Lecture 7

Introduction to neural networks

Information Systems
(Machine Learning)

Andrey Filchenkov

08.11.2018

Lecture plan

- History of neural networks
- Single layer neural network
- Completeness problem of neural networks
- Multilayer neural networks
- Backpropagation
- Modularity
- Computational graph
- DNN best practices

- The presentation is prepared with materials of
  - K.V. Vorontsov “Machine Leaning”,
  - E. Gaaves "UVA Deep Learning Course“,
  - F.F. Li and A. Karpathy's course "Convolutional Neural Networks for Visual Recognition"
  - D. Polykovsky and K. Khrabrov “Neural networks in machine learning”
- Slides are available online: **goo.gl/BspjhF**

Lecture plan

- History of neural networks
- Completeness problem of neural networks
- Multilayer neural networks
- Backpropagation
- Modularity
- Computational graph
- DNN best practices

Early history

1943 Artificial neuron by McCulloch and Pitts

1949 Neuron learning rule by Hebb

1957 Perceptron by Rosenblatt

1960 Perceptron learning rule by Widrow and Hoff

1969 "Perceptrons" by Minski and Papert

1974 Back propagation algorithm by Webros and by Galushkin

Modern history

1980 Convolutional NN by Fukushima

1982 Recurrent NN by Hopfield

1991 "Vanishing gradient problem" was identified by Hochreiter

1997 Long short term memory network by Hochreiter and Schmidhuber

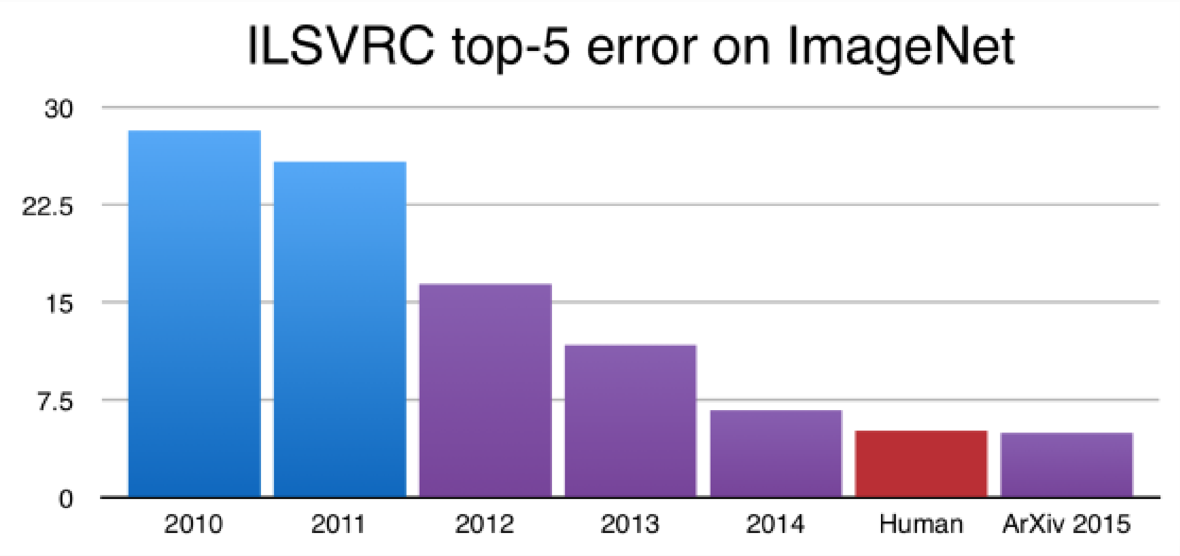
1998 Gradient descent for convolutional NN by LeCun et al.

2006 Deep model by Hinton, Osindero and Teh

Today history

2012 Hinton, Krizhevsky, and Sutskever suggest Dropout

2012 They win ImageNet (and two less known competitions). Deep learning era begins.



What now?

Since 2012, machine learning is getting focused on deep networks, AI and ML are usually referred to as deep learning.

State-of-the art in:

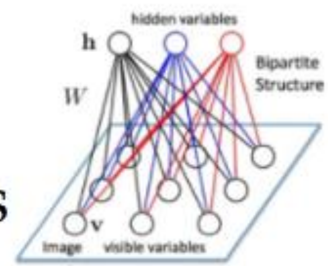
- image processing
- speech recognition
- image generation
- natural language processing and understanding
- learning strategies for games
- …

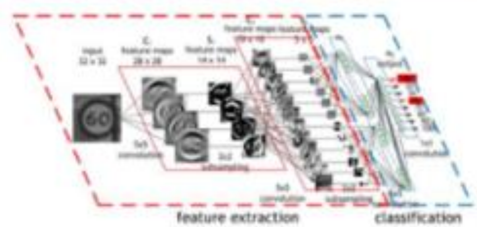
Why now?

Huge datasets

Powerful hardware

New algorithms





Why to go deep?

- Some functions cannot be efficiently represented (in terms of number of tunable elements) by architectures that are too shallow
- Functions that can be compactly represented by a depth k architecture might require an exponential number of computational elements to be represented by a depth $k-1$ architecture
- Deep representations might allow non-local generalization and comprehensibility
- Deep learning gets state of the art results in many fields (vision, audio, NLP, etc.)!

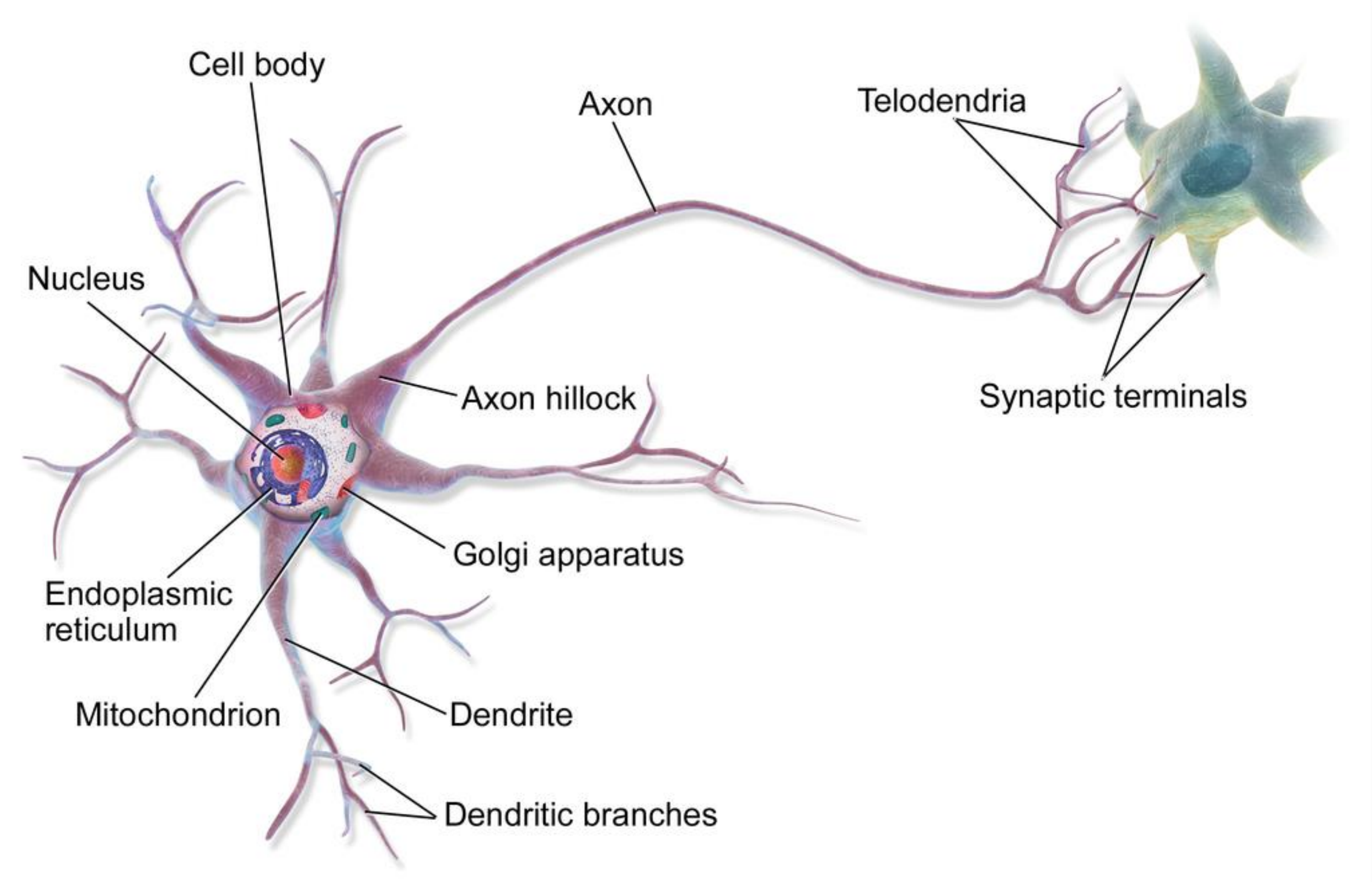
Three ideas

- (Hierarchical) Compositionality
  - Cascade of non-linear transformations
  - Multiple layers of representations
- End-to-End Learning
  - Learning (goal-driven) representations
  - Learning to feature extract
- Distributed Representations
  - Groups of neurons work together
  - Everything is a module

Lecture plan

- History of neural networks
- Single layer neural network
- Completeness problem of neural networks
- Multilayer neural networks
- Backpropagation
- Modularity
- Computational graph
- DNN best practices

Biological intuition

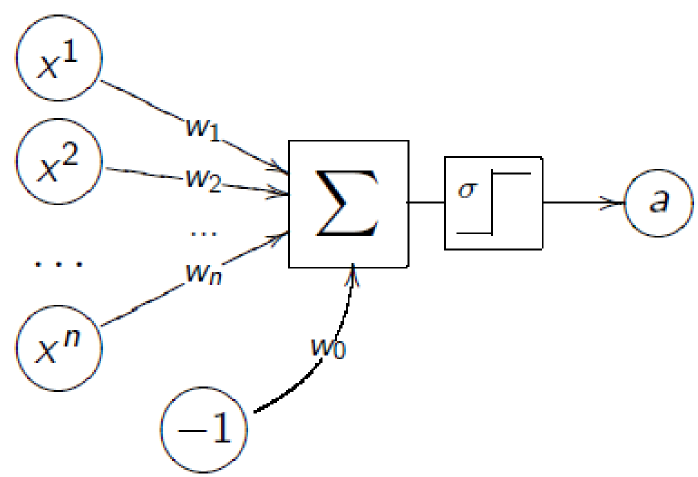


Perceptron

Rosenblatt's perceptron:

$$a_w(x, T^\ell) = \sigma\left(\sum_{i=1}^{n} w_i x^i - w_0\right),$$

where $\sigma(x) = 1$ if $x > 0$ and 0 otherwise.



Neuron

Generalized McCulloch-Pitts neuron:

$$a_w(x, T^\ell) = \sigma\left(\sum_{i=1}^{n} w_i f_i(x) - w_0\right),$$

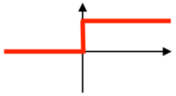
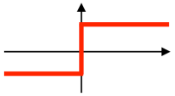
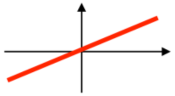
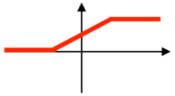
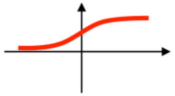
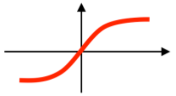
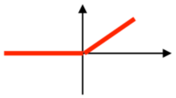
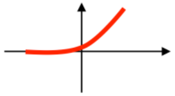
where σ is an activation function.

Generalized McCulloch-Pitts neuron:

$$a_w(x, T^\ell) = \sigma\left(\sum_{i=1}^{n} w_i f_i(x) - w_0\right),$$

where σ is an activation function.

Activation functions

Activation function	Equation	Example	1D Graph
Unit step (Heaviside)	$\phi(z) = \begin{cases} 0, & z < 0, \\ 0.5, & z = 0, \\ 1, & z > 0, \end{cases}$	Perceptron variant	
Sign (Signum)	$\phi(z) = \begin{cases} -1, & z < 0, \\ 0, & z = 0, \\ 1, & z > 0, \end{cases}$	Perceptron variant	
Linear	$\phi(z) = z$	Adaline, linear regression	
Piece-wise linear	$\phi(z) = \begin{cases} 1, & z \geq \frac{1}{2}, \\ z + \frac{1}{2}, & -\frac{1}{2} < z < \frac{1}{2}, \\ 0, & z \leq -\frac{1}{2}, \end{cases}$	Support vector machine	
Logistic (sigmoid)	$\phi(z) = \frac{1}{1 + e^{-z}}$	Logistic regression, Multi-layer NN	
Hyperbolic tangent	$\phi(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$	Multi-layer Neural Networks	
Rectifier, ReLU (Rectified Linear Unit)	$\phi(z) = max(0, z)$	Multi-layer Neural Networks	
Rectifier, softplus	$\phi(z) = \ln(1 + e^z)$	Multi-layer Neural Networks	

Scalar products in supervised learning

Classification:

$$Q(w, T^\ell) = \sum_{i=1}^{\ell} L(\langle w, x_i \rangle y_i) \to \min_w;$$

Regression:

$$Q(w, T^\ell) = \sum_{i=1}^{\ell} (\sigma(\langle w, x_i \rangle) - y_i)^2 \to \min_w.$$

Rosenblatt's rule and Hebb's rule

Rosenblatt's rule for $\{1; 0\}$ classification case for weight learning: for each object $x_{(k)}$ change the weight vector:

$$w^{[k+1]} := w^{[k]} - \eta\left(a_w\left(x_{(k)}\right) - y_{(k)}\right).$$

Hebb's rule for $\{1; -1\}$ classification case for weight learning: for each object $x_{(k)}$ change the weight vector:

if $\langle w^{[k]} x_{(k)} \rangle y_{(k)} < 0$ then $w^{[k+1]} := w^{[k]} + \eta x_{(k)} y_{(k)}$.

Delta rule

Let $L(a_w, x) = (\langle w, x \rangle - 1)^2$.

Delta-rule for weight learning: for each object $x_{(k)}$ change the weight vector:

$$w^{[k+1]} \coloneqq w^{[k]} - \eta\big(\langle w, x_{(k)} \rangle - y_{(k)}\big).$$

Lecture plan

- History of neural networks
- Single layer neural network
- Completeness problem of neural networks
- Multilayer neural networks
- Backpropagation
- Modularity
- Computational graph
- DNN best practices

Completeness problem (for neuron)

Basic idea: synthesize combinations of neurons.

Completeness problem: how rich is the family of functions that can be represented with a neural network?

Start with a single neuron.

Logical functions as neural networks

Logical AND

$$x^1 \wedge x^2 = [x^1 + x^2 - 3/2 > 0]$$

Logical OR

$$x^1 \vee x^2 = [x^1 + x^2 - 1/2 > 0]$$

Logical NOT

$$\neg x^1 = [-x^1 + 1/2 > 0]$$

Two ways of making it more complex

Example (Minkovski):

$$x^1 \oplus x^2$$

Two ways of making it more complex

1. Use non-linear transformation:

$$x^1 \oplus x^2 = [x^1 + x^2 - 2x^1x^2 - 1/2 > 0]$$

2. Build superposition:

$$x^1 \oplus x^2 = [(x^1 \vee x^2) - (x^1 \wedge x^2) - 1/2 > 0]$$

Completeness problem (Boolean functions)

Completeness problem: how rich is the family of functions that can be represented with a neural network?

DNF Theorem:

Any particular Boolean function can be represented by one and only one full disjunctive normal form.

What is with all possible functions?

Gorban Theorem

Theorem (Gorban, 1998)

Let

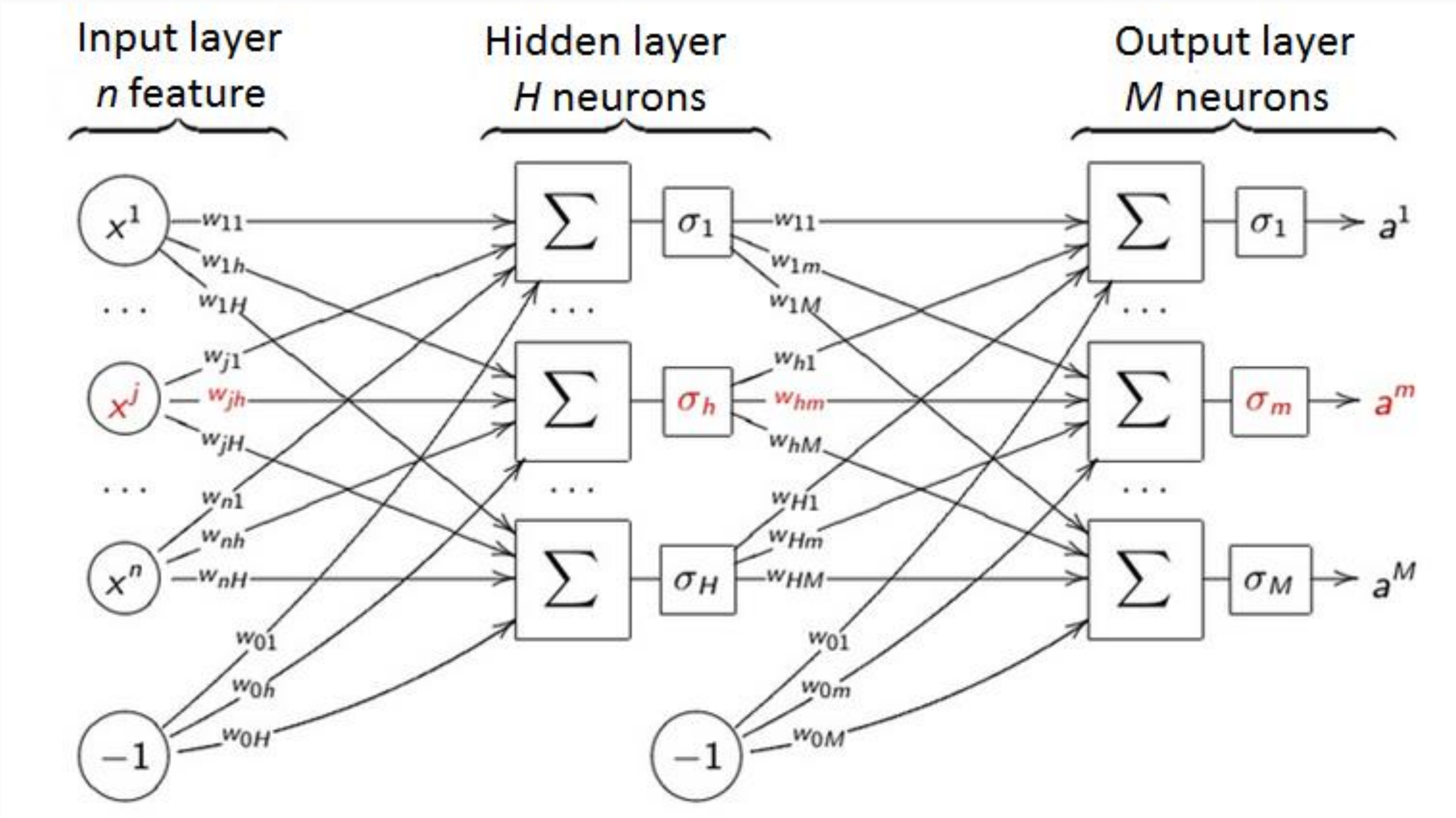
- X be a compact space,
- $C(X)$ be an algebra of continuous on X real-valued functions,
- F be linear subspace $C(X)$, closed with respect to a nonlinear continuous function ϕ and containing constant ($1 \in F$),
- F separates points in X.

Then F is dense in $C(X)$.

Lecture plan

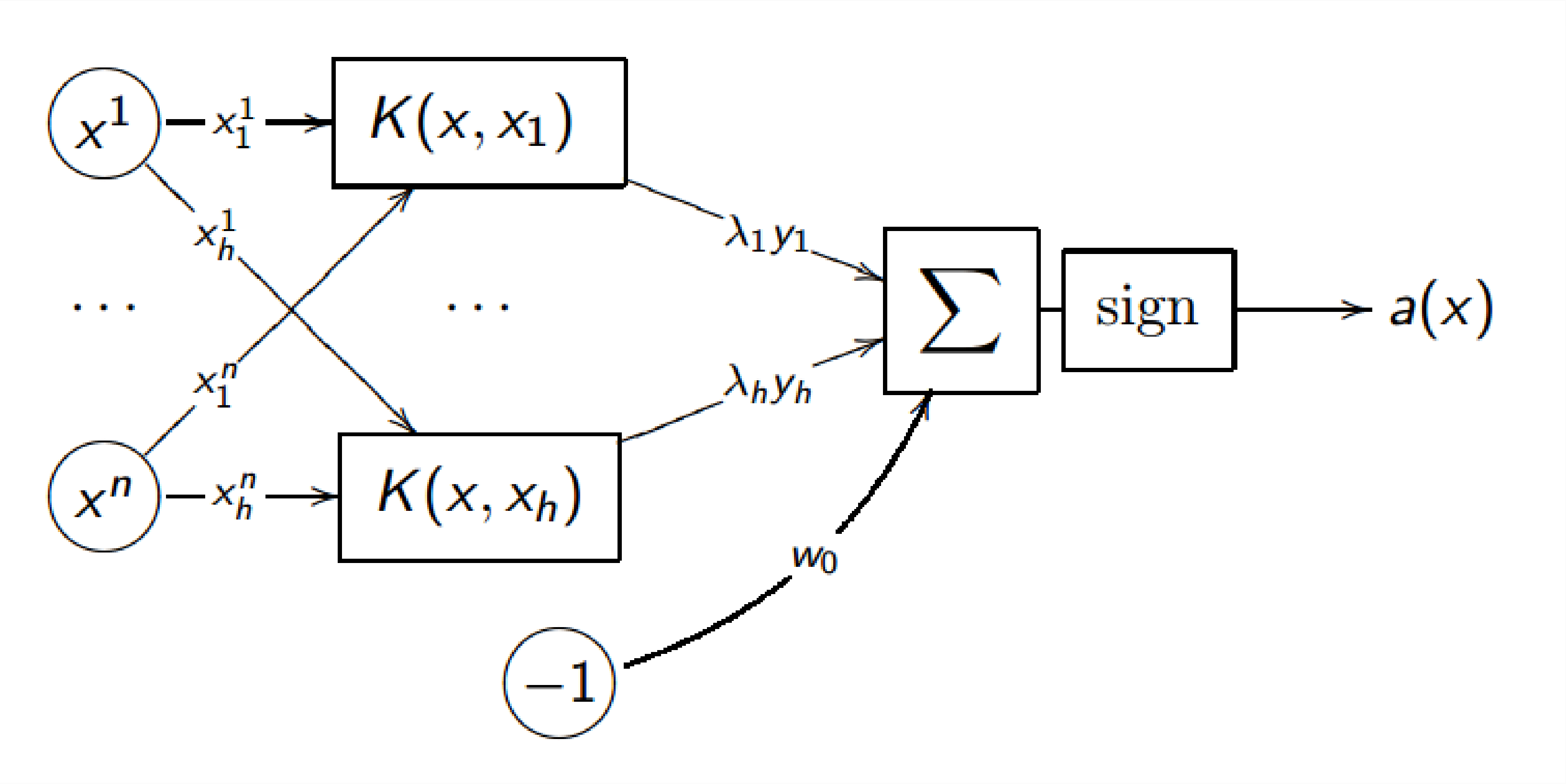
- History of neural networks
- Completeness problem of neural networks
- Single layer neural network
- Multilayer neural networks
- Backpropagation
- Modularity
- Computational graph
- DNN best practices

Multilayer neural network



Neural network for SVM

Order objects by margin:



Multilayer neural network

Any number of layers
Any number of neurons on each layer
Any number of ties between different layers

Weights adjusting

Use SGD to learn weights

$$w = \left(w_{jh}, w_{hm}\right) \in \mathbb{R}^{H(n+M-1)M}:$$

$$w^{[t+1]} = w^{[t]} - \eta \nabla L(w, x_i, y_i),$$

where $L(w, x_i, y_i)$ is a loss function (depends on the problem we are solving).

Lecture plan

- History of neural networks
- Completeness problem of neural networks
- Single layer neural network
- Multilayer neural networks
- Backpropagation
- Modularity
- Computational graph
- DNN best practices

Derivation of functions superposition

$$a^m(x_i) = \sigma_m\left(\sum_{h=0}^{H} w_{hm} u^h(x_i)\right);$$

$$u^h(x_i) = \sigma_h\left(\sum_{j=0}^{J} w_{jh} f_j(x_i)\right);$$

Let $L_i(w) = \frac{1}{2}\sum_{m=1}^{M}\left(a^m(x_i) - y_i^m\right)^2$.

Find partial derivatives

$$\frac{\partial L_i(w)}{\partial a^m};\frac{\partial L_i(w)}{\partial u^h}.$$

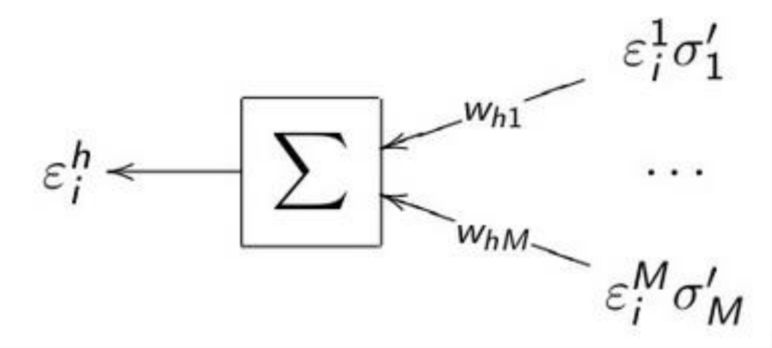
Errors on layers

$$\frac{\partial L_i(w)}{\partial a^m} = a^m(x_i) - y_i^m$$

$\varepsilon_i^m = a^m(x_i) - y_i^m$ is **error on output layer.**

$$\frac{\partial L_i(w)}{\partial u^h} = \sum_{i=1}^{M} \left(a^m(x_i) - y_i^m\right)\sigma'_m\, w_{hm} = \sum_{i=1}^{M} \varepsilon_i^m \sigma'_m\, w_{hm}$$

$\varepsilon_i^h = \sum_{i=1}^{M} \varepsilon_i^m \sigma'_m\, w_{hm}$ is **error on hidden layer.**



32

Backpropagation discussion (advantages)

Advantages:

- efficacy: gradient can be computed in a time, which is comparable to the time of the network processing;
- can be easily applied for any σ, L;
- can be applied in dynamical learning;
- not all the sample objects can be used;
- can be paralleled.

33

Backpropagation discussion (disadvantages)

Disadvantages:

- do not always converge;
- can stuck in local optima;
- number of neurons in the hidden layer should be fixed;
- the more ties, the probable overfitting is;
- "paralysis" of a single neuron and for network.

34

Lecture plan

- History of neural networks
- Completeness problem of neural networks
- Single layer neural network
- Multilayer neural networks
- Backpropagation
- Modularity
- Computational graph
- DNN best practices

Module

- A module is a building block for our network
- Each module is an object/function $a = h(x; \theta)$ that
  - Contains trainable parameters (θ)
  - Receives as an argument an input x
  - And returns an output a based on the activation function h …
- The activation function should be (at least) **first order differentiable (almost) everywhere**

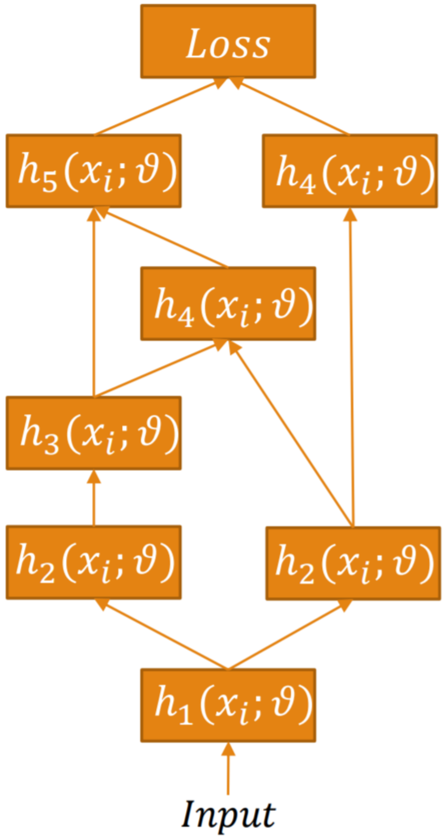
Composition of modules

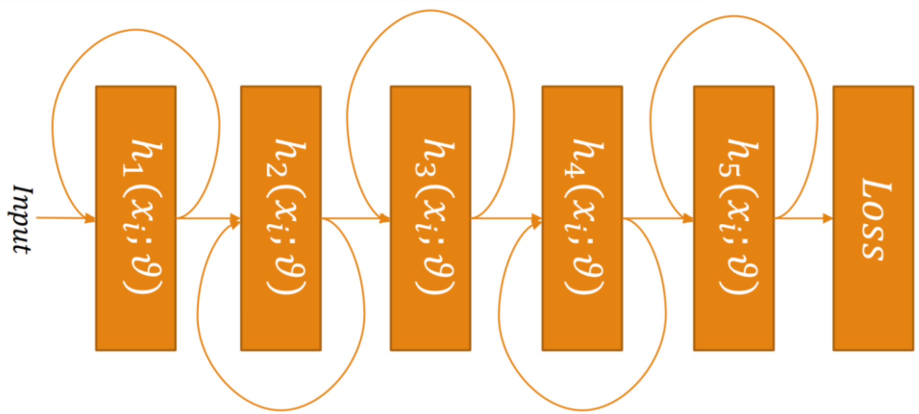
- A neural network is a composition of modules (building blocks)
- Any architecture works
- If the architecture is a feedforward cascade, no special care
- If acyclic, there is right order of computing the forward computations
- If there are loops, these form recurrent connections

Examples

A network may be complicated enough

Interweaved connections
(Directed Acyclic Graph architecture- DAGNN)



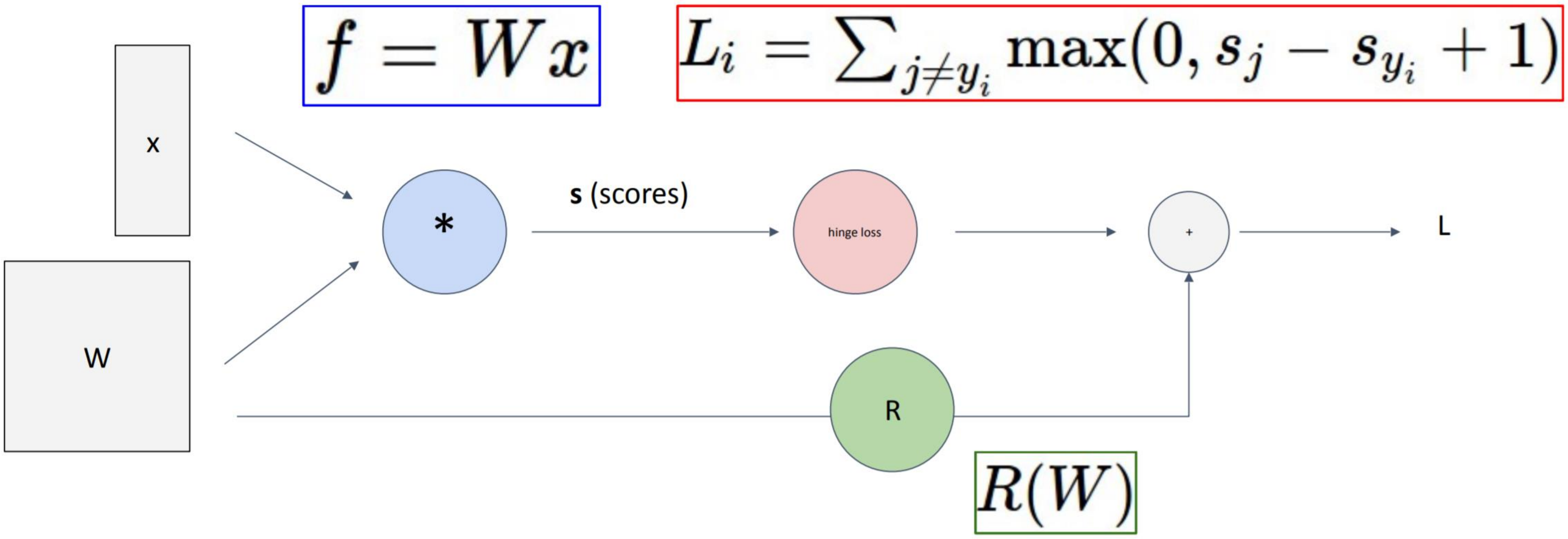


Loopy connections
(Recurrent architecture, special care needed)

Lecture plan

- History of neural networks
- Completeness problem of neural networks
- Single layer neural network
- Multilayer neural networks
- Backpropagation
- Modularity
- Computational graph
- DNN best practices

Computational graph

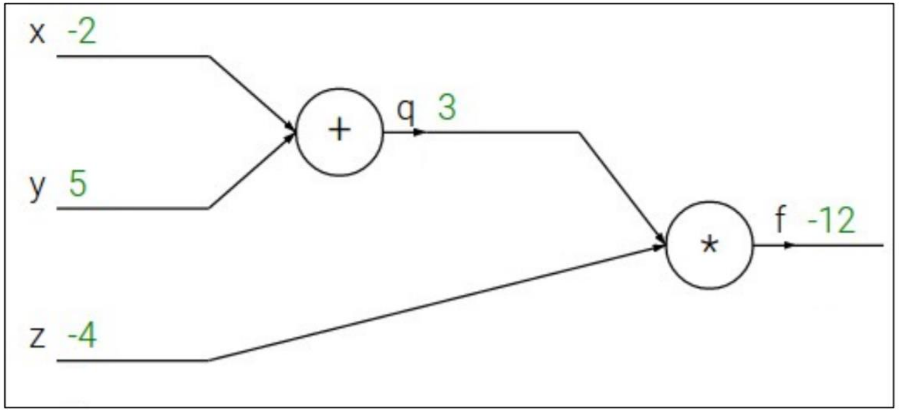


40

Example of conputations (1/13)

$$f(x, y, z) = (x + y)z$$

e.g. x = -2, y = 5, z = -4

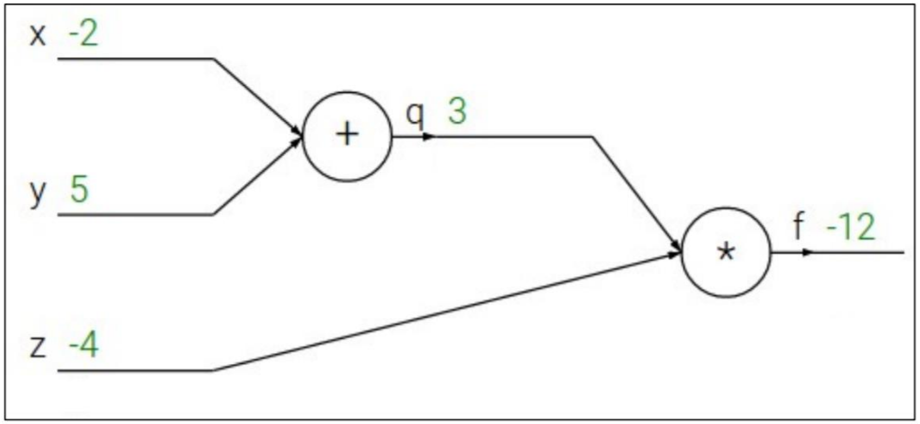


41

Example of computations (2/13)

$$f(x, y, z) = (x + y)z$$

e.g. x = -2, y = 5, z = -4



$$q = x + y \qquad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$

42

Example of computations (3/13)

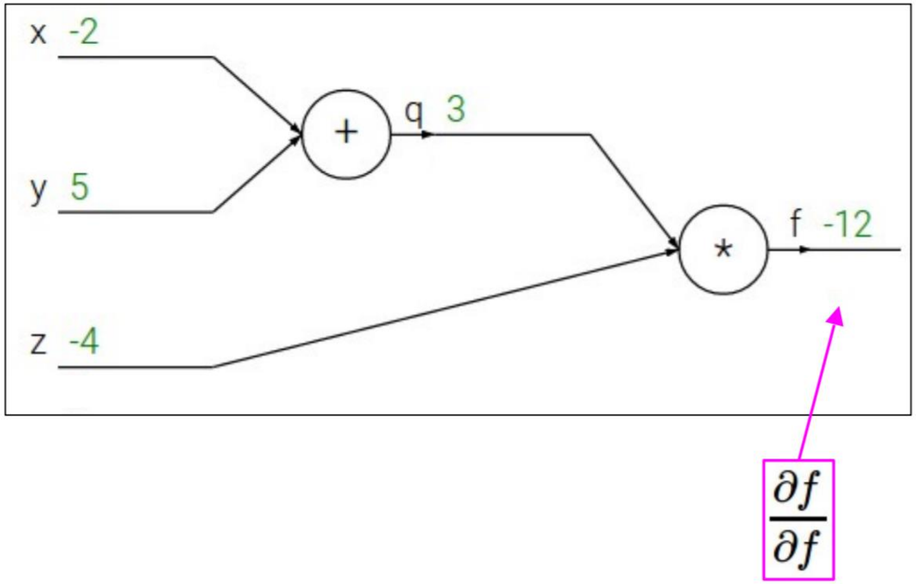
$$f(x, y, z) = (x + y)z$$

e.g. x = -2, y = 5, z = -4

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



43

Example of computations (4/13)

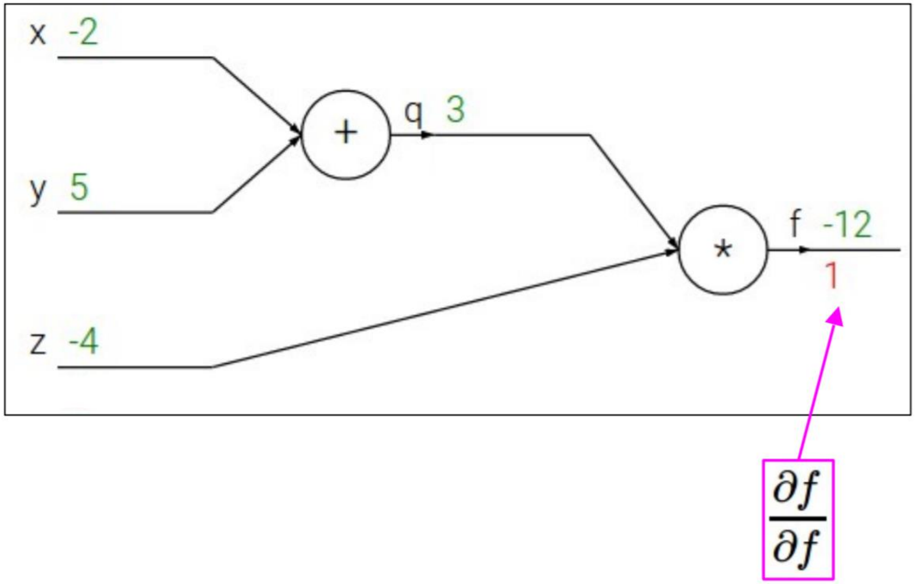
$$f(x, y, z) = (x + y)z$$

e.g. x = -2, y = 5, z = -4

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



44

Example of computations (5/13)

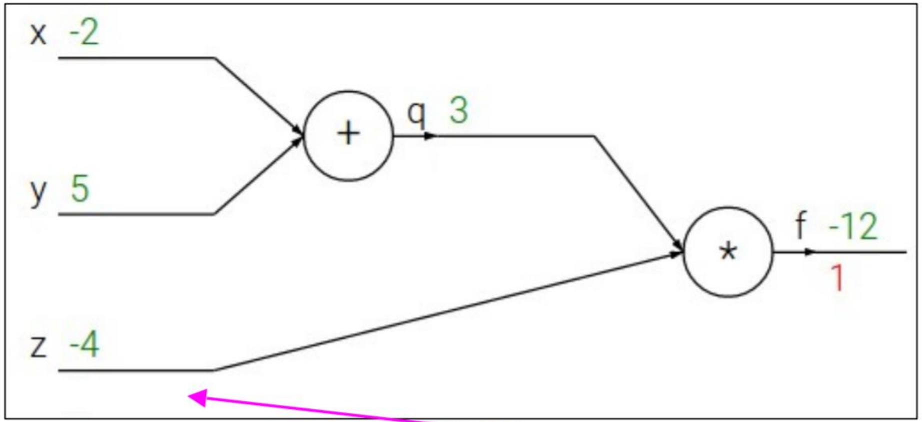
$$f(x, y, z) = (x + y)z$$

e.g. x = -2, y = 5, z = -4

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



$$\frac{\partial f}{\partial z}$$

45

Example of computations (6/13)

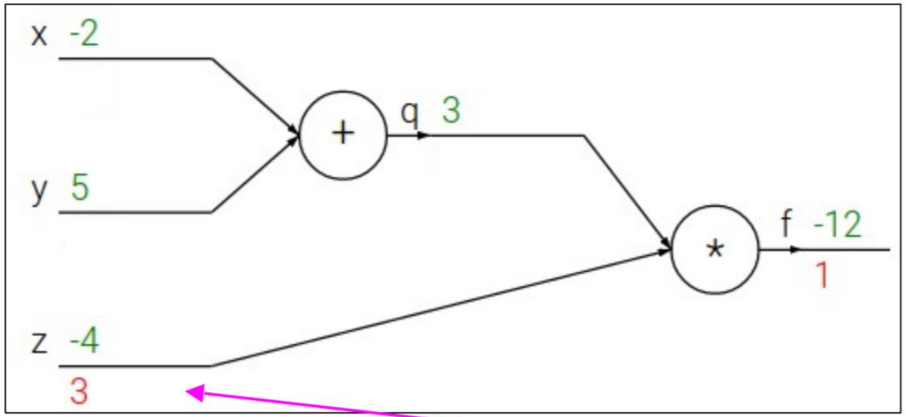
$$f(x, y, z) = (x + y)z$$

e.g. x = -2, y = 5, z = -4

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



$$\frac{\partial f}{\partial z}$$

Example of computations (7/13)

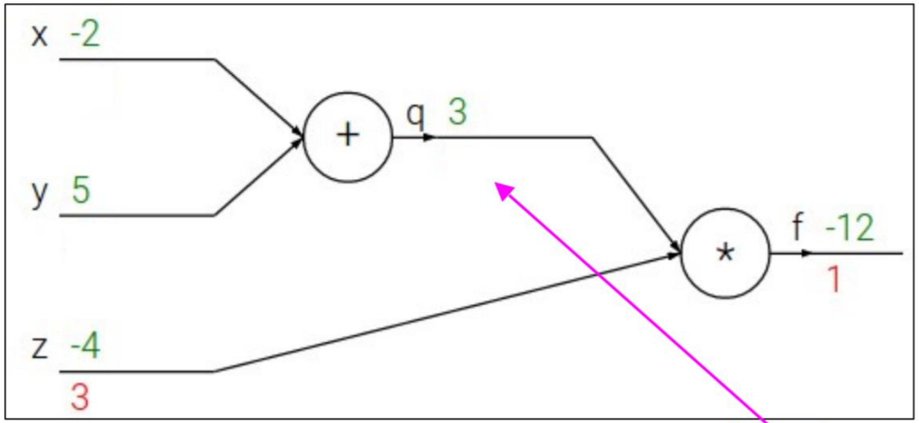
$$f(x, y, z) = (x + y)z$$

e.g. x = -2, y = 5, z = -4

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



$\frac{\partial f}{\partial q}$

47

Example of computations (8/13)

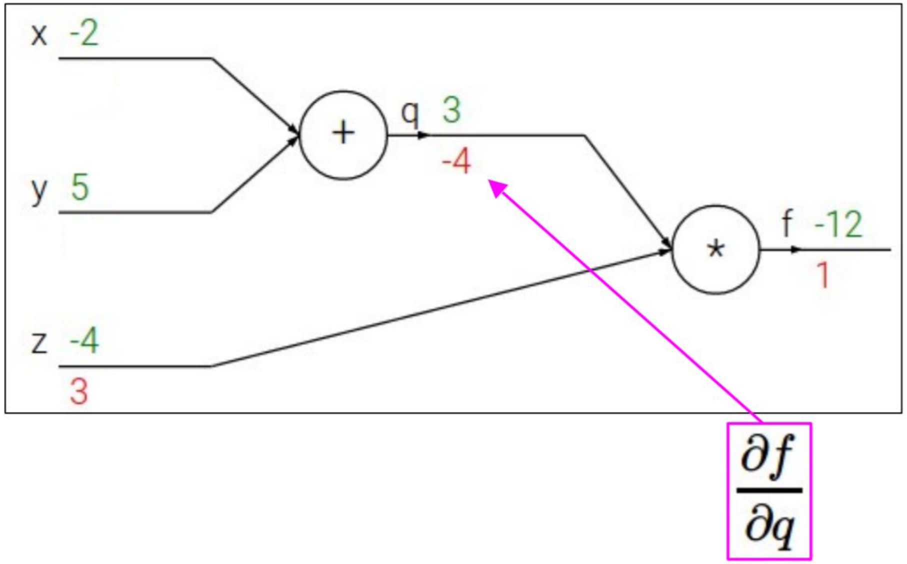
$$f(x, y, z) = (x + y)z$$

e.g. x = -2, y = 5, z = -4

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



48

Example of computations (9/13)

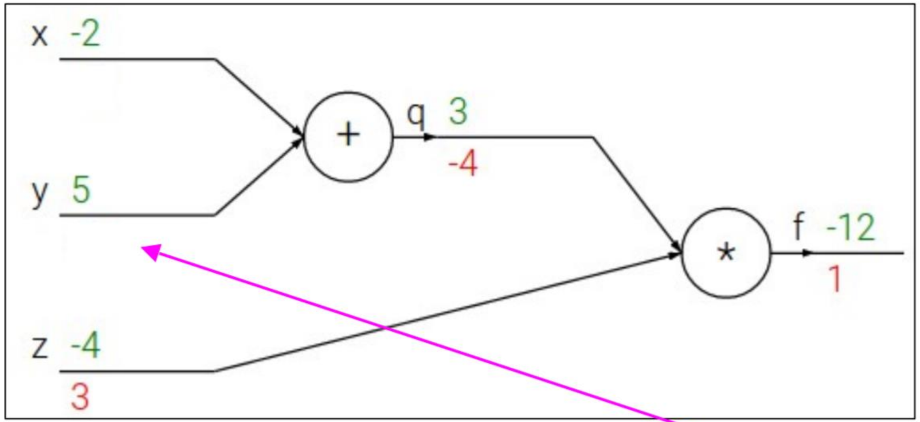
$$f(x, y, z) = (x + y)z$$

e.g. x = -2, y = 5, z = -4

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



$$\frac{\partial f}{\partial y}$$

49

Example of computations (10/13)

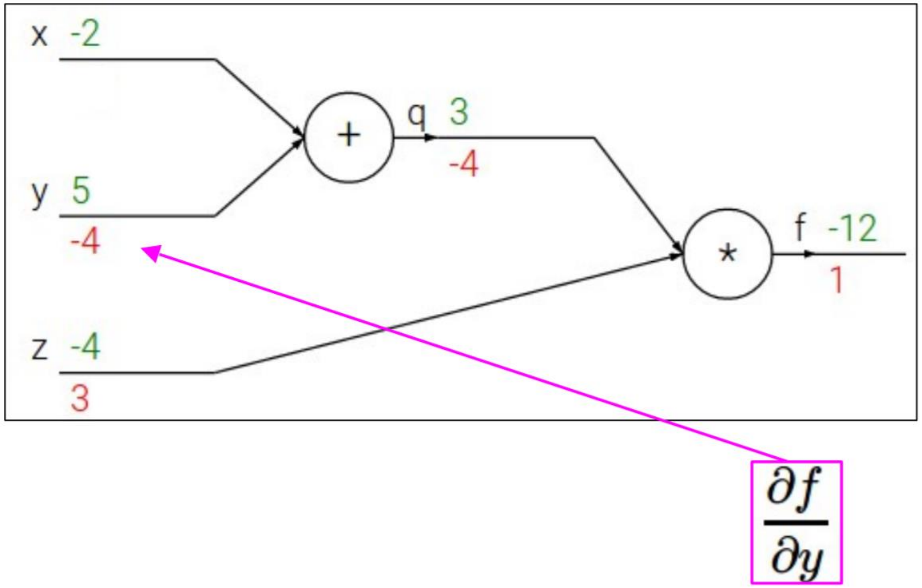
$$f(x,y,z) = (x+y)z$$

e.g. x = -2, y = 5, z = -4

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



Example of computations (11/13)

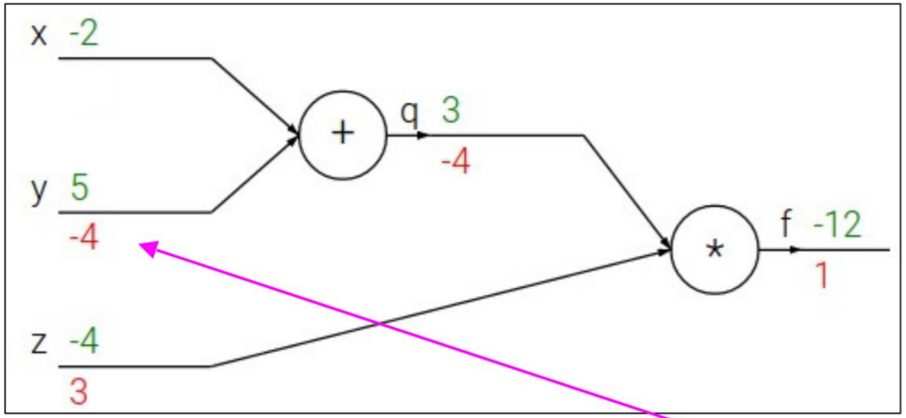
$$f(x, y, z) = (x + y)z$$

e.g. x = -2, y = 5, z = -4

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



$\frac{\partial f}{\partial y}$

Chain rule:

$$\frac{\partial f}{\partial y} = \frac{\partial f}{\partial q} \frac{\partial q}{\partial y}$$

51

Example of computations (12/13)

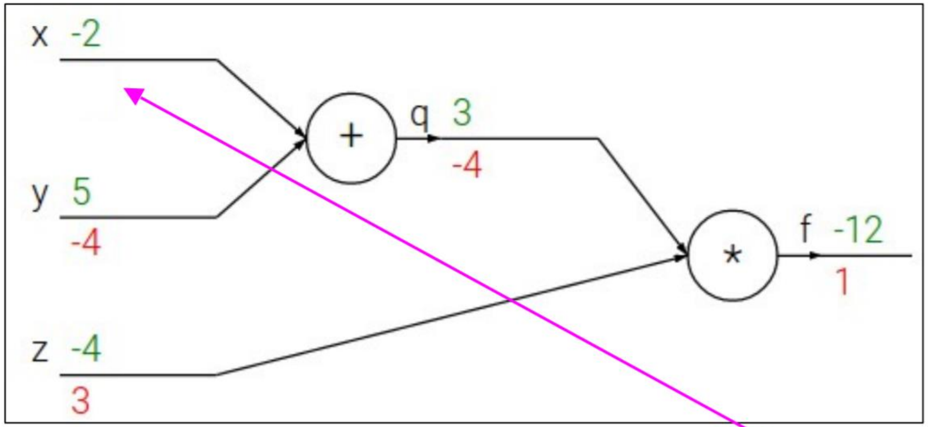
$$f(x, y, z) = (x + y)z$$

e.g. x = -2, y = 5, z = -4

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



$$\boxed{\frac{\partial f}{\partial x}}$$

52

Example of computations (13/13)

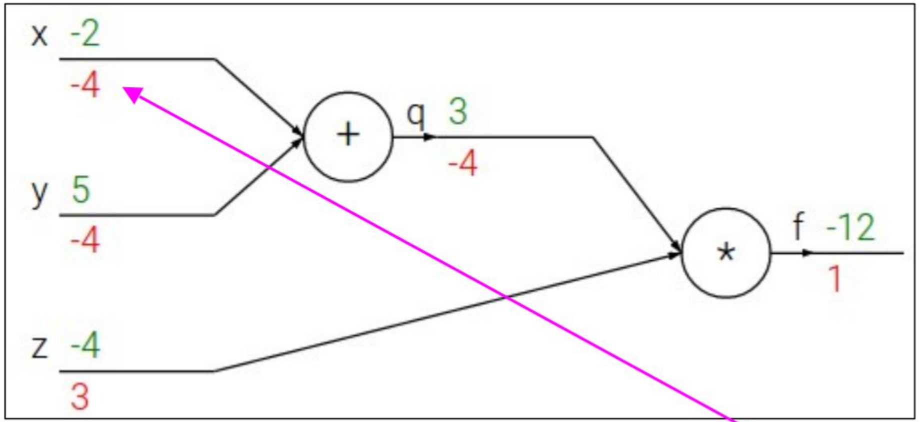
$$f(x,y,z) = (x+y)z$$

e.g. x = -2, y = 5, z = -4

$$\boxed{q = x + y \qquad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1}$$

$$\boxed{f = qz \qquad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q}$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



$$\boxed{\frac{\partial f}{\partial x}}$$

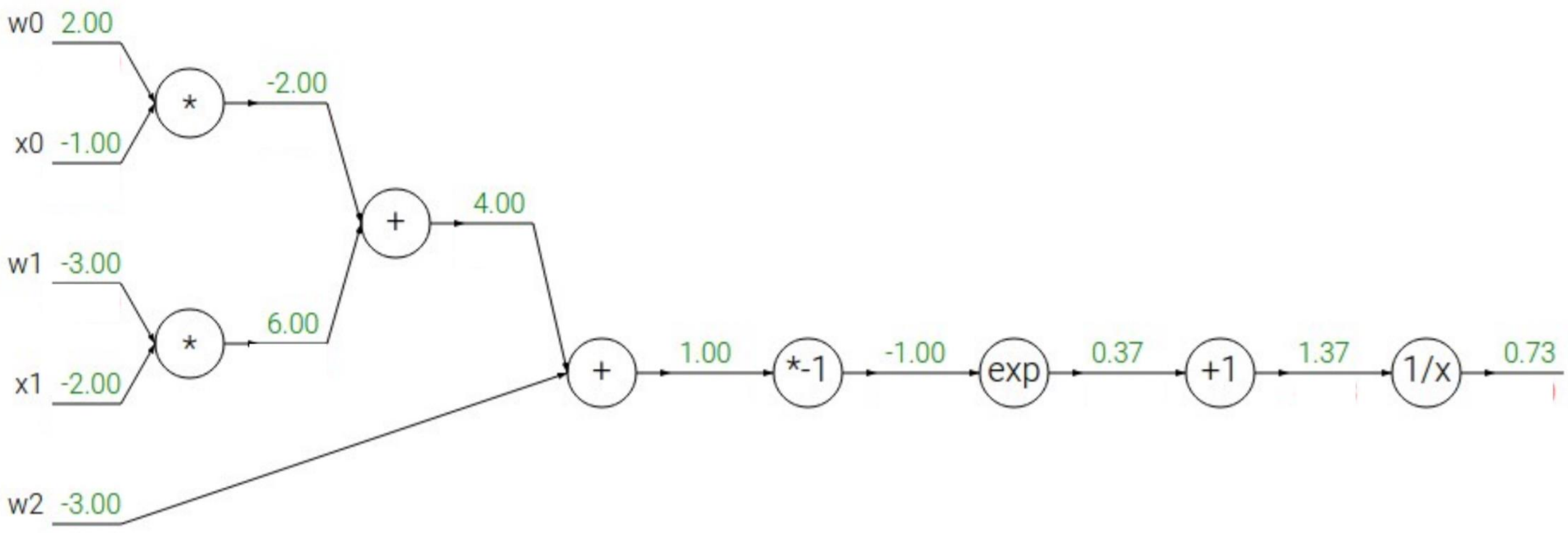
Chain rule:

$$\boxed{\frac{\partial f}{\partial x} = \frac{\partial f}{\partial q}\frac{\partial q}{\partial x}}$$

53

Another example (1/2)

$$f(w,x) = \frac{1}{1+e^{-(w_0x_0+w_1x_1+w_2)}}$$

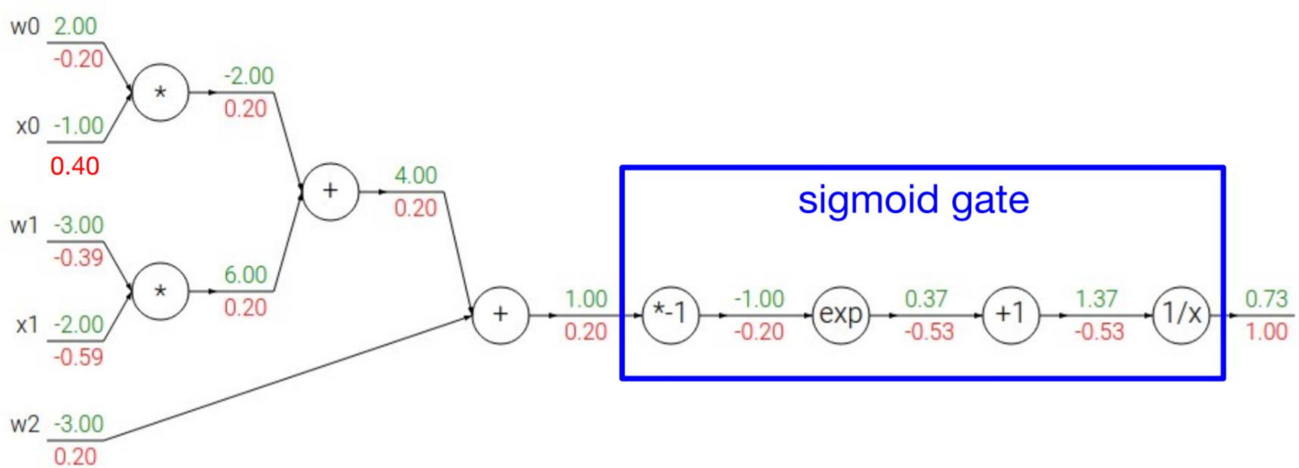


Another example (2/2)

$$f(w,x) = \frac{1}{1+e^{-(w_0x_0+w_1x_1+w_2)}}$$

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

$$\frac{d\sigma(x)}{dx} = \frac{e^{-x}}{(1+e^{-x})^2} = \left(\frac{1+e^{-x}-1}{1+e^{-x}}\right)\left(\frac{1}{1+e^{-x}}\right) = (1-\sigma(x))\,\sigma(x)$$



55

Lecture plan

- History of neural networks
- Completeness problem of neural networks
- Single layer neural network
- Multilayer neural networks
- Backpropagation
- Modularity
- Computational graph
- DNN best practices

The problems

- A deep network has many parameters, which cause overfitting
- A deep network is trying to reveal deep features, which requires a lot of data (and causes overfitting)
- It has many modules in a raw, which cases vanishing gradient problem (the far a module is from loss, the less it learns)

57

Best practices

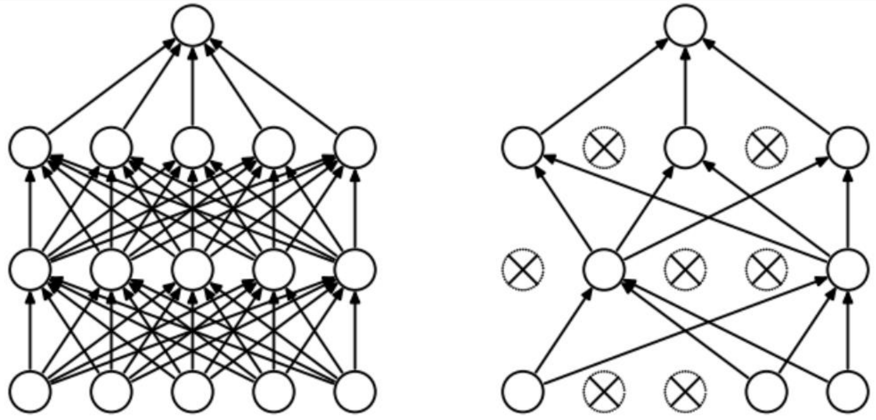
- A deep network has many parameters, use **dropout**
- A deep network is trying to reveal deep features, use **augmented data**
- It has many modules in a raw, which cases the vanishing gradient problem, use
  - data preprocessing (next lecture)
  - special activation functions (next lecture)
  - optimization tricks (next lecture)

58

Dropout (1/2)

Dropout: set the output of each hidden neuron to zero w.p. 0.5

- The neurons which are “dropped out” in this way do not contribute to the forward pass and do not participate in backpropagation
- So every time an input is presented, the neural network samples a different architecture, but all these architectures share weights

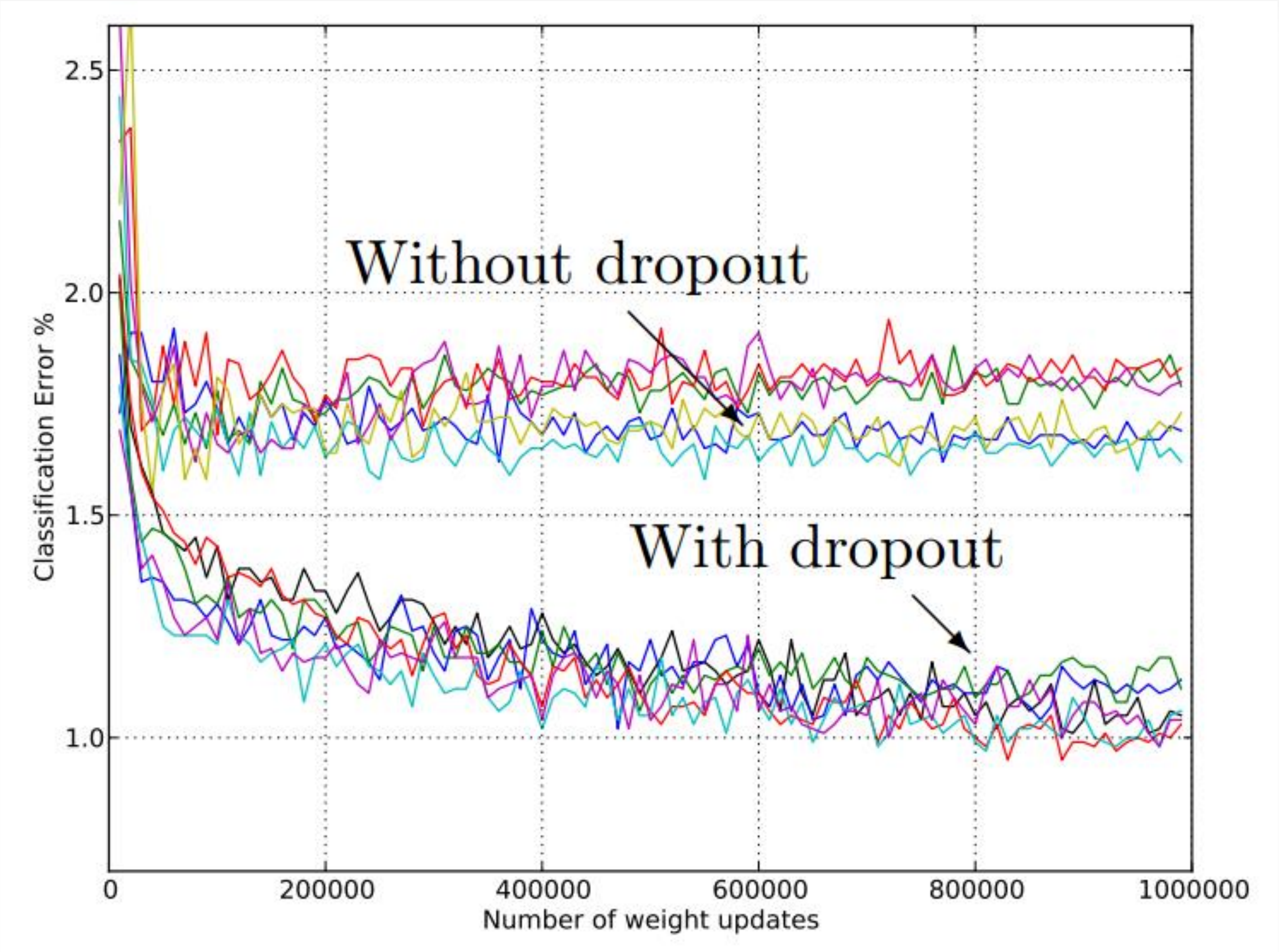


(a) Standard Neural Net (b) After applying dropout.

Dropout (2/2)

- This technique reduces complex co-adaptations of neurons, since a neuron cannot rely on the presence of particular other neurons
- It is forced to learn more robust features that are useful in conjunction with many different random subsets of the other neurons
- Without dropout, a network exhibits substantial overfitting
- Dropout roughly doubles the number of iterations required to converge

Dropout effect



61

Data augmentation

The easiest and most common method to **reduce overfitting** on image data is to artificially **enlarge the dataset** using label-preserving transformations.

Types of data augmentation:

- Image translation
- Horizontal/vertical reflections + cropping
- Changing RGB intensities

62