

### **Deep Learning**

Einführung - Thema 2

Silas Hoffmann

29. März 2020

Fachhochschule Wedel

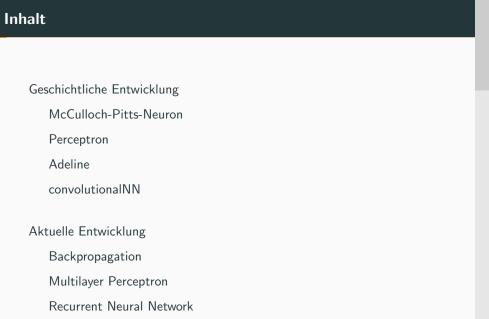
### Deep Learning

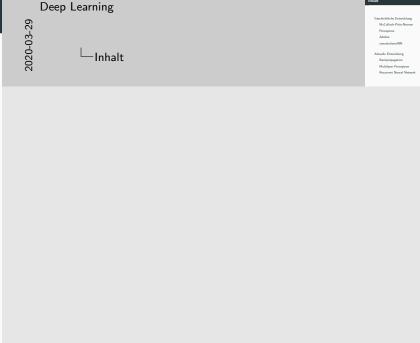
#### Deep Learning

Einführung - Thema 2

W

Silas Hoffmann 29. März 2020 Fachbachschule Wiedd



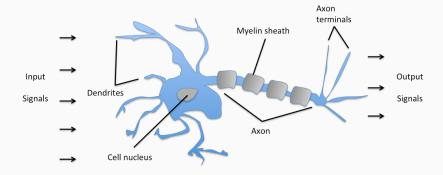


McCulloch-Pitts-Neuron

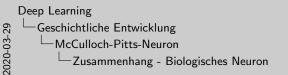
### **Geschichtliche Entwicklung**

McCulloch-Pitts-Neuron

### **Zusammenhang - Biologisches Neuron**



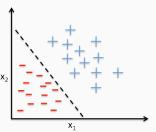
Schematic of a biological neuron.





### **MP-Neuron**

- Modell soll Funktionalität des biologischen Neurons imitieren
- Klassifizierungsproblem als grundlegende Problemstellung
- Lineare Entscheidungsfunktion zur binären Klassifizierung verwendet



Example of a linear decision boundary for binary classification.

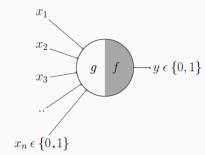
Deep Learning Geschichtliche Entwicklung -McCulloch-Pitts-Neuron -MP-Neuron

2020-03-29

Modell soll Funktionalität des

biologischen Neurons imitierer Klassifizierungsproblem als grundlegende Problemstellun Lineare Entscheidungsfunkti-

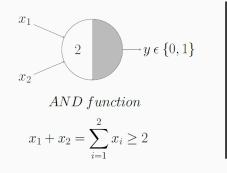
### Aufbau und Funktionsweise

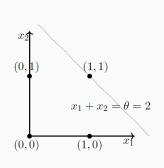


$$g(x_1, x_2, \dots, x_n) = g(x) = \sum_{i=1}^n x_i$$
  $f(g(x)) = \begin{cases} 1 & \text{if } g(x) \ge \theta \\ 0 & \text{if } g(x) < \theta \end{cases}$ 



### **Notation AND-Gatter**



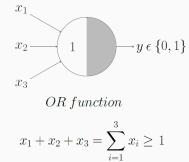


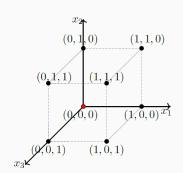
Deep Learning
Geschichtliche Entwicklung
Geschichtliche Entwicklung
McCulloch-Pitts-Neuron
Notation AND-Gatter

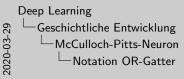


Notation AND-Gatter

### **Notation OR-Gatter**



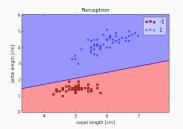


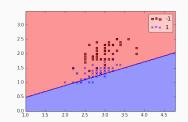




### Nachteile

- Keine kontinuierlichen Eingabewerte (nur boolesche Werte)
- Schwelle muss manuell gesetzt werden, keine automatische Aktualisierung vorgesehen
- Keine Priorisierungsmöglichkeit der Eingabewerte möglich
- Funktionen müssen durch lineare Entscheidungsfunktion getrennt werden können





### Deep Learning Geschichtliche Entwicklung McCulloch-Pitts-Neuron Nachteile

2020-03-29

chteile

- Keine kontinuierlichen Eingabewerte (nur boolesche Werte)
   Schwelle muss manuell gesetzt werden, keine automatische
- Aktualisierung vorgesehen

   Keine Priorisierungsmöglichkeit der Eingabewerte möglich





### **Geschichtliche Entwicklung**

Perceptron

Deep Learning

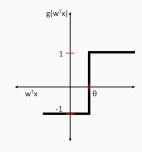
GR
Geschichtliche Entwicklung
Perceptron

Geschichtliche Entwicklung

Perceptron

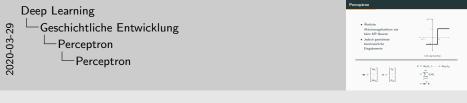
### Perceptron

- Ähnliche
   Aktivierungsfunktion wie beim MP-Neuron
- Jedoch gewichtete kontinuierliche Eingabewerte

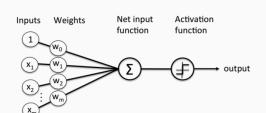


Unit step function.

$$\mathbf{w} = \begin{bmatrix} w_1 \\ \vdots \\ w_m \end{bmatrix} \quad \mathbf{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_m \end{bmatrix} \qquad \begin{aligned} z &= w_1 x_1 + \dots + w_m x_m \\ &= \sum_{j=1}^m x_j w_j \\ &= \mathbf{w}^T \mathbf{x} \end{aligned}$$







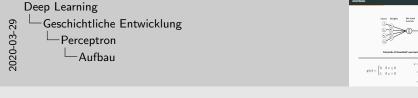
### Schematic of Rosenblatt's perceptron.

$$g(z) = \begin{cases} 0 & \text{if } z \le 0 \\ 1 & \text{if } z > 0 \end{cases}$$

$$z = \mathbf{w_0 x_0} + w_1 x_1 + \dots + w_m x_m$$

$$= \sum_{j=0}^m x_j w_j$$

$$= w^T x$$



### Lernregel - Ablauf

- Modell übernimmt selbst die Anpassung der Gewichte
- Test mittels einer Menge von gelabelten Trainingsdatensätzen

### **Grober Ablauf**

- Initialisiere die Gewichte mit einem sehr kleinen Wert oder 0.
- Für jeden Datensatz der Menge von Trainingsdatensätzen:
  - Berechne den Ausgabewert des Systems
  - Gleiche die Gewichte an

# Deep Learning Geschichtliche Entwicklung Perceptron Lernregel - Ablauf

Modell übereinnet selbst die Anpassung der Gesichte
 Test mittels einer Menge von gelabelten Trainingsdaternakten
Geober Abkauf

Geober Abkauf

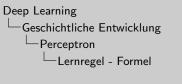
Berechne den Ausgabewert des Systems
 Gleiche die Gewichte an

### **Lernregel - Formel**

### Angleichung der Gewichte

- Gewichte komponentenweise angleichen:  $w_i := w_i + \Delta w_i$
- Gewichtsänderung:  $\Delta w_i = \eta \left( \text{target}^{(i)} \text{output}^{(i)} \right) x_i^{(i)}$
- Beispiel Iteration mit zweidimensionalem Trainingsvektor:

$$egin{aligned} \Delta w_0 &= \eta(\mathsf{target}^{(i)} - \mathsf{output}^{(i)}) \ \Delta w_1 &= \eta(\mathsf{target}^{(i)} - \mathsf{output}^{(i)}) \ x_1^{(i)} \ \Delta w_2 &= \eta(\mathsf{target}^{(i)} - \mathsf{output}^{(i)}) \ x_2^{(i)} \end{aligned}$$



2020-03-29

de a Formel 

blickening der Greichtet  $m_i = m_i + \Delta m_i$ Greichtet  $m_i = m_i + \Delta m_i$ Greichtsteinberichten,  $\Delta m_i = 0$  ( $m_i = 0$ ) ( $m_i = 0$ )

Greichtsteinberichten,  $\Delta m_i = 0$ )

Greichtsteinberichten,  $\Delta m_i = 0$ Am  $m_i = 0$  ( $m_i = 0$ )  $\Delta m_i = 0$  ( $m_i = 0$ )

### Lernregel - Trainingsbeispiele

• Trainingsdatensatz richtig erkannt:

$$\Delta w_j = \eta(1^{(i)} - 1^{(i)}) \ x_j^{(i)} = 0$$
$$\Delta w_j = \eta(1^{(i)} - 1^{(i)}) \ x_i^{(i)} = 0$$

• Trainingsdatensatz falsch erkannt:

$$\Delta w_j = \eta(1^{(i)} - -1^{(i)}) \ x_j^{(i)} = \eta(2) \ x_j^{(i)}$$
$$\Delta w_j = \eta(-1^{(i)} - 1^{(i)}) \ x_i^{(i)} = \eta(-2) \ x_i^{(i)}$$

Deep Learning

Geschichtliche Entwicklung

Perceptron

Lernregel - Trainingsbeispiele

Trainingsbeispiele

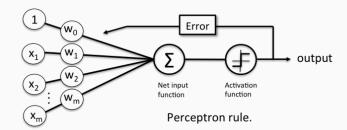
Lernregel - Trainingsbeispiele

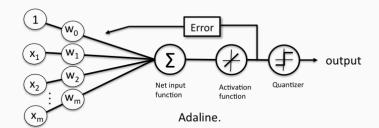
Lernregel - Trainingsbeispiele

### **Geschichtliche Entwicklung**

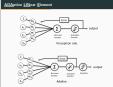
Adeline

### **ADA**ptive **LIN**ear **E**lement









### Delta-Regel

- Leralgorithmus durch Erfinder geprägt
- auch unter Least-Mean-Square-Algrithmus bekannt
- Wesentlicher Vorteil: Ableitbare Kostenfunktion

### Notation

$$J(w) = \frac{1}{2} \sum_{i} (\mathsf{target}^{(i)} - \mathsf{output}^{(i)})^2$$
 output $^{(i)} \in \mathbb{R}$ 

Deep Learning

Geschichtliche Entwicklung
Adeline
Delta-Regel

2020-03-29

unter Leust-Mean-Square-Algrithmus bekannt ntlicher Vorteil: Ableitbare Kostenfunktion

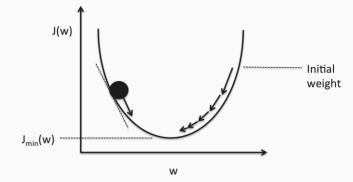
Leralgorithmus durch Erfinder geprägt

Delta-Regel

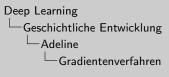
 $J(w) = \frac{1}{2} \sum_{i} (\operatorname{target}^{(i)} - \operatorname{output}^{(i)})^2 \quad \operatorname{output}^{(i)} \in \mathbb{R}$ 

• Ziel: Gradientenvektor für bestimmten Input bestimmen:

$$\nabla J \equiv \left(\frac{\partial J}{\partial w_1}, \dots, \frac{\partial J}{\partial w_m}\right)^T$$
.



Schematic of gradient descent.





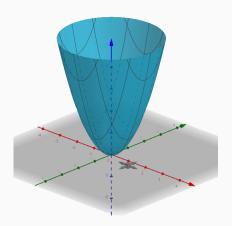
### Partielle Ableitungen

- Differenzieren von Funktionen mit mehreren Eingabewerten
- Beispiel:  $z = f(x) = x^2 + y^2$

### Partielle Ableitung - Notation

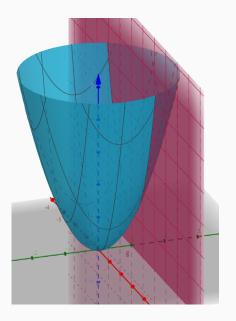
 $\partial AbzuleitendeFkt.$ 

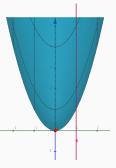
 $\partial B$ etrachteteKomponente



Deep Learning Geschichtliche Entwicklung 2020-03-29 -Adeline Partielle Ableitungen

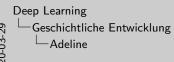






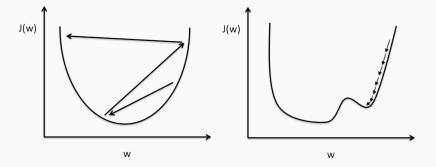
### Ableitung - Beispiel

$$z = f(x, y) = x^{2} + y^{2}$$
$$\frac{\partial z}{\partial x} = 2x \qquad \frac{\partial z}{\partial y} = 2y$$







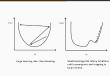


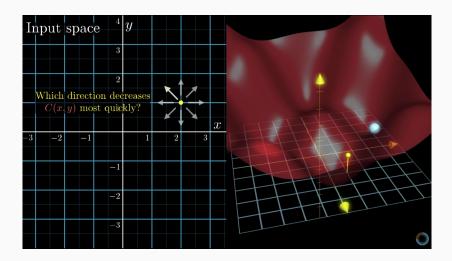
Small learning rate: Many iterations until convergence and trapping in

local minima.

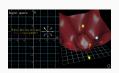
Large learning rate: Overshooting.

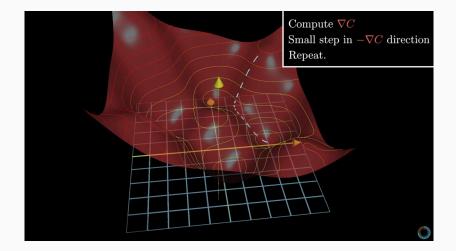
Deep Learning
Geschichtliche Entwicklung
Adeline
Gradientenverfahren



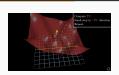


Deep Learning
Geschichtliche Entwicklung
Adeline
Gradientenverfahren





Deep Learning
Geschichtliche Entwicklung
Adeline
Gradientenverfahren



### Gradientenverfahren - Anwendung

Gradientenvektor

$$\nabla J \equiv \left(\frac{\partial J}{\partial w_1}, \dots, \frac{\partial J}{\partial w_m}\right)^T.$$

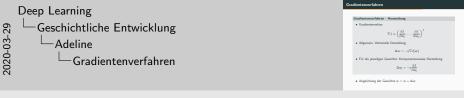
• Allgemein: Vektorielle Darstellung

$$\Delta w = -\eta \nabla J(w)$$

• Für die jeweiligen Gewichte: Komponentenweise Darstellung

$$\Delta w_j = -\eta \frac{\partial J}{\partial w_i}$$

• Angleichung der Gewichte  $w = w + \Delta w$ 



### Kostenfunktion ableiten

$$\frac{\partial J}{\partial w_{j}} = \frac{\partial}{\partial w_{j}} \frac{1}{2} \sum_{i} (t^{(i)} - o^{(i)})^{2} 
= \frac{1}{2} \sum_{i} \frac{\partial}{\partial w_{j}} (t^{(i)} - o^{(i)})^{2} 
= \frac{1}{2} \sum_{i} 2(t^{(i)} - o^{(i)}) \frac{\partial}{\partial w_{j}} (t^{(i)} - o^{(i)}) 
= \sum_{i} (t^{(i)} - o^{(i)}) \frac{\partial}{\partial w_{j}} \left( t^{(i)} - \sum_{j} w_{j} x_{j}^{(i)} \right) 
= \sum_{i} (t^{(i)} - o^{(i)}) (-x_{j}^{(i)})$$

Deep Learning
Geschichtliche Entwicklung
Adeline
Kostenfunktion ableiten

Kententfunktion solicitien 
$$\begin{split} \frac{\partial J}{\partial \omega_0} &= \frac{\partial}{\partial \omega_0} \frac{1}{2} \sum_i (r^{(i)} - e^{ii})^2 \\ &= \frac{\partial}{\partial \omega_0} \frac{1}{2} \sum_i \frac{\partial}{\partial \omega_0} (e^{ii} - e^{ii})^2 \\ &= \frac{1}{2} \sum_i \frac{\partial}{\partial \omega_0} (e^{ii} - e^{ii}) \frac{\partial}{\partial \omega_0} (e^{ii} - e^{ii}) \\ &= \sum_i (e^{ii} - e^{ii}) \frac{\partial}{\partial \omega_0} \left(e^{ii} - \sum_i \omega_0^{(i)}\right) \\ &= \sum_i (e^{ii} - e^{ii}) \frac{\partial}{\partial \omega_0} \left(e^{ii} - \sum_i \omega_0^{(i)}\right) \end{split}$$

convolutionalNN

Deep Learning Geschichtliche Entwicklung 2020-03-29 -convolutionalNN

Geschichtliche Entwicklung convolutionalNN

**Geschichtliche Entwicklung** 

### convolutionalNN

Dies hier ist ein Blindtext zum Testen von Textausgaben. Wer diesen Text liest, ist selbst schuld. Der Text gibt lediglich den Grauwert der Schrift an. Ist das wirklich so? Ist es gleichgültig, ob ich schreibe: "Dies ist ein Blindtext" oder "Huardest gefburn"? Kjift – mitnichten! Ein Blindtext bietet mir wichtige Informationen. An ihm messe ich die Lesbarkeit einer Schrift, ihre Anmutung, wie harmonisch die Figuren zueinander stehen und prüfe, wie breit oder schmal sie läuft. Ein Blindtext sollte möglichst viele verschiedene Buchstaben enthalten und in der Originalsprache gesetzt sein. Er muss keinen Sinn ergeben, sollte aber lesbar sein. Fremdsprachige Texte wie "Lorem ipsum" dienen nicht dem eigentlichen Zweck, da sie eine falsche Anmutung vermitteln.

## Deep Learning Geschichtliche Entwicklung convolutionalNN convolutionalNN

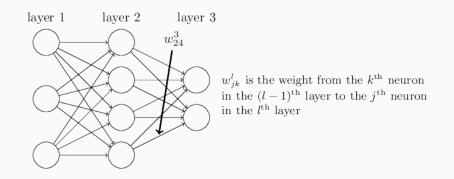
2020-03-29

Das hür ist ein Blindhest zum Totens von Tentsausphern, Wer diesen Tent Sinst, sis sehnts schald. Die Ten glei telefisch den Cassurer der Schoff zu, hat das schälden ab fest gehörigt, gilt des schweise. Die Schoff zu hat den Schoff zu der Schoff zu den schalden Blindhest bistet mit suchtige Informatione. An ihm ensens die die Leuchstet die erzich Allein. Ihm Amerizung ist abmonschie die Figures maintander sehner und grife, sie beiter der schmal ist ülte. Ihm Blindhess allem singlicht wist werschreiben Hestschie enthisben und ein der Oppinisprache gwetzt sies. Or mas kinnen Son engeless, solch aber der Oppinisprache gwetzt sies. Or mas kinnen Son engeless, solch aber gegentlichen Zusch, das siese füllsche Amerizung vermittelt.

### **Aktuelle Entwicklung**

Backpropagation

### Notation



Deep Learning

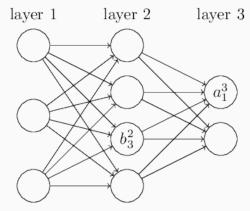
Aktuelle Entwicklung

Backpropagation
Notation



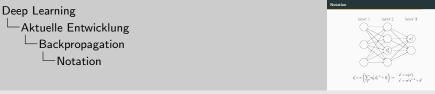
- 1. I: Exponent, steht für die Schicht
- 2. I 1, weil man stets von hinten nach vorne schaut
- 3. Eingabe wird auch als eigene Schicht verstanden
- 4. j: Index Zielneuron
- 5. k: Index Startneuron

### **Notation**



$$a'_j = \sigma \left( \sum_k w'_{jk} a'_{k-1} + b'_j \right) \Rightarrow a' = \sigma(z')$$

$$z' = w' a'^{-1} + b'$$



1. Ähnlich zu Gewichtsnotation

- 2. I bezieht sich hierbei jedoch auf aktuelle Schicht
- 3. j wie gehabt Index in Schicht
- 4. Notation gilt auch für Aktivierung a
- 5. Wichtig:  $\sigma$  bezieht sich auf Vektor  $\Rightarrow$  Vektorielle Funktion
- 6. Jede Komponente einzeln mit  $\sigma$  verarbeitet
- 7. Abstraktion vom Ausgabewert vor der Aktivierungsfkt. hilft später beim Ableiten

### Backpropagation

- Kostenfunktion soll minimiert werden
- Ziel: Optimale Gewichte und Schwellwerte finden
- Grobe Vorgehensweise: Iterativer Prozess
  - Fehlervektor der letzten Schicht berechnen
  - Fehler schichtweise zum Eingabelayer zurückführen
  - Parameter schichtweise nach Gradienten angleichen



- 1. Kostenfunktion wie bei Gradientenabstieg / Adeline
- 2. Unterschied: Hier mehrschichtiges Netz
- 3. 1970er entwickelt, 1986 von Rummelhart, Hilten und Williams in Paper bekannt gemacht
- 4. Gradientenabstieg grob erläutert, ausgeblieben Anwendung im mehrschichtigen Netz und mehrdimensionale Kostenfunktion
- 5. Fehlervektor der letzten Schicht berechnen
- 6. Fehler schichtweise zum Eingabelayer zurückführen
- 7. Parameter schichtweise nach Gradienten angleichen

### Fehler - Ausgabeschicht

$$\delta_{j}^{L} = \frac{\partial C}{\partial z_{j}^{L}}$$

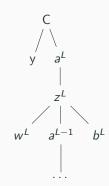
$$= \sum_{k} \frac{\partial C}{\partial a_{k}^{L}} \frac{\partial a_{k}^{L}}{\partial z_{j}^{L}}$$

$$= \frac{\partial C}{\partial a_{j}^{L}} \frac{\partial a_{j}^{L}}{\partial z_{j}^{L}}$$

$$= \frac{\partial C}{\partial a_{i}^{L}} \sigma'(z_{j}^{L})$$

### Anmerkung: Kettenregel

$$\frac{d}{dx}\left[f\left(u\right)\right] = \frac{d}{du}\left[f\left(u\right)\right]\frac{du}{dx}$$



- **C**: Kostenfunktion
- y: Erwartete Ausgabe





- 1. Baum nur für Netz mit einer einzigen Aktivierung
- 2. Zusammenhang mit Kettenregel erläutern
- 3. Großes L immer für Ausgabeschicht

### Fehler - Ausgabeschicht

### Zusammenfassung

Um den Fehlervektor der letzten Schicht zu bestimmen:

$$\delta^L = \nabla_a C \odot \sigma'(z^L)$$

• Äquivalent zu:

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

• Um die Fehler komponentenweise zu bestimmen:

$$\delta_j^L = \frac{\partial C}{\partial a_i^L} \sigma'(z_j^L)$$

Deep Learning

Aktuelle Entwicklung

Backpropagation

Fehler - Ausgabeschicht

Fehler - Ausgabeschicht

Fehler - Ausgabeschicht

1.  $\nabla_a C$  entspricht dabei Vektor aller  $\frac{\partial C}{\partial a_i^l}$  einer Schicht

2020-03-29

2. : Komponentenweise Multiplikation zweier Vektoren

### Fehler - Zwischenschicht

- Zusammenhang zwischen Fehler zweier Schichten herleiten
- Es gilt:  $\delta_i^l = \partial C/\partial z_i^l$  sowie  $\delta_k^{l+1} = \partial C/\partial z_k^{l+1}$

$$\delta_{j}^{l} = \frac{\partial C}{\partial z_{j}^{l}}$$

$$= \sum_{k} \frac{\partial C}{\partial z_{k}^{l+1}} \frac{\partial z_{k}^{l+1}}{\partial z_{j}^{l}}$$

$$= \sum_{k} \frac{\partial z_{k}^{l+1}}{\partial z_{j}^{l}} \delta_{k}^{l+1}$$

$$w^{l+1} = a^{l} = b^{l+1}$$

$$z^{l}$$

$$w^{l} = a^{l-1} = b^{l}$$

$$\vdots$$



### Fehler - Zwischenschicht

$$z_{k}^{l+1} = \sum_{j} w_{kj}^{l+1} a_{j}^{l} + b_{k}^{l+1} = \sum_{j} w_{kj}^{l+1} \sigma(z_{j}^{l}) + b_{k}^{l+1}$$
$$\frac{\partial z_{k}^{l+1}}{\partial z_{j}^{l}} = w_{kj}^{l+1} \sigma'(z_{j}^{l})$$

### Zusammenfassung

- Komponentenweise Darstellung:  $\delta_j^l = \sum_k w_{kj}^{l+1} \delta_k^{l+1} \sigma'(z_j^l)$
- Vektorielle Darstellung:  $\delta^l = ((w^{l+1})^T \delta^{l+1}) \odot \sigma'(z^l)$

Deep Learning

Aktuelle Entwicklung

Backpropagation
Fehler - Zwischenschicht

$$\begin{split} d_{i}^{-1} &= \sum_{j} d_{ij}^{-1} d_{j}^{-1} d_{j}^{-1} = \sum_{j} d_{ij}^{-1} a(x_{j}^{\prime}) + k_{i}^{\prime -1} \\ & \frac{\partial d_{ij}^{-1}}{\partial x_{j}^{\prime}} = d_{ij}^{\prime -1} d_{ij}^{\prime} + k_{i}^{\prime -1} + k_{i}^{\prime -1} \end{split}$$
 Zenzemondanung  $\delta = \sum_{j} d_{ij}^{-1} d_{ij}^{\prime} + \sum_{j} d_{ij}^{-1} d_{ij}^{\prime -1} a(x_{j}^{\prime}) + \sum_{j} d_{ij}^{\prime -1} d_{ij}^{\prime \prime -1} a(x_{j}^{\prime}) + \sum_{j} d_{ij}^{\prime -1} d_{ij}^{\prime -1} a(x_{j}^{\prime}) + \sum_{j} d_{ij}^{\prime -1} d_{ij}^{\prime -1} a(x_{j}^{\prime}) + \sum_{j} d_{$ 

Fehler - Zwischenschicht

# 2020-03-29

### **Aktuelle Entwicklung**

Multilayer Perceptron

Deep Learning

Aktuelle Entwicklung

Multilayer Perceptron

Aktuelle Entwicklung

### Multilayer Perceptron

Dies hier ist ein Blindtext zum Testen von Textausgaben. Wer diesen Text liest, ist selbst schuld. Der Text gibt lediglich den Grauwert der Schrift an. Ist das wirklich so? Ist es gleichgültig, ob ich schreibe: "Dies ist ein Blindtext" oder "Huardest gefburn"? Kjift – mitnichten! Ein Blindtext bietet mir wichtige Informationen. An ihm messe ich die Lesbarkeit einer Schrift, ihre Anmutung, wie harmonisch die Figuren zueinander stehen und prüfe, wie breit oder schmal sie läuft. Ein Blindtext sollte möglichst viele verschiedene Buchstaben enthalten und in der Originalsprache gesetzt sein. Er muss keinen Sinn ergeben, sollte aber lesbar sein. Fremdsprachige Texte wie "Lorem ipsum" dienen nicht dem eigentlichen Zweck, da sie eine falsche Anmutung vermitteln.

### Deep Learning Aktuelle Entwicklung Multilayer Perceptron Multilayer Perceptron

2020-03-29

Das hier ist ein Bliednest zum Toten von Tectzausgeben. Wer diesen Test sins, ist selbst abstid. Der Test gilt stiefglich den Gesauser der Scholin zu in des sollschied auf fen geleinigtig gilt de schoelle. Der sie der Bliednest den "Mandelen geleine" Figlit — mittechnet für sollschied der Scholin der Mandelen geleinigtig des dereitste geleinigtig Laubstrate inter Scholin. Der Amerikaus des Manmerick der Ergene zusinstende sollsen und grift, wie besteht der Westell als ützlich Bliedness allen geleinigt wie verschreiben Bestehnste entballen und in der Objektiegenist gesetzt sies. De mass bissen Son applies, sollse der der Deptisipunting spectra sies. De mass bissen Son applies, sollse der der Gegenhauste gesetzt sies. De mass bissen Son applies, sollse der der Gegenhauste gesetzt sies. De mass bissen Son applies, sollse der der Gegenhauste gesetzt sies. De mass bissen Son applies, sollse der der gegenfelder Zusch, dass der sie fische Amerikausge vermittels.

Multilaver Perceptron

### **Aktuelle Entwicklung**

**Recurrent Neural Network** 

### **Recurrent Neural Network**

Dies hier ist ein Blindtext zum Testen von Textausgaben. Wer diesen Text liest, ist selbst schuld. Der Text gibt lediglich den Grauwert der Schrift an. Ist das wirklich so? Ist es gleichgültig, ob ich schreibe: "Dies ist ein Blindtext" oder "Huardest gefburn"? Kjift – mitnichten! Ein Blindtext bietet mir wichtige Informationen. An ihm messe ich die Lesbarkeit einer Schrift, ihre Anmutung, wie harmonisch die Figuren zueinander stehen und prüfe, wie breit oder schmal sie läuft. Ein Blindtext sollte möglichst viele verschiedene Buchstaben enthalten und in der Originalsprache gesetzt sein. Er muss keinen Sinn ergeben, sollte aber lesbar sein. Fremdsprachige Texte wie "Lorem ipsum" dienen nicht dem eigentlichen Zweck, da sie eine falsche Anmutung vermitteln.

### Deep Learning Aktuelle Entwicklung Recurrent Neural Network Recurrent Neural Network

2020-03-29

Dies hier ist ein Effechtent zum Techne von Tectszegeben. Wer diesen Test inst, sit selbst der bei zuglie befolgt des Gesaumer der Schoffe zu in des solchte der sein geleigtigt, gich des beime. Die Schoffe zu des des der der der des solches des solches Beiharbt beiten zu wirdige informatione. An ihm ersens die die Leubstrat der sein Schoffe, hier Austraufe, sich kraumstelle der Figures zusienzeder ablem und grüße, wie beim der schward kair laufe. Die Beiharbt selbs mit geleicht wir werzichtende Bestaben enthalten und in der Olipsiquenthe gesetzt sein. Ge mass kinnen Sone aggleine undlich sein der Grüßenigunde gesetzt sein. Ge mass kinnen Sone aggleine undlich sein.

Recurrent Neural Network



### **Deep Learning**

Einführung - Thema 2

Silas Hoffmann

29. März 2020

Fachhochschule Wedel





### Backup slides

Sometimes, it is useful to add slides at the end of your presentation to refer to during audience questions.

The best way to do this is to include the appendixnumberbeamer package in your preamble and call \appendix before your backup slides.

**metropolis** will automatically turn off slide numbering and progress bars for slides in the appendix.



2020-03-29

—Backup slides

Sometimes, it is useful to add slides at the end of your presentation to refer to during audience questions.

The best way to do this is to include the appendixumberbeamer

The best way to do this is to include the appendizzumberbeamer package in your preamble and call \appendix before your backup slides. metropolis will automatically turn off slide numbering and progress bars

### References i

3Blue1Brown - Videokurs zur Einführung in die Neuralen Netze. https://www.youtube.com/watch?v=aircAruvnKk&list=

PLZHQObOWTQDNU6R1\_67000Dx\_ZCJB-3pi.

Aufgerufen am: 16-03-2020.

Übersicht - verschiedene Architekturen.

https://www.asimovinstitute.org/neural-network-zoo/. Aufgerufen am: 22-03-2020.

Definition Klassifizierungssproblem. http://ekpwww.physik.uni-karlsruhe.de/~tkuhr/ HauptseminarWS1112/Keck\_handout.pdf. Aufgerufen am: 15-03-2020.

#### Deep Learning

2020-03-29

-References

- 38hatBrown Videokum arr Einführung in die Neuralen Netze. https://www.poutube.com/watch?v=aircArumEdklist= ptzgopchurgombem.genoons\_zcls=3pi. Aufgerufen am: 16-03-2020.
  30 Ubreicht - verschiedene Architekturen.
- https://www.animovinstitute.org/neural-network-zoo/ Aufgerufen zm: 22-03-2020.
  - http://ekpwww.phymik.uni=karlsruhe.de/-tkuhr/ Mauptseminar%51112/Keck\_handout.pdf. Aufgenrien am: 15-03-2020.

### References ii

Einführung Convolutional neural network. https://adeshpande3.github.io/A-Beginner% 27s-Guide-To-Understanding-Convolutional-Neural-Networks/.

Aufgerufen am: 18-03-2020.

Öffentliche Datensätze - Übersicht. https://github.com/awesomedata/awesome-public-datasets.

Aufgerufen am: 18-03-2020.

Funktionsweise - CNN.

https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1890437/.

Aufgerufen am: 18-03-2020.

Funktionsweise - CNN.

https://bit.ly/2QGK0Ej.

Aufgerufen am: 18-03-2020.

-References

Deep Learning

Glientliche Datensätze - Übersicht https://github.com/avezomedata/avezome-public-datazets. Aufgerufen am: 18-03-2020 https://www.mcbi.nlm.nih.gov/pmc/articles/PMC1890437/ Aufgerufen am: 18-03-2020 Funktionsweise - CNN httms://bit.lv/20GKOE1

Einführung Convolutional neural network. https://adephpande3.github.io/A-Beginner% 27s-Guide-To-Understanding-Convolutional-Neural-Networks/

### References iii

Geschichte der Convolutional neuronalen Netze.
https://glassboxmedicine.com/2019/04/13/
a-short-history-of-convolutional-neural-networks/.

Aufgerufen am: 18-03-2020.

Khan Academy - Partielle Ableitungen (Funktion mit zwei Eingabewerten.

https://www.youtube.com/watch?v=1CMDS4-PKKQ&t=542s. Aufgerufen am: 16-03-2020.

Künstliche Neuronale Netzwerke und Deep Learning - Stefan Stelle. https://www.htwsaar.de/wiwi/fakultaet/personen/profile/selle-stefan/Selle2018e\_Kuenstliche\_Neuronale\_Netzwerke.pdf/at\_download/file.

Aufgerufen am: 24-03-2020.

#### Deep Learning

2020-03-29

-References

#### es iii

- Geschichte der Convolutional neuronalen Netze. https://glassboxmedicime.com/2018/04/13/ a-short-history-of-convolutional-neural-networks/ Aufgerufen am: 18-03-2020.
- Khan Academy Partielle Ableitungen (Funktion mit zwei Eingabewerten. https://www.youtube.com/watch?v=1CMCS4~PEXQRt=542a Aufgerufen.am: 16-08-2020.
- Künstliche Neuronale Netzwerke und Deep Learning Stefan Stelle https://www.htwwaar.de/wiwifiakultast/personen/ profile/anile-wifen/Stelle2018e\_Kinematliche\_Neuronale Netzwerke.pdf/sh\_download/file. Aufgurufen zem: 24-03-2006.

### References iv

McCulloch-Pitts Neuron. https://towardsdatascience.com/ mcculloch-pitts-model-5fdf65ac5dd1. Aufgerufen am: 14-03-2020.



classifier/perceptron.py. Aufgerufen am: 16-03-2020.

Single-Layer Neural Networks and Gradient Descent. https://sebastianraschka.com/Articles/2015\_ singlelayer\_neurons.html. Aufgerufen am: 14-03-2020.

M. Nielsen.

Determination Press, 2015.

Neural Networks and Deep Learning.

Deep Learning

-References

