Grundlagen der Wissensverarbeitung

Künstliche neuronale Netze

Dipl.-Inform. I. Boersch
Prof. Dr.-Ing. Jochen Heinsohn
FB Informatik und Medien
Mai 2014

Beispiel 1

Januar 2014: Google kauft DeepMind Technologies für

480 Millionen Euro

DeepMind Technologies: britisch, klein (75 Mitarbeiter), gegründet 2011 vom Neurowissenschaftler Demis Hassabis, Technologien: **künstliche neuronale Netze, reinforcement learning, convolutional networks**

Beispiel 2: email vom 31.03.2014

Sehr geehrter Herr Dr. Heinsohn,

ich bin verantwortlich für das Kapitalmarktgeschäft und die Marketingaktivitäten in der xxxxxx Bank. In diesem Funktion habe ich in den letzten Quartalen den **Einsatz neuronaler Netzwerke forciert** und möchte nun einen größeren Personenkreis durch ein Inhouse-Seminar schulen lassen. Bereits in der Vergangenheit haben wir hierfür sehr gerne und erfolgreich mit Fachhochschulen zusammengearbeitet (z.B. bzgl. der mathematischen Bewertung von exotischen Optionen). Wir möchten nun diese Praxis weiter fortsetzen und würden Sie gerne hierfür gewinnen.

Geplant wäre eine **eintägige Veranstaltung bei uns im Hause für zirka 15-20 Personen**

.....

Wir **verwenden derzeit im Marketing einfache neuronale Netze für Zielgruppenselektionen (Mehrschicht-Perzeptron, 2 Hidden Layer),** können uns aber auch vorstellen, das Einsatzspektrum um Kapitalmarkaktivitäten und um den Einsatz in der Vertriebssteuerung zu erweitern. Zudem möchte ich perspektivisch Aufgaben des **Databased Marketing auf genetische Algorithmen, bzw. evolutionäre Algorithmen** ausrichten.

Vor diesem Hintergrund habe ich eine Online-Recherche vorgenommen und **bin auf Ihren Lehrstuhl aufmerksam geworden**. Haben Sie Interesse daran, unsere Mitarbeiter zu diesem Thema durch ein Inhouse-Seminar bei uns im Hause zu schulen?

Für Rückfragen stehe ich Ihnen gerne zur Verfügung und freue mich auf eine Rückantwort.

Mit freundlichen Grüßen

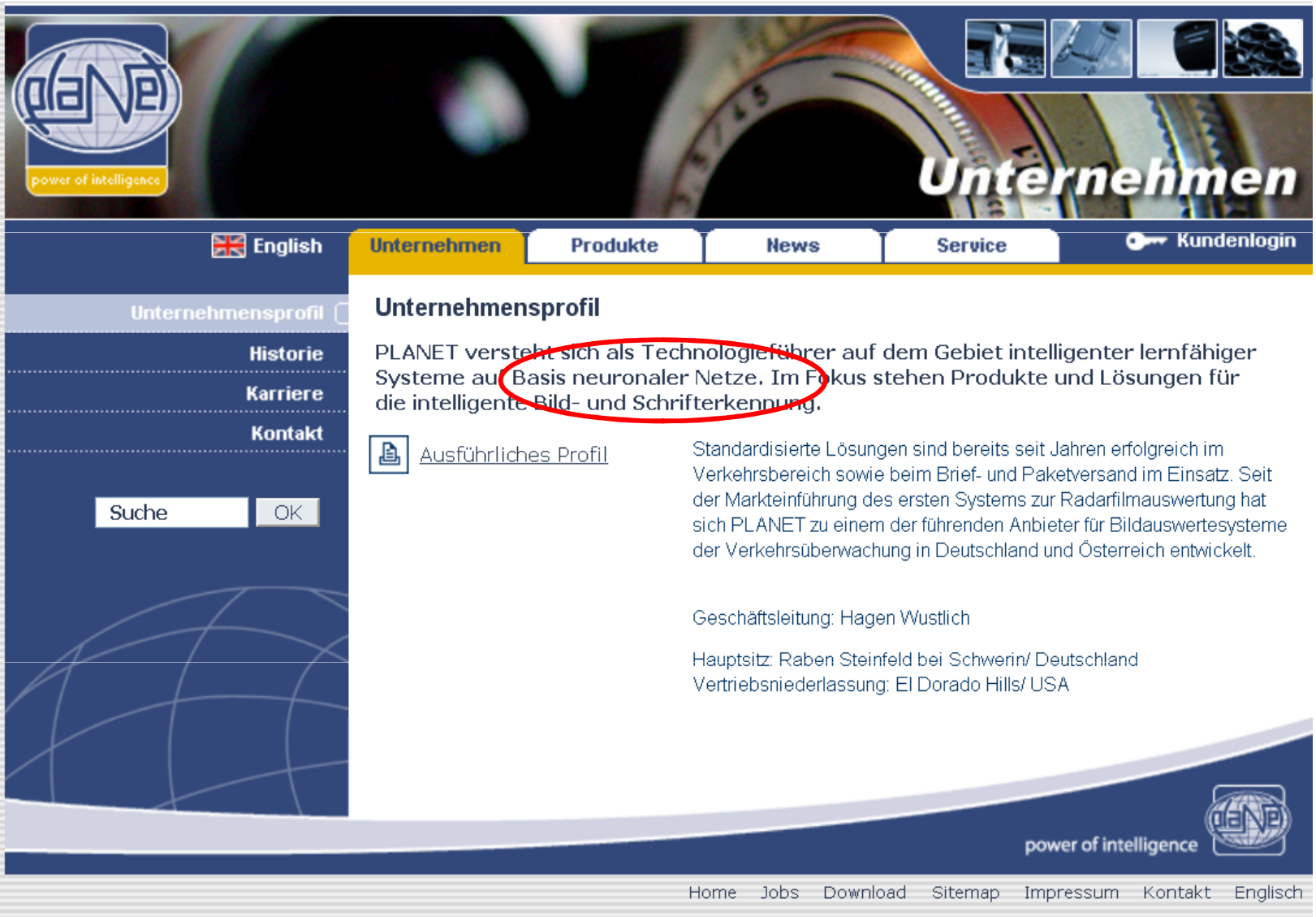
Motivation – Anwendungsbeispiele künstlicher neuronaler Netze (KNN)

„Eines der überzeugendsten Merkmale für die Rechtfertigung einer (neuen) Theorie ist der Grad ihrer Verwendbarkeit in Wirtschaft, Produktion, Dienstleistung, etc...." (Kinnebrock,1994).

Breites Anwendungsspektrum KNN, z.B.:

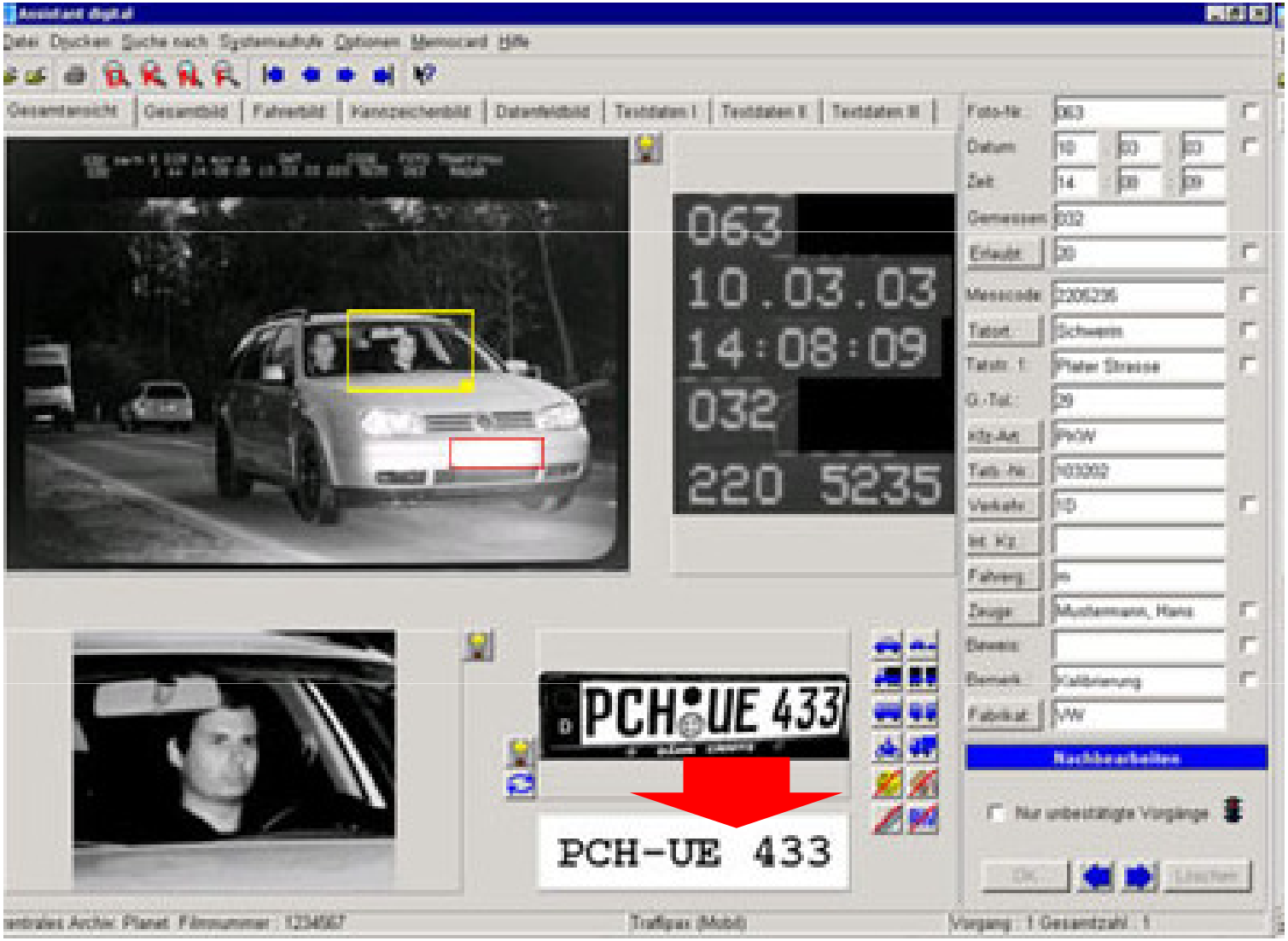
- Qualitätskontrolle von Motoren durch Abhören des Motorengeräusches
- Vorlesen eines geschriebenen Textes
- Erkennen handgeschriebener Buchstaben (Daimler Benz AG Ulm)
- Management von Rentenfonts (Allg. Deutsche Direktbank)
- Vorhersage von Wirtschaftsdaten/Aktienkursen (Siemens AG)
- Bonitätsanalyse von Kreditnehmern (Neckermann Versand AG)
- Flexible Tourenplanung mit NN (Karstadt AG)

Beispiel 3: Neuronale Netze in der Bilderkennung



Quelle: PLANET intelligent systems GmbH, Mai 2009
http://www.planet.de/unternehmen/index.html

Beispiel 3: Intelligente Bild- und Schrifterkennung



Quelle: PLANET intelligent systems GmbH, Mai 2009
http://www.planet.de/produkte/verkehr/module_asdigital.html

Beispiel 4: Neuronale Netze in der Prognose

» Kontakt » Suche » Sitemap » Impressum » English

otto group

Presse

- Home
- Die Gruppe
- Segmente
- Die Unternehmen
- Karriere
- Geschäftsbericht
- Presse
  - » Pressemitteilungen
  - » Pressefotos/Logos
  - » TV-Schnittmaterial
  - » Unternehmens-informationen

Pressemitteilung

Hamburg, 19. Mai 2008

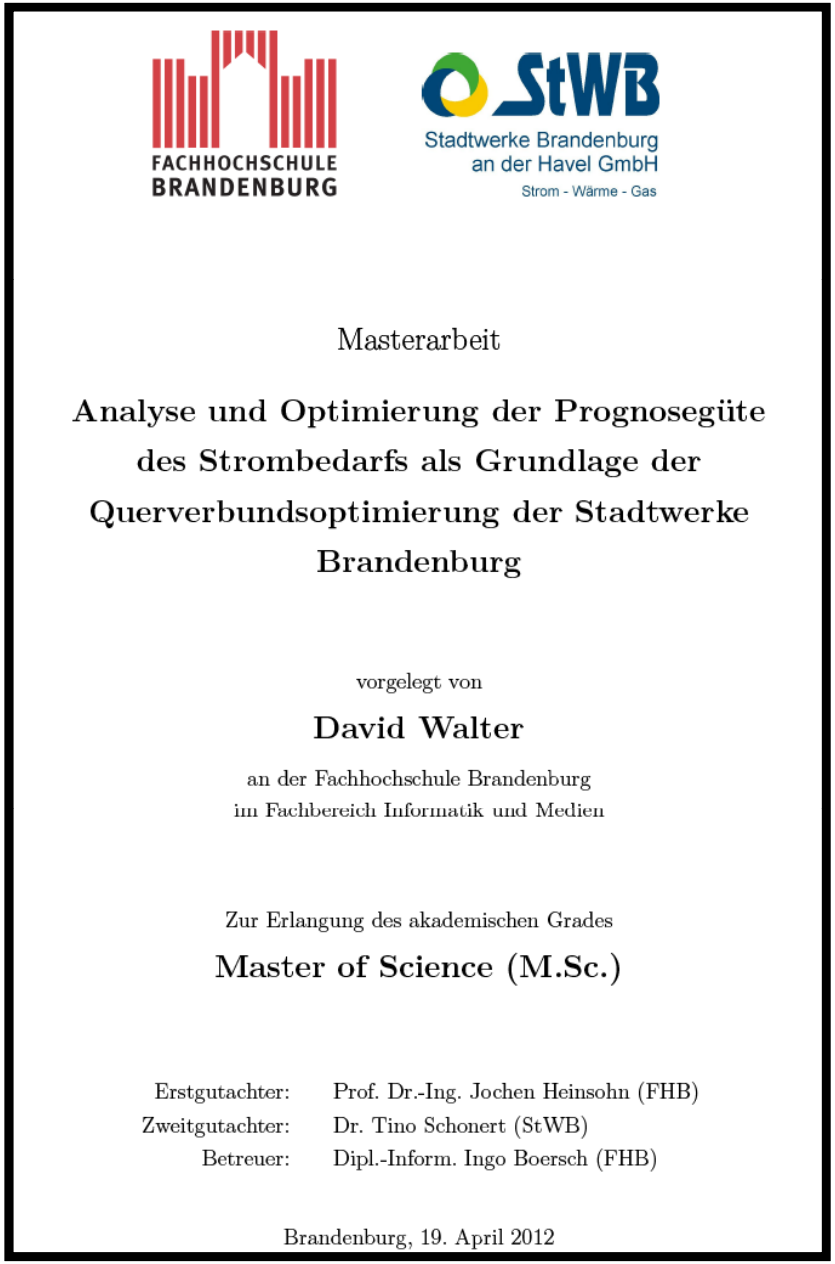
Otto Group gründet Joint Venture mit Phi-T und revolutioniert Artikel-Absatz-Prognosen

Die Otto Group, Hamburg, größte Versandhandelsgruppe der Welt und die Physics Information Technologies GmbH, Phi-T, Karlsruhe, haben das Gemeinschaftsunternehmen Phi-T products & services gegründet. Dies kündigten Hans-Otto Schrader, Vorstandsvorsitzender der Otto Group und Professor Michael Feindt, Mitbegründer von Phi-T, heute auf einer Pressekonferenz in Hamburg an. Ziel der Zusammenarbeit von Otto und Phi-T ist der branchenexklusive Einsatz der auf künstlichen neuronalen Netzen beruhenden NeuroBayes-Technologie von Prof. Feindt zur Optimierung von Artikel-Absatz-Prognosen innerhalb der Otto Group. Durch die Zusammenarbeit mit Phi-T erhält Otto Zugang zu den modernsten wissenschaftlichen Erkenntnissen und Methoden im Bereich des Data Mining und sichert sich somit einen entscheidenden Wettbewerbsvorteil.

Quelle: Otto (GmbH & Co KG), Mai 2009
http://www.ottogroup.com/701.html

Beispiel 5: Bedarfsprognose

Vorhersage des Strombedarfs der Stadt Brandenburg in den nächsten 24 Stunden mit Hilfe neuronaler Netze



FACHHOCHSCHULE BRANDENBURG

StWB Stadtwerke Brandenburg an der Havel GmbH
Strom - Wärme - Gas

Masterarbeit

Analyse und Optimierung der Prognosegüte des Strombedarfs als Grundlage der Querverbundsoptimierung der Stadtwerke Brandenburg

vorgelegt von

David Walter

an der Fachhochschule Brandenburg
im Fachbereich Informatik und Medien

Zur Erlangung des akademischen Grades

Master of Science (M.Sc.)

Erstgutachter: Prof. Dr.-Ing. Jochen Heinsohn (FHB)
Zweitgutachter: Dr. Tino Schonert (StWB)
Betreuer: Dipl.-Inform. Ingo Boersch (FHB)

Brandenburg, 19. April 2012

Was macht menschliche Intelligenz aus ?

U.a.: die Fähigkeit, Informationen und Informationseinheiten zueinander in Relation setzen zu können.

Beispiele:

- Visuelles Überprüfen von Schweißnähten: Ein erfahrener Meister hat gelernt, fehlerfreie und fehlerhafte Schweißnähte zu identifizieren
- Handschrifterkennnung: Wir alle können handgeschriebene Buchstaben erkennen, obwohl keiner dem anderen gleicht

Welche drei Grundtypen von Abbildungen gibt es?

- berechenbar: Es gibt einen Algorithmus; konventionell programmierbar (Sinusfkt.)
- nicht berechenbar und intuitiv nicht erfassbar (chaotisch, Würfelwurf -> Augenzahl)
- nicht berechenbar, aber intuitiv erfassbar (Kranker Patient -> Heilende Therapie)

Gibt es Systeme, die Abbildungen des Typs 3 erlernen?

Das biologische Vorbild „Gehirn“

- 2 Prozent des Körpergewichts, 25 Prozent der Körperenergie
- Neuron als elementare Verarbeitungseinheit, insgesamt 10^{10} Neuronen
- Hochgradiges Parallelverarbeitungssystem, ca. 10.000 Verbindungen/Neuron
- Insgesamt 10^{14} Verbindungen (sog. Synapsen)

- [Bei der Geburt sind bereits alle Neuronen vorhanden]
- Lernen und Vergessen bedeuten Veränderung von Verbindungen

Aber: Viele Funktionen des menschlichen Gehirns sind noch nicht verstanden ...

- Blue Brain Projekt (Schweiz, USA):
  - Nachbau eines Säugetiergehirns (zunächst Ratte) auf neuronaler Ebene mit realistischen Neuronenmodellen
  - Stand Mai 2009: 10.000 Neuronen, 400 Neuronenarten, 30 Millionen Synapsen
- 3 Großprojekte zur Kartierung und Nachbau: Human Brain Project (Europa), Brain Activity Map Project (USA), Human Connectome Project (USA)

Das biologische Vorbild „Gehirn“

Computer	Gehirn
von-Neumann-Architektur	Parallelverarbeitungs-system
Wenige Prozessoren	Milliarden von Prozessoren
Stärke: Bearbeitung von mathematischen Rechenaufgaben	Stärke: Wiedererkennen einer Person (auch leicht verändert oder auf einem Foto)

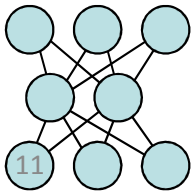
Sie kennen
EVA

Reizverarbeitung im Gehirn
(vereinfacht)

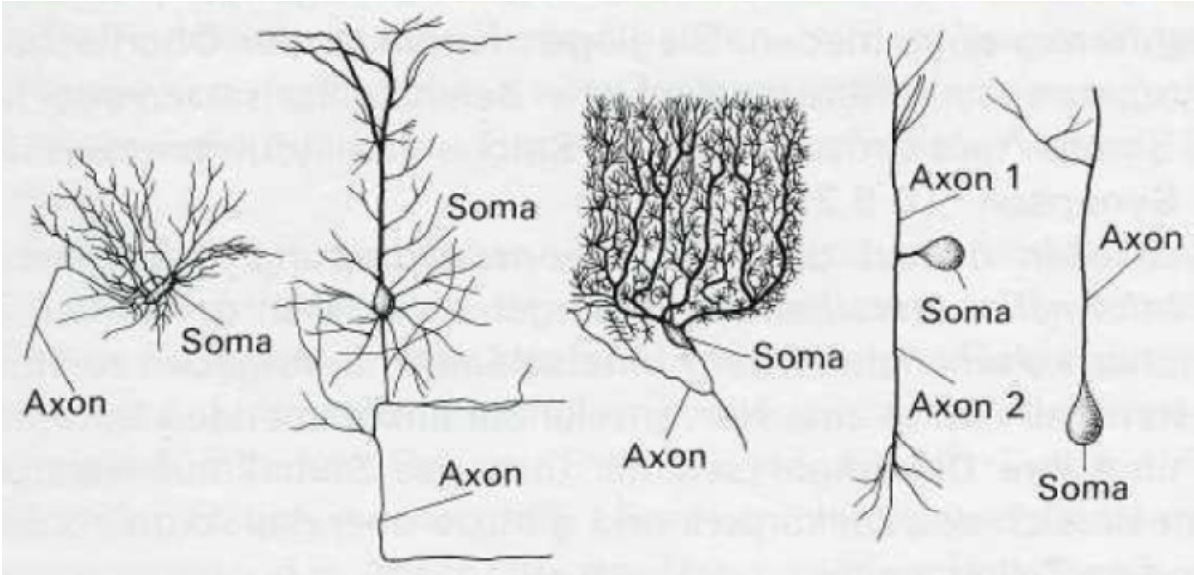
Rezeptoren: nehmen Reize auf, erzeugen elektro-chemische Signale = ähnlich Sensoren in der Technik (Licht, Druck, Temp., Ton, ...)

Verarbeitung der Reize in **Nervenknoten** (bspw. Gehirn)

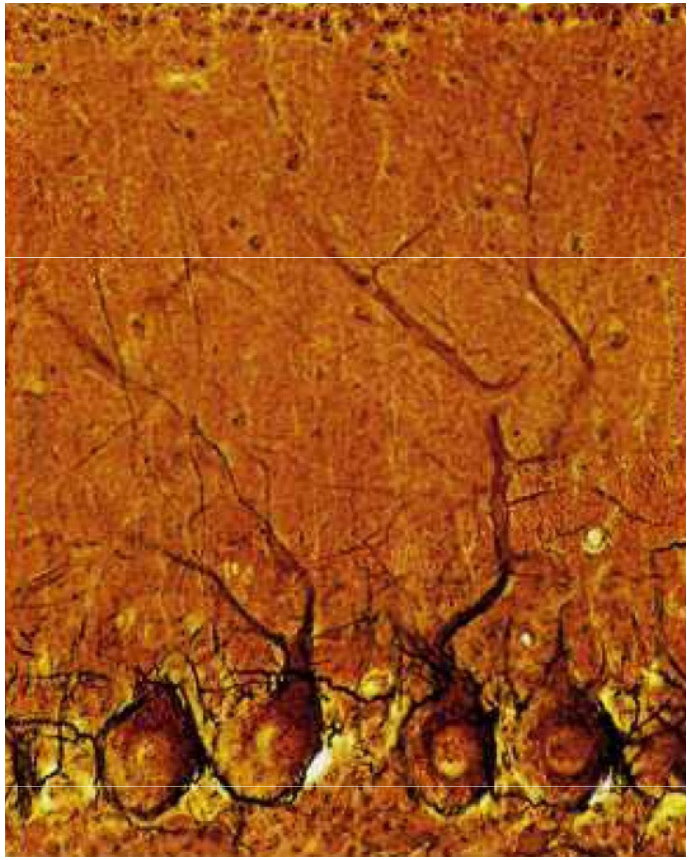
Effektoren (angesteuerte Gewebeteile, z.B. Muskeln, Drüsen) = Aktoren in der Technik (Motor, Licht, Pumpe, ...)



Reale Nervenzellen



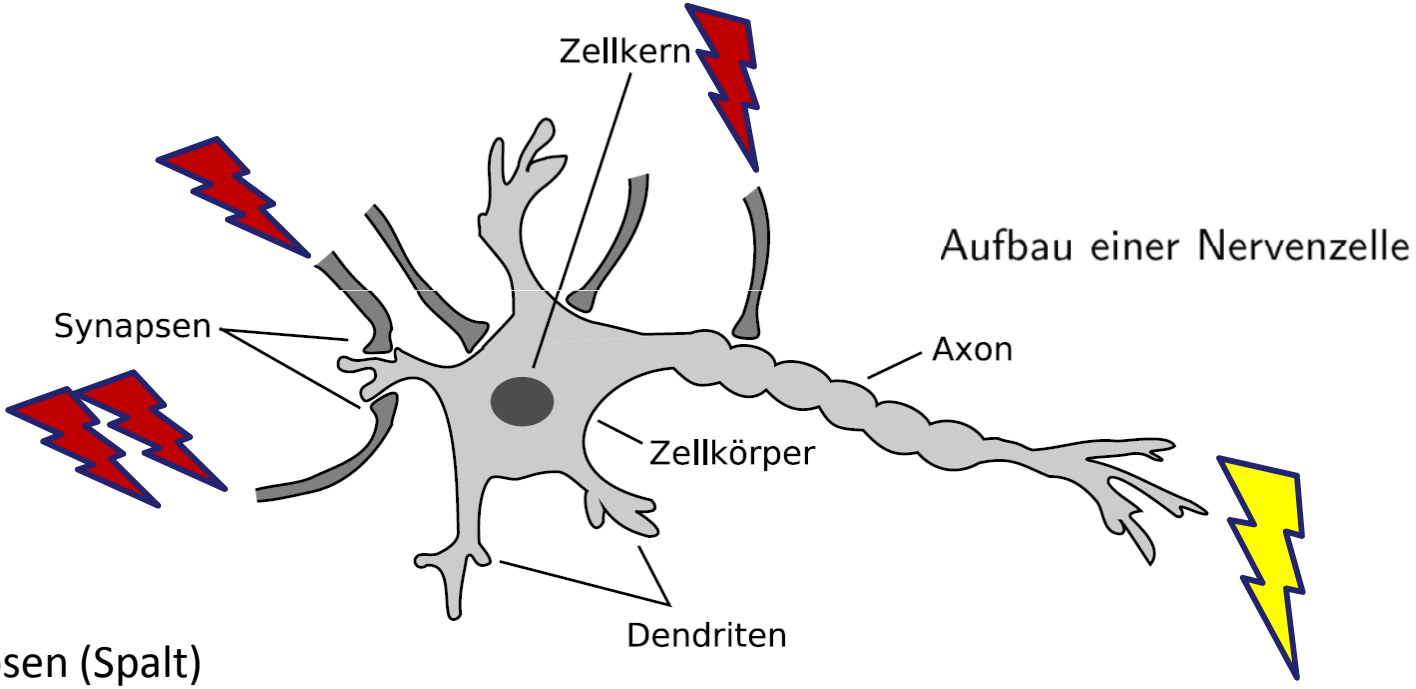
Verschiedene Typen von Nervenzellen.
Von links nach rechts: Korbzelle aus dem Kleinhirn, große Pyramidenzelle aus der Großhirnrinde, Purkinje-Zelle aus dem Kleinhirn, bipolare Schaltzelle und monopolare Nervenzelle
(Soma = Zellkörper)



Nervenzellen aus dem Kleinhirn einer Katze

Quelle: Anatomische und physiologische Grundlagen des Verhaltens. Lehr- und Arbeitsbuch Sek2, B. Löwe, W. D. Thiel, J. Prantl, 1994

Aufbau eines Neurons



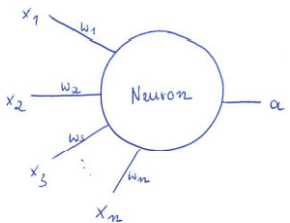
Synapsen (Spalt)

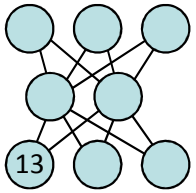
- können ankommende Potentialwerte (elektro-chem. Reize) verstärken oder hemmen
- können diese Wirkung im Laufe der Zeit verändern

Arbeitsweise:

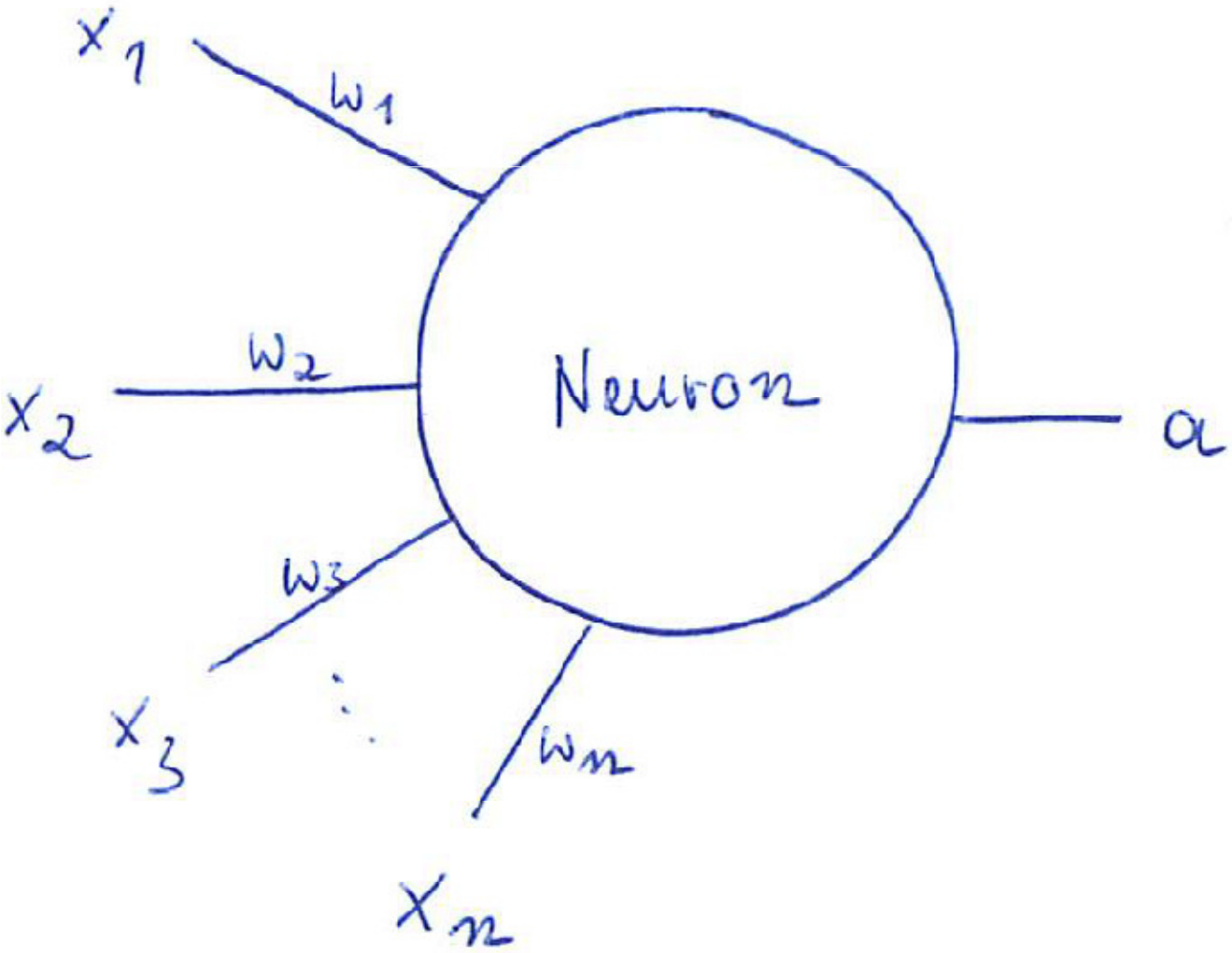
- Wenn die Summe der Eingabewerte als elektrisches Potential einen Schwellwert überschreitet, wird das Neuron aktiv - es „feuert„

- Siehe bspw. EEG im Biosignal-Labor



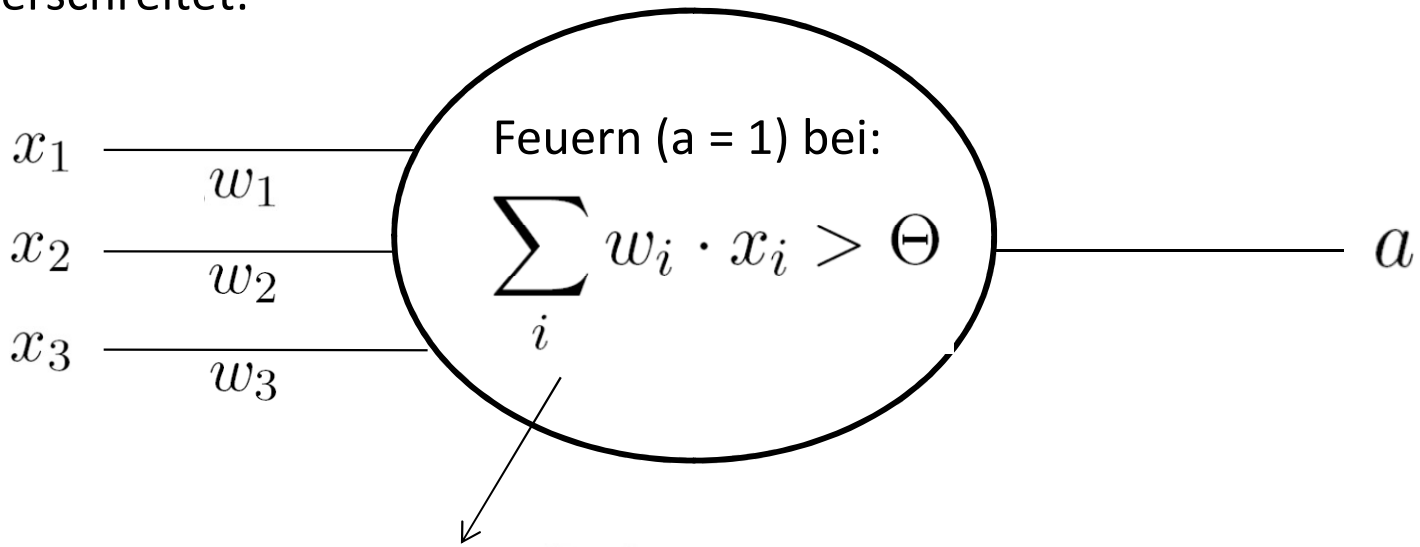


Vereinfachte Vorstellung



Das mathematische Modell eines Neurons

- Neuron-Zustände a: 0 und 1 (Das Neuron „feuert" oder „feuert nicht")
- Eingabewerte x sind 0 und 1
- Synapsen verstärken oder hemmen = Multiplikation mit positiven oder negativen Zahlen w (sog. Wichtungen)
- Das Neuron „feuert“, wenn die Netzaktivität net einen Schwellwert (Theta) überschreitet:



- Netzaktivität net: $net = \sum_i w_i \cdot x_i$

Schwellwert als Wichtung eines ON-Neurons

Vereinfachung, um den Schwellwert nicht immer extra betrachten zu müssen:

Statt zu testen, ob die Summe den Schwellwert Θ überschreitet, zählt man -Θ zu den Gewichten und vergleicht die dann entstehende Summe mit 0:

Die Bedingung zum Feuern (dass also a = 1 wird):

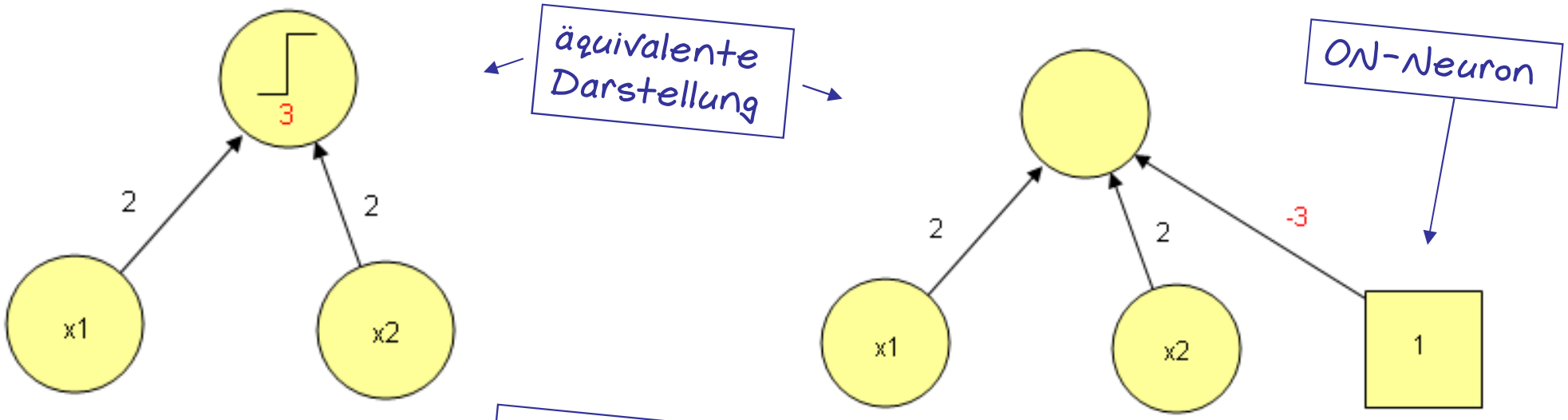
$$\sum_i w_i \cdot x_i > \Theta$$

ist äquivalent zu:

$$\sum_i w_i x_i - \Theta > 0$$

Schwellwert als Wichtung eines ON-Neurons

- BIAS: Zusatzeingabe mit Wert 1 und dem Gewicht –Θ (Minus! Theta)
- Beispiel: Neuron mit zwei Eingängen x1 und x2



äquivalente Darstellung

```
2 * x1 + 2 * x2 > 3
```

```
2 * x1 + 2 * x2 + 1 * -3  > 0
```

Definition 1: Neuron (Perzeptron)

Es seien $x_1, x_2, x_3, \ldots x_n$ Eingangswerte von der Größe 0 oder 1. Zudem seien die Synapsenwerte $w_1, w_2, w_3, \ldots w_n$ beliebige reelle Zahlen (Gewichte) und

$$net = \sum_i w_i \cdot x_i$$: Netzaktivität.

Dann ist

$$a = f(net)$$

der Ausgabewert des Neurons, wobei die Funktion f(net) definiert ist durch

$$f(net) = \begin{cases} 1 & falls\ net > 0 \\ 0 & falls\ net \leq 0 \end{cases}$$

Transfer- bzw. Aktivierungsfunktion

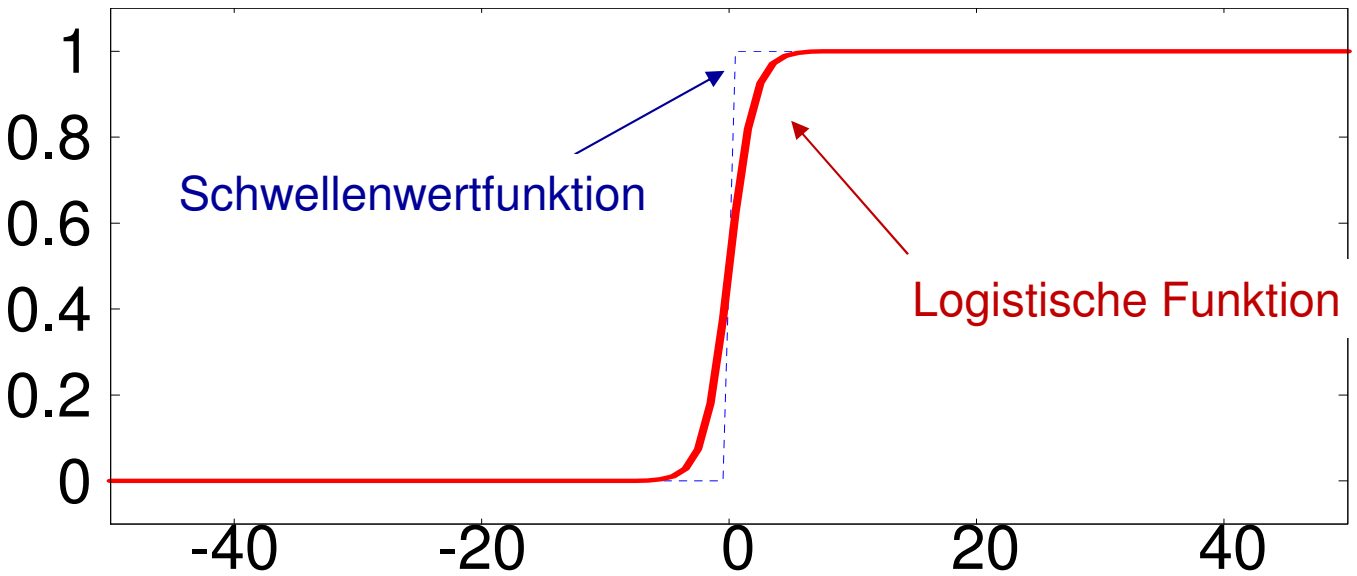
f kann auch anders aussehen

Mögliche Transferfunktionen f(net)

- **Schwellenwertfunktion** (Binäre Funktion): Abrupter Wechsel von 0 („feuert nicht") zu 1 („feuert“)

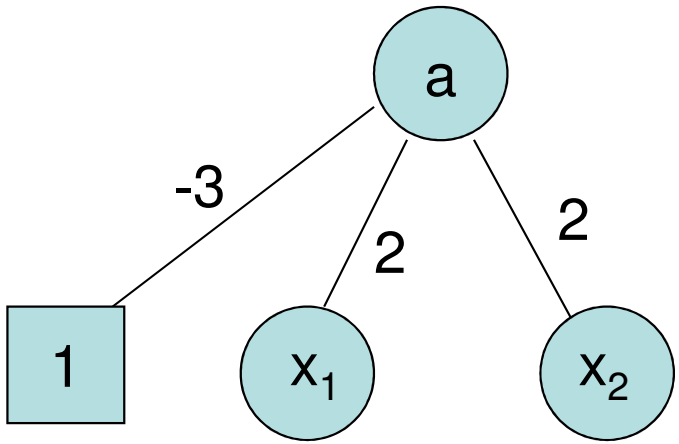
- **Logistische Funktion,** eine sigmoide (=s-förmige) Funktion: Fließender Übergang, wird verwendet, wenn Differenzierbarkeit verlangt wird

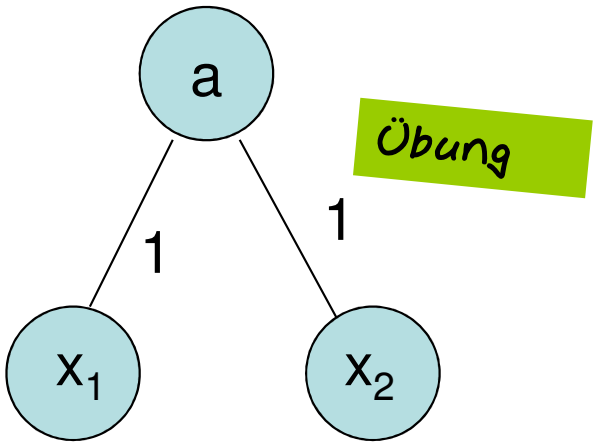
$$a = f(net) = \frac{1}{1 + e^{-net}}$$



Beispiel: Boolesche Funktionen

UND und ?





x_1	x_2	net	a=f(net)
0	0	-3	0
0	1	-1	0
1	0	-1	0
1	1	1	1

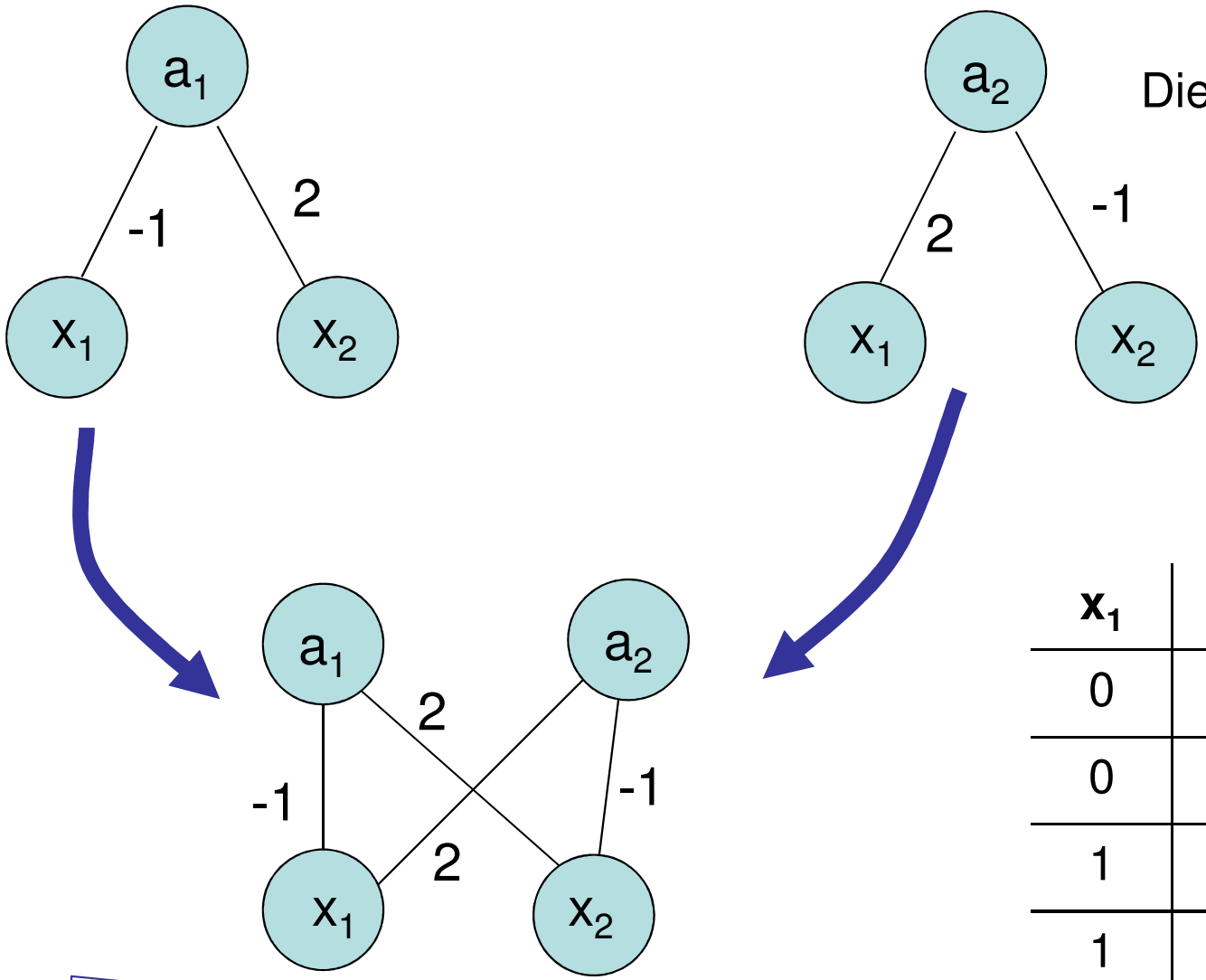
net	a=f(net)

Wir haben jetzt ein formales Modell eines einzelnen Neurons (Perzeptron) und seines Verhaltens.

Künstliche Neuronen werden zu leistungsfähigen Netzen, wenn sie in großer Anzahl zusammengeschaltet werden.

Sie erfahren nun, wie das geht.

Beispiel: Zusammenschalten von Neuronen

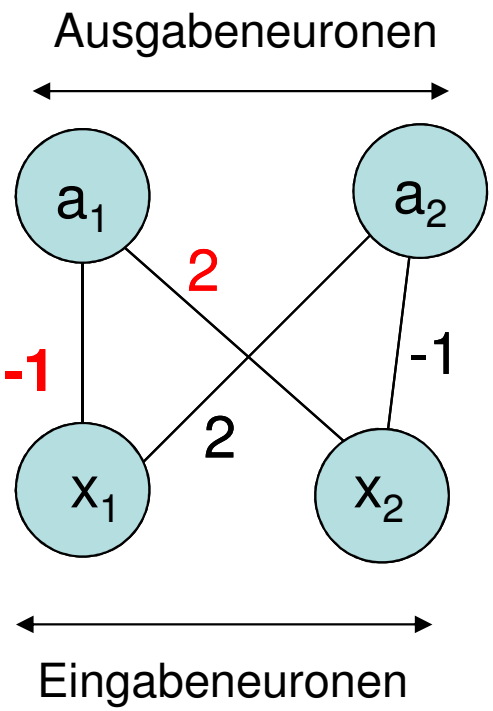


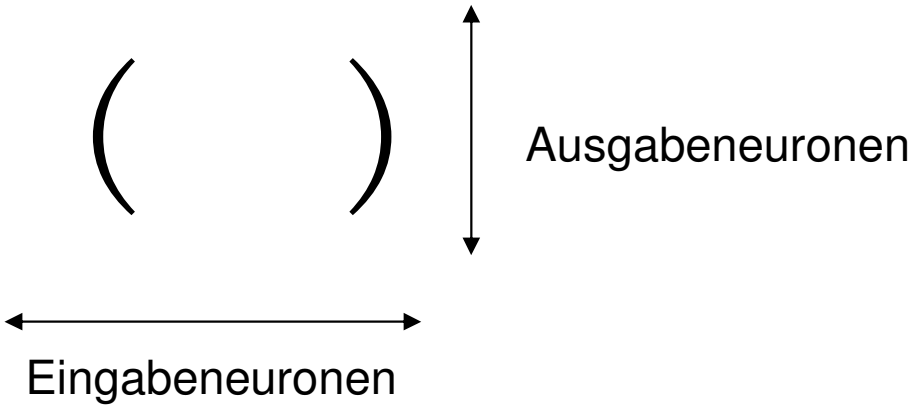
Die Schwellwerte sind hier 0.

x_1	x_2	a_1	a_2
0	0	0	0
0	1	1	0
1	0	0	1
1	1	1	1

Ein neuronales Netz

Wichtungsmatrix





Und mit Vektoren dargestellt:

Sei $x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$ $a = \begin{pmatrix} a_1 \\ a_2 \end{pmatrix}$ sowie Wichtungsmatrix $W = \begin{pmatrix} -1 & 2 \\ 2 & -1 \end{pmatrix}$, dann gilt:

$$net = W \bullet x$$

$$a = f(net)$$

Hier ist net der Vektor der Netzaktivitäten und f(net) der Vektor

$$f(net) = \begin{pmatrix} f(net_1) \\ f(net_2) \end{pmatrix}$$

Für das Beispiel:

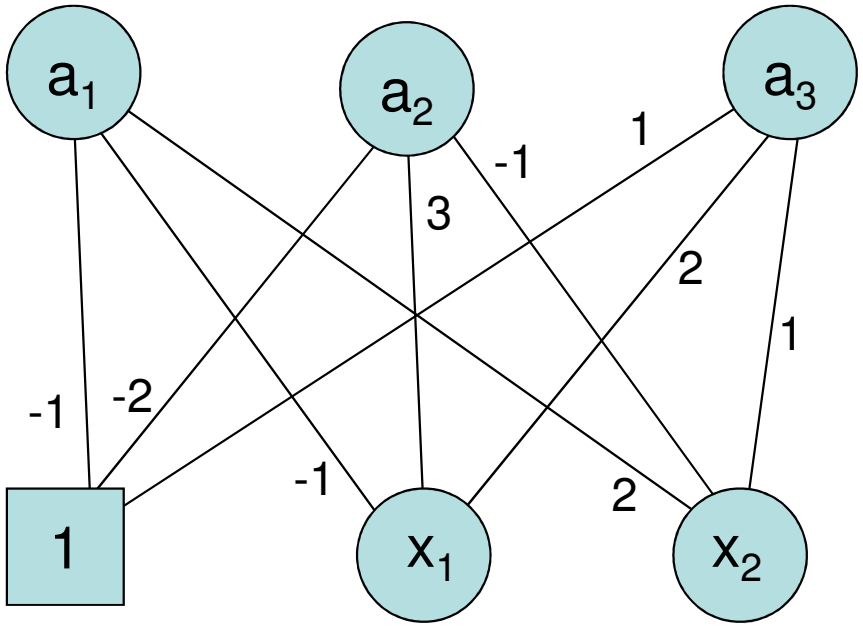
Übung

$$x = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \qquad net = W \bullet x = \begin{pmatrix} -1 & 2 \\ 2 & -1 \end{pmatrix} \bullet \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} -1 \\ 2 \end{pmatrix}$$

$$a = f(net) = \begin{pmatrix} f(-1) \\ f(2) \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

Beispiel eines neuronalen Netzes mit BIAS

Welche logischen Funktionen sind dargestellt?



Ein (einstufiges) neuronales Netz

x_1	x_2	a_1	a_2	a_3
0	0			
0	1			
1	0			
1	1			

Übung

Definition 2: Neuronales Netz (einstufig)

Synapsen

Ein (einstufiges) neuronales Netz ist gegeben durch eine **nxm-Matrix W**, deren Elemente reelle Zahlen sind, sowie durch eine **vektorielleTransferfunktion f**, so dass jedem binären Inputvektor x ein Outputvektor a zugeordnet wird entsprechend der Vorschrift:

$$net = W \bullet e$$

$$a = f(net)$$

Etwas zur Geschichte

- 1969: M. Minsky, S. Papert veröffentlichen das Buch *Perceptrons*, in dem sie nachweisen, dass es wichtige logische Funktionen gibt, die mit dem Formalismus dieses Typs nicht beschreibbar sind, z.B.: XOR

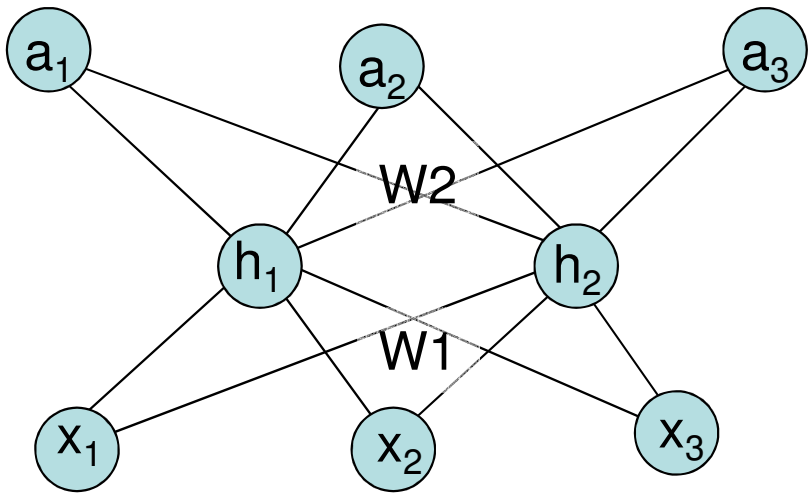
- Niedergang der NN-Forschung, da für Anwendungen nicht mehr interessant, denn ganze Funktionsklassen sind nicht modellierbar

- >10 Jahre später: Entdeckung, dass diese Aussage für mehrstufige Netze nicht gilt

- Seit 1985 gibt es einen geeigneten Lernalgorithmus (Backpropagation-Algorithmus)

> mehrstufige Netze?

Beispiel: Ein dreischichtiges neuronales Netz



- kein BIAS (die Schwellwerte sind 0)
- Eingabewerte x_1, x_2, x_3
- Ausgabewerte a_1, a_2, a_3
- Versteckte Neuronen h_1, h_2
- engl. hidden neurons, hidden layer

- Dreischichtig (3 Neuronenschichten) = zweistufig (2 Wichtungsschichten)

Die Aktivierung der versteckten Neuronen erhält man durch:

$$h = f(net1)$$

$$net1 = W1 \bullet x$$

Hierbei sind ***h*** und ***x*** Vektoren (***h*** = hiddenlayer, ***x*** = Eingabevektor).

Die zweite Stufe wird beschrieben durch:

$$a = f(net2)$$

$$net2 = W2 \bullet h$$

Hier ist ***a*** der Ausgabevektor und ***W2*** die Matrix der Gewichte der zweiten Stufe

Komplexe logische Funktionen

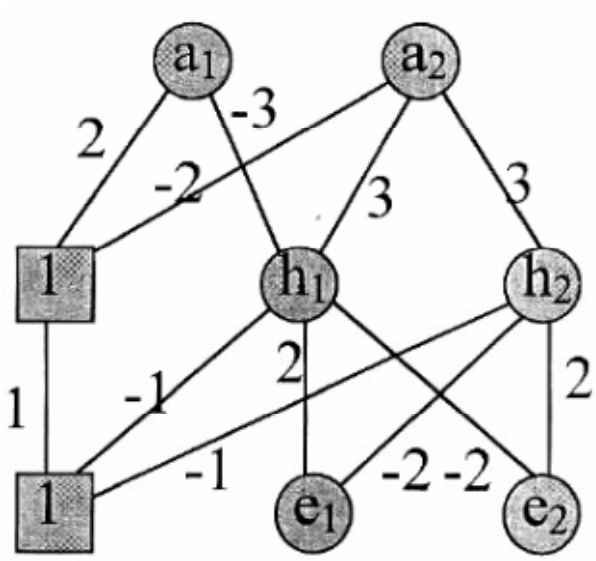
- Mehrere Neuronen lassen sich zusammenschalten, so dass man mehrere Ausgabekanäle erhält.
- Möglich sind auch mehrschichtige Netze. Die Ausgabe der einen Schicht ist die Eingabe in die darüberliegendeSchicht. Einige Funktionen lassen sich nicht mit 2 Neuronen-Schichten realisieren (XOR).
- Man unterscheidet dann Eingabeneuronen, Ausgabeneuronen und versteckte Neuronen (hidden neurons)

Beispiel:

Ein dreischichtiges neuronales Netz,

die Ausgabe a2 realisiert die

XOR-Funktion



Definition 3: Mehrstufiges neuronales Netz

Es sei x ein Eingabevektor und a ein Ausgabevektor sowie h_1, h_2, ... Hilfsvektoren.

Es sei f eine Transferfunktion und W1, W2, W3 .... Matrizen (die nicht notwendig quadratisch sein müssen).

Dann berechnet ein n-stufiges neuronales Netz den Ausgabevektor a aus dem Eingabevektor x wie folgt:

$$h_1 = f(W1 \bullet x)$$

$$h_2 = f(W2 \bullet h_1)$$

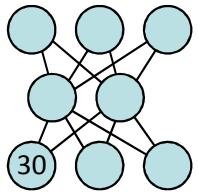
$$h_3 = f(W3 \bullet h_2)$$

.....

$$a = f(Wn \bullet h_{n-1})$$

(Die Schwellwerte sind in den Matrizen enthalten, wenn man z.B. x_1 auf 1 einfriert – Kennen Sie schon).

Die Vektoren h_1, h_2 ,h_3 ... bilden die verborgenen Schichten (hidden layer).

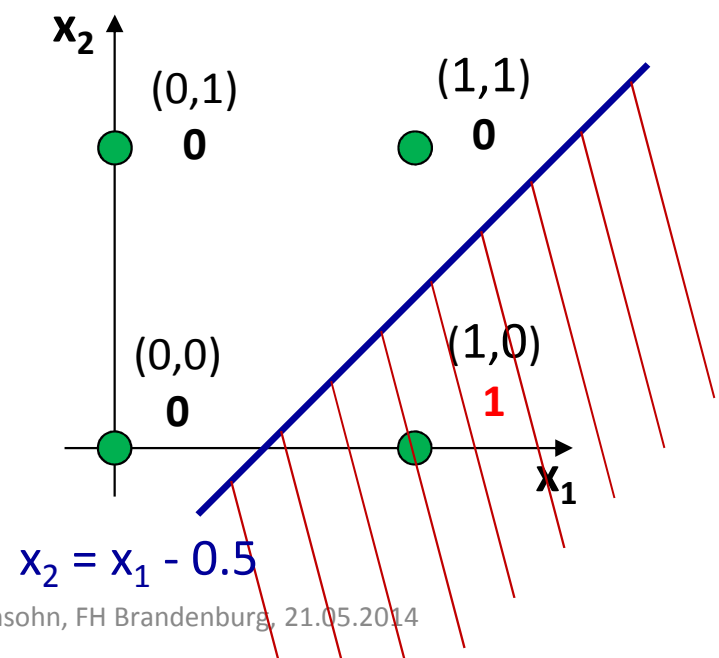


Von der logischen Funktion zum Perzeptron

Gegeben sei:

x_1	x_2	a
0	0	0
0	1	0
1	0	1
1	1	0

Darstellung im Eingaberaum:



1. Gerade so wählen, dass sie die Ausgabewerte 0 und 1 trennt, z.B. $x_2 = x_1 - 0.5$

2. Ungleichung des „feuernden" Teilraumes mit Hilfe der Geraden:

$a = 1$ falls $x_2 < x_1 - 0.5$

3. Ungleichung umformen in net > 0

hier beide Seiten $-x_2$

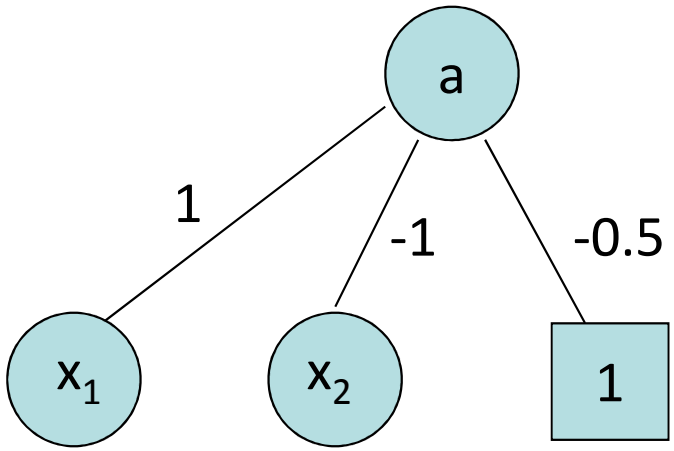
-> $a = 1$ falls $x_1 - x_2 - 0.5 > 0$

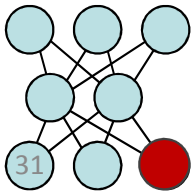
-> $net = x_1 - x_2 - 0.5$!!

4. Wichtungen ablesen

$net = 1 * x_1 + (-1) * x_2 + (-0.5) * 1$

Übung





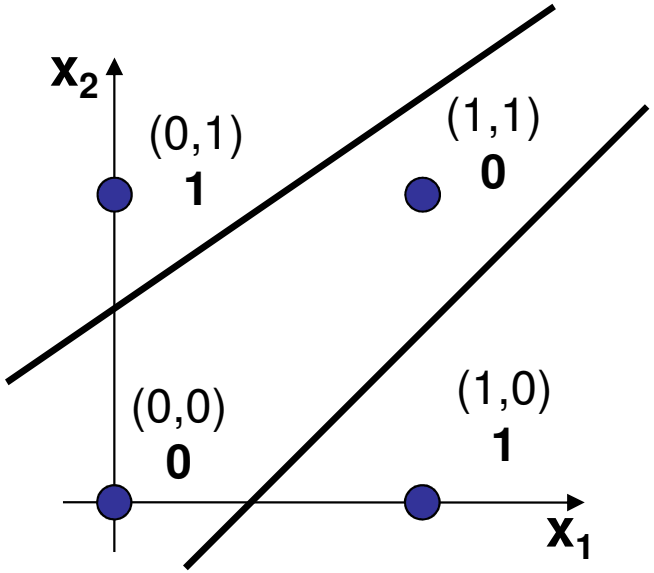
Wann reicht ein einstufiges Netz nicht mehr?

Gegeben sei:

XOR-Funktion

x_1	x_2	a
0	0	0
0	1	1
1	0	1
1	1	0

Darstellung im Eingaberaum:



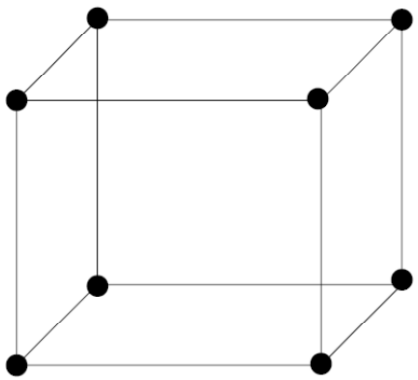
Zwei Geraden notwendig!

Also :

Für XOR gibt es kein einstufiges Netz, da ein einstufiges Netz nur <u>eine</u> lineare Ungleichung auswerten kann.

Darstellbarkeit bei 3 und n Eingabeneuronen

Eingaberaum bei 3 Eingabebits:



Eingabevektoren = Eckpunkte eines Würfels

lineare Teilbarkeit = Beschreibung der Grenzfläche durch eine Ebene ist möglich

Allgemein bei n Eingabeneuronen:

- n Eingabebits
- n-dimensionaler Raum
- (n-1)-dimensionale Hyperebene teilt die Eingabevektoren in 2 Gruppen, die auf 1 bzw. 0 abgebildet werden.

Ausdruckskraft, Repräsentationsfähigkeit

Ein **einstufiges** neuronales Netz kann nur linear teilbare Funktionen darstellen.

Ein **zweistufiges** neuronales Netz darstellen kann jede beliebige Funktion darstellen.

Wie viele versteckte Neuronen sind nötig?

Die vom Netz zu adaptierende Funktion sei gegeben durch:

$$(x_1,a_1),(x_2,a_2),....,(x_n,a_n)$$

In der versteckten Schicht müssen n verschiedene Belegungen möglich sein:

Bei k hidden neurons:

$$2^k \geq n \text{ bzw. } k \geq ld(n)$$

<u>Beispiel:</u> n=17 verschiedene Muster zu lernen -> k > …

Was haben wir erreicht?

- Dreischichtige Netze (= 2 Wichtungsschichten) sind universelle Approximatoren, d.h. mit einem solchen Netz lassen sich alle binären Funktionen darstellen (mathematische Funktionen, Steueranweisungen für Roboter, Prognosen, etc.)

Wo kommen die Gewichte her? Vorschrift zur Berechnung der Gewichte w existiert für die meisten Anwendungen nicht. Lassen sich - wie beim biologischen Vorbild - die „richtigen" Gewichte erlernen?

Ja, man benötigt

- **Trainingsdaten**: Eine Menge von Vektorpaaren (Eingabevektor, gewünschter Ausgabevektor)
- **Lernalgorithmus**, der aus Trainingsdaten und aktueller Ausgabe des Netzes die Wichtungsänderungen berechnet

Beispiel 1: Überprüfung der Kreditwürdigkeit

- Mit Hilfe lernfähiger Neuronaler Netze wird in diesem Anwendungsgebiet das Entscheidungsverhalten von Domain-Experten - hier z.B. den in den Kreditvergabeprozess einer Bank eingebundenen Mitarbeiter - abgebildet.
- Die Entscheidung lautet „kreditwürdig" oder "nicht kreditwürdig"
- Einbezogene Merkmale:
  Merkmale der persönlichen, wirtschaftlichen und rechtlichen Situation des Antragstellers
- Trefferquoten von über 70%

Erreicht wurde u.a.:

- Objektivierung der Vergabepolitik
- Zeitersparnis (die Beurteilung dauert noch 2 Minuten)
- Einsparung
- Entlastung der Mitarbeiter von Routinetätigkeiten

(Quelle: Deutsche Allgemeine Treuhand AG, 1996)

Beispiel 2: Lernen von Buchstaben (JNNS)

- Eingabevektoren der Länge 35: (eine 5x7-Grauwertmatrix)
- Ausgabevektoren der Länge 26 (für die 26 Buchstaben des Alphabets)
- Hidden Layer: 10 versteckte Neuronen
- Initialisierung der Gewichte: Setze für alle Gewichte (kleine) Zufallszahlen

In der Lernphase werden dem Netz viele bekannte Zuordnungen präsentiert:

- Wähle einen beliebigen Eingabevektor x
- Berechne mit den momentanen Gewichten den Ausgabevektor a
- Vergleiche den Ausgabevektor a mit dem Zielvektor y. Falls a=y, setze mit einem neuen Eingabevektor fort. Andernfalls verbessere zuvor noch die Gewichte nach einer geeigneten Korrekturformel.

Praktisch in der nächsten Woche …

Rückblick

Biologische Vorbilder Gehirn und Neuron

Wie wird ein Neuron im Computer abgebildet?

- Neuronenmodell Perzeptron
- Schwellwert als ON-Neuron
- Netzaktivität, Transferfunktion (Aktivierungsfunktion)
- Beispiel UND, ?

Welche Funktionen können ausgedrückt werden?

Wie werden Neuronen zu Netzen verbunden?

- Zusammenschalten (mehrere Ausgänge, mehrere Schichten)
- Vektordarstellung

Anwendungsbeispiele

weiter mit: Lernen

- Lernen im Gehirn

  Die Hebbsche These

- Lernen am Perzeptron

  Zwei Phasen

  Die Delta-Regel

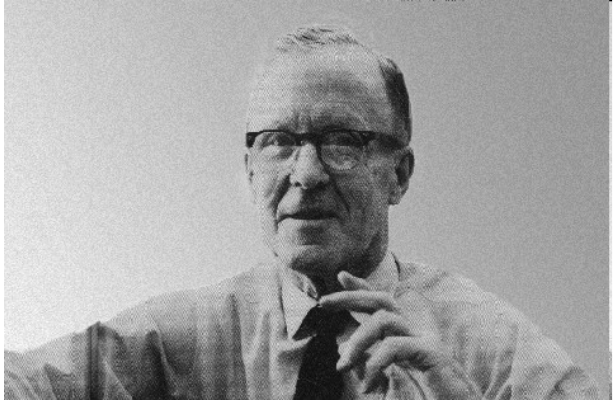
- Lernen am Multilayer-Perzeptron

  Das Backpropagation-Verfahren

- Maschinelles Lernen

  Einen Schritt zurück

Lernen im Gehirn – Hebbsche These



Wie lernt das Gehirn?

Hierzu formulierte 1949 Donald O. Hebb die Hebbsche These:

„When an axion of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased.„

Was bedeutet das?

- Bei gleichzeitiger Aktivität der präsynaptischen und postsynaptischen Zelle wird die Synapse verstärkt.
- Fire together – wire together

Synaptische Plasitizität

- neuronale Mechanismen bis heute nicht geklärt

Zwei Phasen der Anwendung

- Gegeben ist die darzustellende (unbekannte) logische Funktion. Konkret liegen Eingabevektoren x vor, denen Ausgabevektoren y zugeordnet sind. Diese Funktion soll durch ein Netz dargestellt werden.
- Für das Netz ist eine Topologie zu wählen. (Heuristiken)

1 Lernphase:

- Die **Gewichte** sind so zu bestimmen, dass das Netz in der gewählten Topologie die vorgegebene Funktion darstellt
- Rechenintensiv, aber einmalig
- Minimierung einer Fehlerrate

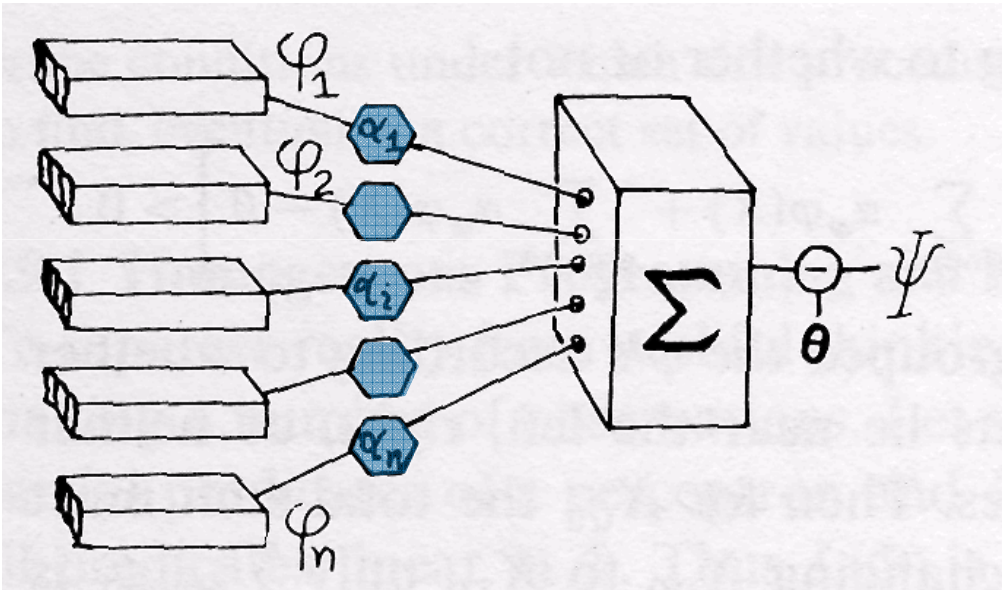
2 Recall-Phase (Einsatzphase):

- Nachdem die Gewichte gelernt wurden, ist das Netz beliebig oft einsetzbar
- Geringer Rechenaufwand

Lernen beim Perzeptron

- Frank Rosenblatt 1958
- erstes lernfähiges Neuronenmodell - das Perzeptron

- Wie sind die Wichtungen zu ändern?
  - Lernregel: **Delta-Regel**



Perzeptron
Quelle: Perceptrons, Marvin Minsky & Seymour Papert, 1969

Lernen beim Perzeptron – Delta-Regel

neue i-te Wichtung: $w_i(t+1) = w_i(t) + \Delta w_i$

alte Wichtung

Wichtungsänderung

Wichtungsänderung: $\Delta w_i = \eta \cdot x_i \cdot \delta$

Lernrate eta

Eingabe i

Fehler des Neurons

Fehler: $\delta = y - a$

Soll

Ist (Ausgang des Neurons)

Schwellwert hier als Wichtung: $w_{n+1} = -\Theta$ und $x_{n+1} = 1$
Lernt immer mit.

Beispiel

$$w_i(t+1) = w_i(t) + \eta \cdot x_i \cdot (y - a)$$

gegeben ein Netz: $w_1(t) = -4$, $w_2(t) = 5$, $\Theta(t) = 2$,
d.h. als ON-Neuron $x_3 = konstant\, 1$, $w_3 = -2$
zu lernen: $x_1 = 1$, $x_2 = 0$, $Sollwert\ y = 1$
Lernrate: $\eta = 0.3$

Berechnen Sie die Wichtungsänderungen!

Netzaktivität: $net = x_1 w_1 + x_2 w_2 + x_2 w_3 = 1 \cdot -4 + 0 \cdot 5 + 1 \cdot -2 = -6$
Aktivität: $a = f(net) = f(-6) = 0$
Fehler: $\delta = Soll - Ist = y - a = 1 - 0 = 1$

Wichtungsänderung 1: $\triangle\, w_1 = \eta x_1 \delta = 0.3 \cdot 1 \cdot 1 = 0.3$
neue Wichtung: $w_1(t+1) = w_1(t) + \triangle\, w_1 = -4 + 0.3 = -3.7$

analog Wichtungsänderung 2: $\triangle\, w_2 = \eta x_2 \delta = 0.3 \cdot 0 \cdot 1 = 0$
neue Wichtung: $w_2(t+1) = w_2(t) + \triangle\, w_2 = 5 + 0 = 5$
w_2 lernt nicht wegen $x_2 = 0$

analog Wichtungsänderung 3: $\triangle\, w_3 = \eta x_3 \delta = 0.3 \cdot 1 \cdot 1 = 0.3$
neue Wichtung: $w_3(t+1) = w_3(t) + \triangle\, w_3 = -2 + 0.3 = -1.7$
w_3 lernt immer wegen ON-Neuron
$\Theta(t+1) = 1.7$ Neuron feuert nun leichter

Test mit gleicher x-Eingabe führt zu $net = -3.7 - 1.7 = -5.4$, damit näher am Feuern,
$a = f(-5.4) = 0$ reicht aber noch nicht

Beispiel

$$w_i(t+1) = w_i(t) + \eta \cdot x_i \cdot (y - a)$$

Berechnen Sie die Wichtungsänderungen!

gegeben ein Netz: $w_1(t) = -4$, $w_2(t) = 5$, $\Theta(t) = 2$,
d.h. als ON-Neuron $x_3 = konstant\, 1$, $w_3 = -2$
zu lernen: $x_1 = 1$, $x_2 = 0$, $Sollwert\, y = 1$
Lernrate: $\eta = 0.3$

Netzaktivität: $net = x_1 w_1 + x_2 w_2 + x_2 w_3 = 1 \cdot -4 + 0 \cdot 5 + 1 \cdot -2 = -6$
Aktivität: $a = f(net) = f(-6) = 0$
Fehler: $\delta = Soll - Ist = y - a = 1 - 0 = 1$

Wichtungsänderung 1: $\triangle\, w_1 = \eta x_1 \delta = 0.3 \cdot 1 \cdot 1 = 0.3$
neue Wichtung: $w_1(t+1) = w_1(t) + \triangle\, w_1 = -4 + 0.3 = -3.7$

analog Wichtungsänderung 2: $\triangle\, w_2 = \eta x_2 \delta = 0.3 \cdot 0 \cdot 1 = 0$
neue Wichtung: $w_2(t+1) = w_2(t) + \triangle\, w_2 = 5 + 0 = 5$
w_2 lernt nicht wegen $x_2 = 0$

analog Wichtungsänderung 3: $\triangle\, w_3 = \eta x_3 \delta = 0.3 \cdot 1 \cdot 1 = 0.3$
neue Wichtung: $w_3(t+1) = w_3(t) + \triangle\, w_3 = -2 + 0.3 = -1.7$
w_3 lernt immer wegen ON-Neuron
$\Theta(t+1) = 1.7$ Neuron feuert nun leichter

Test mit gleicher x-Eingabe führt zu $net = -3.7 - 1.7 = -5.4$, damit näher am Feuern,
$a = f(-5.4) = 0$ reicht aber noch nicht

Lernen beim Perzeptron - Algorithmus

Das Training des Perzeptrons erfolgt nach folgendem Algorithmus:

Initialisiere alle Wichtungen und den Schwellwert mit beliebigen Werten, z. B. 0.
Wiederhole, bis alle Muster korrekt klassifiziert werden:

Wähle den nächsten (oder einen zufälligen) Eingabevektor x.
Berechne net und die Klassifikation a.
Berechne den Fehler δ zur gewünschten Ausgabe.
Modifiziere Wichtungen und Schwellwert nach Gleichung (10.3) und (10.4).

Funktioniert das sicher für die repräsentierbaren Funktionen? Ja!

Satz 10.2 (Konvergenz des Perzeptron-Lernverfahrens)
Der Perzeptron-Lernalgorithmus terminiert für linear separierbare Boolesche Funktionen.

Beispiel Lernen der ODER-Funktion

gut zum Nachrechnen

Tab. 10.2: Lernverlauf für das ODER-Prädikat

Zyklus	x_1	x_2	y	net	a	δ	Δw_1	Δw_2	$\Delta\Theta$	w_1	w_2	Θ
Init										0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0
	0	1	1	0	0	1	0	1	-1	0	1	-1
	1	0	1	0	1	0	0	0	0	0	1	-1
	1	1	1	1	1	0	0	0	0	0	1	-1
2	0	0	0	0	1	-1	0	0	1	0	1	0
	0	1	1	1	1	0	0	0	0	0	1	0
	1	0	1	0	0	1	1	0	-1	1	1	-1
	1	1	1	2	1	0	0	0	0	1	1	-1
3	0	0	0	0	1	-1	0	0	1	1	1	0
	0	1	1	1	1	0	0	0	0	1	1	0
	1	0	1	1	1	0	0	0	0	1	1	0
	1	1	1	2	1	0	0	0	0	1	1	0
4	0	0	0	0	0	**0**	0	0	0	1	1	0
	0	1	1	1	1	**0**	0	0	0	1	1	0
	1	0	1	1	1	**0**	0	0	0	1	1	0
	1	1	1	2	1	**0**	0	0	0	1	1	0

Historie

1. Delta-Regel nicht anwendbar für Multilayer-Perzeptron und
2. beschränkte Ausdruckskraft des Perzeptrons

   (Minsky & Papert 1969)

⇒ Stagnation der künstlichen neuronalen Netze

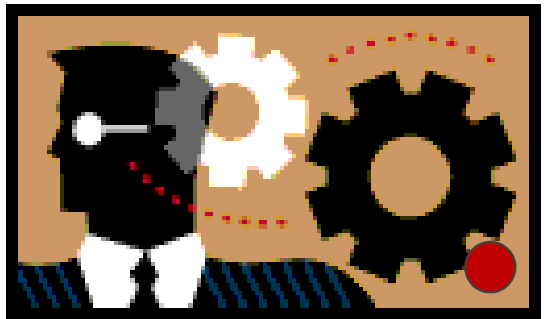
Wie können MLPs lernen? **. . . ? . . . . . .? . .? . . . . !**

– **1986 Rumelhart, Hinton, Williams** (und andere vor ihnen)

zeigen dass die Delta-Regel eine Form des **Gradientenabstiegs** ist und verallgemeinern zur „Generalisierten Delta-Regel“, dem sog. **Backpropagation-Verfahren**

– dafür notwendig: differenzierbare Aktivierungsfunktion

=> Renaissance der künstlichen neuronalen Netze



Ende Teil1

Beim nächsten Mal:

- Wie lernt ein Multilayerperzeptron?
- Maschinelles Lernen – The Big Picture
- Neuronale Netze praktisch

Übung:

- Vom Netz zur Funktion
- Von der Funktion zum Netz
- Lernen Perzeptron