On the interplay of network structure and gradient convergence in deep learning

Vamsi K. Ithapu[⋆] Sathya N. Ravi[⋆] Vikas Singh[†,⋆]

[⋆] Computer Sciences [†] Biostatistics and Medical Informatics

University of Wisconsin Madison

Sep 28, 2016

Overview

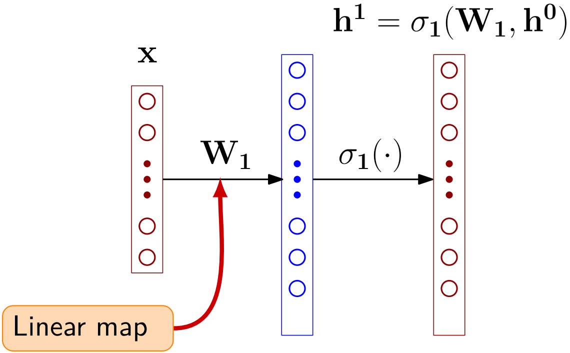
1. Background
   - Motivation
2. Problem
   - Solution strategy
   - Single-layer Networks
   - Multi-layer Networks
3. Discussion

Deep Learning – *Neural Networks*

$\mathbf{x}$: inputs, $\mathbf{h}$: hidden representations, $\mathbf{y}$: outputs
Training data $\{\mathbf{x}, \mathbf{y}\} \in \mathcal{X}$

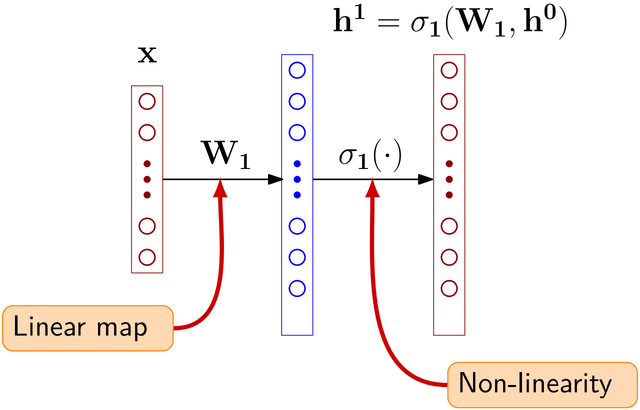
Deep Learning – *Neural Networks*

x: inputs, **h**: hidden representations, **y**: outputs
Training data $\{\mathbf{x}, \mathbf{y}\} \in \mathcal{X}$



Deep Learning – *Neural Networks*

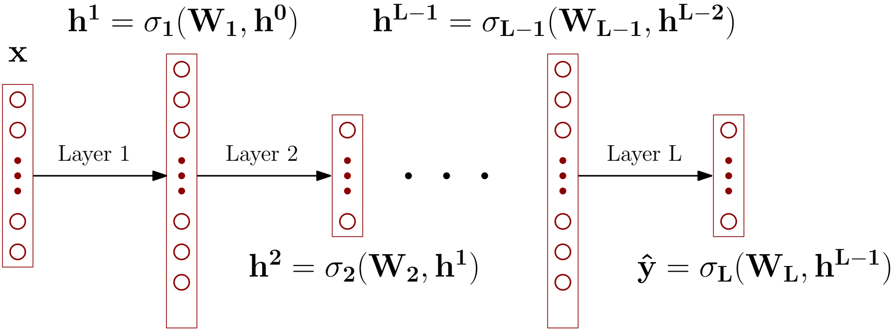
x: inputs, **h**: hidden representations, **y**: outputs
Training data $\{\mathbf{x}, \mathbf{y}\} \in \mathcal{X}$



Deep Learning – *Neural Networks*

x: inputs, **h**: hidden representations, **y**: outputs

Depth L Network



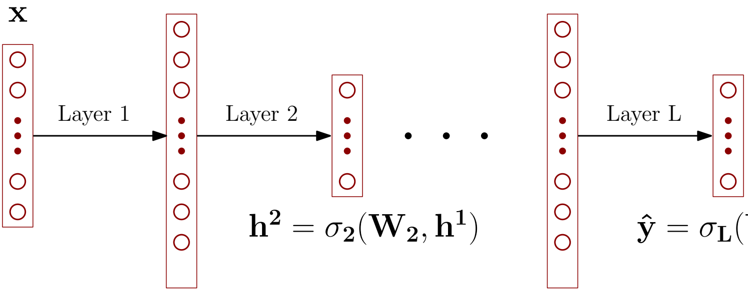
Deep Learning – *Neural Networks*

$\sigma(\cdot)$: Nonlinear
Monotonic
Non-convex
Non-smooth

x: inputs, **h**: hidden representations, **y**: outputs

Depth L Network

$$\mathbf{h^1} = \sigma_\mathbf{1}(\mathbf{W_1}, \mathbf{h^0}) \qquad \mathbf{h^{L-1}} = \sigma_\mathbf{L-1}(\mathbf{W_{L-1}}, \mathbf{h^{L-2}})$$



$$\mathbf{h^2} = \sigma_\mathbf{2}(\mathbf{W_2}, \mathbf{h^1}) \qquad \hat{\mathbf{y}} = \sigma_\mathbf{L}(\mathbf{W_L}, \mathbf{h^{L-1}})$$

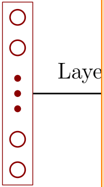
Deep Learning – *Neural Networks*

x: inputs,

Depth *L*

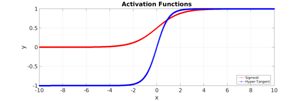
$\mathbf{h^1}$

$\mathbf{x}$

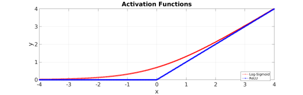


Typical choices of $\sigma(\cdot)$:

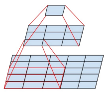
Sigmoid or Hyperbolic Tangent



Rectified Linear Unit (ReLU)

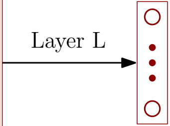


Convolution + Sub-sampling



$\sigma(\cdot)$: Nonlinear
Monotonic
Non-convex
Non-smooth

$\mathbf{W_{L-1}, h^{L-2}})$



$$\hat{\mathbf{y}} = \sigma_{\mathbf{L}}(\mathbf{W_L}, \mathbf{h^{L-1}})$$

Deep Learning – *Neural Networks*

Learning Objective: $$\min_{\mathbf{W}} \quad \mathbb{E}_{\mathbf{x},\mathbf{y}\sim\mathcal{X}}\mathcal{L}(\mathbf{x},\mathbf{y};\mathbf{W})$$

$$\mathbf{W} := \{\mathbf{W}_1 \ldots, \mathbf{W}_L\}$$

Deep Learning – *Neural Networks*

Non-convex

Learning Objective: $$\min_{\mathbf{W}} \quad \mathbb{E}_{\mathbf{x},\mathbf{y}\sim\mathcal{X}}\mathcal{L}(\mathbf{x},\mathbf{y};\mathbf{W})$$

$$\mathbf{W} := \{\mathbf{W}_1 \ldots, \mathbf{W}_L\}$$

Deep Learning – *Neural Networks*

Non-convex

Learning Objective: $\min_{\mathbf{W}} \quad \mathbb{E}_{\mathbf{x},\mathbf{y}\sim\mathcal{X}}\mathcal{L}(\mathbf{x},\mathbf{y};\mathbf{W})$

$$\mathbf{W} := \{\mathbf{W}_1 \ldots, \mathbf{W}_L\}$$

Stochastic Gradients are used

Gradient backpropagation

Deep Learning – *Neural Networks*

Stochastic Gradients are used ... with some tricks!

Deep Learning – *Neural Networks*

Stochastic Gradients are used ... with some tricks!

- *Appropriate* Nonlinearities

ReLU, Log-sigmoid, Max-pooling etc.

Deep Learning – *Neural Networks*

Stochastic Gradients are used ... with some tricks!

- *Appropriate* Nonlinearities

ReLU, Log-sigmoid, Max-pooling etc.

- Initializations

Pretrain (Warm-start) the network layers

$\rightarrow$ Using unlabeled data – Unsupervised Pretraining

Deep Learning – *Neural Networks*

Stochastic Gradients are used ... with some tricks!

• *Appropriate* Nonlinearities

ReLU, Log-sigmoid, Max-pooling etc.

• Initializations

Pretrain (Warm-start) the network layers

$\rightarrow$ Using unlabeled data – Unsupervised Pretraining

• Learning mechanisms

Stochastically learn parts of network

$\rightarrow$ Dropout, DropConnect

Deep Learning – *Neural Networks*

Stochastic Gradients are used ... with some tricks!

- *Appropriate* Nonlinearities

ReLU, Log-sigmoid, Max-pooling etc.

- Initializations

Pretrain (Warm-start) the network layers

→ Using unlabeled data – Unsupervised Pretraining

- Learning mechanisms

Stochastically learn parts of network

→ Dropout, DropConnect

- Large Dataset sizes

Deep Learning – *Neural Networks*

Attractive empirical success

Deep Learning – *Neural Networks*

Attractive empirical success

... some interesting theoretical results

Arora et. al. 2013, Dauphin et. al. 2014, Patel et. al. 2015

Deep Learning – *Neural Networks*

Attractive empirical success

... some interesting theoretical results

Arora et. al. 2013, Dauphin et. al. 2014, Patel et. al. 2015

Theme of most works

Deep Learning – *Neural Networks*

Attractive empirical success
... some interesting theoretical results
Arora et. al. 2013, Dauphin et. al. 2014, Patel et. al. 2015

Theme of most works

$\rightarrow$ Analyze a *given* architecture/structure
the depth L, hidden layer lengths $(d_1, \ldots, d_{L-1})$
hidden layer activations are known

Deep Learning – *Neural Networks*

Attractive empirical success

... some interesting theoretical results

Arora et. al. 2013, Dauphin et. al. 2014, Patel et. al. 2015

Theme of most works

$\rightarrow$ Analyze a *given* architecture/structure

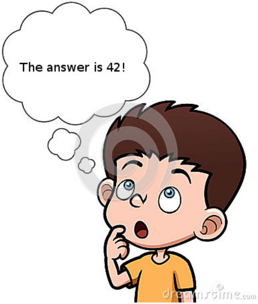
the depth L, hidden layer lengths $(d_1, \ldots, d_{L-1})$

hidden layer activations are known

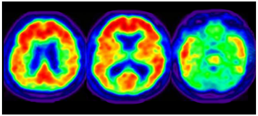
$\rightarrow$ Existence of some network structure is proven

The Problem

What is the best possible network for the given task?

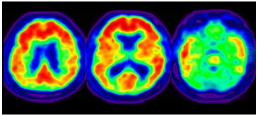


The Motivating Application



Amyloid PET Images
Collected from Middle-aged Adults

The Motivating Application



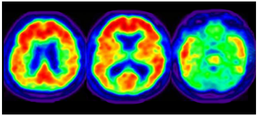
Amyloid PET Images
Collected from Middle-aged Adults

Deep Network
Predictor

→

The probability of disease in future

The Motivating Application



Amyloid PET Images
Collected from Middle-aged Adults

Deep Network Predictor

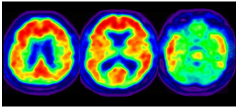
→

The probability of disease in future

Send to trial ✓

✗ **Do not send to trial**

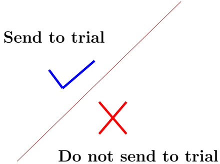
The Motivating Application



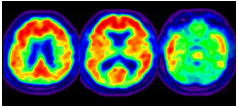
Amyloid PET Images
Collected from Middle-aged Adults

Deep Network Predictor →

The probability of disease in future



The Motivating Application



Amyloid PET Images
Collected from Middle-aged Adults

Deep Network Predictor

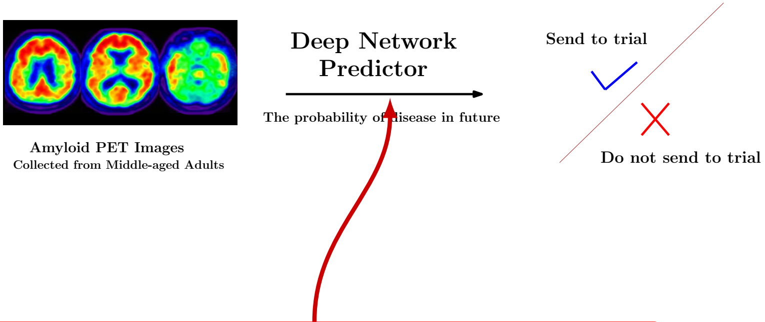
The probability of disease in future

Send to trial ✓

✗ Do not send to trial

Bottleneck on the available #instances
Brain image acquisition is costly!

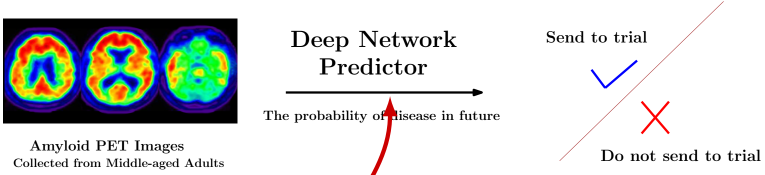
The Motivating Application



- Cheapest – #computations, $cost

Dollar value associated per hour of computation (e.g., using Amazon Web Services)

The Motivating Application

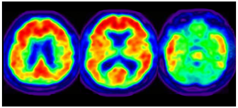


- Cheapest – #computations, $cost

Dollar value associated per hour of computation (e.g., using Amazon Web Services)

- Richer (Largest) models are desired

The Motivating Application



Amyloid PET Images
Collected from Middle-aged Adults

Deep Network Predictor

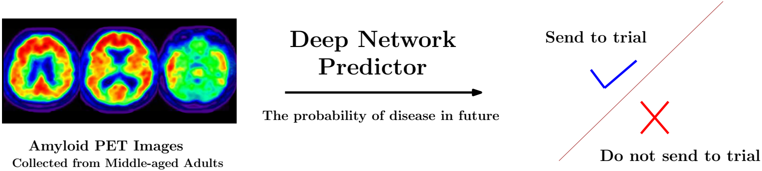
The probability of disease in future

Send to trial ✓

Do not send to trial ✗

Some false-positives allowed

The Motivating Application



A non-expert is going to setup the learning

The Problem – reformulated

We need informed or systematic design strategies for the choosing network structure

The Solution strategy

What is the best possible network for the given task?
Need informed design strategies

Part I

The Solution strategy

What is the best possible network for the given task?
Need informed design strategies

Part I

Construct the relevant bounds

- Gradient convergence + Learning Mechanism + Network/Data Statistics

The Solution strategy

What is the best possible network for the given task?
Need informed design strategies

Part I

Construct the relevant bounds

- Gradient convergence + Learning Mechanism + Network/Data Statistics

Part II

The Solution strategy

What is the best possible network for the given task?
Need informed design strategies

Part I

Construct the relevant bounds

- Gradient convergence + Learning Mechanism + Network/Data Statistics

Part II

Construct design procedures using the bounds

- For the given dataset, a *pre-specified* convergence level
  Find the depth, hidden layer lengths, etc.

The Solution strategy – This work

What is the best possible network for the given task?
Need informed design strategies

Part I

Construct the relevant bounds

- Gradient convergence + Learning Mechanism + Network/Data Statistics

Part II

Construct design procedures using the bounds

- For the given dataset, a *pre-specified* convergence level
  Find the depth, hidden layer lengths, etc.

The Interplay

Gradient convergence + Learning Mechanism + Network/Data Statistics

The Interplay

Gradient convergence + Learning Mechanism + Network/Data Statistics

$\rightarrow$ The depth parameter L

The Interplay

Gradient convergence + Learning Mechanism + Network/Data Statistics

→ The depth parameter L

→ The layer lengths $(d_0, d_1, \ldots, d_{L-1}, d_L)$

The Interplay

Gradient convergence + Learning Mechanism + Network/Data Statistics

→ The depth parameter L
→ The layer lengths $(d_0, d_1, \ldots, d_{L-1}, d_L)$
→ The activation functions $(\sigma_1, \ldots, \sigma_L)$

The Interplay

Gradient convergence + Learning Mechanism + Network/Data Statistics

→ The depth parameter L

→ The layer lengths $(d_0, d_1, \ldots, d_{L-1}, d_L)$

→ The activation functions $(\sigma_1, \ldots, \sigma_L)$

Bounded and Smooth; Focus on Sigmoid

The Interplay

Gradient convergence + Learning Mechanism + Network/Data Statistics

→ The depth parameter L
→ The layer lengths $(d_0, d_1, \ldots, d_{L-1}, d_L)$
→ The activation functions $(\sigma_1, \ldots, \sigma_L)$
Bounded and Smooth; Focus on Sigmoid
→ Average first-moment

The Interplay

Gradient convergence + Learning Mechanism + Network/Data Statistics

→ The depth parameter L
→ The layer lengths $(d_0, d_1, \ldots, d_{L-1}, d_L)$
→ The activation functions $(\sigma_1, \ldots, \sigma_L)$
Bounded and Smooth; Focus on Sigmoid
→ Average first-moment
$\mu_x = \frac{1}{d_0} \sum_j \mathbb{E}x_j$, $\tau_x = \frac{1}{d_0} \sum_j \mathbb{E}^2 x_j$

The Interplay

Gradient convergence + Learning Mechanism + Network/Data Statistics

$$\min_{\mathbf{W}} \quad f(\mathbf{W}) := \mathbb{E}_{\mathbf{x},\mathbf{y}\sim\mathcal{X}}\mathcal{L}(\mathbf{x},\mathbf{y};\mathbf{W})$$

The Interplay

Gradient convergence + Learning Mechanism + Network/Data Statistics

$$\min_{\mathbf{W}} \quad f(\mathbf{W}) := \mathbb{E}_{\mathbf{x},\mathbf{y}\sim\mathcal{X}}\mathcal{L}(\mathbf{x},\mathbf{y};\mathbf{W})$$

$\rightarrow \mathcal{L} := \ell_2$ Loss

The Interplay

Gradient convergence + Learning Mechanism + Network/Data Statistics

$$\min_{\mathbf{W}} \quad f(\mathbf{W}) := \mathbb{E}_{\mathbf{x},\mathbf{y}\sim\mathcal{X}}\mathcal{L}(\mathbf{x},\mathbf{y};\mathbf{W})$$

$\rightarrow \mathcal{L} := \ell_2$ Loss

Stochastic Gradients $\quad \mathbf{W} \in \mathbb{R}^d$

OR

Projected Gradients $\quad \mathbf{W} \in \Omega :=$ Box-constraint $[-w, w]^d$

The Interplay – Gradient Convergence

Ideally interested in generalization

The Interplay – Gradient Convergence

Ideally interested in generalization

Train faster, generalize better:
Stability of stochastic gradient descent

Moritz Hardt* Benjamin Recht† Yoram Singer‡

February 9, 2016

The Interplay – Gradient Convergence

Ideally interested in generalization

Train faster, generalize better:
Stability of stochastic gradient descent

Moritz Hardt* Benjamin Recht† Yoram Singer‡

February 9, 2016

Convergence instead?

R : Last iteration – *In general*, training time is fixed apriori

The Interplay – Gradient Convergence

Ideally interested in generalization

Train faster, generalize better:
Stability of stochastic gradient descent

Moritz Hardt* Benjamin Recht† Yoram Singer‡

February 9, 2016

Convergence instead?

R : Last iteration – *In general*, training time is fixed apriori

The expected gradients $\Delta := \mathbb{E}_{R,\mathbf{x},\mathbf{y}} \|\nabla_{\mathbf{W}} f(\mathbf{W}^R)\|^2$

The Interplay – Gradient Convergence

Ideally interested in generalization

Train faster, generalize better:
Stability of stochastic gradient descent

Moritz Hardt* Benjamin Recht† Yoram Singer‡

Control on last/stopping iteration

Convergence instead?

R : Last iteration – *In general*, training time is fixed apriori

The expected gradients $\Delta := \mathbb{E}_{R,\mathbf{x},\mathbf{y}} \|\nabla_{\mathbf{W}} f(\mathbf{W}^R)\|^2$

The Interplay – Gradient Convergence

Ideally interested in generalization

Train faster, generalize better:
Stability of stochastic gradient descent

Moritz Hardt* Benjamin Recht† Yoram Singer‡

Control on last/stopping iteration

Convergence instead?

R : Last iteration – *In general*, training time is fixed apriori

The expected gradients $\Delta := \mathbb{E}_{R,\mathbf{x},\mathbf{y}} \|\nabla_{\mathbf{W}} f(\mathbf{W}^R)\|^2$

Under mild assumptions, Δ can be bounded whenever R is chosen randomly [Ghadimi and Lan 2013]

The Interplay – Gradient Convergence

Gradients backpropagation

$+$

randomly stop after some iterations

The Interplay – Gradient Convergence

Single-layer Network

The Interplay – Gradient Convergence

Single-layer Network

Expected Gradients

For 1-layer network with stepsizes $\gamma^k = \frac{\gamma}{k^\rho}$ $(\rho > 0)$ and $P_R(k) = \gamma^k(1 - 0.75\gamma^k)$, we have

$$\Delta \leq \left(\frac{D_f}{\mathcal{H}_N} + \Psi \right)$$

The Interplay – Gradient Convergence

Decreasing stepsizes

Single-layer Network

Expected Gradients

For 1-layer network with stepsizes $\gamma^k = \frac{\gamma}{k^\rho}$ $(\rho > 0)$ and $P_R(k) = \gamma^k(1 - 0.75\gamma^k)$, we have

$$\Delta \leq \left(\frac{D_f}{\mathcal{H}_N} + \Psi \right)$$

The Interplay – Gradient Convergence

Decreasing stepsizes

Single-layer Network

Expected Gradients

For 1-layer network with stepsizes $\gamma^k = \frac{\gamma}{k^\rho}$ $(\rho > 0)$ and $P_R(k) = \gamma^k(1 - 0.75\gamma^k)$, we have

$$\Delta \leq \left(\frac{D_f}{\mathcal{H}_N} + \Psi\right)$$

N: Maximum allowable iterations

the stopping distribution $R \in [1, N]$ $(N \gg R)$

The Interplay – Gradient Convergence

Decreasing stepsizes

Single-layer Network

Expected Gradients

For 1-layer network with stepsizes $\gamma^k = \frac{\gamma}{k^\rho}$ $(\rho > 0)$ and $P_R(k) = \gamma^k(1 - 0.75\gamma^k)$, we have

$$\Delta \leq \left(\frac{D_f}{\mathcal{H}_N} + \Psi \right)$$

the stopping distribution $R \in [1, N]$ $(N \gg R)$

N: Maximum allowable iterations

Δ : Expected gradients

The Interplay – Gradient Convergence

Single-layer Network

Expected Gradients

For 1-layer network with stepsizes $\gamma^k = \frac{\gamma}{k^\rho}$ $(\rho > 0)$ and $P_R(k) = \gamma^k(1 - 0.75\gamma^k)$, we have

$$\Delta \leq \left(\frac{D_f}{\mathcal{H}_N} + \Psi \right)$$

The Interplay – Gradient Convergence

Single-layer Network

Expected Gradients

For 1-layer network with stepsizes $\gamma^k = \frac{\gamma}{k^\rho}$ $(\rho > 0)$ and $P_R(k) = \gamma^k(1 - 0.75\gamma^k)$, we have

$$\Delta \leq \left(\frac{D_f}{\mathcal{H}_N} + \Psi \right)$$

$D_f \approx f(\mathbf{W}^1)$

The Interplay – Gradient Convergence

Single-layer Network

Expected Gradients

For 1-layer network with stepsizes $\gamma^k = \frac{\gamma}{k^\rho}$ $(\rho > 0)$ and $P_R(k) = \gamma^k(1 - 0.75\gamma^k)$, we have

$$\Delta \leq \left(\frac{D_f}{\mathcal{H}_N} + \Psi \right)$$

$D_f \approx f(\mathbf{W}^1)$

$\mathcal{H}_N \approx 0.2\gamma GenHar(N, \rho)$
N : Maximum allowable iterations

The Interplay – Gradient Convergence

Single-layer Network

Expected Gradients

For 1-layer network with stepsizes $\gamma^k = \frac{\gamma}{k^\rho}$ $(\rho > 0)$ and $P_R(k) = \gamma^k(1 - 0.75\gamma^k)$, we have

$$\Delta \leq \left(\frac{D_f}{\mathcal{H}_N} + \Psi \right)$$

$D_f \approx f(\mathbf{W}^1)$

$\mathcal{H}_N \approx 0.2\gamma GenHar(N, \rho)$
N : Maximum allowable iterations

Goodness of fit – Influence of $\mathbf{W}^1$

The Interplay – Gradient Convergence

Single-layer Network

Expected Gradients

For 1-layer network with stepsizes $\gamma^k = \frac{\gamma}{k^\rho}$ $(\rho > 0)$ and $P_R(k) = \gamma^k(1 - 0.75\gamma^k)$, we have

$$\Delta \leq \left(\frac{D_f}{\mathcal{H}_N} + \Psi \right)$$

$D_f \approx f(\mathbf{W}^1)$

$\mathcal{H}_N \approx 0.2\gamma GenHar(N, \rho)$
N : Maximum allowable iterations

Goodness of fit – Influence of $\mathbf{W}^1$
- Sublinear decay vs. N

The Interplay – Gradient Convergence

Single-layer Network

Expected Gradients

For 1-layer network with stepsizes $\gamma^k = \frac{\gamma}{k^\rho}$ $(\rho > 0)$ and $P_R(k) = \gamma^k(1 - 0.75\gamma^k)$, we have

$$\Delta \leq \left(\frac{D_f}{\mathcal{H}_N} + \Psi \right)$$

$\Psi \approx q\frac{d_0 d_1 \gamma}{B}$
$(0.05 < q < 0.25)$
$d_0 d_1 := \#\text{unknowns}$

The Interplay – Gradient Convergence

Single-layer Network

Expected Gradients

For 1-layer network with stepsizes $\gamma^k = \frac{\gamma}{k^\rho}$ $(\rho > 0)$ and $P_R(k) = \gamma^k(1 - 0.75\gamma^k)$, we have

$$\Delta \leq \left(\frac{D_f}{\mathcal{H}_N} + \Psi \right)$$

$\Psi \approx q\frac{d_0 d_1 \gamma}{B}$
$(0.05 < q < 0.25)$
$d_0 d_1 :=$ #unknowns

Influence of #free parameters (degrees of freedom)

The Interplay – Gradient Convergence

Single-layer Network

Expected Gradients

For 1-layer network with stepsizes $\gamma^k = \frac{\gamma}{k^\rho}$ $(\rho > 0)$ and $P_R(k) = \gamma^k(1 - 0.75\gamma^k)$, we have

$$\Delta \leq \left(\frac{D_f}{\mathcal{H}_N} + \Psi\right)$$

$\Psi \approx q\frac{d_0 d_1 \gamma}{B}$
$(0.05 < q < 0.25)$
$d_0 d_1 :=$ #unknowns

Influence of #free parameters (degrees of freedom)
Bias from mini-batch size

The Interplay – Gradient Convergence

Single-layer Network

Expected Gradients

For 1-layer network with stepsizes $\gamma^k = \frac{\gamma}{k^\rho}$ $(\rho > 0)$ and $P_R(k) = \gamma^k(1 - 0.75\gamma^k)$, we have

$$\Delta \leq \left(\frac{D_f}{\mathcal{H}_N} + \Psi \right)$$

- **Ideal scenario:** Large #samples; Small network

The Interplay – Gradient Convergence

Single-layer Network

Expected Gradients

For 1-layer network with stepsizes $\gamma^k = \frac{\gamma}{k^\rho}$ $(\rho > 0)$ and $P_R(k) = \gamma^k(1 - 0.75\gamma^k)$, we have

$$\Delta \leq \left(\frac{D_f}{\mathcal{H}_N} + \Psi \right)$$

- Ideal scenario: Large #samples; Small network
- Realistic scenario:
  Reasonable network size; Large B with long training time

The Interplay – Gradient Convergence

for small ρ i.e, slow stepsize decay

$P_R(k)$ approaches a uniform distribution

The Interplay – Gradient Convergence

for small ρ i.e, slow stepsize decay

$P_R(k)$ approaches a uniform distribution

$$\Delta \lesssim \left(\frac{5D_f}{N\gamma} + \Psi \right)$$

The Interplay – Gradient Convergence

for small ρ i.e, slow stepsize decay

$P_R(k)$ approaches a uniform distribution

$$\Delta \lesssim \left(\frac{5D_f}{N\gamma} + \Psi \right)$$

when $\rho = 0$ i.e., constant stepsize

$P_R(k) := UNIF[1, N]$

The Interplay – Gradient Convergence

for small ρ i.e, slow stepsize decay

$P_R(k)$ approaches a uniform distribution

$$\Delta \lesssim \left(\frac{5D_f}{N\gamma} + \Psi \right)$$

when $\rho = 0$ i.e., constant stepsize

$P_R(k) := UNIF[1, N]$

$$\Delta \leq \left(\frac{D_f}{N\gamma} + \Psi \right)$$

The Interplay – Gradient Convergence

for small ρ i.e, slow stepsize decay

$P_R(k)$ approaches a uniform distribution

$$\Delta \lesssim \left(\frac{5 D_f}{N\gamma} + \Psi \right)$$

when $\rho = 0$ i.e., constant stepsize

$P_R(k) := UNIF[1, N]$

Uniform stopping may not be interesting!

$$\Delta \leq \left(\frac{D_f}{N\gamma} + \Psi \right)$$

The Interplay – Gradient Convergence

Single-layer Network + *Customized* $P_R(k)$

The Interplay – Gradient Convergence

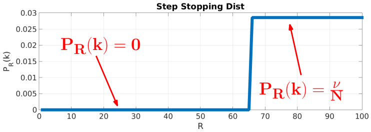
Single-layer Network + *Customized* $P_R(k)$

Push R to be as close as possible to N

The Interplay – Gradient Convergence

Single-layer Network + *Customized* $P_R(k)$

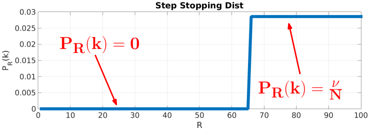
Push R to be as close as possible to N



The Interplay – Gradient Convergence

Single-layer Network + *Customized* $P_R(k)$

Push R to be as close as possible to N



Expected Gradients + $P_R(\cdot)$ *from above example*

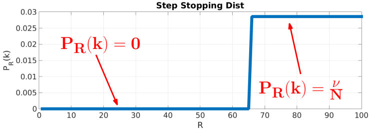
For 1-layer network with constant stepsize γ, we have

$$\Delta \leq \nu \left(\frac{5D_f}{N\gamma} + \Psi \right)$$

The Interplay – Gradient Convergence

Single-layer Network + *Customized* $P_R(k)$

Push R to be as close as possible to N



Expected Gradients + $P_R(\cdot)$ *from above example*

For 1-layer network with constant stepsize γ, we have

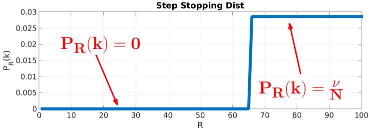
$$\Delta \leq \nu \left(\frac{5D_f}{N\gamma} + \Psi \right)$$

require $P_R(k) \leq P_R(k+1)$

The Interplay – Gradient Convergence

Single-layer Network + *Customized* $P_R(k)$

Push R to be as close as possible to N



For $\nu \gg 1$, $R \to N$

Expected Gradients + $P_R(\cdot)$ *from above example*

For 1-layer network with constant stepsize γ, we have

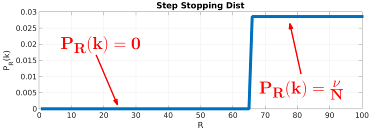
$$\Delta \leq \nu \left(\frac{5D_f}{N\gamma} + \Psi \right)$$

require $P_R(k) \leq P_R(k+1)$

The Interplay – Gradient Convergence

Single-layer Network + *Customized* $P_R(k)$

Push R to be as close as possible to N



For $\nu \gg 1$, $R \to N$
bound too loose

Expected Gradients + $P_R(\cdot)$ *from above example*

For 1-layer network with constant stepsize γ, we have

$$\Delta \leq \nu \left(\frac{5D_f}{N\gamma} + \Psi \right)$$

require $P_R(k) \leq P_R(k+1)$

The Interplay – Gradient Convergence

Single-layer Network

Using T independent random stopping iterations

The Interplay – Gradient Convergence

Single-layer Network

Using T independent random stopping iterations

Large deviation estimate

The Interplay – Gradient Convergence

Single-layer Network

Using T independent random stopping iterations

Large deviation estimate

Let $\epsilon > 0$ and $0 < \delta \ll 1$.
An (ϵ, δ)-solution guarantees $Pr\left(\min_t \|\nabla_{\mathbf{W}} f(\mathbf{W}^{R_t})\|^2 \leq \epsilon\right) \geq 1 - \delta$

The Interplay

Gradient convergence + Learning Mechanism + Network/Data Statistics

The Interplay

Gradient convergence + Learning Mechanism + Network/Data Statistics

Multi-layer Neural Network

$L - 1$ single-layer networks *put together*

The Interplay

Gradient convergence + Learning Mechanism + Network/Data Statistics

Multi-layer Neural Network

$L - 1$ single-layer networks *put together*

Typical mechanism

The Interplay

Gradient convergence + Learning Mechanism + Network/Data Statistics

Multi-layer Neural Network

$L - 1$ single-layer networks *put together*

Typical mechanism

- Initialize (or Warm-start or Pretrain) each of the layers *sequentially*

The Interplay

Gradient convergence + Learning Mechanism + Network/Data Statistics

Multi-layer Neural Network

$L - 1$ single-layer networks *put together*

Typical mechanism

- Initialize (or Warm-start or Pretrain) each of the layers *sequentially*
  $\mathbf{x} \to \tilde{\mathbf{x}}$ (w.p. $1 - \zeta$, the j^{th} unit is 0)

The Interplay

Gradient convergence + Learning Mechanism + Network/Data Statistics

Multi-layer Neural Network

$L-1$ single-layer networks *put together*

Typical mechanism

- Initialize (or Warm-start or Pretrain) each of the layers *sequentially*

  $\mathbf{x} \rightarrow \tilde{\mathbf{x}}$ (w.p. $1-\zeta$, the j^{th} unit is 0)

  $\mathbf{h}^1 = \sigma(\mathbf{W}^1\tilde{\mathbf{x}}) \qquad \mathcal{L}(\mathbf{x}, \mathbf{W}) = \|\mathbf{x} - \mathbf{h}^1\|^2 \quad \text{with} \quad \mathbf{W} \in [-w, w]^d$

  Referred to as a Denoising Autoencoder

The Interplay

Gradient convergence + Learning Mechanism + Network/Data Statistics

Multi-layer Neural Network

$L-1$ single-layer networks *put together*

Typical mechanism

- Initialize (or Warm-start or Pretrain) each of the layers *sequentially*
  $\mathbf{x} \rightarrow \tilde{\mathbf{x}}$ (w.p. $1-\zeta$, the j^{th} unit is 0)
  $\mathbf{h}^1 = \sigma(\mathbf{W}^1\tilde{\mathbf{x}}) \qquad \mathcal{L}(\mathbf{x}, \mathbf{W}) = \|\mathbf{x} - \mathbf{h}^1\|^2 \quad \text{with} \quad \mathbf{W} \in [-w, w]^d$
  Referred to as a Denoising Autoencoder
- $L-1$ such DAs are learned
  $\mathbf{x} \rightarrow \mathbf{h}^1 \rightarrow \ldots \mathbf{h}^{L-2} \rightarrow \mathbf{h}^{L-1}$

The Interplay

Gradient convergence + **Learning Mechanism** + Network/Data Statistics

Multi-layer Neural Network

$L - 1$ single-layer networks *put together*

The Interplay

Gradient convergence + **Learning Mechanism** + Network/Data Statistics

Multi-layer Neural Network

$L-1$ single-layer networks *put together*

Typical mechanism

- Bring in the **y**s; perform backpropagation

The Interplay

Gradient convergence + Learning Mechanism + Network/Data Statistics

Multi-layer Neural Network

$L - 1$ single-layer networks *put together*

Typical mechanism

- Bring in the **y**s; perform backpropagation
  Use stochastic gradients; start at L^{th}-layer
  Propagate the gradients

The Interplay

Gradient convergence + Learning Mechanism + Network/Data Statistics

Multi-layer Neural Network

$L - 1$ single-layer networks *put together*

Typical mechanism

- Bring in the **y**s; perform backpropagation
  Use stochastic gradients; start at L^{th}-layer
  Propagate the gradients

→ Dropout
  Update only a fraction (ζ) of all the parameters

The Interplay – Learning Mechanism

Multi-layer Neural Network

The Interplay – Learning Mechanism

Multi-layer Neural Network

The new mechanism – Randomized stopping strategy at all stages

The Interplay – Learning Mechanism

Multi-layer Neural Network

The new mechanism – Randomized stopping strategy at all stages

- $L - 1$ layers are initialized to (α, δ_α) solutions

  α : Goodness of pretraning

The Interplay – Learning Mechanism

Multi-layer Neural Network

The new mechanism – Randomized stopping strategy at all stages

- $L - 1$ layers are initialized to (α, δ_α) solutions
  α : Goodness of pretraning
- Gradient backpropagation is performed to a (ϵ, δ) solution

The Interplay – The most general result

Multi-layer Neural Network

For L-layered network with dropout rate ζ and constant stepsize γ, pretrained to (α, δ_α), we have

$$\Delta \leq \left(\frac{D_f}{Ne} + \Pi \right)$$

The Interplay – The most general result

Multi-layer Neural Network

For L-layered network with dropout rate ζ and constant stepsize γ, pretrained to (α, δ_α), we have

$$\Delta \leq \left(\frac{D_f}{Ne} + \Pi\right)$$

First known result for multi-layer deep networks

The Interplay – The most general result

Multi-layer Neural Network

For L-layered network with dropout rate ζ and constant stepsize γ, pretrained to (α, δ_α), we have

$$\Delta \leq \left(\frac{D_f}{Ne} + \Pi \right)$$

First known result for multi-layer deep networks
Unsupervised pretraining

The Interplay – The most general result

Multi-layer Neural Network

For L-layered network with dropout rate ζ and constant stepsize γ, pretrained to (α, δ_α), we have

$$\Delta \leq \left(\frac{D_f}{Ne} + \Pi \right)$$

First known result for multi-layer deep networks
Unsupervised pretraining + Dropout learning

The Interplay – The most general result

Multi-layer Neural Network

For L-layered network with dropout rate ζ and constant stepsize γ, pretrained to (α, δ_α), we have

$$\Delta \leq \left(\frac{D_f}{Ne} + \Pi \right)$$

First known result for multi-layer deep networks
Unsupervised pretraining $+$ Dropout learning $+$ Network structure

The Interplay – The most general result

Multi-layer Neural Network

For L-layered network with dropout rate ζ and constant stepsize γ, pretrained to (α, δ_α), we have

$$\Delta \leq \left(\frac{D_f}{Ne} + \Pi \right)$$

First known result for multi-layer deep networks
Unsupervised pretraining + Dropout learning + Network structure
.... to convergence and estimation

The Interplay – The most general result

Multi-layer Neural Network

For L-layered network with dropout rate ζ and constant stepsize γ, pretrained to (α, δ_α), we have

$$\Delta \leq \left(\frac{D_f}{Ne} + \Pi \right)$$

Δ : Expected *projected* gradients

The Interplay – The most general result

Multi-layer Neural Network

For L-layered network with dropout rate ζ and constant stepsize γ, pretrained to (α, δ_α), we have

$$\Delta \leq \left(\frac{D_f}{Ne} + \Pi \right)$$

The Interplay – The most general result

Multi-layer Neural Network

For L-layered network with dropout rate ζ and constant stepsize γ, pretrained to (α, δ_α), we have

$$\Delta \leq \left(\frac{D_f}{Ne} + \Pi \right)$$

$D_f \approx f(\mathbf{W}^1)$ (after pretraining)

The Interplay – The most general result

Multi-layer Neural Network

For L-layered network with dropout rate ζ and constant stepsize γ, pretrained to (α, δ_α), we have

$$\Delta \leq \left(\frac{D_f}{Ne} + \Pi \right)$$

$D_f \approx f(\mathbf{W}^1)$ (after pretraining)
N: Backpropagation iterations

The Interplay – The most general result

Multi-layer Neural Network

For L-layered network with dropout rate ζ and constant stepsize γ, pretrained to (α, δ_α), we have

$$\Delta \leq \left(\frac{D_f}{Ne} + \Pi \right)$$

$D_f \approx f(\mathbf{W}^1)$ (after pretraining)
N: Backpropagation iterations

$e := \zeta^2 g(\alpha, \gamma, w)$
Encodes the influence of pretraining, stepsize and box-constraint

The Interplay – The most general result

Multi-layer Neural Network

For L-layered network with dropout rate ζ and constant stepsize γ, pretrained to (α, δ_α), we have

$$\Delta \leq \left(\frac{D_f}{Ne} + \Pi \right)$$

- Usefulness of the representations

i.e., is $\mathbf{h}_{L-1}$ already good-enough in predicting $\mathbf{y}$

The Interplay – The most general result

Multi-layer Neural Network

For L-layered network with dropout rate ζ and constant stepsize γ, pretrained to (α, δ_α), we have

$$\Delta \leq \left(\frac{D_f}{Ne} + \Pi \right)$$

- Usefulness of the representations

i.e., is $\mathbf{h}_{L-1}$ already good-enough in predicting $\mathbf{y}$

- Noise added by dropout

The Interplay – The most general result

Multi-layer Neural Network

For L-layered network with dropout rate ζ and constant stepsize γ, pretrained to (α, δ_α), we have

$$\Delta \leq \left(\frac{D_f}{Ne} + \Pi \right)$$

The Interplay – The most general result

Multi-layer Neural Network

For L-layered network with dropout rate ζ and constant stepsize γ, pretrained to (α, δ_α), we have

$$\Delta \leq \left(\frac{D_f}{Ne} + \Pi \right)$$

$\Pi : \Pi(\alpha, \zeta, \gamma, B, w, \#\text{freedom})$

The Interplay – The most general result

Multi-layer Neural Network

For L-layered network with dropout rate ζ and constant stepsize γ, pretrained to (α, δ_α), we have

$$\Delta \leq \left(\frac{D_f}{Ne} + \Pi \right)$$

$\Pi : \Pi(\alpha, \zeta, \gamma, B, w, \#\text{freedom})$

Polynomial in $d_0, \ldots, d_L$ and L

The Interplay – The most general result

Multi-layer Neural Network

For L-layered network with dropout rate ζ and constant stepsize γ, pretrained to (α, δ_α), we have

$$\Delta \leq \left(\frac{D_f}{Ne} + \Pi \right)$$

$\Pi : \Pi(\alpha, \zeta, \gamma, B, w, \#\text{freedom})$

Polynomial in $d_0, \ldots, d_L$ and L

Linear in α, Polynomial in ζ

The Interplay – The most general result

Multi-layer Neural Network

For L-layered network with dropout rate ζ and constant stepsize γ, pretrained to (α, δ_α), we have

$$\Delta \leq \left(\frac{D_f}{Ne} + \Pi \right)$$

$\Pi : \Pi(\alpha, \zeta, \gamma, B, w, \#\text{freedom})$

- Polynomial in $d_0, \ldots, d_L$ and L
- Linear in α, Polynomial in ζ

Complex interplay of
- Learning modules &
- Network hyper-parameters

The Interplay – Some Implications

Multi-layer Neural Network

$$\Delta \leq \left(\frac{D_f}{Ne} + \Pi \right)$$

Interesting trends/outcomes (First theoretical results)

The Interplay – Some Implications

Multi-layer Neural Network

$$\Delta \leq \left(\frac{D_f}{Ne} + \Pi \right)$$

Interesting trends/outcomes (First theoretical results)

→ Dropout Compensates Pretraining

The Interplay – Some Implications

Multi-layer Neural Network

$$\Delta \leq \left(\frac{D_f}{Ne} + \Pi \right)$$

Interesting trends/outcomes (First theoretical results)

$\rightarrow$ Dropout Compensates Pretraining

Small $\alpha \implies \zeta \sim 1$ (Faster convergence)

The Interplay – Some Implications

Multi-layer Neural Network

$$\Delta \leq \left(\frac{D_f}{Ne} + \Pi \right)$$

Interesting trends/outcomes (First theoretical results)

$\rightarrow$ Dropout Compensates Pretraining

Small $\alpha \implies \zeta \sim 1$ (Faster convergence)

Large $\alpha \implies \zeta \sim 0$ (Slower convergence)

The Interplay – Some Implications

Multi-layer Neural Network

$$\Delta \leq \left(\frac{D_f}{Ne} + \Pi \right)$$

Interesting trends/outcomes (First theoretical results)

$\rightarrow$ Dropout Compensates Pretraining

Small $\alpha \implies \zeta \sim 1$ (Faster convergence)

Large $\alpha \implies \zeta \sim 0$ (Slower convergence)

No control on $\alpha \implies$ Set ζ to 0.5

The Interplay – Some Implications

Multi-layer Neural Network

$$\Delta \leq \left(\frac{D_f}{Ne} + \Pi \right)$$

Interesting trends/outcomes (First theoretical results)

→ Dropout Compensates Pretraining

Small $\alpha \implies \zeta \sim 1$ (Faster convergence)
Large $\alpha \implies \zeta \sim 0$ (Slower convergence)
No control on $\alpha \implies$ Set ζ to 0.5

Pretraining can be bypassed for small networks

The Interplay – Some Implications

Multi-layer Neural Network

$$\Delta \leq \left(\frac{D_f}{Ne} + \Pi \right)$$

Interesting trends/outcomes (First theoretical results)

→ Dropout Compensates Pretraining

Small $\alpha \implies \zeta \sim 1$ (Faster convergence)
Large $\alpha \implies \zeta \sim 0$ (Slower convergence)
No control on $\alpha \implies$ Set ζ to 0.5

Pretraining can be bypassed for small networks
Everything breaks loose for large networks

The Interplay – Some Implications

Multi-layer Neural Network

$$\Delta \leq \left(\frac{D_f}{Ne} + \Pi \right)$$

Interesting trends/outcomes (First theoretical results)

→ Dropout Compensates Pretraining

- Small $\alpha \implies \zeta \sim 1$ (Faster convergence)
- Large $\alpha \implies \zeta \sim 0$ (Slower convergence)
- No control on $\alpha \implies$ Set ζ to 0.5

Pretraining can be bypassed for small networks

Everything breaks loose for large networks

- Only restoration is very large datasets and N

The Interplay – Some Implications

Multi-layer Neural Network

$$\Delta \leq \left(\frac{D_f}{Ne} + \Pi \right)$$

Interesting trends/outcomes (First theoretical results)

The Interplay – Some Implications

Multi-layer Neural Network

$$\Delta \leq \left(\frac{D_f}{Ne} + \Pi \right)$$

Interesting trends/outcomes (First theoretical results)

A *tall-lean* network is equivalent to *short-fat* one

The Interplay – Some Implications

Multi-layer Neural Network

$$\Delta \leq \left(\frac{D_f}{Ne} + \Pi \right)$$

Interesting trends/outcomes (First theoretical results)

A *tall-lean* network is equivalent to *short-fat* one

Depth hurts – but may be not too much

The Interplay – Some Implications

Multi-layer Neural Network

$$\Delta \leq \left(\frac{D_f}{Ne} + \Pi \right)$$

Interesting trends/outcomes (First theoretical results)

A *tall-lean* network is equivalent to *short-fat* one

Depth hurts – but may be not too much

Short-fat network asks for large sample size

The Interplay – Some Implications

Multi-layer Neural Network

$$\Delta \leq \left(\frac{D_f}{Ne} + \Pi \right)$$

Interesting trends/outcomes (First theoretical results)

A *tall-lean* network is equivalent to *short-fat* one

Depth hurts – but may be not too much

Short-fat network asks for large sample size

Small networks on small samples may be a bad combination

The Interplay – Some Implications

Multi-layer Neural Network

$$\Delta \leq \left(\frac{D_f}{Ne} + \Pi \right)$$

Interesting trends/outcomes (First theoretical results)

A *tall-lean* network is equivalent to *short-fat* one

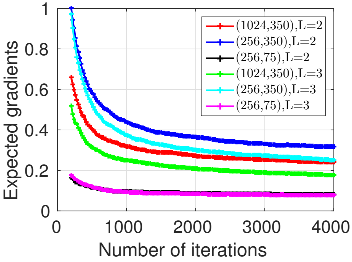
Depth hurts – but may be not too much

Short-fat network asks for large sample size

Small networks on small samples may be a bad combination

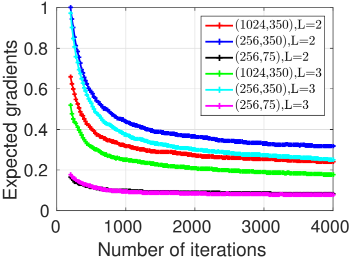
Family of networks that guarantee the same convergence

The Interplay – Experiments

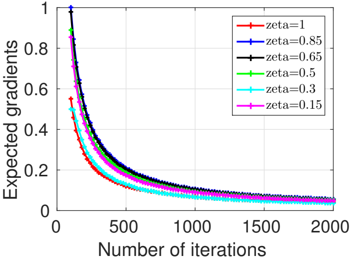


$\hat{\Delta}$ vs. L, d_ls

The Interplay – Experiments

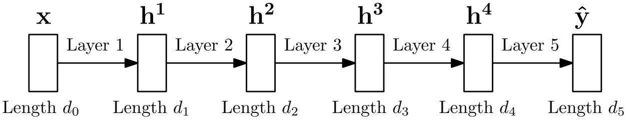


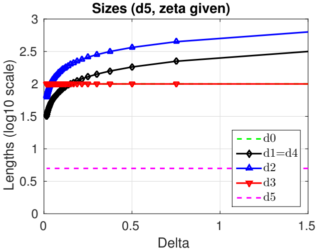
$\hat{\Delta}$ vs. L, d_ls



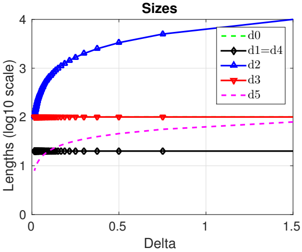
$\hat{\Delta}$ vs. ζ

The Interplay – Experiments





Designs given d_5, ζ and L



Designs given L

Conclusions & Ongoing Work

Conclusions

Gradient Convergence + Learning Mechanisms + Network/Data structure
$\rightarrow$ Small tweaks to existing procedures
$\rightarrow$ Theoretical understanding for many existing empirical studies
$\rightarrow$ New trends/outcomes

Conclusions & Ongoing Work

Conclusions

Gradient Convergence + Learning Mechanisms + Network/Data structure
$\rightarrow$ Small tweaks to existing procedures
$\rightarrow$ Theoretical understanding for many existing empirical studies
$\rightarrow$ New trends/outcomes

Ongoing Work

$\rightarrow$ Extensions to non-smooth $\sigma_l(\cdot)$s and complex $\Omega(\mathbf{W})$

$\rightarrow$ Part II

Find the best network for the given task

The end...

Thank you!
Questions?

NIH AG040396, NSF CAREER 1252725, NSF CCF 1320755, the UW grants ADRC AG033514, ICTR 1UL1RR025011 and CPCP AI117924