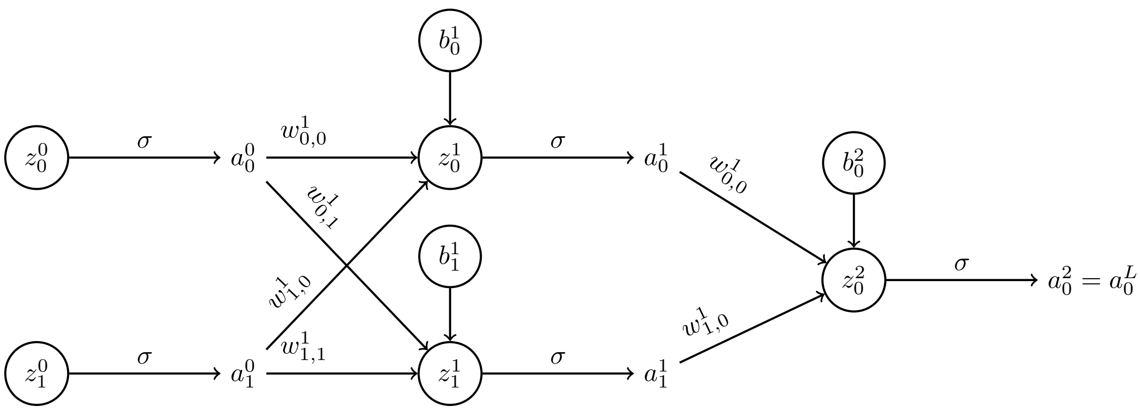
Matrizen in Neuronalen Netzwerken

Tim Zollner

17.02.2025

1 Feed-Forward

1.1 Grafik



1.2 Formeln

$$z_0^1 = a_0^0 \cdot w_{0,0} + a_1^0 \cdot w_{1,0} + b^1$$

allgemein:

$$z_0^1 = b^1 + \sum_{j=0}^{n^1} a_j^0 \cdot w_{j,0}$$

$$a_j^l = \sigma(z_j^l)$$

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

1.3 Berechnung als Vektor

$$\vec{z}^l = \begin{pmatrix} z_0^l \\ z_1^l \\ ... \\ z_j^l \end{pmatrix} \qquad \vec{a}^l = \begin{pmatrix} a_0^l \\ a_1^l \\ ... \\ a_j^l \end{pmatrix} \qquad \vec{b}^l = \begin{pmatrix} b_0^l \\ b_1^l \\ ... \\ b_j^l \end{pmatrix} \qquad W^l = \begin{pmatrix} w_{0,0}^l & w_{1,0}^l & ... & w_{j,0}^l \\ w_{0,1}^l & w_{1,1}^l & ... & w_{j,1}^l \\ ... & ... & ... & ... \\ w_{0,i}^l & w_{1,i}^l & ... & w_{j,i}^l \end{pmatrix}$$

$$\vec{a}^l = W^l \cdot \vec{z}^{l-1} + \vec{b}^l$$

$$\begin{pmatrix} a_0^l \\ a_1^l \\ ... \\ a_j^l \end{pmatrix} = \begin{pmatrix} z_0^{l-1} \\ z_1^{l-1} \\ ... \\ z_j^{l-1} \end{pmatrix} \cdot \begin{pmatrix} w_{0,0}^l & w_{1,0}^l & ... & w_{j,0}^l \\ w_{0,1}^l & w_{1,1}^l & ... & w_{j,1}^l \\ ... & ... & ... & ... \\ w_{0,i}^l & w_{1,i}^l & ... & w_{j,i}^l \end{pmatrix} + \begin{pmatrix} b_0^l \\ b_1^l \\ ... \\ b_j^l \end{pmatrix}$$

$$\begin{pmatrix} a_0^l \\ a_1^l \\ ... \\ a_j^l \end{pmatrix} = \begin{pmatrix} w_{0,0}^l \cdot z_0^{l-1} + w_{1,0}^l \cdot z_1^{l-1} + ... + w_{j,0}^l \cdot z_j^{l-1} \\ w_{0,1}^l \cdot z_0^{l-1} + w_{1,1}^l \cdot z_1^{l-1} + ... + w_{j,1}^l \cdot z_j^{l-1} \\ ... \\ w_{0,i}^l \cdot z_0^{l-1} + w_{1,i}^l \cdot z_1^{l-1} + ... + w_{j,i}^l \cdot z_j^{l-1} \end{pmatrix} + \begin{pmatrix} b_0^l \\ b_1^l \\ ... \\ b_j^l \end{pmatrix}$$

2 Backpropagation

2.1 Verlustfunktion

$$E_j = \frac{1}{2}(y_j - a_j^L)^2$$

$$E = \frac{1}{2}\sum_j (y_j - a_j^L)^2$$

$$\frac{dE_j}{da_j^L} = (a_j^L - y_j)$$

2.2 Lernvorgang

Anpassen der Gewichts- und BiasNeuronen:

$$w^l \rightarrow w^l - \frac{\eta}{N} \cdot \sum_{k=0}^{N} \frac{\partial C^k}{\partial w^l}$$

$\eta =$ Lernrate $\quad N =$ Anzahl der Trainingsbeispiele $\quad k =$ Trainingsbeispiel

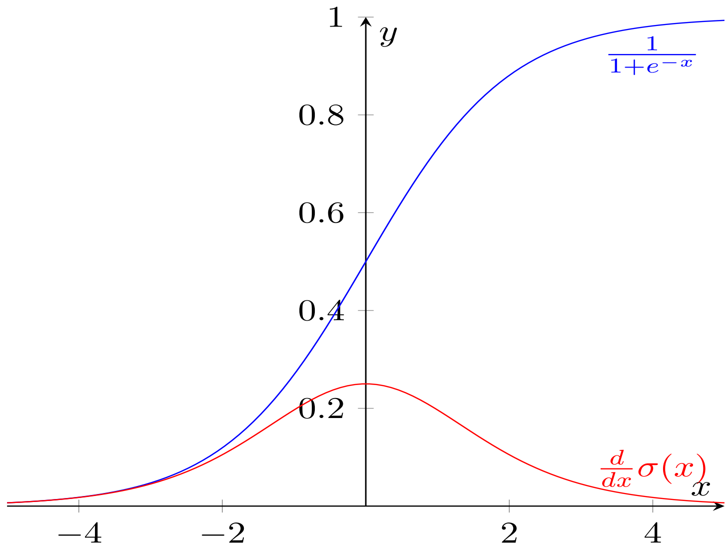
2.3 Ableitung der sigmoid Funktion

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

$$\frac{d}{dx}\sigma(x) = \frac{0(\cdot 1 + e^{-x}) - 1 \cdot e^{-x} \cdot -1}{(1+e^{-x})^2} = \frac{e^{-x}}{(1+e^{-x})^2} = \frac{1+e^{-x}-1}{(1+e^{-x})^2}$$

$$= \frac{1+e^{-x}}{(1+e^{-x})^2} - \frac{1}{(1+e^{-x})^2} = \frac{1}{1+e^{-x}} - \frac{1}{(1+e^{-x})^2}$$

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



2.4 Fehler(δ)

-Zwischenwert zur leichteren Berechnung des Gradienten

$$\delta_j^l = \frac{\partial E}{\partial z_j^l} \qquad \frac{\partial E}{\partial w_{j,i}^l} = \delta_i^l \cdot a_j^{l-1} \qquad \frac{\partial E}{\partial b_j^l} = \delta_j^l$$

Berechnung des Fehlers des letzten Layers durch Anwendung der Kettenregel:

$$\delta_j^L = \frac{\partial E}{\partial z_j^L} = \frac{\partial E}{\partial a_j^L} \cdot \frac{\partial a_j^L}{\partial z_j^L}$$

Einsetzen der Ableitungen:

$$= \frac{\partial E}{\partial a_j^L} \cdot \sigma(z_j^L) \cdot (1 - \sigma(z_j^L))$$

$$= (a_j^L - y_j) \cdot \sigma(z_j^L) \cdot (1 - \sigma(z_j^L))$$

Berechnung des Fehlers des vorletzten Layers:

$$\delta_j^{L-1} = \frac{\partial E}{\partial z_j^{L-1}} = {\color{green}\frac{\partial E}{\partial a_j^{L-1}}} \cdot {\color{blue}\frac{\partial a_j^{L-1}}{\partial z_j^{L-1}}}$$

$${\color{blue}\frac{\partial a_j^{L-1}}{\partial z_j^{L-1}} = \sigma'(z_j^{L-1})}$$

$${\color{green}\frac{\partial E}{\partial a_j^{L-1}} = \sum_{i=0}^{n^L} \frac{\partial E}{\partial z_i^L} \cdot \frac{\partial z_i^L}{\partial a_j^{L-1}}} = \sum_{i=0}^{n^L} \delta_i^L \cdot {\color{red}\frac{\partial z_i^L}{\partial a_j^{L-1}}}$$

$$\frac{\partial z_i^L}{\partial a_j^{L-1}} = \frac{\partial}{\partial a_j^{L-1}}(\sum_{p=0}^{n^L} a_p^{L-1} \cdot w_{p,i}^L + b_p^L) = w_{j,i}^L$$

$$\frac{\partial E}{\partial a_j^{L-1}} = \sum_{i=0}^{n^L} \delta_i^L \cdot w_{j,i}^L$$

Zusammenführen der Einzelergebnisse:

$$\delta_j^{L-1} = [\sum_{i=0}^{n^L} \delta_i^L \cdot w_{j,i}^L] \cdot \sigma'(z_j^{L-1})$$

Allgemein:

$$\delta_j^l = [\sum_{i=0}^{n^{l+1}} \delta_i^{l+1} \cdot w_{j,i}^{l+1}] \cdot \sigma'(z_j^l)$$

2.5 Berechnung des Fehlers als vektor

2.5.1 Fehler des letzten Layers

$$\delta_j^L = (z_j^L - y_j) \cdot \sigma'(a_j^L)$$

$$\vec{\delta^L} = \begin{pmatrix} \delta_0^L \\ \delta_1^L \\ ... \\ \delta_j^L \end{pmatrix} \qquad \vec{y} = \begin{pmatrix} y_0 \\ y_1 \\ ... \\ y_j \end{pmatrix} \qquad \vec{a^L} = \begin{pmatrix} a_0^L \\ a_1^L \\ ... \\ a_j^L \end{pmatrix} \qquad \vec{z^L} = \begin{pmatrix} z_0^L \\ z_1^L \\ ... \\ z_j^L \end{pmatrix}$$

$$\vec{\delta^L} = (\vec{z^L} - \vec{y^L}) \odot \sigma'(\vec{a^L})$$

2.5.2 Definition $\odot$

$$\begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} \odot \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{pmatrix} = \begin{pmatrix} a_0 \cdot b_0 \\ a_1 \cdot b_1 \\ a_2 \cdot b_2 \\ a_3 \cdot b_3 \end{pmatrix}$$

2.5.3 Fehler eines beliebigen Layers

$$\delta_j^l = [\sum_{i=0}^{n^{l+1}} \delta_i^{l+1} \cdot w_{j,i}^{l+1}] \cdot \sigma'(z_j^l)$$

$$\delta_0^l = (\delta_0^{l+1} \cdot w_{0,0}^{l+1} + \delta_1^{l+1} \cdot w_{0,1}^{l+1} + ... + \delta_i^{l+1} \cdot w_{0,i}^{l+1}) \cdot \sigma'(z_0^l)$$

$$\delta_1^l = (\delta_0^{l+1} \cdot w_{1,0}^{l+1} + \delta_1^{l+1} \cdot w_{1,1}^{l+1} + ... + \delta_i^{l+1} \cdot w_{1,i}^{l+1}) \cdot \sigma'(z_1^l)$$

$$W^{l+1} = \begin{pmatrix} w_{0,0}^{l+1} & w_{1,0}^{l+1} & ... & w_{j,0}^{l+1} \\ w_{0,1}^{l+1} & w_{1,1}^{l+1} & ... & w_{j,1}^{l+1} \\ ... & ... & ... & ... \\ w_{0,i}^{l+1} & w_{1,i}^{l+1} & ... & w_{j,i}^{l+1} \end{pmatrix} \qquad (W^{l+1})^T = \begin{pmatrix} w_{0,0}^{l+1} & w_{0,1}^{l+1} & ... & w_{0,i}^{l+1} \\ w_{1,0}^{l+1} & w_{1,1}^{l+1} & ... & w_{1,i}^{l+1} \\ ... & ... & ... & ... \\ w_{j,0}^{l+1} & w_{j,1}^{l+1} & ... & w_{j,i}^{l+1} \end{pmatrix}$$

$$\Rightarrow \vec{\delta^l} = (W^{l+1})^T \cdot \vec{\delta^{l+1}} \odot \sigma'(\vec{z^l})$$

2.6 Feed-Forward Berechnung als Matriz

$$A^l = \begin{pmatrix} a_0^{l,0} & a_0^{l,1} & ... & a_0^{l,k} \\ a_1^{l,0} & a_1^{l,1} & ... & a_1^{l,k} \\ ... & ... & ... & ... \\ a_j^{l,0} & a_j^{l,1} & ... & a_j^{l,k} \end{pmatrix}$$

$$A^l = \sigma(W^l \cdot A^{[l-1]} + \vec{b^l})$$

Erklärung Matrix + vektor

$$\begin{pmatrix} a_0 & a_1 & a_2 \\ b_0 & b_1 & b_2 \\ c_0 & c_1 & c_2 \end{pmatrix} + \begin{pmatrix} d \\ e \\ f \end{pmatrix} = \begin{pmatrix} a_0 + d & a_1 + d & a_2 + d \\ b_0 + e & b_1 + e & b_2 + e \\ c_0 + f & c_1 + f & c_2 + f \end{pmatrix}$$

2.7 Backpropagation als Matriz

2.7.1 Fehlerberechnung

$$[\delta^l] = \begin{pmatrix} \delta_0^{l,0} & \delta_0^{l,1} & ... & \delta_0^{l,k} \\ \delta_1^{l,0} & \delta_1^{l,1} & ... & \delta_1^{l,k} \\ ... & ... & ... & ... \\ \delta_j^{l,0} & \delta_j^{l,1} & ... & \delta_j^{l,k} \end{pmatrix}$$

$$[\delta^L] = (Z^L - Y^L) \odot \sigma'(A^L)$$

$$[\delta^l] = (W^{l+1})^T \cdot [\delta^{l+1}] \odot \sigma'(A^l)$$

$$\Rightarrow \frac{\partial E}{B^l} = [\delta^l]$$

2.7.2 Berechnung der Gradienten der Gewichte

$$[\delta^l] = \begin{pmatrix} \delta_0^{l,0} & \delta_0^{l,1} & ... & \delta_0^{l,k} \\ \delta_1^{l,0} & \delta_1^{l,1} & ... & \delta_1^{l,k} \\ ... & ... & ... & ... \\ \delta_j^{l,0} & \delta_j^{l,1} & ... & \delta_j^{l,k} \end{pmatrix} \qquad A^{l-1} = \begin{pmatrix} a_0^{l-1,0} & a_0^{l-1,1} & ... & a_0^{l-1,k} \\ a_1^{l-1,0} & a_1^{l-1,1} & ... & a_1^{l-1,k} \\ ... & ... & ... & ... \\ a_j^{l-1,0} & a_j^{l-1,1} & ... & a_j^{l-1,k} \end{pmatrix}$$

$$W^l = \begin{pmatrix} w_{0,0}^l & w_{1,0}^l & ... & w_{j,0}^l \\ w_{0,1}^l & w_{1,1}^l & ... & w_{j,1}^l \\ ... & ... & ... & ... \\ w_{0,i}^l & w_{1,i}^l & ... & w_{j,i}^l \end{pmatrix}$$

$$\frac{\partial E^k}{\partial w_{j,i}^{l,k}} = \delta_i^{l,k} \cdot a_j^{l-1,k}$$

$$\frac{\partial E}{\partial w_{0,0}^l} = \sum_{k=0}^{N} \delta_0^{l,k} \cdot a_0^{l-1,k}$$

$$\frac{\partial E}{\partial w_{1,0}^l} = \sum_{k=0}^{N} \delta_0^{l,k} \cdot a_1^{l-1,k}$$

$$\Rightarrow \frac{\partial E}{W^l} = [\delta^l] \cdot (A^{l-1})^T$$

3 Quellen

Michael Nielsen - Neural Networks and Deep Learning(Algorithmus)
Michael Kipp - Neurale Netze und Deep Learning (Algorithmus)
Sudeep Raja - A Derivation of Backpropagation in Matrix Form (Prüfen der Matrizenformeln)