MAX PLANCK INSTITUTE
FOR DYNAMICS OF COMPLEX
TECHNICAL SYSTEMS
MAGDEBURG

CSC

COMPUTATIONAL METHODS IN
SYSTEMS AND CONTROL THEORY

20 YEARS
1998-2018

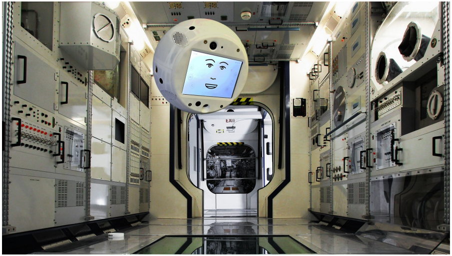
Deep Learning: An Introduction for Applied Mathematicians

Manuel Baumann

June 13, 2018

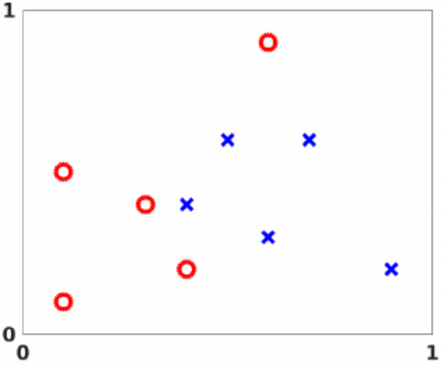
CSC Reading Group Seminar

Disclaimer



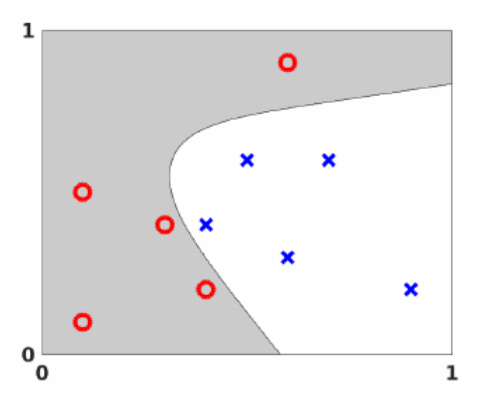
Claim: *I am not an expert in machine learning – But <u>we</u> are!*

Reference



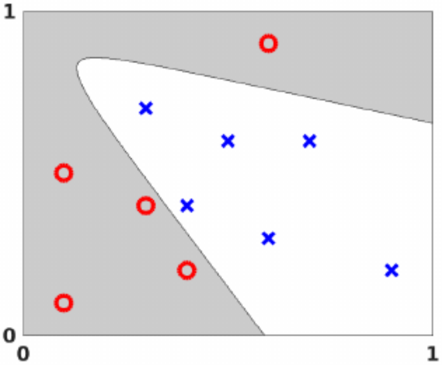
C. F. Higham and D. J. Higham (2018). *Deep Learning: An Introduction for Applied Mathematicians.* arXiv:1801.05894v1.

Reference



C. F. Higham and D. J. Higham (2018). *Deep Learning: An Introduction for Applied Mathematicians.* arXiv:1801.05894v1.

Reference



C. F. Higham and D. J. Higham (2018). *Deep Learning: An Introduction for Applied Mathematicians.* arXiv:1801.05894v1.

New names for old friends

Machine Learning	Numerical Analysis
neuron	smoothed step function

New names for old friends

Machine Learning	Numerical Analysis
neuron	smoothed step function
neural network	directed graph

New names for old friends

Machine Learning	**Numerical Analysis**
neuron	smoothed step function
neural network	directed graph
training phase	parameter fitting

New names for old friends

Machine Learning	Numerical Analysis
neuron	smoothed step function
neural network	directed graph
training phase	parameter fitting
stochastic gradient	steepest descent variant

New names for old friends

Machine Learning	**Numerical Analysis**
neuron	smoothed step function
neural network	directed graph
training phase	parameter fitting
stochastic gradient	steepest descent variant
learning rate	step size in line search

New names for old friends

Machine Learning	Numerical Analysis
neuron	smoothed step function
neural network	directed graph
training phase	parameter fitting
stochastic gradient	steepest descent variant
learning rate	step size in line search
back propagation	adjoint equation

New names for old friends

Machine Learning	Numerical Analysis
neuron	smoothed step function
neural network	directed graph
training phase	parameter fitting
stochastic gradient	steepest descent variant
learning rate	step size in line search
back propagation	adjoint equation
hidden layers	parameter (over-)fitting

New names for old friends

Machine Learning	Numerical Analysis
neuron	smoothed step function
neural network	directed graph
training phase	parameter fitting
stochastic gradient	steepest descent variant
learning rate	step size in line search
back propagation	adjoint equation
hidden layers	parameter (over-)fitting
'deep' learning	large-scale

New names for old friends

Machine Learning	Numerical Analysis
neuron	smoothed step function
neural network	directed graph
training phase	parameter fitting $\rightarrow p$
stochastic gradient	steepest descent variant
learning rate	step size in line search
back propagation	adjoint equation
hidden layers	parameter (over-)fitting
'deep' learning	large-scale
artificial intelligence	$f_p(x)$

New names for old friends

Machine Learning	**Numerical Analysis**
neuron	smoothed step function
neural network	directed graph
training phase	parameter fitting $\rightarrow p$
stochastic gradient	steepest descent variant
learning rate	step size in line search
back propagation	adjoint equation
hidden layers	parameter (over-)fitting
'deep' learning	large-scale
artificial intelligence	$f_p(x)$
???	model-order reduction

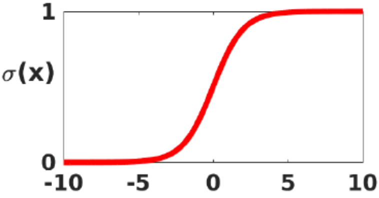
Overview

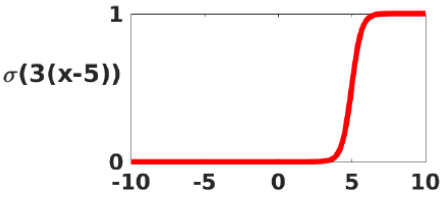
The General Set-up

The sigmoid function,

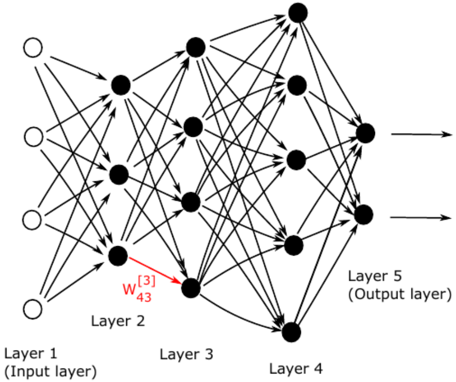
$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

models the bahavior of a neuron in the brain.





A neural network



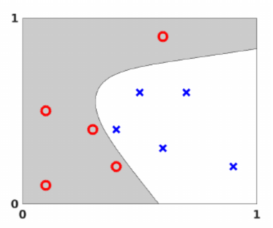
$$a^{[1]} = x \in \mathbb{R}^{n_1},$$

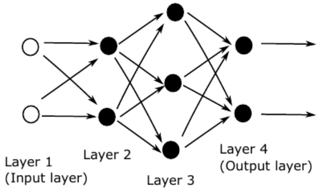
$$a^{[l]} = \sigma\left(W^{[l]} a^{[l-1]} + b^{[l]}\right) \in \mathbb{R}^{n_l}, \quad l = 2, 3, ..., L$$

The entering example

$$\min_{W^{[j]},b^{[j]}} \frac{1}{10}\sum_{i=1}^{10}\frac{1}{2}\|y(x^i)-F(x^i)\|_2^2, \quad j\in\{2,3,4\},$$

$$F(x)=\sigma\left(W^{[4]}\sigma\left(W^{[3]}\sigma\left(W^{[2]}x+b^{[2]}\right)+b^{[3]}\right)b^{[4]}\right)\in\mathbb{R}^2$$

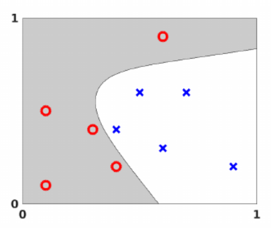


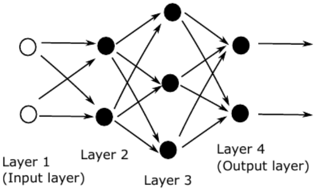


The entering example

$$\min_{p\in\mathbb{R}^{23}} \frac{1}{10}\sum_{i=1}^{10}\frac{1}{2}\|y(x^i)-F(x^i)\|_2^2,$$

$$F(x)=\sigma\left(W^{[4]}\sigma\left(W^{[3]}\sigma\left(W^{[2]}x+b^{[2]}\right)+b^{[3]}\right)b^{[4]}\right)\in\mathbb{R}^2$$





Stochastic Gradient

Objective function:

$$\mathcal{J}(p) = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{2} \| y(x^i) - a^{[L]}(x^i, p) \|_2^2$$

Stochastic Gradient

Objective function:

$$\mathcal{J}(p) = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{2} \|y(x^i) - a^{[L]}(x^i, p)\|_2^2$$

Steepest descent:

$$p \leftarrow p - \eta \nabla \mathcal{J}(p), \quad \eta \in \mathbb{R}_+ \text{ is called 'learning rate'.}$$

Stochastic Gradient

Objective function:

$$\mathcal{J}(p) = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{2}\|y(x^i) - a^{[L]}(x^i,p)\|_2^2 =: \frac{1}{N}\sum_{i=1}^{N} C_i(x^i,p)$$

Steepest descent:

$$p \leftarrow p - \eta\nabla\mathcal{J}(p), \quad \eta \in \mathbb{R}_+ \text{ is called 'learning rate'.}$$

Stochastic gradient:

$$\nabla\mathcal{J}(p) = \frac{1}{N}\sum_{i=1}^{N}\nabla_p C_i(x^i,p) \approx \frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}}\nabla_p C_i(x^i,p)$$

Stochastic Gradient

Objective function:

$$\mathcal{J}(p) = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{2}\|y(x^i) - a^{[L]}(x^i,p)\|_2^2 =: \frac{1}{N}\sum_{i=1}^{N}C_i(x^i,p)$$

Steepest descent:

$$p \leftarrow p - \eta\nabla\mathcal{J}(p), \quad \eta \in \mathbb{R}_+ \text{ is called 'learning rate'.}$$

Stochastic gradient:

$$\nabla\mathcal{J}(p) = \frac{1}{N}\sum_{i=1}^{N}\nabla_p C_i(x^i,p) \approx \frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}}\nabla_p C_i(x^i,p)$$

Back Propagation (1/3)

Now, $p \sim \left\{ \left[W^{[l]}\right]_{j,k}, \left[b^{[l]}\right]_j \right\}$. Let $z^{[l]} := W^{[l]} a^{[l-1]} + b^{[l]}$ and $\delta_j^{[l]} := \frac{\partial C}{\partial z_j^{[l]}}$.

Lemma: Back Propagation

The partial derivatives are given by,

$$\delta^{[L]} = \sigma'(z^{[L]}) \cdot (a^{[L]} - y), \tag{1}$$

$$\delta^{[l]} = \sigma'(z^{[l]}) \cdot (W^{[l+1]})^T \delta^{[l+1]}, \quad 2 \leq l \leq L-1, \tag{2}$$

$$\frac{\partial C}{\partial b_j^{[l]}} = \delta_j^{[l]}, \quad 2 \leq l \leq L, \tag{3}$$

$$\frac{\partial C}{\partial w_{jk}^{[l]}} = \delta_j^{[l]} a_k^{[l-1]}, \quad 2 \leq l \leq L. \tag{4}$$

Back Propagation (2/3)

Proof.

We prove (1) component-wise:

$$\delta_j^{[L]} = \frac{\partial C}{\partial z_j^{[L]}} = \frac{\partial C}{\partial a_j^{[L]}} \frac{\partial a_j^{[L]}}{\partial z_j^{[L]}} = (a_j^{[L]} - y_j)\sigma'(z_j^{[L]}) = (a_j^{[L]} - y_j)(\sigma(z_j^{[L]}) - \sigma^2(z_j^{[L]}))$$

Proof.

We prove (1) component-wise:

$$\delta_j^{[L]} = \frac{\partial C}{\partial z_j^{[L]}} = \frac{\partial C}{\partial a_j^{[L]}} \frac{\partial a_j^{[L]}}{\partial z_j^{[L]}} = (a_j^{[L]} - y_j)\sigma'(z_j^{[L]}) = (a_j^{[L]} - y_j)(\sigma(z_j^{[L]}) - \sigma^2(z_j^{[L]}))$$

Next, we prove (2) component-wise:

$$\delta_j^{[l]} = \frac{\partial C}{\partial z_j^{[l]}} = \sum_{k=1}^{n_{l+1}} \frac{\partial C}{\partial z_k^{[l+1]}} \frac{\partial z_k^{[l+1]}}{\partial z_j^{[l]}} = \sum_{k=1}^{n_{l+1}} \delta_k^{[l+1]} \frac{\partial z_k^{[l+1]}}{\partial z_j^{[l]}}$$
$$= \sum_{k=1}^{n_{l+1}} \delta_k^{[l+1]} w_{kj}^{[l+1]} \sigma'(z_j^{[l]}),$$

where $z_k^{[l+1]} = \sum_{s=1}^{n_l} w_{ks}^{[l+1]} \sigma(z_s^{[l]}) + b_k^{[l+1]}$.

Back Propagation (2/3)

Proof.

We prove (1) component-wise:

$$\delta_j^{[L]} = \frac{\partial C}{\partial z_j^{[L]}} = \frac{\partial C}{\partial a_j^{[L]}} \frac{\partial a_j^{[L]}}{\partial z_j^{[L]}} = (a_j^{[L]} - y_j)\sigma'(z_j^{[L]}) = (a_j^{[L]} - y_j)(\sigma(z_j^{[L]}) - \sigma^2(z_j^{[L]}))$$

Next, we prove (2) component-wise:

$$\delta_j^{[l]} = \frac{\partial C}{\partial z_j^{[l]}} = \sum_{k=1}^{n_{l+1}} \frac{\partial C}{\partial z_k^{[l+1]}} \frac{\partial z_k^{[l+1]}}{\partial z_j^{[l]}} = \sum_{k=1}^{n_{l+1}} \delta_k^{[l+1]} \frac{\partial z_k^{[l+1]}}{\partial z_j^{[l]}}$$
$$= \sum_{k=1}^{n_{l+1}} \delta_k^{[l+1]} w_{kj}^{[l+1]} \sigma'(z_j^{[l]}),$$

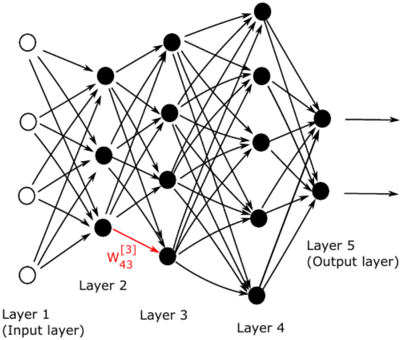
where $z_k^{[l+1]} = \sum_{s=1}^{n_l} w_{ks}^{[l+1]} \sigma(z_s^{[l]}) + b_k^{[l+1]}$.

(3) and (4) similar. □

Back Propagation (3/3)

Interpretation:

- Evaluation of $a^{[L]}$ requires a so-called forward pass: $a^{[1]}, z^{[2]} \to a^{[2]}, z^{[3]} \to ... \to a^{[L]}$
- Compute: $\sigma^{[L]}$ via (1)
- Compute: backward pass (2) $\sigma^{[L]} \to \sigma^{[L-1]} \to ... \to \sigma^{[2]}$
- Gradients via (3)-(4)



MATLAB Example

Acknowledgements

We are grateful to the MatConvNet team for making their package available under a permissive BSD license. The MATLAB code in Listings 6.1 and 6.2 can be found at

`http://personal.strath.ac.uk/d.j.higham/algfiles.html`

as well as an exteneded version that produces Figures 7 and 8, and a MATLAB code that uses `lsqnonlin` to produce Figure 4.

personal.strath.ac.uk/d.j.higham/algfiles.html

See the book Matlab Guide for more info about MATLAB.

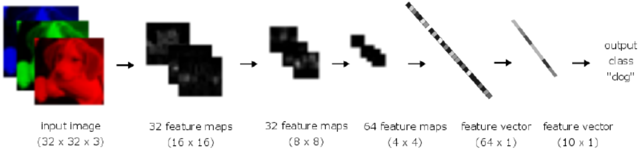
MATLAB files from **Deep Learning: An Introduction for Applied Mathematicians,** by C. F. Higham and D. J. Higham, manuscript, 2018.

- netbp.m from Listing 6.1
- netbpfull.m extended version of netbp.m that produces the figure
- activate.m from Listing 6.2
- nlsrun.m code from section 2 that uses MATLAB's lsqnonlin optimizer

Some more details

Convolutional Neural Network:

- Presented approach unfeasible for large data ($W^{[l]}$ is dense).
- Layers can be pre- and post-precessing steps used in image analysis; *filtering*, *max pooling*, *average pooling*, ...



Avoiding Overfitting:

- Trained network works well on given data, but *not* on new data.
- Splitting: *training* – *validation* data
- Dropout: independently remove neurons

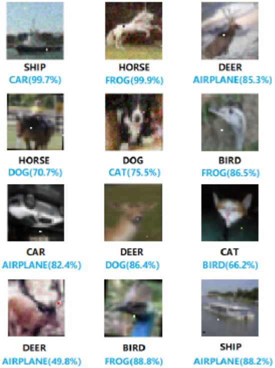
Current Research Directions

Research directions:

- proofs – e.g. when data is assumed to be i.i.d.
- in practice: design of *layers*
- perturbation theory: update trained network
- autoencoders: $\|x - G(F(x))\|_2^2 \to \min$

Two software examples:

- `http://www.vlfeat.org/matconvnet/`
- `http://scikit-learn.org/`



Summary

New names for old friends.

neuron	smoothed step function
neural network	directed graph
training phase	parameter fitting $\to p$
stochastic gradient	steepest descent variant
learning rate	step size in line search
back propagation	adjoint equation
hidden layers	parameter (over-)fitting
'deep' learning	large-scale
artificial intelligence	$f_p(x)$
PCA	POD

Further readings

- Andrew Ng. *Coursera Machine Learning* (online courses)
- M. Nielsen, *Neural Networks and Deep Learning*, Determination Press, 2015.
- I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, Boston, 2016.
- Y. LeCun, Y. Bengio, and G. Hinton, *Deep learning*, Nature, 521 (2015), pp. 436444.
- S. Mallat, *Understanding deep convolutional networks*, Philosophical Transactions of the Royal Society of London A, 374 (2016).