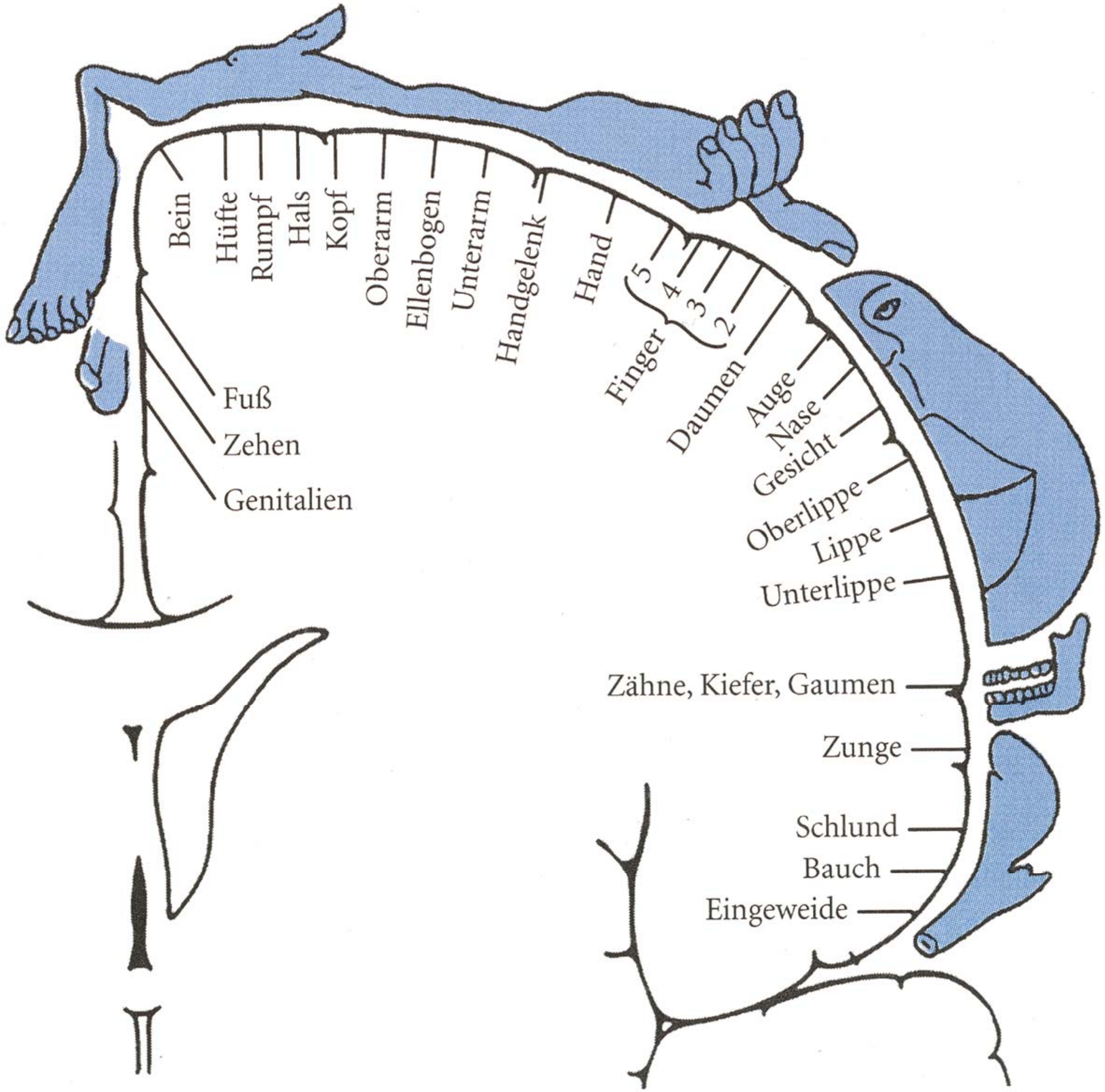
Selbstorganisierende Karten

FH Brandenburg

Dipl.-Inform. I. Boersch
FB Informatik und Medien

FACHHOCHSCHULE BRANDENBURG

- SOM sind
    - Einfach
    - Leistungsfähig
    - Physiologisch motiviert
    - Anschaulich



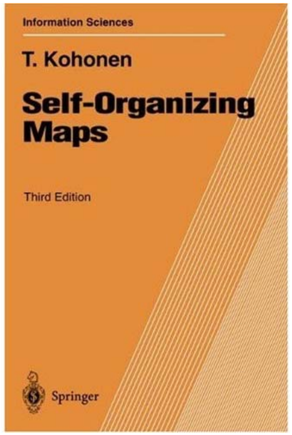
Selbstorganisierende Karten

- **SOM** von engl. **self-organizing maps**
- auch **Kohonen-Karten** nach dem Erfinder Teuvo Kohonen

- T. Kohonen. Self-Organized Formation of Topologically Correct Feature Maps. Biological Cybernetics, 43:59–69, 1982.

- Ausführlich in: Gut erklärt in Abschnitt 10.4 in:

Teuvo Kohonen, 2009

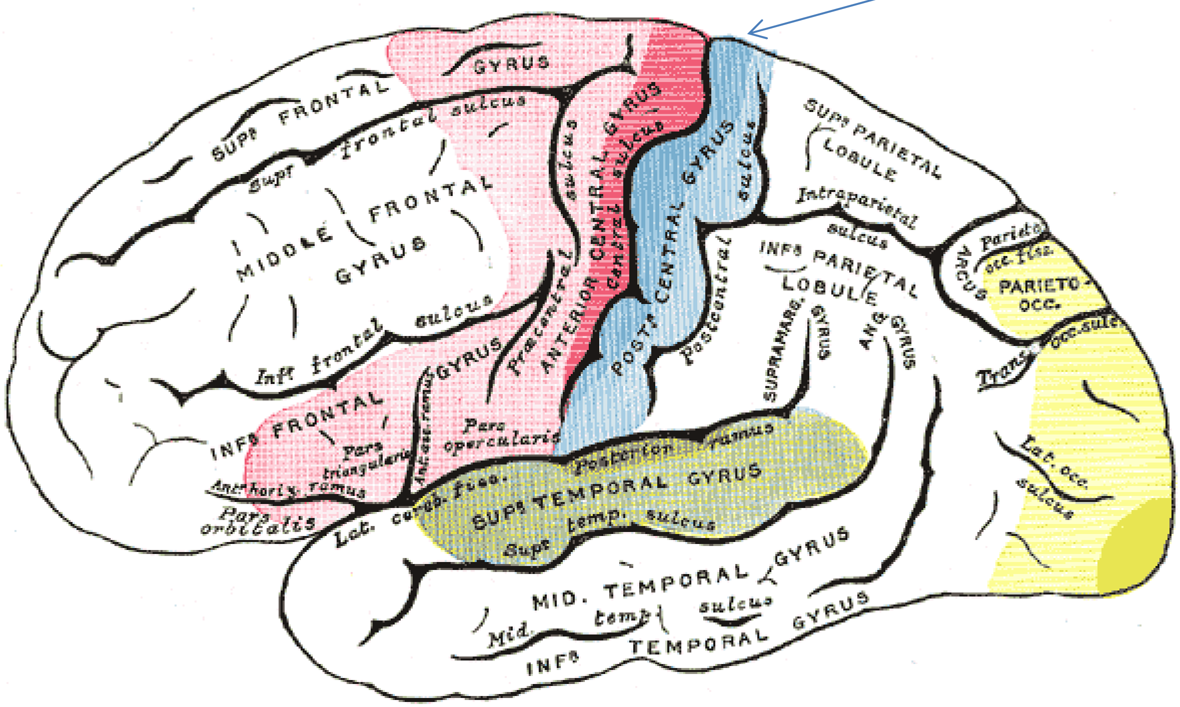


Somatosensorische Felder

- Im Kortex des menschlichen Gehirns finden sich abgrenzbare Neuronengebiete, die beim Auftreten bestimmter Sinnesreize aktiviert werden – sogenannte ***somatosensorische Felder.***

Somatosensorischer Kortex

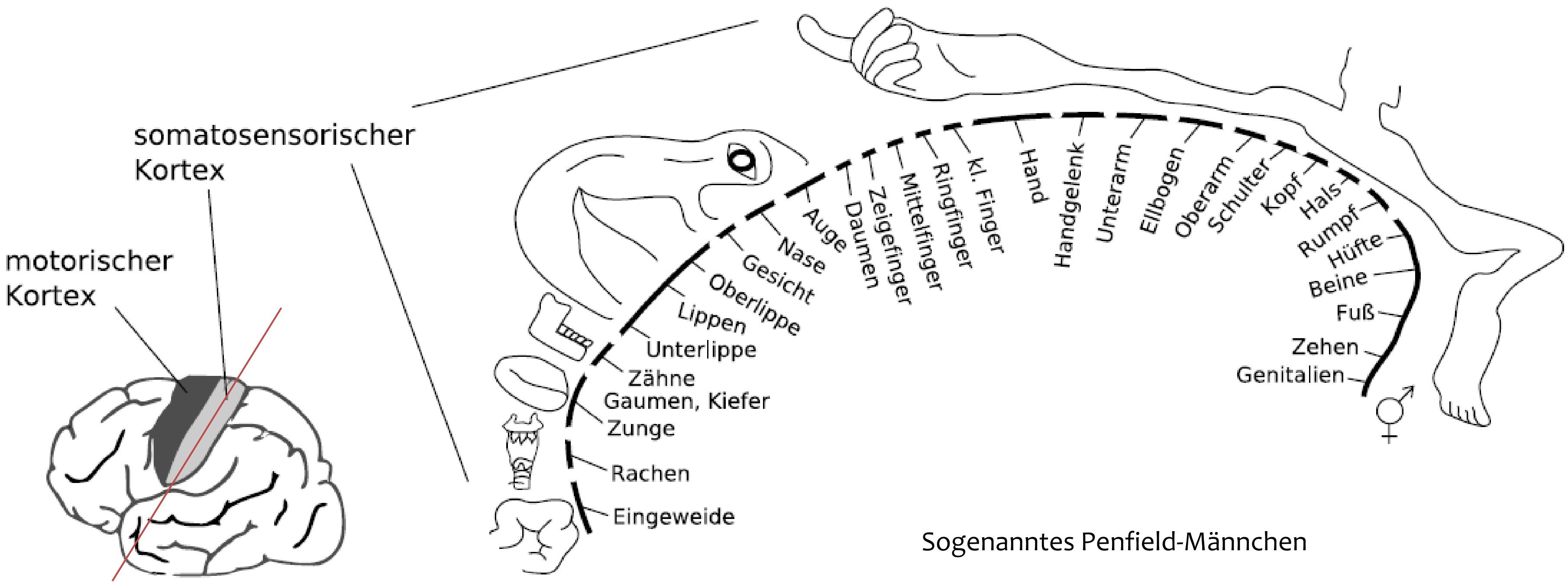
Blickrichtung



Grafik aus Wikipedia

Somatosensorische Felder

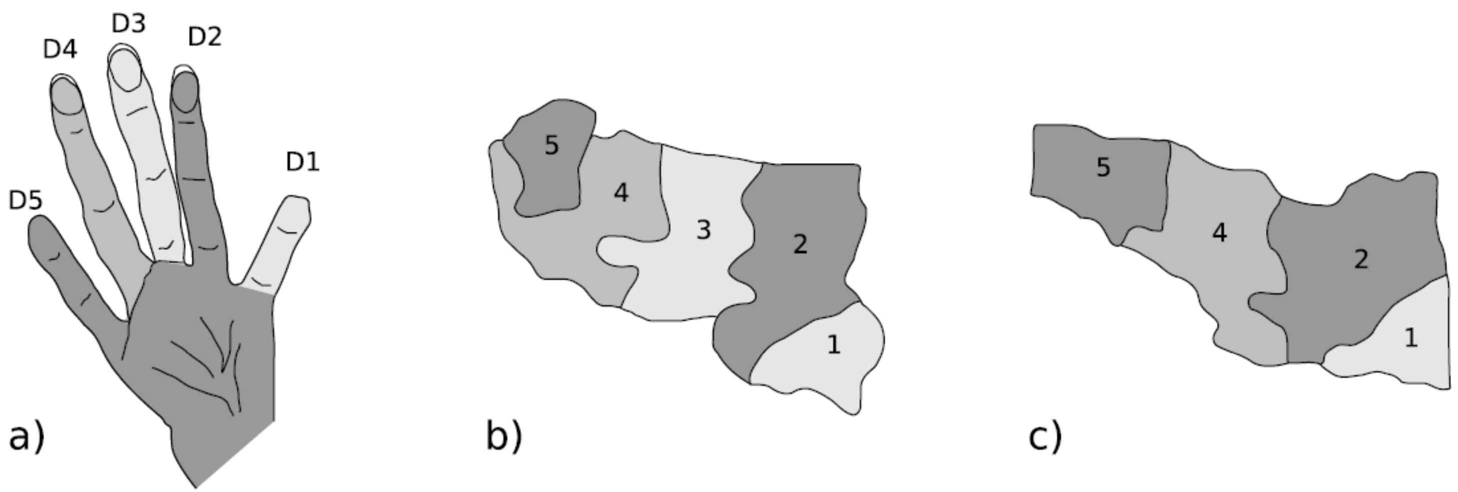
- Felder, die **benachbarten** Sinnesorganen entsprechen, liegen hierbei auch auf der Hirnrinde **nebeneinander**., d.h. die Topologie wird erhalten.
- Entsprechendes gilt für die motorischen Felder.



Sogenanntes Penfield-Männchen

Dynamische Karten in der Großhirnrinde

- Diese topologieerhaltende Abbildung (von Topologie = Nachbarschaft) auf wenige Dimensionen, hier eine Fläche, wird als **Karte** bezeichnet.
- Die Karten im Gehirn kommen bei allen Säugetieren vor und sind **dynamisch**.
  - Beispiel: Beim Verlust von Körperteilen wird der Platz des entsprechenden Feldes mit der Zeit von anderen Feldern eingenommen.



a) Schema einer Affenhand b) Repräsentation der Hand im Kortex c) 2 Monate nach Amputation von Finger D3 (Merzenich , 1984)

Selbstorganisierende Karten

- SOM sind ein Modell der somatosensorischen Felder
- Unüberwachtes Lernen
  - Kein Lehrer
  - Keine richtigen Ergebnisse, kein wertendes Feedback
  - Repräsentation bildet sich ausschließlich aus den Eingabedaten
- Kartentopologie wird vorgegeben

Ziel ist, ähnliche Eingabevektoren benachbart im Ausgaberaum abzubilden.

- Hierbei hat der Ausgaberaum meist weniger Dimensionen als der Eingaberaum, das Verfahren führt so zu einer Dimensionsreduktion der Daten.

Beispiel ‚Zoo'

- gegeben: Tiere durch Eigenschaften beschrieben (Eingaberaum)
- gesucht: Anordnung auf einer 2D-Karte (Ausgaberaum), so dass <u>ähnliche</u> Tiere <u>benachbart</u> sind

- Diplomarbeit von Benjamin Hoepner: Entwurf und Implementierung einer Applikation zur Visualisierung von Lernvorgängen bei Selbstorganisierenden Karten
- http://ots.fh-brandenburg.de/diplomarbeit-von-benjamin-hoepner.html
- Applikation Sombrero: Torus 80x90, a=1; exp=1; r=50; exp=0.995; s=20000

Beispiel ‚Zoo'

Insgesamt 17 Attribute

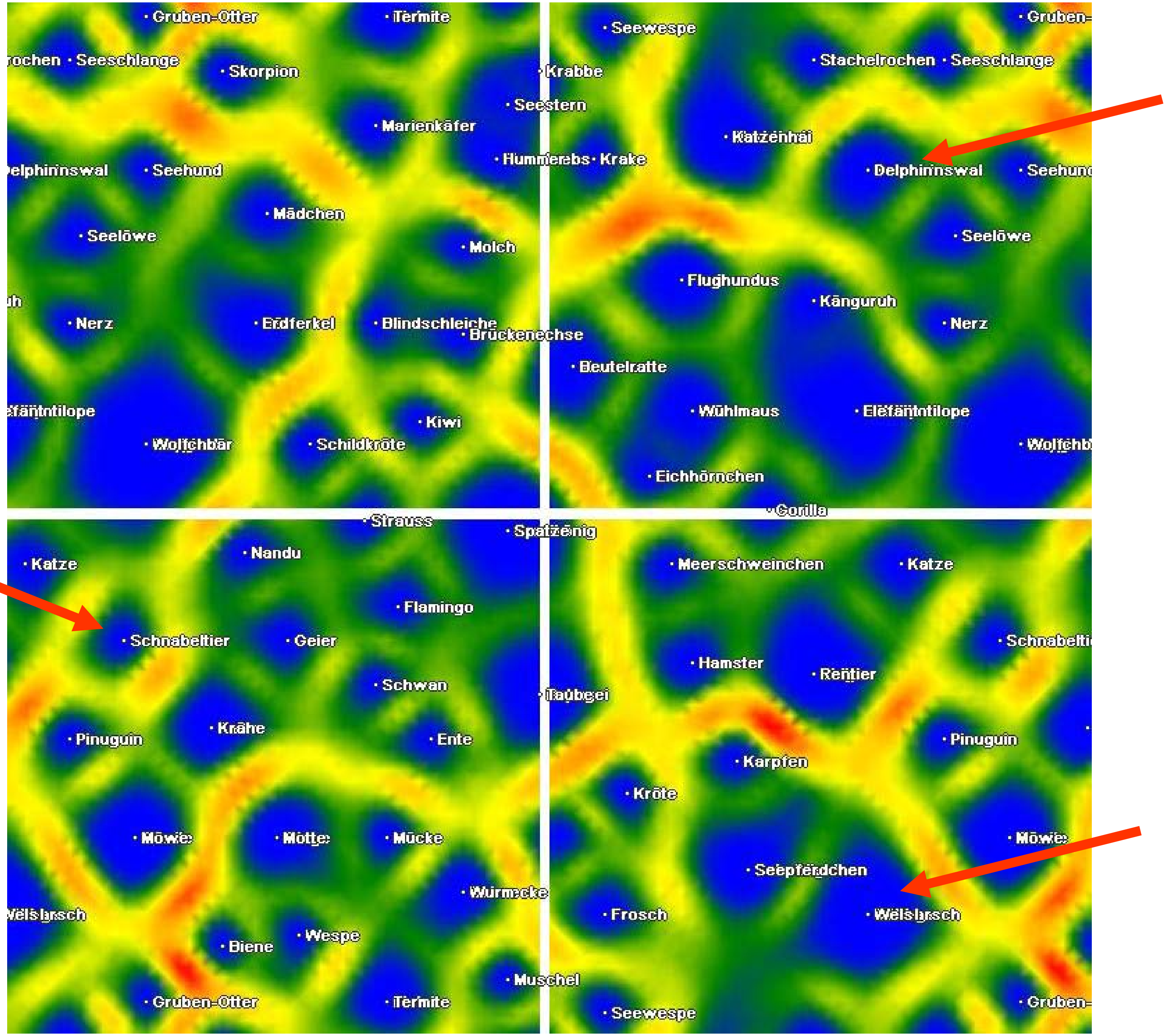
animal	hair	feathers	eggs	milk	airborne	…
Erdferkel	true	false	false	true	false	false
Antilope	true	false	false	true	false	false
Seebarsch	false	false	true	false	false	true
Bär	true	false	false	true	false	false
Eber	true	false	false	true	false	false
Büffel	true	false	false	true	false	false
Kalb	true	false	false	true	false	false
Karpfen	false	false	true	false	false	true
Wels	false	false	true	false	false	true
Meerschweinchen	true	false	false	true	false	false
Gepard	true	false	false	true	false	false
Huhn	false	true	true	false	true	false
Döbel	false	false	true	false	false	true
Muschel	false	false	true	false	false	false
Krabbe	false	false	true	false	false	true
…						

Ähnliche Tiere

Insgesamt 101 Tiere

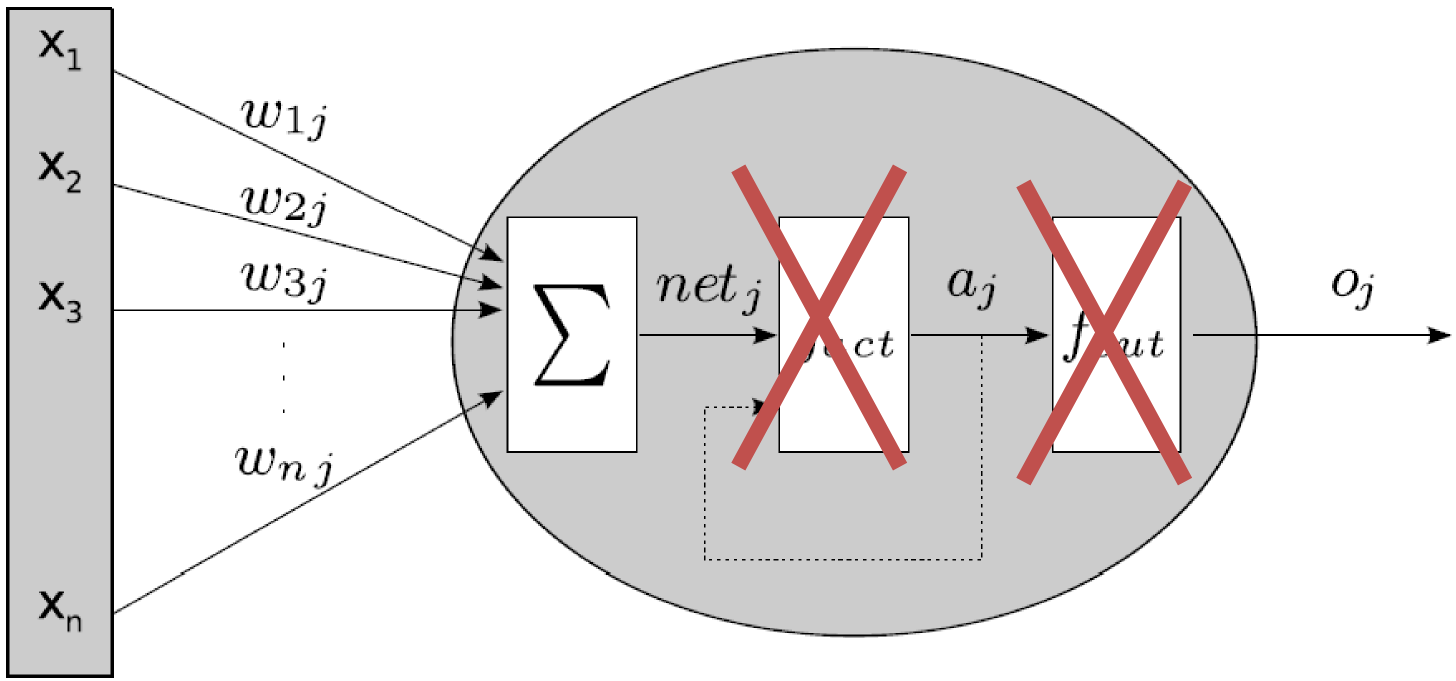
Beispiel ‚Zoo'

- Die erzeugte Karte
- Sind ähnliche Tiere benachbart?
- Reduktion von 17 auf 2 Dimensionen



Modell eines Neurons

- SOM verwenden ein sehr einfaches Neuronenmodell ohne Aktivierungsfunktion



Modell eines Neurons

- SOM verwenden ein sehr einfaches Neuronenmodell ohne Aktivierungsfunktion

$$o_j = net_j = \sum_i w_{ij} \cdot x_i$$

- **Skalarprodukt** aus dem Wichtungsvektor w des Neurons und dem Eingabevektor x

$$o_j = net_j = \sum_i w_{ij} \cdot x_i = \bar{w} \cdot \bar{x} = |\bar{w}||\bar{x}|cos(\alpha)$$

- Wann ist das Skalarprodukt besonders groß und damit das Neuron stark aktiviert?

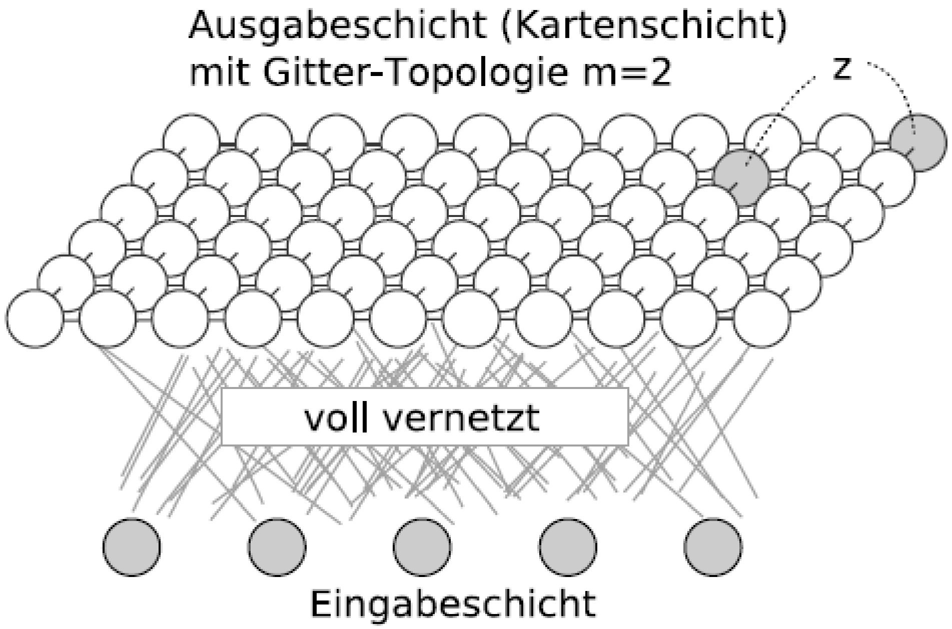
Erregung durch einen Eingabevektor x

- Skalarprodukt aus x und w wird groß, wenn
  - Kleiner Winkel zwischen x und w
  - Komponentenweise Übereinstimmung
  - x und w ähnlich sind

  - Das Neuron reagiert am besten auf Eingabevektoren, die seinem Wichtungsvektor ähneln
  - „Für derartige Muster bin ich zuständig“

Aufbau einer SOM

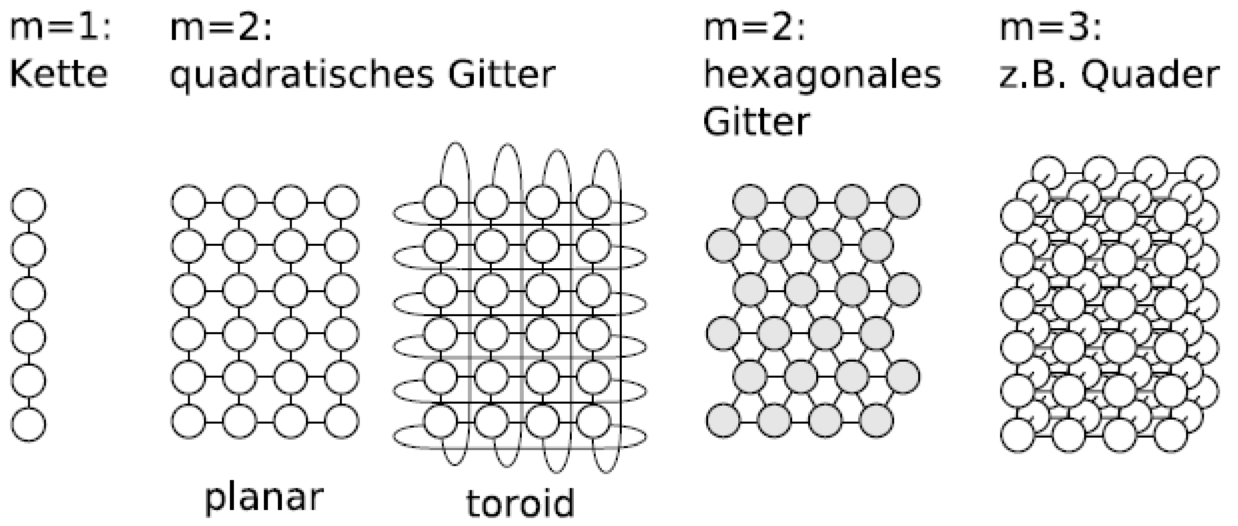
- Eine SOM besteht aus einer Eingabe- und einer Ausgabeschicht.
- Jedes Eingabeneuron ist mit jedem Ausgabeneuron verbunden.
- Innerhalb der Ausgabeschicht ist für jedes Neuron ein Ort definiert und davon abgeleitet zwischen zwei Neuronen ein Abstand z.



z =

Topologie der Ausgabeschicht

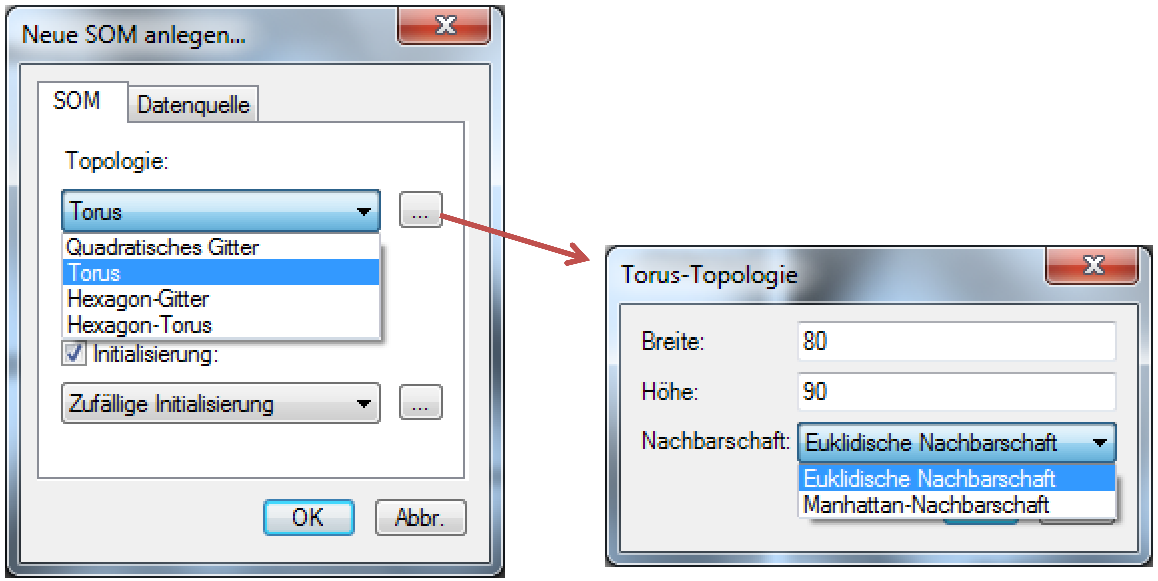
- Topologie wird vorher festgelegt
- Nachbarschaft der Ausgabeneuronen (wer kann wen beeinflussen)
- Oft m-dimensionales quadratisches Gitter, bspw. m =2 (Karte), aber auch m = 1 (Kette) und m = 3
- Vermeiden von Randeffekten durch Verbinden der Ränder

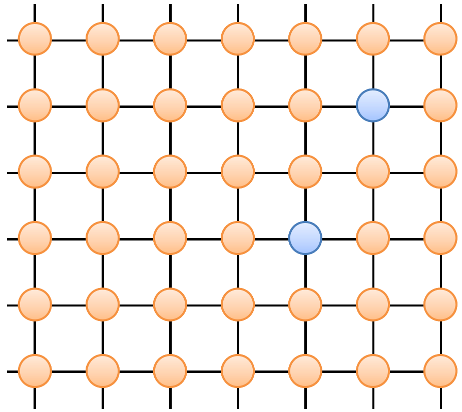


z =

Topologien in Sombrero

- Quadratisches oder hexagonales Gitter
- Jeweils als Torus (Donut) oder flach
- Abstand z zweier Ausgabeneuronen als euklidischer Abstand oder Manhattan-distanz





Z_Euklid = ?
Z_Manhattan = ?

Aktivierung und laterale Inhibition

- Beim Anlegen eines Musters x an die Eingabeschicht erregen sich die Ausgabeneuronen entsprechend ihrer Wichtungen.
- Bei biologischen Neuronen hemmen sich dabei benachbarte Neuronen gegenseitig: laterale Inhibition, bspw. in der Netzhaut
- Folge: Bildung eines Erregungszentrums um ein maximal erregtes Neuron: Gewinnerneuron

- Vereinfacht in der SOM: es gewinnt das Neuron c, dessen Wichtungsvektor w_j am besten mit dem Eingabevektor x übereinstimmt:

$$c = \arg\max_j (x \cdot w_j) = \arg\max_j \sum_i x_i \cdot w_{ij} = \arg\max_j (net_j)$$

c ist der Index des Gewinnerneurons

- Welches Neuron gewinnt?

	x1	x2	x3
	0,32	0,81	0,26

j	w1	w2	w3
1	1,00	0,75	0,00
2	0,51	0,08	0,99
3	0,00	0,84	0,72
4	0,14	0,00	1,00
5	0,89	0,51	0,07
6	0,90	1,00	0,62
7	0,88	0,79	0,86

Training einer SOM

- Initialisierung der Wichtungen mit Zufallszahlen
- Eingabemuster wiederholt anlegen
  - Gewinnerneuron bestimmen
  - Gewinnerneuron lernt
  - Nachbarneuronen des Gewinnerneurons lernen
- Ende nach Zyklenanzahl oder ausreichender Abbildungsgüte

Das Gewinnerneuron lernt

- Gewinnerneuron c
- Wichtungsvektor des Gewinnerneurons w_c
- Eingabevektor x

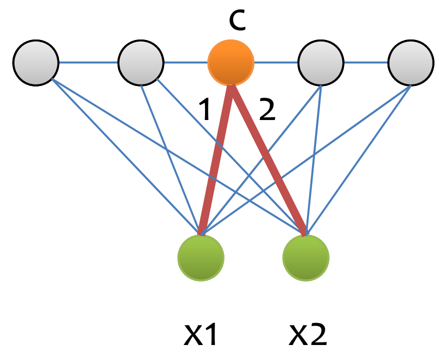
- Wichtungsänderung verringert den Abstand des Wichtungsvektors w_c des Gewinnerneurons c zum Eingabevektor x:

$$w_c = w_c + \triangle w_c$$

$$\triangle w_c = \eta \cdot (x - w_c)$$

- Verschiebung von w_c in Richtung x. Bei $\eta = 1$ würde der Wichtungsvektor w_c durch den Eingabevektor x ersetzt, es gilt also $0 < \eta < 1$.

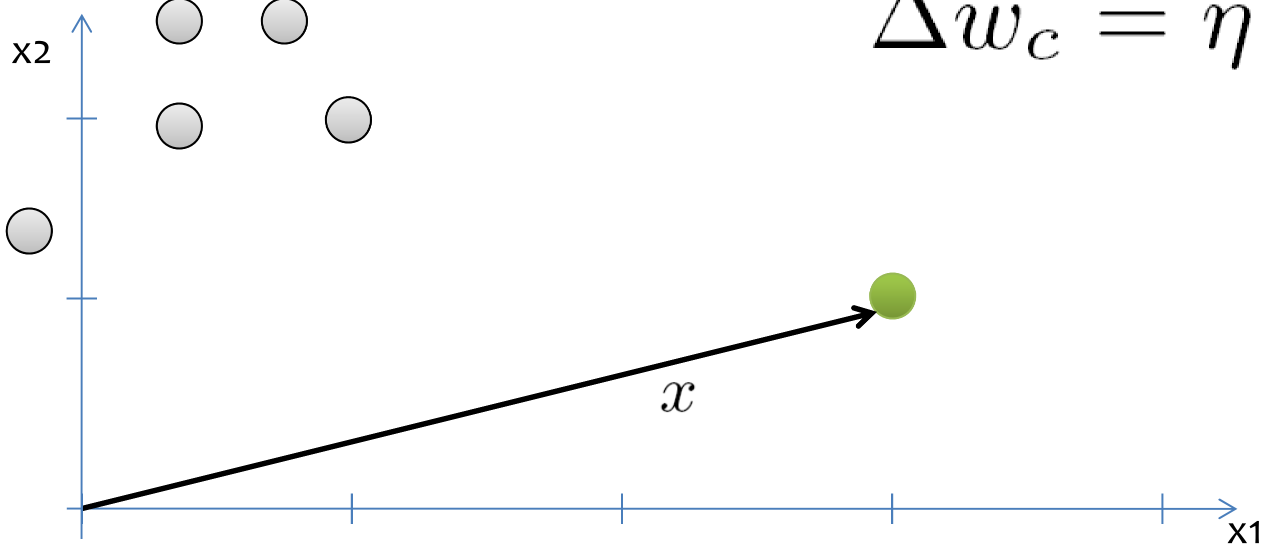
Beispiel



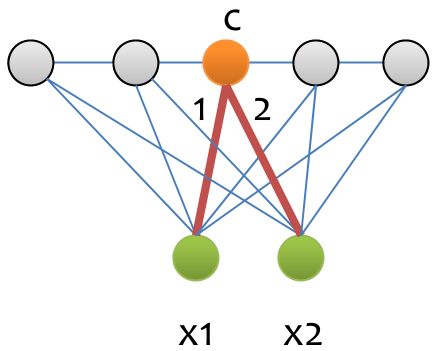
- Mustervektor: x1 = 3 und x2 = 1, Gewinner sei c mit $w_{c1} = 1$ und $w_{c2} = 2$, eta = 0.5

- x1 und x2 spannen den Eingaberaum auf
- Zeichnen wir das Gewinnerneuron c mit seinen Wichtungen w_{c1} und w_{c2} ein

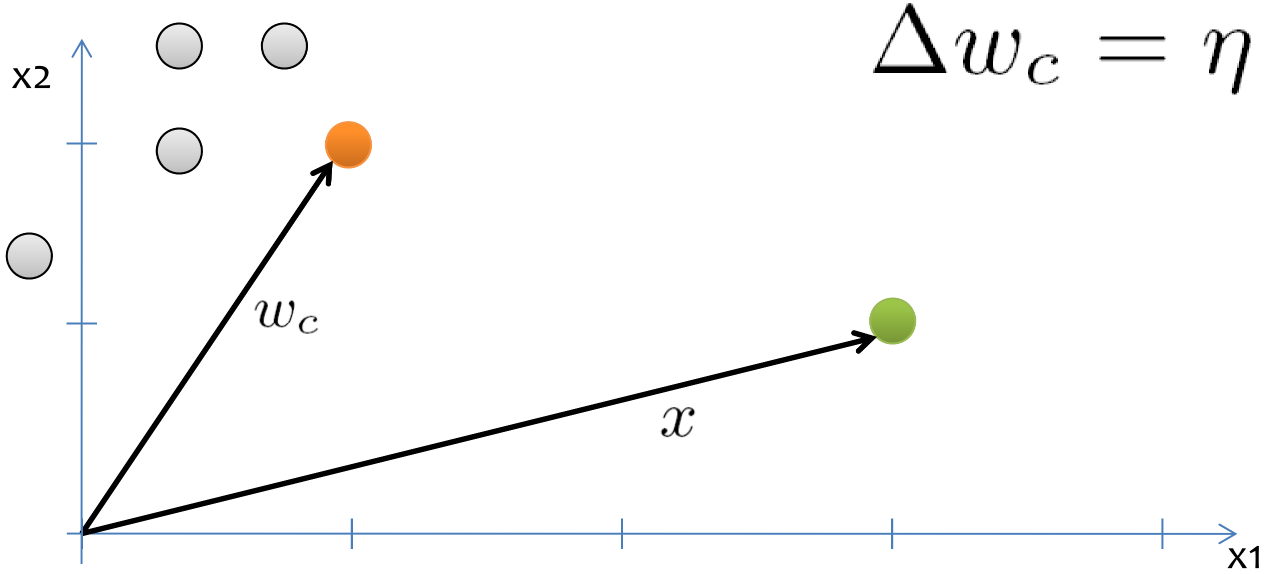
$$\Delta w_c = \eta \cdot (x - w_c)$$



Beispiel

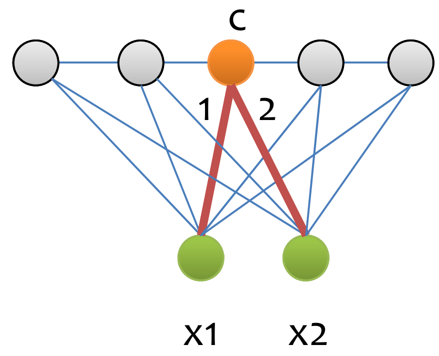


- Mustervektor: x1 = 3 und x2 = 1, Gewinner sei c mit $w_{c1} = 1$ und $w_{c2} = 2$, eta = 0.5
- x1 und x2 spannen den Eingaberaum auf
- Zeichnen wir das Gewinnerneuron c mit seinen Wichtungen w_{c1} und w_{c2} ein

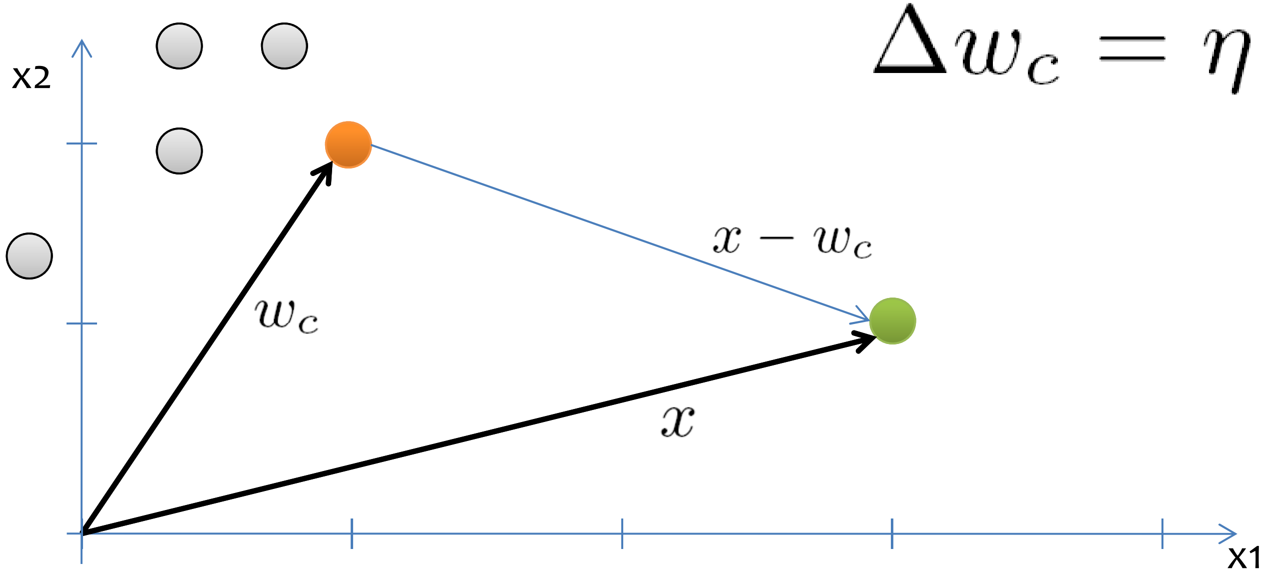


$$\Delta w_c = \eta \cdot (x - w_c)$$

Beispiel

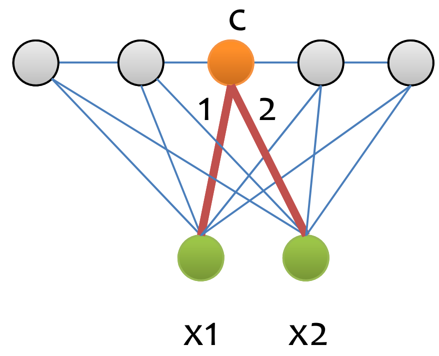


- Mustervektor: x1 = 3 und x2 = 1, Gewinner sei c mit $w_{c1} = 1$ und $w_{c2} = 2$, eta = 0.5
- x1 und x2 spannen den Eingaberaum auf
- Zeichnen wir das Gewinnerneuron c mit seinen Wichtungen w_{c1} und w_{c2} ein



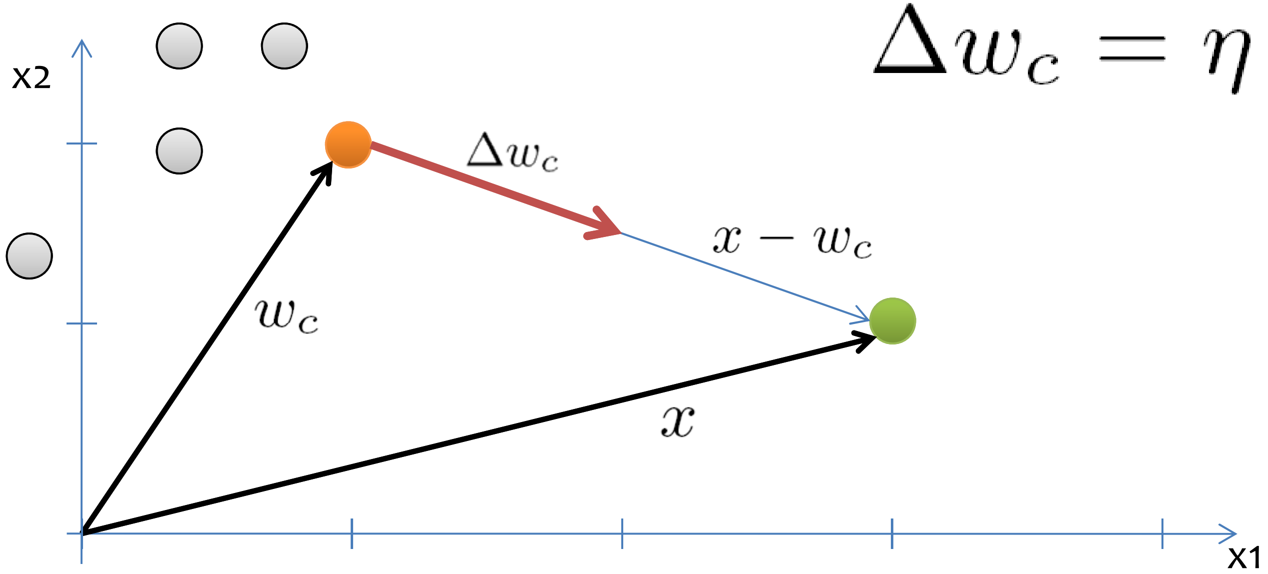
$$\Delta w_c = \eta \cdot (x - w_c)$$

Beispiel

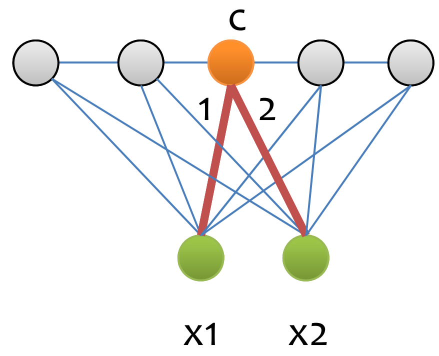


- Mustervektor: x1 = 3 und x2 = 1, Gewinner sei c mit $w_{c1} = 1$ und $w_{c2} = 2$, eta = 0.5
- x1 und x2 spannen den Eingaberaum auf
- Zeichnen wir das Gewinnerneuron c mit seinen Wichtungen w_{c1} und w_{c2} ein

$$\Delta w_c = \eta \cdot (x - w_c)$$

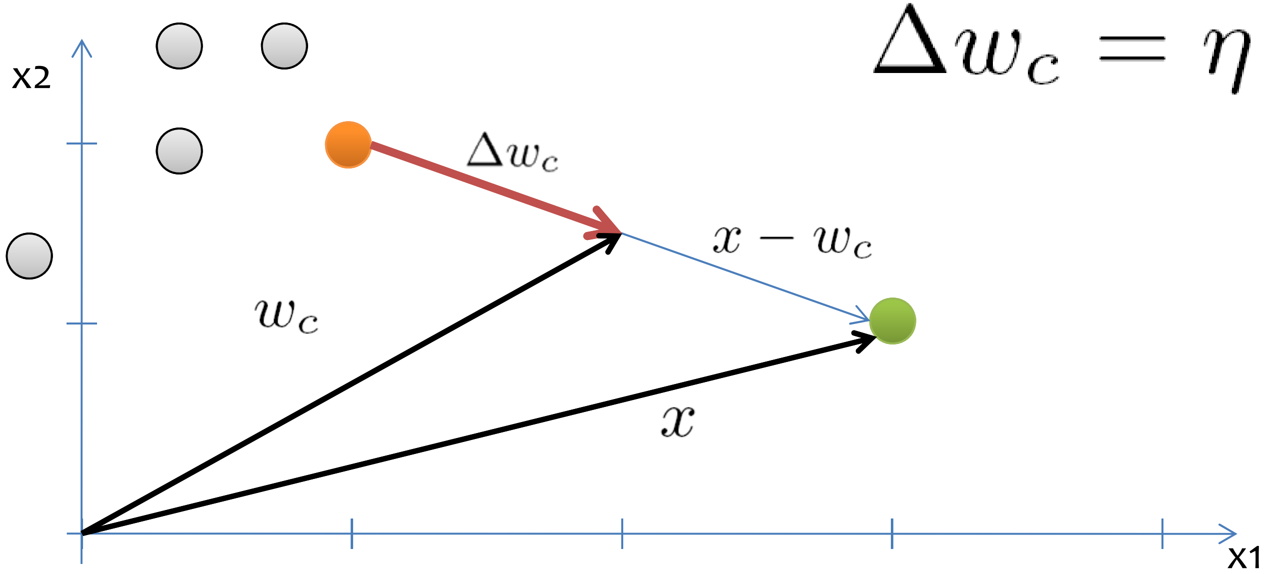


Beispiel



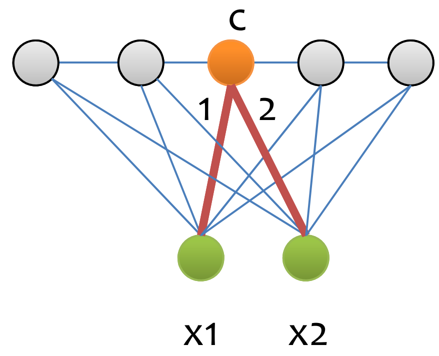
- Mustervektor: x1 = 3 und x2 = 1, Gewinner sei c mit $w_{c1} = 1$ und $w_{c2} = 2$, eta = 0.5

- x1 und x2 spannen den Eingaberaum auf
- Zeichnen wir das Gewinnerneuron c mit seinen Wichtungen w_{c1} und w_{c2} ein



$$\Delta w_c = \eta \cdot (x - w_c)$$

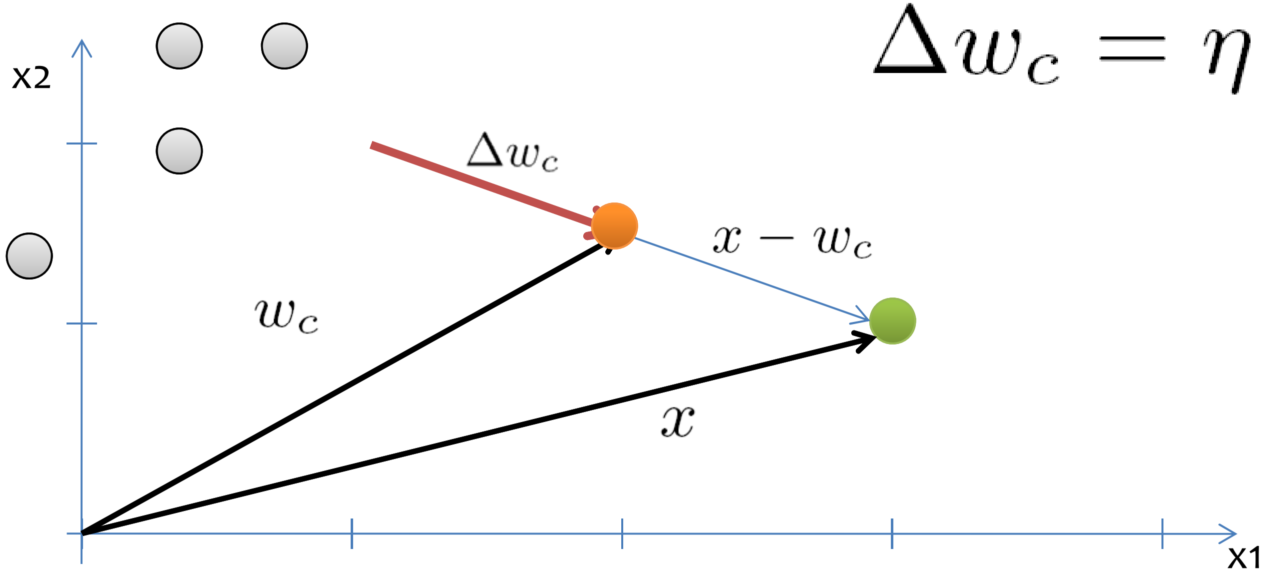
Beispiel



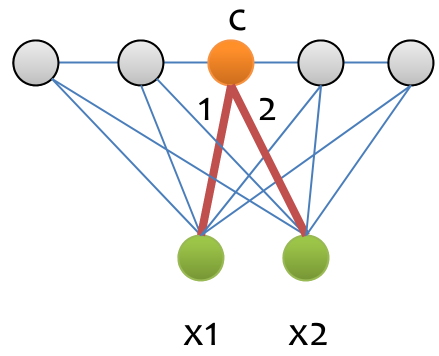
- Mustervektor: x1 = 3 und x2 = 1, Gewinner sei c mit w_{c1} = 1 und w_{c2} = 2, eta = 0.5

- x1 und x2 spannen den Eingaberaum auf
- Zeichnen wir das Gewinnerneuron c mit seinen Wichtungen w_{c1} und w_{c2} ein

$$\Delta w_c = \eta \cdot (x - w_c)$$



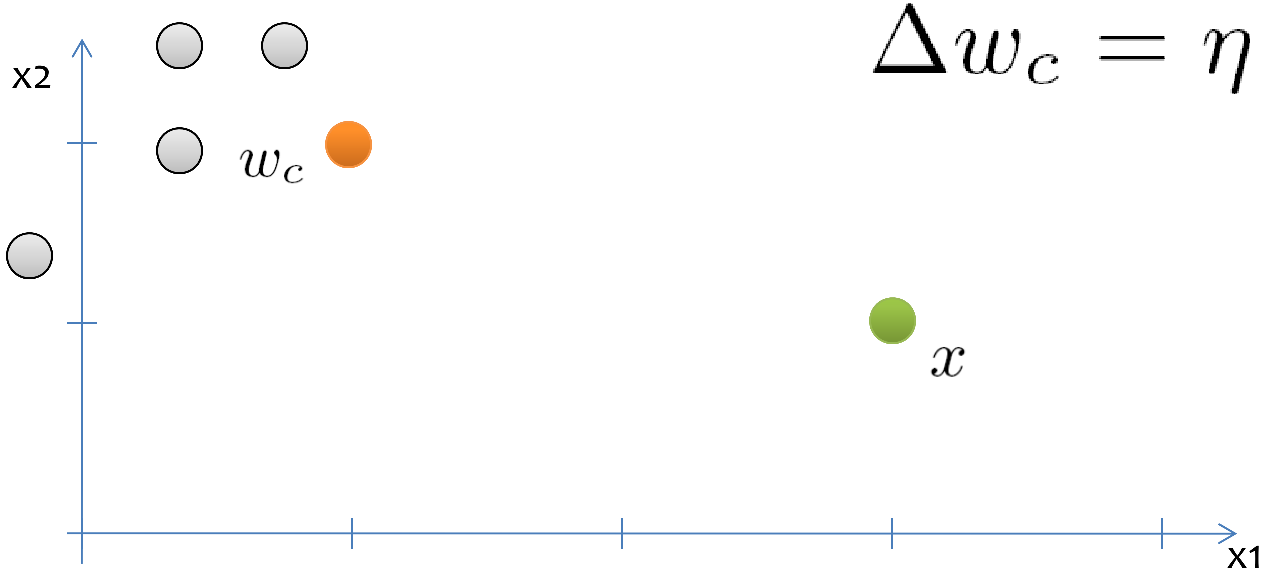
Beispiel



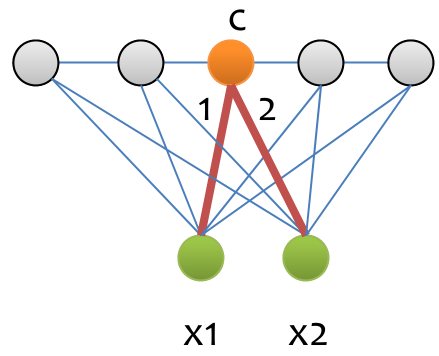
- Mustervektor: x1 = 3 und x2 = 1, Gewinner sei c mit $w_{c1} = 1$ und $w_{c2} = 2$, eta = 0.5

- x1 und x2 spannen den Eingaberaum auf
- Zeichnen wir das Gewinnerneuron c mit seinen Wichtungen w_{c1} und w_{c2} ein

$$\Delta w_c = \eta \cdot (x - w_c)$$

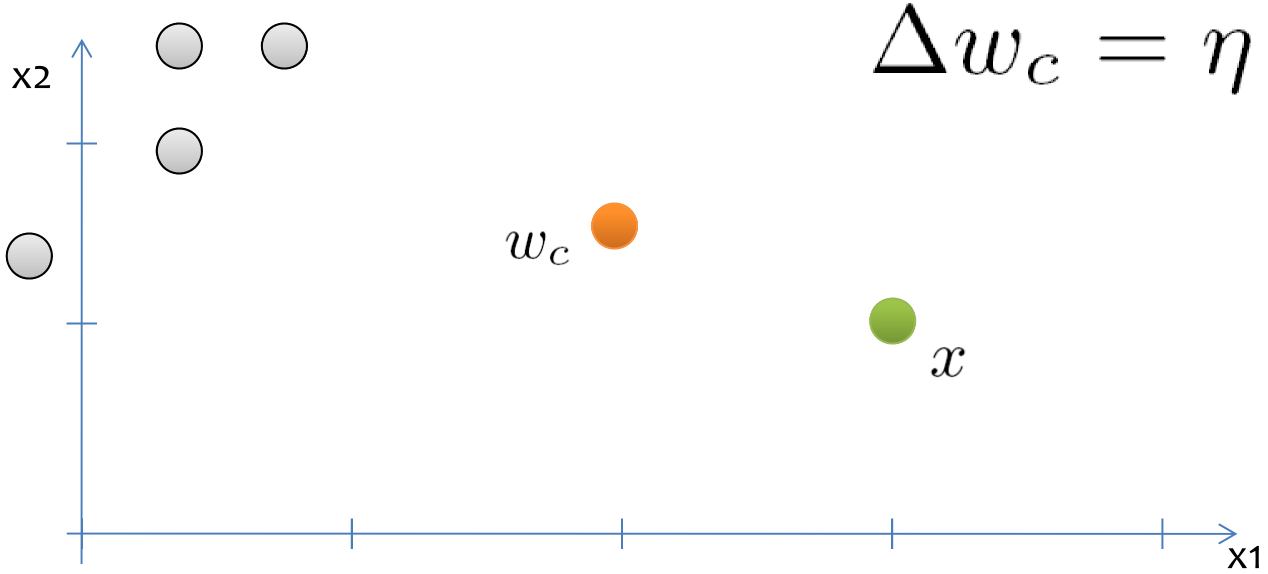


Beispiel

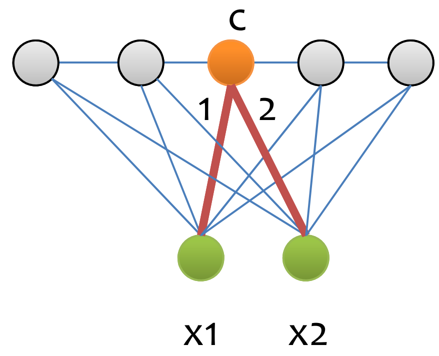


- Mustervektor: x1 = 3 und x2 = 1, Gewinner sei c mit $w_{c1} = 1$ und $w_{c2} = 2$, eta = 0.5
- x1 und x2 spannen den Eingaberaum auf
- Zeichnen wir das Gewinnerneuron c mit seinen Wichtungen w_{c1} und w_{c2} ein

$$\Delta w_c = \eta \cdot (x - w_c)$$



Beispiel

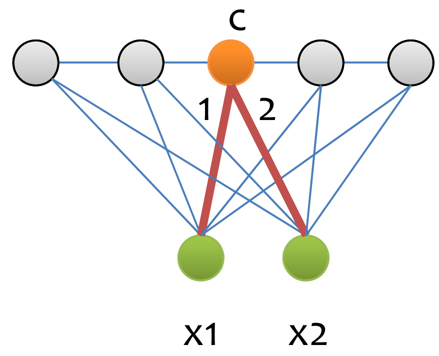


- Mustervektor: x1 = 3 und x2 = 1, Gewinner sei c mit $w_{c1} = 1$ und $w_{c2} = 2$, eta = 0.5

- Berechnen Sie die neuen Wichtungen des Neurons c

$$\Delta w_c = \eta \cdot (x - w_c)$$

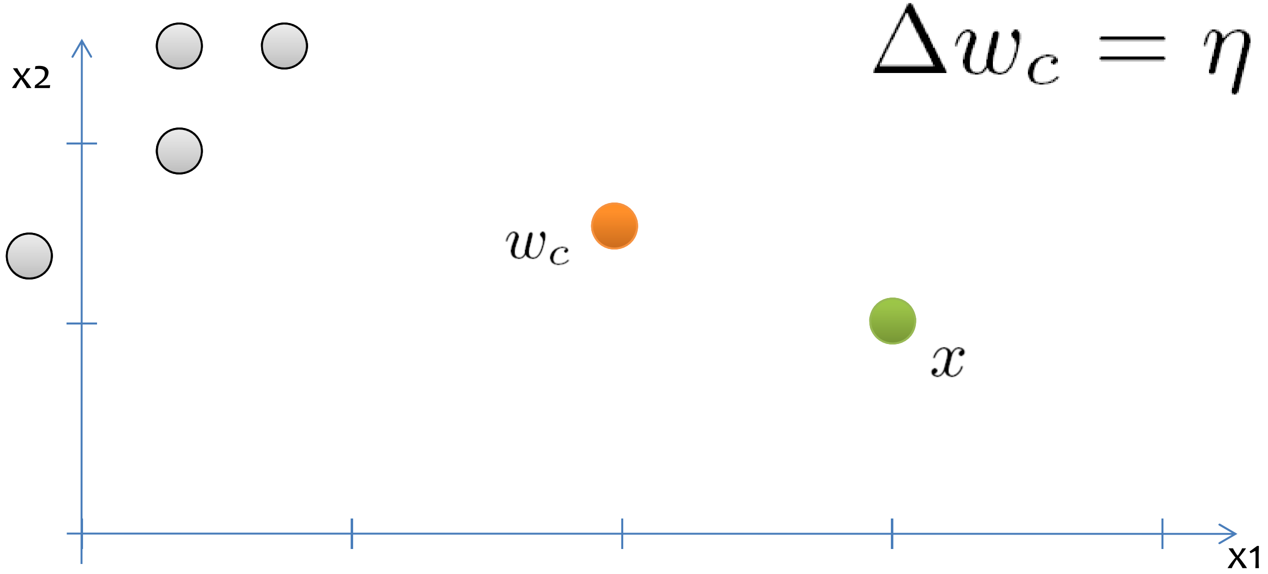
Beispiel



- Mustervektor: x1 = 3 und x2 = 1, Gewinner sei c mit $w_{c1} = 1$ und $w_{c2} = 2$, eta = 0.5

$$w_c = \begin{pmatrix} 1 \\ 2 \end{pmatrix} + 0.5 \cdot \left(\begin{pmatrix} 3 \\ 1 \end{pmatrix} - \begin{pmatrix} 1 \\ 2 \end{pmatrix} \right) = \begin{pmatrix} 2 \\ 1.5 \end{pmatrix}$$

$$\Delta w_c = \eta \cdot (x - w_c)$$



Training einer SOM

- Initialisierung der Wichtungen mit Zufallszahlen
- Eingabemuster wiederholt anlegen
  - Gewinnerneuron bestimmen
  - Gewinnerneuron lernt
  - Nachbarneuronen des Gewinnerneurons lernen
- Ende nach Zyklenanzahl oder ausreichender Abbildungsgüte

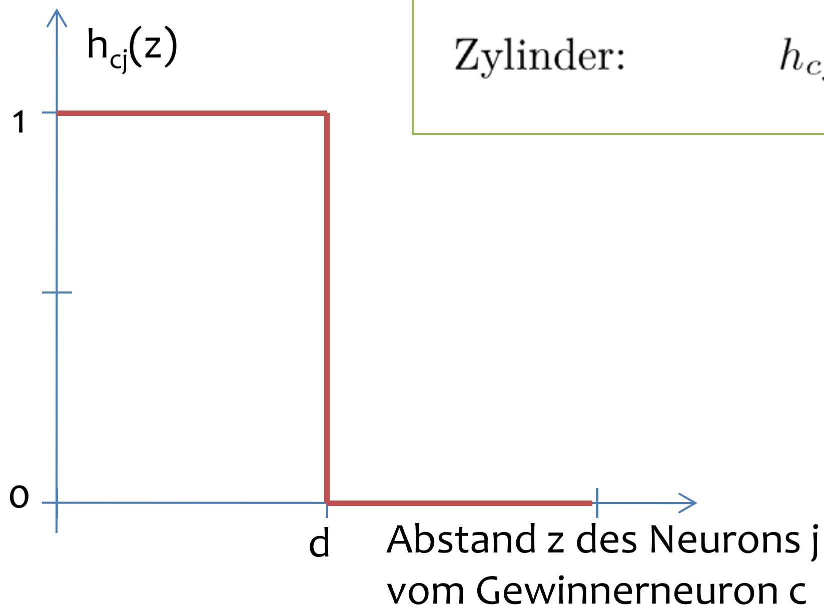
Nachbarneuronen lernen mit

- Die Nachbarn des Gewinnerneurons lernen ebenfalls, und zwar umso stärker, je näher sie sich am Gewinnerneuron befinden:

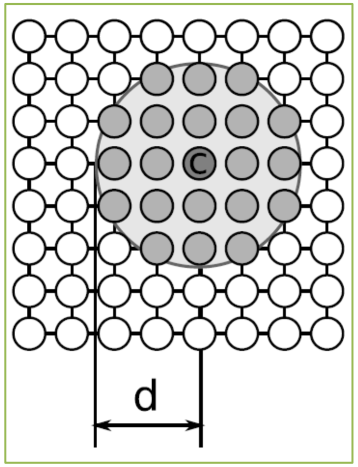
$$\triangle w_j = \eta \cdot h_{cj} \cdot (x - w_j)$$

- <u>Nachbarschaftsfunktion</u> h_{cj} : Wie stark lernt das Neuron j mit, wenn Neuron c gewinnt.
- Oft als <u>Funktion des Abstandes</u> z der Orte der beiden Neuronen j und c

- **Beispiel:**



$$\text{Zylinder:} \qquad h_{cj}(z) = \begin{cases} 1 \text{ falls } z < d \\ 0 \text{ sonst} \end{cases}$$



Nachbarschaftsfunktionen

- Wie stark lernt ein Neuron im Abstand z vom Gewinnerneuron.

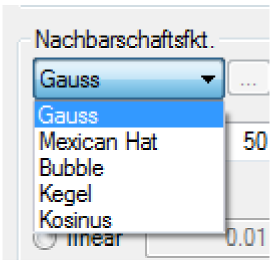
$$\text{Gaußfunktion:} \quad h_{cj}(z) = e^{-(z/d)^2}$$

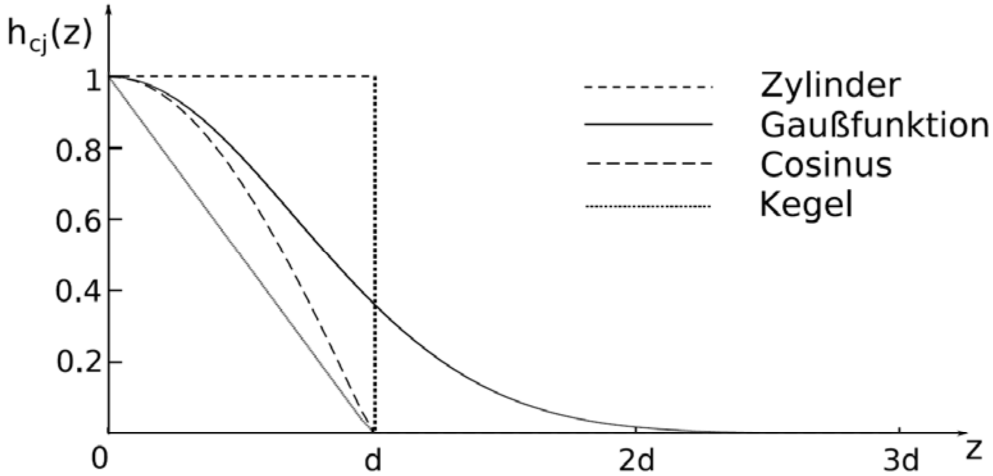
$$\text{Zylinder:} \quad h_{cj}(z) = \begin{cases} 1 \text{ falls } z < d \\ 0 \text{ sonst} \end{cases}$$

$$\text{Kegel:} \quad h_{cj}(z) = \begin{cases} 1 - z/d \text{ falls } z < d \\ 0 \text{ sonst} \end{cases}$$

$$\text{Cosinus:} \quad h_{cj}(z) = \begin{cases} \cos(\pi z/2d) \text{ falls } z < d \\ 0 \text{ sonst} \end{cases}$$

In Sombrero:





Nachbarschaftsfunktionen - Lernradius

Lernradius d

Hierbei wird mit einem großen Lernradius d_0, der oft die ganze Karte erfasst, begonnen, um die Grobstruktur der Karte zu bilden.

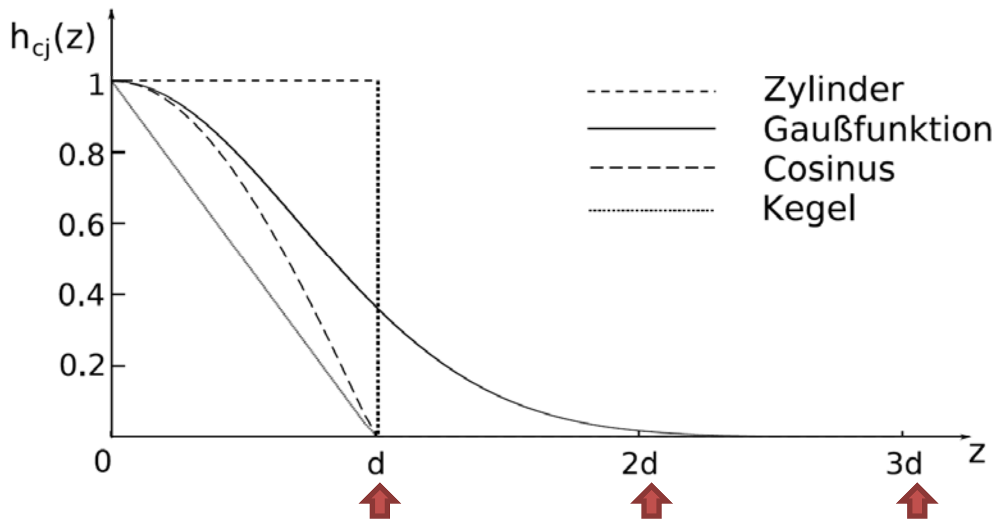
Im Lernverlauf wird d langsam auf 0 abgesenkt, so dass am Ende nur noch das Gewinnerneuron lernt und sich Details herausbilden.

$$\text{Gaußfunktion:} \quad h_{cj}(z) = e^{-(z/d)^2}$$

$$\text{Zylinder:} \quad h_{cj}(z) = \begin{cases} 1 \text{ falls } z < d \\ 0 \text{ sonst} \end{cases}$$

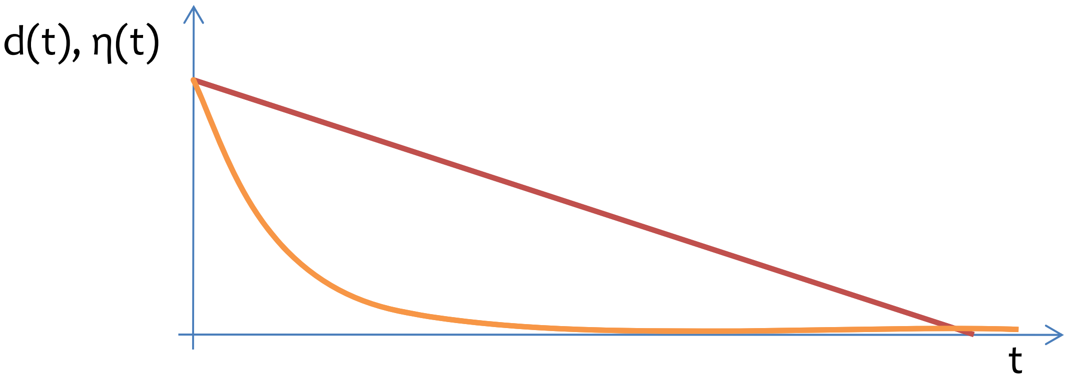
$$\text{Kegel:} \quad h_{cj}(z) = \begin{cases} 1 - z/d \text{ falls } z < d \\ 0 \text{ sonst} \end{cases}$$

$$\text{Cosinus:} \quad h_{cj}(z) = \begin{cases} \cos(\pi z/2d) \text{ falls } z < d \\ 0 \text{ sonst} \end{cases}$$



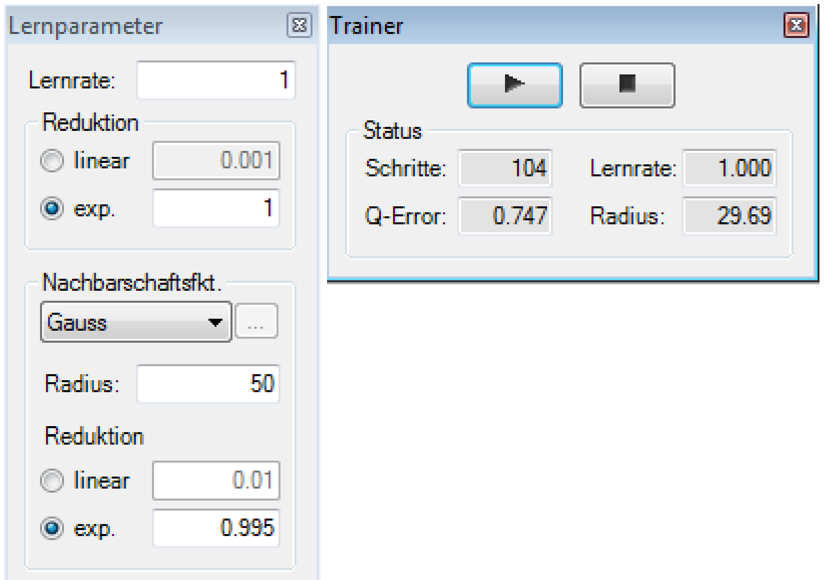
Grobstruktur - Feinstruktur

- Beim Lernen:
  - Absenken des Lernradius d
  - Absenken der Lernrate η



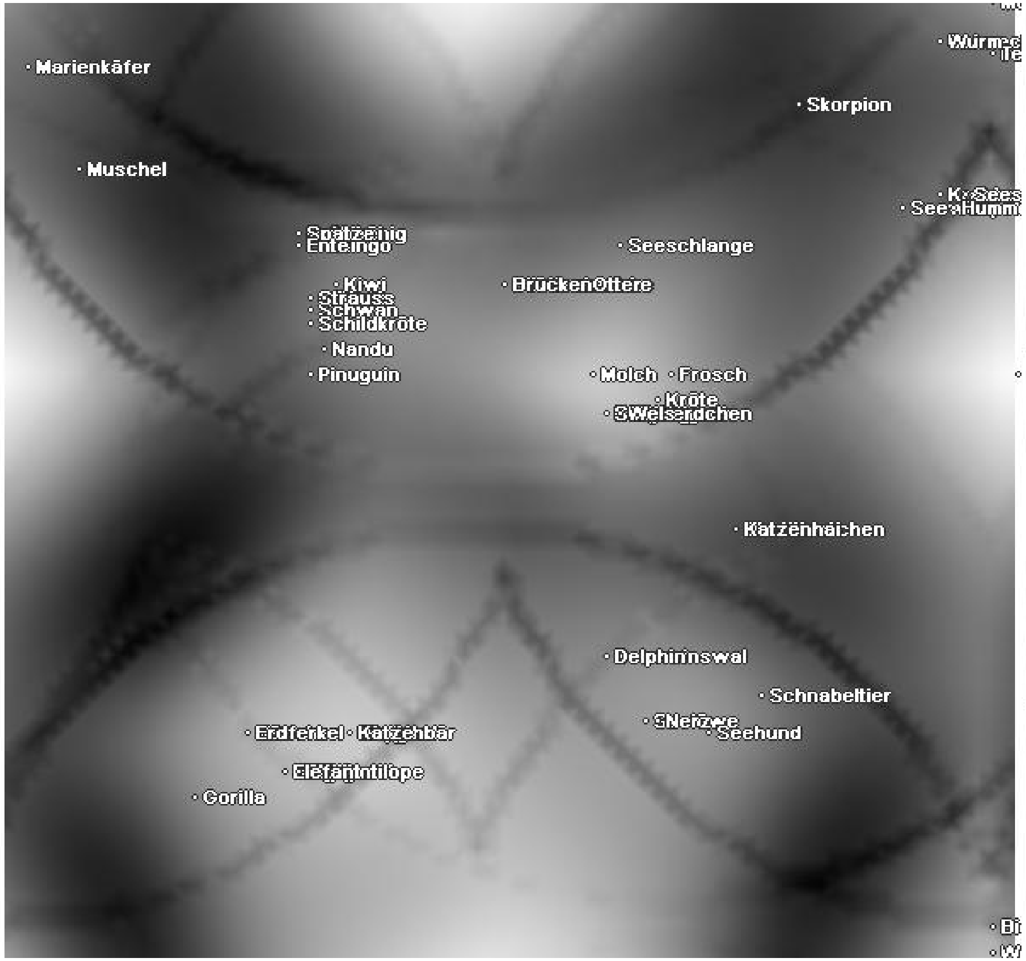
- Linear
  $$\eta(t) = \eta_0 - t \cdot \eta_1 \text{ und } d(t) = d_0 - t \cdot d_1$$
- Exponentiell
  $$\eta(t) = \eta_0 \cdot \eta_1^t \text{ und } d(t) = d_0 \cdot d_1^t$$
  $$\text{mit } 0 < \eta_1, d_1 < 1$$

- Beispiele
  $$d(t) = 50 - 0.001 \cdot t$$
  $$\eta(t) = 1 - 0.001 \cdot t$$
  $$d(t) = 50 \cdot 0.995^t$$
  $$\eta(t) = 1 \cdot 1^t$$
- In Sombrero:

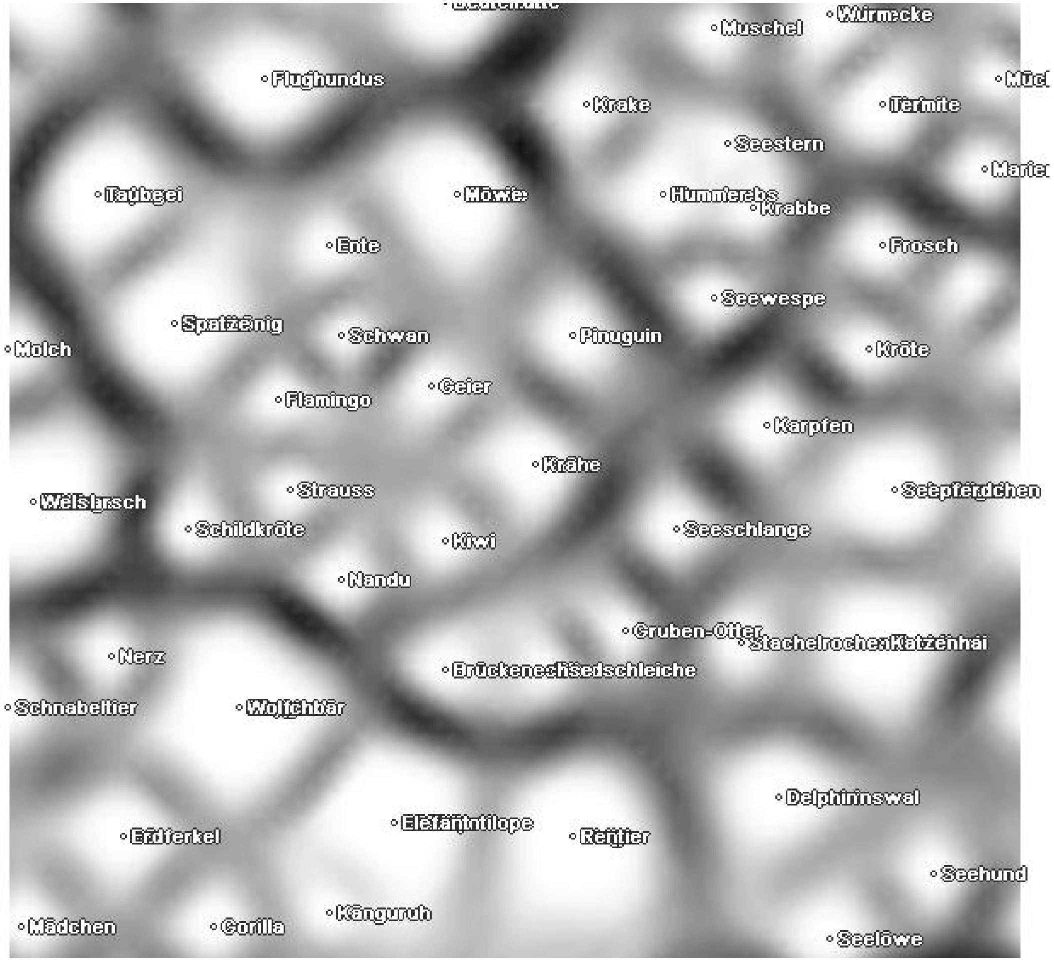


Grobstruktur - Feinstruktur

t = 20



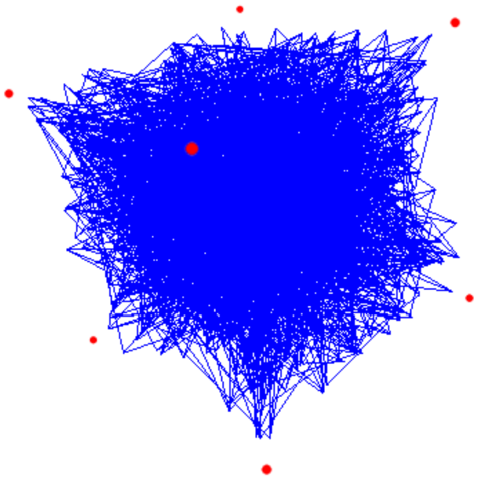
t = 2000

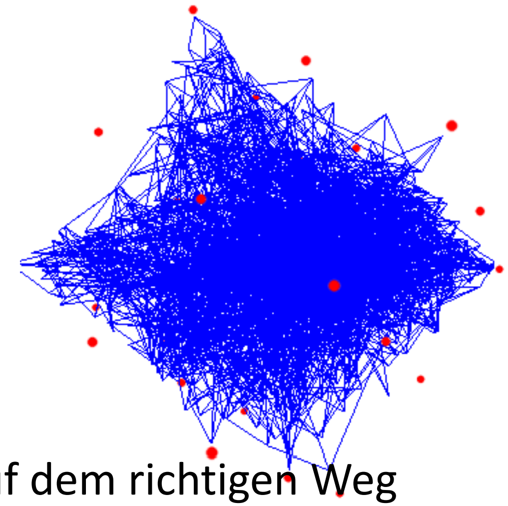


Folgen zu schnellen Absenkens

- Initialzustand

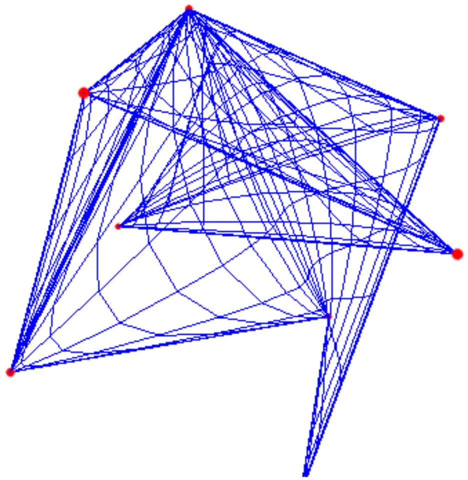
Erfroren beim Start

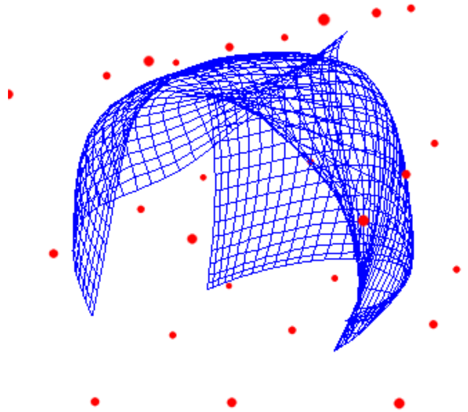




- Topologiedefekte, Grobstruktur unvollendet

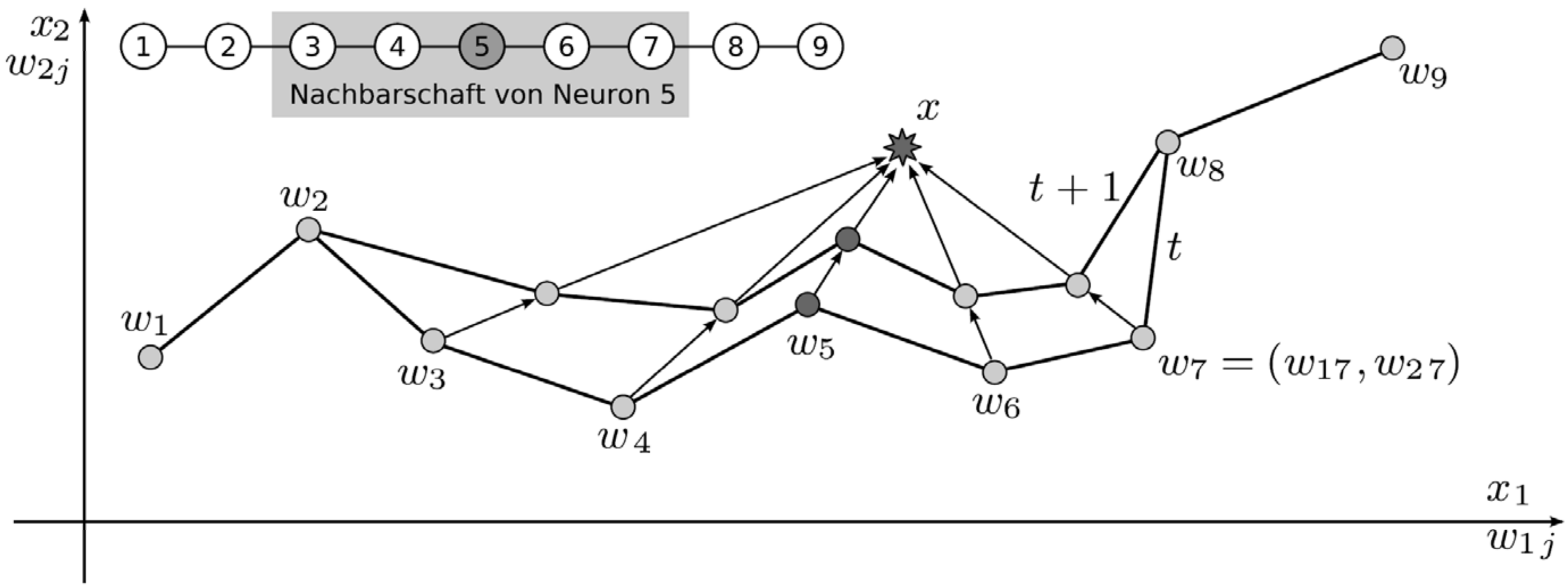
auf dem richtigen Weg





Lernen: n=2, m=1

- zwei Eingabeneuronen (*n = 2), also zweidimensionaler Eingaberaum*
- eindimensionale Karte (*m = 1, Kette) aus neun Kartenneuronen.*



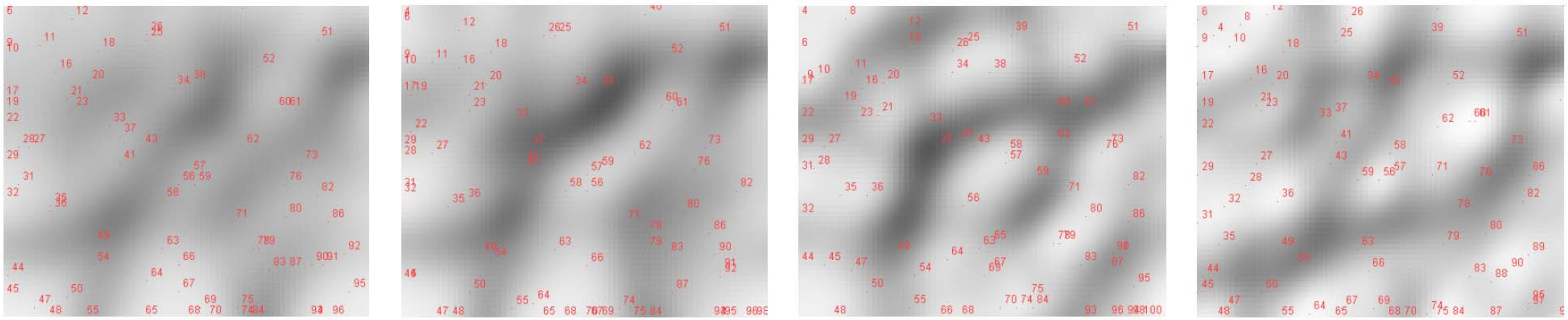
- Beim Anlegen des Mustervektors *x wird Neuron 5 als Sieger ermittelt. Zur Nachbarschaft* von Neuron 5 gehören die Neuronen 3, 4, 6 und 7. Somit wird die Wichtung *w5 stark, w4* und *w6 mittel sowie w3 und w7 schwach in Richtung x verändert (Pfeile). Obwohl* der Wichtungsvektor *w8 sehr nah an x liegt, wird er nicht verändert, da Neuron 8 in der* Kette kein Nachbar des Gewinnerneurons ist.

Lernmetapher Kupferschmiede

- Ein Kupferschmied hämmert auf einer Metallplatte. Hierbei bestimmt
  - *der Mustervektor x, wohin*
  - *die Lernrate η, wie stark*
  - *der Lernradius d, mit welcher Hammergröße und*
  - *die Nachbarschaftsfunktion hcj, mit welcher Hammerform geschlagen wird.*

- Quadratisches Gitter 70×60, 102 Mustervektoren trainiert, U-Matrix, SOMARFF*

Lernradius=10
h_{cj} =Gauß



η =0.1 η =0.3 η =0.7 η =0.9

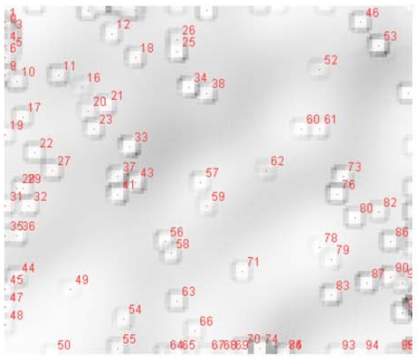
*SOMARFF: Diplomarbeit Sven Schröder, FH Brandenburg

Lernmetapher Kupferschmiede

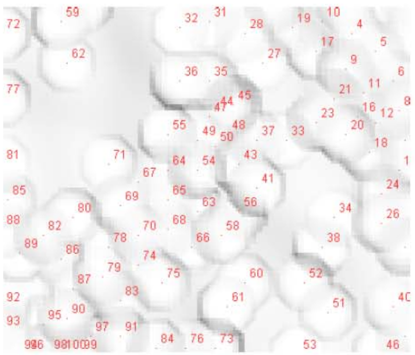
- Ein Kupferschmied hämmert auf einer Metallplatte. Hierbei bestimmt
  - *der Mustervektor x, wohin*
  - *die Lernrate η, wie stark*
  - *der Lernradius d, mit welcher Hammergröße und*
  - *die Nachbarschaftsfunktion hcj, mit welcher Hammerform geschlagen wird.*

- Quadratisches Gitter 70×60, 102 Mustervektoren trainiert, U-Matrix, SOMARFF

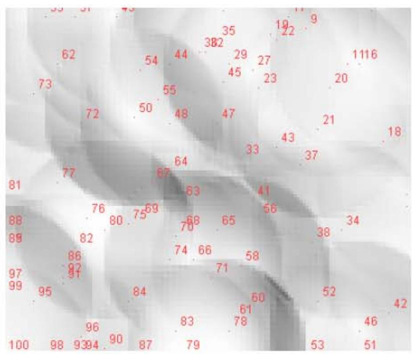
Lernrate=1.0
h_{cj}=Kegel



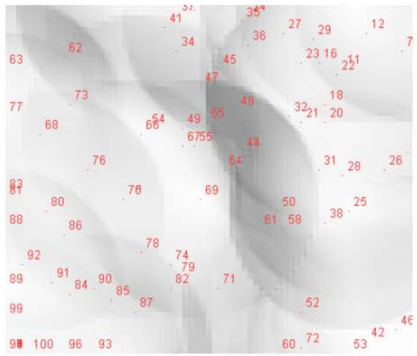
d=2



d=5



d=15



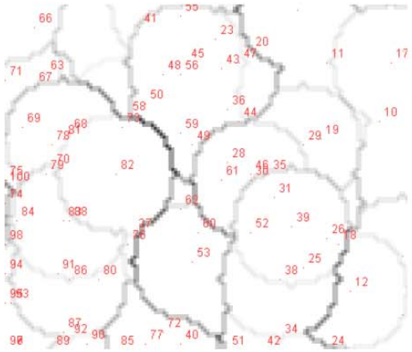
d=20

Lernmetapher Kupferschmiede

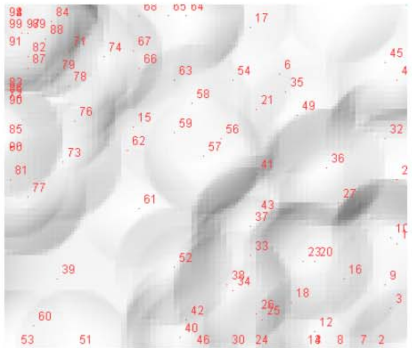
- Ein Kupferschmied hämmert auf einer Metallplatte. Hierbei bestimmt
    - *der Mustervektor x, wohin*
    - *die Lernrate η, wie stark*
    - *der Lernradius d, mit welcher Hammergröße und*
    - *die Nachbarschaftsfunktion hcj, mit welcher Hammerform geschlagen wird.*

- Quadratisches Gitter 70×60, 102 Mustervektoren trainiert, U-Matrix, SOMARFF

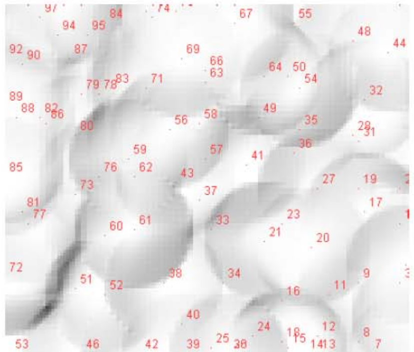
Lernrate=1.0
Radius=10



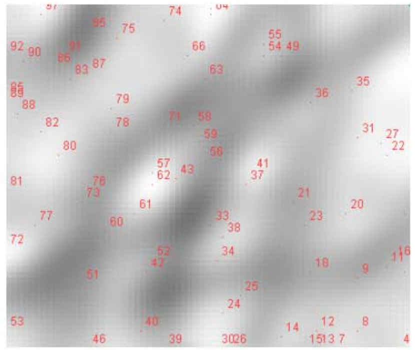
Zylinder



Kegel



Cosinus



Gaußfunktion

Verlauf des Lernens

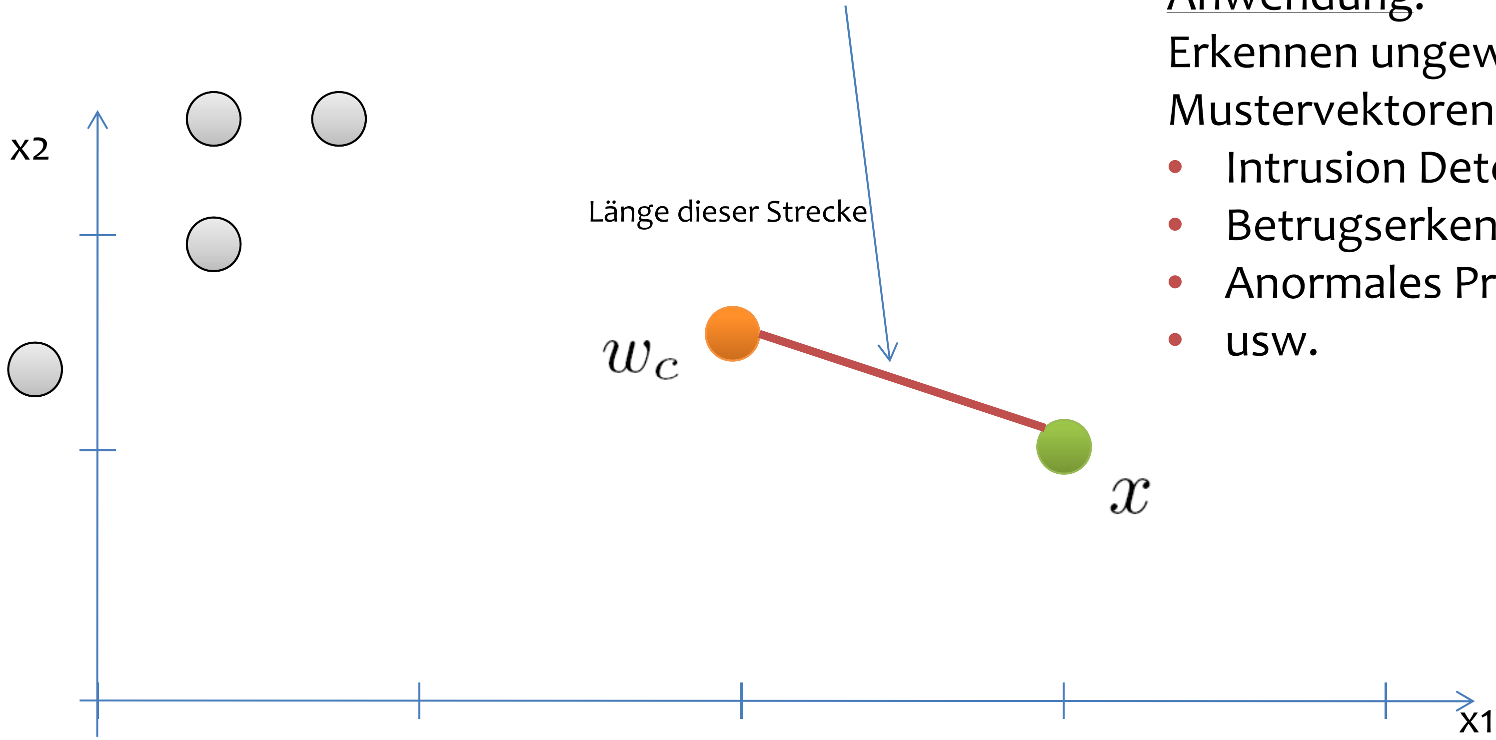
- Durch die fortlaufende Präsentation von Trainingsdatensätzen ordnen sich die Wichtungsvektoren und damit die Ausgabeneuronen bestimmten Teilräumen im Eingaberaum zu – *sie wandern im Eingaberaum umher.*
- Teilräume mit vielen Trainingsmustern ziehen hierbei eine größere Menge von Kartenneuronen an als dünn besetzte Teilräume.
- Durch das gleichzeitige Lernen von benachbarten Neuronen finden sich diese auch in benachbarten Gebieten des Eingaberaumes wieder.
- Tendenz der Karte, sich wie ein elastisches Netz an die Dichteverteilung der Trainingsdaten anzupassen.

- Die Kartenneuronen versuchen somit,
  - die Verteilung der Trainingsdaten im Eingaberaum möglichst gut anzunähern (d. h. das gleiche Gebiet/Teilraum abzudecken) und dabei
  - den Abstand zu ihren Nachbarn klein zu halten (d. h. die vorgegebene Topologie zu erhalten).

Quantisierungsfehler eines Musters x

- Ob eine Karte einen Eingabevektor x gut repräsentiert, zeigt der Quantisierungsfehler des Musters:

$$||x - w_c||$$



Anwendung:
Erkennen ungewöhnlicher Mustervektoren für bspw.

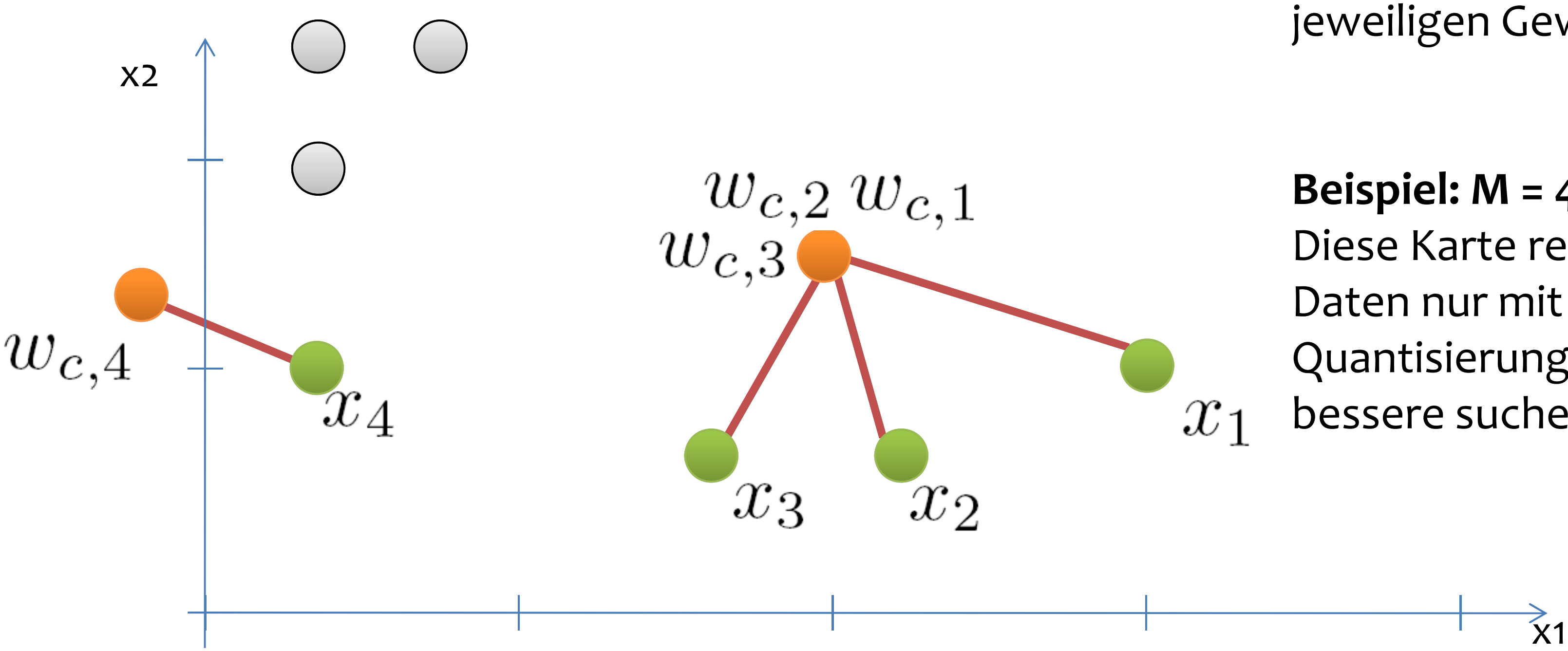
- Intrusion Detektion,
- Betrugserkennung,
- Anormales Prozessverhalten
- usw.

Mittlerer Quantisierungsfehler einer Karte

- Der mittlere Quantisierungsfehler wird über alle Mustervektoren berechnet und als Qualitätsmaß einer SOM und Abbruchkriteriterium des Lernvorgangs verwendet:

$$e_{quant} = \frac{1}{M} \sum_{p=1}^{M} ||x_p - w_{c,p}||$$

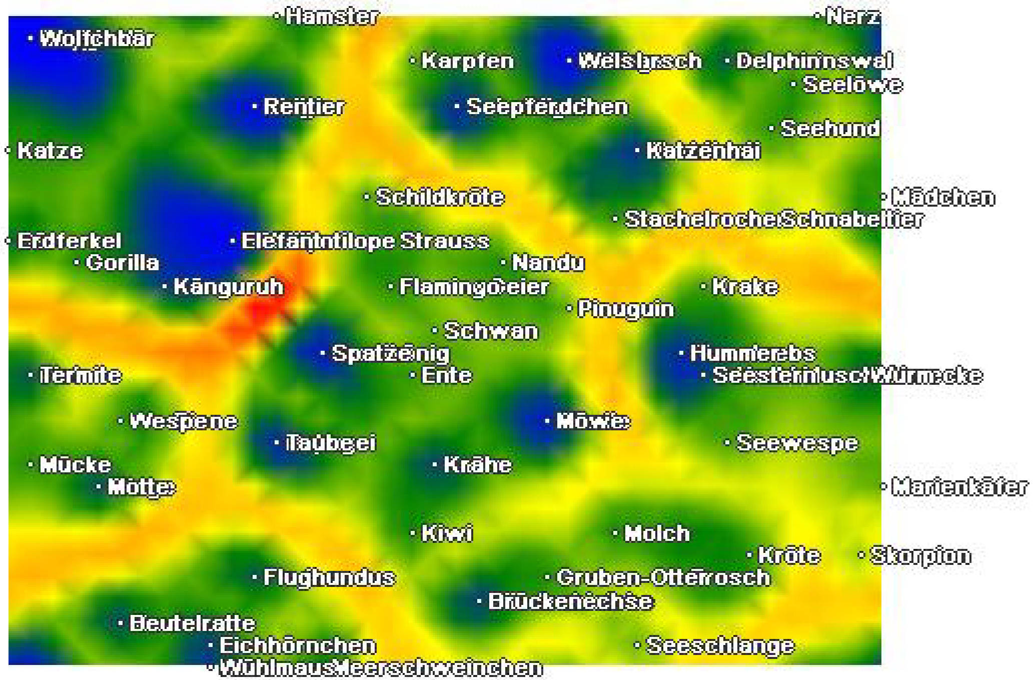
x_p ist das p-te Muster,
M die Anzahl aller Muster und
$w_{c,p}$ der Wichtungsvektor des jeweiligen Gewinnerneurons.

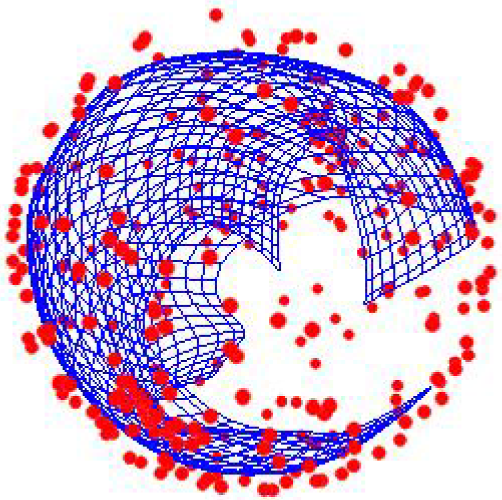


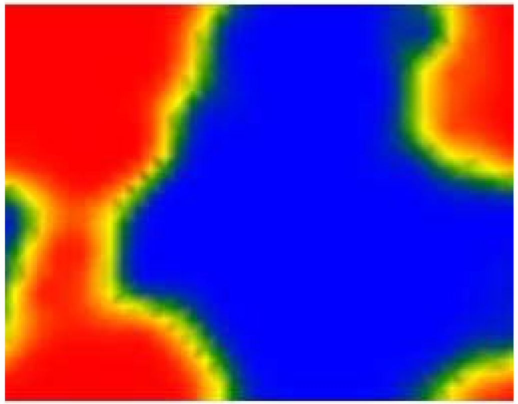
Beispiel: M = 4
Diese Karte repräsentiert die Daten nur mit einem hohen Quantisierungsfehler – bessere suchen.

Visualisierung

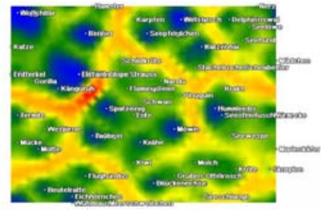
- U-Matrix
- Komponenten-Matrix
- Karte im Eingaberaum
- u.v.a.







Visualisierung U-Matrix



- U-Matrix, für unified distance matrix
- Ausgabeneuronen in ihrer topologischen Anordnung (Gitter, Kette usw.)
- Neuronen (und Neuronenzwischenräume) werden gefärbt

Welches Merkmal der Neuronen wird zur Färbung genutzt?

- Die U-Matrix färbt die Ausgabeneuronen mit der Summe der Abstände ihres Wichtungsvektors zu den Wichtungsvektoren ihrer direkten Nachbarneuronen.

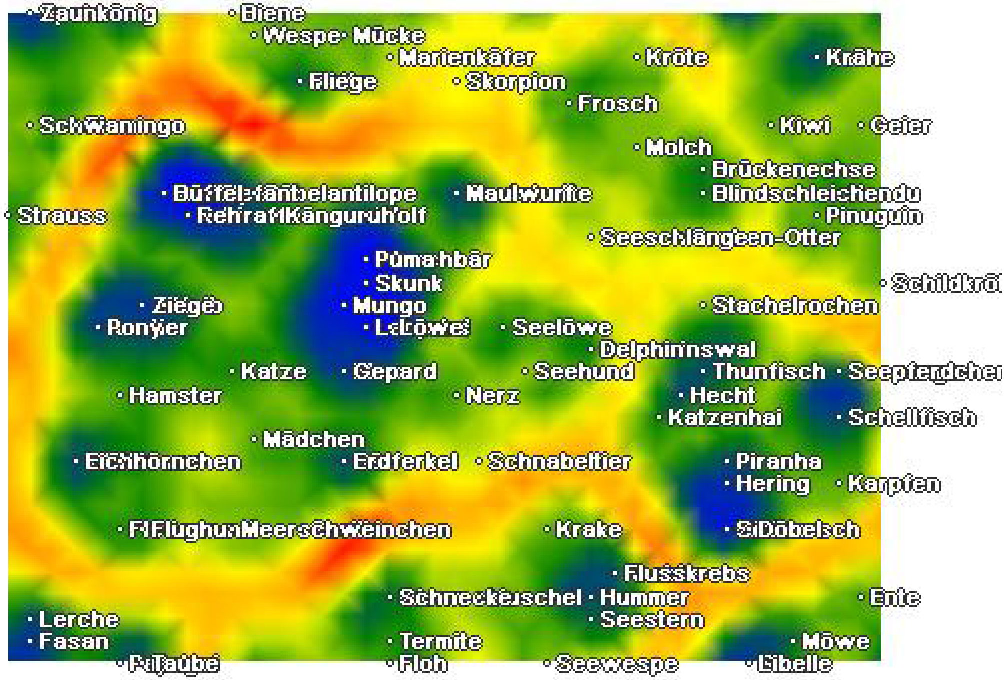
- Ein Neuron mit einem hohen U-Wert ist somit ….. ….. ….. ….. ….. ….. ….. ….. ….. ….. ?
- Die U-Matrix verdeutlicht Clustergrenzen in der Karte.

- Die **Kalibrierte Karte** kann erstellt werden, wenn die Trainingsdatensätze über einen Namen oder ein Klasse verfügen. Hierzu wird für jedes Trainingsmuster das Gewinnerneuron bestimmt und dieses mit der Klasse des Trainingsmusters benannt – beim Zoo-Beispiel die Tiernamen

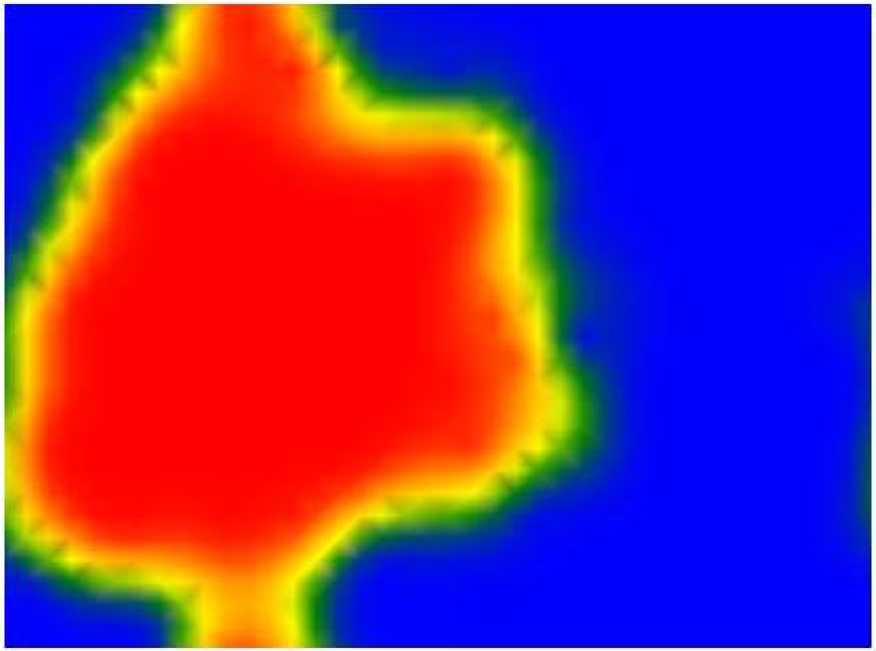
Visualisierung Komponenten-Matrix

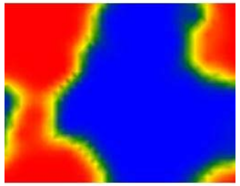
- Die Komponentenmatrix stellt nur die Wichtungen eines speziellen Eingabeneurons durch Einfärbung der Ausgabeneuronen dar.

Kalibrierte U-Matrix der Datenmenge ‚Zoo'



Komponentenmatrix des Eingabeneurons **hair**

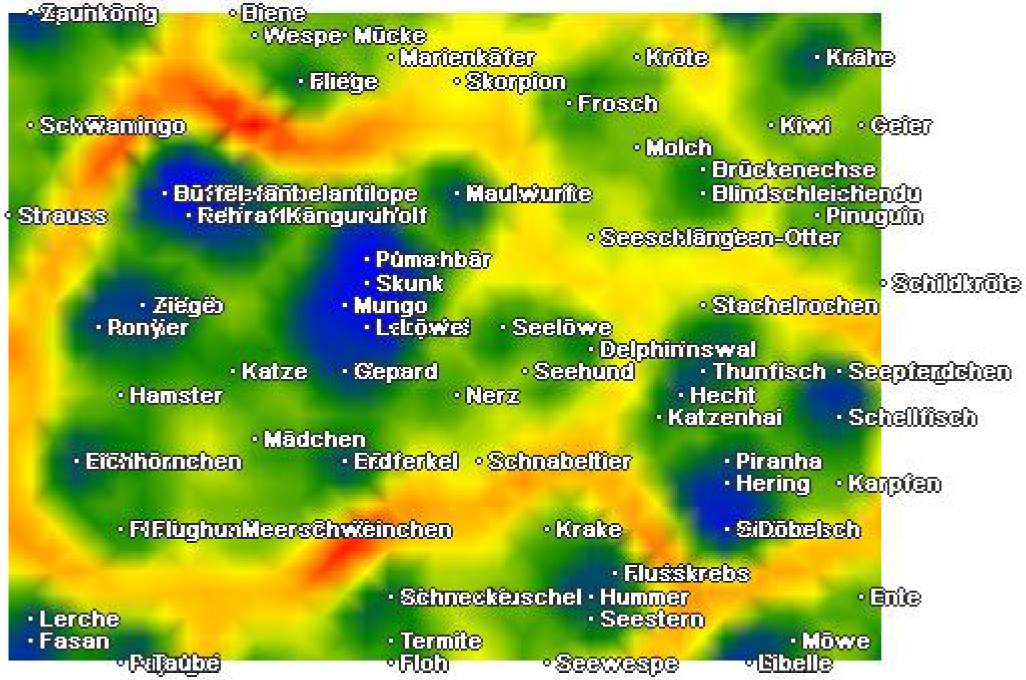




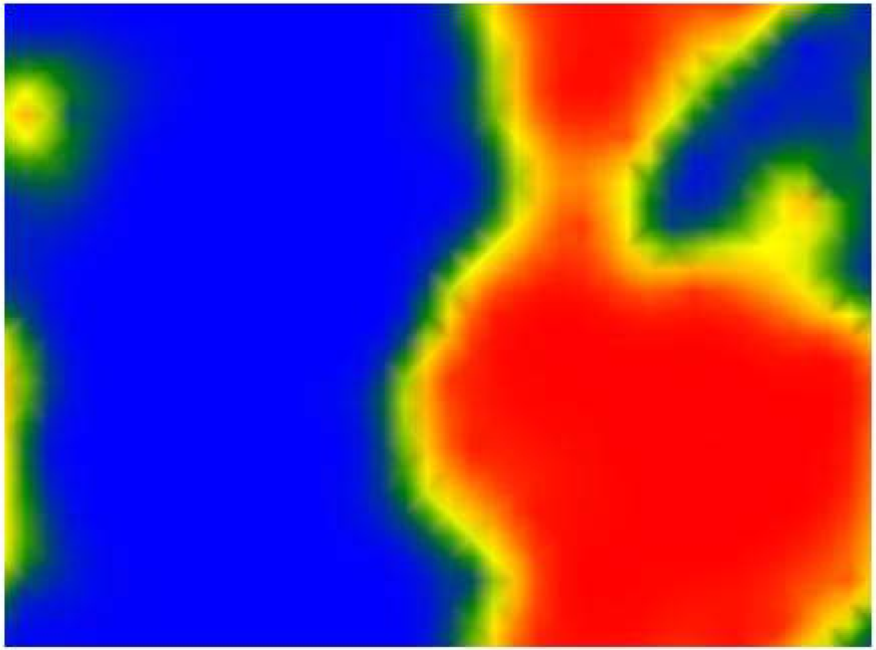
Visualisierung Komponenten-Matrix

- Die Komponentenmatrix stellt nur die Wichtungen eines speziellen Eingabeneurons durch Einfärbung der Ausgabeneuronen dar.

Kalibrierte U-Matrix der Datenmenge ‚Zoo‘



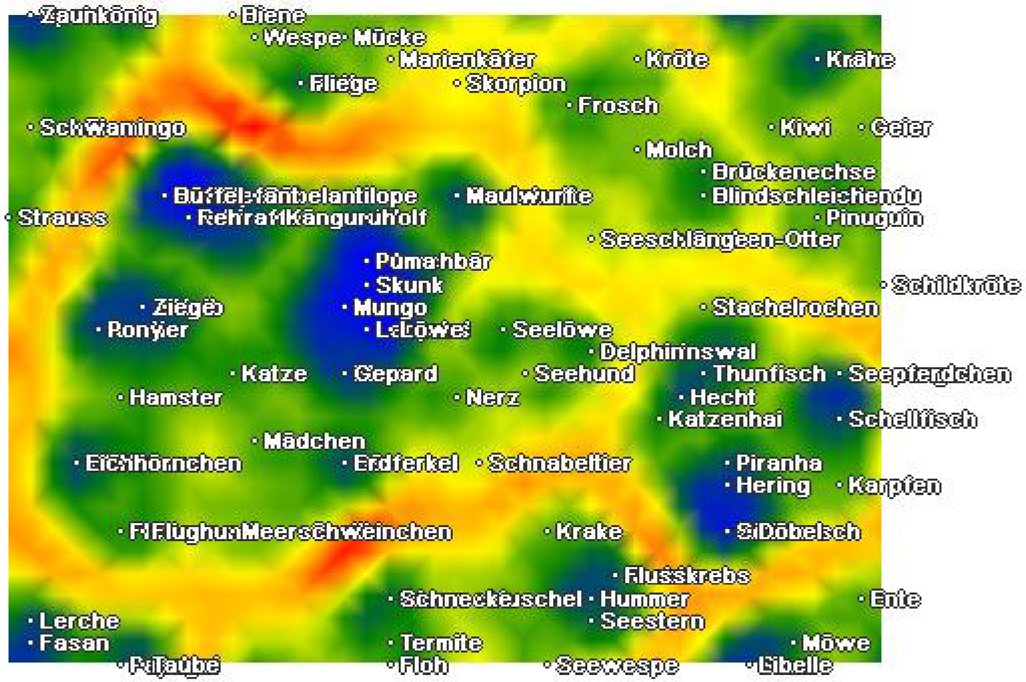
Komponentenmatrix des Eingabeneurons **aquatic**



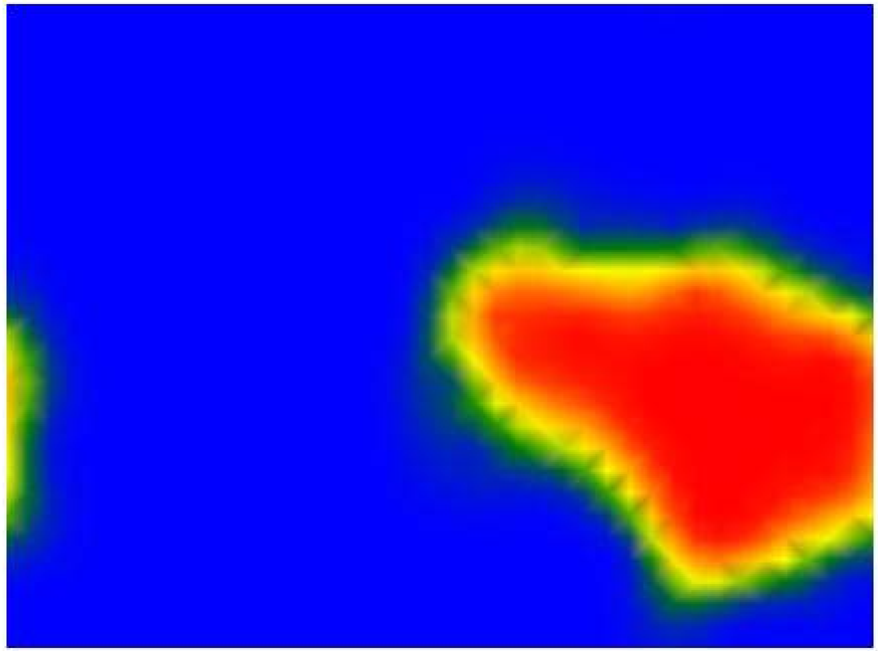
Visualisierung Komponenten-Matrix

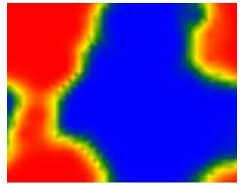
- Die Komponentenmatrix stellt nur die <u>Wichtungen eines speziellen Eingabeneurons</u> durch Einfärbung der Ausgabeneuronen dar.

Kalibrierte U-Matrix der Datenmenge ‚Zoo'



Komponentenmatrix des Eingabeneurons **fins**

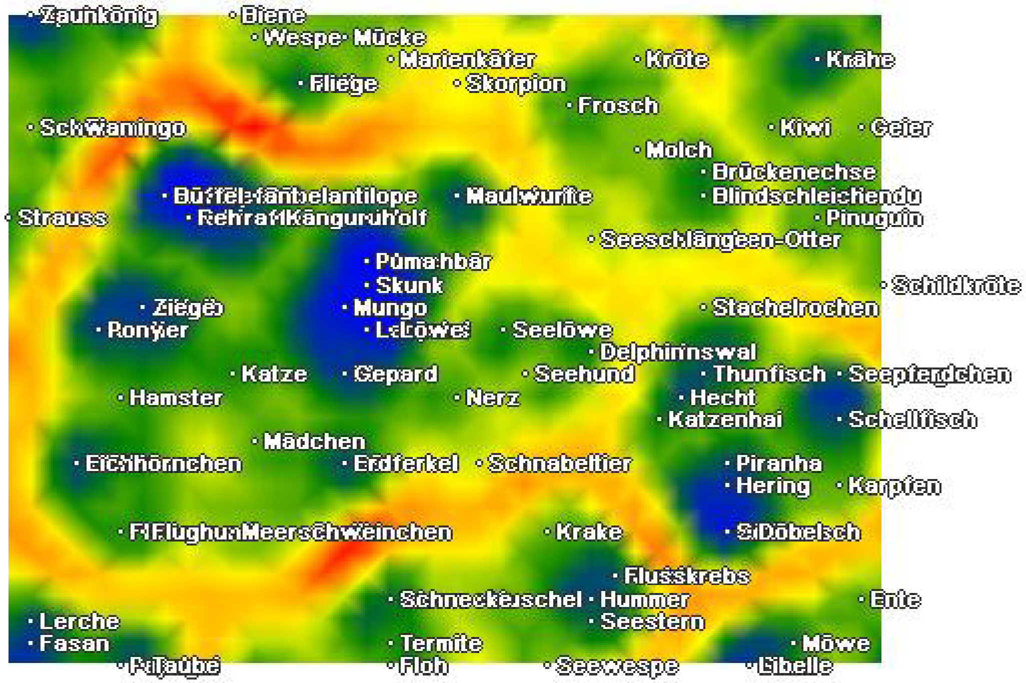




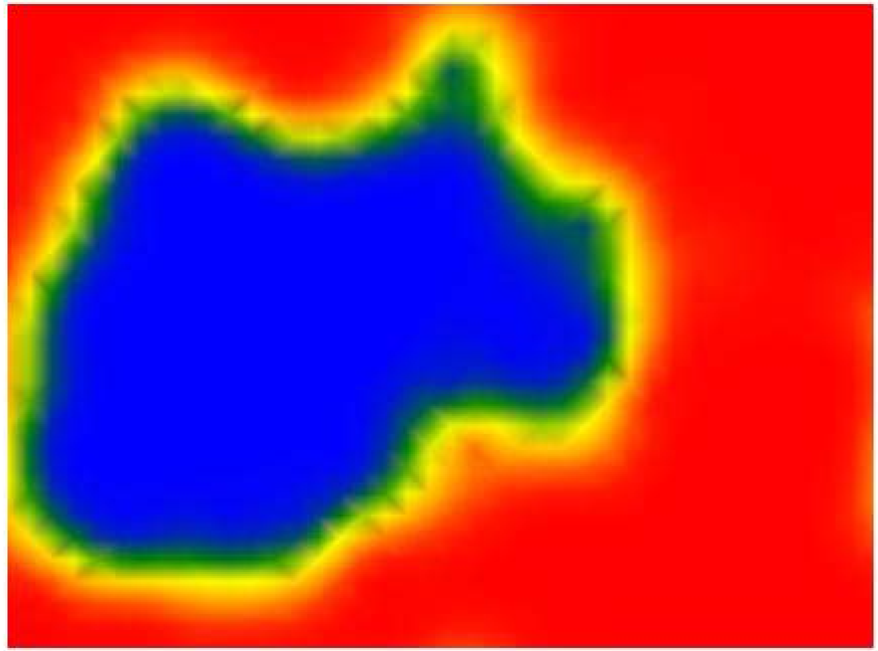
Visualisierung Komponenten-Matrix

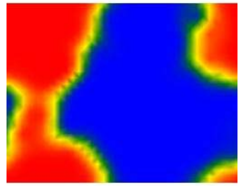
- Die Komponentenmatrix stellt nur die <u>Wichtungen eines speziellen Eingabeneurons </u>durch Einfärbung der Ausgabeneuronen dar.

Kalibrierte U-Matrix der Datenmenge ‚Zoo'



Komponentenmatrix des Eingabeneurons **eggs**

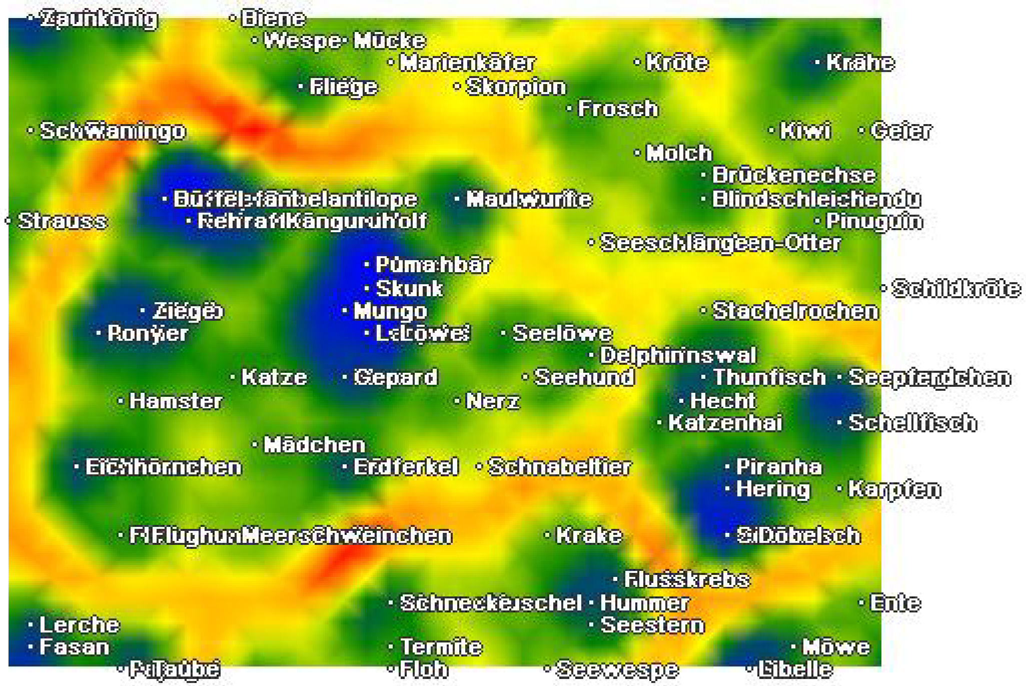




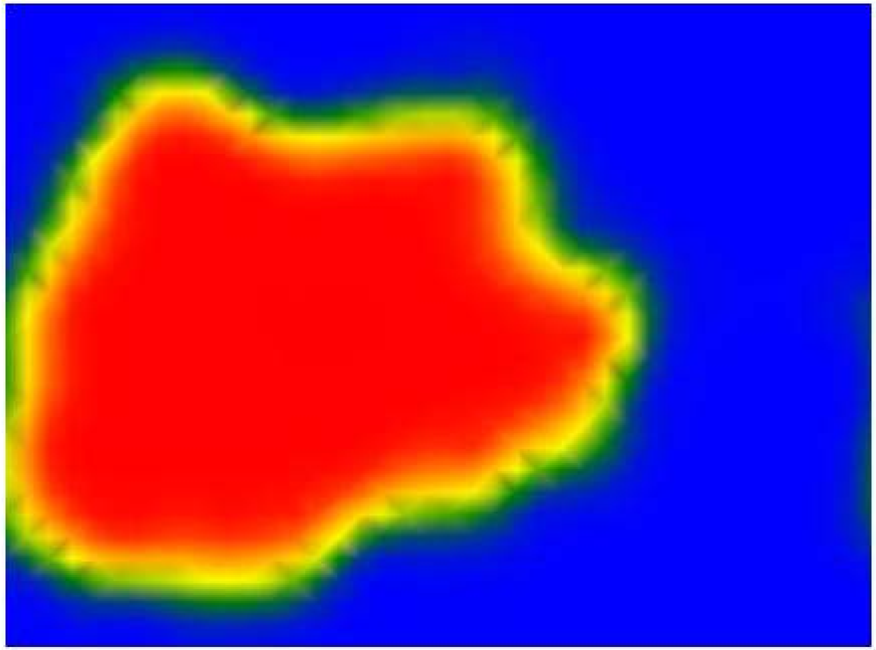
Visualisierung Komponenten-Matrix

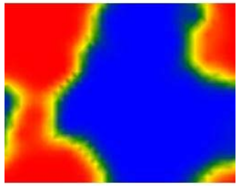
- Die Komponentenmatrix stellt nur die <u>Wichtungen eines speziellen Eingabeneurons</u> durch Einfärbung der Ausgabeneuronen dar.

Kalibrierte U-Matrix der Datenmenge ‚Zoo‘



Komponentenmatrix des Eingabeneurons **milk**

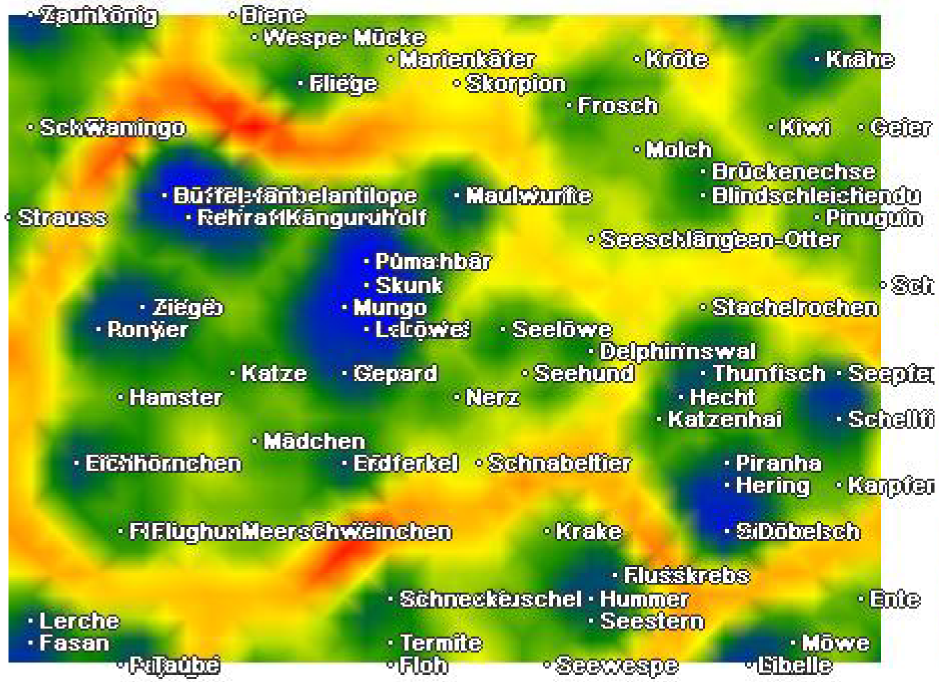




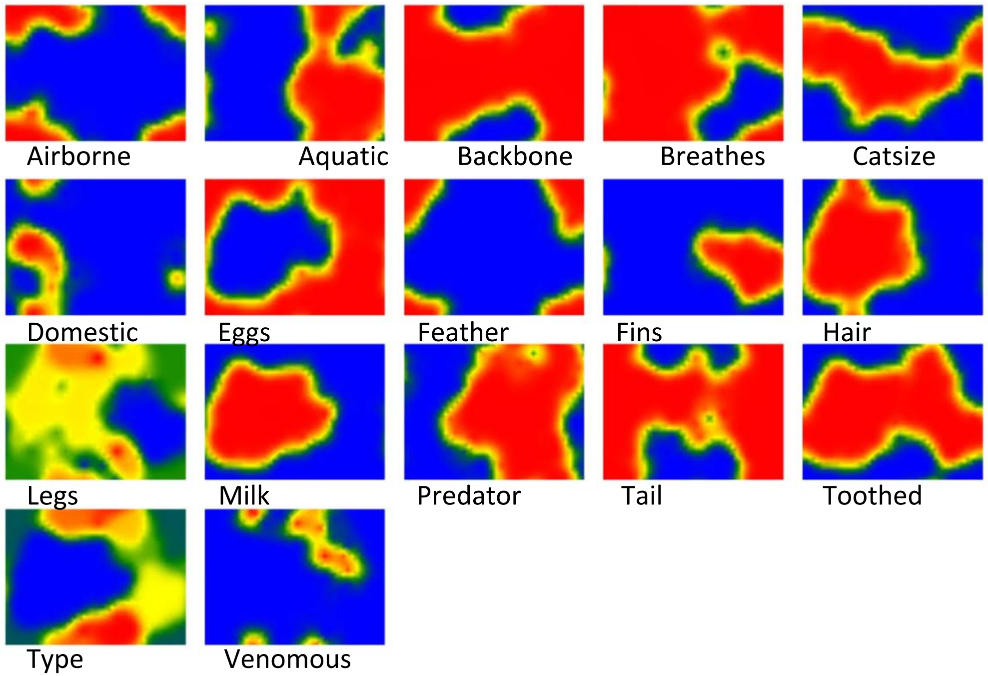
Visualisierung Komponenten-Matrix

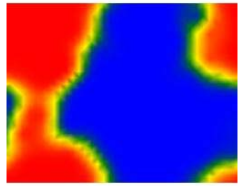
- Die Komponentenmatrix stellt nur die Wichtungen eines speziellen Eingabeneurons durch Einfärbung der Ausgabeneuronen dar.

Kalibrierte U-Matrix der Datenmenge ‚Zoo‘

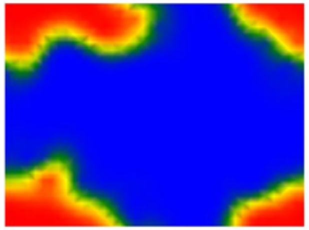


Alle Komponentenmatrizen

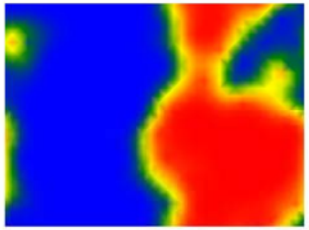




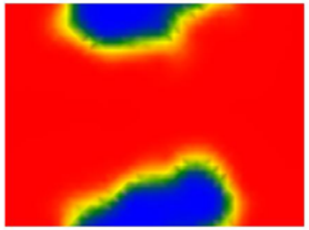
Visualisierung Komponenten-Matrix



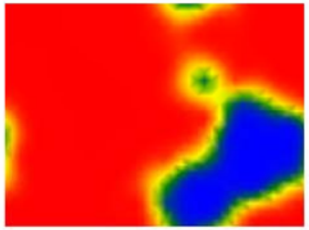
Airborne



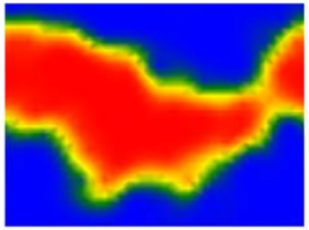
Aquatic



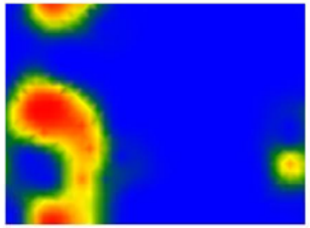
Backbone



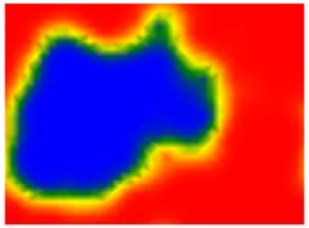
Breathes



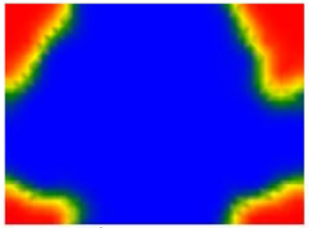
Catsize



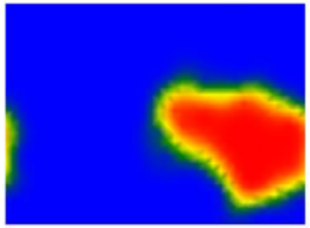
Domestic



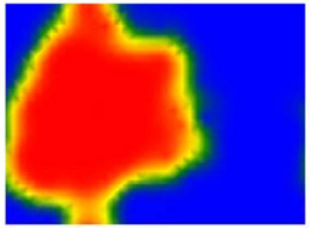
Eggs



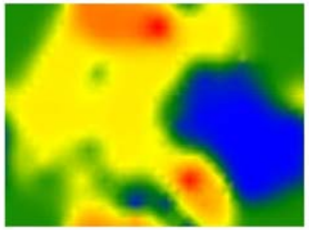
Feather



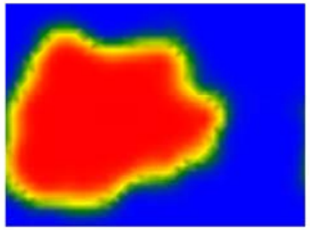
Fins



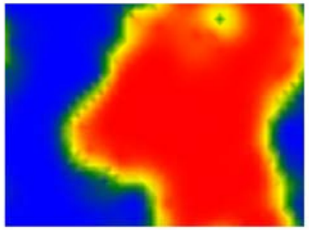
Hair



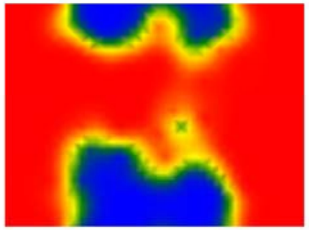
Legs



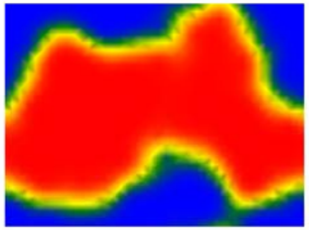
Milk



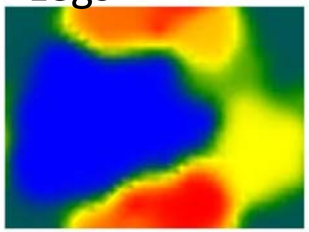
Predator



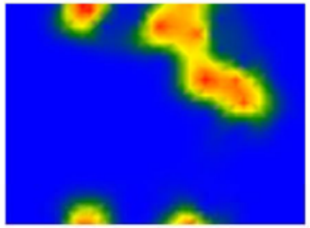
Tail



Toothed

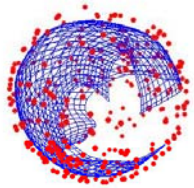


Type



Venomous

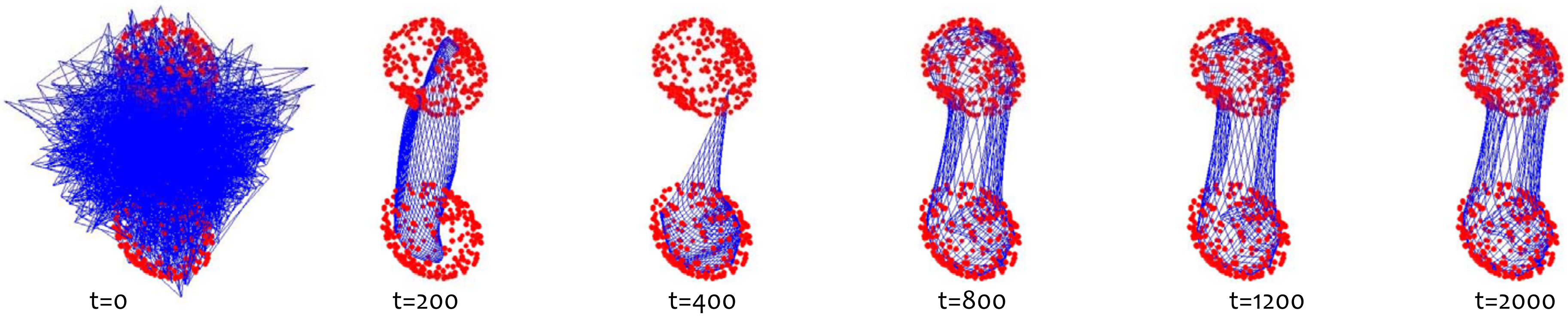
Visualisierung ‚Karte im Eingaberaum'

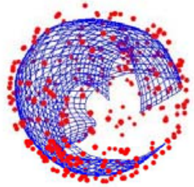


- Karte im Eingaberaum, auch virtual net
- Ausgabeneuronen und ihre topologischen Verbindungen im Eingaberaum
- Wichtungsvektor eines Ausgabeneurons als Koordinatenangabe im Eingaberaum
- Optional Trainingsdatensätze (Punkte im Eingaberaum)
- Visualisierbar bis n = 3

Beispiel: Datenmenge zwei 3D-Kugeln, Karte ein 2D-Gitter (m=2, n=3)

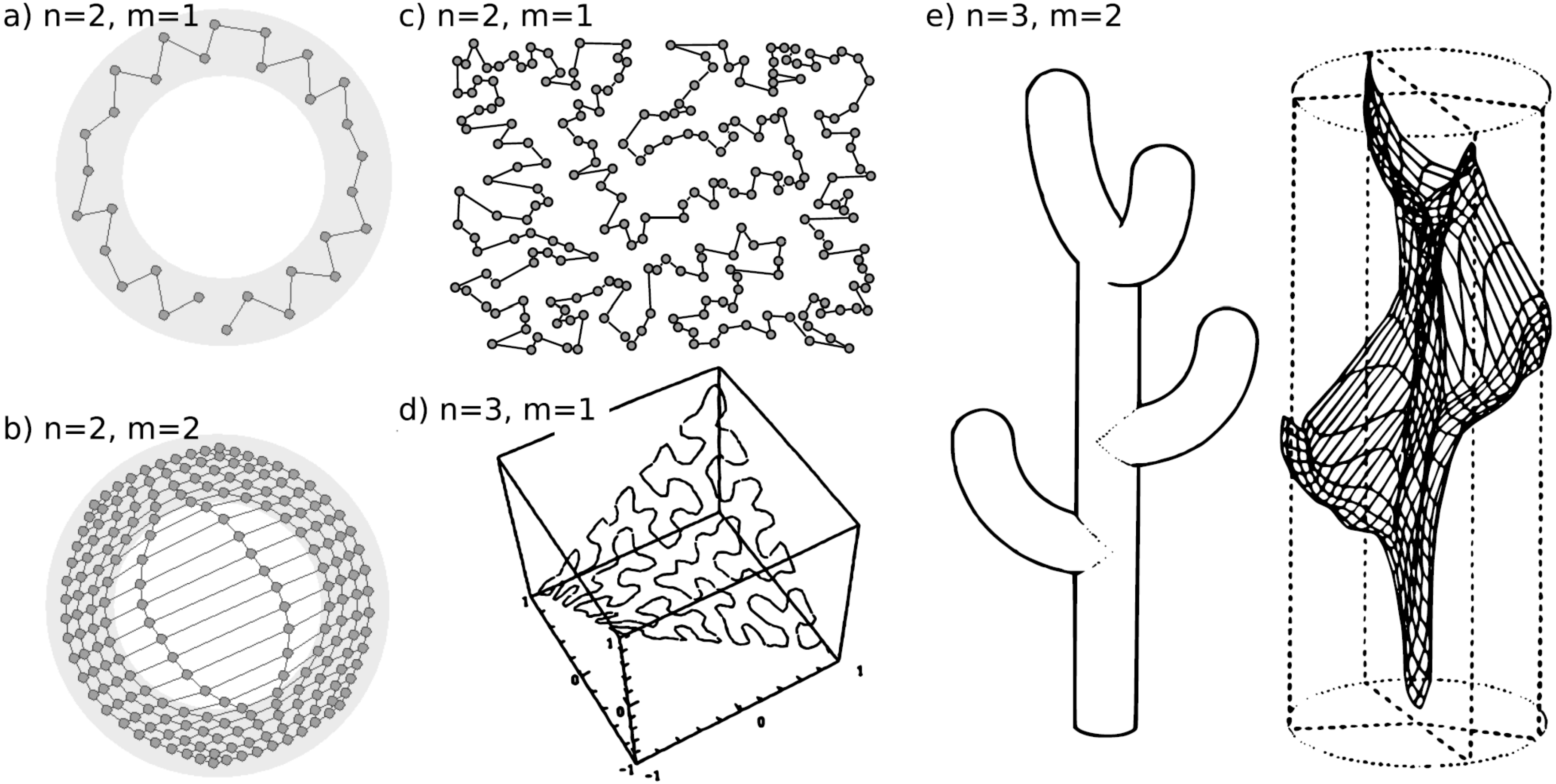
- Sei *m* die Dimension der Ausgabeschicht und *n die Anzahl der Eingabeneuronen*





Visualisierung ‚Karte im Eingaberaum‘

m die Dimension der Ausgabeschicht, *n die Anzahl der Eingabeneuronen*



Karte im Eingaberaum für verschiedene Dimensionen a) und b) Eingabemuster nur in grauen Bereichen, Applet DemoGNG c) Lösung eines TSPs mit 230 Städten d) Kette mit drei Eingabeneuronen (Ritter 03) e) Kaktus aus (Kohonen 01): links die 3-D-Verteilung der Eingabemuster, rechts die erzeugte 2-D-Karte

Anwendungsfelder

- *Dimensionsreduktion*
- *Optimierung*
- *Data Mining – Cluster*
- *Data Mining – Regeln*
- *Überwachung und Anomaliedetektion*
- *Kontextkarten (engl. contextual maps)*

Anwendungsfelder

- ***Dimensionsreduktion***
- *Optimierung*
- *Data Mining – Cluster*
- *Data Mining – Regeln*
- *Überwachung und Anomaliedetektion*
- *Kontextkarten (engl. contextual maps)*

- *Für überwachte Lernverfahren (z. B. Backpropagation-Verfahren, Entscheidungsbaumlernen)* ist es schwierig, in Trainingsdatenmengen mit vielen Merkmalen (hochdimensionaler Eingaberaum) die Merkmalskombinationen zu identifizieren, die für die Klassifikation wichtig sind.

- Mit selbstorganisierenden Karten kann die Datenmenge auf eine Karte mit wenigen Dimensionen abgebildet werden.

- Das überwachte Lernverfahren verwendet dann als Eingabemuster den Ort des Gewinnerneurons in der SOM.

Anwendungsfelder

- *Dimensionsreduktion*
- ***Optimierung***
- *Data Mining – Cluster*
- *Data Mining – Regeln*
- *Überwachung und Anomaliedetektion*
- *Kontextkarten (engl. contextual maps)*

- *Für das Problem des* ***Handlungsreisenden*** *können mit Kohonen-Karten* suboptimale Lösungen für hunderte Städte gefunden werden, indem die Koordinaten der Städte als zweidimensionale Eingabemuster kodiert und eine eindimensionale, geschlossene Topologie (Ring) verwendet wird.

- Der Ring sollte mehr Neuronen als Städte aufweisen und wird als Kreis initialisiert.

Anwendungsfelder

- *Dimensionsreduktion*
- *Optimierung*
- ***Data Mining – Cluster***
- *Data Mining – Regeln*
- *Überwachung und Anomaliedetektion*
- *Kontextkarten (engl. contextual maps)*

- *Beim Finden von Strukturen in großen Datenmengen ist* anfangs oft nicht klar, in wie viele und welche Klassen eine Objektmenge (z. B. Telefonkunden) sinnvoll zerlegt werden kann.
- Die *Clusteranalyse findet geeignete Klassen,* beschreibt sie und ordnet die Daten den Klassen zu.
- Durch die **Dimensionsreduktion und Visualisierungsmöglichkeiten** können selbstorganisierende Karten das Auffinden von Clustern unterstützen. Sie sind zudem relativ robust gegenüber fehlenden Attributwerten.

Anwendungsfelder

- *Dimensionsreduktion*
- *Optimierung*
- *Data Mining – Cluster*
- ***Data Mining – Regeln***
- *Überwachung und Anomaliedetektion*
- *Kontextkarten (engl. contextual maps)*

- Sind die Klassengebiete innerhalb der Karte bekannt, so sind diejenigen **Komponenten** (Merkmale im Eingabevektor) **besonders zur Beschreibung einer Klasse geeignet**, deren Komponentenmatrix innerhalb des Klassengebietes wenig und außerhalb stark variiert.

- Somit lässt sich eine Menge kompakter Fuzzy-Regeln zur Klassifikation ableiten.

Anwendungsfelder

- *Dimensionsreduktion*
- *Optimierung*
- *Data Mining – Cluster*
- *Data Mining – Regeln*
- ***Überwachung und Anomaliedetektion***
- *Kontextkarten (engl. contextual maps)*

- *Komplexe Systeme wie Netzwerke oder Produktionsanlagen* verfügen über eine Vielzahl von verteilten Sensoren. Die aktuellen Werte der Sensorik zeigen ein Bild des Systemzustandes.

- Eine mit den normalen Systemzuständen trainierte und kalibrierte Karte kann **die aktuelle Verfassung des Gesamtsystems** als Punkt oder Weg in der Karte darstellen.

- Ein **großer Quantisierungsfehler** deutet auf einen **ungewöhnlichen Zustand** hin, der nicht beim Training auftrat, und dient damit zum Erkennen von Anomalien und Fehlerzuständen.

Anwendungsfelder

- *Dimensionsreduktion*
- *Optimierung*
- *Data Mining – Cluster*
- *Data Mining – Regeln*
- *Überwachung und Anomaliedetektion*
- ***Kontextkarten (engl. contextual maps)***

- *Wird statt eines Eingabevektors dessen **Kontext*** (beispielsweise **Vorgänger und Nachfolger**) als Trainingsvektor einer Clusteranalyse verwendet, so erfolgt die Gruppierung der Muster nach **Ähnlichkeit der Kontexte**.
- In einer Kohonen-Karte sind dann solche Muster benachbart, die oft in gleicher Umgebung auftreten, sie müssen sich dabei **nicht ähneln**.
- Bei Anwendung auf einfache englische Sätze entstehen so Kartenbereiche mit Verben, Substantiven und Adverbien.

MMM - Mindestmerkmenge

- Selbstorganisierende Karten sind eine [redacted] motivierte Modellierung eines [redacted]hten Lernprozesses in der Großhirnrinde.
- Wesentliches Merkmal ist die Vorgabe einer [redacted] und die laterale [redacted]
- Die Karten passen sich an die Häufigkeitsverteilungen in den [redacted] an, und erzeugen so eine (aufs Wesentliche) [redacted] reduzierte Repräsentation.
- Die technischen Eigenschaften führen zu spannenden und erfolgreichen Anwendungsfeldern in der sensornahen Wissensverarbeitung.
- Hoc[redacted] Daten lassen sich visualisieren.

MMM - Mindestmerkmenge

- Selbstorganisierende Karten sind eine biologisch motivierte Modellierung eines unüberwachten Lernprozesses in der Großhirnrinde.
- Wesentliches Merkmal ist die Vorgabe einer Topologie und die laterale Inhibition.
- Die Karten passen sich an die Häufigkeitsverteilungen in den Trainingsdaten an, und erzeugen so eine (aufs Wesentliche) dimensionsreduzierte Repräsentation.
- Die technischen Eigenschaften führen zu spannenden und erfolgreichen Anwendungsfeldern in der sensornahen Wissensverarbeitung.
- Hochdimensionale Daten lassen sich visualisieren.

Anwenden

Verstehen

Zusammenfassung

- Somatosensorische Felder, Penfield-Männchen, Dynamik
- Dimensionsreduktion am Beispiel Zoo
- Neuronenaktivierung mittels Skalarprodukt
- Aufbau aus Ein- und Ausgabeschicht
- Topologien der Ausgabeschicht: Gitter, Torus, Dimension m, Neuronenabstand
- Laterale Inhibition, Erregungszentrum, Gewinnerneuron
- Trainingsalgorithmus, Lernen des Gewinners und seiner Nachbarn
- Nachbarschaftsfunktionen, Lernradius
- Grobstruktur und Feinstruktur durch Absenken von d und η
- Kupferschmied-Metapher
- Quantisierungsfehler eines Musters, Mittlerer Quantisierungsfehler einer Karte
- Visualisierung, U-Matrix, Komponenten-Matrix, Karte im Eingaberaum
- Anwendungsfelder
- Nebenbei Sombrero

Aufgaben aus Kap.10

Aufgabe 10.15 (SOM)

a) *Welchem Lernparadigma werden SOM zugeordnet?*
b) *Was sind somatosensorische Felder im Gehirn?*
c) *Nennen Sie Eigenschaften einer Abbildung, so dass eine Karte entsteht.*
d) *Stellen Sie den Aufbau einer SOM dar und benennen Sie alle Bestandteile.*
e) *Nennen Sie drei Topologien von selbstorganisierenden Karten.*

Aufgabe 10.16 (SOM-Lernvorgang)

a) *Wie wird das Gewinnerneuron bestimmt?*
b) *Wie werden die Wichtungen verändert? (Hinweis: kleiner Unterschied zwischen dem Gewinnerneuron und anderen Neuronen)*
c) *Welche Rolle spielt die Nachbarschaftsfunktion beim Lernen?*
d) *Welche zwei Parameter steuern den Lernprozess?*
e) *Der Lernprozess kann als Wanderung der Kartenneuronen im Eingaberaum interpretiert werden – diese ist aber nicht ziellos, sondern?*
f) *Wie ist der Quantisierungsfehler einer Karte bezüglich eines Musters definiert und was drückt er aus?*
g) *Erläutern Sie die Kupferschmied-Metapher.*

Aufgabe 10.17 (SOM-Visualisierung)

a) *Nennen Sie drei Arten von Visualisierungen von SOM mit je einer Beispielmethode.*
b) *Was stellt die Komponentenmatrix dar?*
c) *Wie funktioniert die Darstellungsform „Karte im Eingaberaum"?*
d) *Wie berechnet sich ein Element der U-Matrix?*
e) *Was versteht man unter Kalibrierung einer Karte?*

Aufgabe 10.18 (SOM-Anwendung)

a) *Nennen Sie Anwendungsfelder von SOM.*
b) *Wie können SOM das Backpropagation-Verfahren unterstützen?*
c) *Erläutern Sie die Grundidee der SOM-Anwendung Kontextkarten.*
d) *Wie kann eine SOM zur Anomaliedetektion verwendet werden?*