

Deep Learning

Einführung - Thema 2

Silas Hoffmann

29. März 2020

Fachhochschule Wedel

Deep Learning

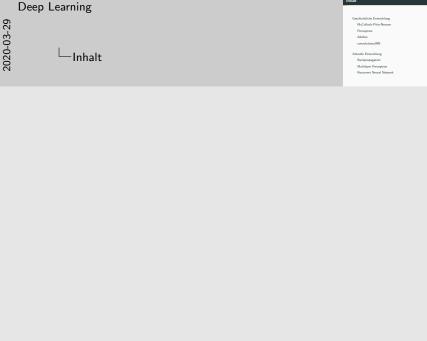
Deep Learning

Einführung - Thema 2

W

Silas Hoffmann 29. März 2020 Fachbachschule Wiedd



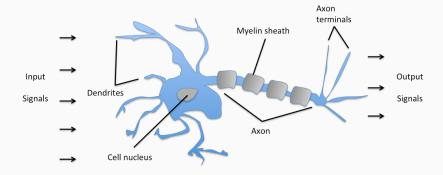


McCulloch-Pitts-Neuron

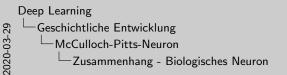
Geschichtliche Entwicklung

McCulloch-Pitts-Neuron

Zusammenhang - Biologisches Neuron



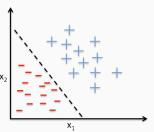
Schematic of a biological neuron.





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.

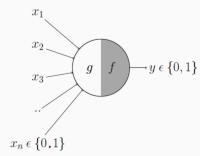
Deep Learning Geschichtliche Entwicklung -McCulloch-Pitts-Neuron -MP-Neuron

2020-03-29

Modell soll Funktionalität des

biologischen Neurons imitierer Klassifizierungsproblem als grundlegende Problemstellun Lineare Entscheidungsfunkti-

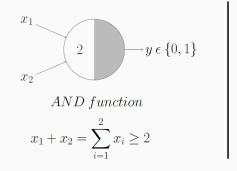
Aufbau und Funktionsweise

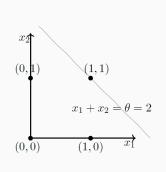


$$g(x_1, x_2, \dots, x_n) = g(x) = \sum_{i=1}^n x_i$$
 $f(g(x)) = \begin{cases} 1 & \text{if } g(x) \ge \theta \\ 0 & \text{if } g(x) < \theta \end{cases}$



Notation AND-Gatter



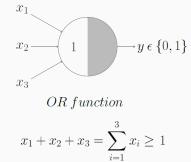


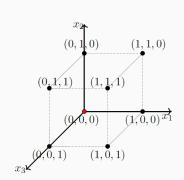
Deep Learning
Geschichtliche Entwicklung
Geschichtliche Entwicklung
McCulloch-Pitts-Neuron
Notation AND-Gatter



Notation AND-Gatter

Notation OR-Gatter





Deep Learning

Geschichtliche Entwicklung

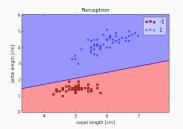
McCulloch-Pitts-Neuron

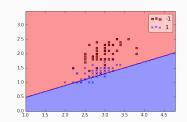
Notation OR-Gatter



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



- 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







Geschichtliche Entwicklung

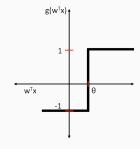
Geschichtliche Entwicklung

Perceptron

Perceptron Geschichtlic

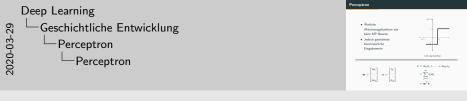
Perceptron

- Jedoch gewichtete kontinuierliche Eingabewerte

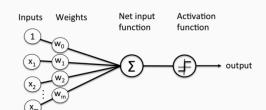


Unit step function.

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







Schematic of Rosenblatt's perceptron.

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

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

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

$$= w^T x$$



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 übersinnst sellsat die Angassung der Gesichte
Test mittels einer Mange von gelabelten Trainingsdatenaktzen
Geber Ablasif

Infeldiniere die Gewichte mit einem sehr kleinen Wert oder O.
Für jedem Datensatz der Menge von Trainingsdatenaktzen:
Berchote den Angabowert des Systems

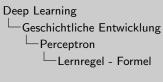
Gleiche die Gewichte an

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:

$$egin{aligned} \Delta \mathit{w}_0 &= \eta \big(\mathsf{target}^{(i)} - \mathsf{output}^{(i)} \big) \ \Delta \mathit{w}_1 &= \eta \big(\mathsf{target}^{(i)} - \mathsf{output}^{(i)} \big) \ \mathit{x}_1^{(i)} \ \Delta \mathit{w}_2 &= \eta \big(\mathsf{target}^{(i)} - \mathsf{output}^{(i)} \big) \ \mathit{x}_2^{(i)} \end{aligned}$$



2020-03-29

de Formel $\begin{aligned} & \text{bishouse} \text{ der Gracibite} \\ & \text{supplication} \\ & \text{supplication} \\ & \text{supplication} \\ & \text{der gracibite} \\ & \text{de$

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:

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

Deep Learning

Geschichtliche Entwicklung

Perceptron

Lernregel - Trainingsbeispiele

or Trainingsbeispiele

or Trainingsbeispiele

trainingsbeispiele

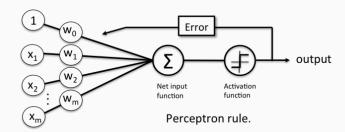
constant and a section of the se

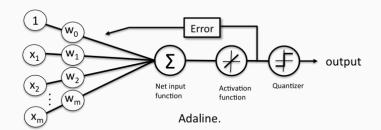
Geschichtliche Entwicklung

Adeline

e entwicki

ADAptive LINear Element









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

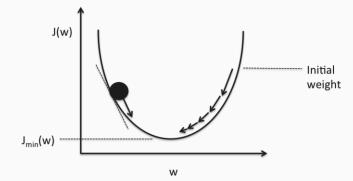
• Leralgorithmus durch Erfenter sparigit • such sinter Leart-Mon-Square-Algorithmus belannt • Wesserdicher Vortsel: Ableithare Konterfunktion

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

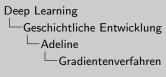
Delta-Regel

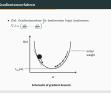
• Ziel: Gradientenvektor für bestimmten Input bestimmen:

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



Schematic of gradient descent.





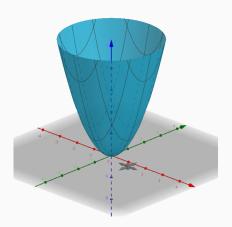
Partielle Ableitungen

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

Partielle Ableitung - Notation

 $\partial AbzuleitendeFkt.$

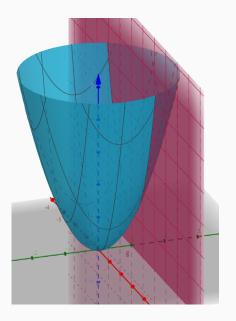
 $\overline{\partial BetrachteteKomponente}$

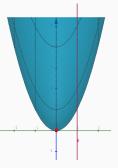


Deep Learning
Geschichtliche Entwicklung
Adeline
Partielle Ableitungen



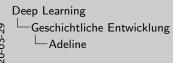
Partielle Ableitungen



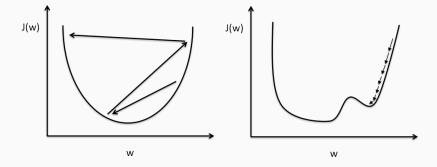


Ableitung - Beispiel

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





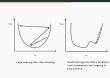


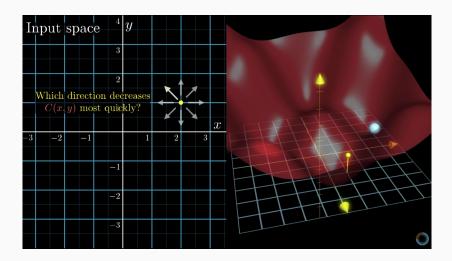
Small learning rate: Many iterations until convergence and trapping in

local minima.

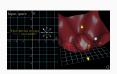
Large learning rate: Overshooting.

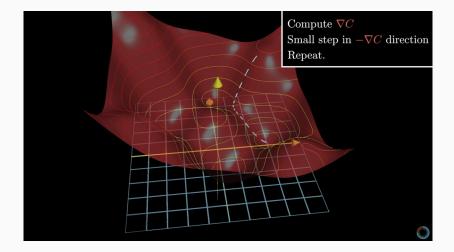
Deep Learning
Geschichtliche Entwicklung
Adeline
Gradientenverfahren



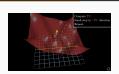


Deep Learning
Geschichtliche Entwicklung
Adeline
Gradientenverfahren





Deep Learning
Geschichtliche Entwicklung
Adeline
Gradientenverfahren



Gradientenverfahren - Anwendung

Gradientenvektor

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

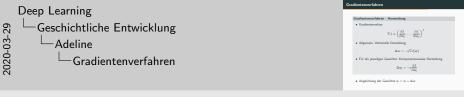
• Allgemein: Vektorielle Darstellung

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

• 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

Kontentfunktion abbetten
$$\begin{split} \frac{\partial J}{\partial \omega_0} &= \frac{\partial}{\partial \omega_0} \frac{1}{2} \sum_{i} (e^{i(1} - \omega^{(i)})^2 \\ &= \frac{1}{2} \sum_{i} \sum_{i} \frac{\partial \omega_i}{\partial \omega_i} (e^{i(1} - \omega^{(i)})^2 \\ &= \frac{1}{2} \sum_{i} \sum_{i} \frac{\partial \omega_i}{\partial \omega_i} (e^{i(1} - \omega^{(i)})^2 \\ &= \sum_{i} (e^{i(1)} - \omega^{(i)}) \frac{\partial}{\partial \omega_i} (e^{i(2)} - \omega^{(i)}) \\ &= \sum_{i} (e^{i(1)} - \omega^{(i)}) \frac{\partial}{\partial \omega_i} (e^{i(2)} - \sum_{i} \omega_i e^{i(1)}) \\ &= \sum_{i} (e^{i(1)} - \omega^{(i)}) \sum_{i} (e^{i(1)} - \sum_{i} \omega_i e^{i(1)}) \end{split}$$

Deep Learning

Geschichtliche Entwicklung

-convolutionalNN

Geschichtliche Entwicklung

convolutionalNN

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

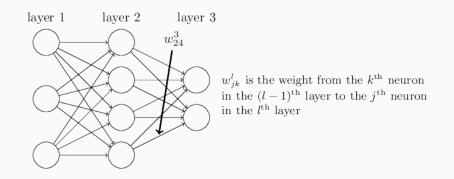
2020-03-29

Dass har ist die Blindhest zum Testen von Testasagsben. Wer diesen Test San, ist währte krahlt. Der Test glei sleigheit den Gesausert der Schrift aus, ihr das wicklich wir her geleigigig, gich schreibeite. Der Schrift aus der den Schrift der Schrift auf der Schrift aus der Blindheit bietet fern sichlige fellermatione. An ihm ressen sich die Leuksbeit der sich Allein. An Amerikang sich kennensisch die Figures zusänsache sichen voll gerich, den seine den schreibe sich sich sich Einfelten salle mitglich wir verschreiben Brachtsbeit enthalte und in die Gründsprachte gesetzt wis. Gross keinen Son eighen, suchs aber der Objektsprache gesetzt wis. Gross keinen Son eighen, suchs aber der Schrift aus der Schrift aus der Schrift aus der gegentlichen Zond, d. au sie fillsich Amerikang verwirtels.

Aktuelle Entwicklung

Backpropagation

Notation



Deep Learning

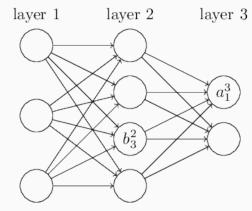
Aktuelle Entwicklung

Backpropagation
Notation



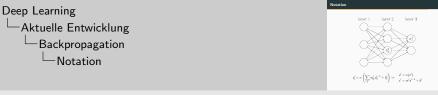
- 1. I: Exponent, steht für die Schicht
- 2. I 1, weil man stets von hinten nach vorne schaut
- 3. Eingabe wird auch als eigene Schicht verstanden
- 4. j: Index Zielneuron
- 5. k: Index Startneuron

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: σ bezieht sich auf Vektor \Rightarrow Vektorielle Funktion
- 6. Jede Komponente einzeln mit σ 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

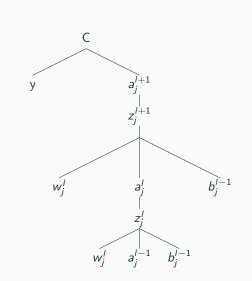


2020-03-29

Kitatonfarktion sall minimiert surden
Ziel Optimale Gazzielle und Schwalberte finden
Ziel Optimale Gazzielle und Schwalberte finden
Geste Vorgebenweise Instrüter Dezense
Fallwahrte freiten Schatz Bescheide
Fallwahrte freiten Schatz Bescheider zurschlaßen
Fallwahrte freiten Schatz Bescheider zurschlaßen
Fallwahrte Schatzen und Gradenten soglichen

- 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

Aufbau







Fundamentale Gleichungen - 1

• Fehlervektor der letzten Schicht berechnen: $\delta^L = \nabla_a C \odot \sigma'(z^L)$

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

Anmerkung: Kettenregel

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





1. Großes L immer für Ausgabeschicht

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

Multilaver Perceptron

—Aktuelle Entwicklung Recurrent Neural Network

Aktuelle Entwicklung

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

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 - verschiedens 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://glassboomedicine.com/2019/04/13/ a-short-history-of-convolutional-neural-networks/ Aufgersien 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.

Künstiche Neuronale Netzverke und Deep Learning - Stefan Stella https://www.htwwaar.de/stwiftskultast/personem/ profile/anile-weten/Stella0018e_Eusenstliche_Neuronale Netzwerke.pdf/st_download/file.

References iv

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

Perceptron - Python Implementierung. https://github.com/rasbt/mlxtend/blob/master/mlxtend/

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

