

Feedward Neural Networks

Benedikt Zönnchen

22. Februar 2023

Motivation

Motivation

Beispiel (Problem)

Wir wollen die Art eines Kleidungsstücks auf einem Bild erkennen. Wie lösen wir diese Aufgabe?

Motivation

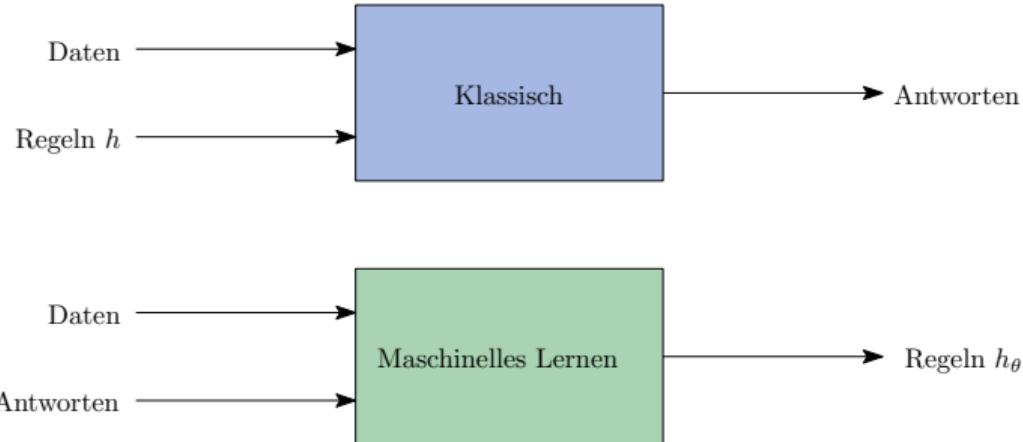
Wir wollen eine Funktion h , die uns für eine Eingabe \mathbf{x} (ein $m \times n$ -Pixelbild) ausgibt, welches Kleidungsstück \mathbf{x} ist.

$$h(\mathbf{x}) = \text{'T-Shirt'}$$

$$h : [0; 1]^{m \times n} \rightarrow \{\text{'T-Shirt'}, \text{'Pullover'}, \dots, \text{'Sneaker'}\}.$$

Motivation

- (1) Klassisch: Konstruiere Algorithmus h der die Antwort berechnet
- (2) **Überwachtes Maschinelles Lernen:** Konstruiere eine parametrisierbaren Algorithmus h_θ , dessen Parameter θ durch Antworten kalibriert werden



Diese Kalibrierung nennen wir auch **Lernen**.

Motivation

Definition (Maschinelles Lernen)

Ein Computerprogramm lernt von der Erfahrung \mathcal{E} hinsichtlich einer Aufgabe \mathcal{T} und einem Leistungskriterium \mathcal{P} , wenn sich seine Leistung bei der Durchführung von \mathcal{T} , wie sie von \mathcal{P} gemessen wurde, mit der Erfahrung \mathcal{E} verbessert [2].

Computationalism

Computationalism

“Brains have the same computational capacity as Turing machines.”
– Hilary Putnam (1975)

Computationalism

Falls dies zutrifft ist es zumindest theoretisch möglich ein funktionierendes Gehirn wie das unsere aus einem ganz anderen Material zu bauen.

⇒ Es kommt rein auf die 'Software' an, welche die (mental)en Zustände berechnet.

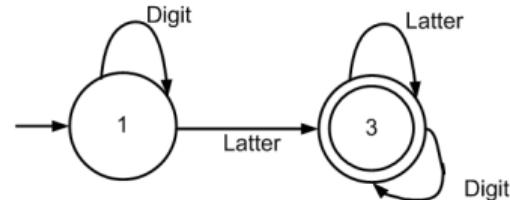
Wichtiges Gedankenexperiment: *Das chinesische Zimmer*.

Computationalism

The Symbol-system Paradigm

Das Gehirn als automatisches formales System (Hypothese) [1]:

- Unsere mentale Fähigkeiten lassen sich zumindest teilweise auf automatische formale Systeme zurückführen.
- Mentale Zustände sind Repräsentationen
- Es gibt eine Art **Sprache des Denkens**



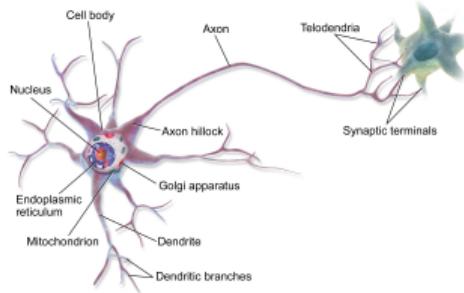
Schach ist beispielsweise ein formales System.

Computationalism

The Connectionist Computational Paradigm

Das Gehirn als vernetztes Netzwerk (Hypothese) [1]:

- Untersuchung von funktionalen Architekturen die durch neuronale Netze inspiriert sind
- Anstatt einer zentralen Recheneinheit gibt es ein Netzwerk aus Knoten
- Jeder Knoten nimmt an der Informationsverarbeitung teil (verteilt und parallel)



Computationalism

The Connectionist Computational Paradigm

FIG. 1 — Organization of a biological brain. (Red areas indicate active cells, responding to the letter X.)

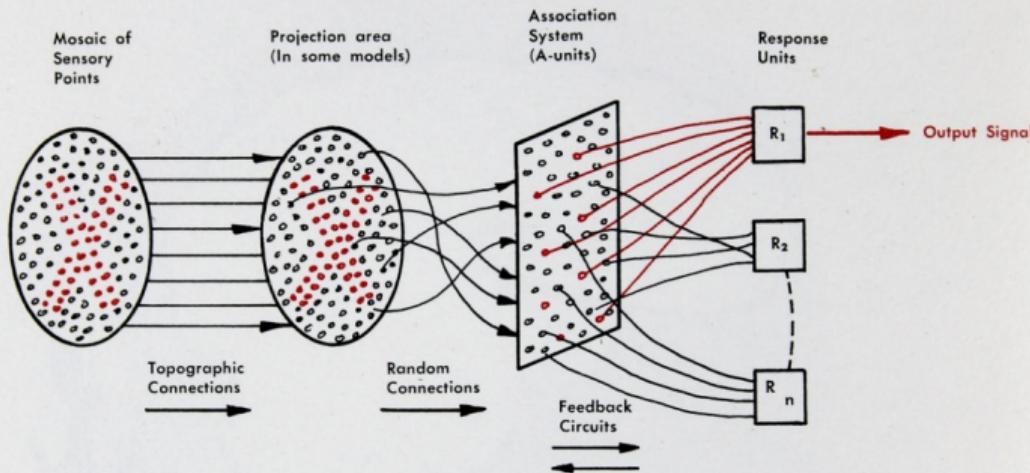
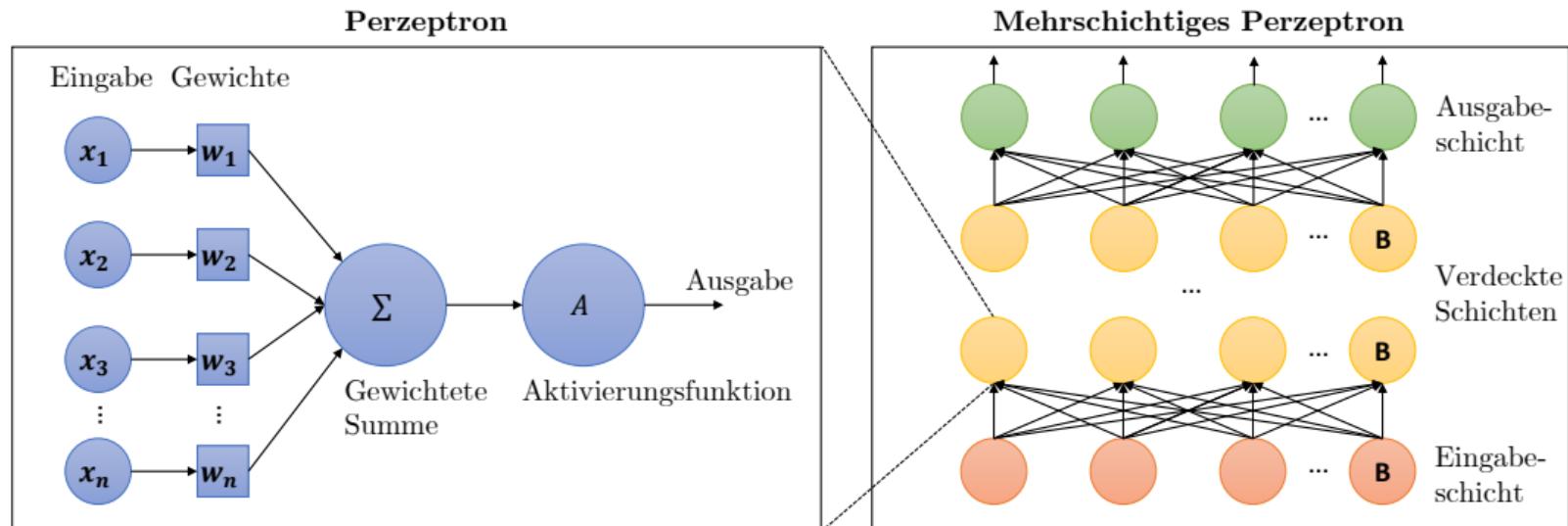


FIG. 2 — Organization of a perceptron.

Abbildung: Von Frank Rosenblatt

Feedward Neural Networks

Feedward Neural Networks



Feedward Neural Networks

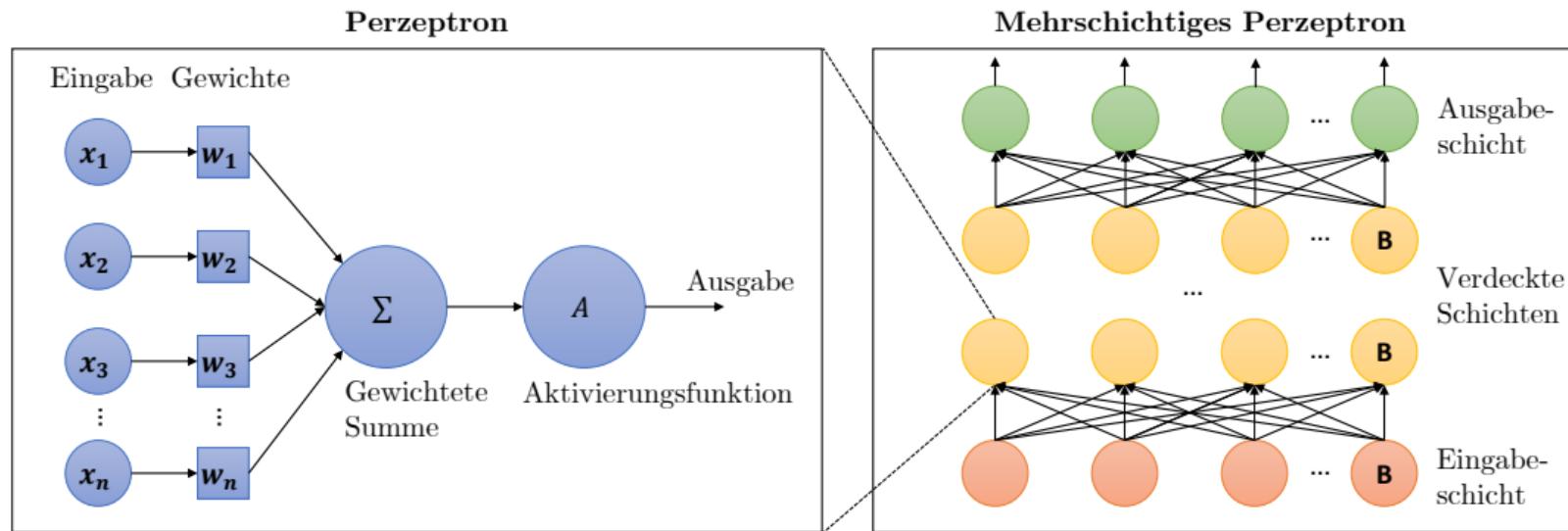
Ein paar Fakten:

- Erste Einführung im Jahr 1943 und erste Erfolge bis in die 1990er
- Neuronale Netze sind abstrakte, vereinfachte Umsetzungen der Verknüpfung biologischer Neuronen.
- Sie können im Prinzip **jede Funktion** ($\mathbb{R}^m \rightarrow \mathbb{R}^m$) annähern!
- Ausgabe ist **probabilistisch!**

Feedward Neural Networks

Perzeptron

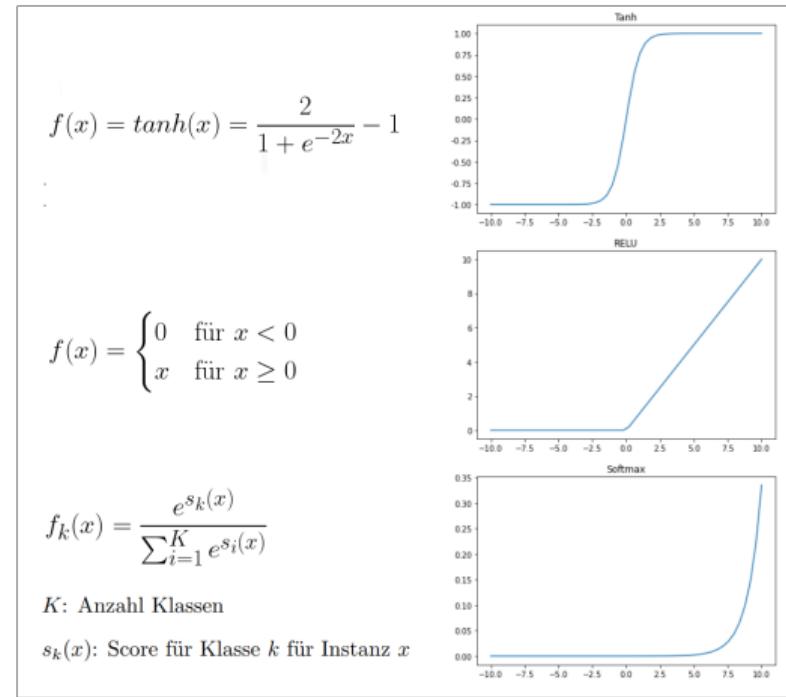
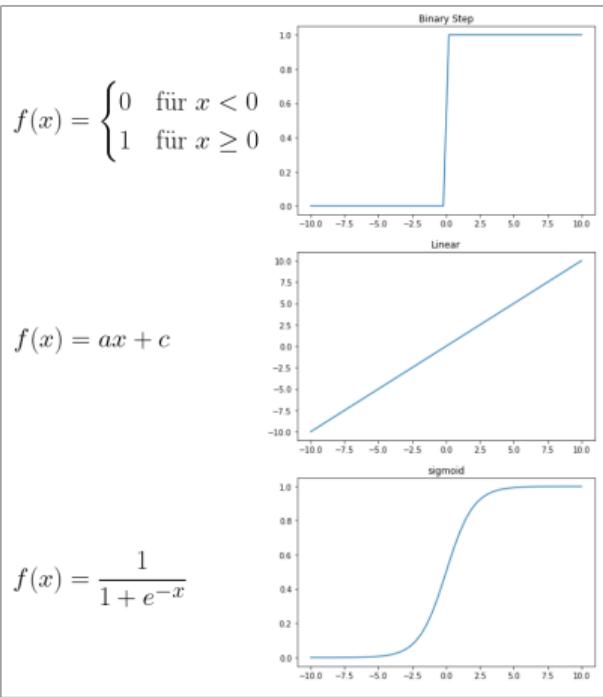
Basiselement eines neuronalen Netzes ist das Perzeptron, das zu mehrschichtigen Architekturen (daher das Deep in Deep Learning) kombiniert werden kann.



Feedward Neural Networks

Aktivierungsfunktion

Durch die Aktivierungsfunktion wird die Nicht-Linearität ermöglicht.



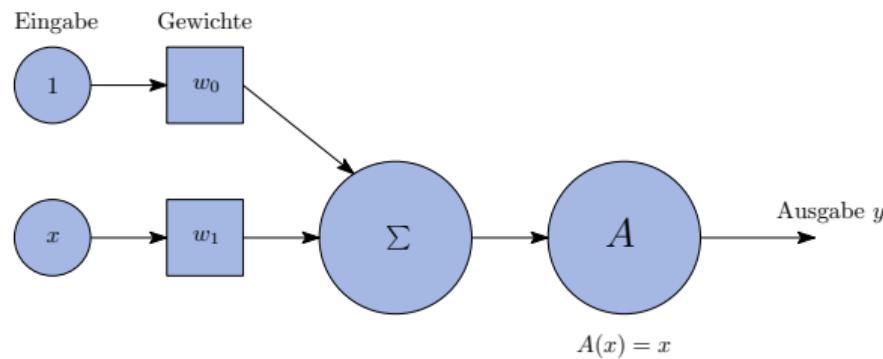
Feedward Neural Networks

Einfachstes Neuronales Netz

Eine (parametrisierbare) Geradengleichung

$$h_{\theta}(x) = w_0 \cdot x + w_1$$

mit den Gewichten $w_0, w_1 \in \theta$ ist im Prinzip ein neuronales Netz:



Nehmen wir nun an wir hätten Datenpunkte (Antworten) $(x_1, y_1) = (0, 0)$ und $(x_2, y_2) = (1, 1)$ gegeben.

Feedward Neural Networks

Training

Definition ((informell) Training)

Beim Training eines neuronalen Netzes suchen wir Parameter θ (hier w_0, w_1), sodass der Fehler zwischen unserem Netz hier (hier $h_\theta(x_1), h(x_2)$) und den Trainingsdatenpunkten (hier y_1, y_2) möglichst klein wird.

Wir kalibrieren unsere Neuronen sodass diese bei bestimmten Eingaben 'richtig' feuern.

Feedward Neural Networks

Training (Gradient Descent)

Wir haben Antworten: $(x_1, y_1) = (0, 0)$ und $(x_2, y_2) = (1, 1)$. Initialisiere Parameter zufällig, z.B.: $w_0 = 3, w_1 = 6$

$$h_{\theta}(x) = \underbrace{3}_{w_0} x + \underbrace{6}_{w_1}$$

Daraus ergeben sich Fehler:

$$y_1 - h_{\theta}(x_1) = y_1 - h_{\theta}(0) = 0 - 6 = -6$$

$$y_2 - h_{\theta}(x_2) = y_2 - h_{\theta}(1) = 1 - 9 = -8$$

Wir suchen die Parameter θ , sodass der Fehler (oder auch die Kostenfunktion)

$$J(\theta) = (y_1 - h_{\theta}(x_1))^2 + (y_2 - h_{\theta}(x_2))^2$$

minimal wird!

Feedward Neural Networks

Training (Gradient Descent)

Berechne Gradient des Fehlers

$$\begin{aligned} J(\theta) &= (y_1 - h_\theta(x_1))^2 + (y_1 - h_\theta(x_2))^2 \\ &= -\frac{1}{2}w_1^2 + \frac{1}{2}(1 - w_1 - w_0)^2 \end{aligned}$$

wobei wir nach den **Parametern** ableiten:

$$\nabla J(\theta) = \begin{pmatrix} \frac{\partial J}{\partial w_0} \\ \frac{\partial J}{\partial w_1} \end{pmatrix} = \begin{pmatrix} w_0 + w_1 - 1 \\ w_0 - 1 \end{pmatrix} = \begin{pmatrix} 8 \\ 2 \end{pmatrix}$$

Wir können nun iterativ neue **Parameter** berechnen:

$$\theta^{i+1} = \theta^i - \mu \cdot \nabla J(\theta^i)$$

bis $\nabla J(\theta^i) \approx 0$ wird.

Feedward Neural Networks

Kettenregel

Wir haben die Kostenfunktion

$$\begin{aligned} J(\theta) &= (\mathbf{y}_1 - h_\theta(\mathbf{x}_1))^2 + (\mathbf{y}_1 - h_\theta(\mathbf{x}_2))^2 + \dots + (\mathbf{y}_n - h_\theta(\mathbf{x}_m))^2 \\ &= \sum_{i=1}^m (\mathbf{y}_i - h_\theta(\mathbf{x}_i))^2 \end{aligned}$$

und wollen iterativ unsere Parameter θ anpassen:

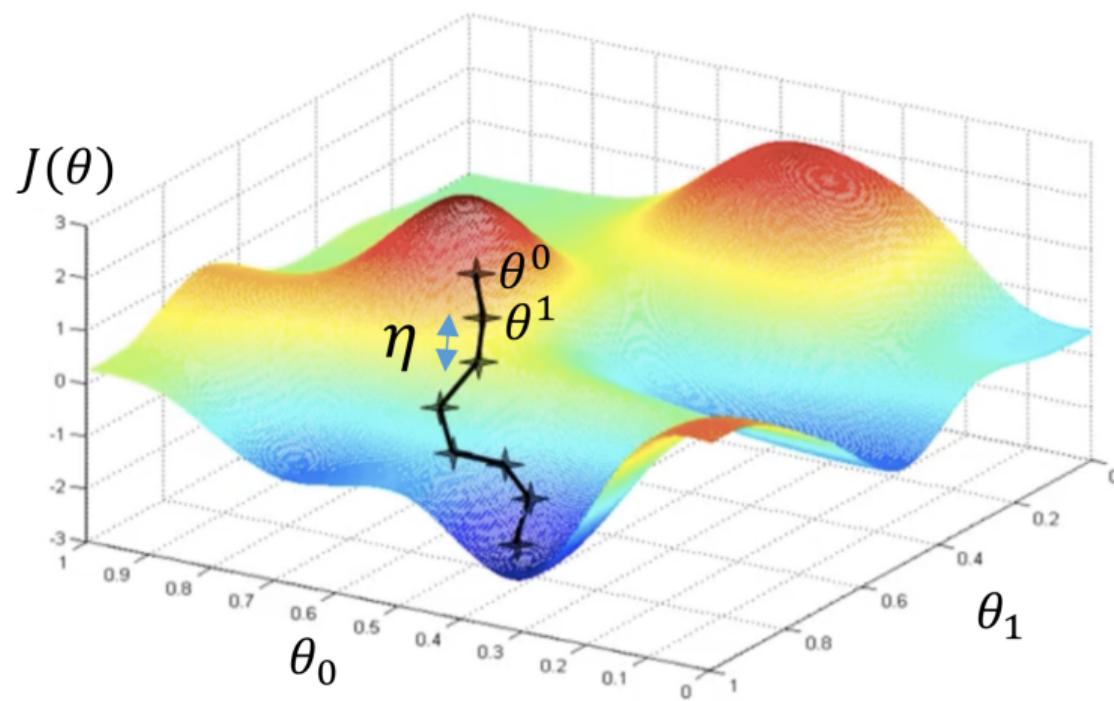
$$\theta^{i+1} = \theta^i - \mu \cdot \nabla J(\theta^i)$$

bis $\nabla J(\theta^i) \approx 0$ wird. Wir brauchen demnach die Gradienten

$$\nabla J(\theta^i)$$

Feedward Neural Networks

Training (Gradient Descent)



Training im Detail

Training im Detail

Ablauf

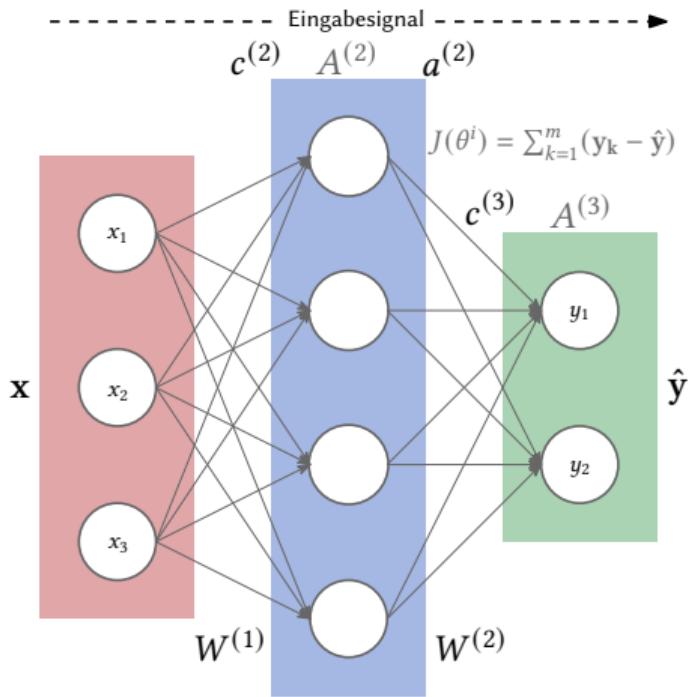
Trainingsablauf:

Training im Detail

Ablauf

Trainingsablauf:

(1) Vorhersage $\hat{y} = h_{\theta^i}(\mathbf{x})$ berechnen

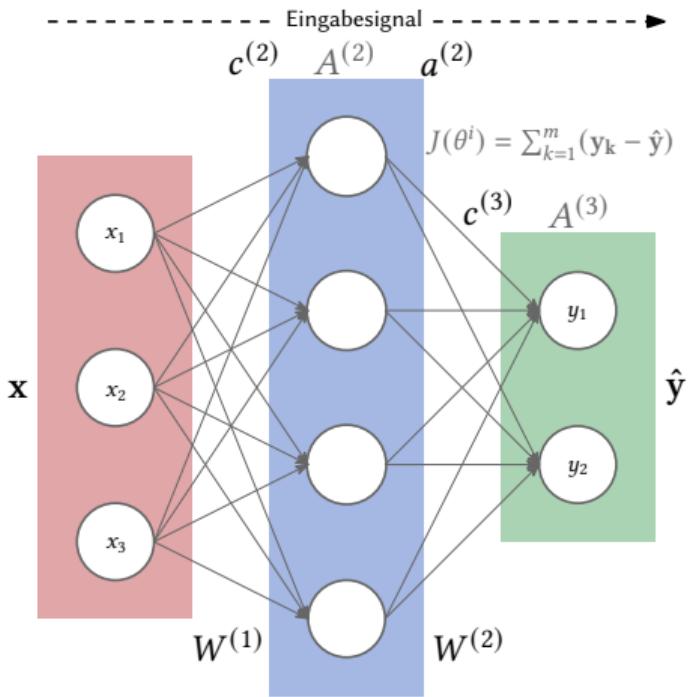


Training im Detail

Ablauf

Trainingsablauf:

- (1) Vorhersage $\hat{y} = h_{\theta^i}(\mathbf{x})$ berechnen
- (2) Fehler $J(\theta^i)$ berechnen $\nabla J(\theta^i)$ berechnen

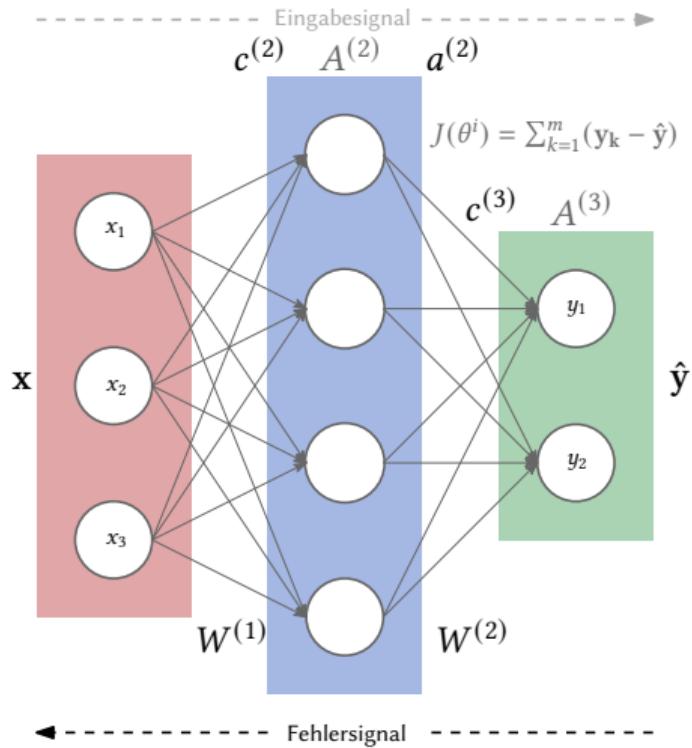


Training im Detail

Ablauf

Trainingsablauf:

- (1) Vorhersage $\hat{y} = h_{\theta^i}(\mathbf{x})$ berechnen
- (2) Fehler $J(\theta^i)$ berechnen $\nabla J(\theta^i)$ berechnen
- (3) Gradient(en) des Fehlers bestimmen

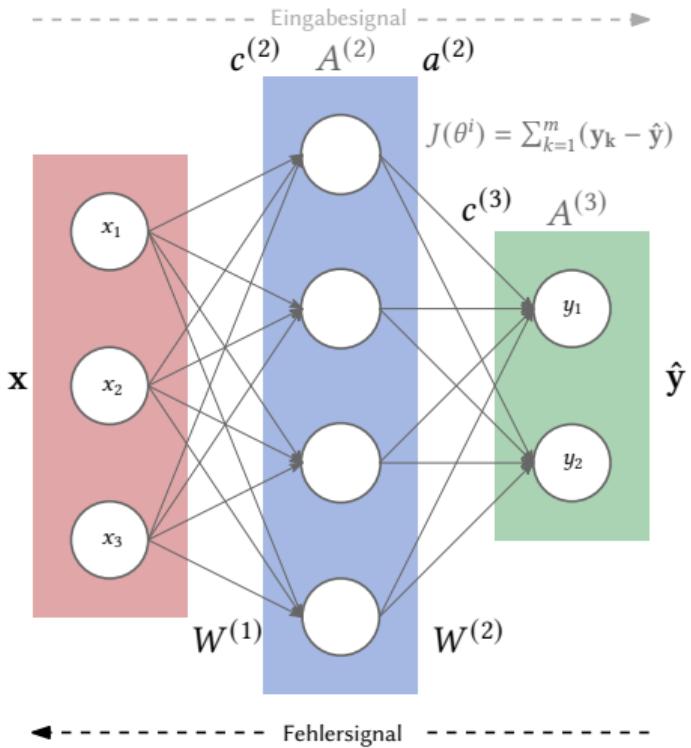


Training im Detail

Ablauf

Trainingsablauf:

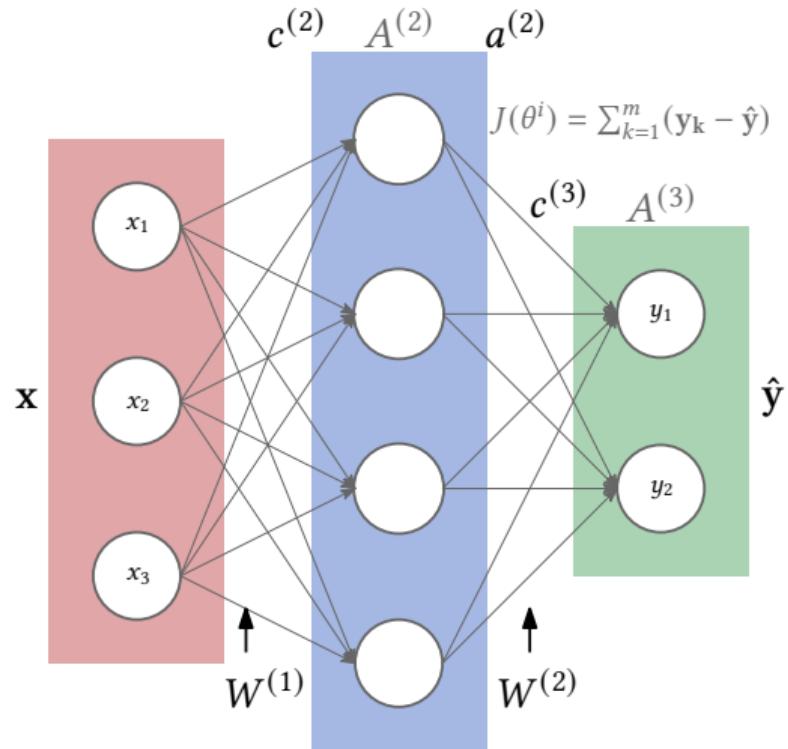
- (1) Vorhersage $\hat{y} = h_{\theta^i}(\mathbf{x})$ berechnen
- (2) Fehler $J(\theta^i)$ berechnen $\nabla J(\theta^i)$ berechnen
- (3) Gradient(en) des Fehlers bestimmen
- (4) Parameter anpassen
$$\theta^{i+1} = \theta^i - \mu \cdot \nabla J(\theta^i)$$



Training im Detail

Vorhersage berechnen

Vorhersage berechnen:

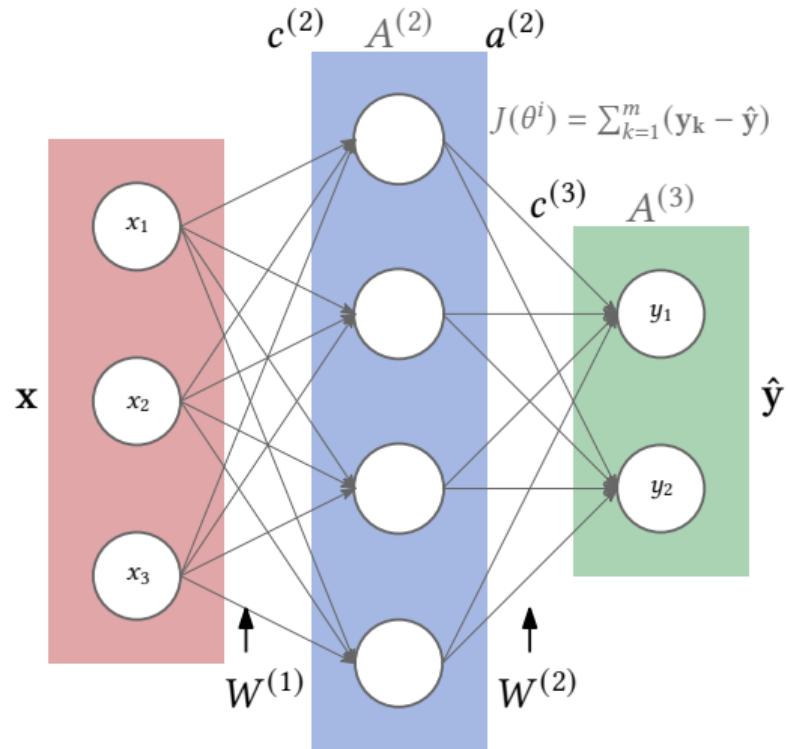


Training im Detail

Vorhersage berechnen

Vorhersage berechnen:

$$(1) \quad c^{(2)} = \mathbf{x}W^{(1)}, \text{ mit } W^{(1)} \in \mathbb{R}^{3 \times 4}$$



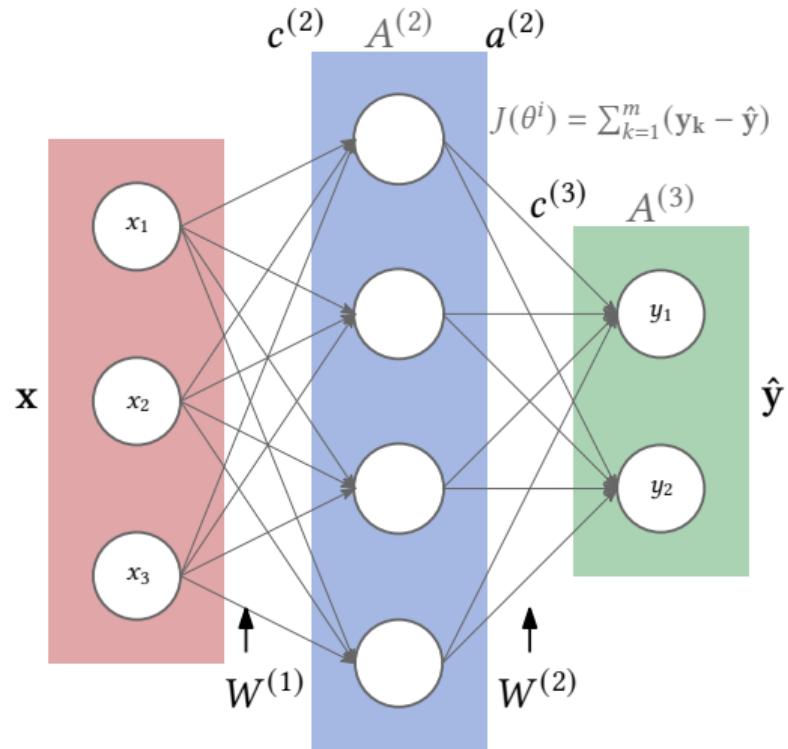
Training im Detail

Vorhersage berechnen

Vorhersage berechnen:

$$(1) \quad c^{(2)} = \mathbf{x}W^{(1)}, \text{ mit } W^{(1)} \in \mathbb{R}^{3 \times 4}$$

$$(2) \quad a^{(2)} = A^{(2)}(c^{(2)})$$

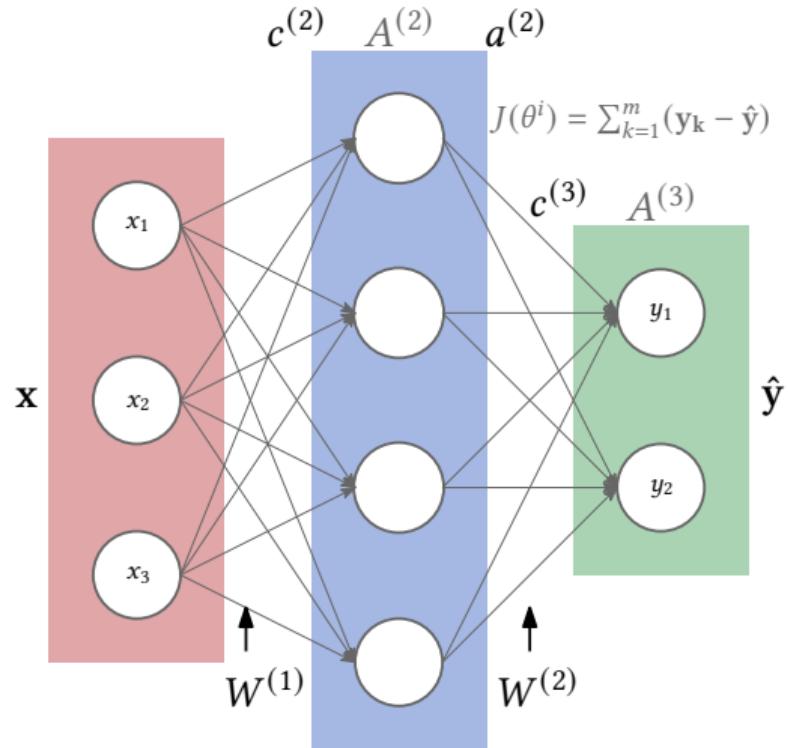


Training im Detail

Vorhersage berechnen

Vorhersage berechnen:

- (1) $c^{(2)} = \mathbf{x}W^{(1)}$, mit $W^{(1)} \in \mathbb{R}^{3 \times 4}$
- (2) $a^{(2)} = A^{(2)}(c^{(2)})$
- (3) $c^{(3)} = a^{(2)}W^{(2)}$, mit $W^{(2)} \in \mathbb{R}^{4 \times 2}$

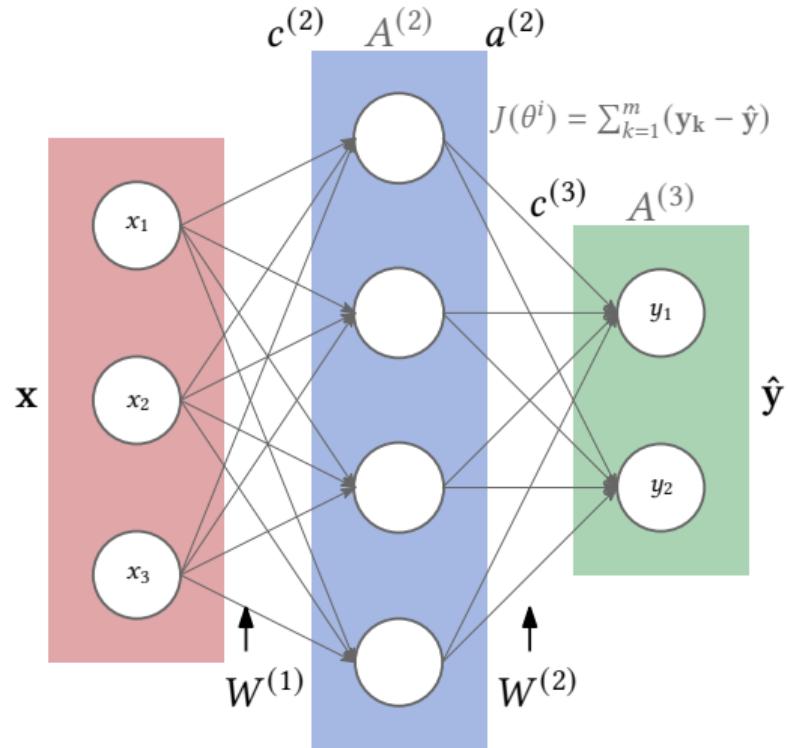


Training im Detail

Vorhersage berechnen

Vorhersage berechnen:

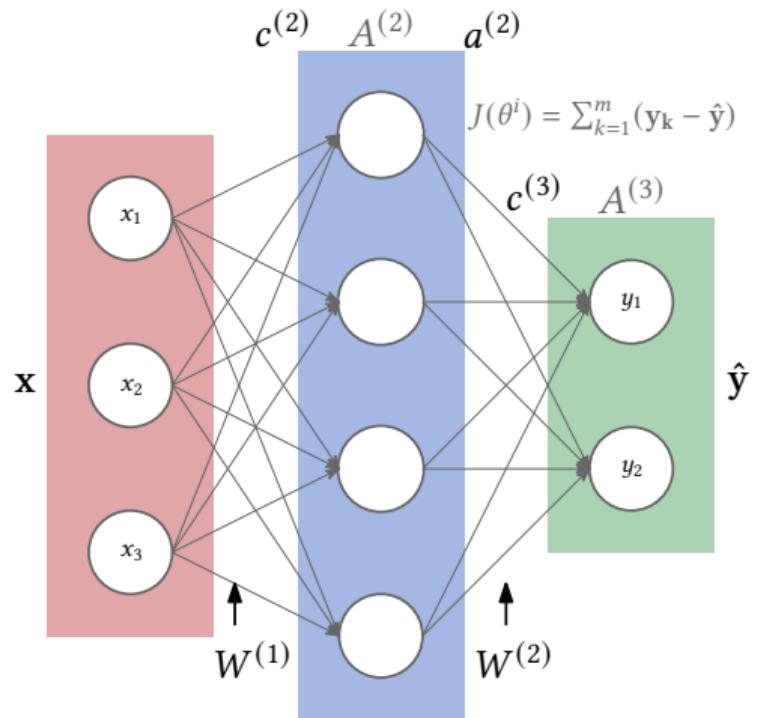
- (1) $c^{(2)} = \mathbf{x}W^{(1)}$, mit $W^{(1)} \in \mathbb{R}^{3 \times 4}$
- (2) $a^{(2)} = A^{(2)}(c^{(2)})$
- (3) $c^{(3)} = a^{(2)}W^{(2)}$, mit $W^{(2)} \in \mathbb{R}^{4 \times 2}$
- (4) $\mathbf{y} = A^{(3)}(c^{(3)})$



Training im Detail

Berechnung der Gradienten

Gradient(en) des Fehlers bestimmen:



Training im Detail

Berechnung der Gradienten

Gradient(en) des Fehlers bestimmen:

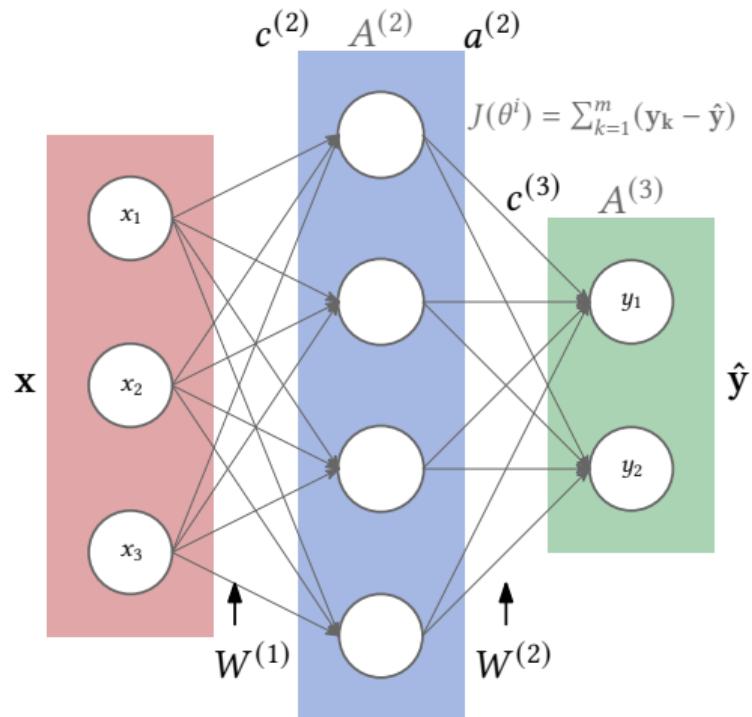
$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial q} \frac{\partial q}{\partial x}$$

Ist die Magie der Kettenregel. Beispiel:

$$f(x, y) = (x + y)^2 \text{ wir setzen } q = (x + y).$$

Dann können wir rechnen:

$$\frac{\partial f}{\partial x} = \frac{\partial q^2}{\partial q} \frac{\partial q}{\partial x} = 2q \cdot \frac{\partial(x + y)}{\partial x} = 2(x + y)$$

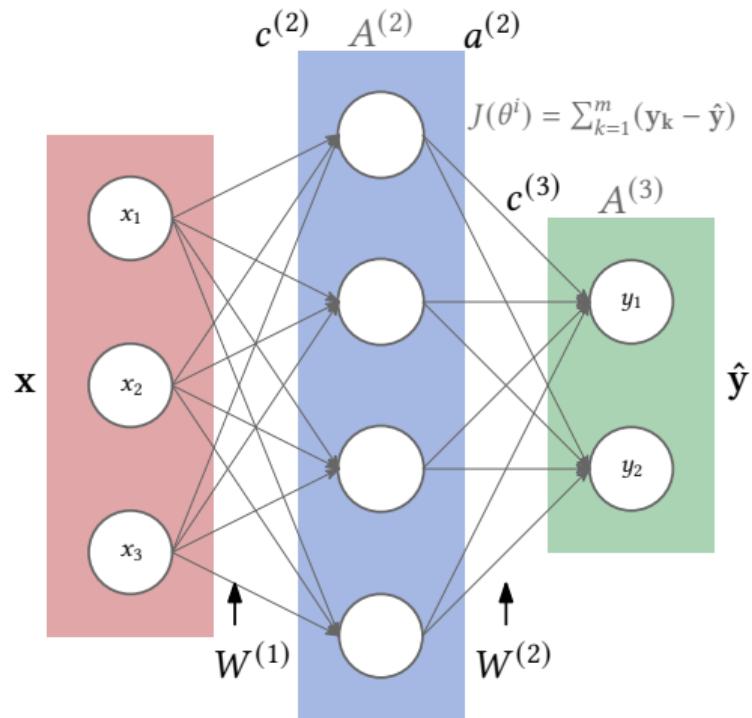


Training im Detail

Berechnung der Gradienten

Gradient(en) des Fehlers bestimmen:

$$\nabla J(\theta) \text{ d.h. } \frac{\partial J}{\partial W^{(k)}}$$

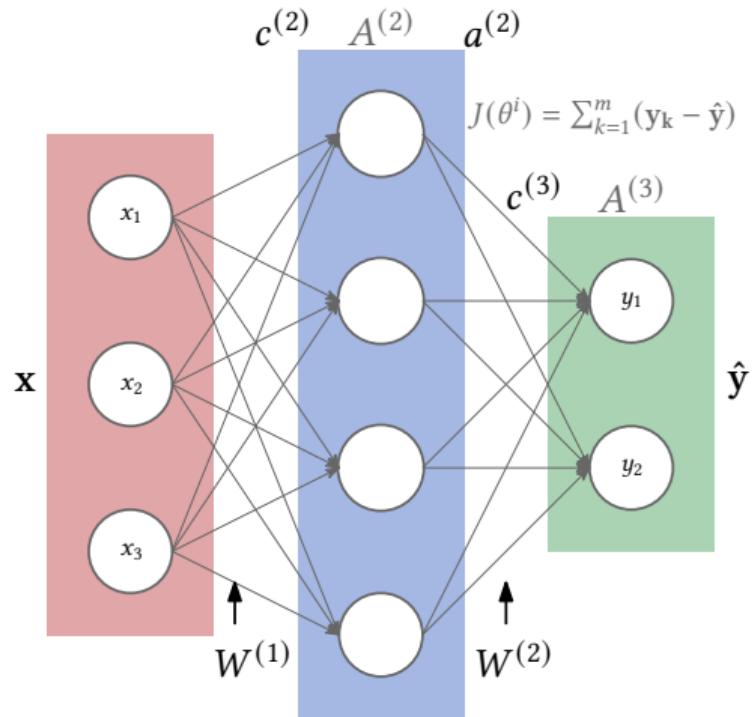


Training im Detail

Berechnung der Gradienten

Gradient(en) des Fehlers bestimmen:

$$\begin{aligned}\frac{\partial J}{\partial W^{(2)}} &= \frac{\partial J}{\partial \hat{\mathbf{y}}} \frac{\partial \hat{\mathbf{y}}}{\partial W^{(2)}} \\ &= \frac{\partial J}{\partial \hat{\mathbf{y}}} \frac{\partial \mathbf{y}^{(3)}}{\partial c^{(3)}} \frac{\partial c^{(3)}}{\partial W^{(2)}}\end{aligned}$$

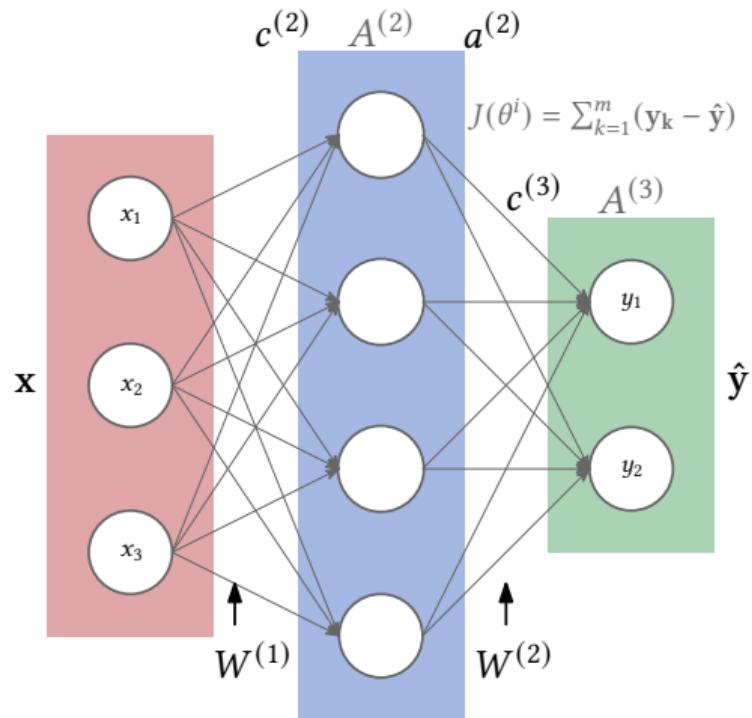


Training im Detail

Berechnung der Gradienten

Gradient(en) des Fehlers bestimmen:

$$\begin{aligned}\frac{\partial J}{\partial W^{(1)}} &= \frac{\partial J}{\partial a^{(2)}} \cdot \frac{\partial a^{(2)}}{\partial c^{(2)}} \frac{\partial c^{(2)}}{\partial W^{(1)}} \\ &= \frac{\partial J}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial c^{(3)}} \frac{\partial c^{(3)}}{\partial a^{(2)}} \cdot \frac{\partial a^{(2)}}{\partial c^{(2)}} \frac{\partial c^{(2)}}{\partial W^{(1)}}\end{aligned}$$



Feedward Neural Networks

Training (Gradient Descent)

Probleme:

- Black Box Problem
- Berechnungsaufwand (besonders beim Training) \Rightarrow hoher Ressourcenverbrauch
- Je komplexer das Netz desto mehr Daten werden fürs Training gebraucht
- Keine Garantie für Korrektheit
- Können kaum mit zeitlichen Abhängigkeiten umgehen (Sprache, Melodien, usw.)
- Qualität hängt von den Trainingsdaten und der ausgewählten Architektur zusammen (induzierte Verzerrung)

Feedward Neural Networks

Networks and Brains

“Although neural networks may initially seem biologically plausible as they ‘take inspiration from the brain’ many of the established techniques used by neural networks modellers (especially the backpropagation of errors) are not biologically realistic.” – A. Browne

Referenzen I

- [1] McLaughlin, B. P. (2008). Computationalism, connectionism, and the philosophy of mind.
- [2] Mitchell, T. M. (1997). *Machine Learning*. McGraw-Hill, New York.