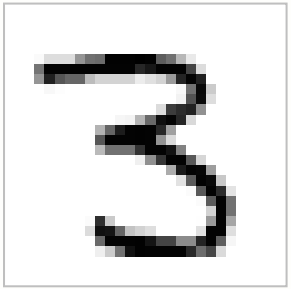
NEURONALE NETZE

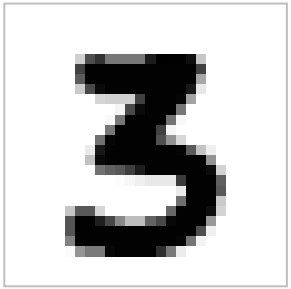
Handschriftliche Zahlen erkennen

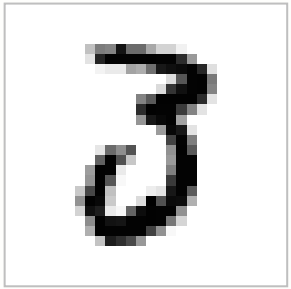
Jasper Gude

3. Dezember 2023
Carl-Friedrich-Gauß-Gymnasium

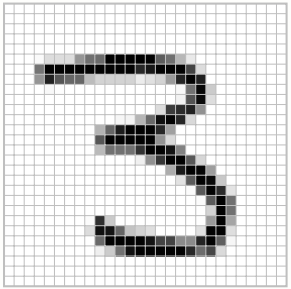
2.1 Modellierung des Problems

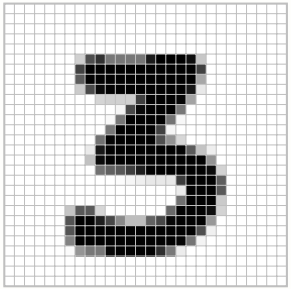


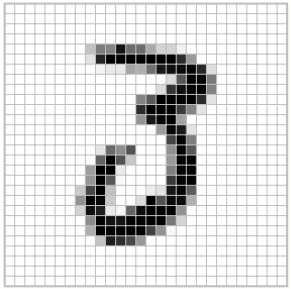




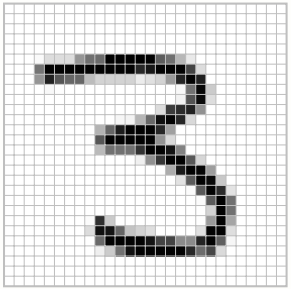
2.2 Modellierung des Problems

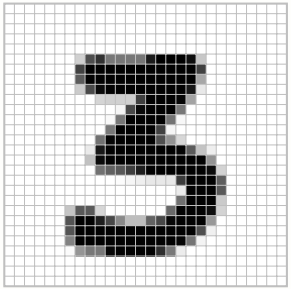


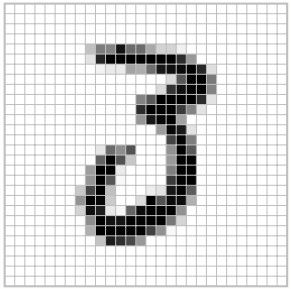




2.3 Modellierung des Problems

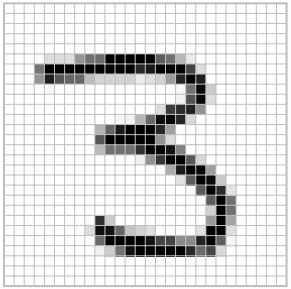






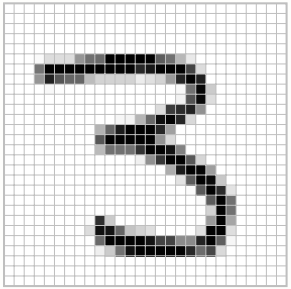
0,00 | 0,25 | 0,50 | 0,75 | 1,00

3.1 Überführung auf eine Netzstruktur

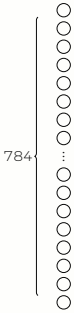


28px × 28px

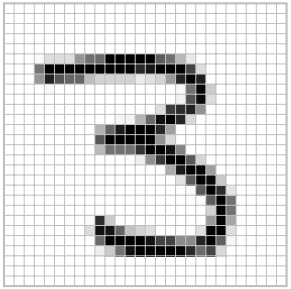
3.2 Überführung auf eine Netzstruktur



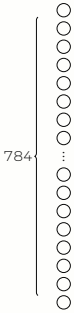
28px × 28px



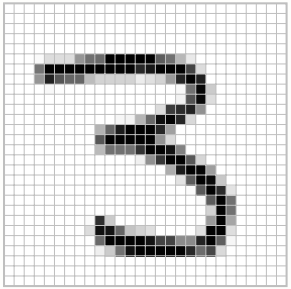
3.3 Überführung auf eine Netzstruktur



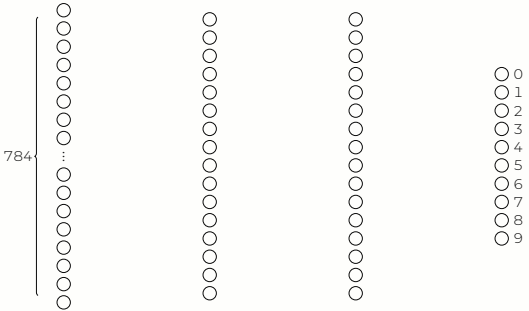
28px × 28px



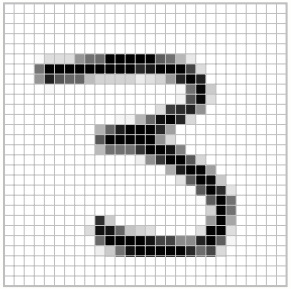
3.4 Überführung auf eine Netzstruktur



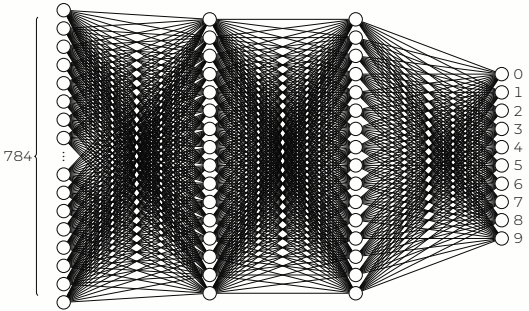
28px × 28px



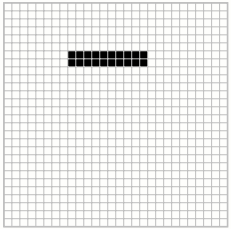
3.5 Überführung auf eine Netzstruktur



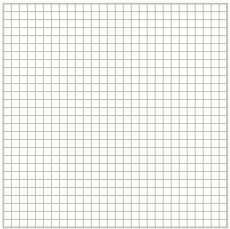
28px × 28px



4.1 Gewichtungen setzen



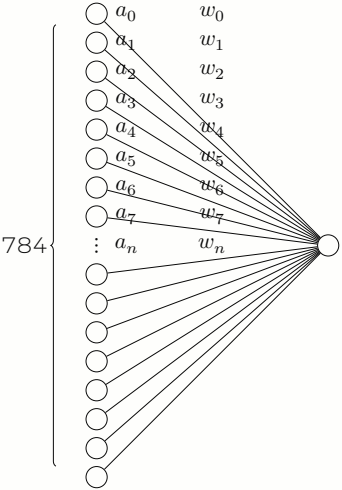
Inputs



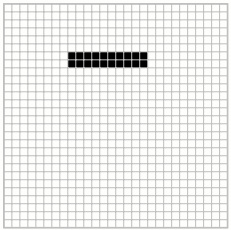
Gewichte

Linearkombination

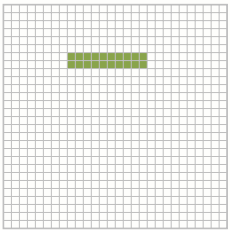
$$w_0 a_0 + w_1 a_1 + \cdots + w_n a_n$$



4.2 Gewichtungen setzen



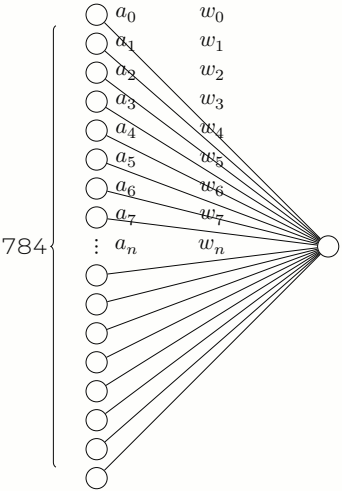
Inputs



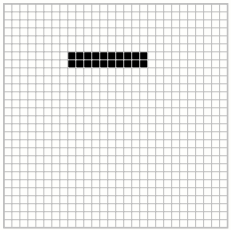
Gewichte

Linearkombination

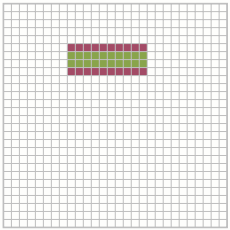
$$w_0 a_0 + w_1 a_1 + \cdots + w_n a_n$$



4.3 Gewichtungen setzen



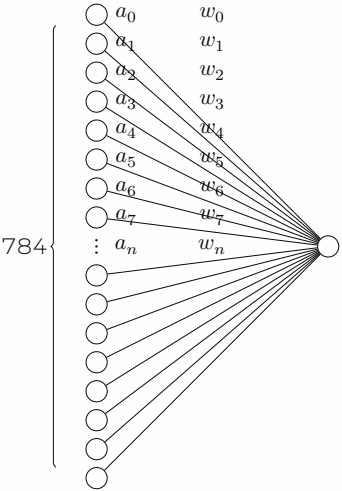
Inputs



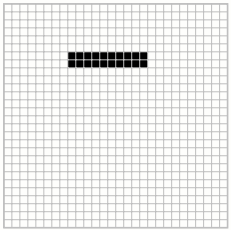
Gewichte

Linearkombination

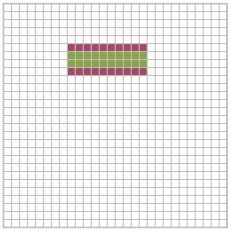
$$w_0 a_0 + w_1 a_1 + \cdots + w_n a_n$$



4.4 Gewichtungen setzen



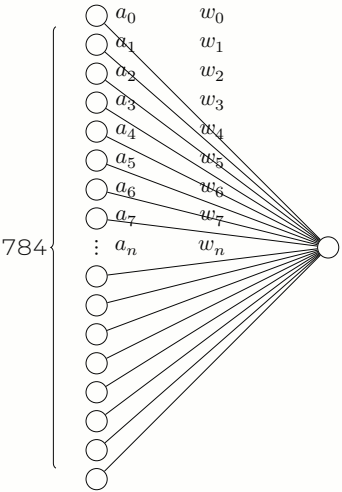
Inputs



Gewichte

Linearkombination

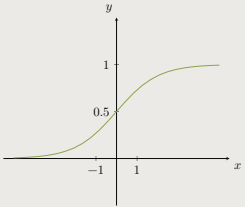
$$w_0 a_0 + w_1 a_1 + \cdots + w_n a_n - b$$

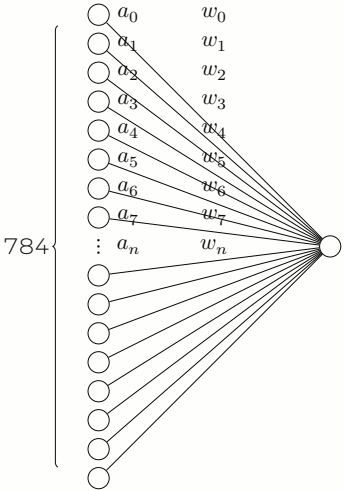


5 Zahlenbereich begrenzen

Sigmoidfunktion

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

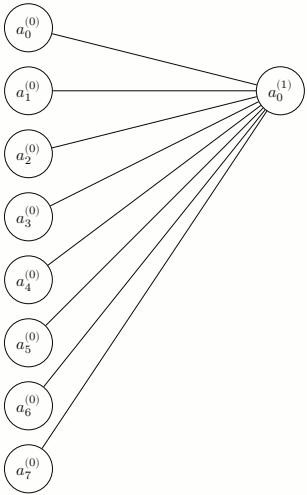




6.1 Alles zusammen setzen

Aktivierungsfunktion

$$a_0^{(1)} = \sigma(w_{0,0}a_0^{(0)} + w_{0,1}a_1^{(0)} + \cdots + w_{0,n}a_n^{(0)} - b_0)$$

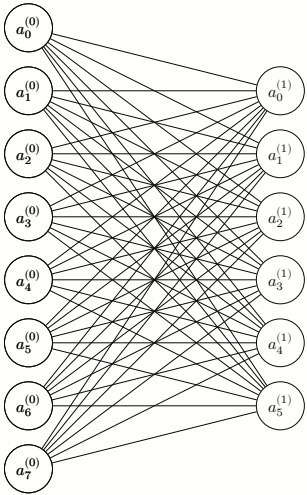


Alles zusammen setzen

Aktivierungsfunktion

$$a_0^{(1)} = \sigma(w_{0,0}a_0^{(0)} + w_{0,1}a_1^{(0)} + \cdots + w_{0,n}a_n^{(0)} - b_0)$$

$$\begin{bmatrix} a_0^{(1)} \\ a_1^{(1)} \\ \vdots \\ a_k^{(1)} \end{bmatrix} = \sigma\left(\begin{bmatrix} w_{0,0} & w_{0,1} & \cdots & w_{0,n} \\ w_{1,0} & w_{1,1} & \cdots & w_{1,n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{k,0} & w_{k,1} & \cdots & w_{k,n} \end{bmatrix} \begin{bmatrix} a_0^{(0)} \\ a_1^{(0)} \\ \vdots \\ a_n^{(0)} \end{bmatrix} + \begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_k \end{bmatrix} \right)$$

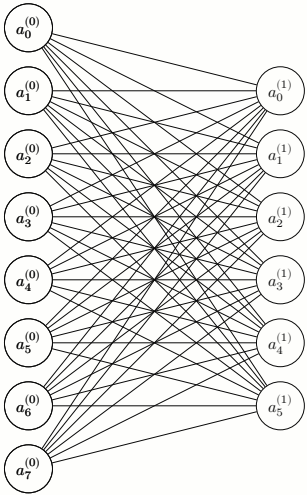


6.3 Alles zusammen setzen

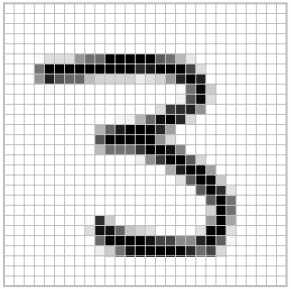
Aktivierungsfunktion

$$\vec{a^{(1)}} = \sigma(W\vec{a^{(0)}} + \vec{b})$$

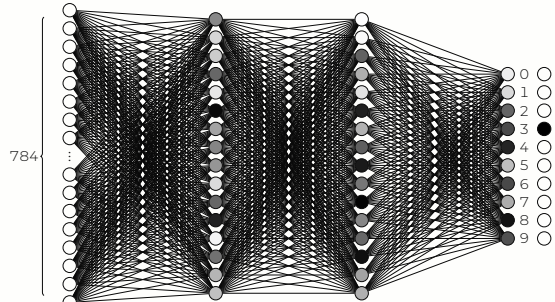
$$\begin{bmatrix} a_0^{(1)} \\ a_1^{(1)} \\ \vdots \\ a_k^{(1)} \end{bmatrix} = \sigma \left(\begin{bmatrix} w_{0,0} & w_{0,1} & \cdots & w_{0,n} \\ w_{1,0} & w_{1,1} & \cdots & w_{1,n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{k,0} & w_{k,1} & \cdots & w_{k,n} \end{bmatrix} \begin{bmatrix} a_0^{(0)} \\ a_1^{(0)} \\ \vdots \\ a_n^{(0)} \end{bmatrix} + \begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_k \end{bmatrix} \right)$$



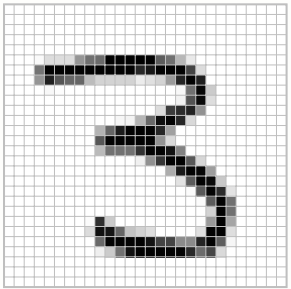
7.1 Fehler bestimmen



28px × 28px



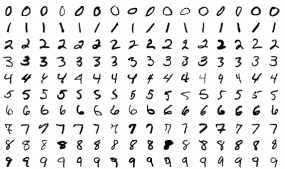
7.2 Fehler bestimmen



28px × 28px

$$C_3(\vec{W}) = \text{Summe}\left\{\begin{array}{lcl}
\left(0.17\,0\right) & - & \left(0.00\right)^2 \\
\left(0.48\,1\right) & - & \left(0.00\right)^2 \\
\left(0.38\,2\right) & - & \left(0.00\right)^2 \\
\left(0.20\,3\right) & - & \left(1.00\right)^2 \\
\left(0.76\,4\right) & - & \left(0.00\right)^2 \\
\left(0.65\,5\right) & - & \left(0.00\right)^2 \\
\left(0.21\,6\right) & - & \left(0.00\right)^2 \\
\left(0.72\,7\right) & - & \left(0.00\right)^2 \\
\left(0.68\,8\right) & - & \left(0.00\right)^2 \\
\left(0.81\,9\right) & - & \left(0.00\right)^2
\end{array}\right.$$

7.3 Fehler bestimmen



Dataset D

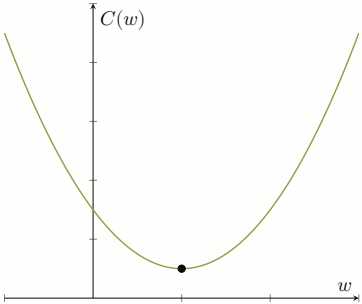
Fehlerfunktion: Mean Squared Error

m : Anzahl der Einträge in D
d : Eintrag (Pixelvektor) in D

$$C_D(\vec{W}) = \frac{1}{m} \sum_{d \in D} (\sum_{i=0}^{9} (a_i^{(3,d)} - s_i^{(d)})^2)$$

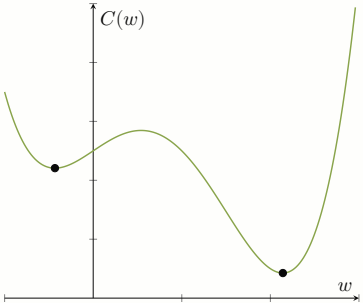
8.1 Fehler minimieren

Globales Minimum durch Ableiten



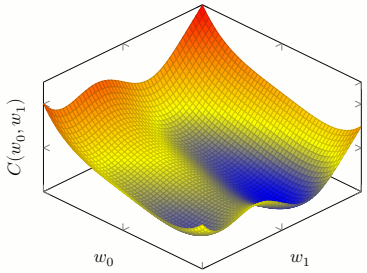
8.2 Fehler minimieren

Zwei lokale Minima; globales Minimum durch Vergleich



8.3 Fehler minimieren

Finden lokaler Minima durch Ableiten nicht möglich



Fehler minimieren

Gradient Descent

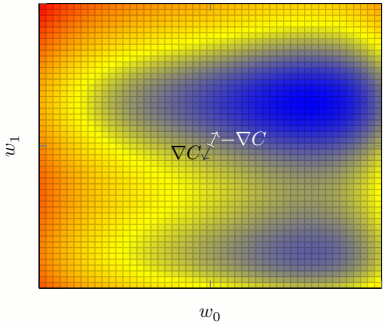
$\nabla C(\vec{W})$: Gradient der Fehlerfunktion C an Stelle $\vec{W}$

Vektor in Richtung des größten Anstiegs

η : Lernrate

Springende Werte verhindern

$$\vec{W_{neu}} = \vec{W} - \eta \nabla C(\vec{W})$$



8.5 Fehler minimieren

Gradient Descent

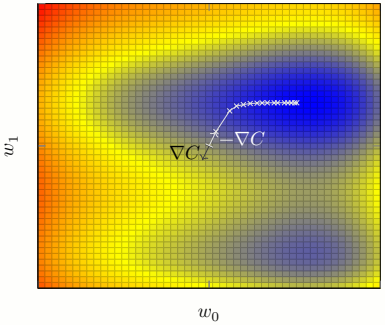
$\nabla C(\vec{W})$: Gradient der Fehlerfunktion C an Stelle $\vec{W}$

Vektor in Richtung des größten Anstiegs

η : Lernrate

Springende Werte verhindern

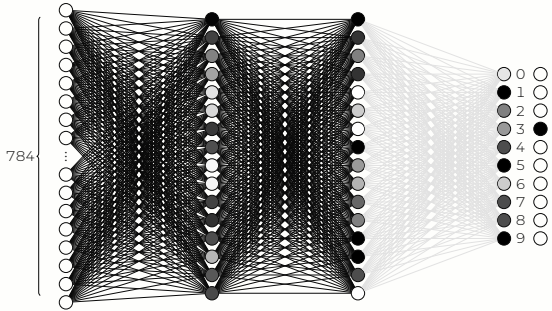
$$\vec{W_{neu}} = \vec{W} - \eta \nabla C(\vec{W})$$



9.1 Gradienten bestimmen

Backpropagation

Bias verändern

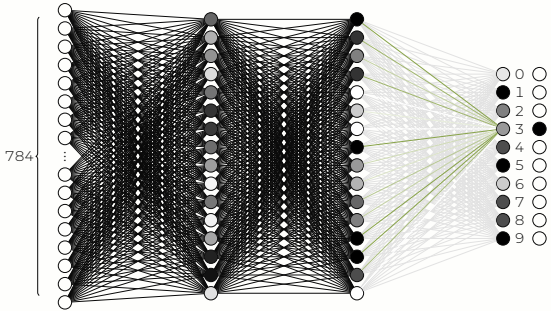


9.2 Gradienten bestimmen

Backpropagation

Bias verändern
Gewichte verändern

Proportional zu Neuronen aus vorheriger Schicht



9.3 Gradienten bestimmen

Backpropagation

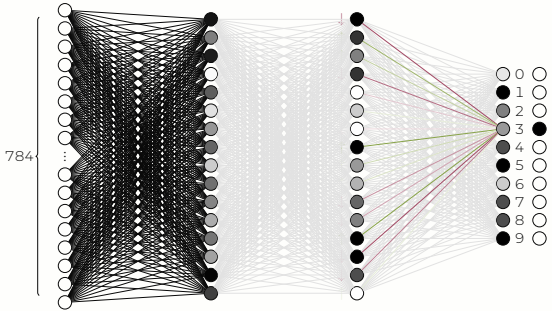
Bias verändern

Gewichte verändern

Proportional zu Neuronen aus vorheriger Schicht

Neuronen verändern

Gewichte aus vorheriger Schicht verändern



[Mös17] Cedric Mössner. *Neuronale Netze*. 2017

[San17] Grant Sanderson. *Neural Networks. The basics of neural networks, and the math behind how they learn*. 2017

[Wik23] Wikipedia. *Künstliches neuronales Netz*. 2023

Jasper Gude

Hockenheim, 3. Dezember 2023