

Computational Intelligence

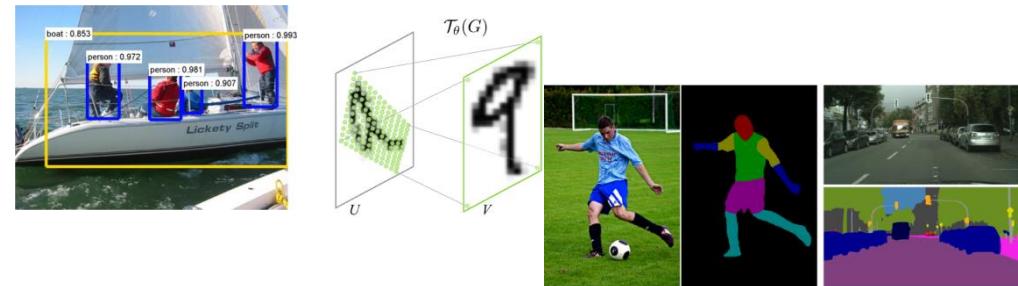
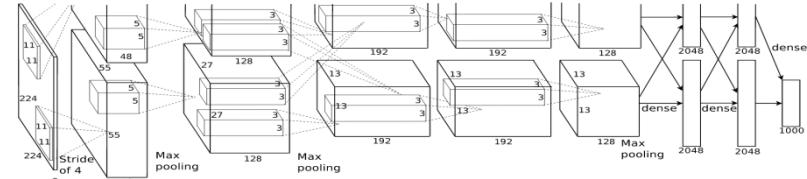
7. Rekurrente Netze cont., Unüberwachtes Lernen

Prof. Dr. Sven Behnke

Letzte Vorlesung

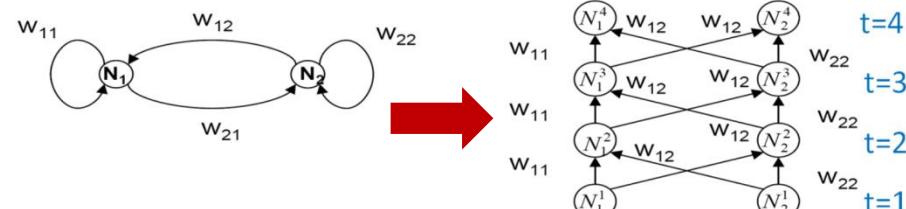
■ Deep Learning

- Kategorisierung von Bildern
- Objektdetektion
- Deformierbare Modelle
- Semantische Segmentierung



■ Rekurrente Neuronale Netze

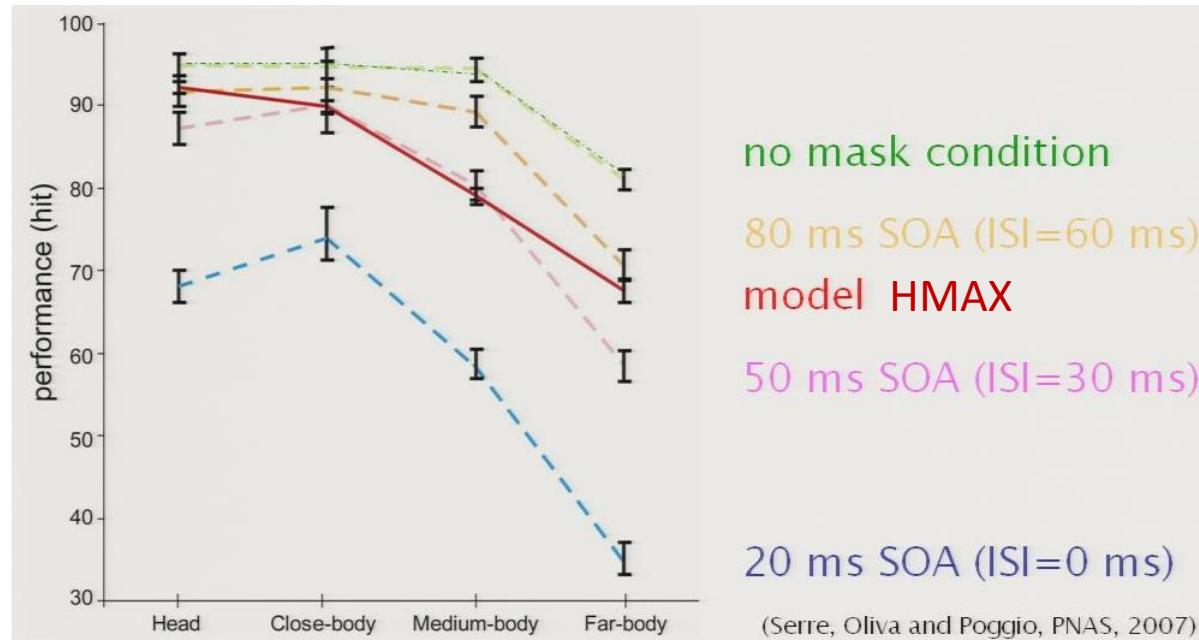
- Umwandlung in vorwärtsgerichtetes Netz durch Entfaltung entlang der Zeitachse:



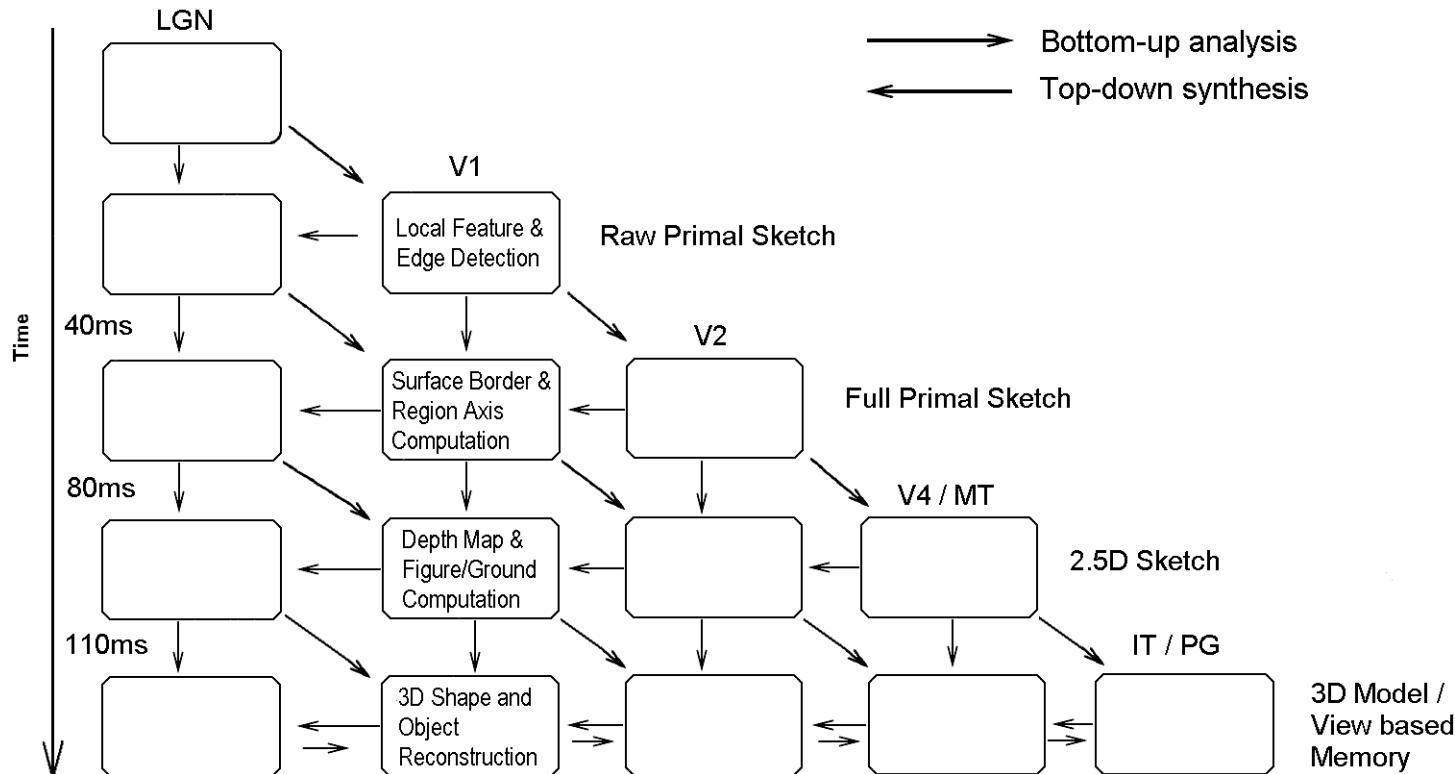
- Weight Sharing
- Training durch Backpropagation Through Time (BPTT)

Feed-forward-Modelle können menschliche Performanz nicht erklären

- Erkennungsleistung steigt mit der Beobachtungszeit



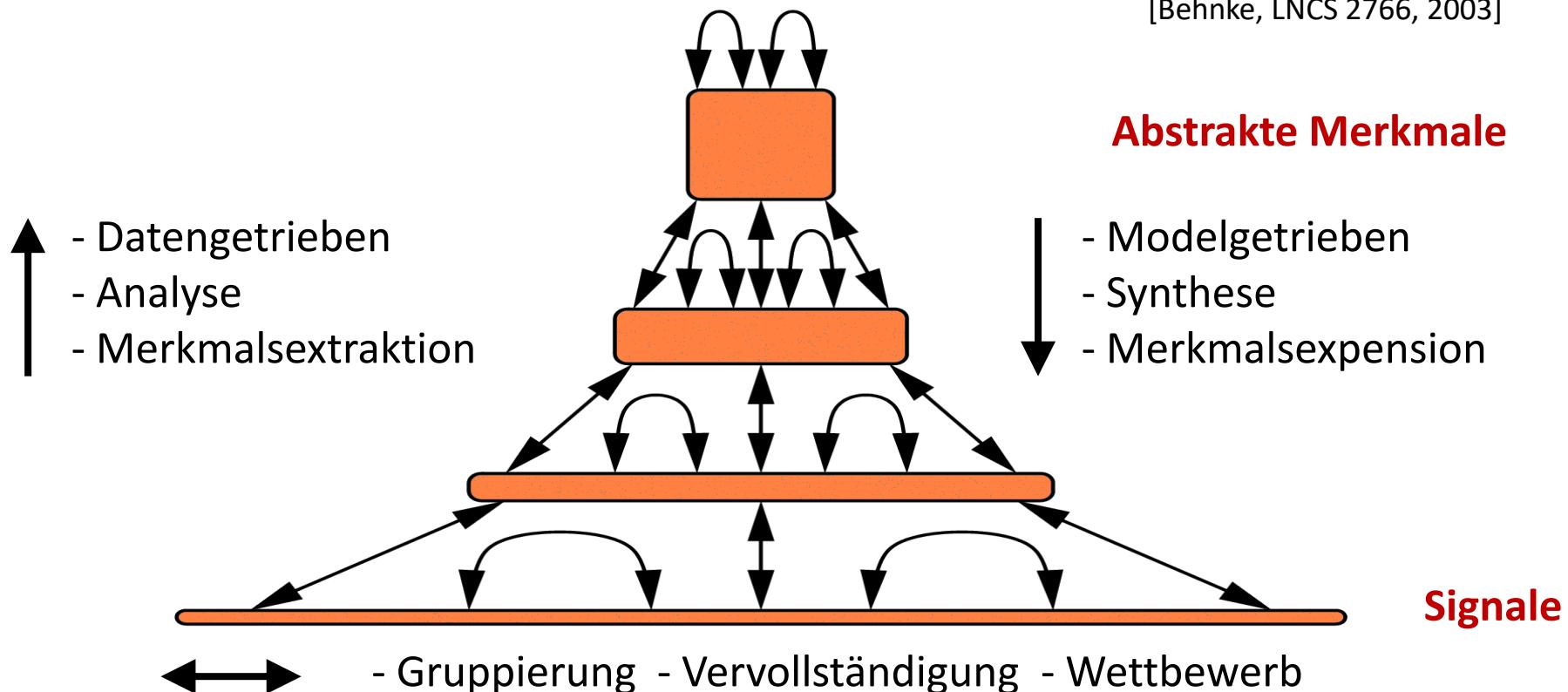
Bottom-up, Lateral und Top-down – Verarbeitung



[Lee et al. 1998]

NEURONALE ABSTRAKTIONSPYRAMIDE

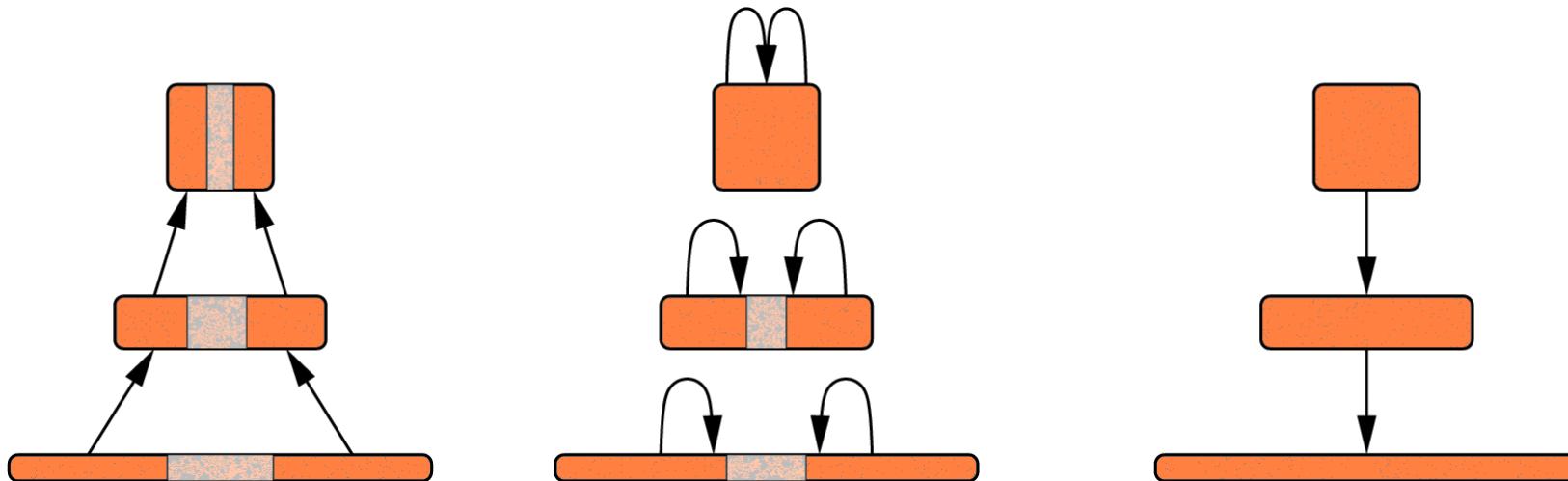
[Behnke, LNCS 2766, 2003]



ITERATIVE INTERPRETATION

[Behnke, LNCS 2766, 2003]

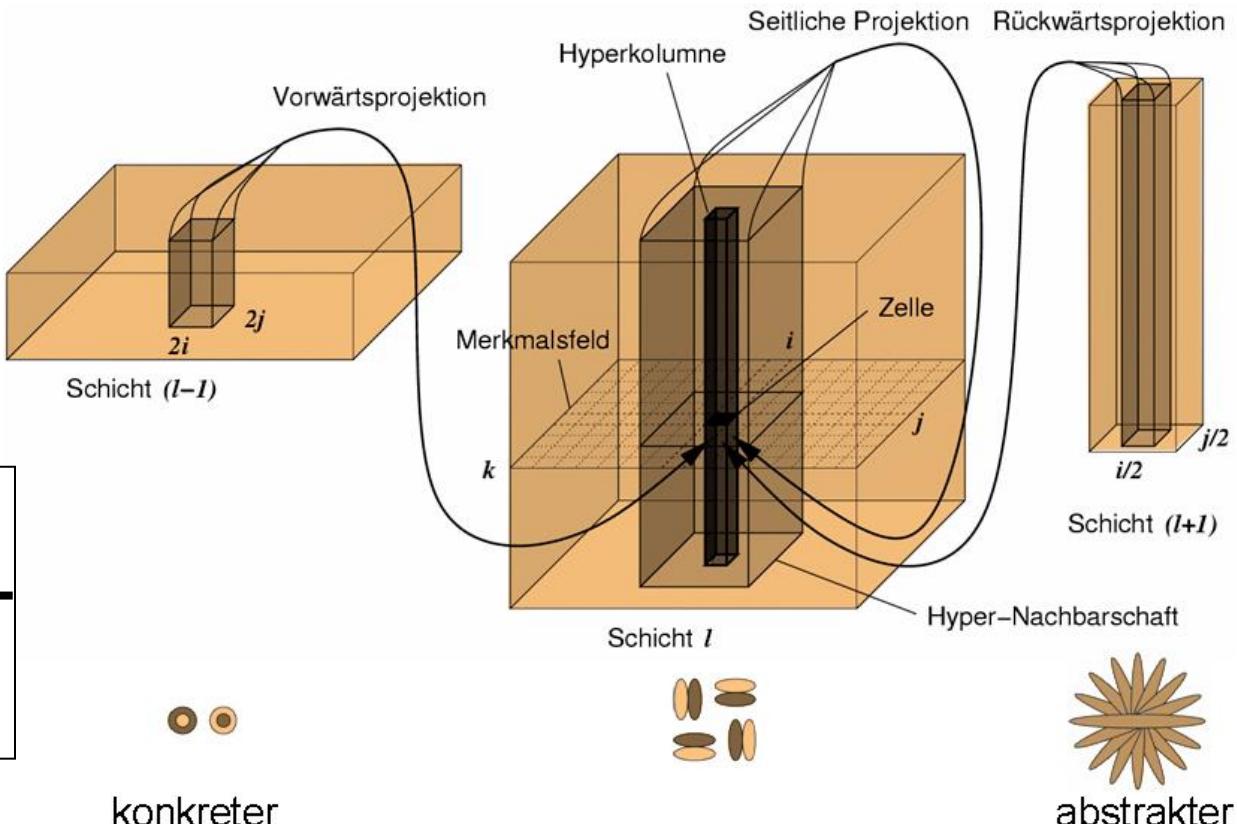
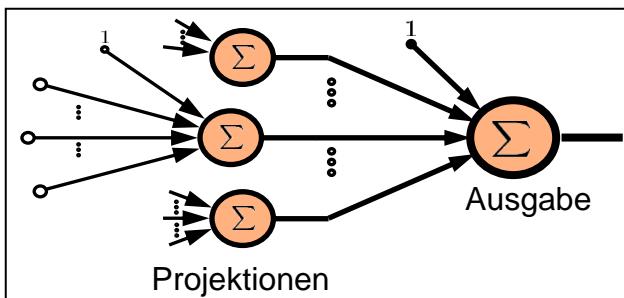
- Interpretiere die offensichtlichsten Stellen im Bild zuerst



- Nutze partielle Interpretation als Kontext um lokale Mehrdeutigkeiten aufzulösen

LOKALE REKURRENTE VERBINDUNGSSTRUKTUR

Prozessorelement

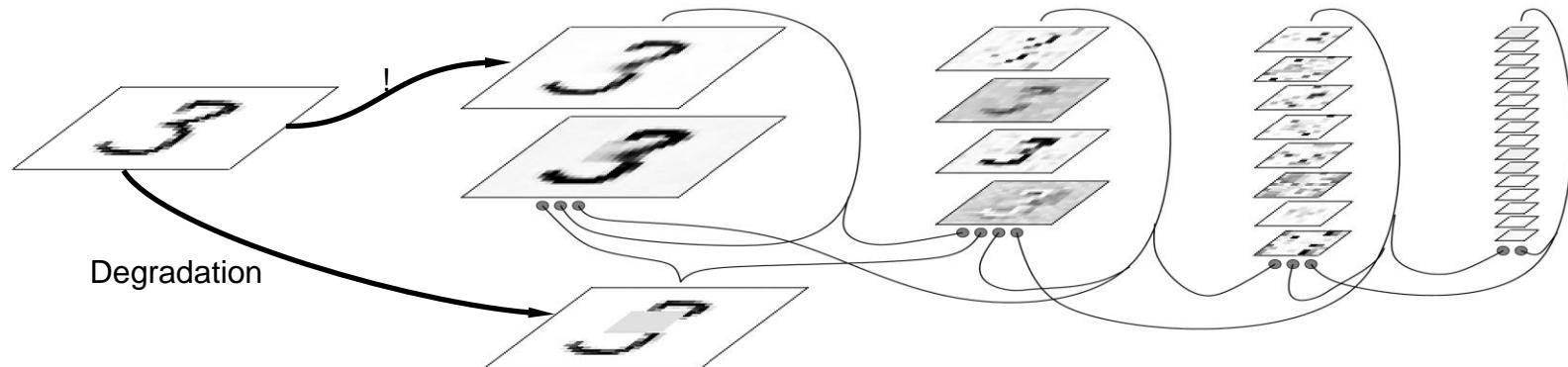


konkreter

[Behnke, LNCS 2766, 2003]

REKONSTRUKTION VON ZIFFERN

[Behnke, IJCAI'01]

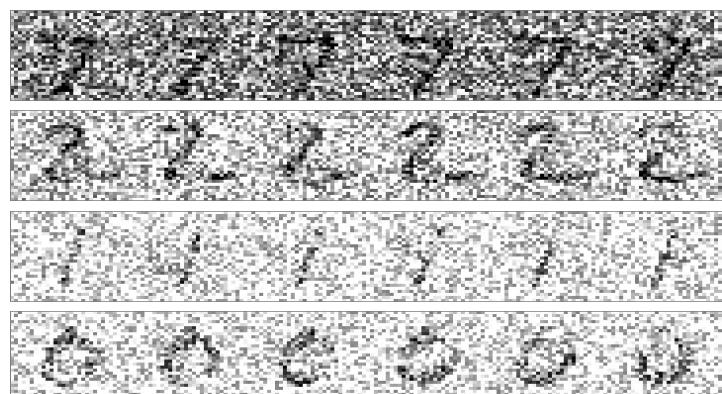
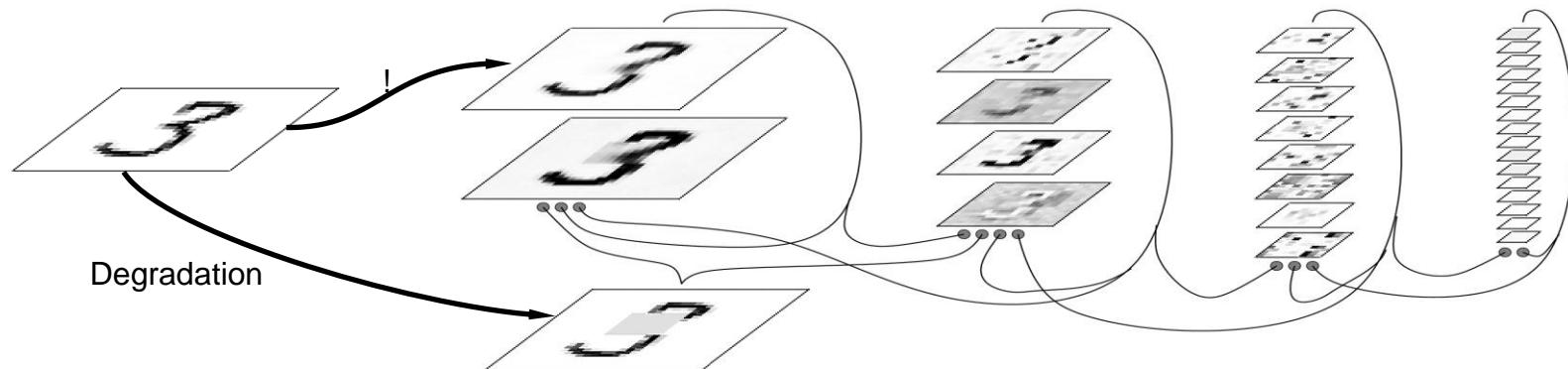


	Input	3	6	12	Output	Target
7	7	7	7	7		
2	2	2	2	2		
1	1	1	1	1		
0	0	0	0	0		
4	4	4	4	4		

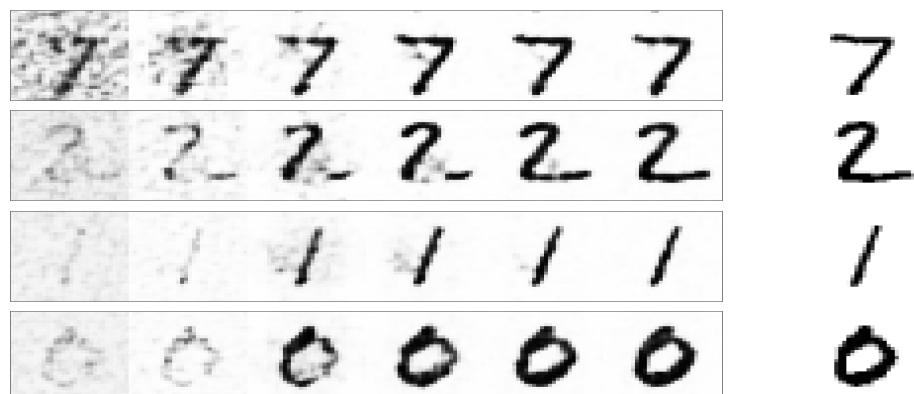
	Input	3	6	12	Output	Target
7	7	7	7	7		
2	2	2	2	2		
1	1	1	1	1		
0	0	0	0	0		
4	4	4	4	4		

REKONSTRUKTION VON ZIFFERN

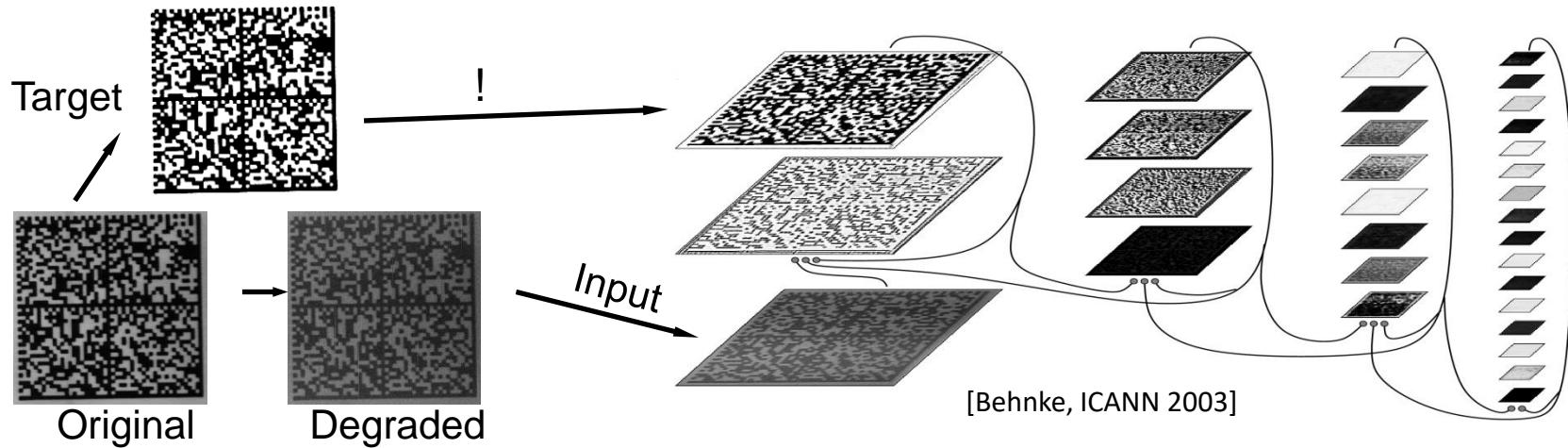
[Behnke, IJCAI'01]



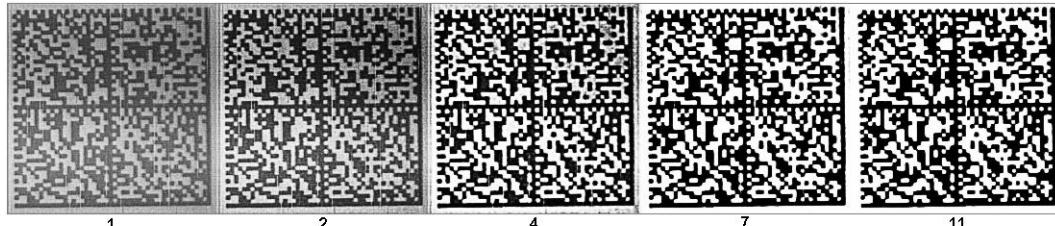
Input



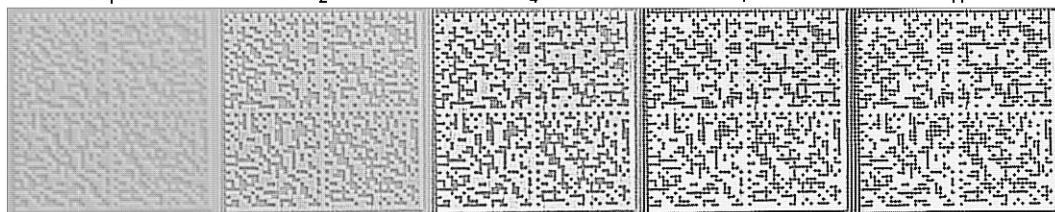
BINARISIERUNG VON MATRIXCODES



Output



Hidden



LOKALISIERUNG VON GESICHTERN

[Behnke, KES'03]

■ BiOLD-

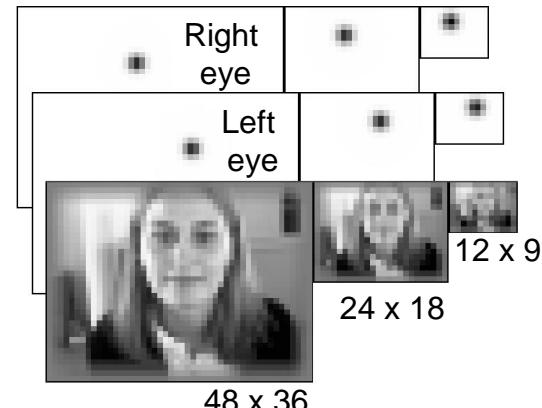
Datenmenge:

- 1521 Bilder
- 23 Personen



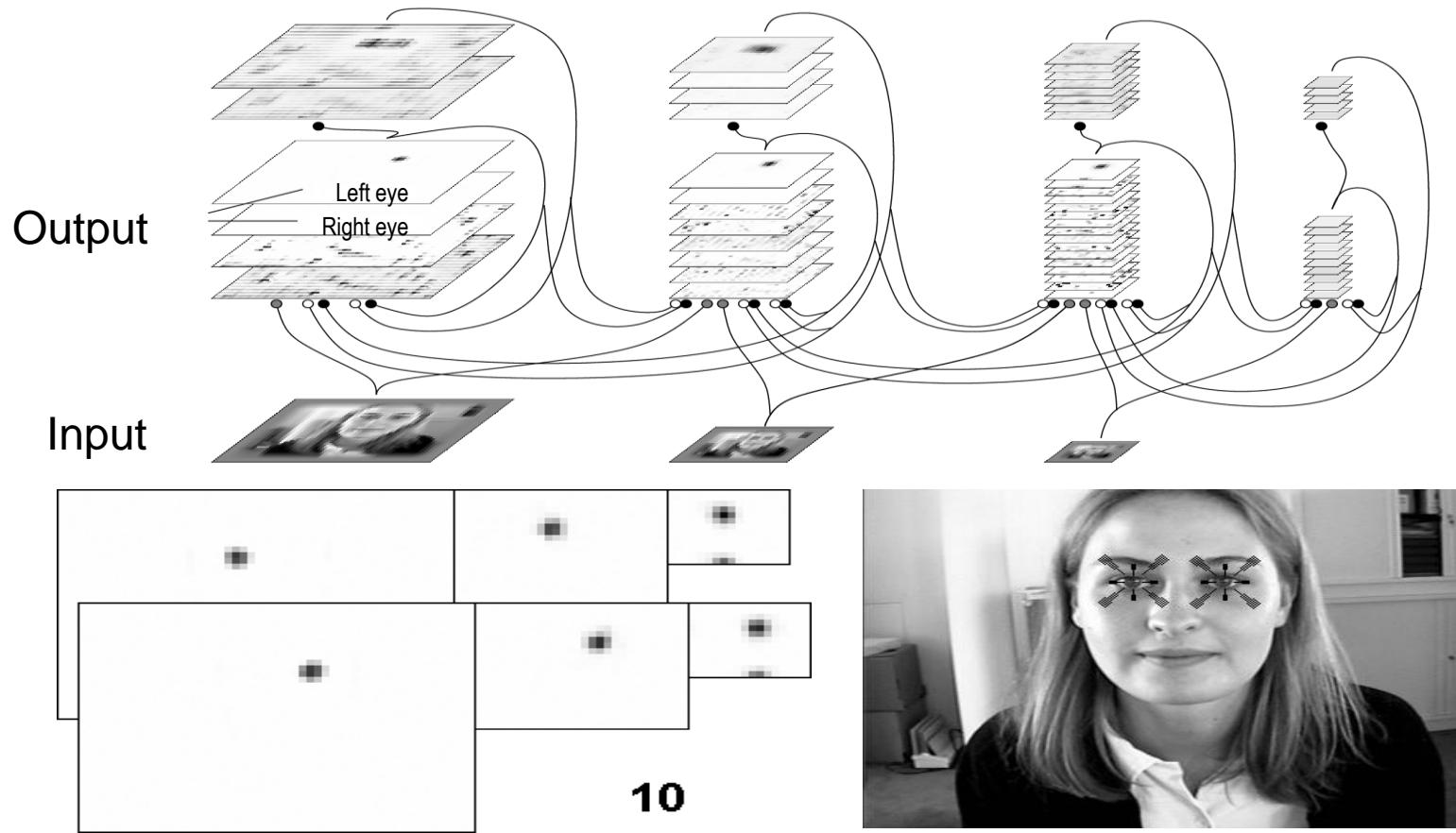
■ Codiere

Augenpositionen
durch
Aktivitätsblobs



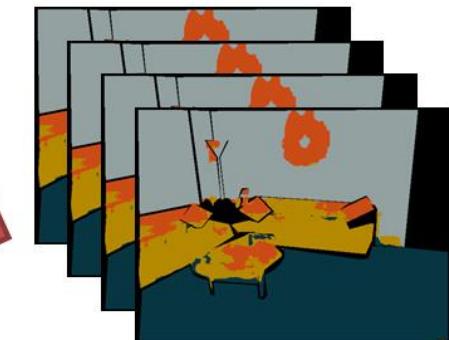
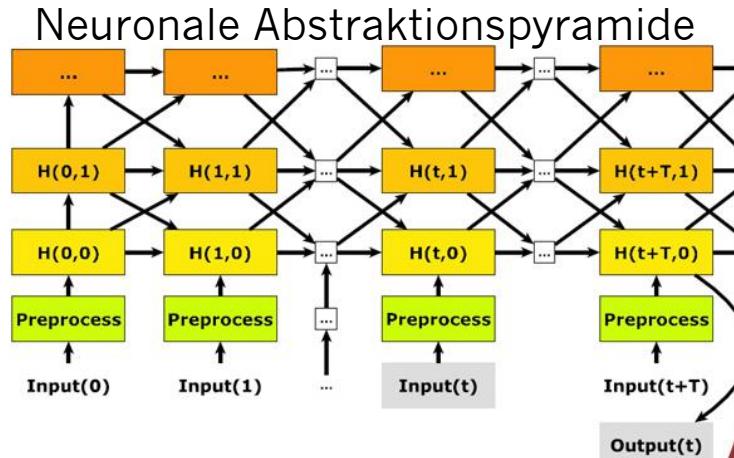
LOKALISIERUNG VON GESICHTERN

[Behnke, KES'03]



SEMANTISCHE SEGMENTIERUNG VON RGB-D-VIDEO

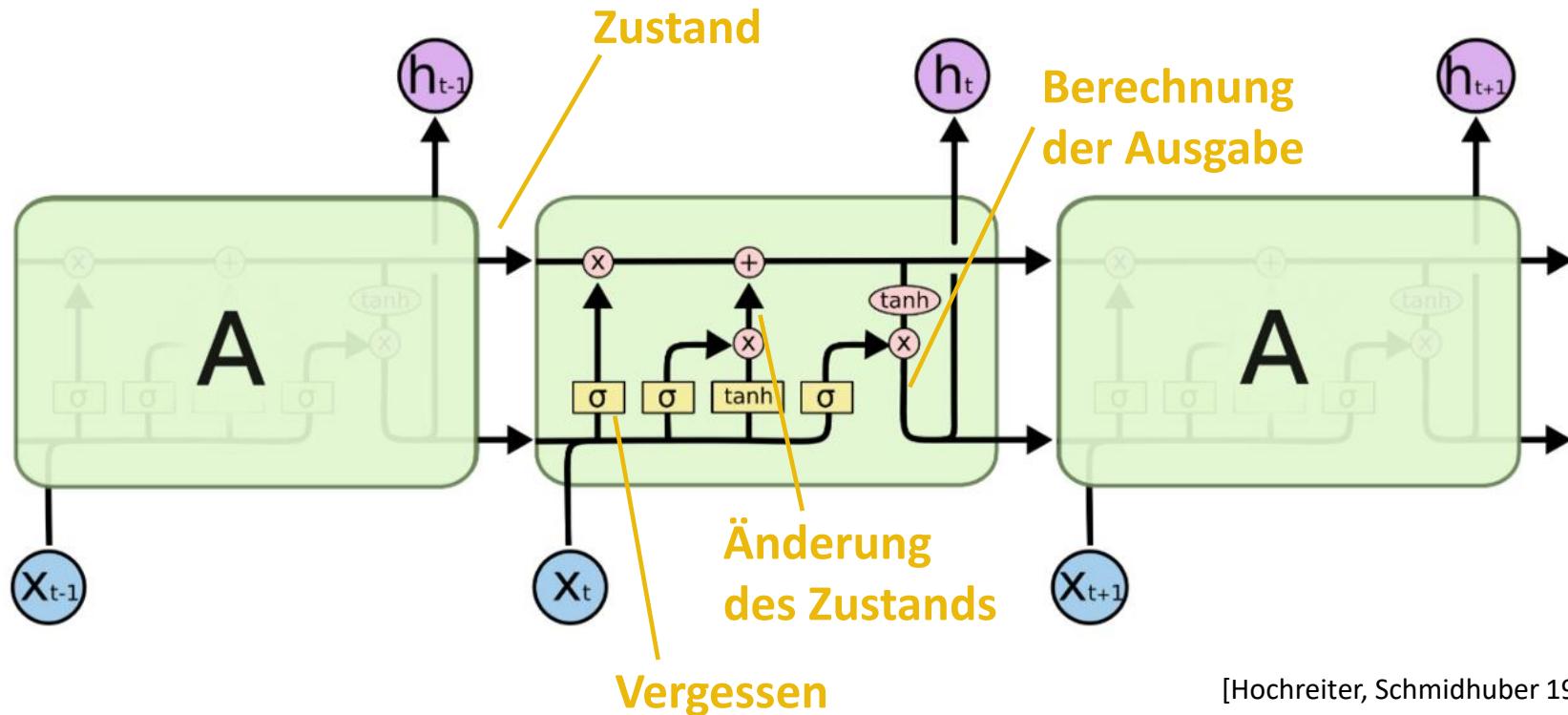
- Rekursive Berechnung kann Informationen über die Zeit integrieren



[Pavel, Schulz, Behnke, Neural Networks 2017]

LONG SHORT-TERM MEMORY (LSTM)

- Problem: Modellierung längerfristiger Abhängigkeiten
- Kernidee: Verdrahte Speicherung des Zustands, kontrolliere Änderungen



[Hochreiter, Schmidhuber 1997]

GENERIEREN VON BILDBESCHREIBUNGEN

■ Rekurrentes neuronales Netz

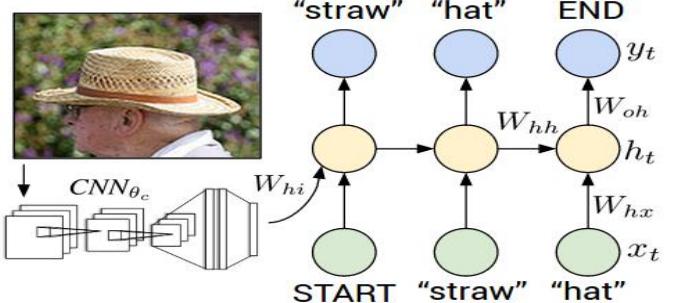
[Karpathy, Fei-Fei 2015]



man in black shirt is playing guitar.

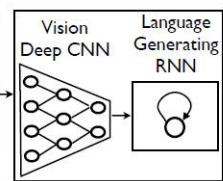


construction worker in orange safety vest is working on road.



two young girls are playing with lego toy.

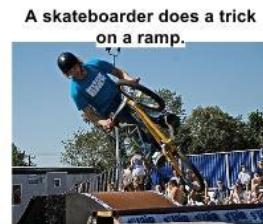
GENERIEREN VON BILDBESCHREIBUNGEN



A group of people shopping at an outdoor market.
There are many vegetables at the fruit stand.



Two dogs play in the grass.



A skateboarder does a trick on a ramp.



A dog is jumping to catch a frisbee.



A group of young people playing a game of frisbee.



Two hockey players are fighting over the puck.



A little girl in a pink hat is blowing bubbles.



A refrigerator filled with lots of food and drinks.



A herd of elephants walking across a dry grass field.



A close up of a cat laying on a couch.



A red motorcycle parked on the side of the road.



A yellow school bus parked in a parking lot.

Describes without errors

Describes with minor errors

Somewhat related to the image

Unrelated to the image

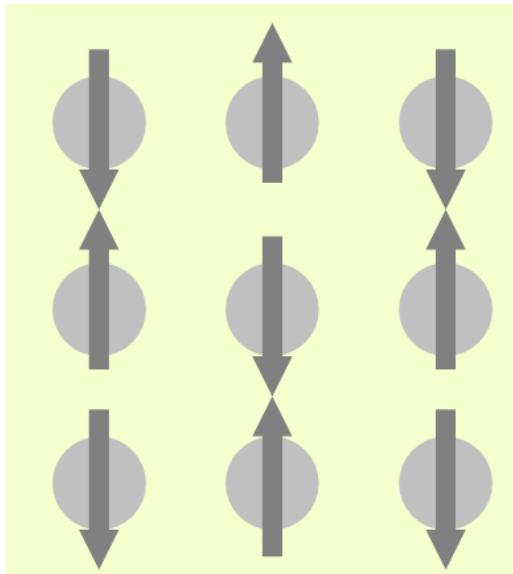
TRÄUMENDE TIEFE NETZWERKE



[Mordvintsev et al 2015]

HOPFIELD-NETZE [JOHN HOPFIELD, 1982]

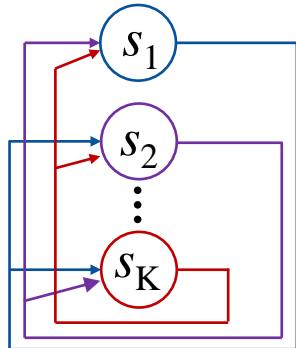
- Hopfield-Netze realisieren Assoziativspeicher.
- Muster können eingeprägt und später aus Fragmenten oder verrauschten Eingaben rekonstruiert werden.
- Sie sind physikalisch motiviert:



- Elementarmagnete (Neuronen) beeinflussen sich gegenseitig
- Sie drehen sich dabei auf der Suche nach dem *energetisch günstigsten Zustand* (Minimum der Energiefunktion)
- Es werden zwei Drehwinkel, sog. **Spins** zugelassen

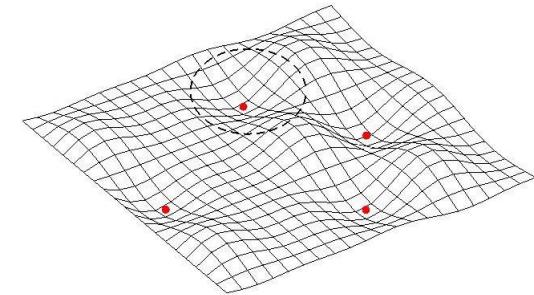
HOPFIELD-NETZWERK

[Hopfield 1982]

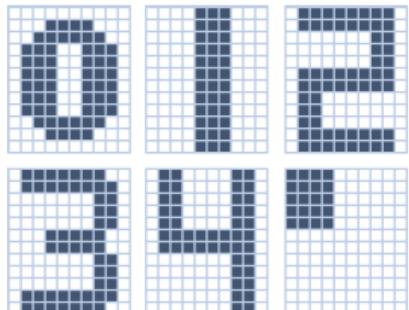


- Vollständig vernetzte binäre Units
- Symmetrische Gewichte $w_{ij}=w_{ji}$, $w_{ii}=0$
- Nichtlineares Dynamisches System minimiert Energie
- Updateregel:

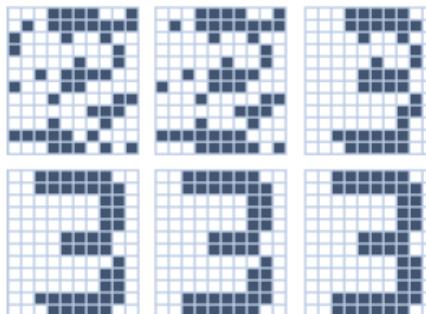
$$s_i \leftarrow \begin{cases} +1 & \text{if } \sum_j w_{ij} s_j \geq \theta_i, \\ -1 & \text{otherwise.} \end{cases}$$



Gespeicherte Muster



Schrittweises Entrauschen

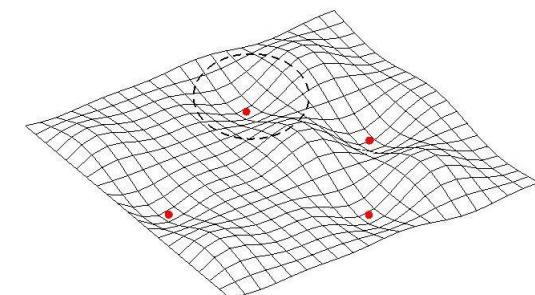


BERECHNUNGSREIHENFOLGE

- Asynchrone Aktualisierung
 - In jedem Zeitpunkt ändert nur ein zufällig ausgewähltes Neuron seine Aktivierung
 - Ursprüngliche Propagierungsregel, gut für theoretische Analyse
- Synchrone Aktualisierung
 - Alle Neuronen ändern ihren Zustand gleichzeitig (wie bei anderen Modellen auch)
- Der Zustand des Netzes zum Zeitpunkt t ist der Vektor s^t der Unit-Ausgaben zu diesem Zeitpunkt
 - Ein Netz mit K Neuronen hat einen Zustandsraum der Größe 2^K
 - Nach Anlegen eines Eingabemusters ändert das Netz so lange seinen Zustand, bis ein Attraktor erreicht wird
 - Die Stabilität eines Zustands ist abhängig von den Gewichten und den Schwellwerten

STABILITÄT

- Rekurrente Netze sind nicht notwendig stabil. Es kann zu Oszillationen und Chaos kommen.
- Hinreichend für Stabilität ist, wenn die Gewichtsmatrix W symmetrisch ist und auf der Hauptdiagonalen Nullen hat (Cohen&Grossberg, 1983)
- Betrachte Energiefunktion: $E = -\frac{1}{2} \sum_{i,j} w_{ij} s_i s_j - \sum_i \theta_i s_i$
=> Netzwerkenergie verringert sich bei jedem Zustandsübergang
(und bleibt konstant, wenn Zustand sich nicht ändert)
- Energiefunktion muss nach endlich vielen Schritten ein Minimum erreichen => stabiler Zustand = Attraktor
- Start-Zustände, die auf den Attraktor konvergieren bilden sein Attraktionsbecken
- Durch Wahl der Gewichte w_{ij} kann die Potentiallandschaft geformt werden



EINPRÄGEN DER MUSTER

- Die Wahl der Gewichte richtet sich nach den zu speichernden Mustern
- Es werden für unkorrelierte binäre Muster $\vec{x}_1, \dots, \vec{x}_p$ die Gewichte auf die durchschnittliche Anzahl der gemeinsamen Einsen gesetzt:

$$w_{ij} = \frac{1}{n} \sum_{m=1}^p x_{mi} x_{mj}$$

x_{mi} ist die i -te Komponente des Eingabemusters \vec{x}_m

- Begrenzte Speicherkapazität $|P|_{MAX} \approx 0.14 \cdot |K|$, wobei K Anzahl der Pixel im Muster ist
- Reduziert sich durch Korrelation der Muster

HOPFIELDNETZE ALS AUTOASSOZIATOREN

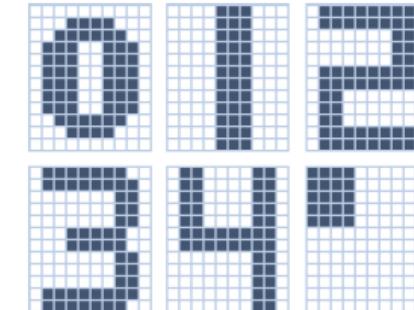
Ein Autoassoziator a hat folgende Eigenschaften:

- Bei Eingabe eines bekannten Musters wird genau dieses wieder ausgegeben

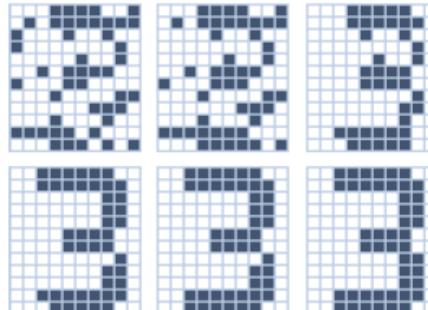
$$a(p) = p$$

- Auch bei verrauschten Daten, die nahe an einem bekannten Muster liegen, erzeugt das Netz die gleiche Ausgabe

$$a(p + \varepsilon) = p$$



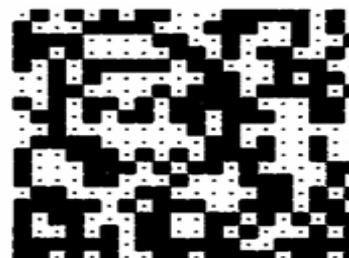
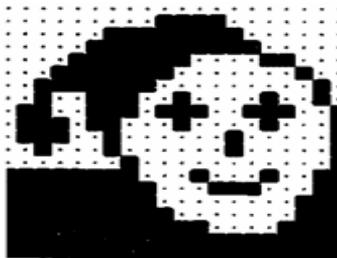
Trainingsbeispiele mit je 10x12 binären Pixeln



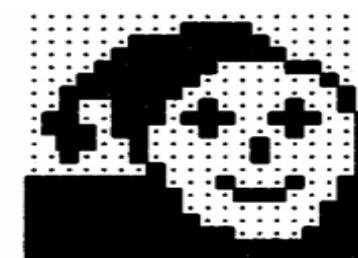
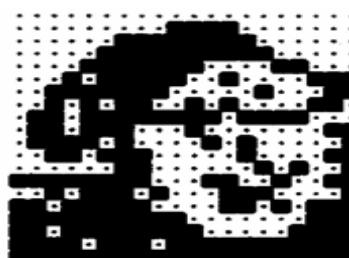
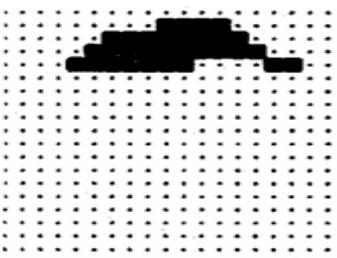
Konvergenz des Netzes bei Eingabe einer stark verrauschten 3

VERVOLLSTÄNDIGUNG VON MUSTERN

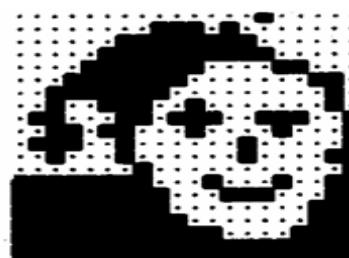
Gespeicherte Muster



Vervollständigung



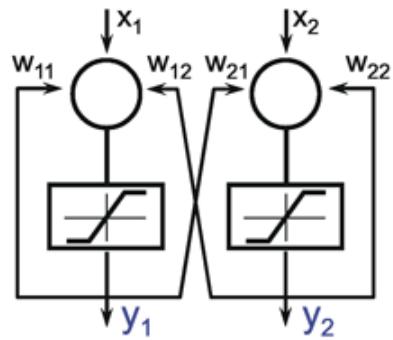
Entrauschen



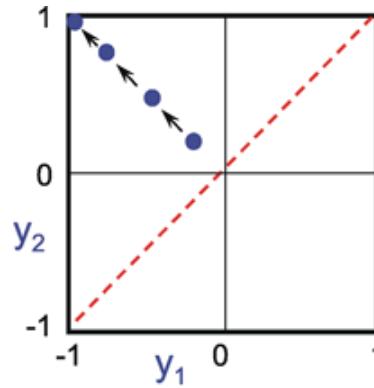
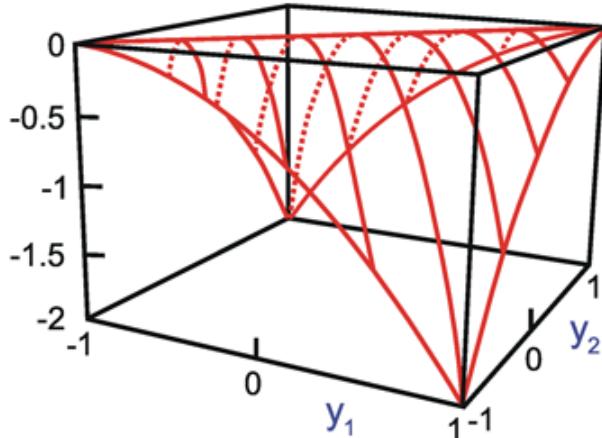
HOPFIELD-NETZE MIT KONTINUIERLICHER AKTIVIERUNG

$$w_{21} = w_{12} = -1$$

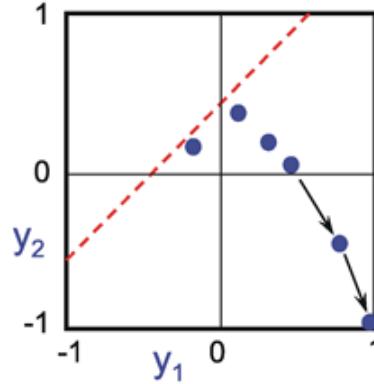
$$w_{11} = w_{22} = 1$$



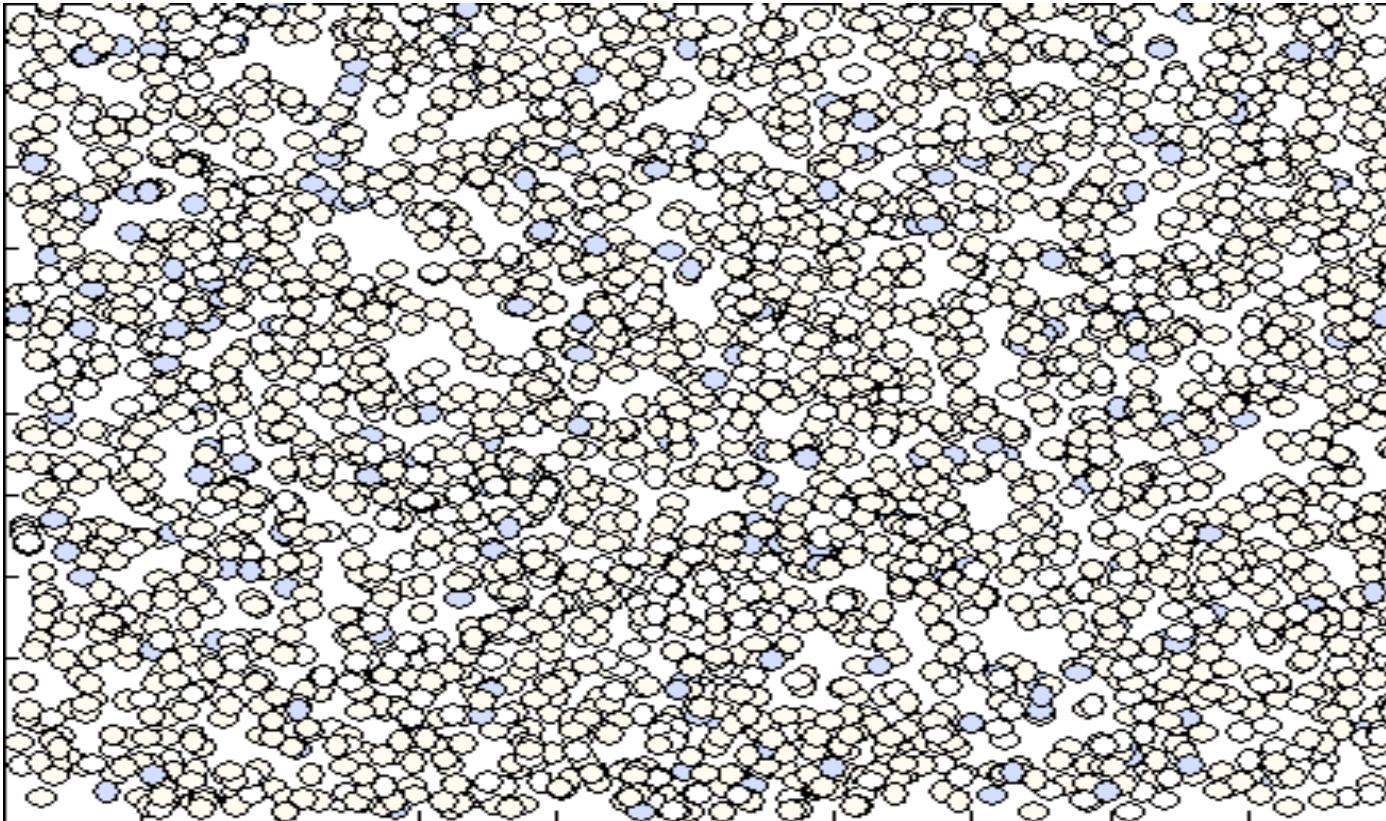
energy



$$\begin{aligned}\theta_1 &= 0.5 \\ \theta_2 &= 0\end{aligned}$$

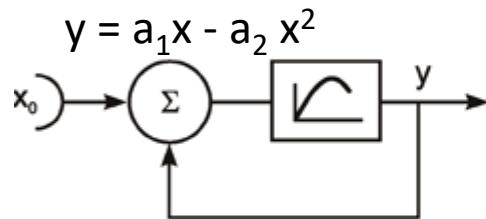


KONTINUIERLICHE ATTRAKTOREN

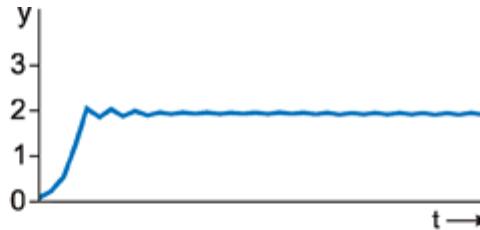


- Lokale Erregung + globale Hemmung
- Stabile Aktivitätsblobs können verschoben werden

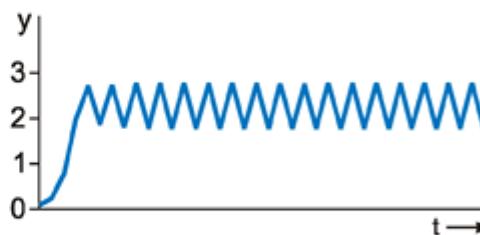
OSZILLATIONEN UND CHAOS



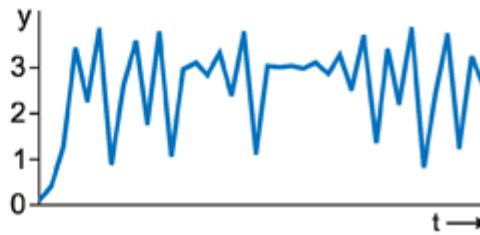
$x_0 = 0.1$



$$a_2 = 2.5$$



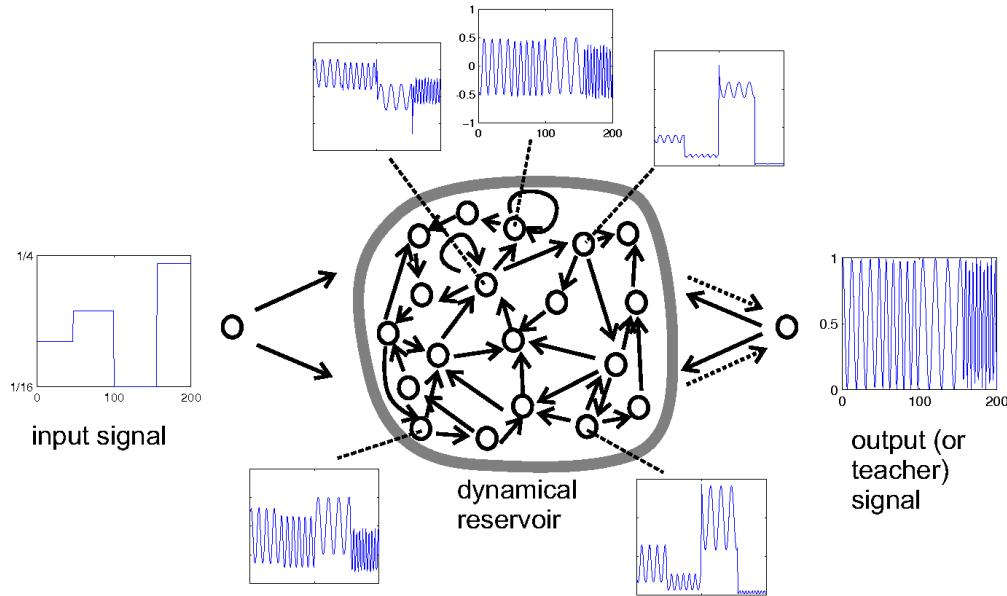
$$a_2 = 3.2$$



$$a_2 = 3.8$$

- Verhalten hängt von Parameterwahl ab.

ECHO STATE NETWORKS [HERBERT JAEGER, 2003]



- Zufällige Initialisierung des Reservoirs
- Abklingende Dynamik (Spektralradius kleiner 1)
- Nur Ausgabe-Gewichte werden trainiert

COMPUTATIONAL INTELLIGENCE UNÜBERWACHTES LERNEN

Prof. Dr. Sven Behnke

DREI ARTEN VON LERNEN

■ Überwachtes Lernen: modelliert $p(y|x)$

- Lerne einen kontinuierlichen Ausgabewert oder ein Klassen-Label aus der Eingabe vorherzusagen

■ Reinforcement-Lernen: Agent interagiert mit Umgebung

- Wähle Aktionen, die Summe der **Beloohnungen maximieren**

■ Unüberwachtes Lernen: modellierte $p(x)$

- Lerne generatives Modell, das erklärt, warum manche Datenvektoren vorkommen und warum andere nicht vorkommen oder
- Lerne eine Energiefunktion, die Daten niedrige Energie zuweist und Nicht-Daten hohe Energie zuweist oder
- Entdecke interessante Merkmale; trenne Quellen, die gemischt wurden; finde Invarianten, usw.

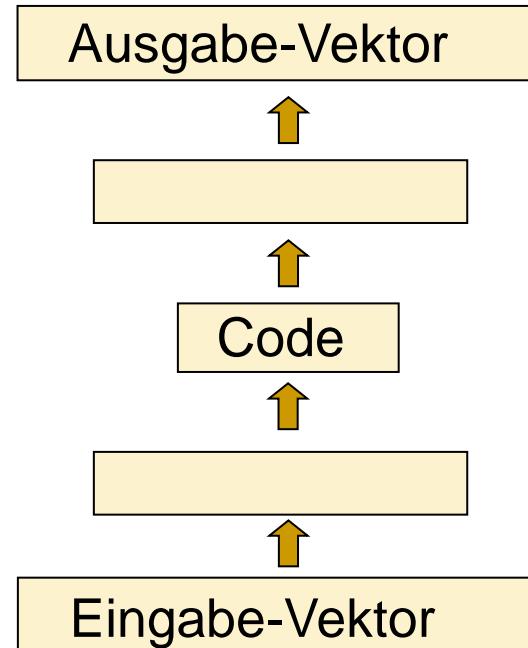
ZIELE DES UNÜBERWACHTEN LERNENS

- Ohne eine gewünschte Ausgabe oder ein Belohnungssignal nicht klar, was gelernt werden soll
- Entdecke nützliche Struktur in den Daten, ohne Hilfe von Labeln oder gewünschten Ausgaben
 - Erzeuge Repräsentationen, die sich besser für nachfolgendes überwachtes Lernen oder Reinforcement-Lernen eignen
 - Erzeuge Dichte-Modell, das benutzt werden kann für:
 - Klassifikation: Teste, welches Modell den Datenvektor am besten beschreibt
 - Überwachung: Detektiere unwahrscheinliche Zustände
 - Interpretation: Extrahiere interpretierbare Faktoren (Ursache-Wirkungs-Zusammenhänge)
- Beschleunige Lernen für hochdimensionale Eingaben
 - Dimensionsreduktion
 - Dekorrelation
 - Erzeugung unabhängiger Merkmale
- Deep Learning
 - Lerne hierarchische Struktur Schicht für Schicht
 - Ausgabe unüberwachten Lernens ist Eingabe für die nächste Schicht

AUTO-ENCODER

- Versuche, die Eingabe durch einen Flaschenhals zu pressen
- Aktivitäten der verdeckten Unis bilden effizienten Code
- Es gibt keinen Platz für Redundanz im Flaschenhals
- Extrahiert häufig unabhängige Merkmale

Gewünschte Ausgabe = Eingabe



LINEARE AUTOENCODER LERNEN HAUPTKOMPONENTENANALYSE

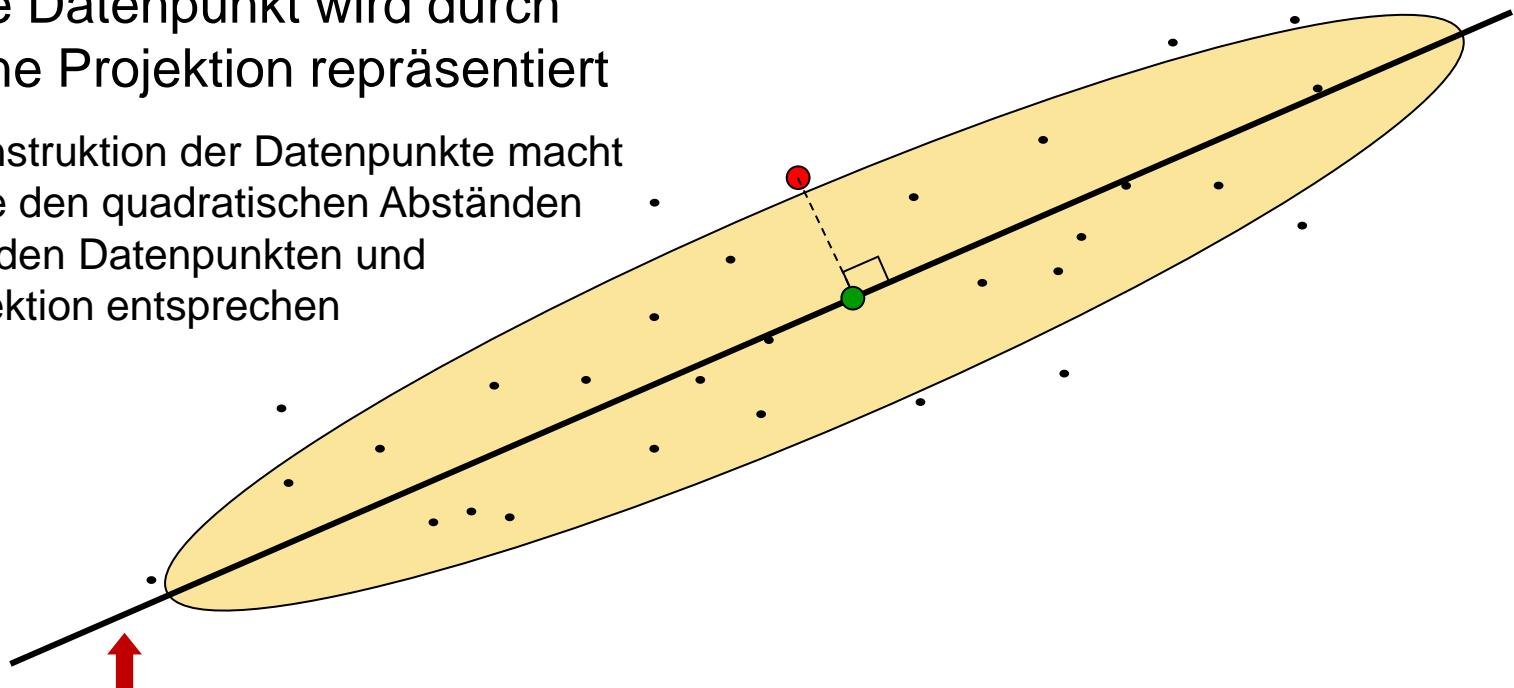
- Wenn das Netzwerk linear ist, lernt es eine lineare Transformation der Daten, die den quadratischen Rekonstruktionsfehler minimiert
 - Genau das tut die Hauptkomponentenanalyse (PCA)
- Die M Hidden-Units spannen den Raum der ersten M Hauptkomponenten der Daten auf
 - Die Gewichtsvektoren müssen nicht orthogonal sein
 - Sie tendieren dazu, die gleiche Varianz zu haben

HAUPTKOMPONENTENANALYSE (PRINCIPAL COMPONENTS ANALYSIS, PCA)

- N-dimensionale Eingaben
- Findet $M \leq N$ orthogonale Richtungen, in denen die Daten die meiste Varianz haben
- Diese M Hauptkomponenten formen einen Unterraum
 - Datenpunkt N -dimensional repräsentiert durch Projektion auf diesen Unterraum
=> Verlust aller Information über den Ort des Datenpunkts in den verbleibenden $N-M$ orthogonalen Richtungen
 - Rekonstruktion durch Mittelwert (über die ganze Datenmenge) in den $N-M$ Richtungen, die nicht repräsentiert sind
=> Rekonstruktionsfehler ist die Summe der quadratischen Abweichungen vom Mittelwert über alle unrepräsentierten Richtungen

PCA MIT N=2 UND M=1

- Der rote Datenpunkt wird durch die grüne Projektion repräsentiert
- Die Rekonstruktion der Datenpunkte macht Fehler, die den quadratischen Abständen zwischen den Datenpunkten und ihrer Projektion entsprechen

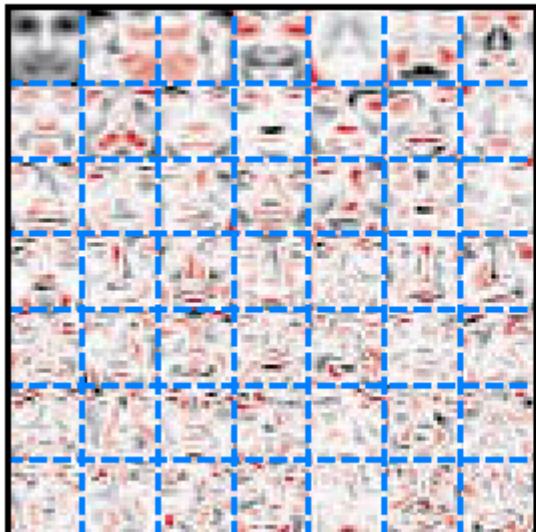


**Erste Hauptkomponente:
Richtung der größten Varianz**

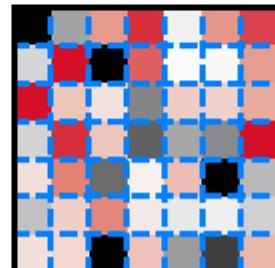
HAUPTKOMPONENTENANALYSE (PCA) FÜR GESICHTER

- Finde orthogonale Basis-Bilder
- Rekonstruiere Eingabe als Linearkombination der Basis-Bilder

PCA



×



=

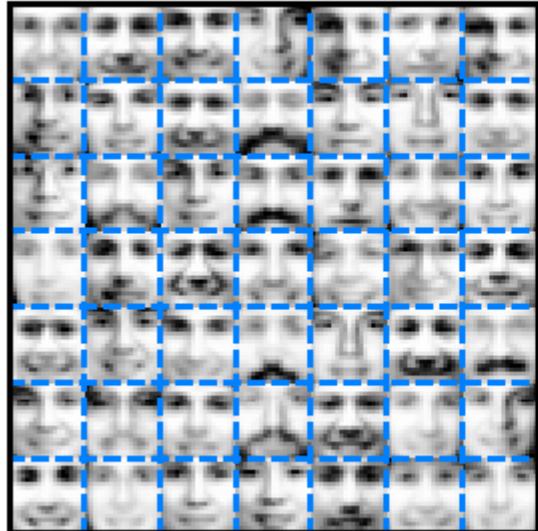
Original



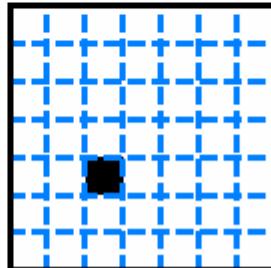
VEKTOR-QUANTISIERUNG (GRUPPIERUNG) VON BILDERN

- Bilder rekonstruiert durch das Basis-Bild, das der Eingabe am nächsten (im Sinne des Euklidschen Abstands) ist

VQ

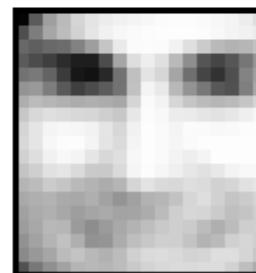
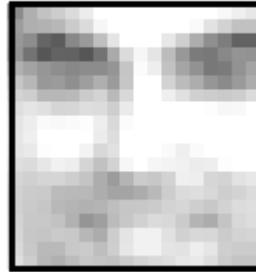


×



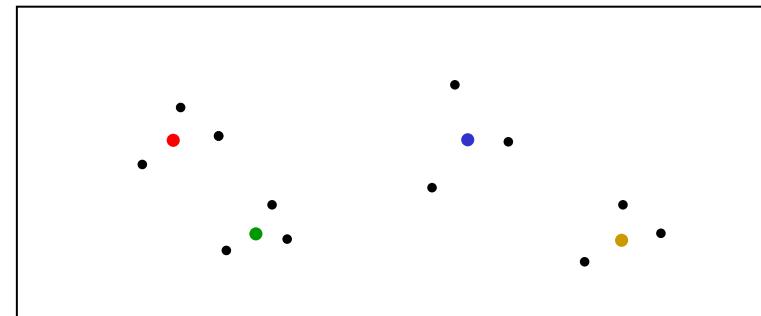
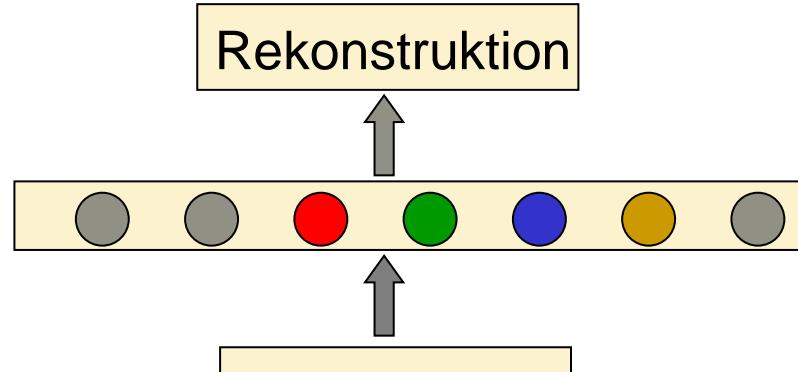
=

Original



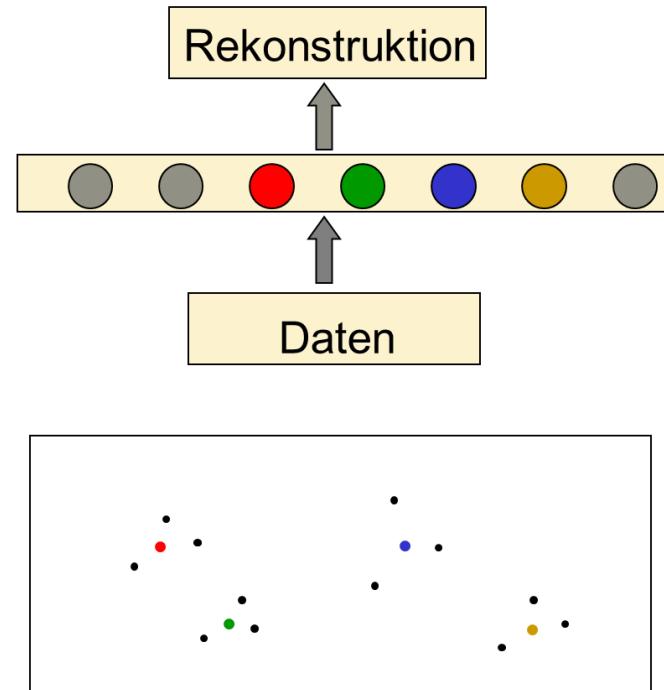
AUTOENCODER UND GRUPPIERUNG

- Wenn die Hidden-Units Aktivität 1 haben falls die Eingabe ähnlich dem Gewichtsvektor ist und Aktivität 0 sonst, entsteht eine Gruppierung der Daten (Clustering)
- Jede Hidden-Unit repräsentiert einen Cluster
- Rekonstruktion der Eingabe als nächstes Cluster-Zentrum



GRUPPIERUNG UND BACKPROPAGATION

- Sorge dafür, dass Input->Hidden-Gewichte und Hidden->Output-Gewichte gleich sind
 - Normalerweise kann man nicht durch binäre Hidden Units zurückpropagieren, aber in diesem Fall sind die Gradienten für die Input->Hidden-Gewichte immer Null.
 - Wenn der Gewinner sich nicht ändert → kein Gradient
 - Der Gewinner ändert sich genau dann, wenn zwei Hidden Units den gleichen Fehler haben -> kein Gradient
- Folglich wirkt der Fehlergradient nur auf die Ausgabegewichte. Er zieht den Gewichtsvektor in Richtung des Datenvektors
- Die Kräfte gleichen sich aus, wenn der Gewichtsvektor im Schwerpunkt eines Clusters liegt. Dies minimiert den quadratischen Rekonstruktionsfehler.



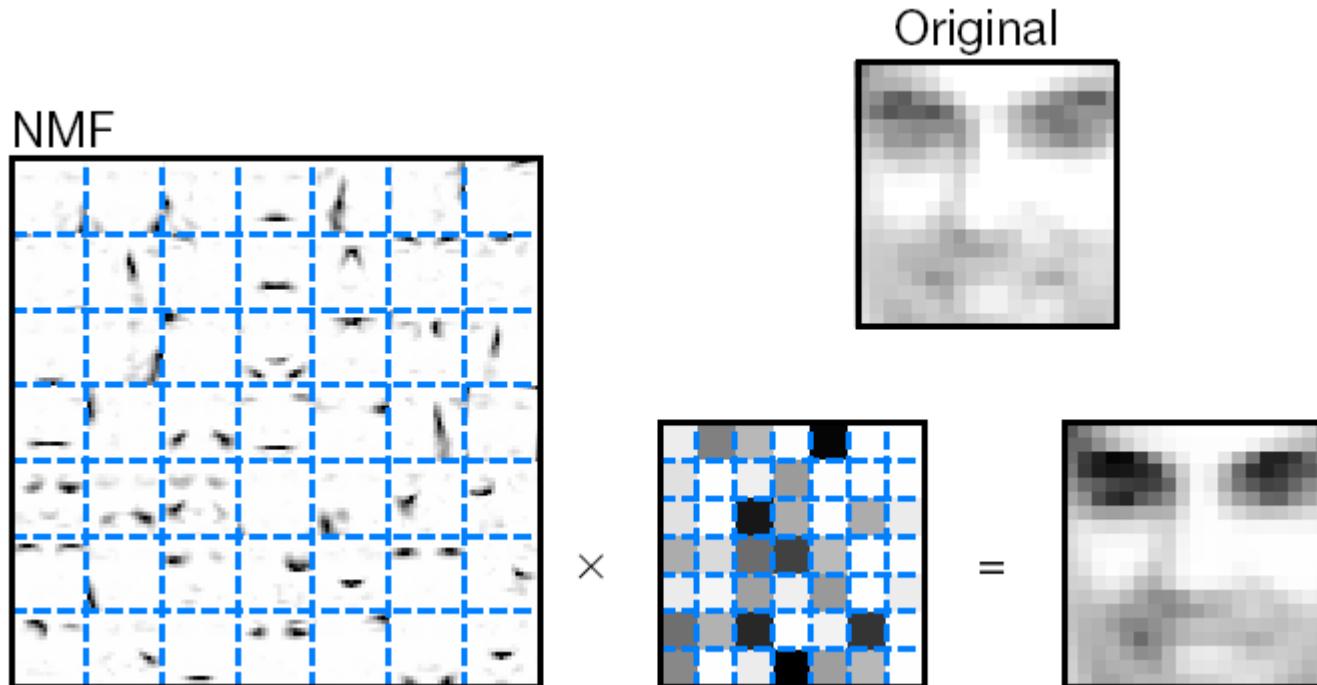
VERSCHIEDENE REPRÄSENTATIONEN

- PCA ist mächtig, weil verteilte Repräsentationen erzeugt werden, aber beschränkt, weil diese linear mit den Daten zusammenhängen
 - Autoencoder mit mehreren verdeckten Schichten und nichtlinearen Transferfunktionen haben dieses Problem nicht
- Gruppierung ist mächtig, weil sehr nicht-lineare Repräsentationen benutzt werden, aber beschränkt, weil die Repräsentationen lokal sind (unäre Codierung).
- Man möchte Repräsentationen, die sowohl verteilt sind, als auch nichtlinear
 - Leider sind diese schwierig zu lernen

	Lokal	Verteilt
Linear		PCA
nicht-linear	Gruppierung	Gewünscht

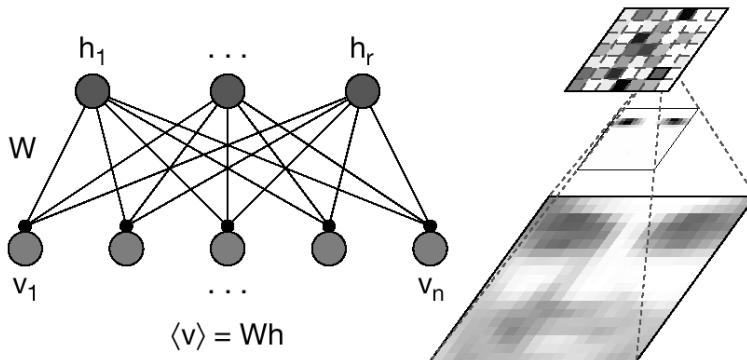
NICHT-NEGATIVE MATRIX-FAKTORISIERUNG (NMF)

- Wie PCA, nur dass die Koeffizienten nichtnegativ sind



NICHT-NEGATIVE MATRIX-FAKTORISIERUNG (NMF)

- Lee & Seung '99
- Zerlege Matrix in nichtnegative Faktoren
- Qualitätsmaß Divergenz $D(V \parallel WH)$
- Iterative Minimierung durch Gradientenabstieg
- Multiplikative Updates
=> Nichtnegativität

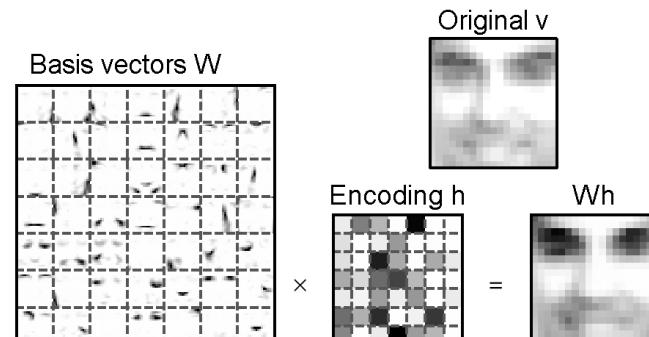


$$V \approx WH$$

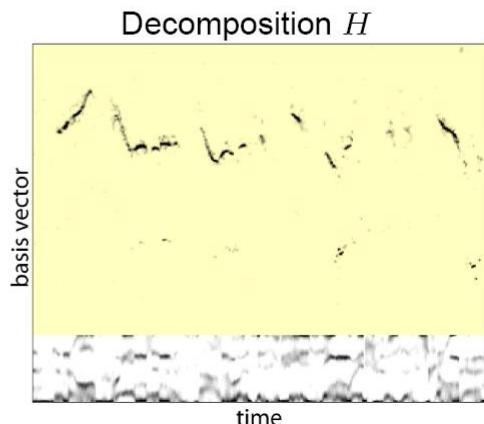
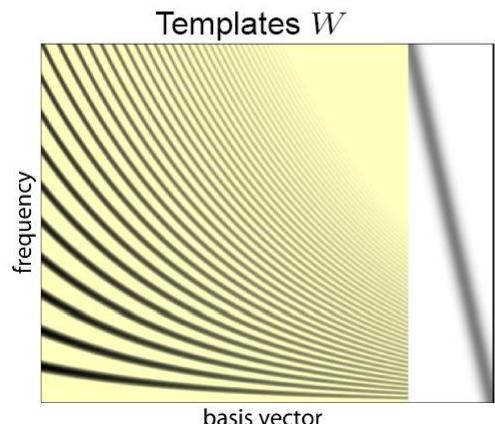
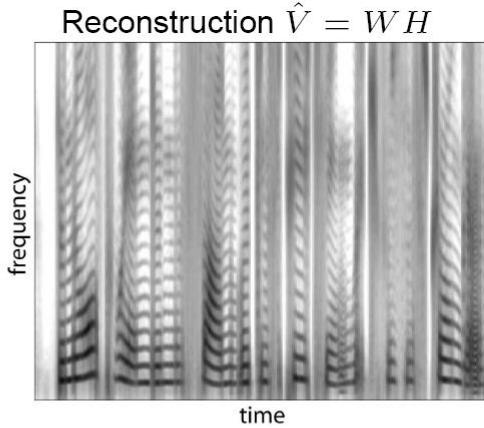
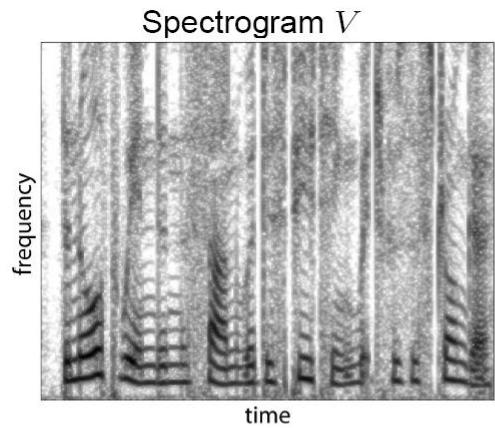
$$D(A \parallel B) = \sum_{ij} (A_{ij} \log \frac{A_{ij}}{B_{ij}} - A_{ij} + B_{ij})$$

$$H_{a\mu} \leftarrow H_{a\mu} \frac{\sum_i W_{ia} V_{i\mu} / (WH)_{i\mu}}{\sum_k W_{ka}};$$

$$W_{ia} \leftarrow W_{ia} \frac{\sum_\mu H_{a\mu} V_{i\mu} / (WH)_{i\mu}}{\sum_\nu H_{a\nu}}$$



NMF-ANALYSE VON SPRACHE

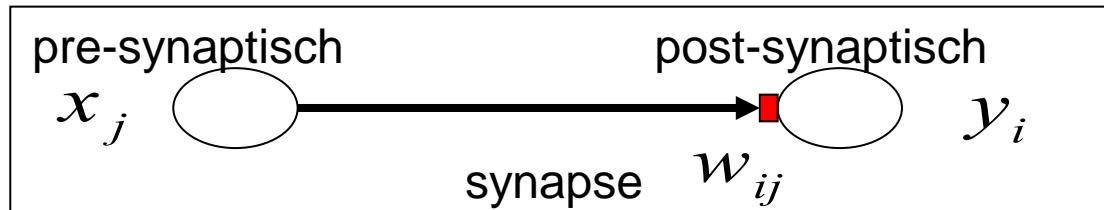


Fundamentalfrequenz

Formanten-Frequenz
Plosive

HEBB'SCHE LERNREGEL

Fire together – wire together



Hebb'sche Lernregel (1949) $\Delta w_{ij} = x_j y_i$

Gewichtsänderung: $w_{ij}^t = w_{ij}^{t-1} + \eta \Delta w_{ij}$

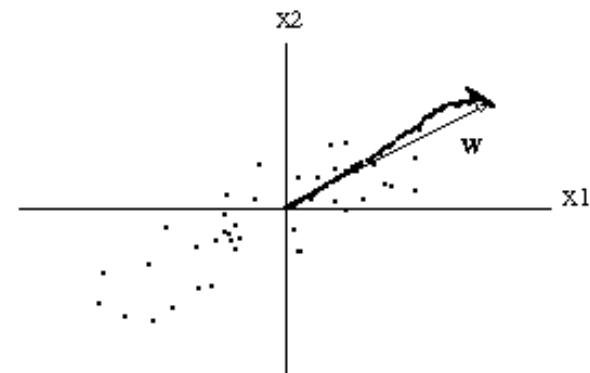
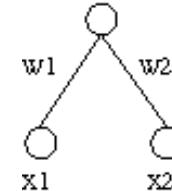
Lernrate
(in $[0,1]$)

Problém: Unbeschränktes Gewichtswachstum

OJA'S LERNREGEL

- Biologische Synapsen können nicht unbeschränkt wachsen
- Selbsthemmender Term von Oja (1982) vorgeschlagen

$$\Delta w_j = \eta y(x_j - w_j y)$$

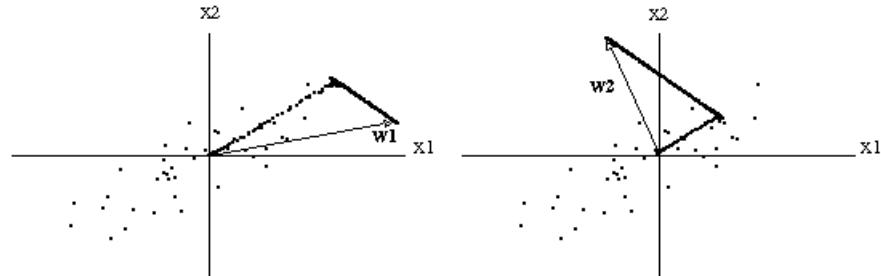


- Der Gewichtsvektor findet die Richtung der maximalen Varianz der Eingabe-Verteilung.

HAUPTKOMPONENTENANALYSE (PCA)

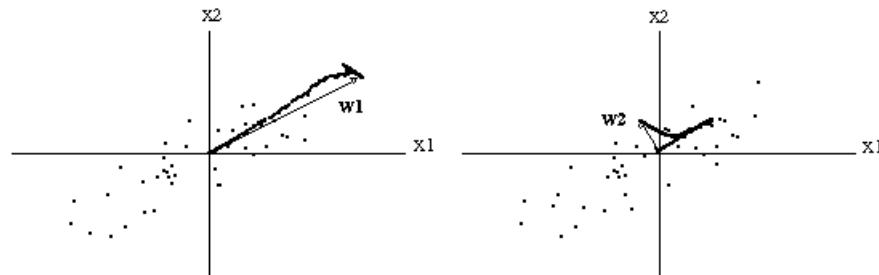
- Die **Oja-Regel** führt zu Gewichten, die den Unterraum der N Hauptachsen der Eingabeverteilung aufspannen

$$\Delta w_{ij} = \eta y_i \left(x_j - \sum_{k=1}^N w_{kj} y_k \right)$$



- **Sanger-Regel** führt zu Gewichten, die N Hauptachsen der Eingabeverteilung entsprechen

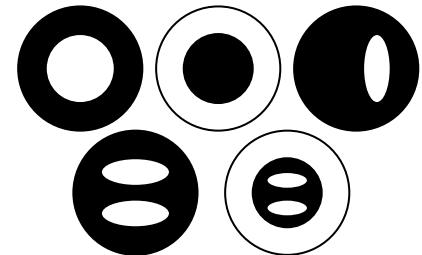
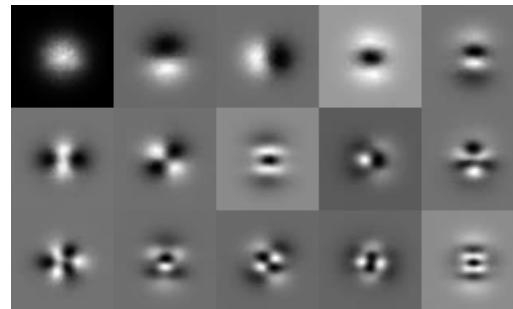
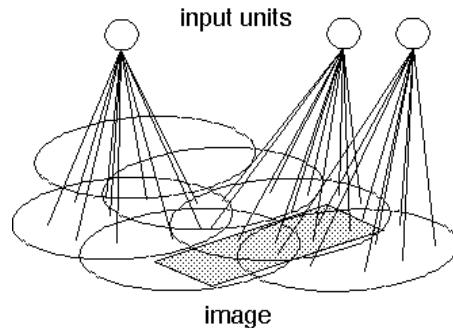
$$\Delta w_{ij} = \eta y_i \left(x_j - \sum_{k=1}^i w_{kj} y_k \right)$$



Nützlich für Merkmalsextraktion und Dimensionsreduktion

HAUPTKOMPONENTEN VISUELLER REIZE

- Aktivierende Reize entsprechen synaptischem Muster
- Gelernte Hauptkomponenten [Hancock et al., 1992]

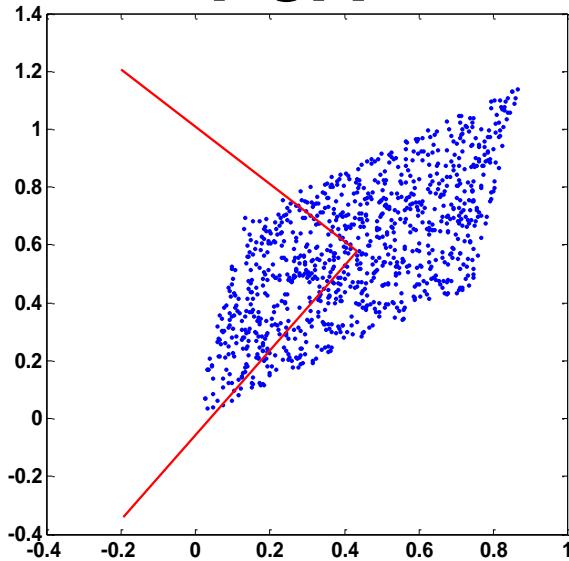


- Probleme:
 - a) PCA kann (im Gegensatz zu manchen Neuronen)
Ortsfrequenz nicht messen, da nichtlineare Operation erforderlich
 - b) Unabhängige Signalquellen können nicht separiert werden

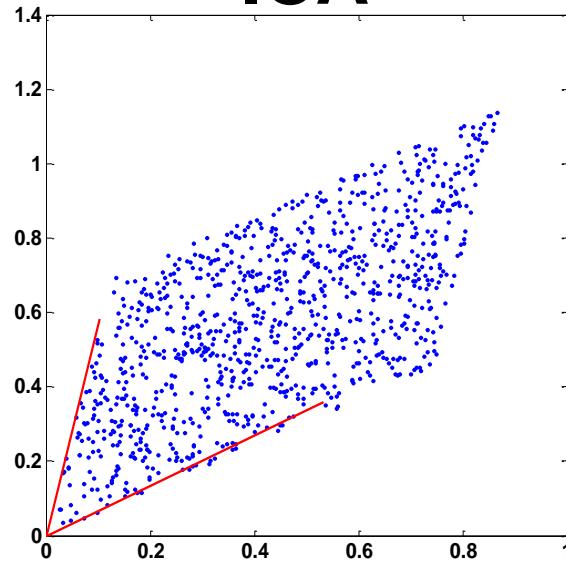
INDEPENDENT COMPONENT ANALYSIS (ICA)

- Lineare Transformation
- Minimiert Korrelation der Komponenten
- Erlaubt Separierung vom Mischungen (Blind Source Separation)

PCA



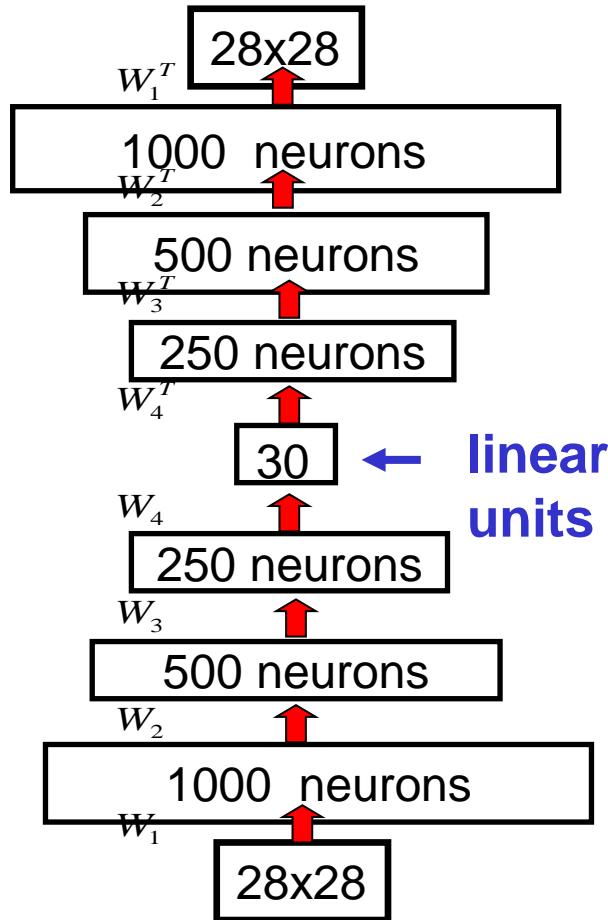
ICA



DEEP AUTOENCODERS

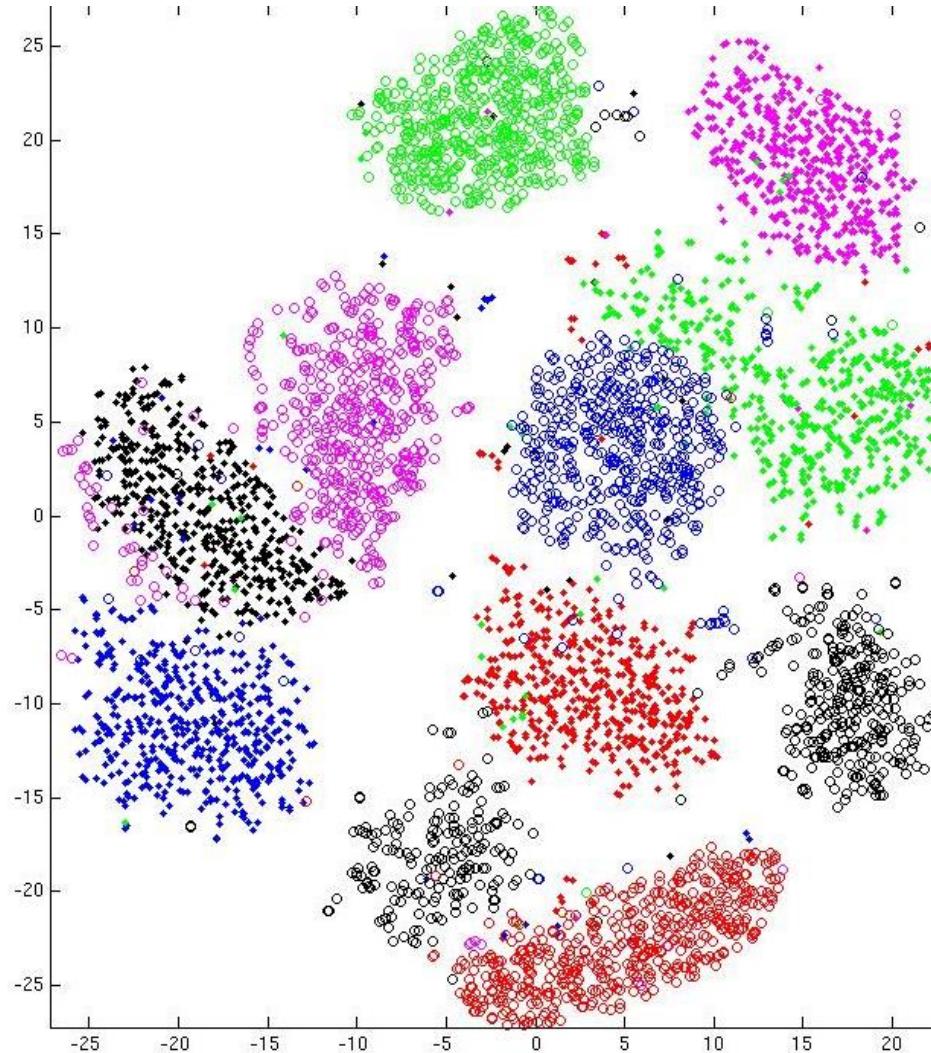
(HINTON & SALAKHUTDINOV, 2006)

- Mehrschichtiger Autoencoder für nichtlineare Dimensionsreduktion
- Schichtweises Training
- Überwachtes Finetuning des gesamten Netzwerks



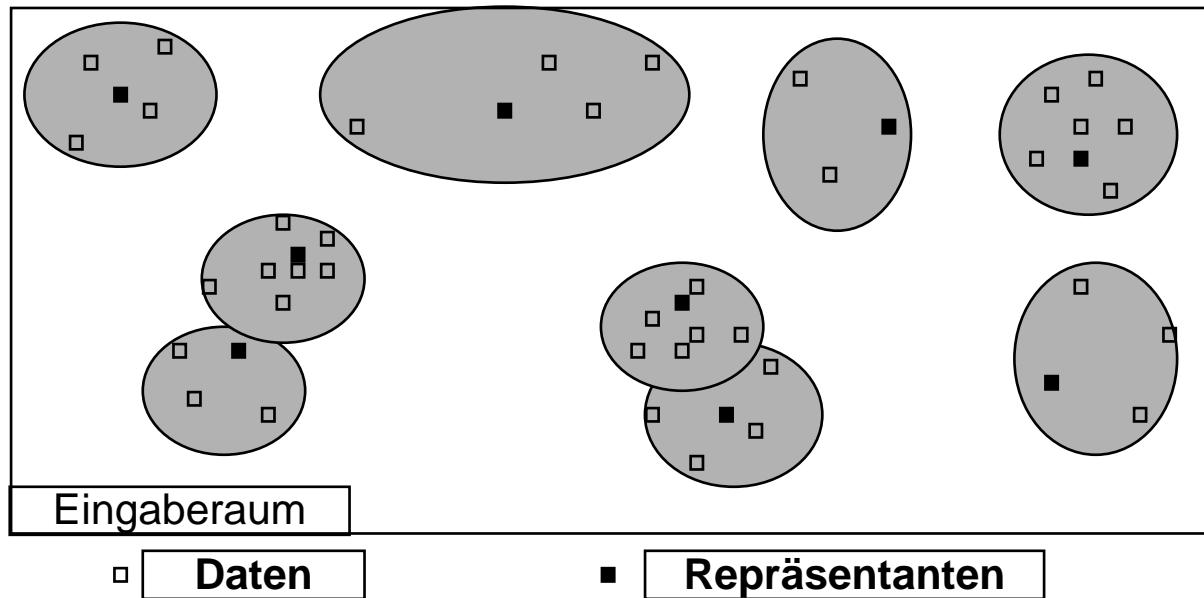
MNIST Ziffern

Unüberwacht
gelernte
Repräsentation



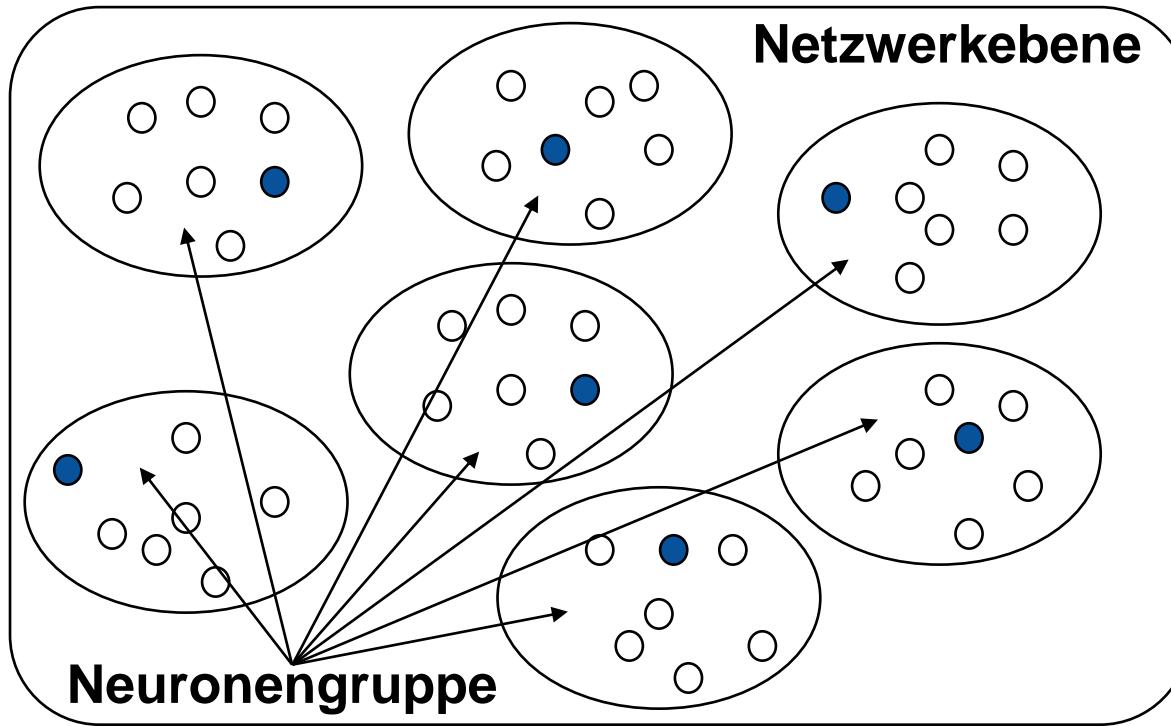
CLUSTERING / GRUPPIERUNG

- Datenvektoren sollen in Gruppen eingeteilt werden, die jeweils von einem Repräsentanten dargestellt werden



WINNER TAKE ALL - STRATEGIE

- Nur die Gewinner jeder Gruppe nehmen am Propagierungs- bzw. Lernvorgang teil



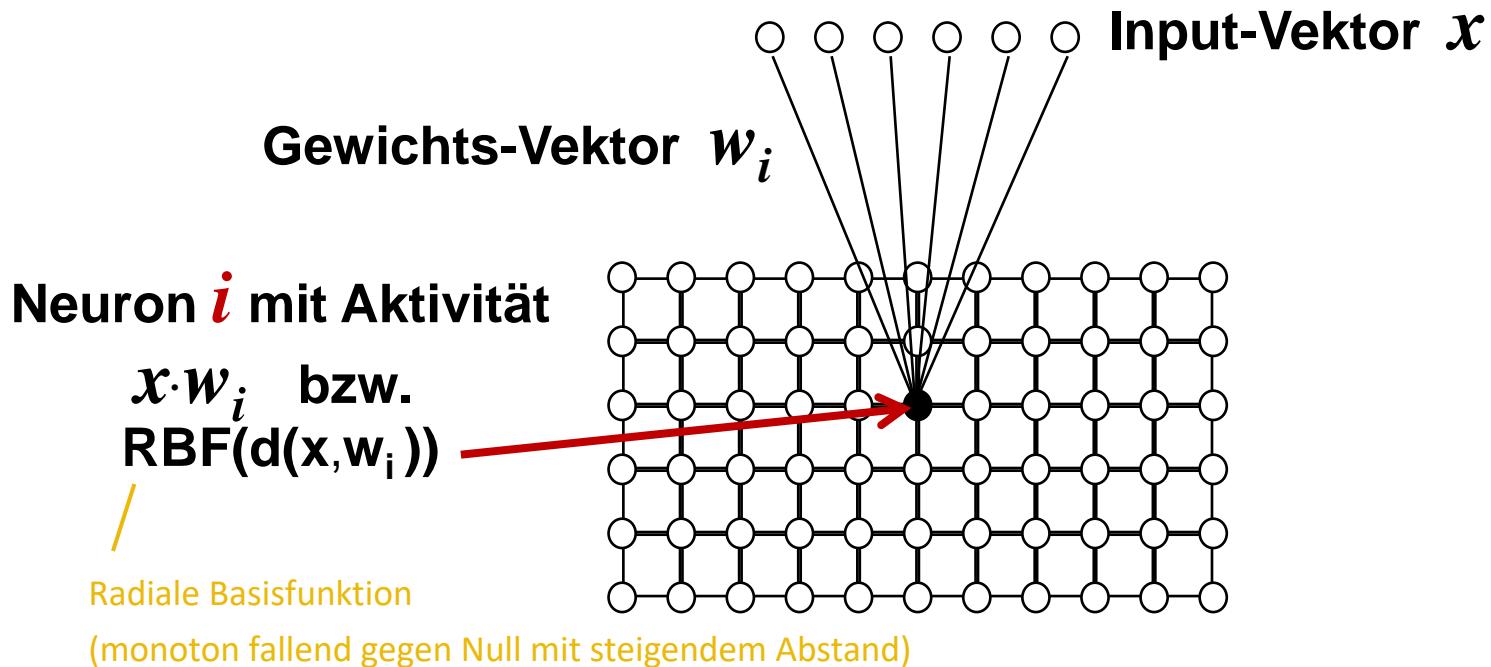
WINNER TAKE ALL - STRATEGIE

Um die Winner-take-all Strategie zu realisieren, kann man

- entweder das Neuron mit minimaler Distanz von der Eingabe (maximaler Aktivität) direkt berechnen, dazu benötigt man eine **Distanzfunktion** im Eingaberaum.
- oder alle Neuronen des Clusters untereinander mit **inhibitorischen** Synapsen verbinden, so dass im Ruhezustand nur noch der Gewinner aktiv sein kann.
- oder die Neuronen mit nach "**Abstand**" im Ausgaberaum variierender Stärke
 - inhibitorisch bei großem Abstand /
 - exitatorisch bei kleinem Abstanduntereinander verbinden.
Dabei bilden sich Aktivitäts-Blobs im Ausgaberaum heraus.

SIEGER-NEURON

- Sieger ist das Neuron mit Gewichtsvektor (=Repräsentant) mit der kleinsten Distanz (bzw. maximalem Skalarprodukt, Kreuzkorrelation,...) zum Eingabevektor
- Das Skalarprodukt ist (normiert) der Cosinus des Winkels zwischen den beiden Vektoren und umso größer, je kleiner der Winkel ist



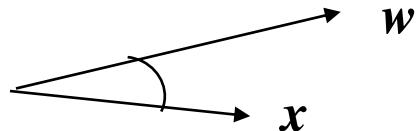
DISTANZFUNKTIONEN

- Im Eingaberaum können sehr verschiedene Distanzen verwendet werden, um die maximale Ähnlichkeit zwischen Input x und Repräsentant w zu ermitteln, z.B.:

- minimale Euklidische Distanz $|x-w|_2 = \sqrt{(x-w)^2}$

- minimale Manhatten-Distanz $|x-w|_1 = \sum_i |x_i - w_i|$

- maximales Skalarprodukt $x \cdot w$



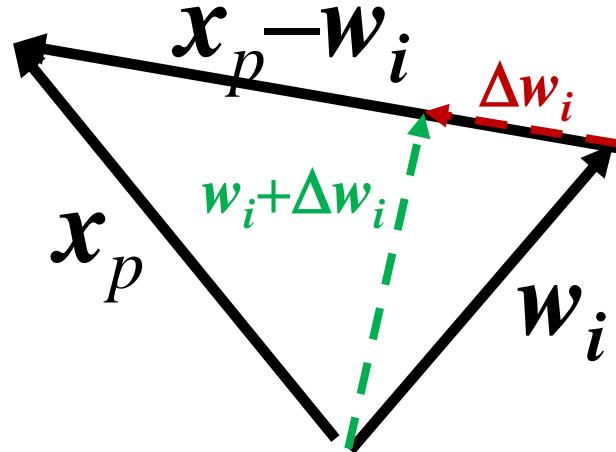
da hier die Länge der Vektoren x und w eine bedeutende Rolle spielt, kann dieses Maß nur dann gut funktionieren, wenn alle Repräsentanten von etwa derselben Größenordnung sind, denn dann misst das Skalarprodukt i.w. den Winkel zwischen Input und Repräsentant, denn $x \cdot w = |x| \cdot |w| \cdot \cos(x, w)$.

Es wird deshalb empfohlen, bei dem Einsatz des Skalarproduktes mit jeder Gewichtsanpassung auch eine Normierung des Gewichtsvektors (z.B. auf 1) vorzunehmen.

KOMPETITIVE LERNREGEL

$$\Delta w_i = \eta(x_p - w_i)$$

- Diese Regel bewegt den Gewichtsvektor w_i ein Stück auf den Eingabevektor x_p zu.
- Die Gewichtsvektoren stellen sich deshalb auf den Schwerpunkt der Gruppe von Eingabevektoren ein, die diesen Repräsentanten als Sieger i ansprechen.



KOMPETITIVE LERNREGEL

Satz:

Die Lernregel $\Delta w_i = \eta(x_p - w_i)$ minimiert den quadratischen Fehler:

$$E = \frac{1}{2} \sum_p (x_p - w_i)^2$$

für Eingaben x_p mit Sieger-Neuronen i .

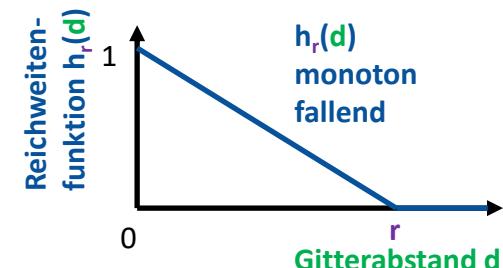
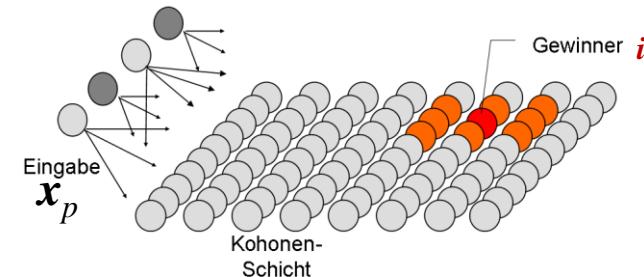
Die Lernregel beschreibt gerade den Gradienten-Abstieg in der Landschaft der Distanzen zu den Repräsentanten (Fehlerlandschaft).

KOHONEN-KARTEN (SELF-ORGANIZING MAP, SOM)

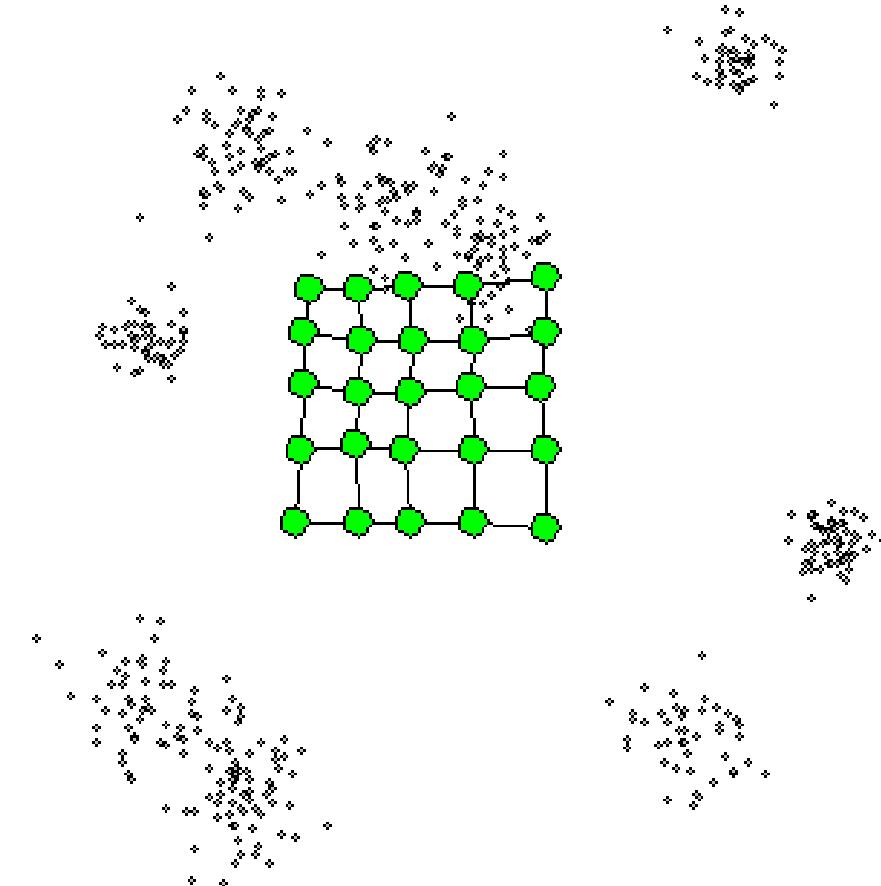
- Die SOM ist ein Zwei-Schichten-Netz, dessen Ausgabeschicht in Rasterform angeordnet ist. (Rechteck- oder Waben-Form)
- Die Gewichtsvektoren w_k werden zufällig initialisiert und als Repräsentanten im Eingaberaum interpretiert
- Es werden nun mehrfach zufällig Eingaben x_p ausgewählt, zu denen anhand einer Distanzfunktion $|x_p - w_k|_2$ im Eingaberaum das Sieger-Neuron i mit der geringsten Distanz $|x_p - w_i|_2$ bestimmt wird
- Alle Gewichte w_k werden in Richtung von x_p gezogen, wobei die Updaterate mit Gitterabstand zum Sieger-Neuron i abnimmt:

$$\Delta w_k = \eta \cdot (x_p - w_k) \cdot h_r(d(i, k)), \text{ mit}$$

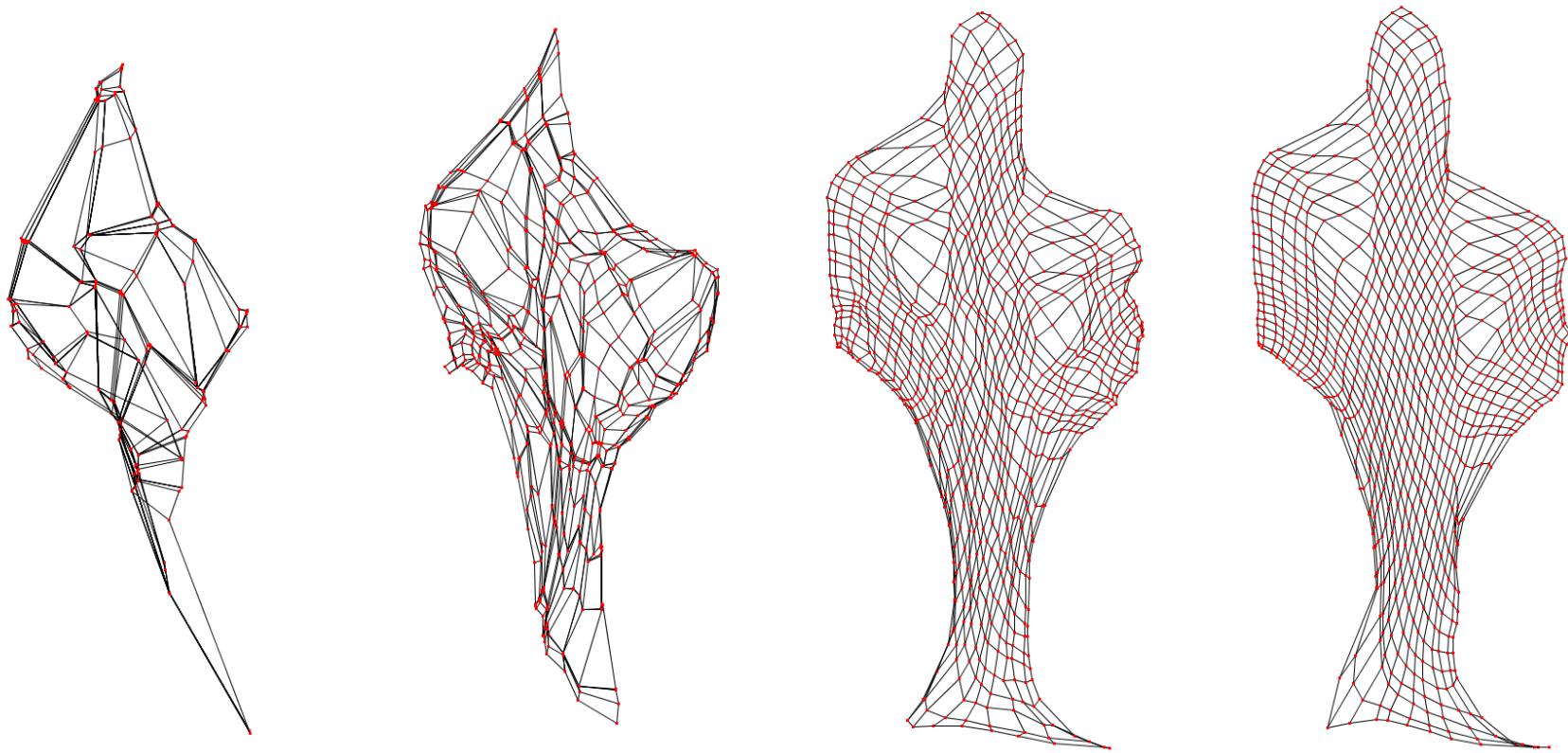
- Lernrate η
- Abstandsmaß $d(i, k)$ im Neuronengitter
- Reichweitenfunktion $h_r(d)$: Gitterabstand \Rightarrow Updaterate
- Reichweite r im Neuronengitter (Zu Beginn des Lernens groß, später kleiner)



SOM-ANIMATION

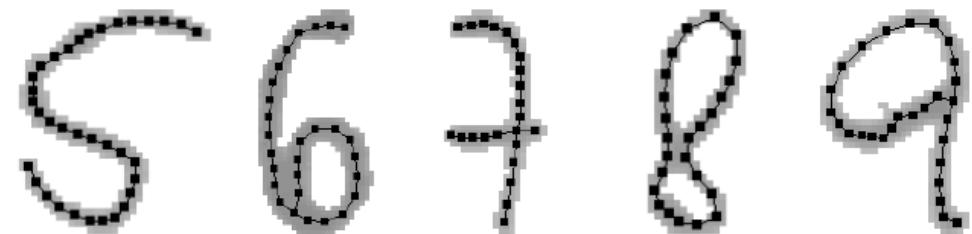
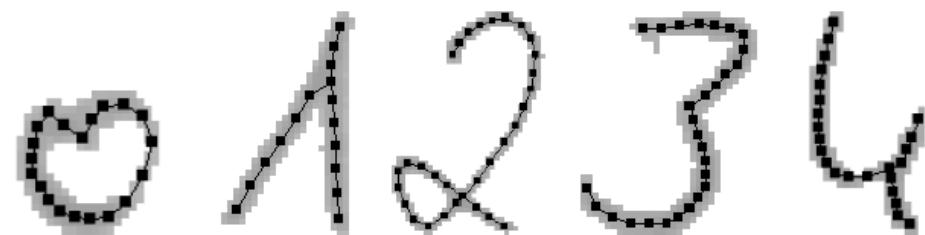
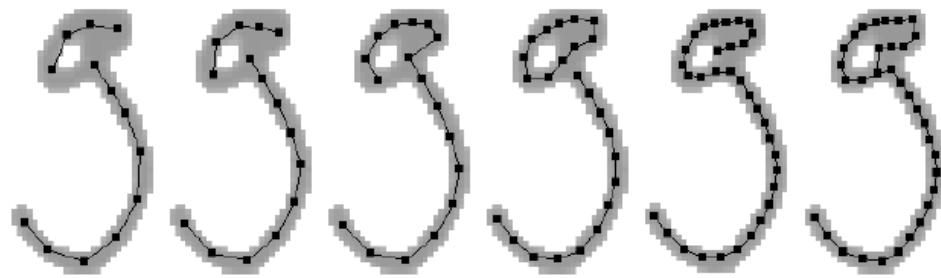


2D-SOM MIT 2D-EINGABE



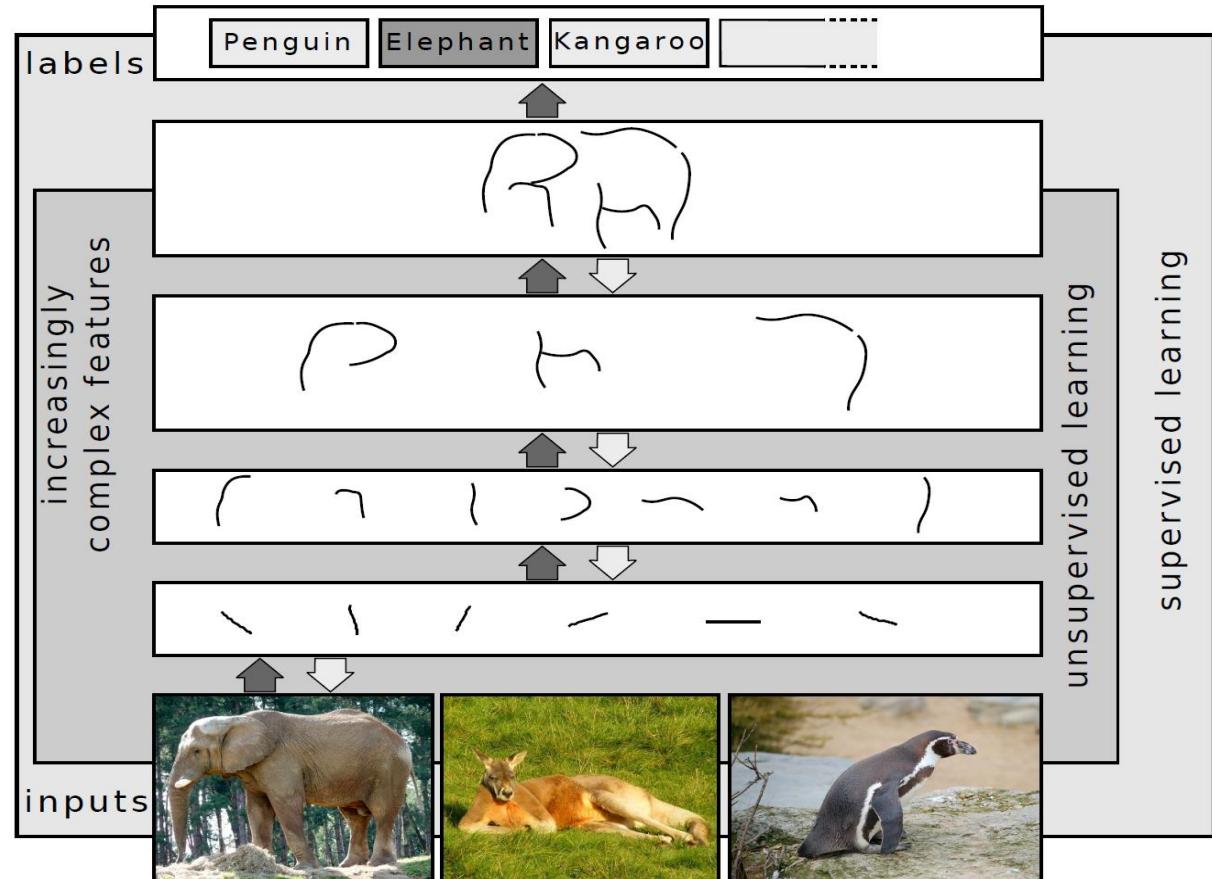
SOM HANDEGESCHRIEBENER ZIFFERN

EINPASSEN VON 1D-SOM IN ZIFFERN



DEEP LEARNING

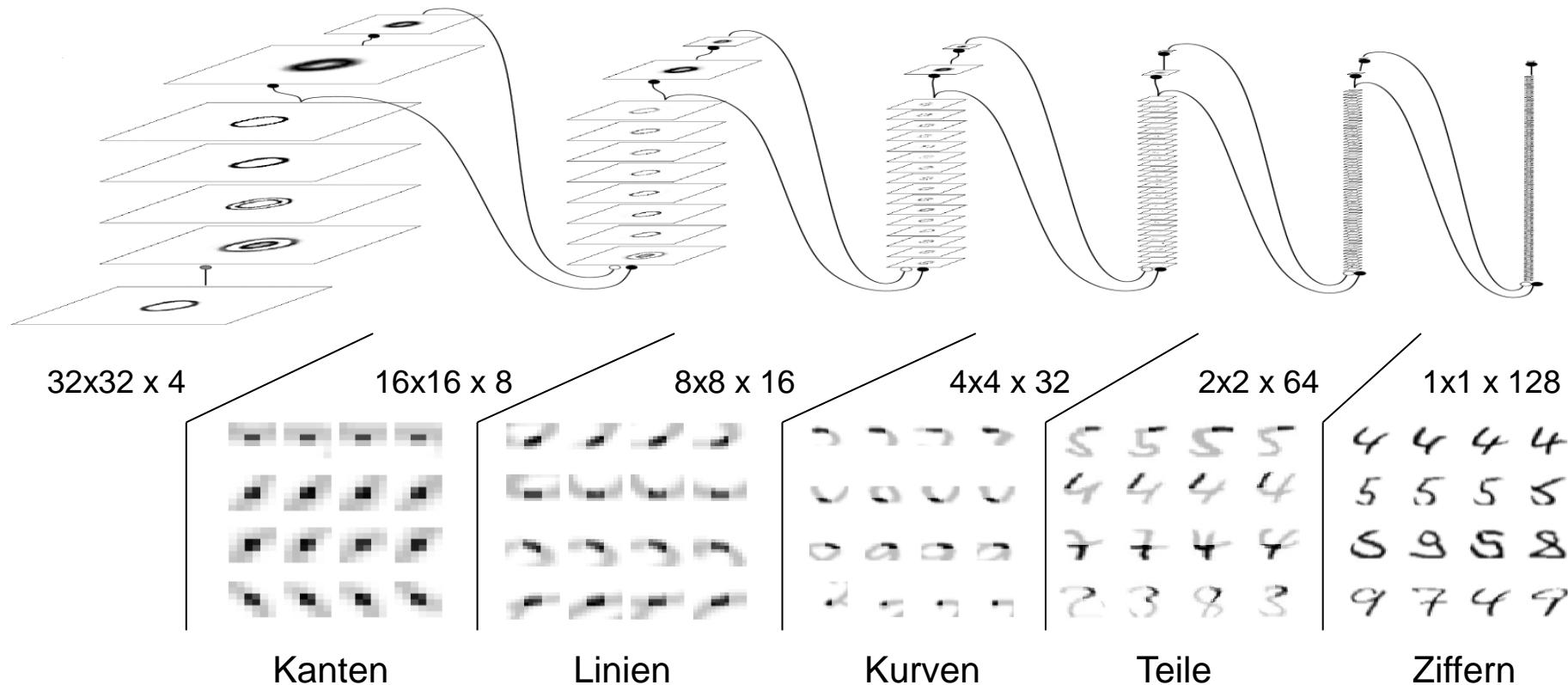
- Lernen hierarchischer Repräsentationen
- Höhere Konzepte durch Kombination niederer Konzepte definiert
- Wiederverwendung von niederen Konzepten in mehrern höheren Konzepten



LERNEN VON MERKMALSHIERARCHIEN

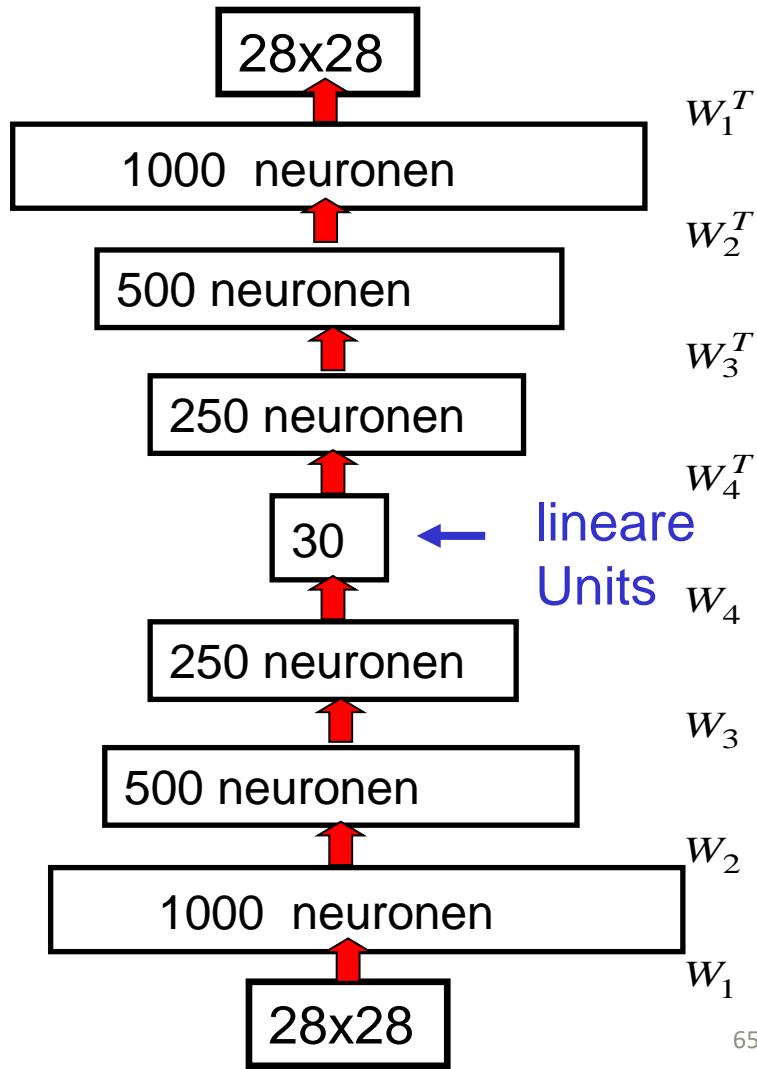
■ Hebbisches Lernen und Wettbewerb

[Behnke, IJCNN1999]

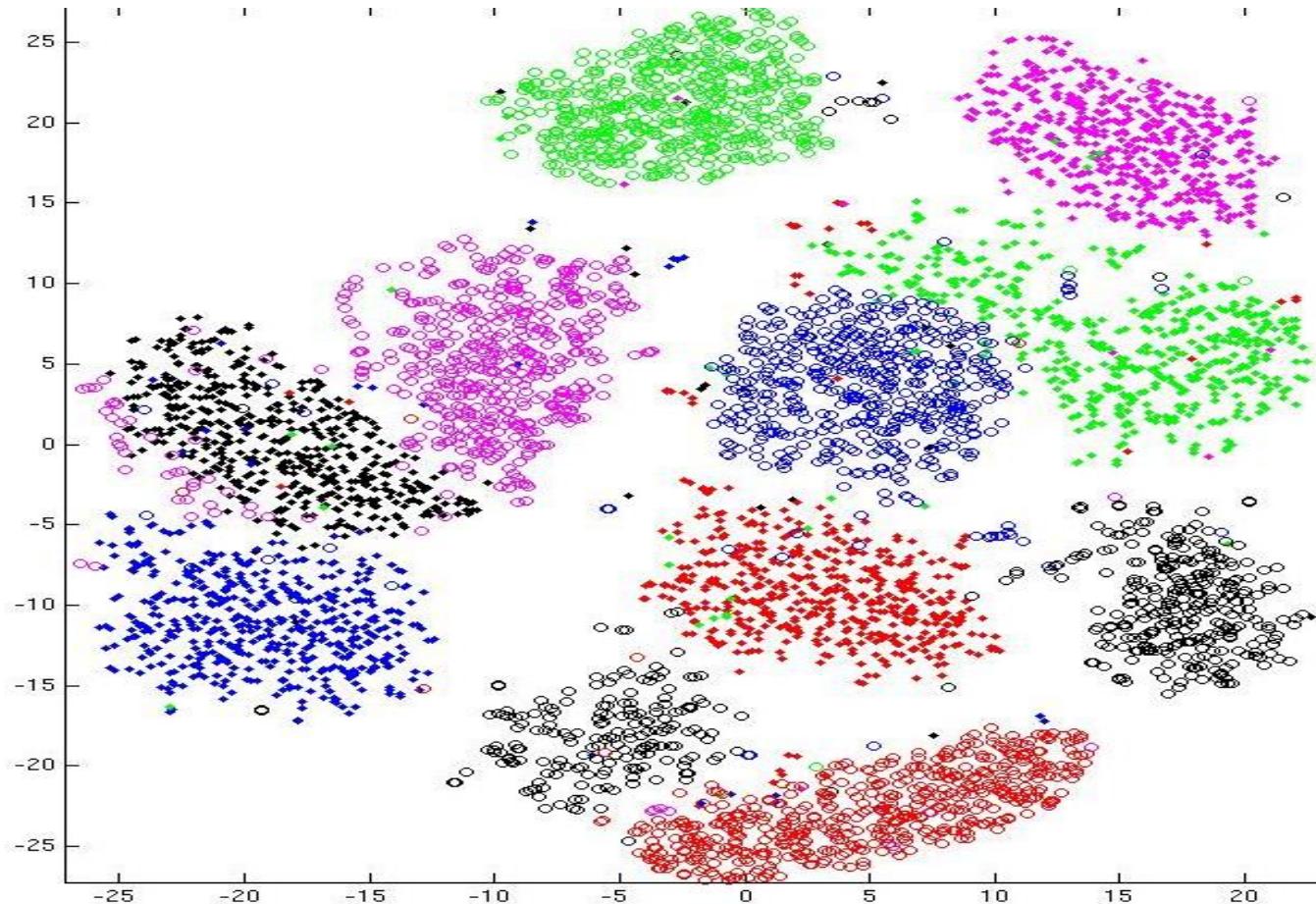


TIEFE AUTOENCODER (HINTON & SALAKHUTDINOV, 2006)

- Trainiert durch schichtweises Lernen
- Zunächst unüberwachtes Training von Boltzmann-Maschinen ohne seitliche Verbindungen (RBM, Contrastive Divergence)
- Danach Finetuning durch Backpropagation

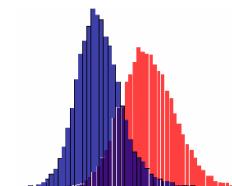


2D-REPRÄSENTATION VON MNIST-ZIFFERN

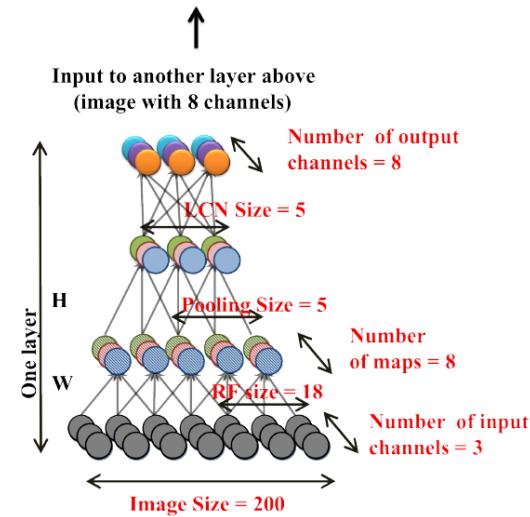


UNÜBERWACHTES LERNEN VON BILDMERKMALEN

- 9 Schichten
- Lokale Verbindungen
- Spärlicher Autoencoder
- L2-Pooling
- Lokale Kontrastnormalisierung
- 1 Milliarde Verbindungen
- Trainiert auf 10 Millionen Bildern
- Unüberwacht gelernte Merkmale:



3x



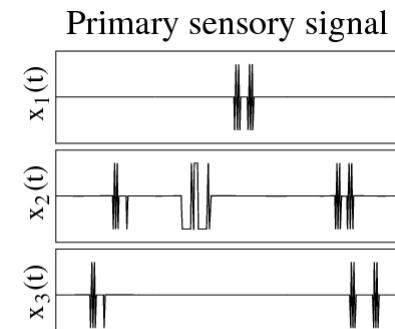
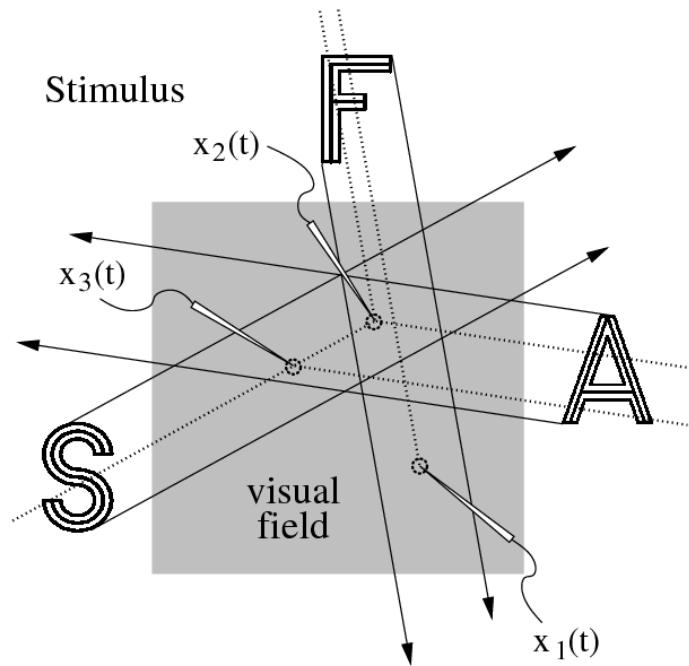
- Überwachtes Training auf ImageNet 2011 (14M Bilder, 22K Kategorien): 15.8%

[Le et al. 2012]

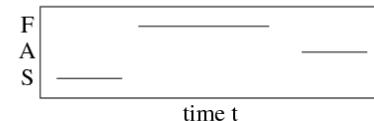
SLOW FEATURE ANALYSIS (SFA) [WISKOTT]

Motivation:

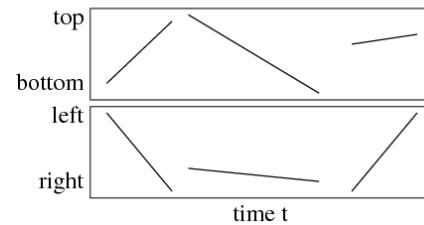
- Einzelne Bildpunkte variieren stark
- Objektidentität und Position variieren langsam



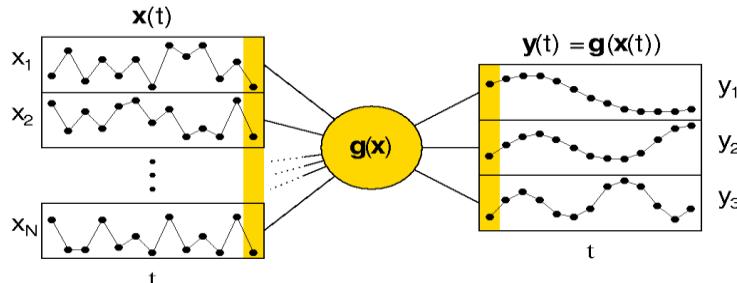
Object identity



Object 2D-location



SLOW FEATURE ANALYSIS - PROBLEM



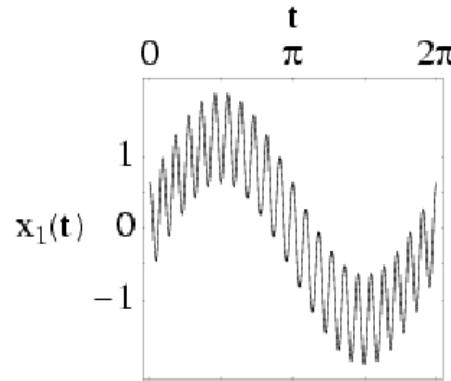
- Gegeben ein (hochdimensionales) Eingabe-Signal $x(t)$.
Finde Funktionen $g_i(x)$, sodass Ausgabe $y_j(t) := g_j(x(t))$ sich möglichst langsam ändert.

Minimiere $\Delta(y_j) := \left\langle \dot{y}_j^2 \right\rangle_t$

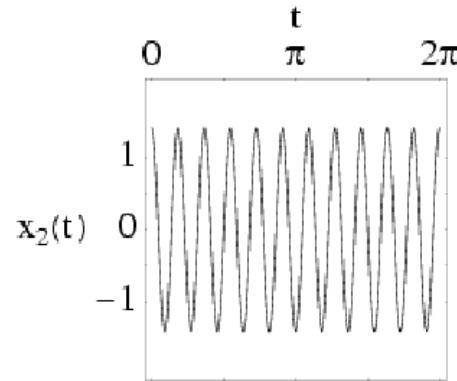
unter den **Nebenbedingungen**

- $\left\langle y_j \right\rangle_t = 0$ (Mittelwert Null),
- $\left\langle y_j^2 \right\rangle_t = 1$ (normierte Varianz),
- $\forall i < j: \left\langle y_i y_j \right\rangle_t = 0$ (Dekorrelation und Reihenfolge).

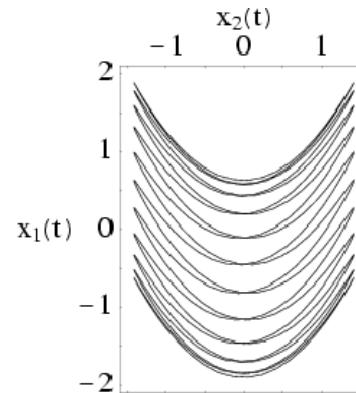
SLOW FEATURES - BEISPIEL



input component $x_1(t)$



input component $x_2(t)$



input trajectory $\mathbf{x}(t)$

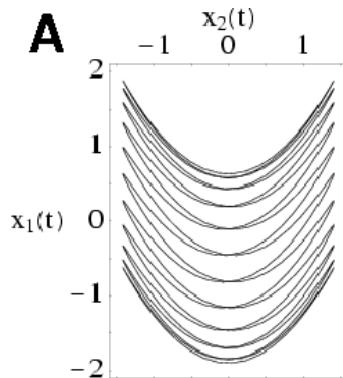
$$x_1(t) := \sin(t) + \cos(11t)^2$$

$$x_2(t) := \cos(11t)$$

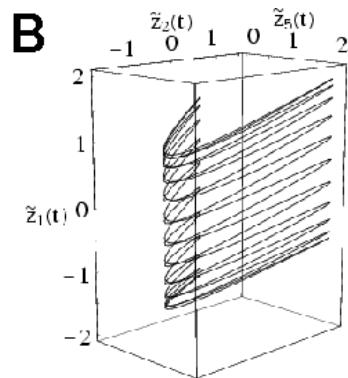
■ Langsames Merkmal: $y(t) = x_1(t) - x_2(t)^2 = \sin(t)$

■ Extrahiert mit Polynom 2. Grades: $g(\mathbf{x}) = x_1 - x_2^2$

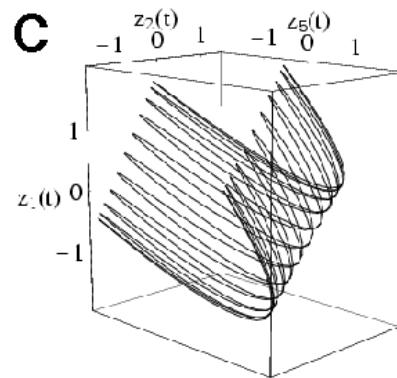
SFA-ALGORITHMUS



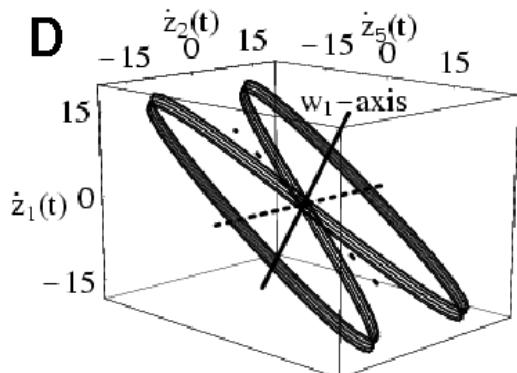
input signal $\mathbf{x}(t)$



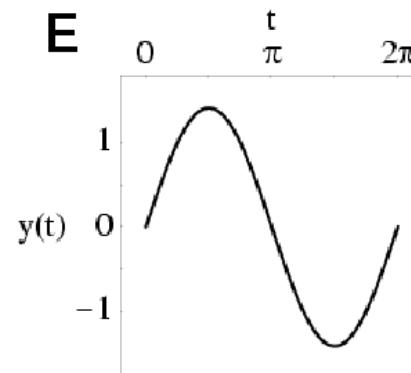
expanded signal $\tilde{\mathbf{z}}(t)$



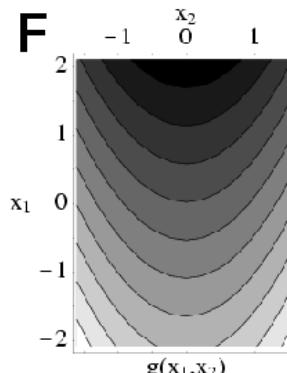
sphered signal $\mathbf{z}(t)$



time derivative signal $\dot{\mathbf{z}}(t)$



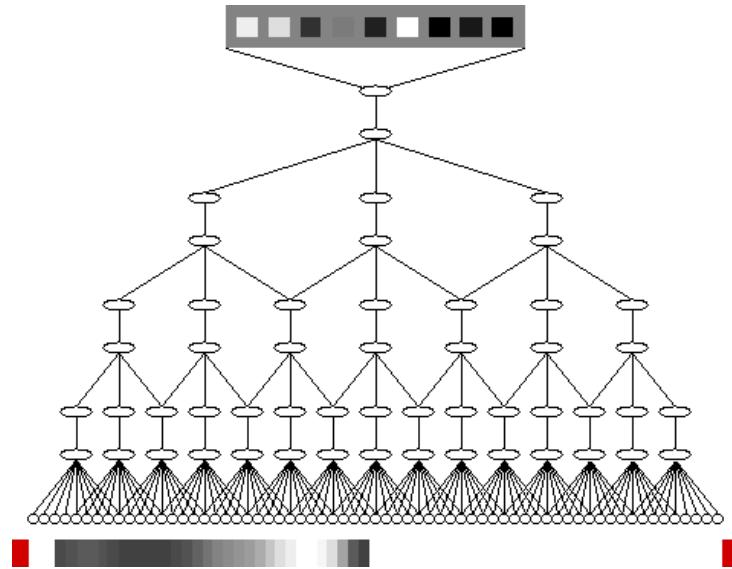
output signal $y(t)$



function $g(\mathbf{x})$

1. Signal expandieren
(z.B. in Polynome
2. Grades)
-> Lineares Problem
2. Signal normalisieren
(Whitening) (Mittelwert
abziehen, PCA)
3. Zeitliche Ableitung
berechnen $\dot{\mathbf{z}}(t)$
4. Finde die Richtungen
mit den kleinsten
Änderungen (PCA mit
kleinsten Eigenwerten)

HIERARCHISCHE SFA



- Verschiedene 1D-Signale
- Trainiert mit Transformationen

■ Untrainiert:

naive network



■ Translationsinvarianz:

trained network
training data



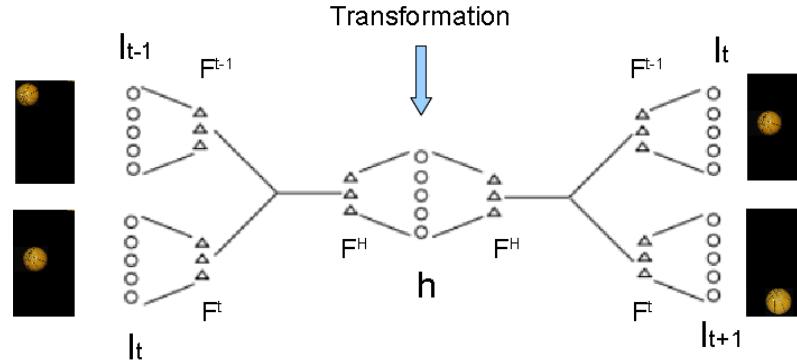
■ Skalierungsinvarianz:

size invariance
trained network
training data



RELATIONALER AUTOENCODER

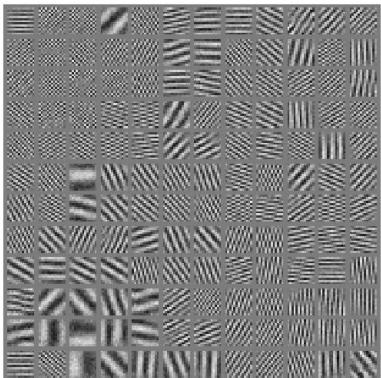
■ Lerne Bildtransformation:



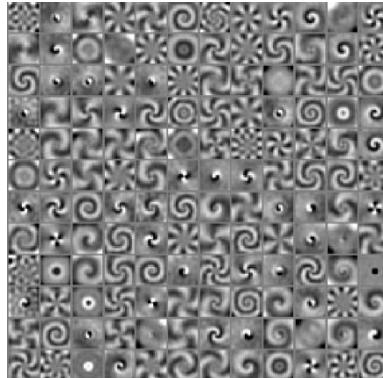
$$h_k(I_{t-1}, I_t) = \sigma\left(\sum_f F_{kf}^H \sum_i F_{if}^{I_{t-1}} I_{t-1,i} \sum_i F_{jf}^{I_t} I_{t,j}\right)$$

$$\hat{I}_{t+1,i}(h_k(I_{t-1}, I_t)) = \sum_f F_{jf}^{I_t} \sum_i F_{if}^{I_{t-1}} I_{t,i} \sum_k F_{kf}^H h_k$$

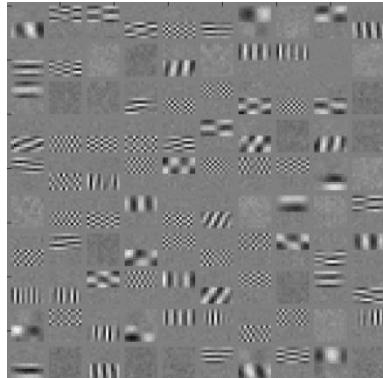
■ Gelernte Basisfunktionen:



Verschiebungen



Drehungen



Zwei Verschiebungen

[Memisevic, PAMI 2013]