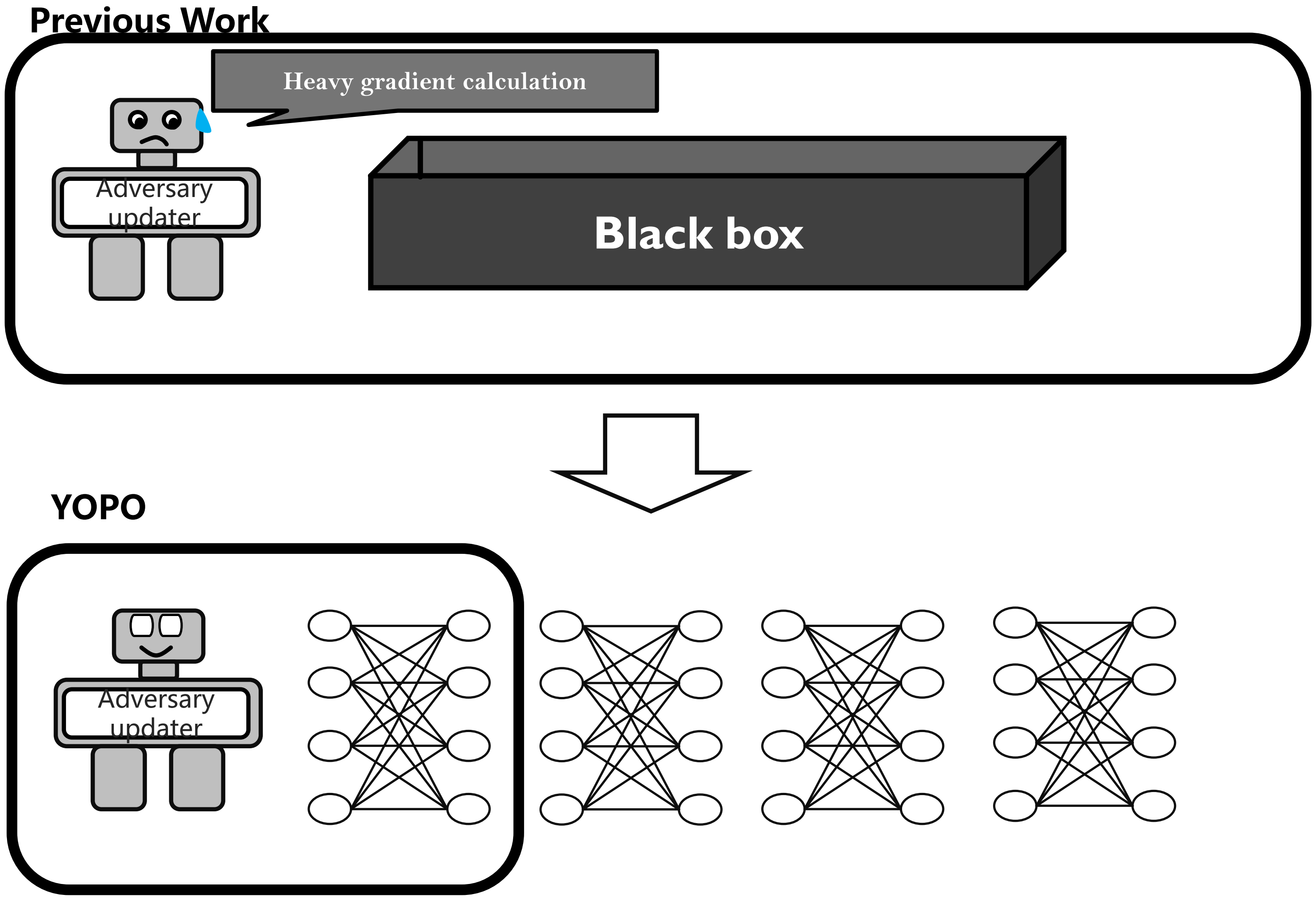
Can adversarial training cheaper

TAKE NEURAL NETWORK ARCHITECTURE INTO CONSIDERATION

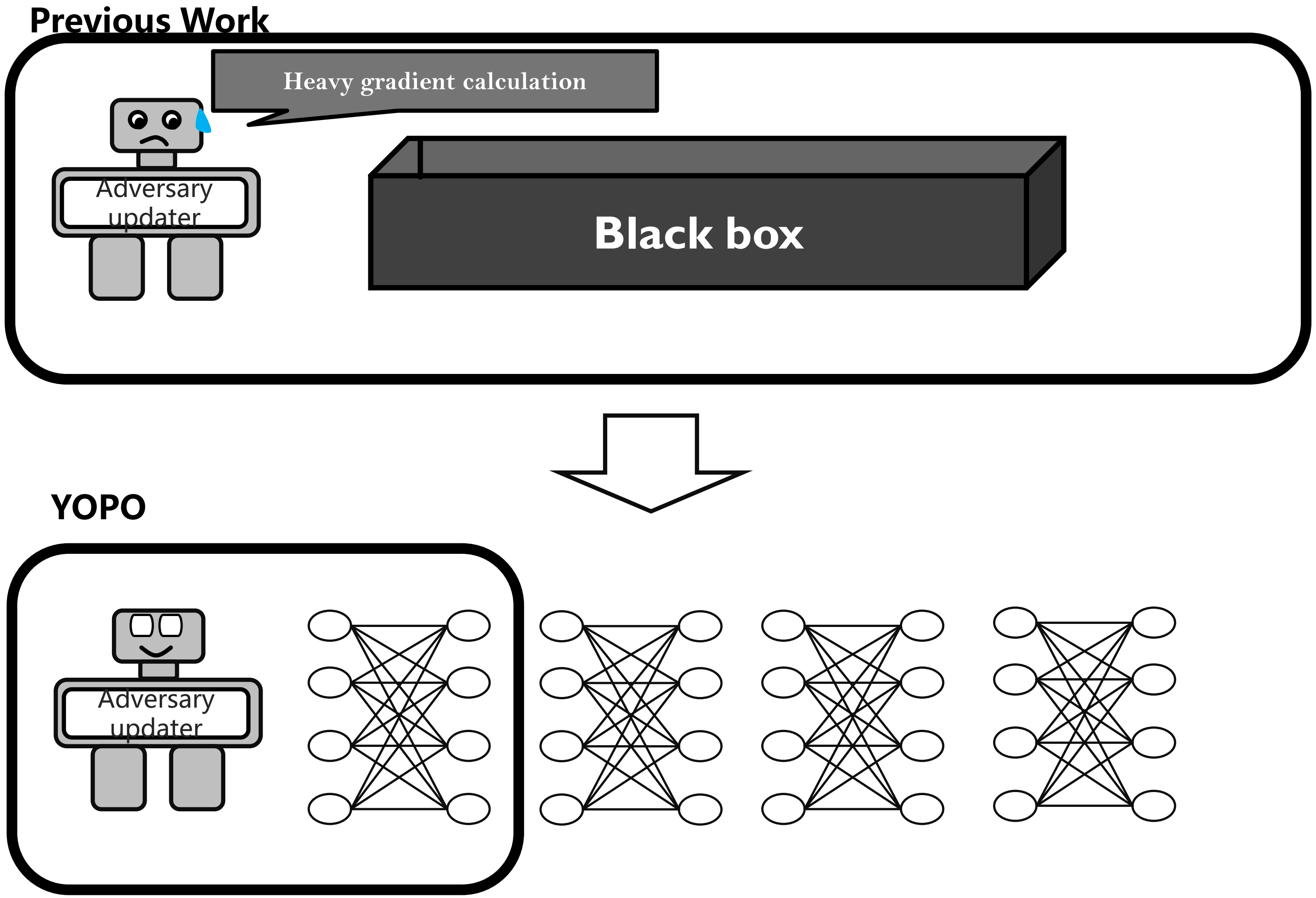
Previous Work



TAKE NEURAL NETWORK ARCHITECTURE INTO CONSIDERATION



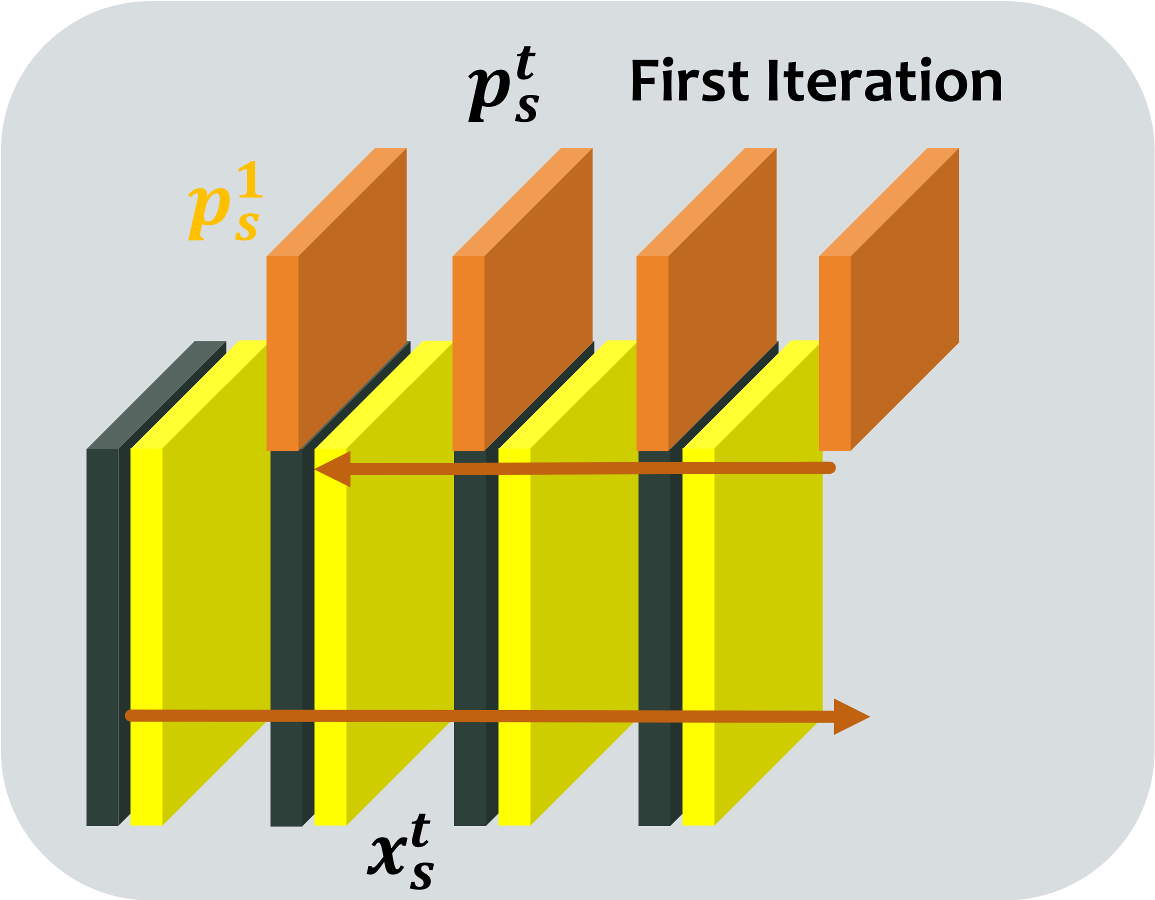
TAKE NEURAL NETWORK ARCHITECTURE INTO CONSIDERATION



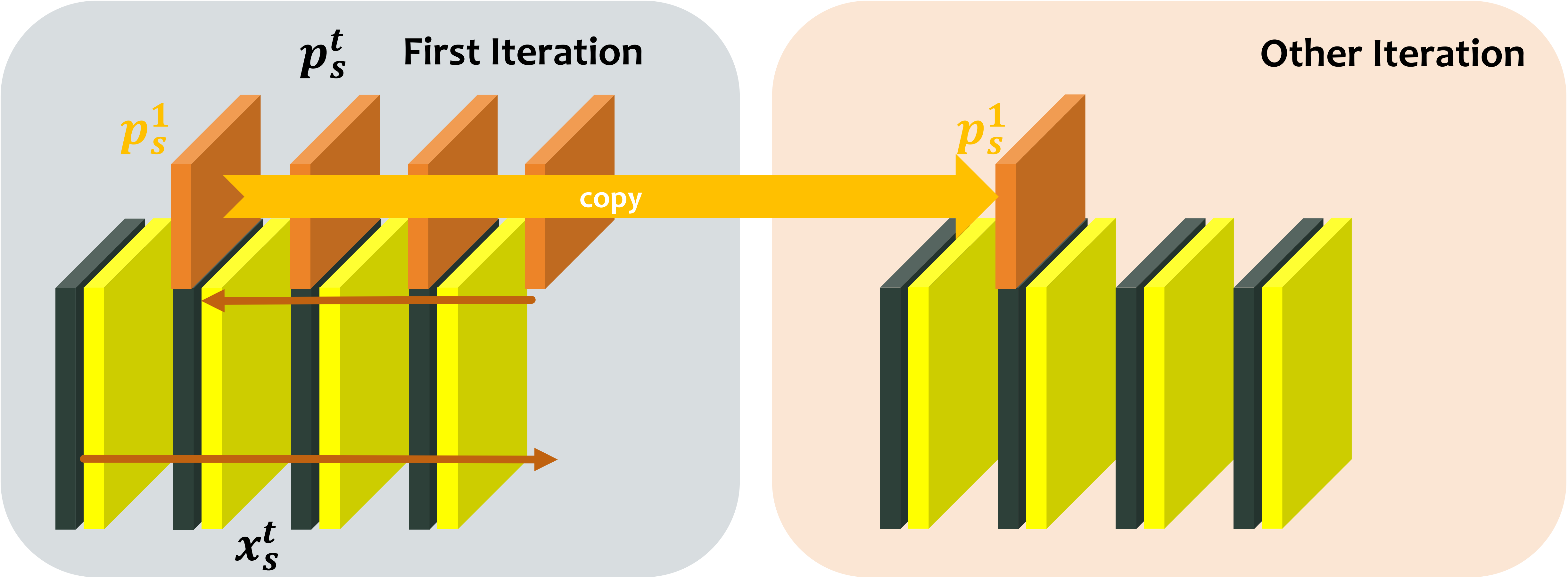
<u>Why ODE's view help?</u>

Using control can exploit the structure of neural network

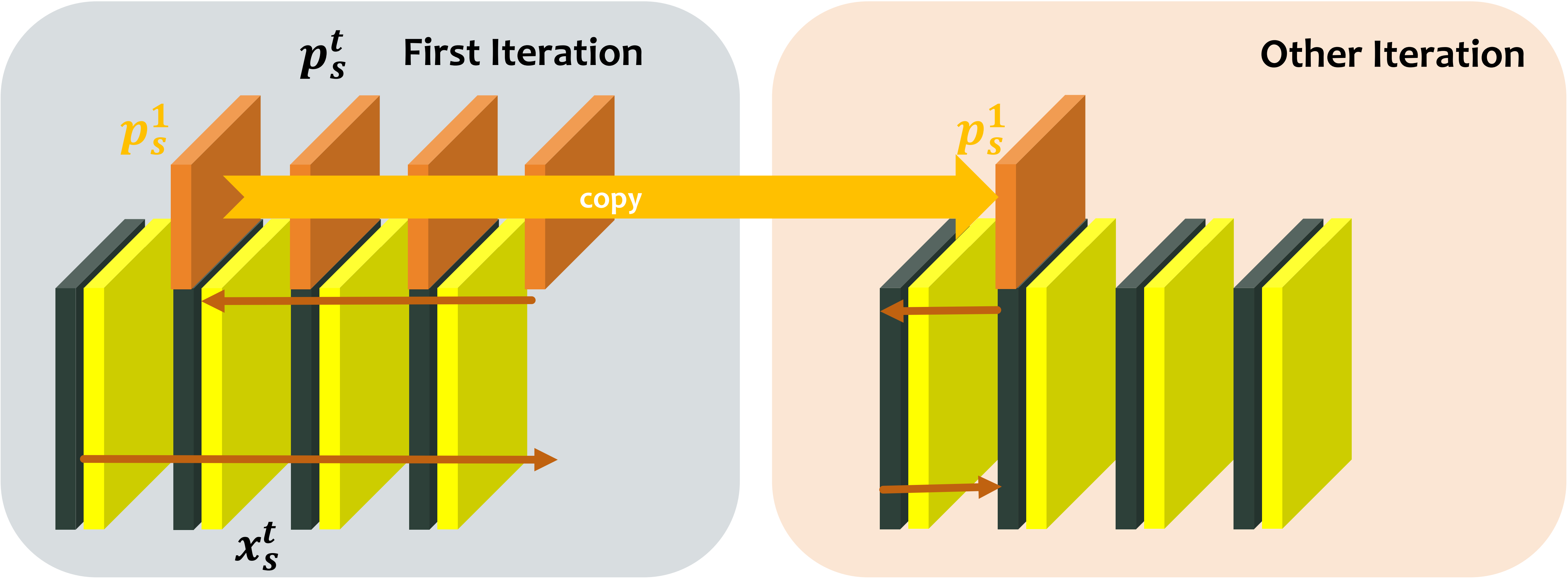
YOPO(YOU ONLY PROPAGATE ONCE)



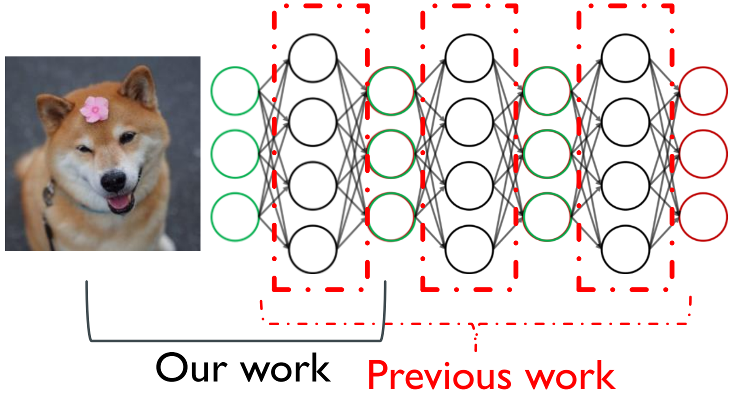
YOPO(YOU ONLY PROPAGATE ONCE)



YOPO(YOU ONLY PROPAGATE ONCE)



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]

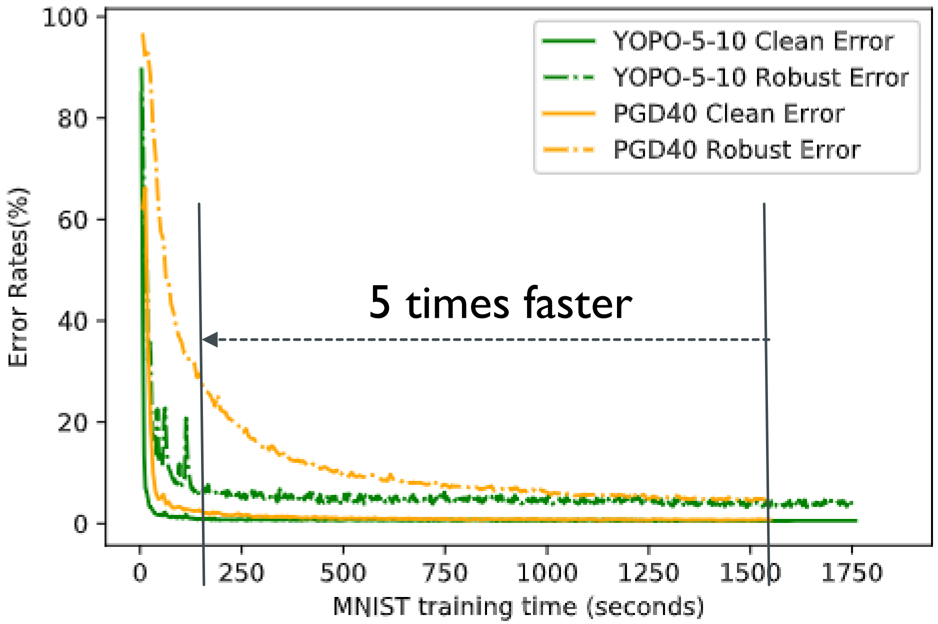
- Our idea: Control can decouple the gradient back propagation with the adversary updating.

RESULT

Training Methods	Clean Data	PGD-20 Attack	Training Time (mins)
Natural train	95.03%	0.00%	233
PGD-3 [24]	90.07%	39.18%	1134
PGD-5 [24]	89.65%	43.85%	1574
PGD-10 [24]	87.30%	47.04%	2713
Free-8 [28][1]	86.29%	47.00%	667
YOPO-3-5 (Ours)	87.27%	43.04%	299
YOPO-5-3 (Ours)	86.70%	47.98%	**476**

[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

DIFFERENTIAL GAME

$$\min_{\theta} \max_{\|\eta\|_\infty \le \epsilon} J(\theta, \eta) := \frac{1}{N}\sum_{i=1}^{N} \ell_i(x_{i,T}) + \frac{1}{N}\sum_{i=1}^{N}\sum_{t=0}^{T-1} R_t(x_{i,t}; \theta_t) \tag{2}$$

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

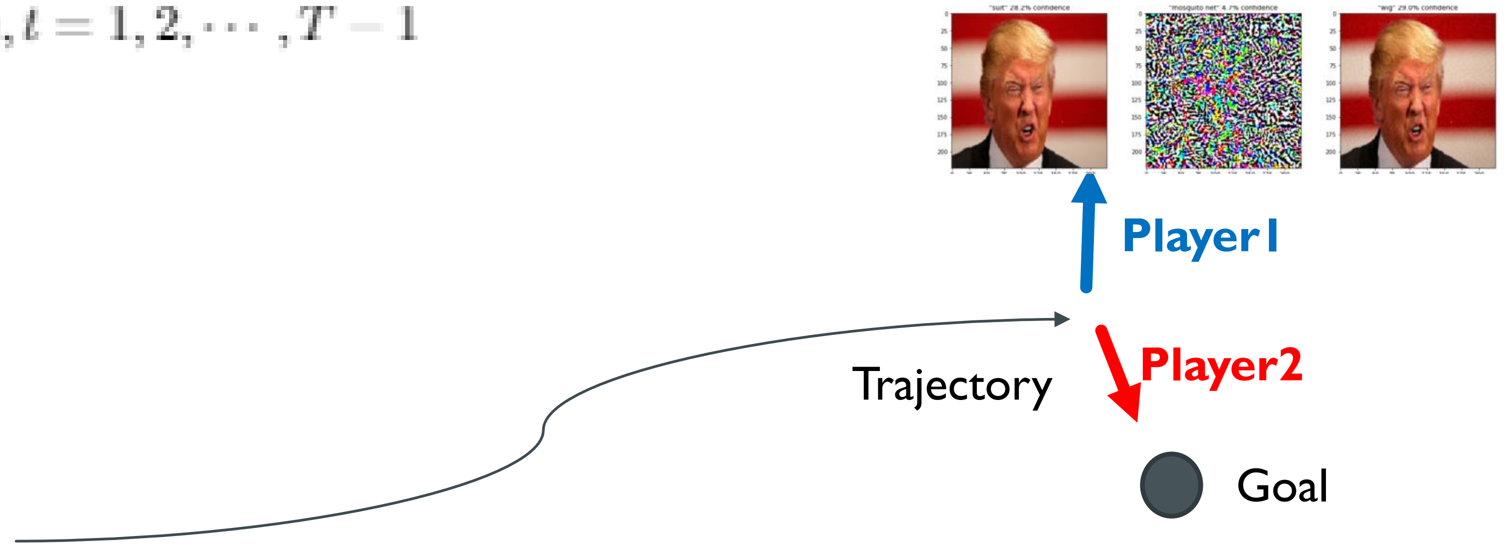
$$x_{i,t+1} = f_t(x_{i,t}, \theta_t), t = 1, 2, \cdots, T-1$$

DIFFERENTIAL GAME

$$\min_{\theta} \max_{\|\eta\|_\infty \le \epsilon} J(\theta,\eta) := \frac{1}{N}\sum_{i=1}^{N} \ell_i(x_{i,T}) + \frac{1}{N}\sum_{i=1}^{N}\sum_{t=0}^{T-1} R_t(x_{i,t};\theta_t) \qquad (2)$$

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

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

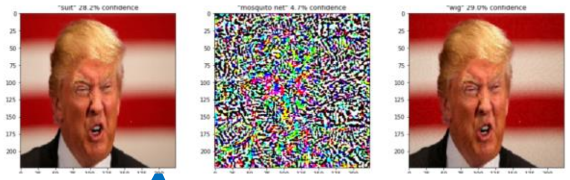


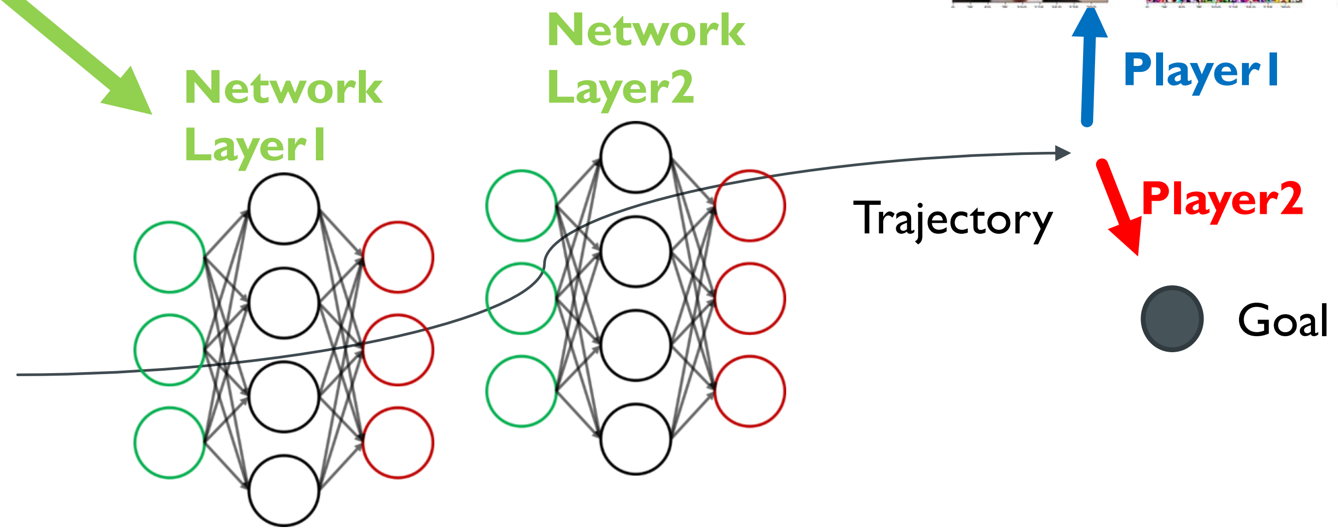
DIFFERENTIAL GAME

$$\min_{\theta} \max_{\|\eta\|_\infty \le \epsilon} J(\theta, \eta) := \frac{1}{N}\sum_{i=1}^{N} \ell_i(x_{i,T}) + \frac{1}{N}\sum_{i=1}^{N}\sum_{t=0}^{T-1} R_t(x_{i,t}; \theta_t) \tag{2}$$

$$\boxed{\begin{aligned} \text{subject to } x_{i,1} &= f_0(x_{i,0} + \eta; \theta_0), i = 1, 2, \cdots, N \\ x_{i,t+1} &= f_t(x_{i,t}, \theta_t), t = 1, 2, \cdots, T-1 \end{aligned}}$$

Composition Structure





WHY DECOUPLING

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

$$x_{s,t+1}^* = \nabla_p H_t(x_{s,t}^*, p_{s,t+1}^*, \theta_t^*), \qquad x_{s,0}^* = x_{s,0} + \eta \tag{5}$$

$$p_{s,t}^* = \nabla_x H_t(x_{s,t}^*, p_{s,t+1}^*, \theta_t^*), \qquad p_{s,T}^* = -\frac{1}{S}\nabla\Phi(x_{s,T}^*) \tag{6}$$

At the same the the parameter of the first layer θ_0^* satisfies

Adversary only appears here

$$\sum_{s=1}^{S} H_t(x_{s,0} + \hat{\eta}, p_{s,t+1}^*, \theta_0^*), \forall \theta \in \Theta_t \geq \sum_{s=1}^{S} H_0(x_{s,0}^*, p_{s,1}^*, \theta_0^*) \geq \sum_{s=1}^{S} H_0(x_{s,0}^*, p_{s,1}^*, \theta), \forall \theta \in \Theta_0, \|\hat{\eta}\|_\infty \leq \epsilon \tag{7}$$

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

$$\sum_{s=1}^{S} H_t(x_{s,t}^*, p_{s,t+1}^*, \theta_t^*) \geq \sum_{s=1}^{S} H_t(x_{s,t}^*, p_{s,t+1}^*, \theta), \forall \theta \in \Theta_t \tag{8}$$

KKT Condition

WHY DECOUPLING

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

$$x_{s,t+1}^* = \nabla_p H_t(x_{s,t}^*, p_{s,t+1}^*, \theta_t^*), \qquad x_{s,0}^* = x_{s,0} + \eta \tag{5}$$

$$p_{s,t}^* = \nabla_x H_t(x_{s,t}^*, p_{s,t+1}^*, \theta_t^*), \qquad p_{s,T}^* = -\frac{1}{S}\nabla\Phi(x_{s,T}^*) \tag{6}$$

Forward propagation

Backward propagation

Feature space

At the same the the parameter of the first layer θ_0^* satisfies

$$\sum_{s=1}^{S} H_t(x_{s,0} + \hat{\eta}, p_{s,t+1}^*, \theta_0^*), \forall\theta \in \Theta_t \geq \sum_{s=1}^{S} H_0(x_{s,0}^*, p_{s,1}^*, \theta_0^*) \geq \sum_{s=1}^{S} H_0(x_{s,0}^*, p_{s,1}^*, \theta), \forall\theta \in \Theta_0, \|\hat{\eta}\|_\infty \leq \epsilon \tag{7}$$

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

$$\sum_{s=1}^{S} H_t(x_{s,t}^*, p_{s,t+1}^*, \theta_t^*) \geq \sum_{s=1}^{S} H_t(x_{s,t}^*, p_{s,t+1}^*, \theta), \forall\theta \in \Theta_t \tag{8}$$

Weight space

WHY DECOUPLING

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

$$x_{s,t+1}^* = \nabla_p H_t(x_{s,t}^*, p_{s,t+1}^*, \theta_t^*), \qquad x_{s,0}^* = x_{s,0} + \eta \tag{5}$$

$$p_{s,t}^* = \nabla_x H_t(x_{s,t}^*, p_{s,t+1}^*, \theta_t^*), \qquad p_{s,T}^* = -\frac{1}{S}\nabla\Phi(x_{s,T}^*) \tag{6}$$

Forward propagation

Backward propagation

Feature space

At the same the the parameter of the first layer θ_0^* satisfies

$$\sum_{s=1}^{S} H_t(x_{s,0} + \hat{\eta}, p_{s,t+1}^*, \theta_0^*), \forall\theta \in \Theta_t \geq \sum_{s=1}^{S} H_0(x_{s,0}^*, p_{s,1}^*, \theta_0^*) \geq \sum_{s=1}^{S} H_0(x_{s,0}^*, p_{s,1}^*, \theta), \forall\theta \in \Theta_0, \|\hat{\eta}\|_\infty \leq \epsilon \tag{7}$$

YOPO-m-n: Gradient way to solve KKT

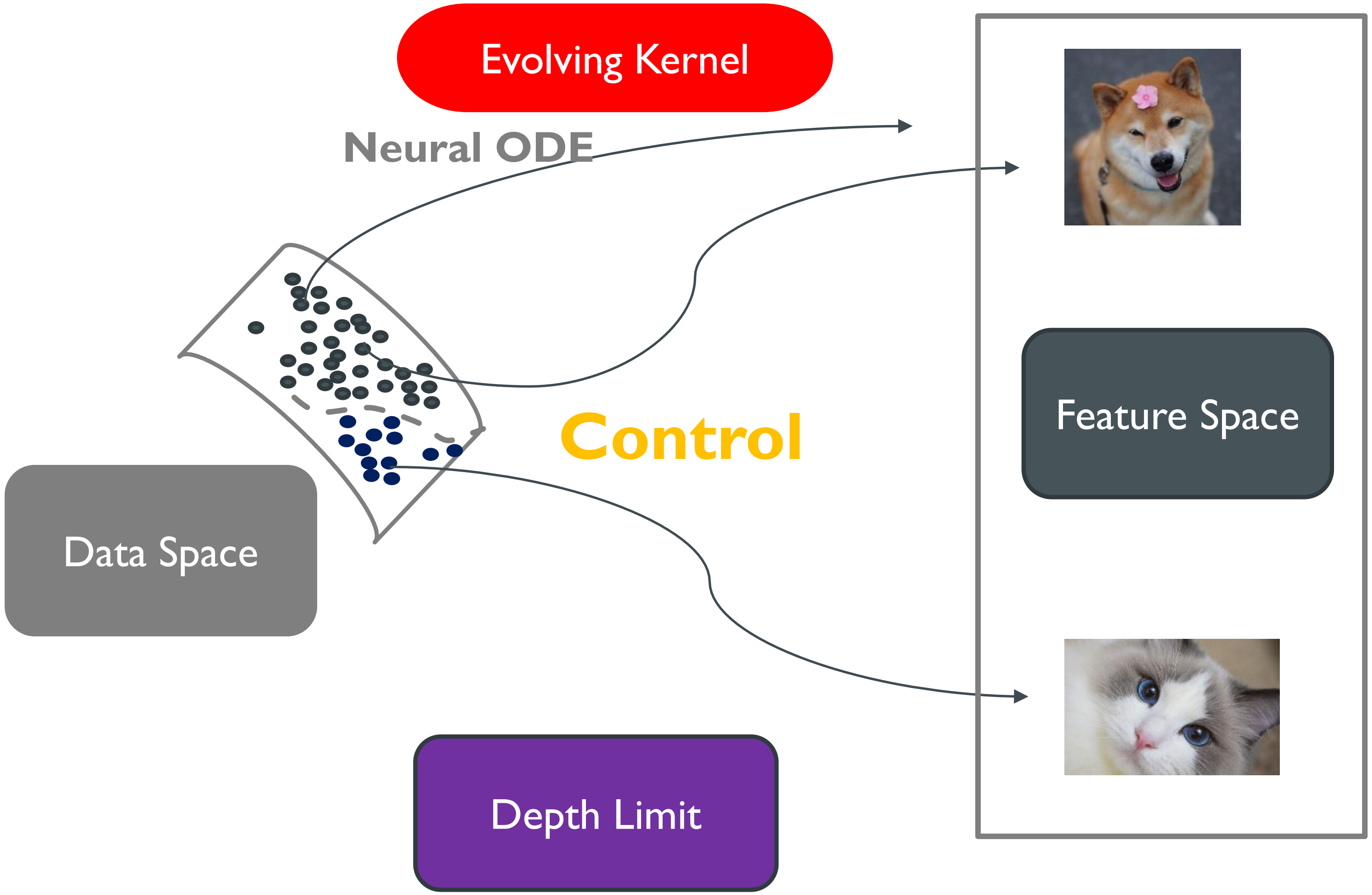
Weight space

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

Gradient ascent to *argmax* of *H*

$$\sum_{s=1}^{S} H_t(x_{s,t}^*, p_{s,t+1}^*, \theta_t^*) \geq \sum_{s=1}^{S} H_t(x_{s,t}^*, p_{s,t+1}^*, \theta), \forall\theta \in \Theta_t \tag{8}$$

TAKE HOME MESSAGE



Architecture

Optimization

Physic

• • •

Other

THANK YOU AND QUESTIONS?

Long Z, Lu Y, Ma X, Dong B. PDE-Net: Learning PDEs from Data arXiv:1710.09668. ICML2018

Lu Y, Zhong A, Li Q, Dong B. Beyond Finite Layer Neural Networks: Bridging Deep Architectures and Numerical Differential Equations arXiv:1710.10121. ICML2018

Zhang S, Lu Y, Liu J, Dong B. Dynamically Unfolding Recurrent Restorer: A Moving Endpoint Control Method for Image Restoration arXiv:1805.07709. ICLR2019

Long Z, Lu Y, Dong B. " PDE-Net 2.0: Learning PDEs from Data with A Numeric-Symbolic Hybrid Deep Network"arXiv:1812.04426. Major Revision JCP.

Dinghuai Zhang, Tianyuan Zhang, Yiping Lu, Zhanxing Zhu, Bin Dong. You Only Propogate Once: Accelerating Adversarial Training via Maximal Principle arXiv:1905.00877

Yiping Lu, Di He, Zhuohan Li, Zhiqing Sun, Bin Dong, Tao Qin, Liwei Wang, Tie-yan Liu. Understanding and Improving Transformer From a Multi-Particle Dynamic System Point of View. arXiv preprint arXiv:1906.02762, 2019.

Bin Dong, Haochen Ju, Yiping Lu, Zuoqiang Shi. CURE: Curvature Regularization For Missing Data Recovery arXiv preprint arXiv:1901.09548, 2019.

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

Always looking forward to cooperation opportunities

Contact: ypluy@stanford.edu

Acknowledgement

Bin Dong

Di He

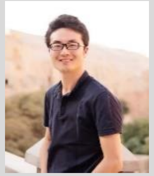
Aoxiao Zhong

Xiaoshuai Zhang

Zhuohan Li

Jikai Hou
Tianyuan Zhang
Dinghuai Zhang

Zhiqing Sun



Zhanxing Zhu

Liwei Wang

Quanzheng Li