Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

HANDSCHRIFTERKENNUNG MITTELS NEURONALER NETZE MIT BACKPROPAGATION

Fachseminar "Machine Learning"

Letztes Update: 12. Januar 2016

Nadja Kurz

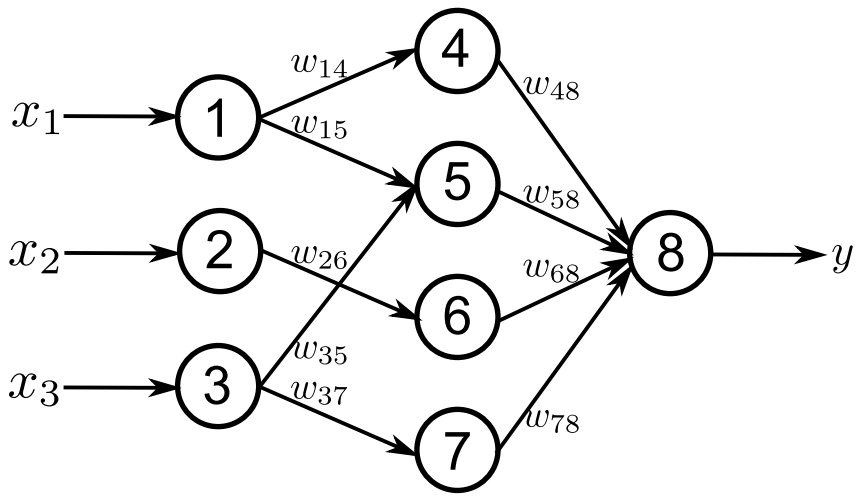
Studienbereich Informatik
Hochschule RheinMain

GLIEDERUNG

NEURONALE NETZWERKE

NEURONALE NETZWERKE

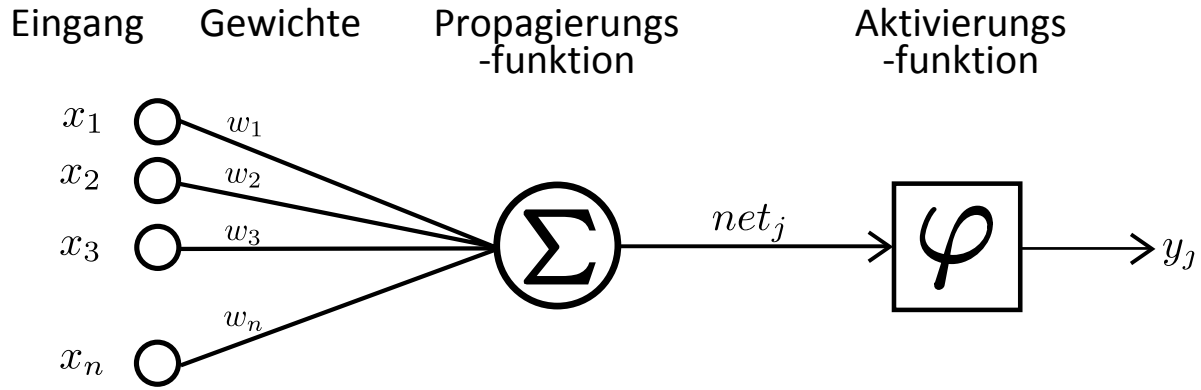
- Idee: Abbildung der menschlichen Gehirnfunktionalität auf technische Systeme.
- Prinzip: Systeme trainieren, mit einer Eingabe eine bestimmte Ausgabe zu erzielen.
  - Neuronales Netz als gewichteter, gerichteter Graph mit min. zwei Schichten (Ein- und Ausgabeschicht)



  - Neuronen als Verarbeitungseinheiten mit einem Zustand
  - Gewichtete Verbindungen zwischen den Neuronen
  - Lernregel, um Verbindungsgewichte zu modifizieren

NEURON

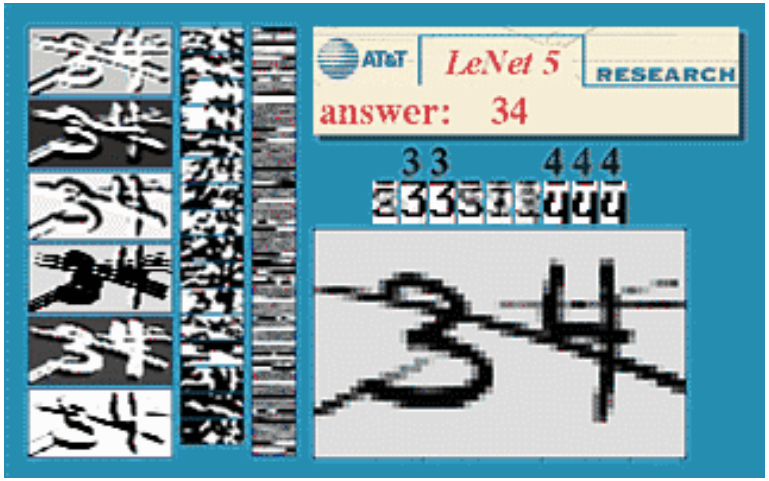
- Idee: Modellierung des biologischen Vorbildes "Daten empfangen, verarbeiten und weiterleiten"



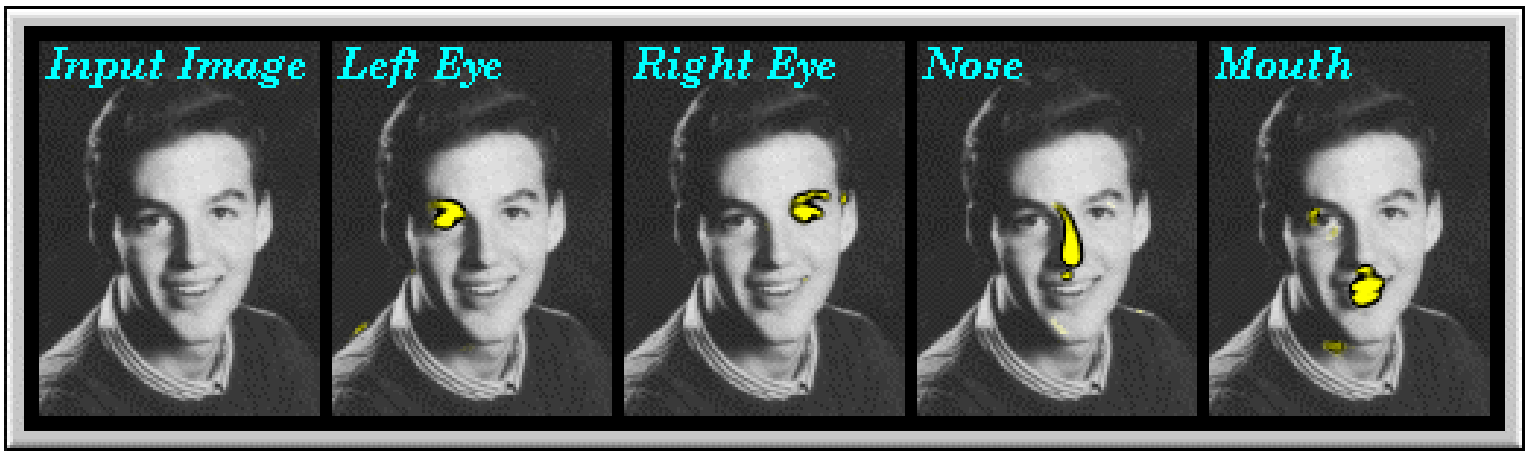
  - Eingangssignale $x_1, x_2, ..., x_n$ verbundener Neuronen der vorigen Schicht
  - Verbindungsgewichte $w_1, w_2, ..., w_n$
  - Propagierungsfunktion $\sum$ erzeugt Netzeingabe net_j
  - Aktivierungsfunktion $\varphi(net_j)$ erzeugt neuen Aktivierungszustand y_j
  - Aktivierungszustand y_j (=Eingangssignal für verbundene Neuronen der nächsten Schicht)

ANWENDUNGEN

- Handschrifterkennung
- Mustererkennung
- Spracherkennung
- Sprachverstehen
- Gesichtsmerkmalerkennung



Source: http://yann.lecun.com/exdb/lenet/



Source: http://ict.debevec.org/ debevec/FaceRecognition/

XOR-PROBLEM (BEISPIEL)

Beispiel für ein neuronales Netz für die XOR-Funktion.

XOR-Problem Bsp.

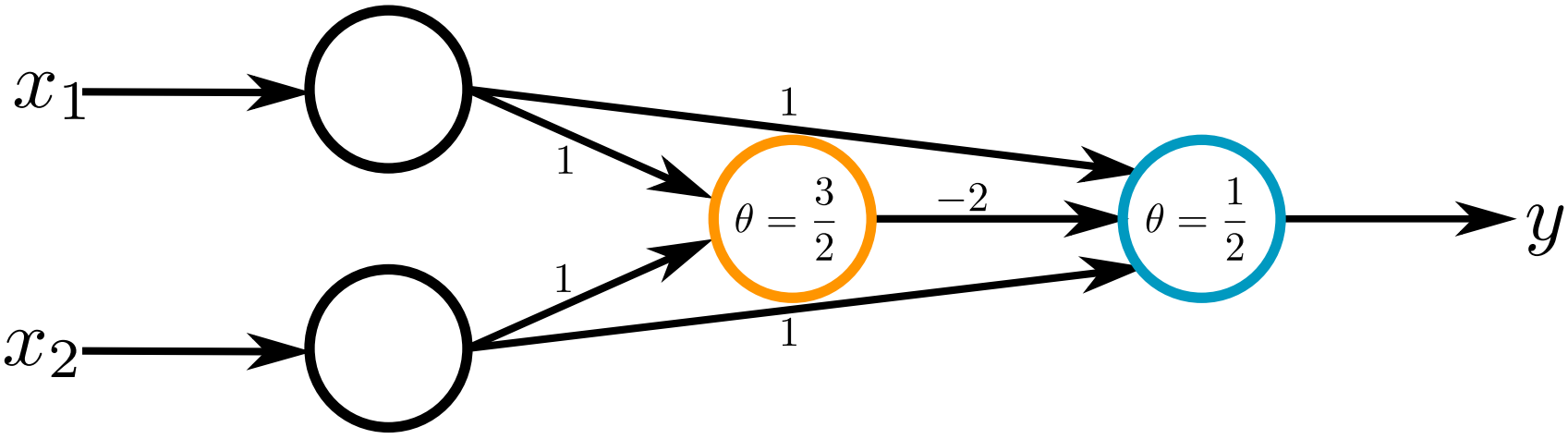
Wir definieren für das Bsp. die Netzeingabe als gewichtete Summe der Eingaben

$$net_j := \sum_{i=1} w_{ij} x_i$$

und die Aktivierungsfunktion als eine Treppenfunktion

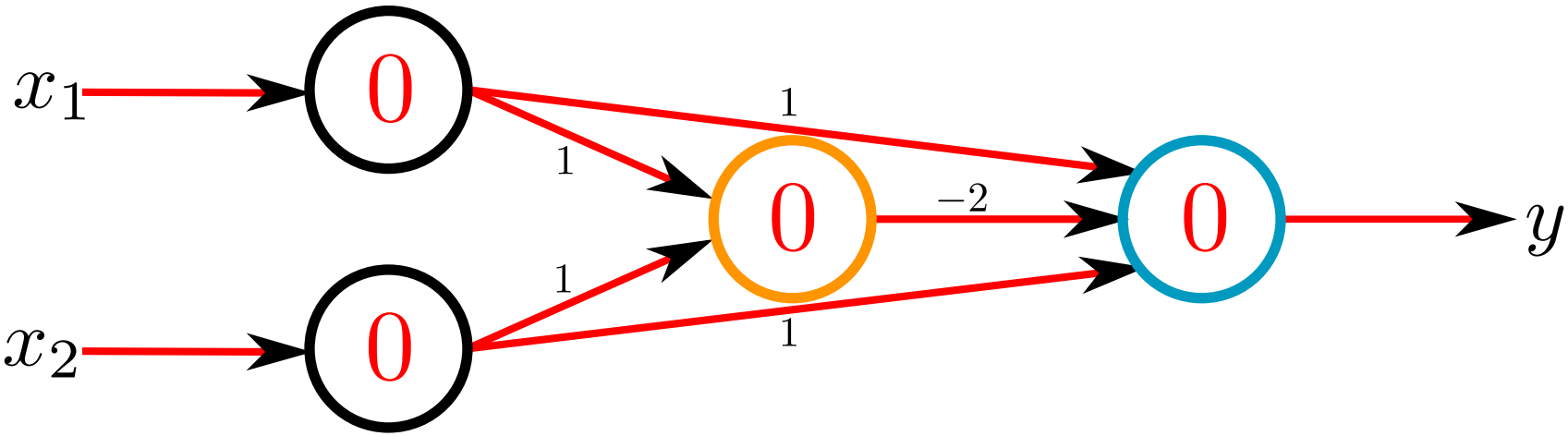
$$\varphi(net_j) := \begin{cases} 1 & \text{, falls } net_j \geq \theta \\ 0 & \text{, sonst} \end{cases}$$

XOR-PROBLEM (BEISPIEL)



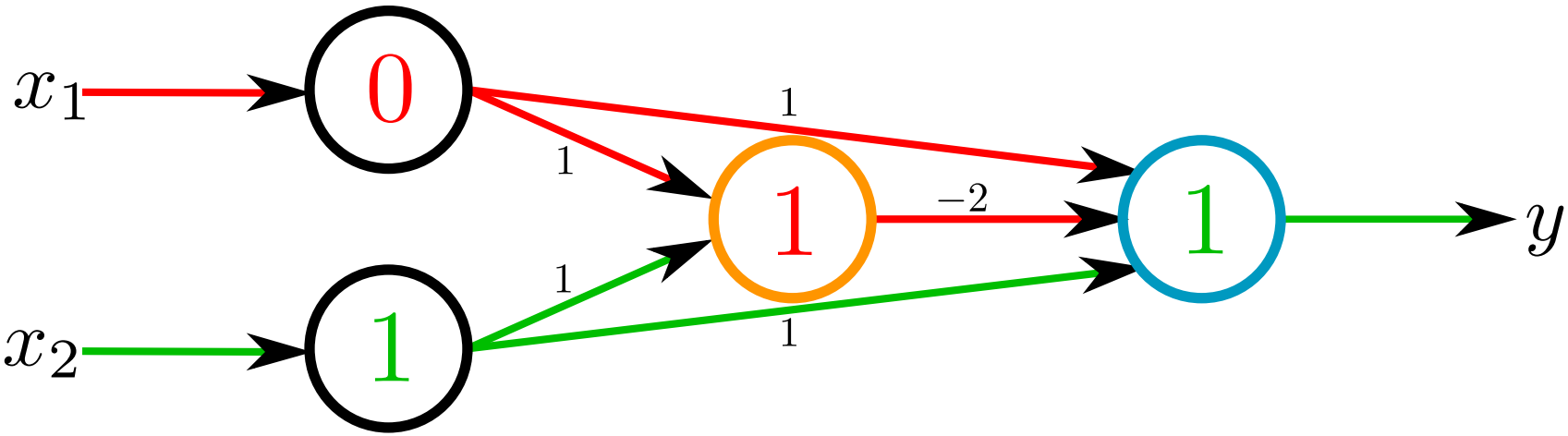
x_1	x_2	net_3	$y_3 = \varphi(net_3)$	net_4	$y_4 = \varphi(net_4)$
0	0				
0	1				
1	0				
1	1				

XOR-PROBLEM (BEISPIEL)



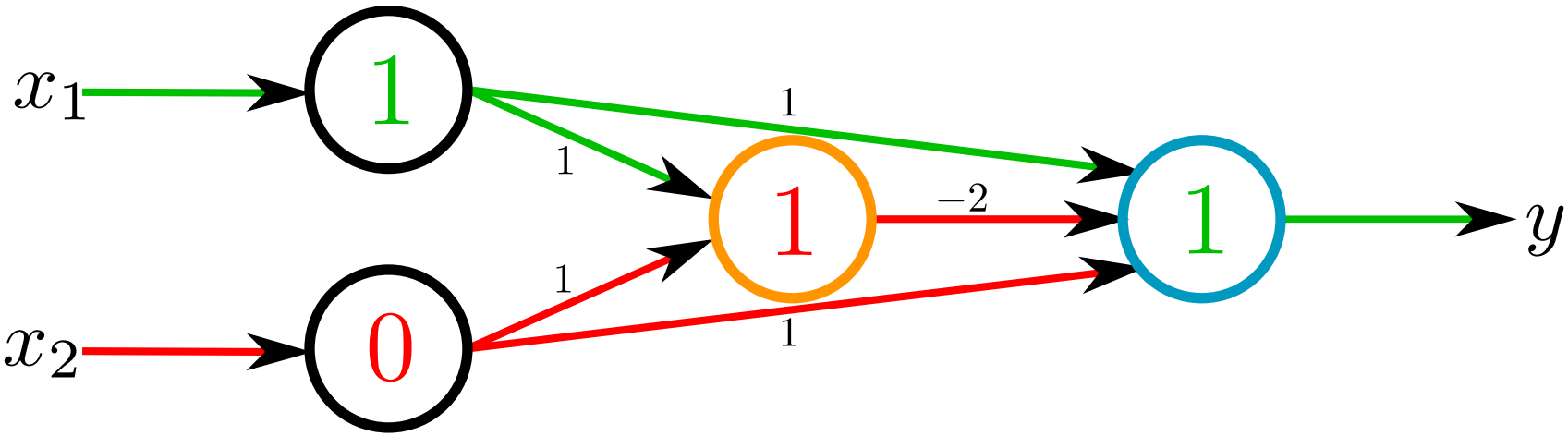
x_1	x_2	net_3	$y_3 = \varphi(net_3)$	net_4	$y_4 = \varphi(net_4)$
0	0	$0 \cdot 1 + 0 \cdot 1 = 0$	0	$0 \cdot 1 + 0 \cdot 1 + 0 \cdot (-2) = 0$	0
0	1				
1	0				
1	1				

XOR-PROBLEM (BEISPIEL)



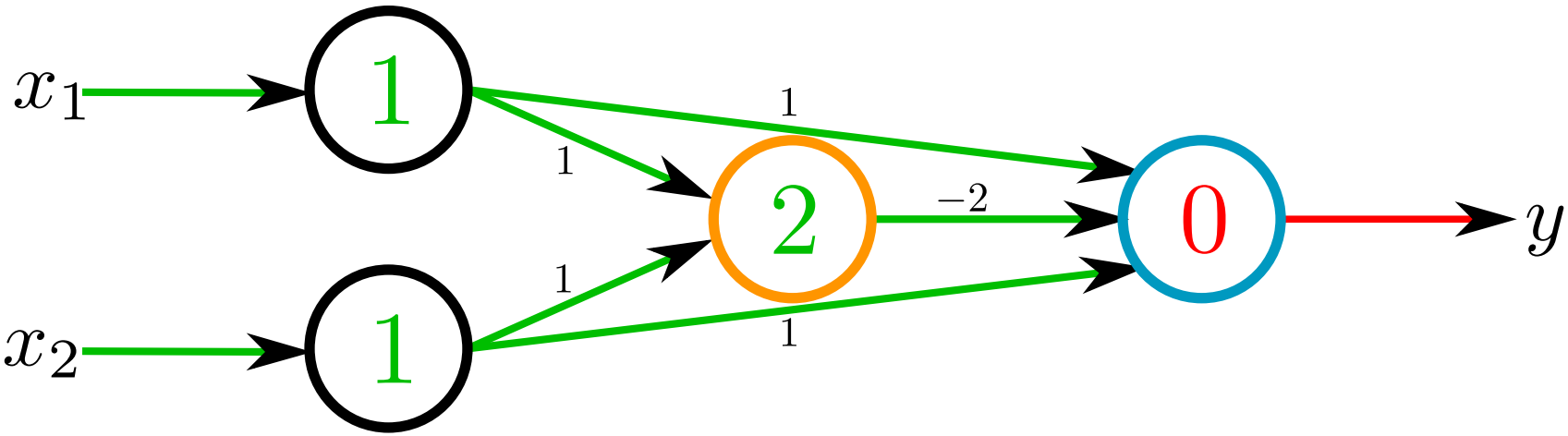
x_1	x_2	net_3	$y_3 = \varphi(net_3)$	net_4	$y_4 = \varphi(net_4)$
0	0	$0 \cdot 1 + 0 \cdot 1 = 0$	0	$0 \cdot 1 + 0 \cdot 1 + 0 \cdot (-2) = 0$	0
0	1	$0 \cdot 1 + 1 \cdot 1 = 1$	0	$0 \cdot 1 + 1 \cdot 1 + 0 \cdot (-2) = 1$	1
1	0				
1	1				

XOR-PROBLEM (BEISPIEL)



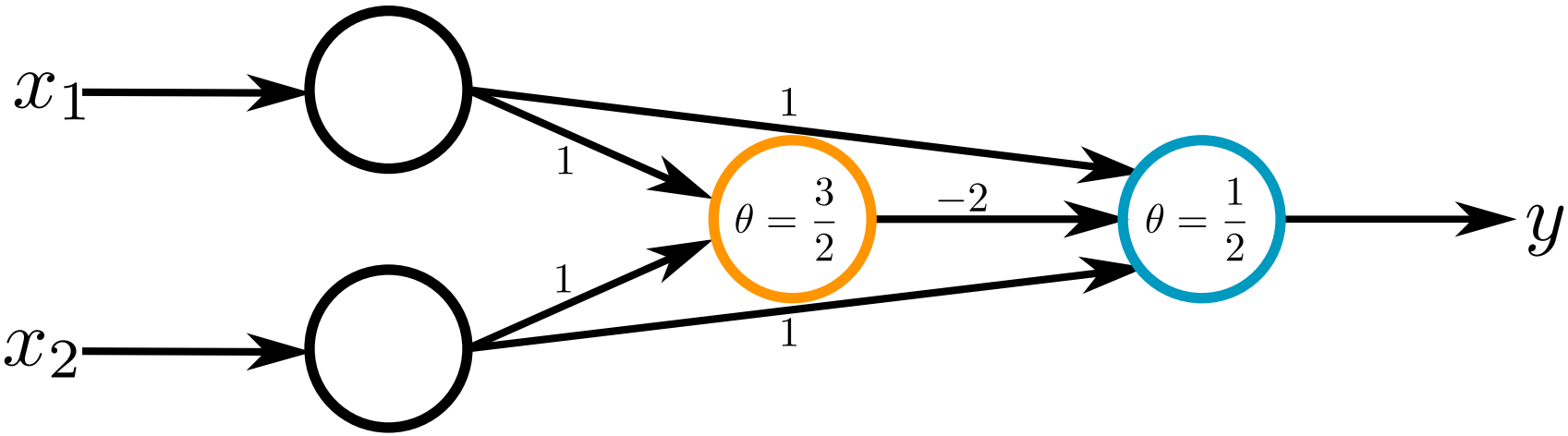
x_1	x_2	net_3	$y_3 = \varphi(net_3)$	net_4	$y_4 = \varphi(net_4)$
0	0	$0 \cdot 1 + 0 \cdot 1 = 0$	0	$0 \cdot 1 + 0 \cdot 1 + 0 \cdot (-2) = 0$	0
0	1	$0 \cdot 1 + 1 \cdot 1 = 1$	0	$0 \cdot 1 + 1 \cdot 1 + 0 \cdot (-2) = 1$	1
1	0	$1 \cdot 1 + 1 \cdot 0 = 1$	0	$1 \cdot 1 + 0 \cdot 1 + 0 \cdot (-2) = 1$	1
1	1				

XOR-PROBLEM (BEISPIEL)



x_1	x_2	net_3	$y_3 = \varphi(net_3)$	net_4	$y_4 = \varphi(net_4)$
0	0	$0 \cdot 1 + 0 \cdot 1 = 0$	0	$0 \cdot 1 + 0 \cdot 1 + 0 \cdot (-2) = 0$	0
0	1	$0 \cdot 1 + 1 \cdot 1 = 1$	0	$0 \cdot 1 + 1 \cdot 1 + 0 \cdot (-2) = 1$	1
1	0	$1 \cdot 1 + 1 \cdot 0 = 1$	0	$1 \cdot 1 + 0 \cdot 1 + 0 \cdot (-2) = 1$	1
1	1	$1 \cdot 1 + 1 \cdot 1 = 2$	1	$1 \cdot 1 + 1 \cdot 1 + 1 \cdot (-2) = 0$	0

XOR-PROBLEM (BEISPIEL)



- XOR-Funktion als neuronales Netz mit mehr als nur Ein- und Ausgabeschicht
- Keine Lösung mit nur zwei Schichten vorhanden
- Weitere Schichten (="Hidden Layer") mit "Hidden Units"

BACKPROPAGATION

BACKPROPAGATION

Backpropagation als Verfahren für mehrschichtige Netzwerke

- Idee: Anpassung der Gewichte durch Vergleich des tatsächlichen Wertes und eines erwarteten Wertes
- Ziel: Minimierung des Fehlers der beiden Werte mittels einer Fehlerfunktion E (vgl. Gradientenabstiegsverfahrens)
- Problem: Für hidden units sind keine gewünschten Ergebnisse bekannt.
  - Wie also Fehler bestimmen?
- Lösung: "Backpropagation" (= Rück-Übertragung)
  - Fehlerberechnung erfolgt von hinten nach vorne (von den Ausgabeneuronen zu den Eingabeneuronen).

LERNALGORITHMUS

Unterteilung des Lernalgorithmus in drei aufeinanderfolgende Phasen

1. Phase: Forward-Pass
   - Bestimmung der Ausgabe des neuronalen Netzes
2. Phase: Fehlerbestimmung
   - Vergleich von Ausgaben aus dem Forward-Pass mit erwarteten Ausgaben
3. Phase: Backward-Pass
   - Verbreitung der Fehlerterme von hinten nach vorne
   - Mit Gewichtsmodifikation

FORWARD-PASS

Netzeingabe

Wir definieren für ein Neuron j die Netzeingabe

$$net_j := \sum_{i=1} w_{ij} x_i$$

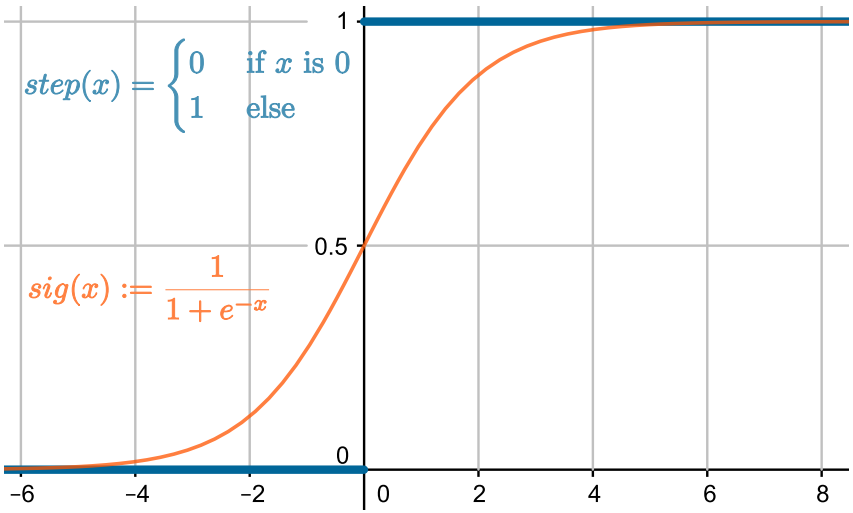
mit w_{ij} als Gewicht zwischen einem Neuron i und einem Neuron j, sowie x_i als Ausgabe eines Neurons i.

Aktivierungsfunktion

Wir definieren für ein Neuron j die Aktivierungsfunktion

$$\varphi(net_j) = \frac{1}{1 + e^{-net_j}}$$

FORWARD-PASS



Ausgabe

Wir definieren für ein Neuron j die Ausgabe

$$y_j := \varphi(net_j) = \frac{1}{1 + e^{-net_j}}$$

FEHLERBESTIMMUNG

Fehlerfunktion

Wir definieren die Fehlerfunktion

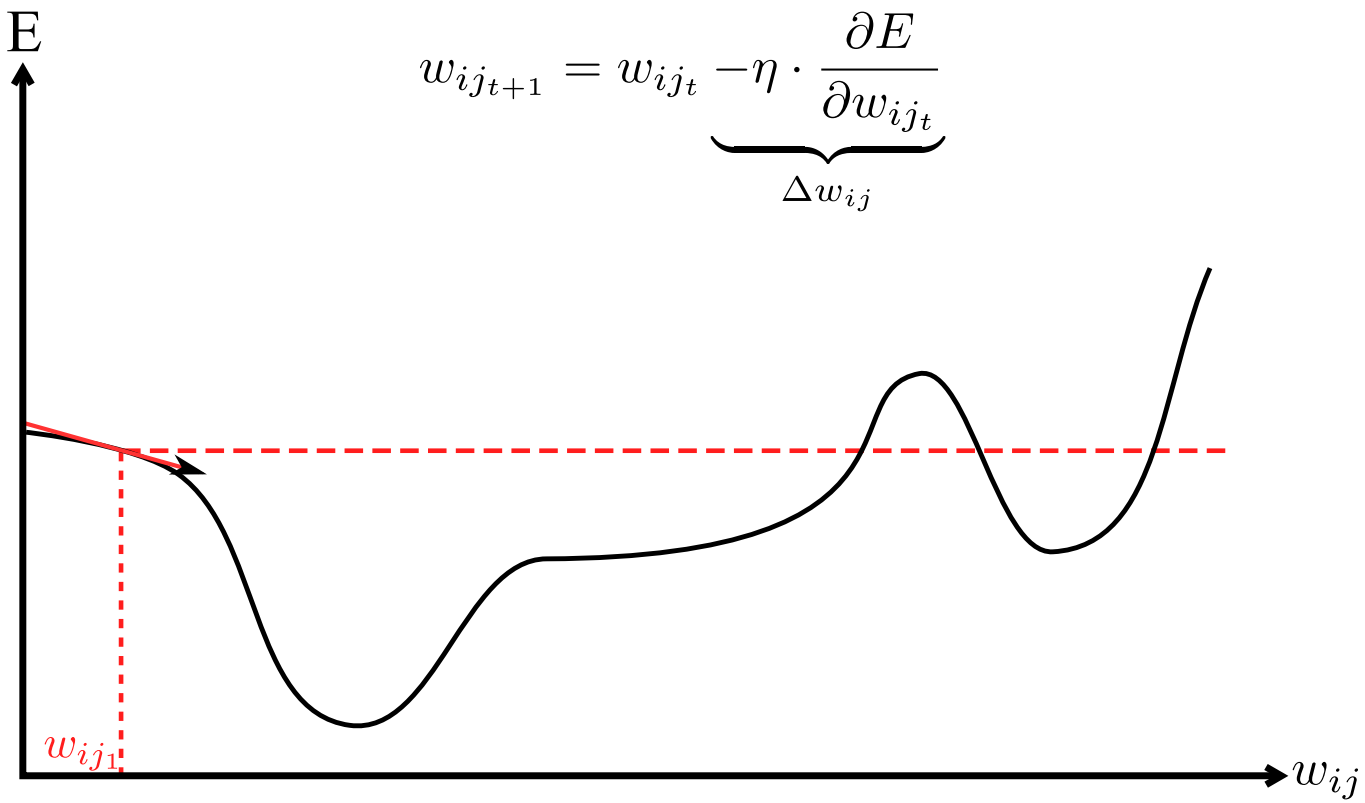
$$E := \frac{1}{2} \sum_j (t_j - y_j)^2$$

mit t_j als erwartete und y_j als tatsächliche Ausgabe.

GRADIENTENABSTIEGSVERFAHREN

- Ziel: Minimierung der Fehlerfunktion
- Lösung: Gradientenabstiegsverfahren

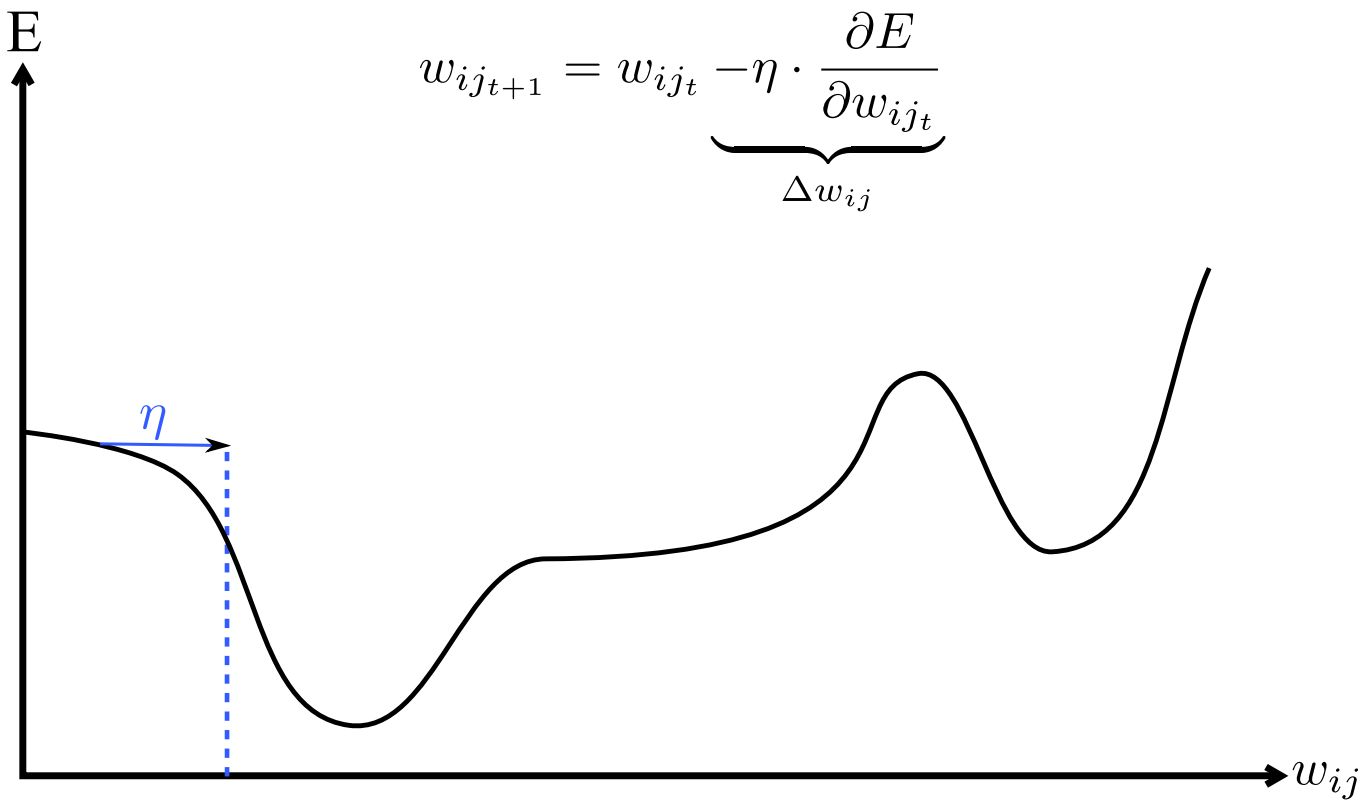
$$w_{ij_{t+1}} = w_{ij_t} \underbrace{-\eta \cdot \frac{\partial E}{\partial w_{ij_t}}}_{\Delta w_{ij}}$$



GRADIENTENABSTIEGSVERFAHREN

- Ziel: Minimierung der Fehlerfunktion
- Lösung: Gradientenabstiegsverfahren

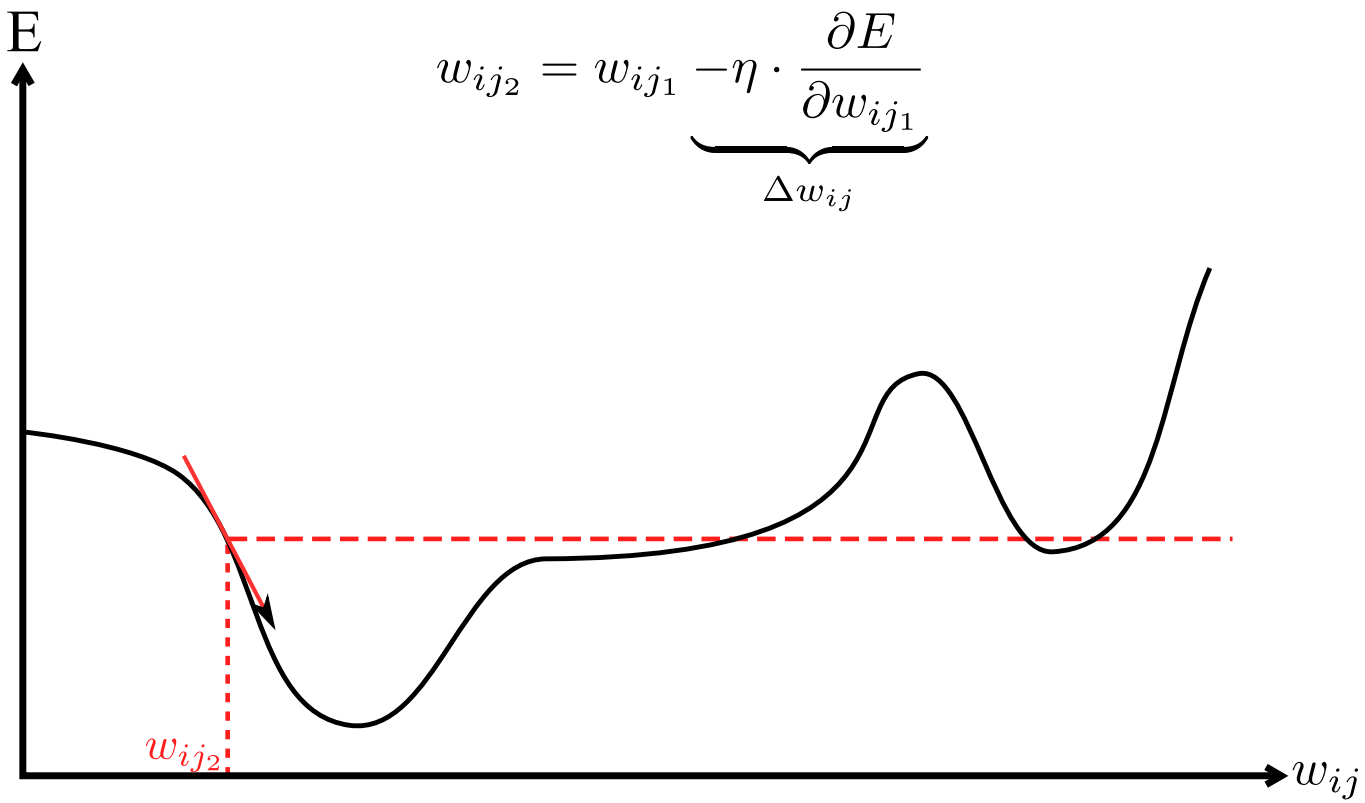
$$w_{ij_{t+1}} = w_{ij_t} \underbrace{-\eta \cdot \frac{\partial E}{\partial w_{ij_t}}}_{\Delta w_{ij}}$$



GRADIENTENABSTIEGSVERFAHREN

- Ziel: Minimierung der Fehlerfunktion
- Lösung: Gradientenabstiegsverfahren

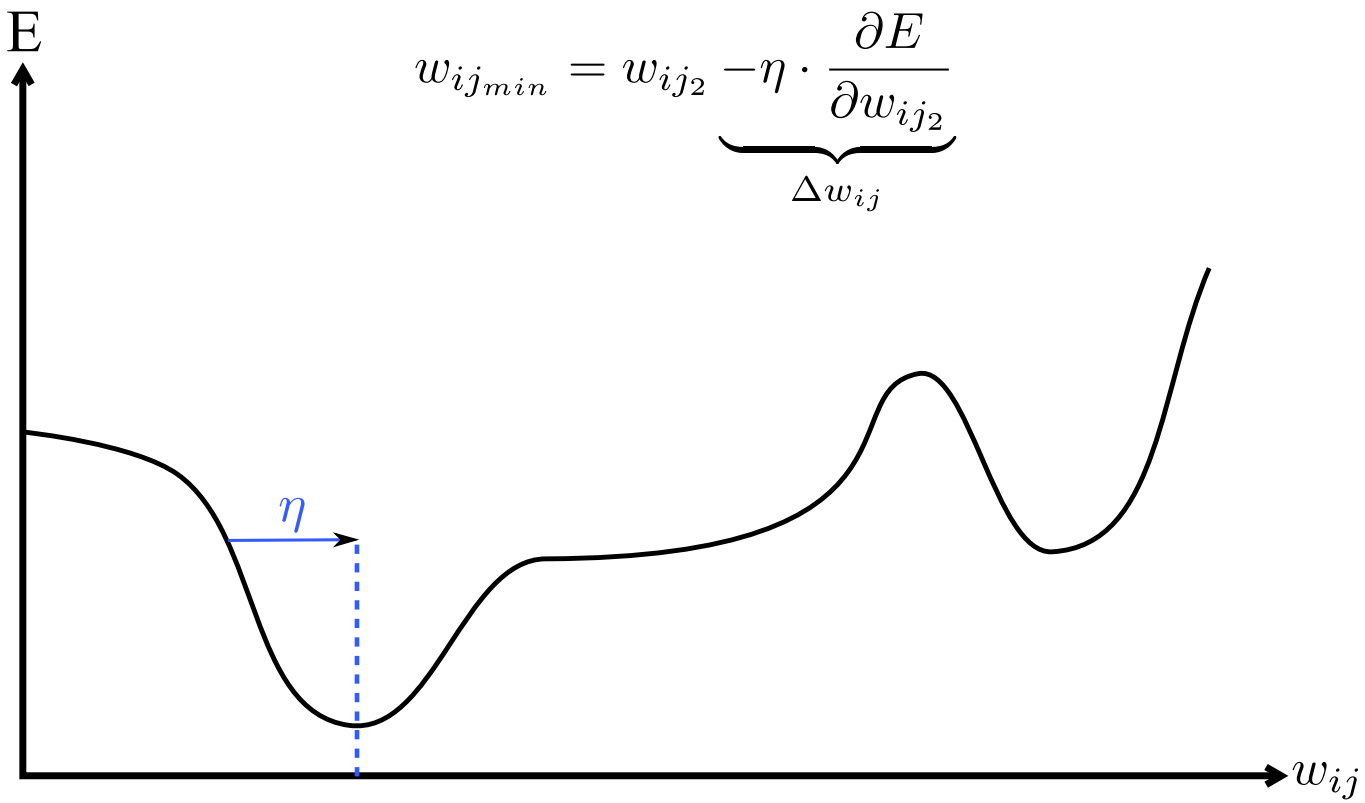
$$w_{ij_2} = w_{ij_1} \underbrace{-\eta \cdot \frac{\partial E}{\partial w_{ij_1}}}_{\Delta w_{ij}}$$



GRADIENTENABSTIEGSVERFAHREN

- Ziel: Minimierung der Fehlerfunktion
- Lösung: Gradientenabstiegsverfahren

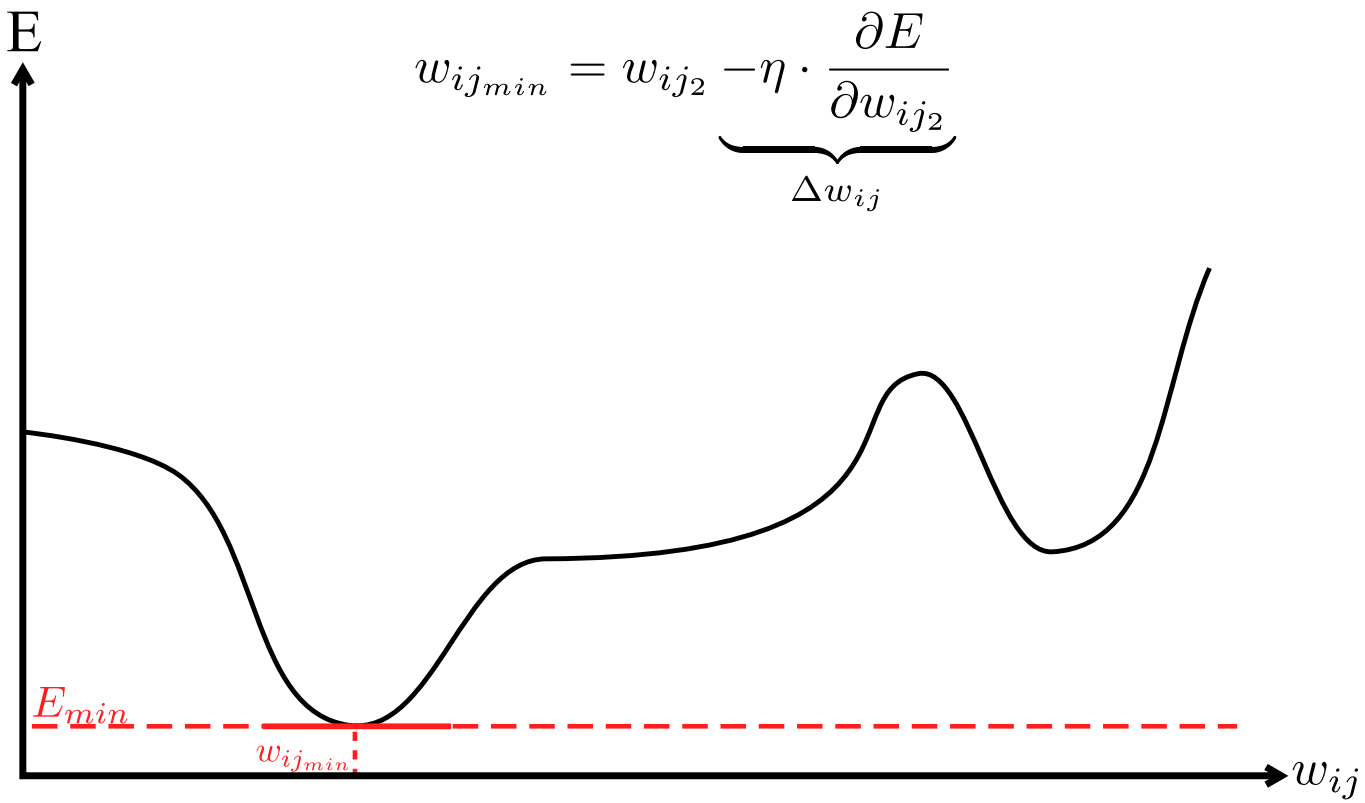
$$w_{ij_{min}} = w_{ij_2} \underbrace{-\eta \cdot \frac{\partial E}{\partial w_{ij_2}}}_{\Delta w_{ij}}$$



GRADIENTENABSTIEGSVERFAHREN

- Ziel: Minimierung der Fehlerfunktion
- Lösung: Gradientenabstiegsverfahren

$$w_{ij_{min}} = w_{ij_2} \underbrace{-\eta \cdot \frac{\partial E}{\partial w_{ij_2}}}_{\Delta w_{ij}}$$



BACKWARD-PASS

Für die Ausgabeschicht gilt

$$\Delta w_{jk} = \eta \delta_k y_j$$

mit j für ein Neuron der inneren Schicht und k für ein Neuron der Ausgabeschicht, sowie

$$\delta_k = \varphi'(net_k)(t_k - y_k)$$

mit t_k als gewünschten und y_k als tatsächlichen Ausgabewert. Das neue Gewicht ergibt sich dann aus

$$w_{jk}^{neu} = w_{jk}^{alt} + \Delta w_{jk}$$

BACKWARD-PASS

Für die inneren Schichten gilt

$$\Delta w_{ij} = \eta \delta_j y_i$$

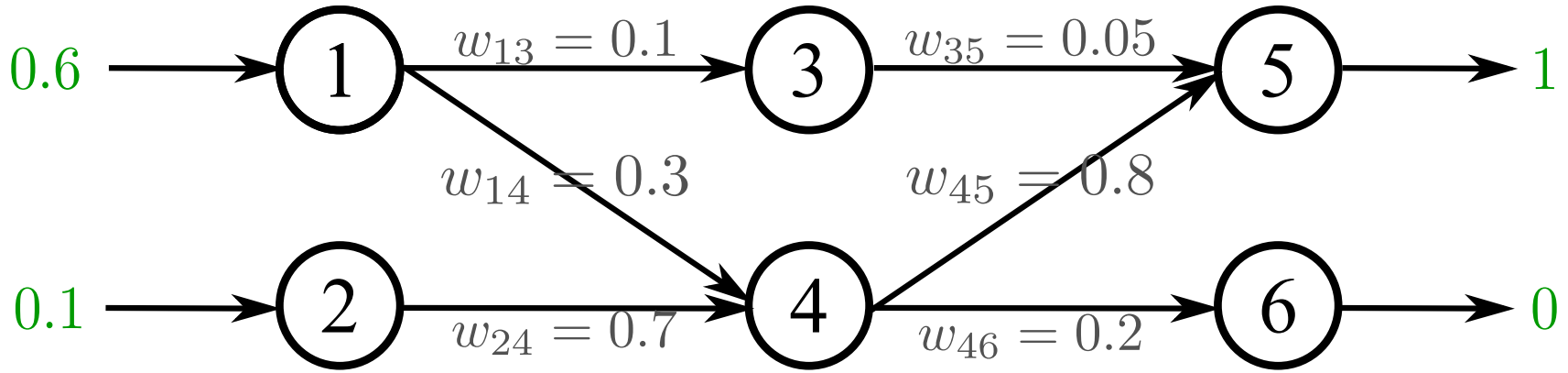
mit i und j für ein Neuron einer inneren oder der Eingabeschicht, sowie

$$\delta_j = \varphi'(net_j) \sum_k \delta_k w_{jk}$$

mit k für ein Neuron einer hinteren Schicht. Das neue Gewicht ergibt sich dann aus

$$w_{ij}^{neu} = w_{ij}^{alt} + \Delta w_{ij}$$

BACKPROPAGATION (BEISPIEL)

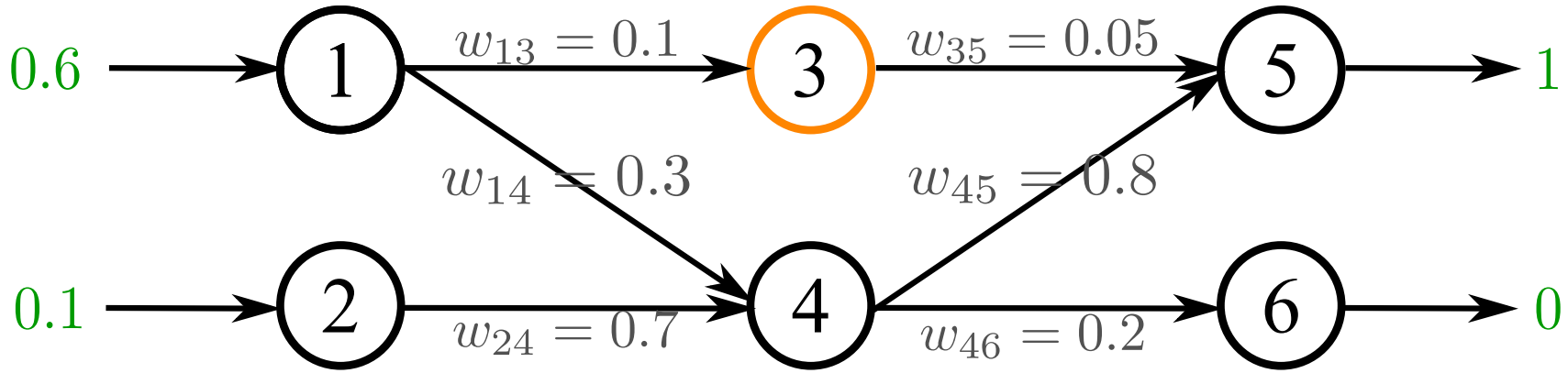


$w_{13} = 0.1$ $w_{35} = 0.05$ $\eta = \frac{1}{2}$

$w_{14} = 0.3$ $w_{45} = 0.8$ $x_1 = 0.6$ $t_5 = 1$

$w_{24} = 0.7$ $w_{46} = 0.2$ $x_2 = 0.1$ $t_6 = 0$

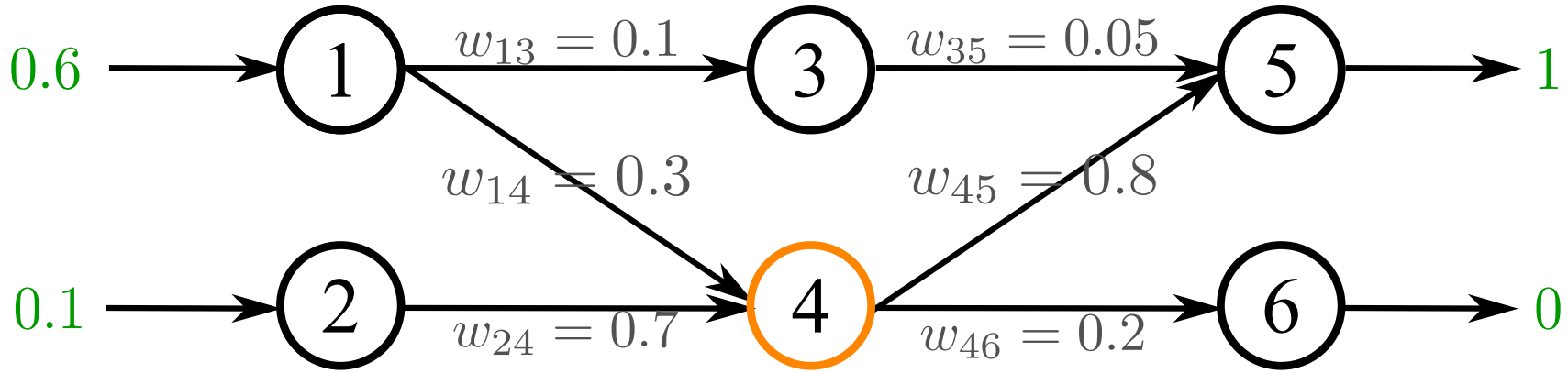
BACKPROPAGATION (BEISPIEL)



Forward Pass

$net_3 = x_1 \cdot w_{13} = 0.6 \cdot 0.1 = 0.06 \quad y_3 = \varphi(net_3) = 0.515$

BACKPROPAGATION (BEISPIEL)

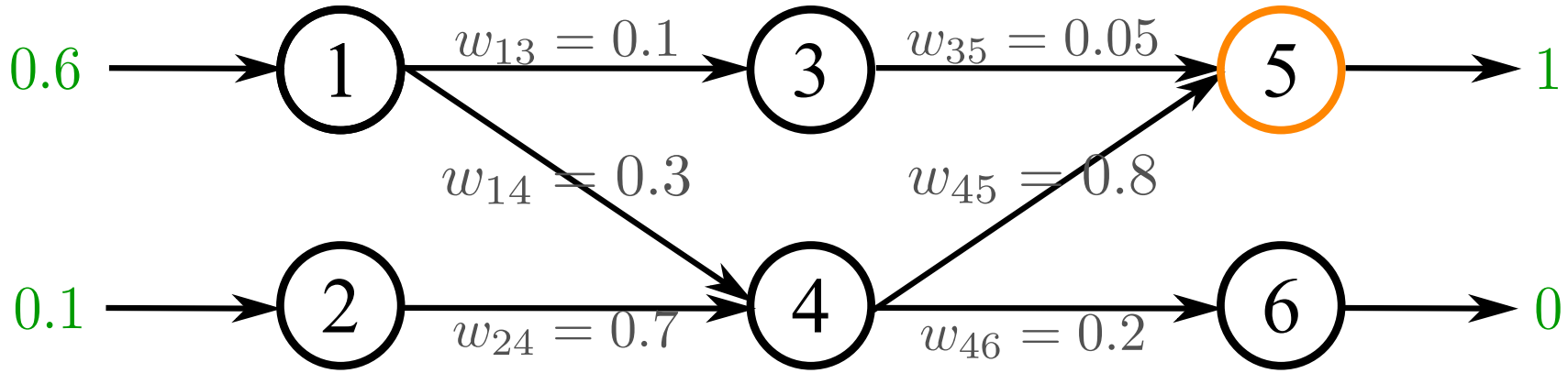


Forward Pass

$net_3 = x_1 \cdot w_{13} = 0.6 \cdot 0.1 = 0.06 \quad y_3 = \varphi(net_3) = 0.515$

$net_4 = x_1 \cdot w_{14} + x_2 \cdot w_{24} = 0.25 \quad y_4 = \varphi(net_4) = 0.562$

BACKPROPAGATION (BEISPIEL)



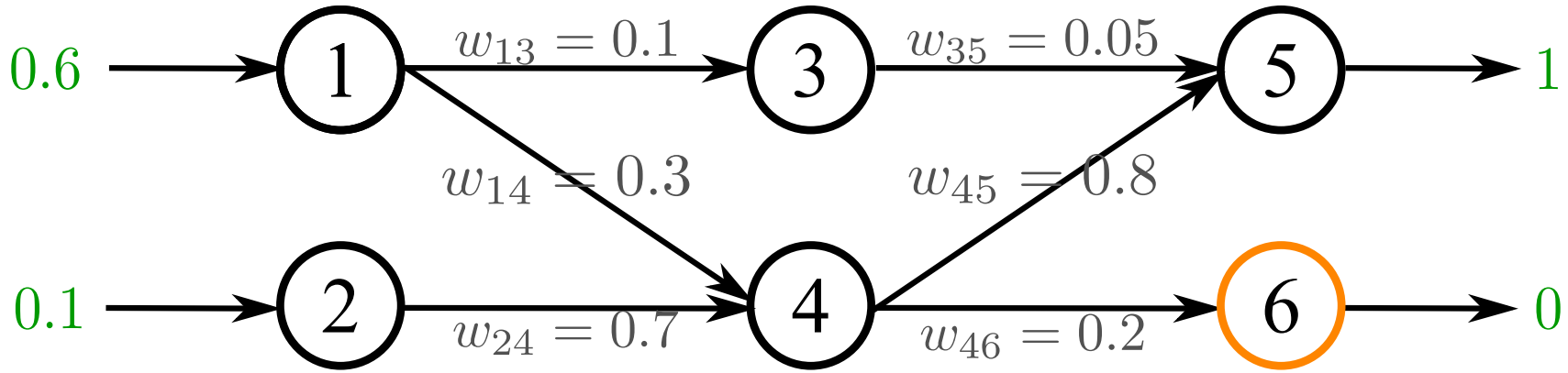
Forward Pass

$net_3 = x_1 \cdot w_{13} = 0.6 \cdot 0.1 = 0.06$ $\quad y_3 = \varphi(net_3) = 0.515$

$net_4 = x_1 \cdot w_{14} + x_2 \cdot w_{24} = 0.25$ $\quad y_4 = \varphi(net_4) = 0.562$

$net_5 = y_3 \cdot w_{35} + y_4 \cdot w_{45} = 0.476$ $\quad y_5 = \varphi(net_5) = 0.617$

BACKPROPAGATION (BEISPIEL)



Forward Pass

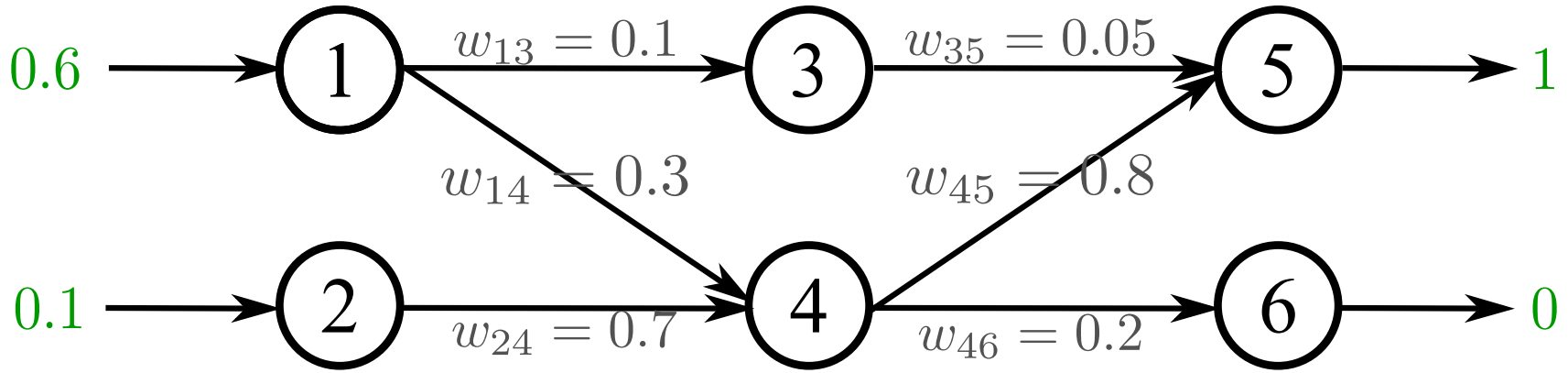
$net_3 = x_1 \cdot w_{13} = 0.6 \cdot 0.1 = 0.06$ $\quad y_3 = \varphi(net_3) = 0.515$

$net_4 = x_1 \cdot w_{14} + x_2 \cdot w_{24} = 0.25$ $\quad y_4 = \varphi(net_4) = 0.562$

$net_5 = y_3 \cdot w_{35} + y_4 \cdot w_{45} = 0.476$ $\quad y_5 = \varphi(net_5) = 0.617$

$net_6 = y_4 \cdot w_{46} = 0.112$ $\quad y_6 = \varphi(net_6) = 0.528$

BACKPROPAGATION (BEISPIEL)



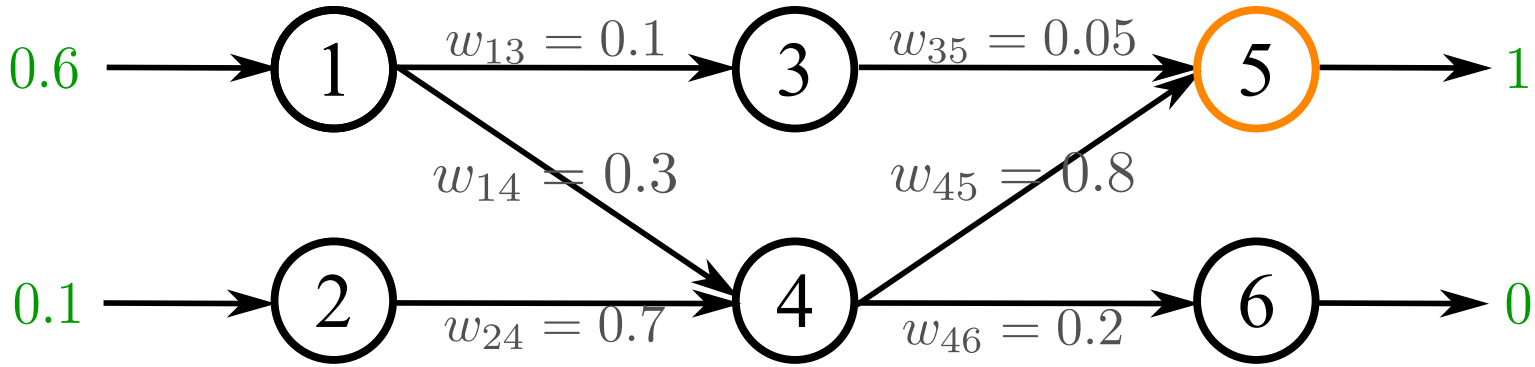
Fehlerbestimmung

$E_5 = (t_5 - y_5)^2 = (1.0 - 0.617)^2 = 0.147$

$E_6 = (t_6 - y_6)^2 = (0.0 - 0.528)^2 = 0.279$

$E_{total} = \frac{1}{2}(E_5 + E_6) = 0.213$

BACKPROPAGATION (BEISPIEL)



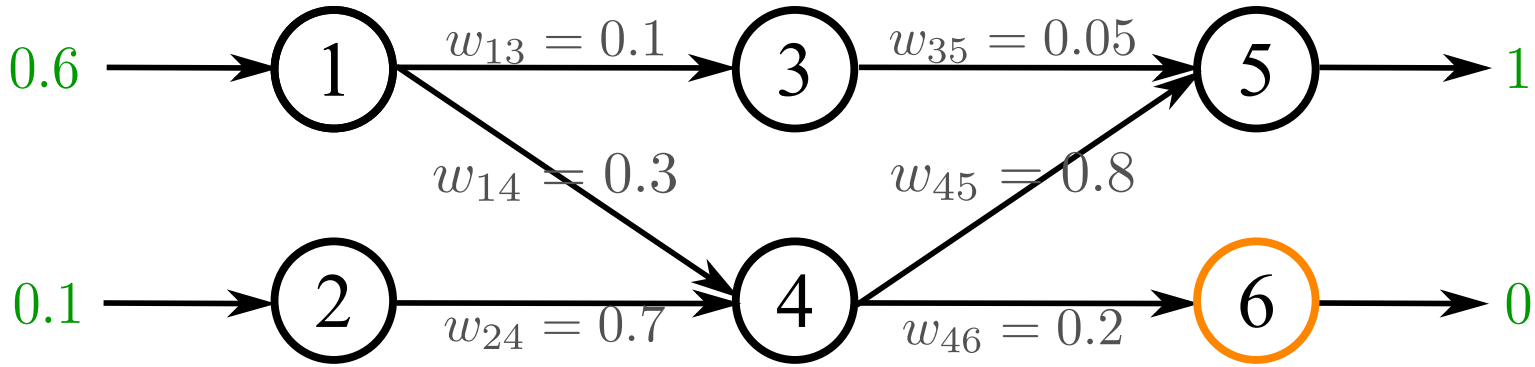
$$\delta_k = \varphi'(net_k)(t_k - y_k)$$

$$\Delta w_{jk} = \eta \delta_k y_j \qquad w_{jk}^{neu} = w_{jk}^{alt} + \Delta w_{jk}$$

Backward Pass (Ausgabeschicht)

$$\delta_5 = \varphi\prime(net_5) \cdot (t_5 - y_5) = \varphi\prime(0.476) \cdot (1 - 0.617) = 0.091$$

BACKPROPAGATION (BEISPIEL)



$$\delta_k = \varphi'(net_k)(t_k - y_k)$$

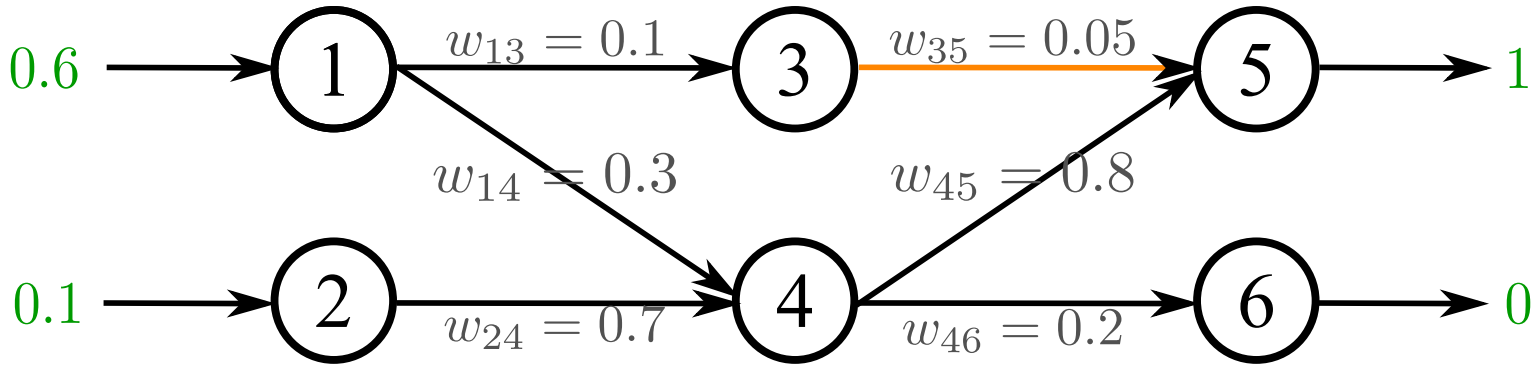
$$\Delta w_{jk} = \eta \delta_k y_j \qquad w_{jk}^{neu} = w_{jk}^{alt} + \Delta w_{jk}$$

Backward Pass (Ausgabeschicht)

$$\delta_5 = \varphi\prime(net_5) \cdot (t_5 - y_5) = \varphi\prime(0.476) \cdot (1 - 0.617) = 0.091$$

$$\delta_6 = \varphi\prime(net_6) \cdot (t_6 - y_6) = \varphi\prime(0.112) \cdot (0 - 0.528) = -0.132$$

BACKPROPAGATION (BEISPIEL)

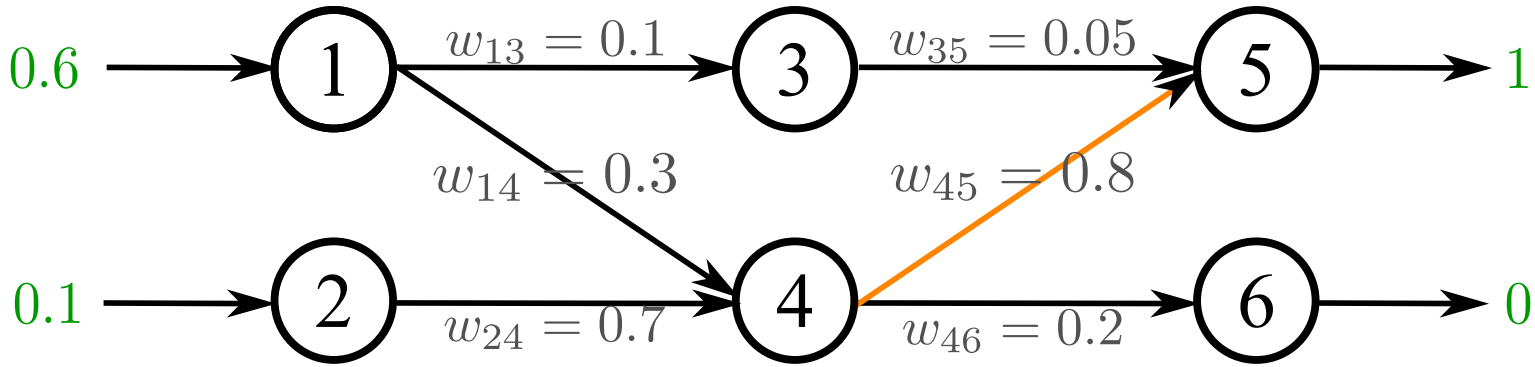


$$\delta_k = \varphi'(net_k)(t_k - y_k)$$
$$\Delta w_{jk} = \eta \delta_k y_j \qquad w_{jk}^{neu} = w_{jk}^{alt} + \Delta w_{jk}$$

Backward Pass (Ausgabeschicht)

$$\delta_5 = \varphi\prime(net_5) \cdot (t_5 - y_5) = \varphi\prime(0.476) \cdot (1 - 0.617) = 0.091$$
$$\delta_6 = \varphi\prime(net_6) \cdot (t_6 - y_6) = \varphi\prime(0.112) \cdot (0 - 0.528) = -0.132$$
$$\Delta w_{35} = \tfrac{1}{2} \cdot \delta_5 \cdot y_3 = 0.023 \quad w_{35}^{neu} = w_{35}^{alt} + \Delta w_{35} = 0.073$$

BACKPROPAGATION (BEISPIEL)



$$\delta_k = \varphi'(net_k)(t_k - y_k)$$

$$\Delta w_{jk} = \eta \delta_k y_j \qquad w_{jk}^{neu} = w_{jk}^{alt} + \Delta w_{jk}$$

Backward Pass (Ausgabeschicht)

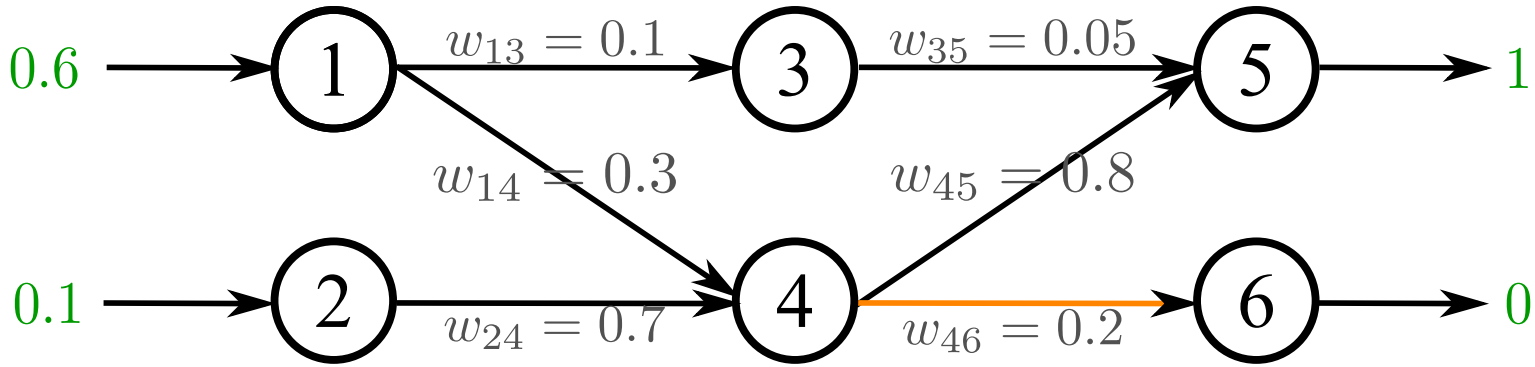
$$\delta_5 = \varphi\prime(net_5) \cdot (t_5 - y_5) = \varphi\prime(0.476) \cdot (1 - 0.617) = 0.091$$

$$\delta_6 = \varphi\prime(net_6) \cdot (t_6 - y_6) = \varphi\prime(0.112) \cdot (0 - 0.528) = -0.132$$

$$\Delta w_{35} = \tfrac{1}{2} \cdot \delta_5 \cdot y_3 = 0.023 \quad w_{35}^{neu} = w_{35}^{alt} + \Delta w_{35} = 0.073$$

$$\Delta w_{45} = \tfrac{1}{2} \cdot \delta_5 \cdot y_4 = 0.026 \quad w_{45}^{neu} = w_{45}^{alt} + \Delta w_{45} = 0.826$$

BACKPROPAGATION (BEISPIEL)



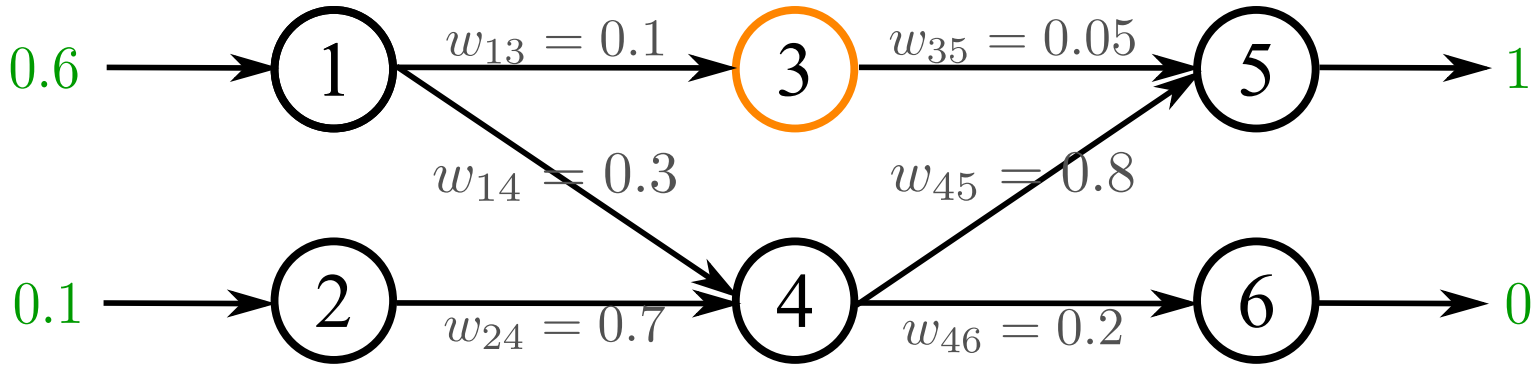
$$\delta_k = \varphi'(net_k)(t_k - y_k)$$
$$\Delta w_{jk} = \eta \delta_k y_j \qquad w_{jk}^{neu} = w_{jk}^{alt} + \Delta w_{jk}$$

Backward Pass (Ausgabeschicht)

$$\delta_5 = \varphi\prime(net_5) \cdot (t_5 - y_5) = \varphi\prime(0.476) \cdot (1 - 0.617) = 0.091$$
$$\delta_6 = \varphi\prime(net_6) \cdot (t_6 - y_6) = \varphi\prime(0.112) \cdot (0 - 0.528) = -0.132$$

$$\Delta w_{35} = \tfrac{1}{2} \cdot \delta_5 \cdot y_3 = 0.023 \qquad w_{35}^{neu} = w_{35}^{alt} + \Delta w_{35} = 0.073$$
$$\Delta w_{45} = \tfrac{1}{2} \cdot \delta_5 \cdot y_4 = 0.026 \qquad w_{45}^{neu} = w_{45}^{alt} + \Delta w_{45} = 0.826$$
$$\Delta w_{46} = \tfrac{1}{2} \cdot \delta_6 \cdot y_4 = -0.037 \qquad w_{46}^{neu} = w_{46}^{alt} + \Delta w_{46} = 0.163$$

BACKPROPAGATION (BEISPIEL)

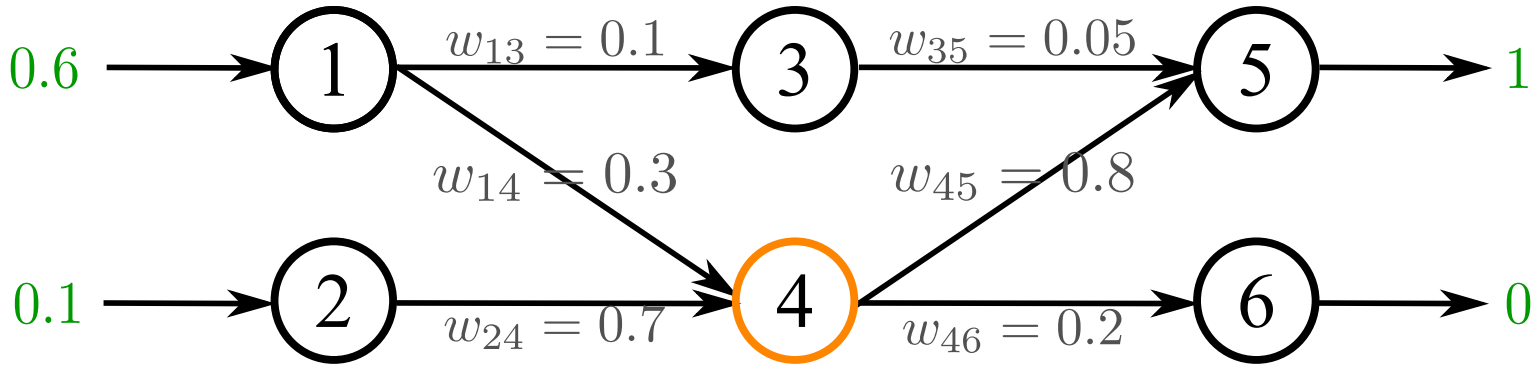


$$\delta_j = \varphi'(net_j) \sum_k \delta_k w_{jk}$$
$$\Delta w_{ij} = \eta \delta_j y_i \qquad w_{ij}^{neu} = w_{ij}^{alt} + \Delta w_{ij}$$

Backward Pass (Innere Schichten)

$\delta_3 = \varphi\prime(net_3) \cdot (\delta_5 \cdot w_{35}^{alt}) = \varphi\prime(0.06) \cdot (0.091 \cdot 0.05) = 5.121 \times 10^{-6}$

BACKPROPAGATION (BEISPIEL)



$$\delta_j = \varphi'(net_j) \sum_k \delta_k w_{jk}$$

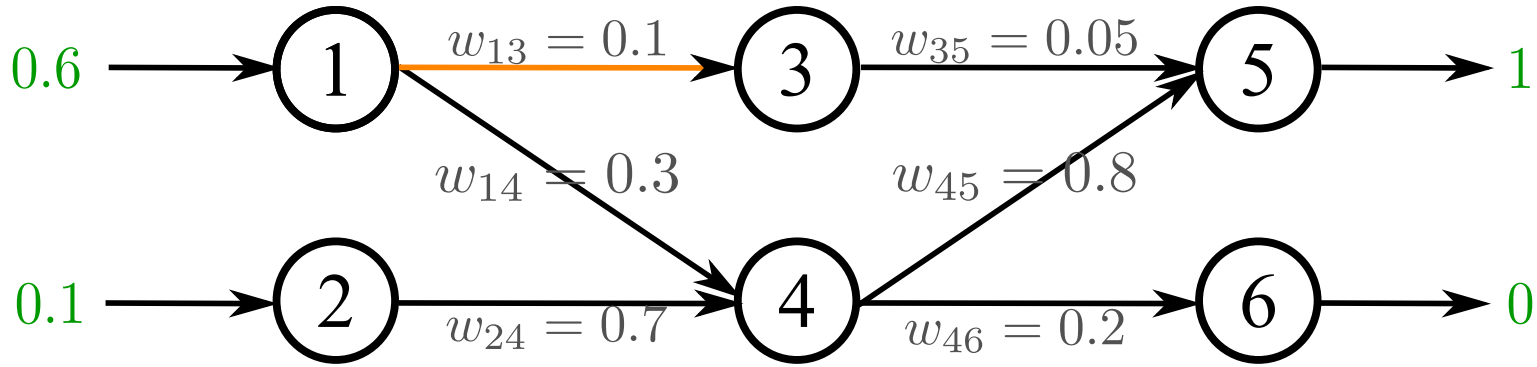
$$\Delta w_{ij} = \eta \delta_j y_i \qquad w_{ij}^{neu} = w_{ij}^{alt} + \Delta w_{ij}$$

Backward Pass (Innere Schichten)

$$\delta_3 = \varphi\prime(net_3) \cdot (\delta_5 \cdot w_{35}^{alt}) = \varphi\prime(0.06) \cdot (0.091 \cdot 0.05) = 5.121 \times 10^{-6}$$

$$\delta_4 = \varphi\prime(net_4) \cdot (\delta_5 \cdot w_{45}^{alt} + \delta_6 \cdot w_{46}^{alt}) = 0.011$$

BACKPROPAGATION (BEISPIEL)

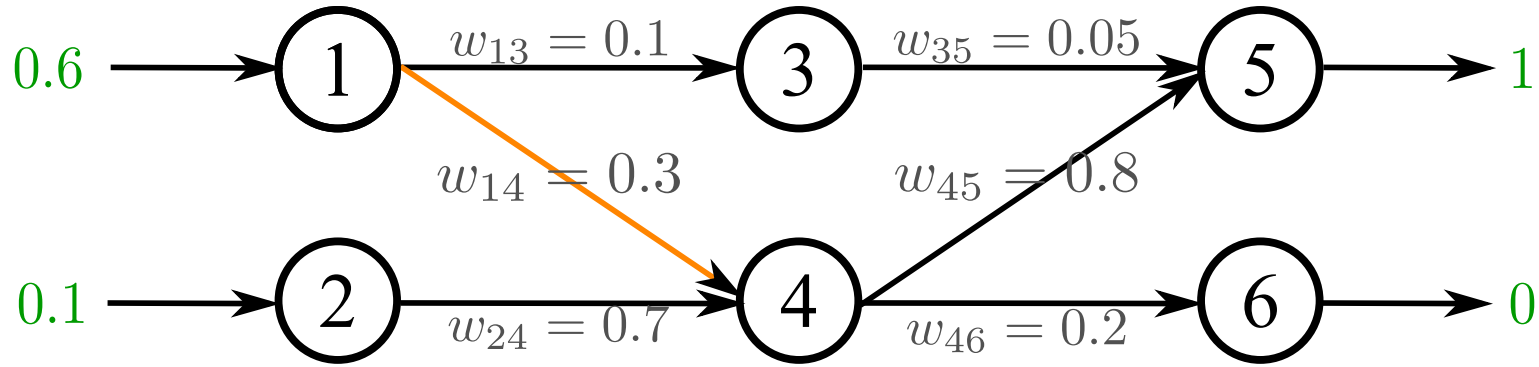


$$\delta_j = \varphi'(net_j) \sum_k \delta_k w_{jk}$$
$$\Delta w_{ij} = \eta \delta_j y_i \qquad w_{ij}^{neu} = w_{ij}^{alt} + \Delta w_{ij}$$

Backward Pass (Innere Schichten)

$$\delta_3 = \varphi\prime(net_3) \cdot (\delta_5 \cdot w_{35}^{alt}) = \varphi\prime(0.06) \cdot (0.091 \cdot 0.05) = 5.121 \times 10^{-6}$$
$$\delta_4 = \varphi\prime(net_4) \cdot (\delta_5 \cdot w_{45}^{alt} + \delta_6 \cdot w_{46}^{alt}) = 0.011$$
$$\Delta w_{13} = \tfrac{1}{2} \cdot \delta_3 \cdot y_1 = 1.536 \times 10^{-6} \qquad w_{13}^{neu} = w_{13}^{alt} + \Delta w_{13} = 0.1$$

BACKPROPAGATION (BEISPIEL)

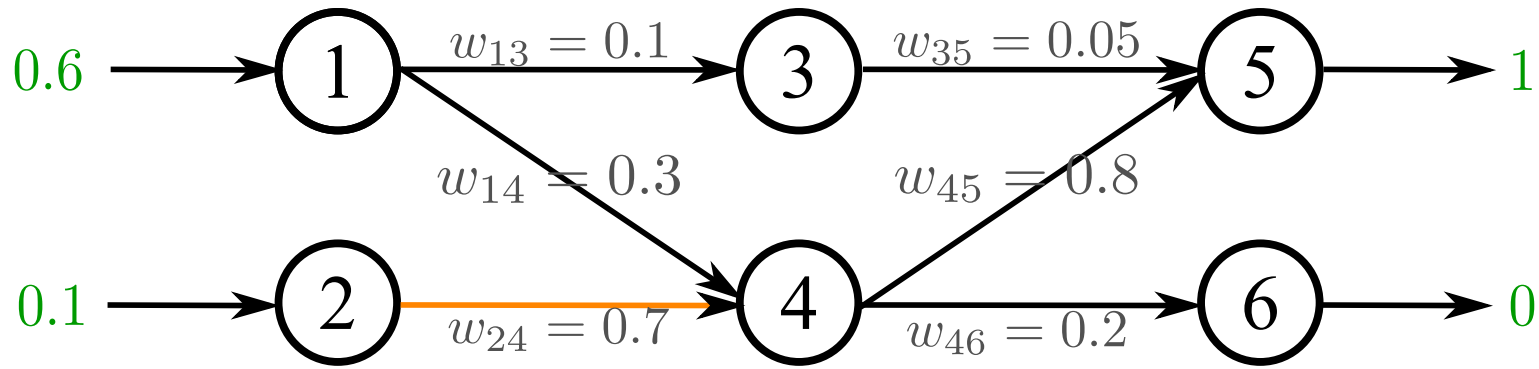


$$\delta_j = \varphi'(net_j) \sum_k \delta_k w_{jk}$$
$$\Delta w_{ij} = \eta \delta_j y_i \qquad w_{ij}^{neu} = w_{ij}^{alt} + \Delta w_{ij}$$

Backward Pass (Innere Schichten)

$$\delta_3 = \varphi\prime(net_3) \cdot (\delta_5 \cdot w_{35}^{alt}) = \varphi\prime(0.06) \cdot (0.091 \cdot 0.05) = 5.121 \times 10^{-6}$$
$$\delta_4 = \varphi\prime(net_4) \cdot (\delta_5 \cdot w_{45}^{alt} + \delta_6 \cdot w_{46}^{alt}) = 0.011$$
$$\Delta w_{13} = \tfrac{1}{2} \cdot \delta_3 \cdot y_1 = 1.536 \times 10^{-6} \qquad w_{13}^{neu} = w_{13}^{alt} + \Delta w_{13} = 0.1$$
$$\Delta w_{14} = \tfrac{1}{2} \cdot \delta_4 \cdot y_1 = 3.409 \times 10^{-3} \qquad w_{14}^{neu} = w_{14}^{alt} + \Delta w_{14} = 0.303$$

BACKPROPAGATION (BEISPIEL)



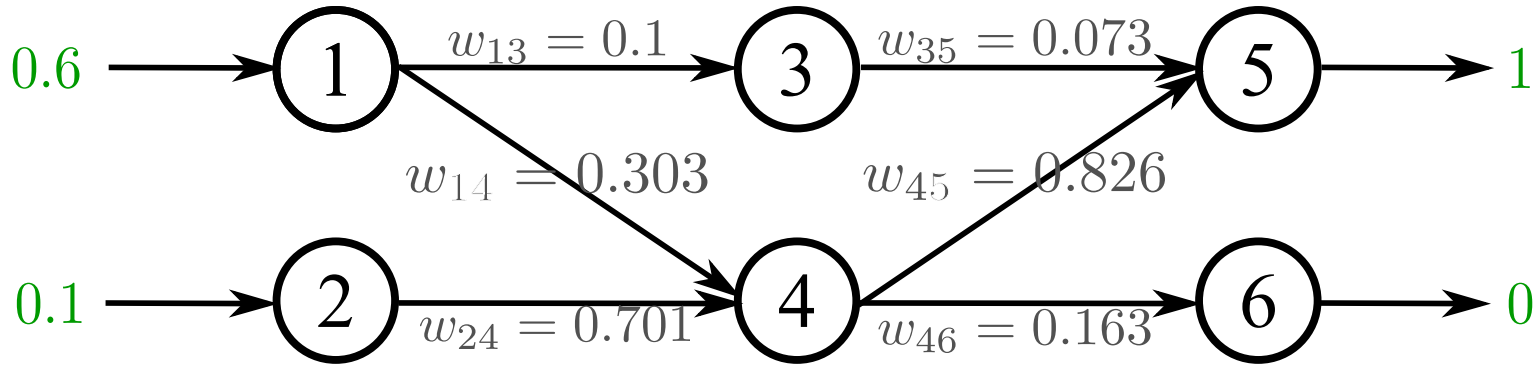
$$\delta_j = \varphi'(net_j) \sum_k \delta_k w_{jk}$$
$$\Delta w_{ij} = \eta \delta_j y_i \qquad w_{ij}^{neu} = w_{ij}^{alt} + \Delta w_{ij}$$

Backward Pass (Innere Schichten)

$$\delta_3 = \varphi\prime(net_3) \cdot (\delta_5 \cdot w_{35}^{alt}) = \varphi\prime(0.06) \cdot (0.091 \cdot 0.05) = 5.121 \times 10^{-6}$$
$$\delta_4 = \varphi\prime(net_4) \cdot (\delta_5 \cdot w_{45}^{alt} + \delta_6 \cdot w_{46}^{alt}) = 0.011$$

$$\Delta w_{13} = \tfrac{1}{2} \cdot \delta_3 \cdot y_1 = 1.536 \times 10^{-6} \qquad w_{13}^{neu} = w_{13}^{alt} + \Delta w_{13} = 0.1$$
$$\Delta w_{14} = \tfrac{1}{2} \cdot \delta_4 \cdot y_1 = 3.409 \times 10^{-3} \qquad w_{14}^{neu} = w_{14}^{alt} + \Delta w_{14} = 0.303$$
$$\Delta w_{24} = \tfrac{1}{2} \cdot \delta_4 \cdot y_2 = 5.682 \times 10^{-4} \qquad w_{24}^{neu} = w_{24}^{alt} + \Delta w_{24} = 0.701$$

BACKPROPAGATION (BEISPIEL)



Forward Pass

$net_3 = 0.06000092178$	$y_3 = 0.5149957319$
$net_4 = 0.25210222$	$y_4 = 0.5626938611$
$net_5 = 0.5020984708$	$y_5 = 0.6229523527$
$net_6 = 0.0917247263$	$y_6 = 0.5229151176$

Fehlerbestimmung

$E_5^{alt} = 0.147$	$E_6^{alt} = 0.279$
$E_5^{neu} = 0.142$	$E_6^{neu} = 0.273$
$E_{total}^{alt} = 0.213$	$E_{total}^{neu} = 0.208$

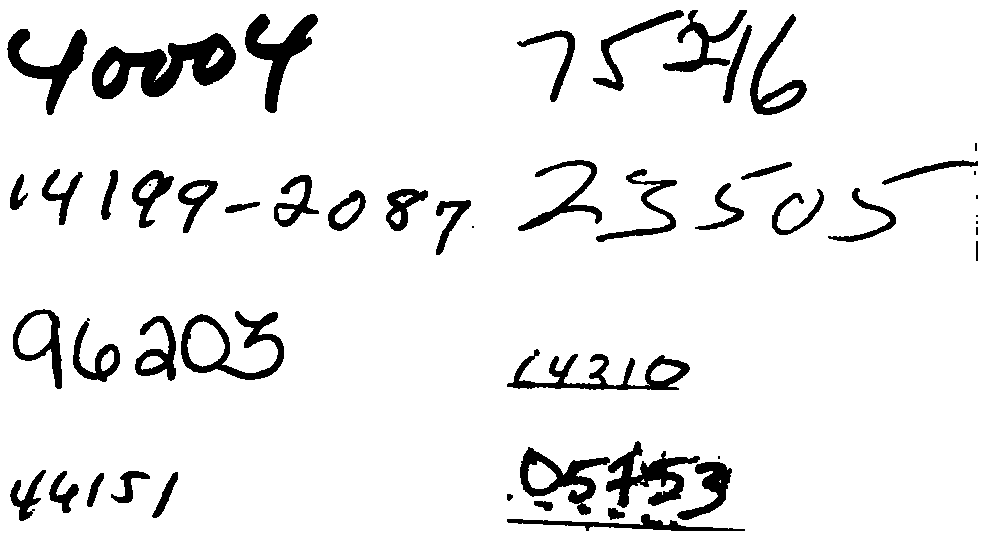
HANDSCHRIFTERKENNUNG MITTELS BACKPROPAGATION

VORSTELLUNG

- Thema: Erkennung handgeschriebener Ziffern mittels eines Backpropagation Netzes
- Ziel: Backpropagation Netze auf reale Bilderkennungsprobleme abbilden, ohne Eingabedaten aufwändig aufzubereiten
- Besonderes Merkmal des Papers: Netzwerkarchitektur

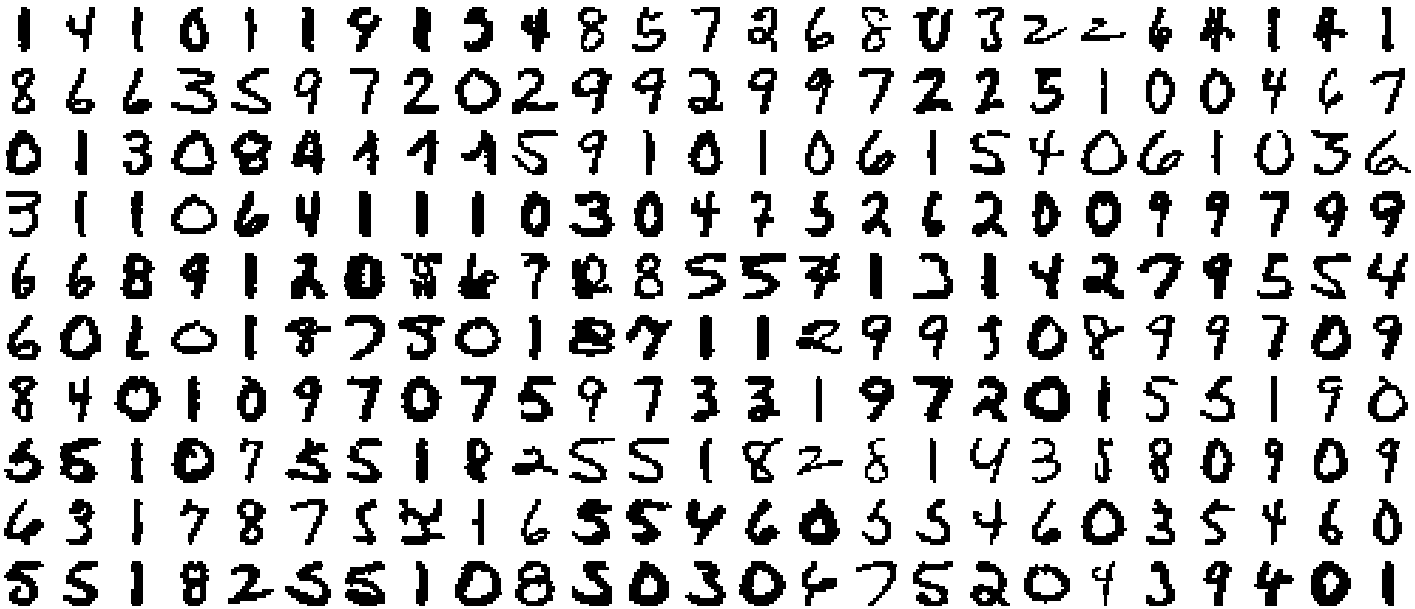
EINGABEDATEN

- Zipcodes
  - Postleitzahlen der USA
  - Nur schwarze und weiße Pixel
  - Gute Differenzierbarkeit vom Hintergrund
  - Nur 10 Ausgabekategorien (Zahlen 0-9)
  - Dennoch reales "Bilderkennungsproblem"
    - Variation in Größe, Schreibstil, Schriftart und Sorgfalt des Schriftbildes



EINGABEDATEN

- Sowohl handgeschriebene, als auch maschinelle Ziffern
- Teilweise fehlerhafte Segmentierung und/oder Klassifizierung der Daten

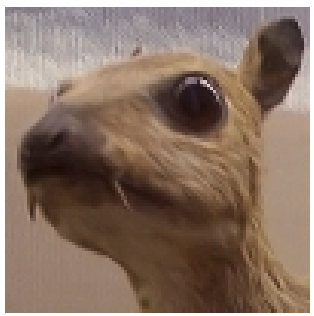


FEATURE MAPS

Feature Map als Ergebnis von "Convolution" (=Faltung) eines Eingabebildes zu einem kleineren Ausgabebild

- Stärkere Repräsentation bestimmter Merkmale

Input image



Convolution Kernel

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

Feature map

- Prinzip: Eingabebild als Matrix
  - 1 Eintrag der Matrix = 1 Pixel des Eingabebildes
- "Kernel- Gewichtsmatrix, die Eingabebild zu Feature map verarbeitet

FEATURE MAPS - FUNKTIONSWEISE

Eingabematrix

1	0	1
0	1	1
1	0	0

×

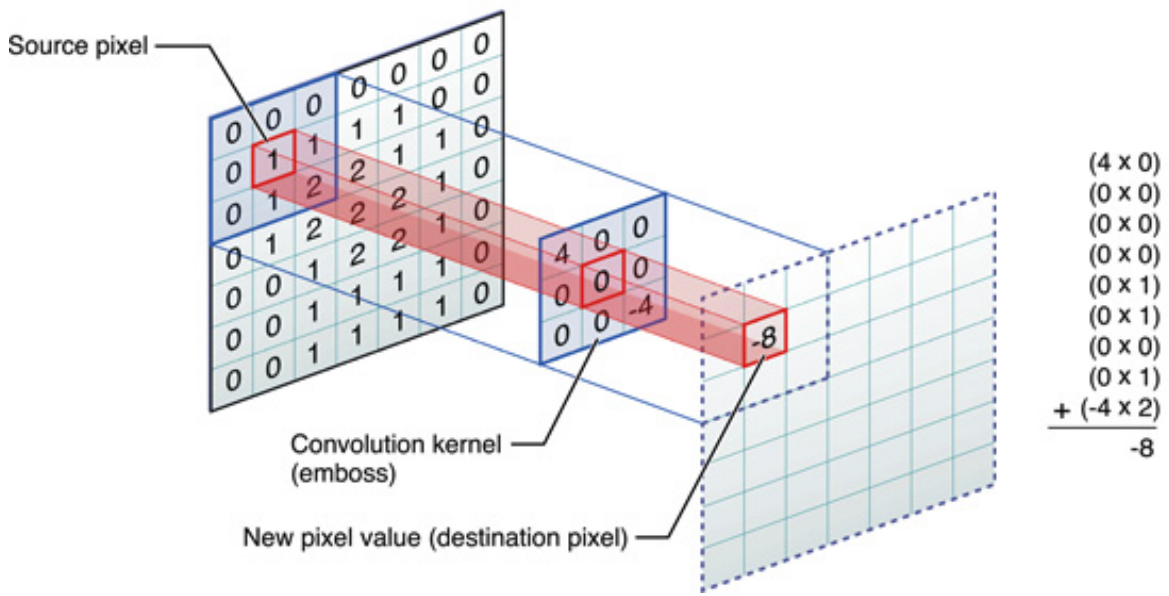
Kernel

1	0
0	1

=

Ausgabematrix

2	1
0	1



FEATURE MAPS - FUNKTIONSWEISE

Eingabematrix

1	0	1
0	1	1
1	0	0

$\times$

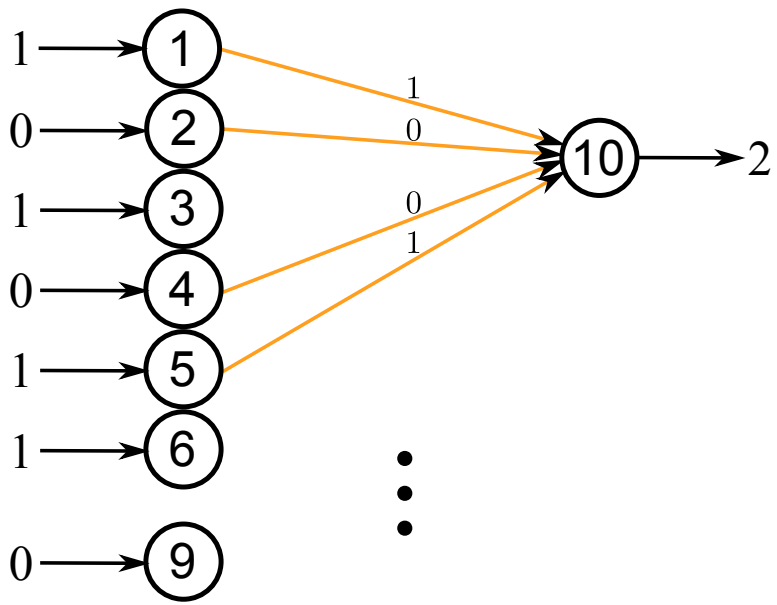
Kernel

1	0
0	1

$=$

Ausgabematrix

2	



FEATURE MAPS - FUNKTIONSWEISE

Eingabematrix

1	0	1
0	1	1
1	0	0

×

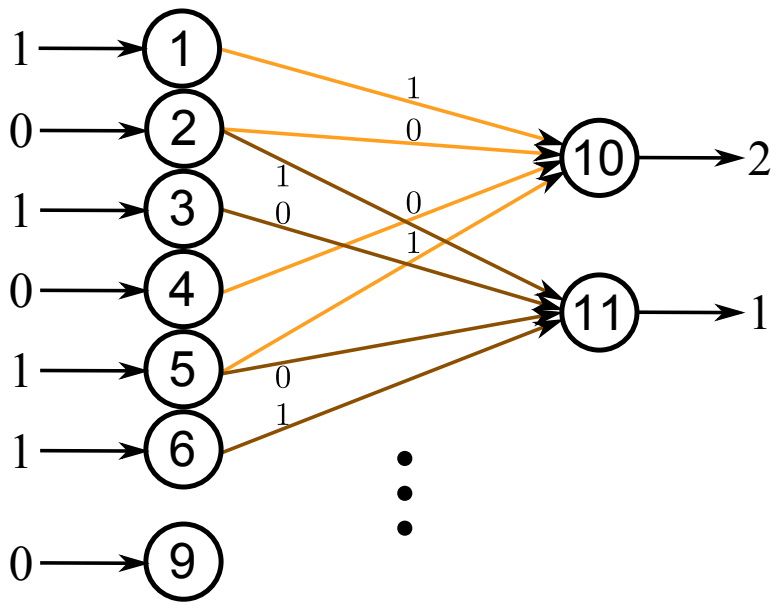
Kernel

1	0
0	1

=

Ausgabematrix

2	1



FEATURE MAPS - FUNKTIONSWEISE

Eingabematrix

1	0	1
0	1	1
1	0	0

×

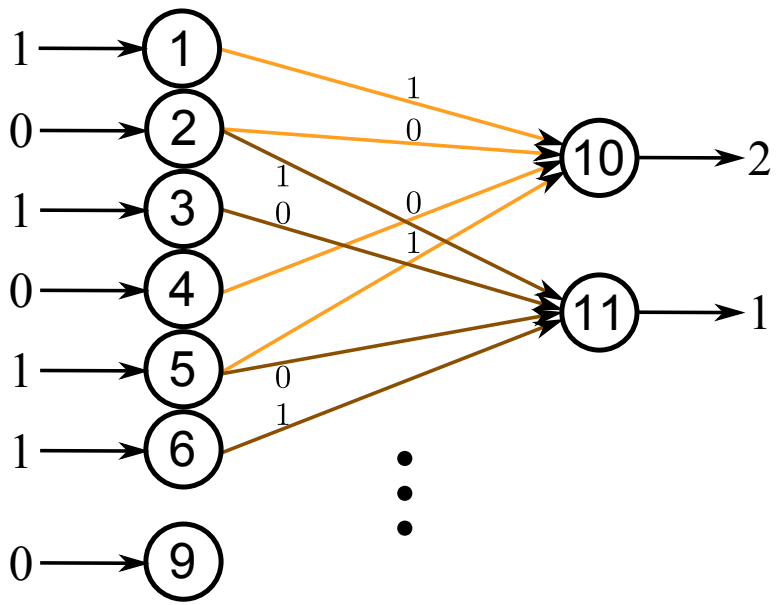
Kernel

1	0
0	1

=

Ausgabematrix

2	1
0	



FEATURE MAPS - FUNKTIONSWEISE

Eingabematrix

1	0	1
0	1	1
1	0	0

×

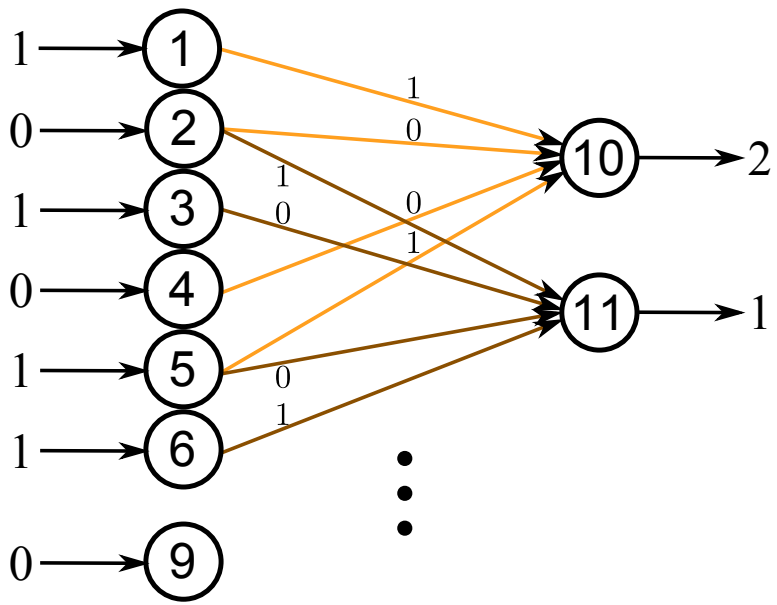
Kernel

1	0
0	1

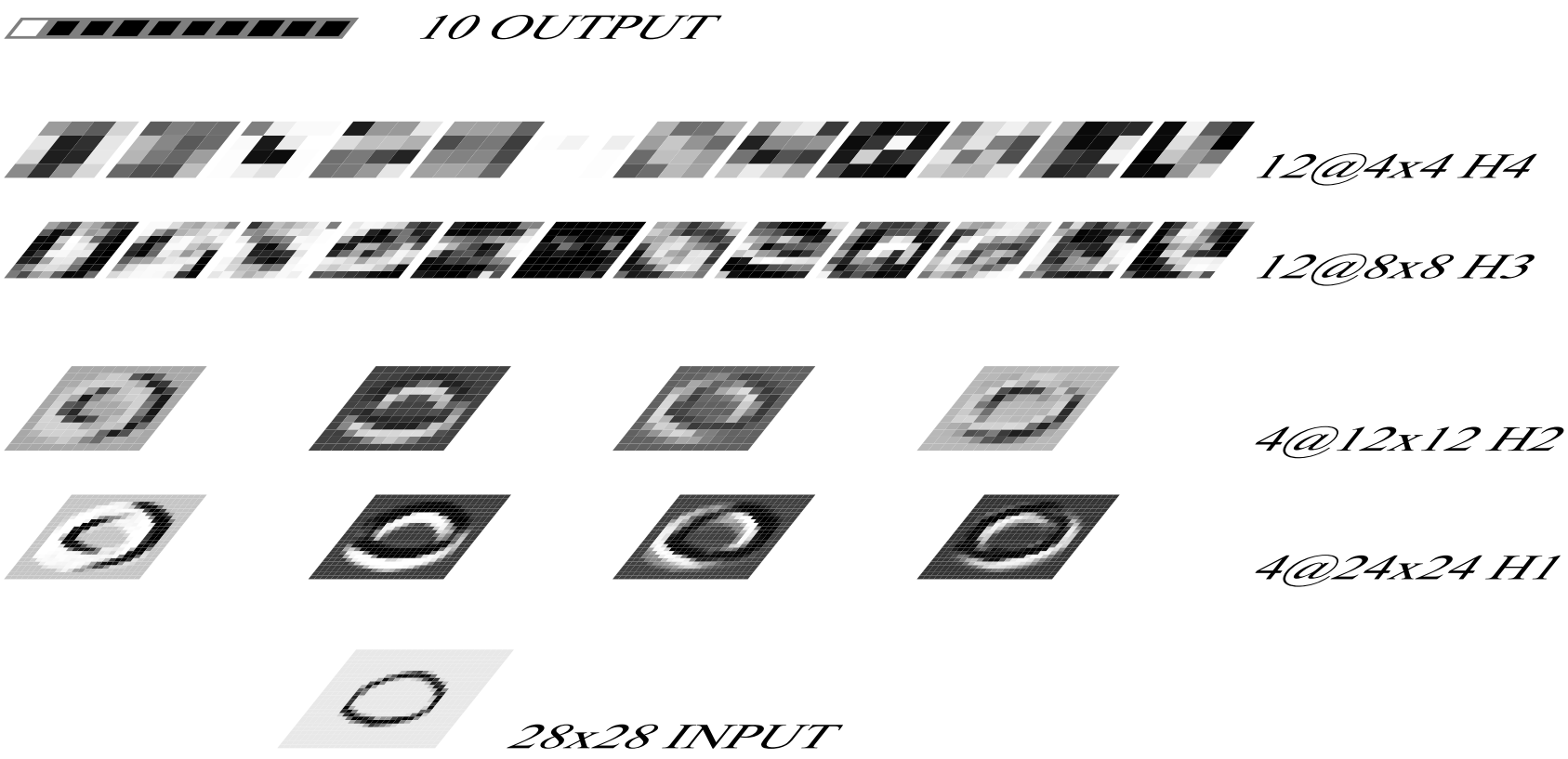
=

Ausgabematrix

2	1
0	1



ARCHITEKTUR



ERGEBNIS

- Fehlerrate nach ersten 30 Trainingsdurchläufen auf zwei verschiedenen Datensätzen:
  - Trainingsset: $E = 1.1\%$
  - Testset: $E = 3.4\%$
- Ziel: Verbesserung der Netzgenauigkeit
- Idee: Ablehnen von Testeingaben bis $E \leq 1\%$ gilt mit best. Ablehnungskriterien

ERGEBNIS

- Bestes Ergebnis für Testset: *Ablehnungsrate* = 5.7%
- Nur handgeschriebene Daten des Testsets: *Ablehnungsrate* = 9%
- Schlussfolgerung: Die meisten Netzfehler durch ...
  - Fehlerhafte Segmentierung
  - Unleserlichkeit
  - Fehlerhafte Kategoriezuweisung
- Trainingsdauer: Auf einer SUN SPARCstation1 ca. 3 Tage für 30 Trainingsdurchläufe

Vielen Dank für Ihre Aufmerksamkeit

QUELLEN

- Prof. Dr.-Ing. Hans-Jürgen Scheibl. Grundlagen neuronaler Netze. available from http://home.f1.htw-berlin.de/scheibl/Algor/index.htm?./Neuronal/Grundlagen.htm
- Roman Kohut. Neuronale Netze als Modell Boolescher Funktionen, 2007. Freiberg (Sachsen), Techn. Univ., Diss., 2007.
- Jürgen Paetz. Soft Computing in der Bioinformatik : Eine grundlegende Einführung und Übersicht, 2006.
- Gerald Reif. Grundlagen neuronale Netze. http://www.iicm.tugraz.at/greif/node10.html
- Michael Koops. Erklärung zum biologischen Neuron. http://www.biologie-lexikon.de/lexikon/neuron.php
- Lars Larsson. Zur Validierung der Spezifikationen von videobildverarbeitenden Komponenten unter realen Anwendungsbedingungen. PhD thesis, 2000.
- Karl F. Wender Günter Daniel Rey. Grundlagen neuronale Netze. available from http://www.neuronalesnetz.de/
- Jeremy Kun. Neural Networks and the Backpropagation Algorithm. http://jeremykun.com/2012/12/09/neural-networks-and-backpropagation/ http://www.cs.hs-rm.de/~ulges/teaching/14NUMANA/index.html
- Stephan Wiesebach Sebastian Seifert, Christian Weidner. Grundlagen Backpropagation Netzwerke. http://cs.uni-muenster.de/Professoren/Lippe/lehre/skripte/wwwnnscript/probleme.html
- Y. Le et al. Cun. Advances in Neural Information Processing Systems 2. chapter Handwritten Digit Recognition with a Back-propagation Network. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1990.
- Tim Dettmers. Understanding Convolution in Deep Learning. http://timdettmers.com/2015/03/26/convolution-deep-learning/
- Apple Inc. Performing Convolution Operations.https://developer.apple.com/library/ios/documentation/Performance/Conceptual/vImage/ConvolutionOperations/ConvolutionOperations.html
- Theano Development Team. Convolutional neural Networks (LeNet).http://deeplearning.net/tutorial/lenet.html
- Yann LeCun. Convolutional neural Networks (LeNet-5). http://yann.lecun.com/exdb/lenet/