

Algorithmik zur Optimierung in neuronalen Netzwerken

Gradient Descent und Backpropagation

Tim Hilt

Date: tbd

Hochschule Esslingen — University of Applied Sciences

Training

- Loss-Funktion

- Gradient Descent

- Backpropagation

- Umsetzung in Keras

Training

- Dient zur Berechnung des Fehlers während dem Training
- Trainingsfehler soll minimiert werden
- \Rightarrow wir suchen den Punkt, an dem die Ableitung der Loss-Funktion 0 wird, der Fehler also nicht mehr abnimmt
- Es gibt eine Vielzahl an Loss-Funktionen, wir betrachten hier die „**Mean Squared Error (MSE)**“:

$$C(w, b) = \frac{1}{2m} \sum_{x=1}^m (y(x) - \hat{y}(x))^2$$

Loss-Funktion

- Dient zur Berechnung des Fehlers während dem Training
- Trainingsfehler soll minimiert werden
- \Rightarrow wir suchen den Punkt, an dem die Ableitung der Loss-Funktion 0 wird, der Fehler also nicht mehr abnimmt
- Es gibt eine Vielzahl an Loss-Funktionen, wir betrachten hier die „**Mean Squared Error (MSE)**“:

$$C(w, b) = \frac{1}{2m} \sum_{x=1}^m (y(x) - \hat{y}(x))^2$$

$C(w, b)$	Cost in Abhängigkeit von w und b
m	Anzahl der Trainingsinstanzen
$y(x)$	Gewünschter Output wenn x Input ist
$\hat{y}(x)$	Tatsächlicher Output des Netzwerkes

Gradient Descent

- Methode um die Weights w und Biases b zu optimieren
- Vorgehen:

Gradient Descent

- Methode um die Weights w und Biases b zu optimieren
- Vorgehen:
 1. Finde die Änderungsrate des Fehlers in Abhängigkeit von den Weights und Biases
($\partial C / \partial w$; $\partial C / \partial b$)

Gradient Descent

- Methode um die Weights w und Biases b zu optimieren
- Vorgehen:
 1. Finde die Änderungsrate des Fehlers in Abhängigkeit von den Weights und Biases ($\partial C/\partial w; \partial C/\partial b$)
 2. Multipliziere die Änderungsrate mit der Lernrate η

Gradient Descent

- Methode um die Weights w und Biases b zu optimieren
- Vorgehen:
 1. Finde die Änderungsrate des Fehlers in Abhängigkeit von den Weights und Biases ($\partial C/\partial w; \partial C/\partial b$)
 2. Multipliziere die Änderungsrate mit der Lernrate η
 3. Ziehe das Produkt aus Änderungsrate und Lernrate von den aktuellen Parametern ab

Gradient Descent

- Methode um die Weights w und Biases b zu optimieren
- Vorgehen:
 1. Finde die Änderungsrate des Fehlers in Abhängigkeit von den Weights und Biases ($\partial C / \partial w$; $\partial C / \partial b$)
 2. Multipliziere die Änderungsrate mit der Lernrate η
 3. Ziehe das Produkt aus Änderungsrate und Lernrate von den aktuellen Parametern ab
 4. Aktualisiere die alten Parameter durch das Ergebnis des letzten Schrittes

$$w_{k+1} = w_k - \eta \frac{\partial C}{\partial w_k}$$

$$b_{k+1} = b_k - \eta \frac{\partial C}{\partial b_k}$$

Problem

Wie finde ich die Änderungsraten $\frac{\partial C}{\partial w}$; $\frac{\partial C}{\partial b}$, die ich für Gradient Descent benötige?

Problem

Wie finde ich die Änderungsraten $\frac{\partial C}{\partial w}$; $\frac{\partial C}{\partial b}$, die ich für Gradient Descent benötige?

⇒ **Idee:** Divide and conquer; Problem in kleinere, handhabbare Probleme zerlegen

Problem

Wie finde ich die Änderungsraten $\frac{\partial C}{\partial w}$; $\frac{\partial C}{\partial b}$, die ich für Gradient Descent benötige?

⇒ **Idee:** Divide and conquer; Problem in kleinere, handhabbare Probleme zerlegen

$$f(a, b, c, d) = ((a + b) \cdot (c + d))^2$$

Problem

Wie finde ich die Änderungsraten $\frac{\partial C}{\partial w}$; $\frac{\partial C}{\partial b}$, die ich für Gradient Descent benötige?

⇒ **Idee:** Divide and conquer; Problem in kleinere, handhabbare Probleme zerlegen

$$f(a, b, c, d) = ((a + b) \cdot (c + d))^2$$

$$g(a, b) = a + b$$

Problem

Wie finde ich die Änderungsraten $\frac{\partial C}{\partial w}$; $\frac{\partial C}{\partial b}$, die ich für Gradient Descent benötige?

⇒ **Idee:** Divide and conquer; Problem in kleinere, handhabbare Probleme zerlegen

$$f(a, b, c, d) = ((a + b) \cdot (c + d))^2$$

$$g(a, b) = a + b$$

$$h(c, d) = c + d$$

Problem

Wie finde ich die Änderungsraten $\frac{\partial C}{\partial w}$; $\frac{\partial C}{\partial b}$, die ich für Gradient Descent benötige?

⇒ **Idee:** Divide and conquer; Problem in kleinere, handhabbare Probleme zerlegen

$$f(a, b, c, d) = ((a + b) \cdot (c + d))^2$$

$$g(a, b) = a + b$$

$$h(c, d) = c + d$$

$$i(g, h) = g \cdot h$$

Problem

Wie finde ich die Änderungsraten $\frac{\partial C}{\partial w}$; $\frac{\partial C}{\partial b}$, die ich für Gradient Descent benötige?

⇒ **Idee:** Divide and conquer; Problem in kleinere, handhabbare Probleme zerlegen

$$f(a, b, c, d) = ((a + b) \cdot (c + d))^2$$

$$g(a, b) = a + b$$

$$h(c, d) = c + d$$

$$i(g, h) = g \cdot h$$

$$f(i) = i^2$$

Backpropagation

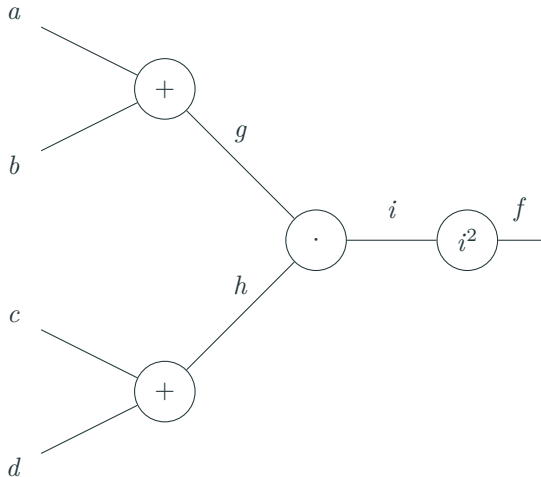
$$f(a, b, c, d) = ((a + b) \cdot (c + d))^2$$

$$g(a, b) = a + b$$

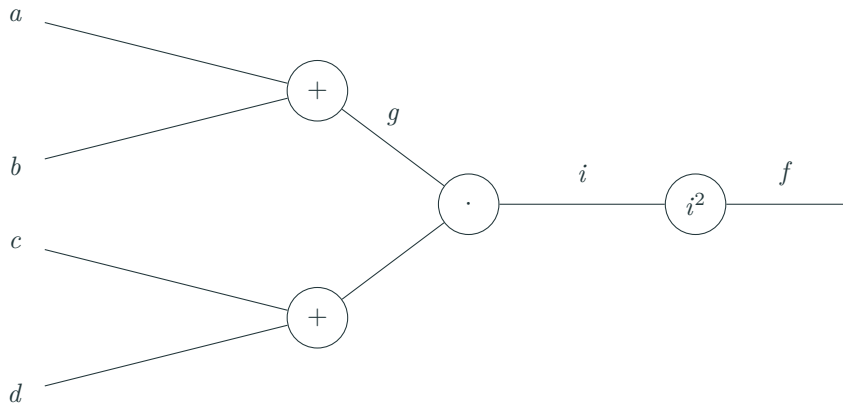
$$h(c, d) = c + d$$

$$i(g, h) = g \cdot h$$

$$f(i) = i^2$$

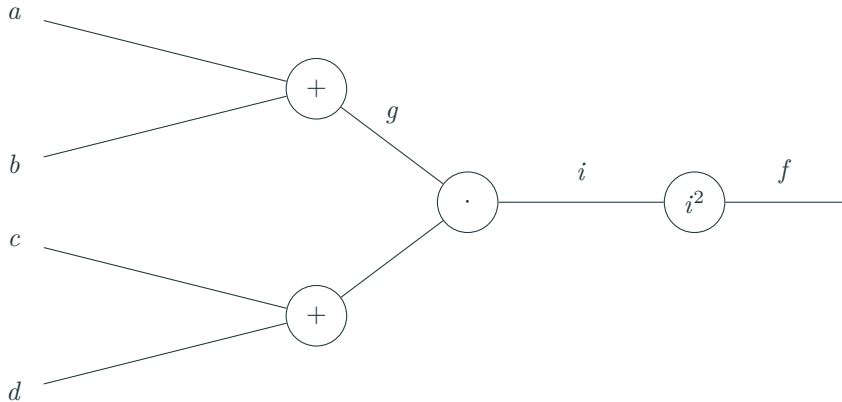


Backpropagation



Frage: $\frac{\partial f}{\partial a}$?

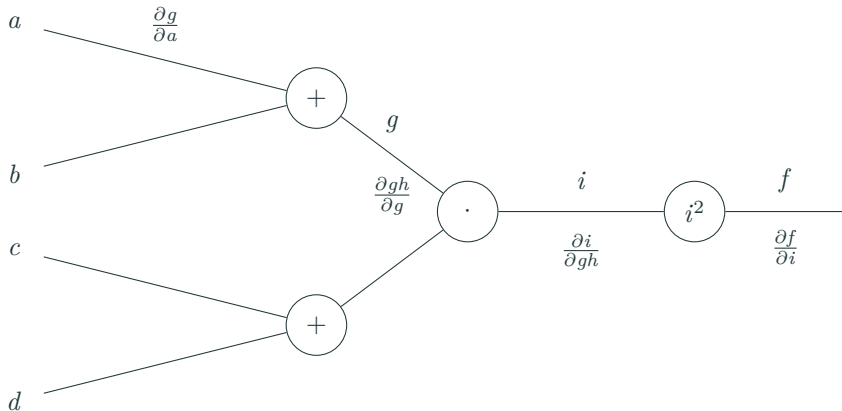
Backpropagation



Frage: $\frac{\partial f}{\partial a}$?

$$\frac{\partial f}{\partial a} = \frac{\partial f}{\partial i} \cdot \frac{\partial i}{\partial g} \cdot \frac{\partial g}{\partial a}$$

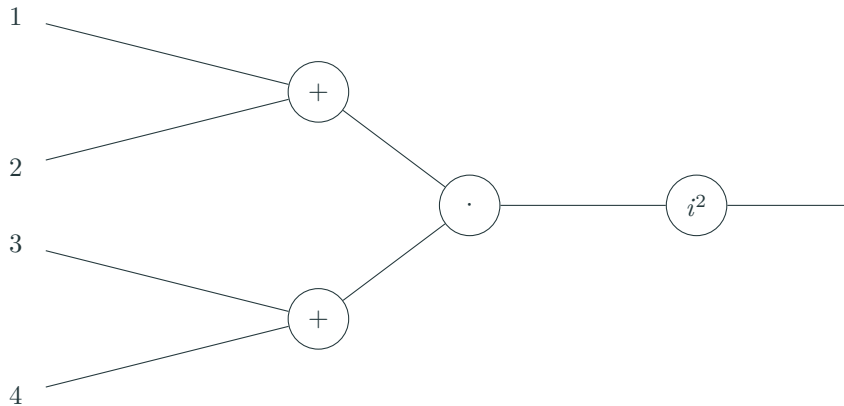
Backpropagation



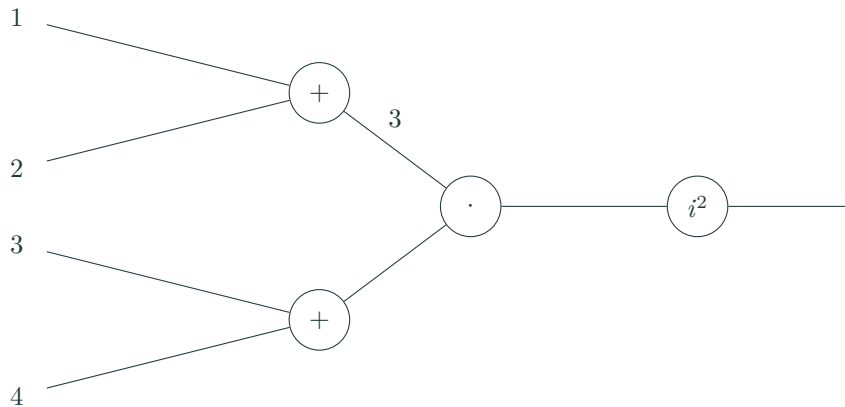
Frage: $\frac{\partial f}{\partial a}$?

$$\frac{\partial f}{\partial a} = \frac{\partial f}{\partial i} \cdot \frac{\partial i}{\partial gh} \cdot \frac{\partial gh}{\partial g} \cdot \frac{\partial g}{\partial a}$$

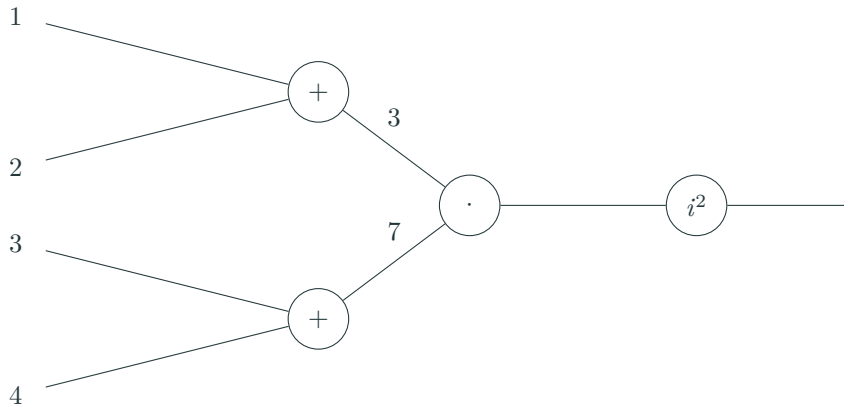
Beispiel



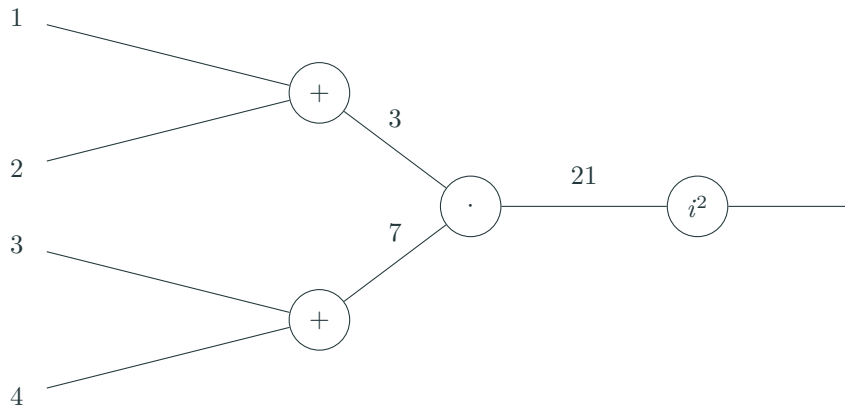
Beispiel



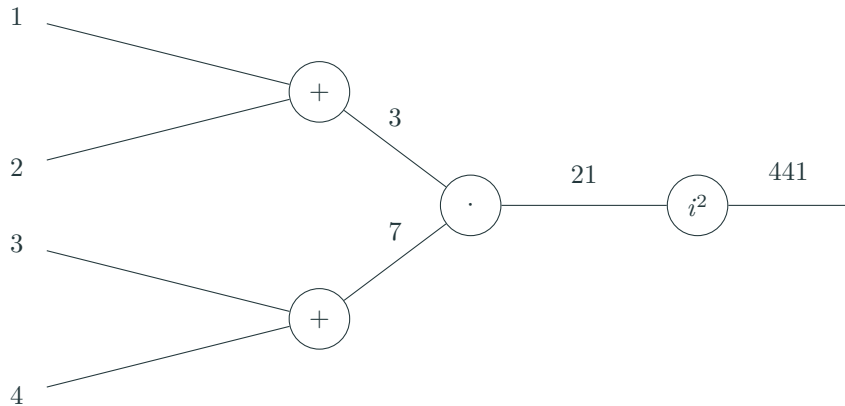
Beispiel



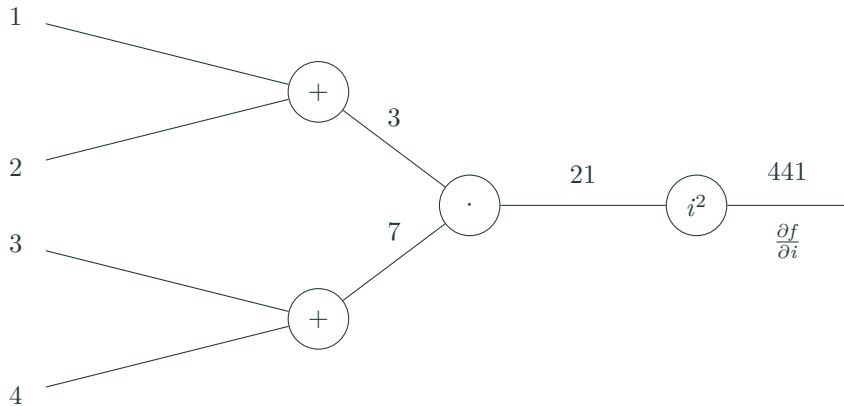
Beispiel



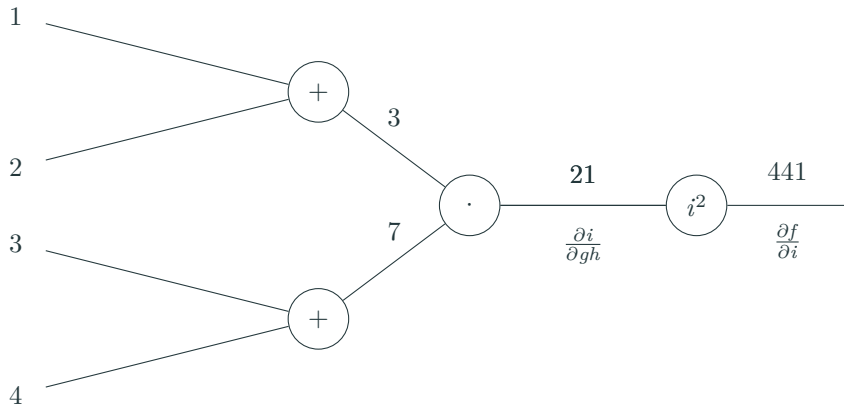
Beispiel



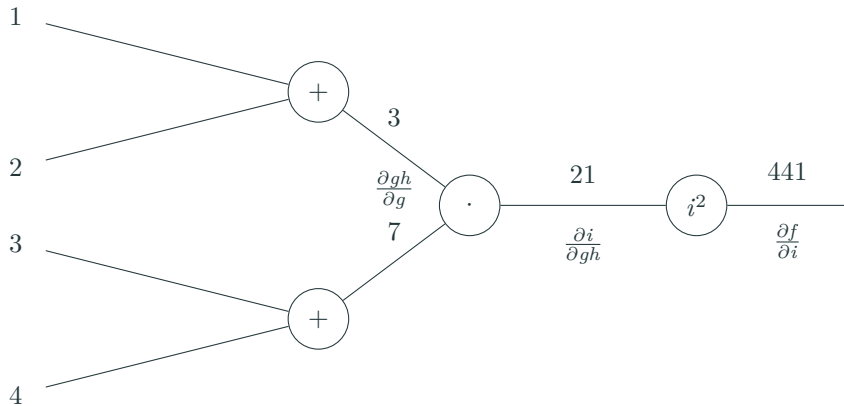
Beispiel



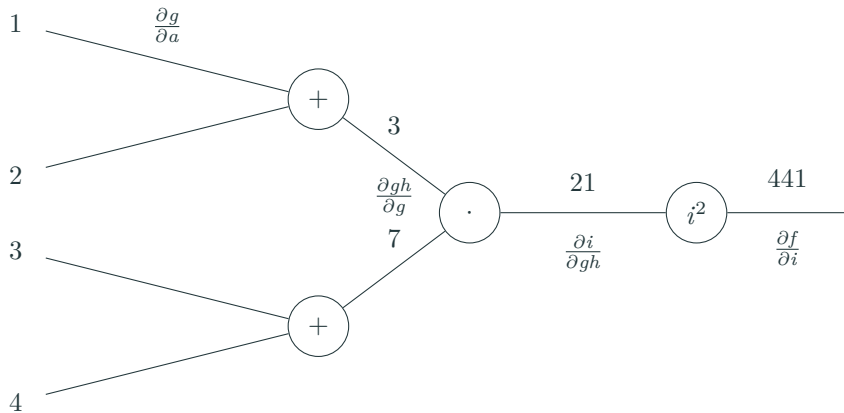
Beispiel



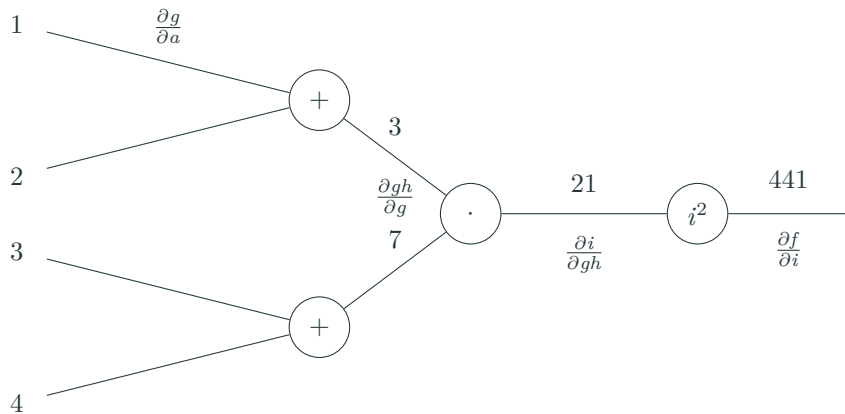
Beispiel



Beispiel



Beispiel







$$\frac{\partial f}{\partial i} = 2 \cdot 21 = 42; \quad \frac{\partial i}{\partial gh} = 1; \quad \frac{\partial gh}{\partial g} = h = 7; \quad \frac{\partial g}{\partial a} = 1 \quad \Rightarrow 7 \cdot 42 = 294$$

Pass

- Vorteil: Schnellere Konvergenz
- Verwendung von optimierter Cost-, Activation- und Gradient-Descent-Funktion

Fragen?

Literaturverzeichnis

-  GÉRON, Aurélien. *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. O'Reilly Media, 2019.
-  BURKOV, Andriy. *The hundred-page machine learning book*. Andriy Burkov Quebec City, Can., 2019.
-  LECUN, Yann; BOTTOU, Léon; BENGIO, Yoshua; HAFFNER, Patrick. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*. 1998, Jg. 86, Nr. 11, S. 2278–2324.
-  XIAO, Han; RASUL, Kashif; VOLLGRAF, Roland. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*. 2017.