Hyper-parameters/Tweaking

Yufeng Ma, Chris Dusold

Virginia Tech

November 17, 2015

Overview

1. Batch Normalization
   - Internal Covariate Shift
   - Mini-Batch Normalization
   - Key Points in Batch Normalization
   - Experiments and Results

2. Importance of Initialization and Momentum
   - Overview of first-order method
   - Momentum & Nesterov's Accelerated Gradient(NAG)
   - Deep Autoencoders & RNN - Echo-State Networks

Challenges to be solved

Challenges to be solved

Reference paper: Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift

Challenges to be solved

Reference paper: Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift

When we are faced with training a Deep Network with saturating nonlinearities:

- Lower/smaller learning rates
- Initialize the weights from Gaussian Distributions

Challenges to be solved

Reference paper: Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift

When we are faced with training a Deep Network with saturating nonlinearities:

- Lower/smaller learning rates
- Initialize the weights from Gaussian Distributions

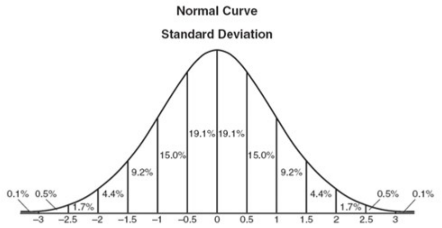


figure credit: www.regentsprep.org

Challenges to be solved

Challenges to be solved

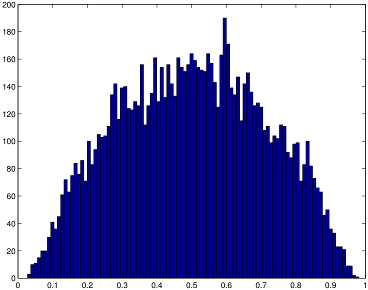
Reasons behind the problem:

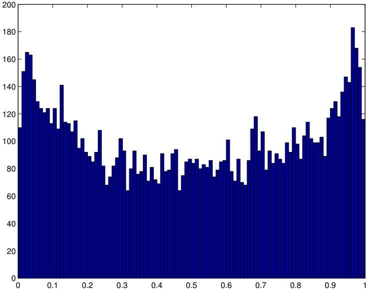
- Parameters change during training
- Input distributions of each layer changes

Challenges to be solved

Reasons behind the problem:

- Parameters change during training
- Input distributions of each layer changes





Sigmoid's output distribution before and after parameter updates

Overview

1. Batch Normalization
   - Internal Covariate Shift
   - Mini-Batch Normalization
   - Key Points in Batch Normalization
   - Experiments and Results
2. Importance of Initialization and Momentum
   - Overview of first-order method
   - Momentum & Nesterov's Accelerated Gradient(NAG)
   - Deep Autoencoders & RNN - Echo-State Networks

Internal Covariate Shift

Internal Covariate Shift

Covariate Shift

Change of input distributions to a Learning System

Internal Covariate Shift

Covariate Shift

Change of input distributions to a Learning System

Extension to part or sub-networks

$$\ell = F_2(F_1(u, \Theta_1), \Theta_2)$$

Internal Covariate Shift

Covariate Shift

Change of input distributions to a Learning System

Extension to part or sub-networks

$$\ell = F_2(F_1(u, \Theta_1), \Theta_2)$$

$$\ell = F_2(x, \Theta_2), \;\; where \; x = F_1(u, \Theta_1)$$

$$\Theta_2 \leftarrow \Theta_2 - \frac{\alpha}{m} \sum_{i=1}^{m} \frac{\partial F_2(x_i, \Theta_2)}{\partial \Theta_2}$$

Internal Covariate Shift

Covariate Shift

Change of input distributions to a Learning System

Extension to part or sub-networks

$$\ell = F_2(F_1(u, \Theta_1), \Theta_2)$$

$$\ell = F_2(x, \Theta_2), \ \ where \ x = F_1(u, \Theta_1)$$

$$\Theta_2 \leftarrow \Theta_2 - \frac{\alpha}{m} \sum_{i=1}^{m} \frac{\partial F_2(x_i, \Theta_2)}{\partial \Theta_2}$$

In terms of change in the distribution of x, Θ_2 will not need to readjust much.

Internal Covariate Shift

Change in the distributions of internal nodes of a deep network

Reducing Internal Covariate Shift

Reducing Internal Covariate Shift

Whitening-LeCun et al., 1998b; Wiesler&Ney, 2011

The network training converges faster if its inputs are whitened-i.e., linearly transformed to have zero means and unit variances, and decorrelated.

Reducing Internal Covariate Shift

Whitening-LeCun et al., 1998b; Wiesler&Ney, 2011

The network training converges faster if its inputs are whitened-i.e., linearly transformed to have zero means and unit variances, and decorrelated.

Goal: Whitening the inputs of each layer to have fixed distributions in order to Reduce the ill effects of Internal Covariate Shift.

Reducing Internal Covariate Shift

Reducing Internal Covariate Shift

- Interspersal lead to reduced gradient descent

$$b \leftarrow b + \Delta b, \ \textit{where } \Delta b \propto -\frac{\partial \ell}{\partial \hat{x}} \overbrace{\cancel{\frac{\partial \hat{x}}{\partial b}}}^{\textit{Ignored}}$$

$$\hat{x} = x - E[x] = u + (b + \Delta b) - E[u + (b + \Delta b)] = u + b - E[u + b]$$

Reducing Internal Covariate Shift

- Interspersal lead to reduced gradient descent

$$b \leftarrow b + \Delta b, \;\; where \;\; \Delta b \propto -\frac{\partial \ell}{\partial \hat{x}} \underbrace{\cancel{\frac{\partial \hat{x}}{\partial b}}}_{Ignored}$$

$$\hat{x} = x - E[x] = u + (b + \Delta b) - E[u + (b + \Delta b)] = u + b - E[u + b]$$

- Normalizations are NOT taken into account in Gradient Descent Optimization.

Reducing Internal Covariate Shift

Reducing Internal Covariate Shift

Introducing Normalization

$$\hat{x} = Norm(x, \mathcal{X})$$

and Jacobians in backpropagation

$$\frac{\partial Norm(x, \mathcal{X})}{\partial x} \quad and \quad \frac{\partial Norm(x, \mathcal{X})}{\partial \mathcal{X}}$$

Reducing Internal Covariate Shift

Introducing Normalization

$$\hat{x} = Norm(x, \mathcal{X})$$

and Jacobians in backpropagation

$$\frac{\partial Norm(x, \mathcal{X})}{\partial x} \; and \; \frac{\partial Norm(x, \mathcal{X})}{\partial \mathcal{X}}$$

New challenges: expensive to compute covariance matrix and its inverse square root.

Covariance matrix

$$Cov[x] = E_{x \in \mathcal{X}}[xx^T] - E[x]E[x]^T$$

Whitening

$$Cov[x]^{-1/2}(x - E[x])$$

Overview

1. Batch Normalization
   - Internal Covariate Shift
   - Mini-Batch Normalization
   - Key Points in Batch Normalization
   - Experiments and Results
2. Importance of Initialization and Momentum
   - Overview of first-order method
   - Momentum & Nesterov's Accelerated Gradient(NAG)
   - Deep Autoencoders & RNN - Echo-State Networks

Mini-Batch Normalization

Mini-Batch Normalization

Two simplifications and Identity Transform

- Normalize each scalar feature independently
- Use mini-batch to estimate the mean and variance instead of whole population
- Ensure Identity Transform can be represented

Mini-Batch Normalization

Two simplifications and Identity Transform

- Normalize each scalar feature independently
- Use mini-batch to estimate the mean and variance instead of whole population
- Ensure Identity Transform can be represented

$$y^{(k)} = \gamma^{(k)} \hat{x}^{(k)} + \beta^{(k)}$$

Two new parameters for each activation are introduced for learning.

Mini-Batch Normalization

Two simplifications and Identity Transform

- Normalize each scalar feature independently
- Use mini-batch to estimate the mean and variance instead of whole population
- Ensure Identity Transform can be represented

$$y^{(k)} = \gamma^{(k)} \hat{x}^{(k)} + \beta^{(k)}$$

Two new parameters for each activation are introduced for learning.

Batch Normalization Transform, see reference paper for details

Overview

1. Batch Normalization
   - Internal Covariate Shift
   - Mini-Batch Normalization
   - Key Points in Batch Normalization
   - Experiments and Results
2. Importance of Initialization and Momentum
   - Overview of first-order method
   - Momentum & Nesterov's Accelerated Gradient(NAG)
   - Deep Autoencoders & RNN - Echo-State Networks

Key Points in Batch Normalization

Key Points in Batch Normalization

- Original parameters and newly introduced γ and β will be trained.

Key Points in Batch Normalization

- Original parameters and newly introduced γ and β will be trained.
- When in inference, the whole population of training data is used for mean and variance statistics instead of the estimate.

$$E(x) \leftarrow E_{\mathcal{B}}[\mu_{\mathcal{B}}]$$

$$Var[x] \leftarrow \frac{m}{m-1} E_{\mathcal{B}}[\sigma^2_{\mathcal{B}}]$$

Key Points in Batch Normalization

- Original parameters and newly introduced γ and β will be trained.
- When in inference, the whole population of training data is used for mean and variance statistics instead of the estimate.

$$E(x) \leftarrow E_{\mathcal{B}}[\mu_{\mathcal{B}}]$$

$$Var[x] \leftarrow \frac{m}{m-1} E_{\mathcal{B}}[\sigma_{\mathcal{B}}^2]$$

- In Convolutional layers, different locations of a feature map should be normalized in the same way.

$$m' = |\mathcal{B}| = m \cdot pq, \ and \ \gamma^{(k)}, \ \beta^{(k)} \ per \ feature \ map$$

Key Points in Batch Normalization

Key Points in Batch Normalization

- Higher learning rates are allowed

Key Points in Batch Normalization

- Higher learning rates are allowed

$$BN(Wu) = BN((aW)u)$$

Key Points in Batch Normalization

- Higher learning rates are allowed

$$BN(Wu) = BN((aW)u)$$

$$\frac{\partial BN(Wu)}{\partial u} = \frac{\partial BN((aW)u)}{\partial u}, \frac{\partial BN(Wu)}{\partial aW} = \frac{1}{a} \cdot \frac{\partial BN((aW)u)}{\partial W}$$

Key Points in Batch Normalization

- Higher learning rates are allowed

$$BN(Wu) = BN((aW)u)$$

$$\frac{\partial BN(Wu)}{\partial u} = \frac{\partial BN((aW)u)}{\partial u}, \frac{\partial BN(Wu)}{\partial aW} = \frac{1}{a} \cdot \frac{\partial BN((aW)u)}{\partial W}$$

- Batch Normalization will regularize the model with less overfitting.

Overview

1. Batch Normalization
   - Internal Covariate Shift
   - Mini-Batch Normalization
   - Key Points in Batch Normalization
   - Experiments and Results
2. Importance of Initialization and Momentum
   - Overview of first-order method
   - Momentum & Nesterov's Accelerated Gradient(NAG)
   - Deep Autoencoders & RNN - Echo-State Networks

Activations over time

Activations over time

Batch Normalization helps train faster and achieve higher accuracy.

Activations over time

Batch Normalization helps train faster and achieve higher accuracy.

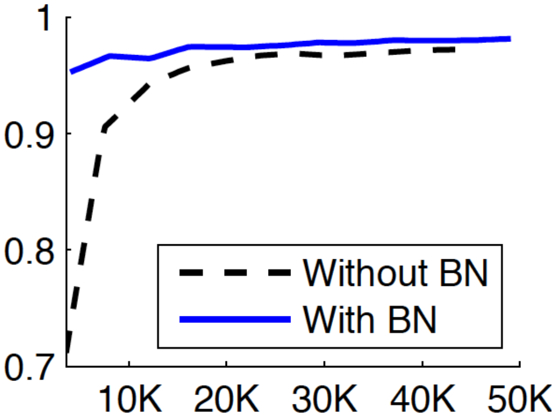


figure credit: reference paper

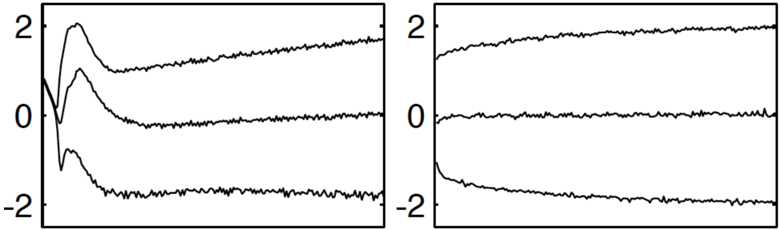
Activations over time

Activations over time

Batch Normalization makes input distribution more stable.

Activations over time

Batch Normalization makes input distribution more stable.



(b) Without BN (c) With BN

figure credit: reference paper

Accelerating Batch Normalization Networks

Accelerating Batch Normalization Networks

Tricks to follow

Accelerating Batch Normalization Networks

Tricks to follow

- Increasing learning rate
- Remove or Reduce Dropout
- Reduce ℓ_2 weight regularization
- Accelerate the learning rate decay
- Remove Local Response Normalization
- Shuffle training examples more thoroughly
- Reduce the photometric distortions

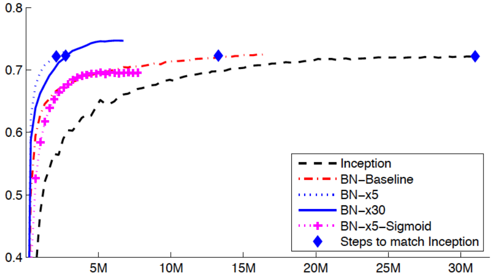
Network Comparisons

Network Comparisons

Inception, BN-Baseline, BN-x5, BN-x30, BN-x5-Sigmoid

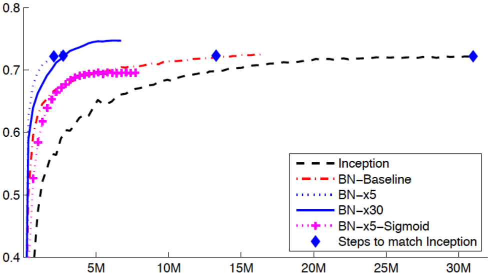
Network Comparisons

Inception, BN-Baseline, BN-x5, BN-x30, BN-x5-Sigmoid



Network Comparisons

Inception, BN-Baseline, BN-x5, BN-x30, BN-x5-Sigmoid



Model	Steps to 72.2%	Max accuracy
Inception	$31.0 \cdot 10^6$	72.2%
BN-Baseline	$13.3 \cdot 10^6$	72.7%
BN-x5	$2.1 \cdot 10^6$	73.0%
BN-x30	$2.7 \cdot 10^6$	74.8%
BN-x5-Sigmoid		69.8%

figure credit: reference paper

Ensemble Classification

Ensemble Classification

Top-5 validation error of 4.9% and test error of 4.82%, exceeds the estimated accuracy of human raters.

Ensemble Classification

Top-5 validation error of 4.9% and test error of 4.82%, exceeds the estimated accuracy of human raters.

Model	Resolution	Crops	Models	Top-1 error	Top-5 error
GoogLeNet ensemble	224	144	7	-	6.67%
Deep Image low-res	256	-	1	-	7.96%
Deep Image high-res	512	-	1	24.88	7.42%
Deep Image ensemble	variable	-	-	-	5.98%
BN-Inception single crop	224	1	1	25.2%	7.82%
BN-Inception multicrop	224	144	1	21.99%	5.82%
BN-Inception ensemble	224	144	6	20.1%	**4.9%***

figure credit: reference paper

Overview

1. Batch Normalization
   - Internal Covariate Shift
   - Mini-Batch Normalization
   - Key Points in Batch Normalization
   - Experiments and Results
2. Importance of Initialization and Momentum
   - Overview of first-order method
   - Momentum & Nesterov's Accelerated Gradient(NAG)
   - Deep Autoencoders & RNN - Echo-State Networks

Challenges to be solved

Challenges to be solved

Reference paper: On the importance of initialization and momentum in deep learning

Challenges to be solved

Reference paper: On the importance of initialization and momentum in deep learning

- Difficult to use first-order method to reach performance previously only achievable by second-order method like Hessian-Free.

Challenges to be solved

Reference paper: On the importance of initialization and momentum in deep learning

- Difficult to use first-order method to reach performance previously only achievable by second-order method like Hessian-Free.

- Well-designed random initialization
- Slowly increasing schedule for momentum parameter

Challenges to be solved

Reference paper: On the importance of initialization and momentum in deep learning

- Difficult to use first-order method to reach performance previously only achievable by second-order method like Hessian-Free.

- Well-designed random initialization
- Slowly increasing schedule for momentum parameter

- No need for sophisticated second-order methods.

Overview of first-order method

First-order Methods

- Vanilla Stochastic Gradient Descent
- SGD + Momentum
- Nesterov's Accelerated Gradient(NAG)
- AdaGrad
- Adam
- Rprop
- RMSProp
- AdaDelta

slide credit: Ishan Misra

Overview

1. Batch Normalization
   - Internal Covariate Shift
   - Mini-Batch Normalization
   - Key Points in Batch Normalization
   - Experiments and Results
2. Importance of Initialization and Momentum
   - Overview of first-order method
   - Momentum & Nesterov's Accelerated Gradient(NAG)
   - Deep Autoencoders & RNN - Echo-State Networks

Several First-order Methods

Several First-order Methods

Notation:

- θ - Parameters of network, f - Objective function, ϵ - Learning rate
- ∇f - Gradient of f, v - Velocity vector, μ - Momentum coefficient

Several First-order Methods

Notation:

- θ - Parameters of network, f - Objective function, ϵ - Learning rate
- ∇f - Gradient of f, v - Velocity vector, μ - Momentum coefficient

Vanilla SGD

$$v_{t+1} = \epsilon \nabla f(\theta_t)$$

$$\theta_{t+1} = \theta_t - v_{t+1}$$

slide credit: Ishan Misra

Several First-order Methods

Several First-order Methods

Rprop Update

if $\nabla f_t \nabla f_{t-1} > 0$

$$v_t = \eta^+ v_{t-1}$$

else if $\nabla f_t \nabla f_{t-1} < 0$

$$v_t = \eta^- v_{t-1}$$

else

$$v_t = v_{t-1}$$

$$\theta_{t+1} = \theta_t - v_t$$

where $0 < \eta^- < 1 < \eta^+$

slide credit: Ishan Misra

Several First-order Methods

Several First-order Methods

AdaGrad

$$r_t = \theta_t^2 + r_{t-1}$$
$$v_{t+1} = \frac{\alpha}{\sqrt{r_t}} \nabla f(\theta_t)$$
$$\theta_{t+1} = \theta_t - v_{t+1}$$

Several First-order Methods

AdaGrad

$$r_t = \theta_t^2 + r_{t-1}$$
$$v_{t+1} = \frac{\alpha}{\sqrt{r_t}} \nabla f(\theta_t)$$
$$\theta_{t+1} = \theta_t - v_{t+1}$$

RMSProp = Rprop + SGD

$$r_t = (1-\gamma)\theta_t^2 + \gamma r_{t-1}$$
$$v_{t+1} = \frac{\alpha}{\sqrt{r_t}} \nabla f(\theta_t)$$
$$\theta_{t+1} = \theta_t - v_{t+1}$$

slide credit: Ishan Misra

Several First-order Methods

Several First-order Methods

AdaDelta

$$v_{t+1} = H^{-1}\nabla f,$$
$$\propto \frac{f'}{f''}$$
$$\propto \frac{1/units\ \ of\ \ \theta}{(1/units\ \ of\ \ \theta)^2}$$
$$\propto units\ \ of\ \ \theta$$

Several First-order Methods

AdaDelta

$$v_{t+1} = H^{-1}\nabla f,$$
$$\propto \frac{f'}{f''}$$
$$\propto \frac{1/units\ \ of\ \ \theta}{(1/units\ \ of\ \ \theta)^2}$$
$$\propto units\ \ of\ \ \theta$$

Adam

$$r_t = (1-\gamma_1)\nabla f(\theta_t) + \gamma_1 r_{t-1}$$
$$p_t = (1-\gamma_2)\nabla f(\theta_t)^2 + \gamma_2 p_{t-1}$$
$$\hat{r}_t = \frac{r_t}{(1-(1-\gamma_1)^t)}$$
$$\hat{p}_t = \frac{p_t}{(1-(1-r_2)^t)}$$
$$v_t = \alpha\frac{\hat{r}_t}{\sqrt{\hat{p}_t}}$$
$$\theta_{t+1} = \theta_t - v_t$$

slide credit: Ishan Misra

Overview

1. Batch Normalization
   - Internal Covariate Shift
   - Mini-Batch Normalization
   - Key Points in Batch Normalization
   - Experiments and Results
2. Importance of Initialization and Momentum
   - Overview of first-order method
   - Momentum & Nesterov's Accelerated Gradient(NAG)
   - Deep Autoencoders & RNN - Echo-State Networks

Momentum and NAG

Momentum and NAG

Notation:

- θ - Parameters of network, f - Objective function, ϵ - Learning rate
- ∇f - Gradient of f, v - Velocity vector, μ - Momentum coefficient

Momentum and NAG

Notation:

- θ - Parameters of network, f - Objective function, ϵ - Learning rate
- ∇f - Gradient of f, v - Velocity vector, μ - Momentum coefficient

Classical Momentum

$$v_{t+1} = \mu v_t - \epsilon \nabla f(\theta_t)$$

$$\theta_{t+1} = \theta_t + v_{t+1}$$

Momentum and NAG

Notation:

- θ - Parameters of network, f - Objective function, ϵ - Learning rate
- ∇f - Gradient of f, v - Velocity vector, μ - Momentum coefficient

Classical Momentum

$$v_{t+1} = \mu v_t - \epsilon \nabla f(\theta_t)$$

$$\theta_{t+1} = \theta_t + v_{t+1}$$

Nesterov's Accelerated Gradient

$$v_{t+1} = \mu v_t - \epsilon \nabla f(\theta_t + \mu v_t)$$

$$\theta_{t+1} = \theta_t + v_{t+1}$$

Relationship between CM and NAG

Relationship between CM and NAG

NAG uses $\theta_t + \mu v_t$ but MISSING the yet unknown correction. Thus when the addition of μv_t results in an immediate undesirable increase in the objective f, NAG uses $\theta_t + \mu v_t$ but MISSING the yet unknown correction. Thus when the addition of μv_t results in an immediate undesirable increase in the objective f,

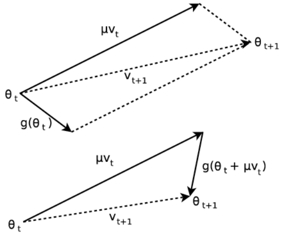


Figure 1. **(Top)** Classical Momentum **(Bottom)** Nesterov Accelerated Gradient

Relationship between CM and NAG

NAG uses $\theta_t + \mu v_t$ but MISSING the yet unknown correction. Thus when the addition of μv_t results in an immediate undesirable increase in the objective f,

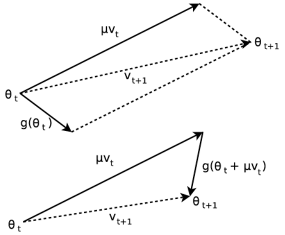


Figure 1. **(Top)** Classical Momentum **(Bottom)** Nesterov Accelerated Gradient

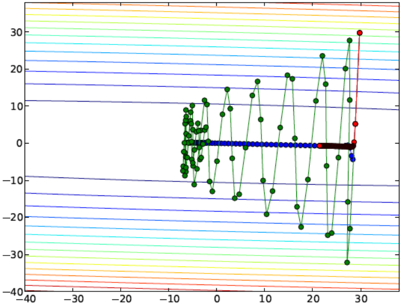


figure credit: reference paper

Relationship between CM and NAG

Relationship between CM and NAG

Apply CM and NAG to a positive definite quadratic objective $q(x) = x^T Ax/2 + b^T x$.

Difference in effective momentum coefficient

- Classical Momentum: μ
- NAG: $\mu(1 - \lambda\epsilon)$, where λ is the eigenvalue of A.

Relationship between CM and NAG

Apply CM and NAG to a positive definite quadratic objective $q(x) = x^T Ax/2 + b^T x$.

Difference in effective momentum coefficient

- Classical Momentum: μ
- NAG: $\mu(1 - \lambda\epsilon)$, where λ is the eigenvalue of A.

- ϵ small, CM and NAG are equivalent
- ϵ large, NAG gives smaller $\mu(1 - \lambda_i\epsilon)$ to stop oscillations.

Overview

1. Batch Normalization
   - Internal Covariate Shift
   - Mini-Batch Normalization
   - Key Points in Batch Normalization
   - Experiments and Results
2. Importance of Initialization and Momentum
   - Overview of first-order method
   - Momentum & Nesterov's Accelerated Gradient(NAG)
   - Deep Autoencoders & RNN - Echo-State Networks

Deep Autoencoders

Deep Autoencoders

Structure of Deep Autoencoder

Structure of Deep Autoencoder

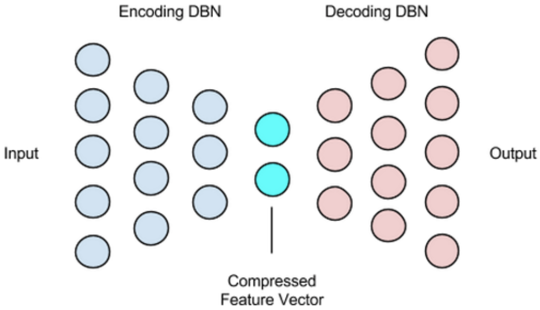


figure credit: http://deeplearning4j.org/deepautoencoder.html

Deep Autoencoders

Deep Autoencoders

- **Sparse Initialization**-each random unit connected to 15 randomly chosen units in the previous layer, drawn from a unit Gaussian
- **Schedule for Momentum Coefficient**

$$\mu_t = min(1 - 2^{-1-log_2(\lfloor t/250 \rfloor + 1)}, \mu_{max})$$

Deep Autoencoders

- **Sparse Initialization**-each random unit connected to 15 randomly chosen units in the previous layer, drawn from a unit Gaussian
- **Schedule for Momentum Coefficient**

$$\mu_t = min(1 - 2^{-1-log_2(\lfloor t/250 \rfloor + 1)}, \mu_{max})$$

  - $\mu_t = 1 - 3/(t+5)$, not strongly convex - Nesterov(1983)
  - constant μ_t, strongly convex - Nesterov(2003)

Deep Autoencoders

- **Sparse Initialization**-each random unit connected to 15 randomly chosen units in the previous layer, drawn from a unit Gaussian
- **Schedule for Momentum Coefficient**

$$\mu_t = min(1 - 2^{-1-log_2(\lfloor t/250 \rfloor + 1)}, \mu_{max})$$

  - $\mu_t = 1 - 3/(t+5)$, not strongly convex - Nesterov(1983)
  - constant μ_t, strongly convex - Nesterov(2003)

task	$0_{(SGD)}$	0.9N	0.99N	0.995N	0.999N	0.9M	0.99M	0.995M	0.999M	SGD_C	HF[†]	HF[*]
Curves	0.48	0.16	0.096	0.091	**0.074**	0.15	0.10	0.10	0.10	0.16	0.058	0.11
Mnist	2.1	1.0	**0.73**	0.75	0.80	1.0	0.77	0.84	0.90	0.9	0.69	1.40
Faces	36.4	14.2	8.5	7.8	**7.7**	15.3	8.7	8.3	9.3	NA	7.5	12.0

table credit: reference paper

RNN - Echo-State Networks

RNN - Echo-State Networks

Echo-State Networks(a family RNNs)

Echo-State Networks(a family RNNs)

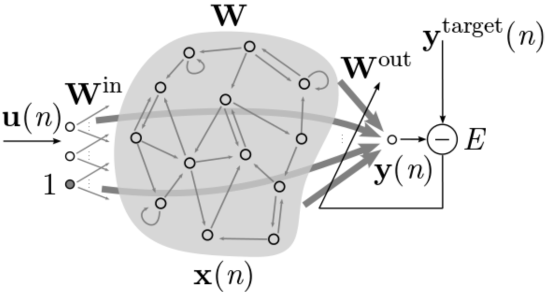


figure credit: Mantas Lukoevicius

- Hidden-to-output connections learned from data
- Recurrent connections fixed to a random draw from a specific distribution

RNN - Echo-State Networks

RNN - Echo-State Networks

ESN-based Initialization

- Spectral Radius of Hidden-to-hidden matrix around 1.1
- Initial scale of Input-to-hidden connections plays an important role (Gaussian draw with a standard deviation of 0.001 achieves good balance, but is Task Dependent)

ESN-based Initialization

- Spectral Radius of Hidden-to-hidden matrix around 1.1
- Initial scale of Input-to-hidden connections plays an important role (Gaussian draw with a standard deviation of 0.001 achieves good balance, but is Task Dependent)

Schedule of Momentum coefficient μ

- $\mu = 0.9$ for the first 1000 parameters;
- $\mu = \mu_0 \in \{0, 0.9, 0.98, 0.995\}$ afterwards;

RNN - Echo-State Networks

ESN-based Initialization

- Spectral Radius of Hidden-to-hidden matrix around 1.1
- Initial scale of Input-to-hidden connections plays an important role (Gaussian draw with a standard deviation of 0.001 achieves good balance, but is Task Dependent)

Schedule of Momentum coefficient μ

- $\mu = 0.9$ for the first 1000 parameters;
- $\mu = \mu_0 \in \{0, 0.9, 0.98, 0.995\}$ afterwards;

problem	biases	0	0.9N	0.98N	0.995N	0.9M	0.98M	0.995M
add $T = 80$	0.82	0.39	0.02	0.21	**0.00025**	0.43	0.62	0.036
mul $T = 80$	0.84	0.48	0.36	0.22	**0.0013**	0.029	0.025	0.37
mem-5 $T = 200$	2.5	1.27	1.02	0.96	**0.63**	1.12	1.09	0.92
mem-20 $T = 80$	8.0	5.37	2.77	0.0144	**0.00005**	1.75	0.0017	0.053

Table credit: reference paper

Overview

1. Batch Normalization
   - Internal Covariate Shift
   - Mini-Batch Normalization
   - Key Points in Batch Normalization
   - Experiments and Results
2. Importance of Initialization and Momentum
   - Overview of first-order method
   - Momentum & Nesterov's Accelerated Gradient(NAG)
   - Deep Autoencoders & RNN - Echo-State Networks

Questions?

Chris Dusold's Part

- Variance-SGD(No More Pesky Learning Rates)
- Adam(Adam: A Method for Stochastic Optimization)
- AdaGrad(Adaptive Subgradient Methods for Online Learning and Stochastic Optimization)