Einführung in die Automatisierungstechnik

Studiengang: Produktionstechnik, Systems Engineering

\- Vorlesung 06 -

Prof. Dr.-Ing. habil. Andreas Fischer
Dr.-Ing. Gerald Ströbel

Bremer Institut für Messtechnik, Automatisierung und Qualitätswissenschaft

Lehrziele und Gliederung

V1 Motivation, Anwendungsbereiche, Prozesse und Methoden der Automatisierungstechnik

V2 Automatisierung in der Produktion

V3 Boolesche Algebra 1

Ü1 Matlab Einführung

V4 Bolsche Algebra 2: Graphen

Ü2 Übung Boolsche Algebra

V5 Fuzzy Logic

Ü3 Fuzzy Logic

V6 **Neuronale Netze**

Ü4 Neuronale Netze

V7 Automatisiertes Messen und Steuern

Ü5 Automatisiertes Messen und Steuern

V8 Speicherprogrammierbare Steuerungen

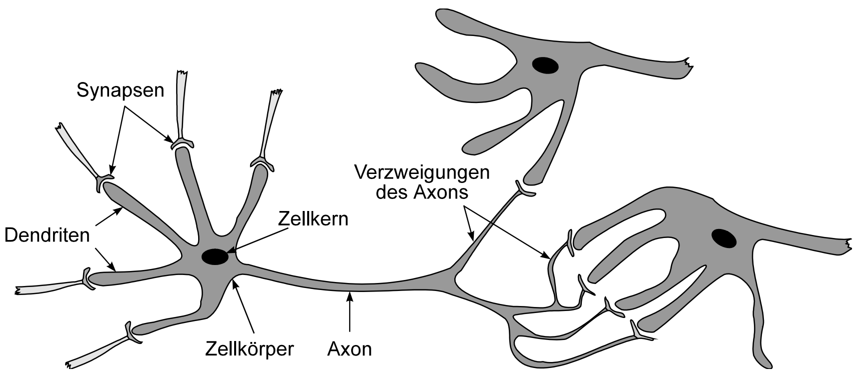
Ü6 Übungen und Musterklausuren

Künstliche Neuronale Netze

- Historie -

auch „Künstliche Neuronale Netze" oder (KNN)

Historie - Hintergrund
Grundlagen KNN
Anwendungen



Künstliche Neuronale Netze
- Historie -

Die Realisierung von KNN greift auf Arbeiten von MacCulloch / Pitts um das Jahr 1940 zurück - Beschreibung von logischen Schwellwertelementen

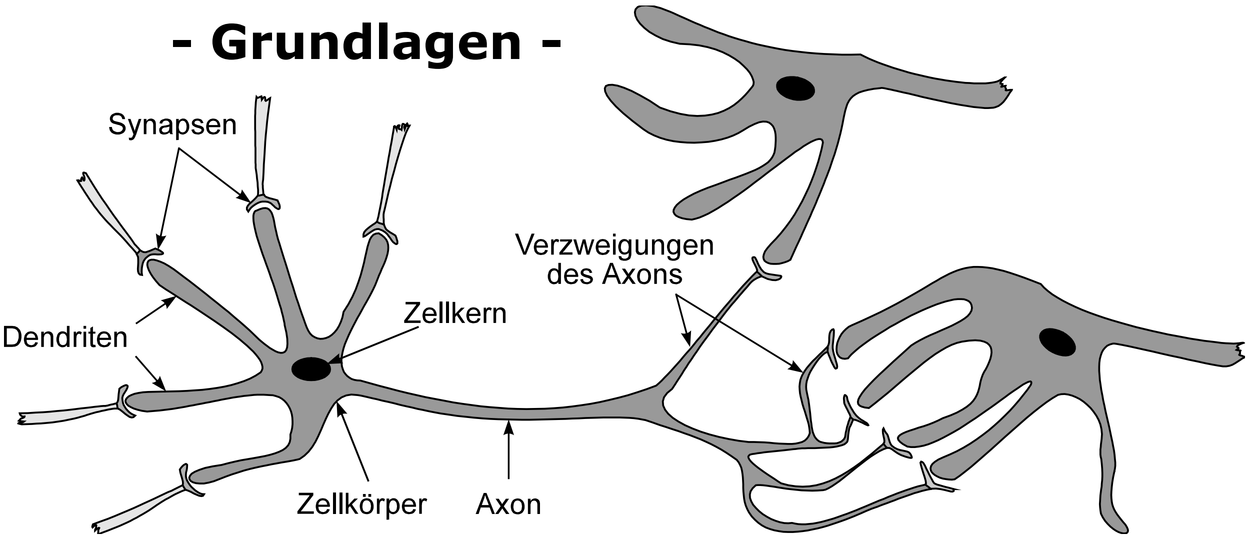
Mit künstlichen neuronalen Netzen wird erstmals versucht, bestimmte Fähigkeiten biologischer Systeme nachzuahmen, das sind Lernfähigkeit und eigenständige Wahl und Gewichtung der neuronalen Verbildungen

Einige Urgesteine des Fachbereichs der Neuronalen Netze. Von links nach rechts: John von Neumann, Donald O. Hebb, Marvin Minsky, Bernard Widrow, Seymour Papert, Teuvo Kohonen, John Hopfield, "in order of appearance" [Krie05]

Künstliche Neuronale Netze

- Grundlagen -

Schematische Darstellung einer Nervenzelle (nach Bruns 1990, S. 80)



System aus Nervenzellen (Neuronen) und **Verbindungen** (Synapsen)
Die Struktur eines neuronalen Netzes ist ein **Graph mit Knoten und Kanten**
Lernverfahren bestimmen, wie die am Neuron ankommenden Signale behandelt werden und welche Bedeutung die einzelnen Verbindungen haben
Technisch: System mit Eingängen/ Übertragungsfunktionen und Ausgängen

Eingaben → KNN → Ausgaben

Künstliche Neuronale Netze
- Grundlagen -

	Gehirn	Computer
Anzahl Recheneinheiten	$\approx 10^{11}$	$\approx 10^{9}$
Art Recheneinheiten	Neurone	Transistoren
Art der Berechnung	massiv parallel	i.d.R. seriell
Datenspeicherung (assoziativ bzw. adressbasiert): Schaltzeit	$\approx 10^{-3}$ s	$\approx 10^{-9}$ s
Theoretische Schaltvorgänge	$\approx 10^{13}$ 1/s	$\approx 10^{18}$ 1/s
Tatsächliche Schaltvorgänge	$\approx 10^{12}$ 1/s	$\approx 10^{10}$ 1/s

Tabelle 1.1: Der (hinkende) Vergleich zwischen Gehirn und Rechner auf einen Blick. Vorlage: [Zel94] in D. Kriesel, NN,

- **Theorie**: Ein Graph mit Knoten (Neuronen) und Kanten (Synapsen)
- **Synapsen** (Kanten) = gerichtete Signalverbindungen (die Strukturen mit netzartige Verbindungen erlauben)
- **Synonym** zu "künstliches Neuronales Netz" = der Begriff **Konnektionismus**

Eingaben → KNN → Ausgaben

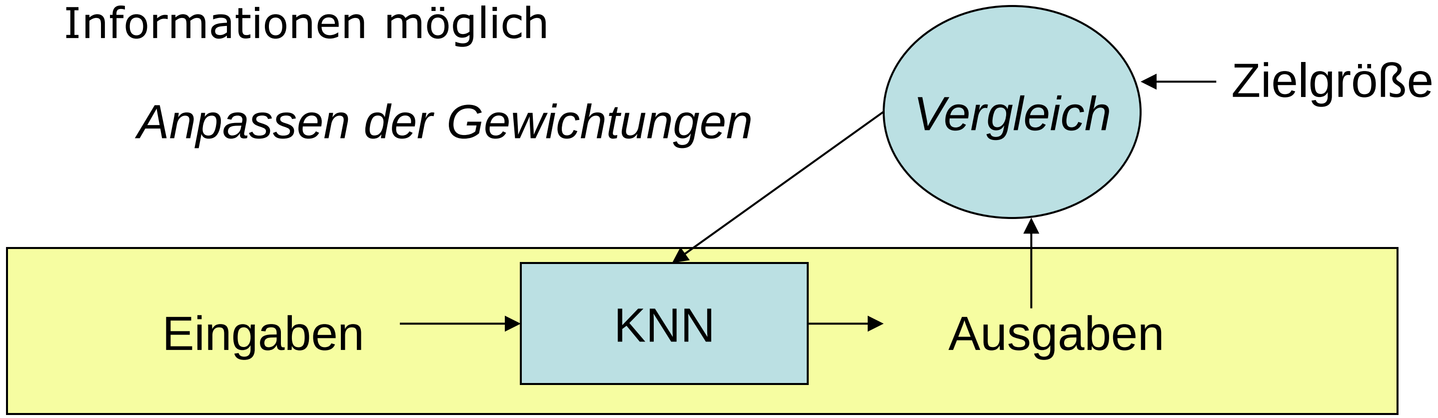
Künstliche Neuronale Netze

- Einsatzgebiete -

Erfolgreiche Einsatzgebiete sind:

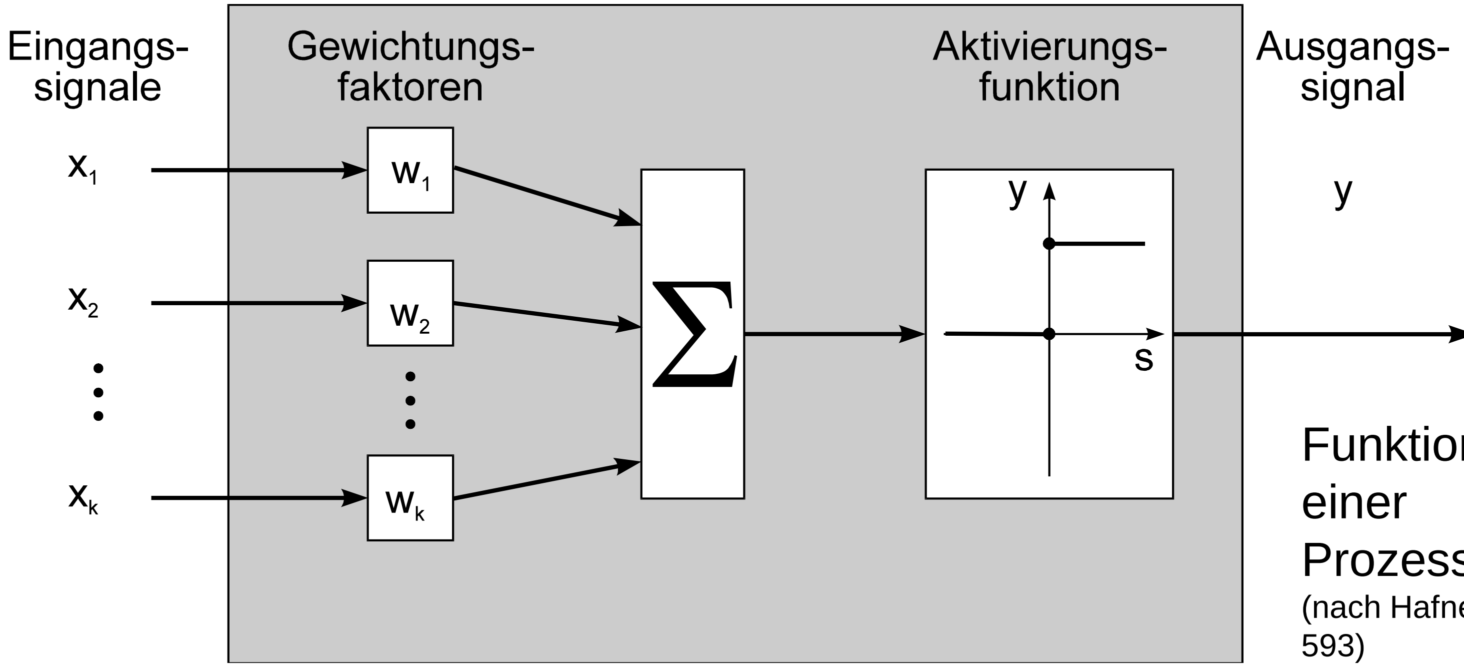
- Modellbildung (trainieren)
- Entscheidungsfindung
- Regelung nichtlinearer Prozesse (Prozess-, Qualitätsregelung)
- Steuerung von Produktionseinrichtungen
- Robotersteuerung (allg. Steuern von Bewegungen)
- Bildverarbeitung (Mustererkennung)
- Technische Diagnose (Mustererkennung, Vorhersage, Prognose)

Vorteil: auch die Verarbeitung fehlerhafter und/oder unvollständiger Informationen möglich



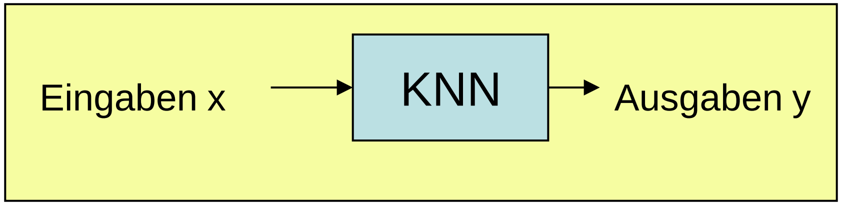
Künstliche Neuronale Netze

- Technische Realisierung -



Funktionsweise einer Prozesseinheit
(nach Hafner u. a. 1992, S. 593)

Eingangssignale $x_1,\ldots, x_k$,
Gewichtungsfaktoren $w_1,\ldots, w_k$
Addition (oder auch Produktbildung)
Aktivierungsfunktion s
Ausgangssignal y



Künstliche Neuronale Netze

- Historie -

Das Ausgangssignal *y* einer PE kann nun folgendermaßen berechnet werden:

$$y = f\left(\sum_{k=1}^{n}\left(w_k \cdot x_k\right)\right) \quad [1]$$

Dabei entspricht
x dem Eingangsvektor,
w dem Gewichtungsvektor und
f (s) der Aktivierungsfunktion

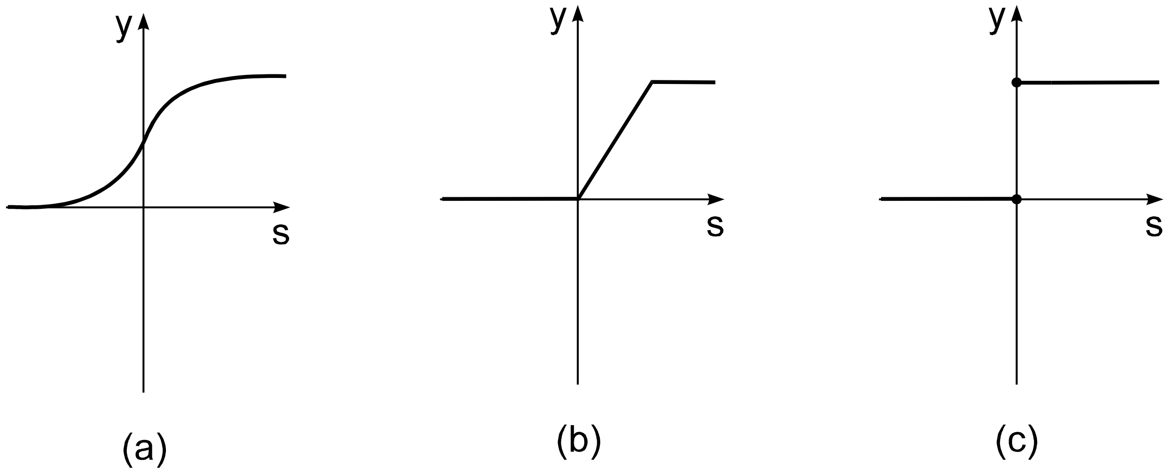
Ein neuronales Netz setzt sich aus mehreren PE zusammen Üblicherweise werden dazu verschiedene Schichten von PE untereinander vernetzt

Künstliche Neuronale Netze

- Aktivierungsfunktionen -

Aktivierungsfunktion

$y = f\,(s)$



Beispiele für Aktivierungs-
funktionen einer Prozesseinheit:
sigmoidale (a),
pseudo-lineare (b) und
treppenartige (c)

$$y = f(s)$$

Unter MATLAB

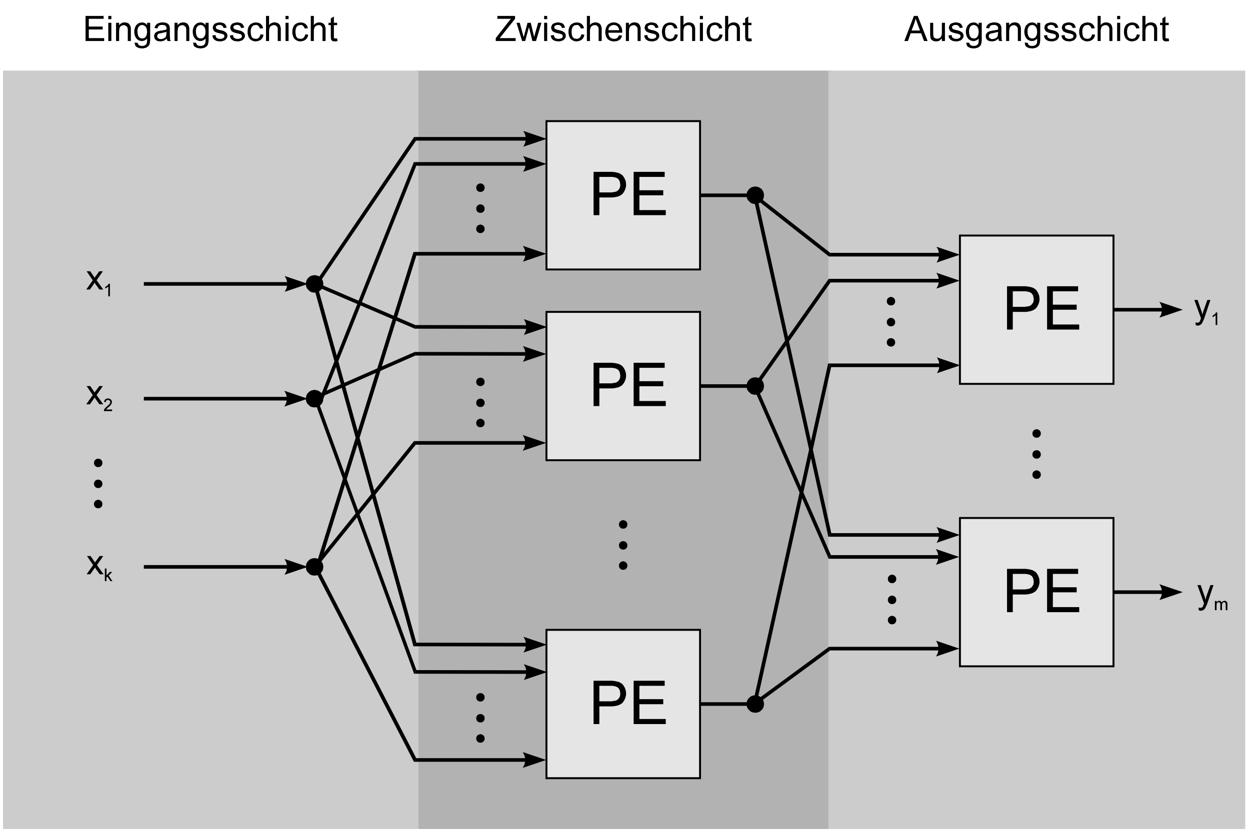
$a = f(n)$ wobei $n = p*w+b$

MATLAB:

a = softmax(n)	Compete transfer Function
a = hardlim(n)	Hard Limit Transfer Function
a = Hardlims(n)	Symmetric Hard Limit Func.
a = logsig(n)	Log-Sigmoid Transfer Funktion
a = poslin(n)	Positive Linear Transfer Func.
a = purelin(n)	Linear Transfer Function
a = radbase(n)	Radial Basis Function
a = satlin(n)	Satlin Transfer Function
a = satlins(n)	Symmetric Satlin Function

Softmaxs, Tan-Sigmoid, Triangular;

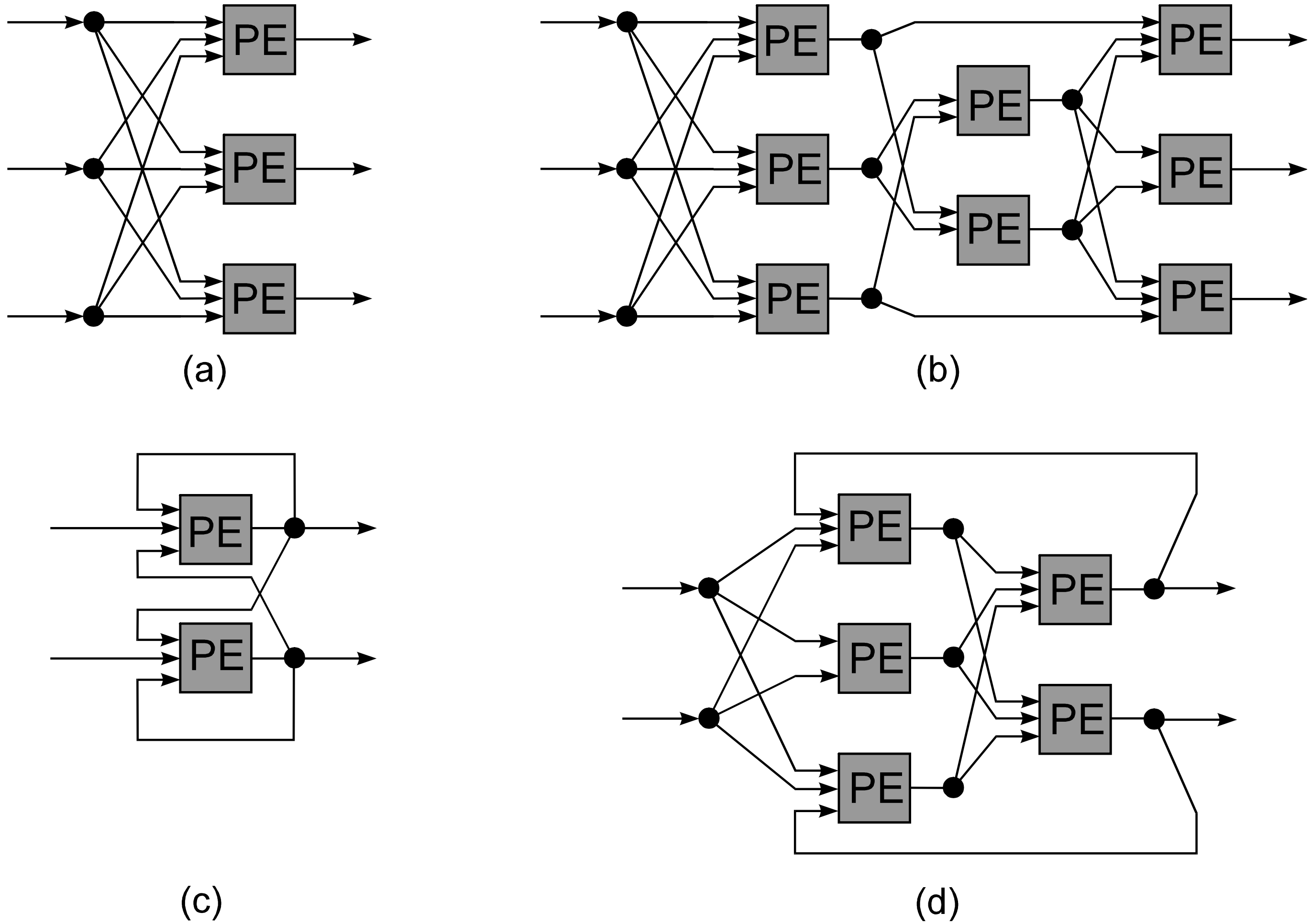
Künstliche Neuronale Netze
- Netzaufbau -



Vernetzung von mehreren Prozesseinheiten

Künstliche Neuronale Netze

- Netzstrukturen -



Verknüpfungsmöglichkeiten neuronaler Netze: (a) und (b) vorwärts gerichtet, (c) und (d) rückgekoppelt (Multi Layer Perceptron)

Künstliche Neuronale Netze

- Implementierung -

- Die Entscheidung, wie die einzelnen PE untereinander zu verknüpfen sind, ist problemabhängig.
- So lassen sich manche Probleme nur von bestimmten Netzwerktypen lösen.
- Die Wahl des Netzwerktyps, also die Bestimmung der PE und deren Verbindungen untereinander, beruht oftmals auf Erfahrungen und erfolgreichen früheren Einsatzes zur Lösung gleicher oder ähnlicher Probleme
- Anwendungsentwicklung = Prototyping -Test –Verifikation
- Anlernen des Netzes (Training)
- Nachteil: Trainingsdaten müssen vorliegen! Zusätzlich Testdaten notwendig

  Diese Daten können ev. in der Vergangenheit gesammelt worden sein, ein grundlegendes Verständnis der Zusammenhänge ist notwendig!
- Computerprogramme können diese Vorgehensweise optimal unterstützen

Künstliche Neuronale Netze

- Die Perzeptron Konvergenzregel -

1. Schritt: Initialisierung der Gewichte

Weise den Gewichten $w_k(0)$ $(0 <= k <= n)$ kleine Zufallswerte zu.

Mit $w_k(t)$ ist das Gewicht der Eingabe x_k zum Zeitpunkt t bzw. Lernschritt t gemeint (hier $t = 0$, also zum Zeitpunkt der Initialisierung).

2. Schritt: Präsentiere eine neue Eingabe und die gewünschte Ausgabe

Eine Eingabe wird dem Netz präsentiert, was für jedes x_k einen entsprechenden Wert bedeutet, und dazu die gewünschte Ausgabe $d(t)$.

3. Schritt: Berechne die eigentliche Ausgabe

Zu den gegebenen Werten wird die tatsächliche Ausgabe berechnet, die sich von der gewünschten Ausgabe unterscheiden kann. Die Ausgabe wird mit der bekannten Formel berechnet:

$$y(t) = f\left(\sum_{k=1}^{n} \left(w_k(t) \cdot x_k(t) \right) \right)$$

Dabei ist im Vergleich zu Gleichung 1 nur der Zeitpunkt t als zusätzliche Abhängigkeit hinzugefügt worden.

4. Schritt: Anpassung der Gewichte

$w_k(t+1) = w_k(t) + l\,[d(t) - y(t)]\, x_k(t), \quad 0 \leq k \leq n$

In dieser Gleichung ist l ein Faktor kleiner 1 und größer 0, auch Lernrate genannt

Wenn die gewünschte Ausgabe $d(t)$ mit der berechneten Ausgabe $y(t)$ übereinstimmt, wird das Gewicht nicht geändert

5. Schritt: Wiederhole durch Gehen zum 2. Schritt

Künstliche Neuronale Netze

- Beispiel 1 -

Einfaches neuronales Netz nach dem Perceptron-Schema (nach Aleksander, Morton 1990, S. 21)
Aufgabe: Unterscheide H und T Buchstaben

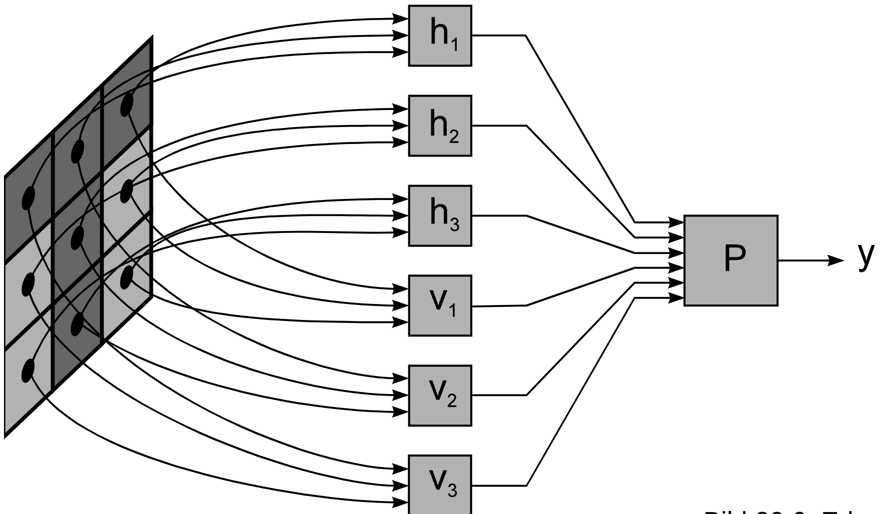


Bild 20.6: Erkennung von H und T Buchstaben

h_1 - h_3 = 1 wenn mindestens 2 Punkte schwarz sind, sonst 0;
v_1 - v_3 = 1 wenn mindestens 2 Punkte schwarz sind, sonst 0;

P hat Aktivierungsfunktion *f(s)* und Gewichte w_i

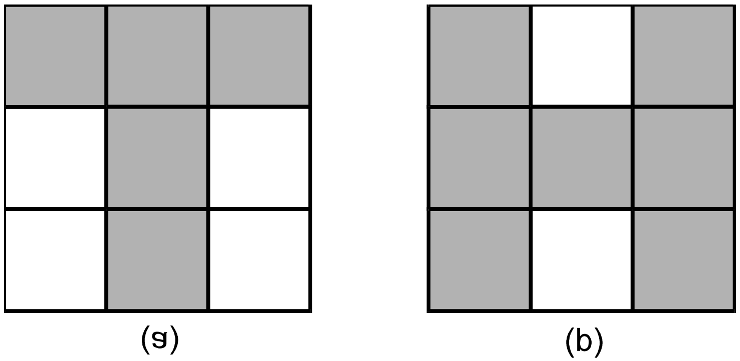
Künstliche Neuronale Netze

- Beispiel 1 -

Einfaches neuronales Netz nach dem Perceptron-Schema (nach Aleksander, Morton 1990, S. 21)

Aufgabe: Unterscheide H und T Buchstaben

Mit einer passenden Aktivierungsfunktion ist nun dafür zu sorgen, dass
y = +1 (das Ergebnis der Summation ist 1) für T-Buchstaben und
y = -1 (das Ergebnis der Summation ist -1) für H-Buchstaben ist.



Trainingsdaten zum Einlernen des Netzes, Buchstabe T (a) und Buchstabe H (b) (nach Aleksander, Morton 1990, S. 23)

a) die Gewichte *h1* und *v*2 sollen positive Werte als Ergebnis haben, z. B. +1,

b) die Gewichte von *v1, v3, h1, h2* und *h3* müssten dann niedrige Werte haben, beispielsweise -1. Da *h1* zum Erkennen beider Buchstabentypen benötigt wird, soll das Gewicht einen niedrigen Wert nahe 0 annehmen.

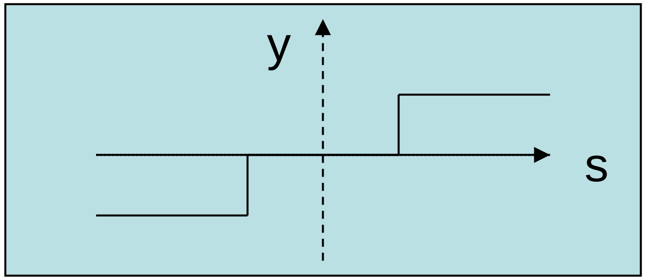
Daraus ergibt sich für w: w = (0, -1, -1, -1, +1, -1)

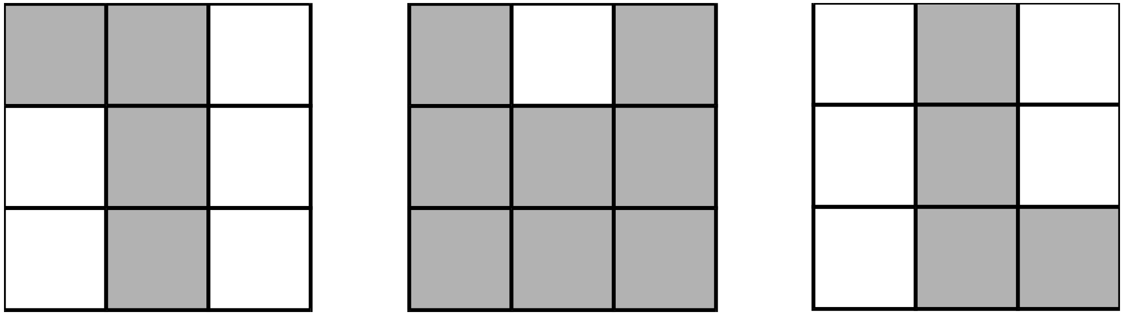
Künstliche Neuronale Netze

- Beispiel 1 -

Aktivierungsfunktion:

$$f(s) = 1 ;\ s \geq 1$$
$$= 0 ;\ -1 < s \geq 1$$
$$= -1 ;\ s \leq -1$$





(a) (b) (c)

Erkennung von Mustern als T- und H-Buchstaben (nach Aleksander, Morton 1990, S. 24)

w = (0, -1, -1, -1, +1, -1)

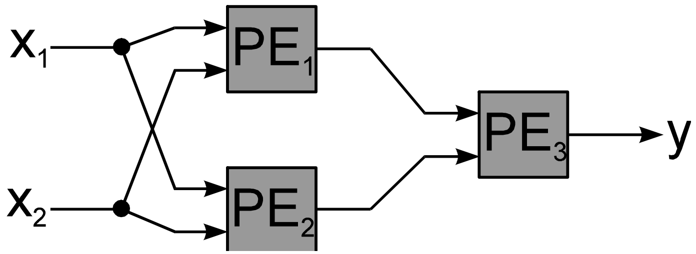
Daraus ergäbe sich für ein T die Summe: 0+0+0 +0+1+0= 1
und für H die Summe = 0+(-1)+(-1)+(-1)+0+(-1)= -4
und für die Beispiele aus Bild 20.8:

a) 0+0+0 +0+1+0 = 1 ; T Buchstabe
b) 0+(-1)+(-1) +(-1)+1+(-1) = -3 ; H Buchstabe
c) 0+0+(-1) +0+1+0= 0 ; weder T noch H

Künstliche Neuronale Netze

- Beispiel 2 -

Die Antivalenz oder das Exklusiv Oder (XOR)



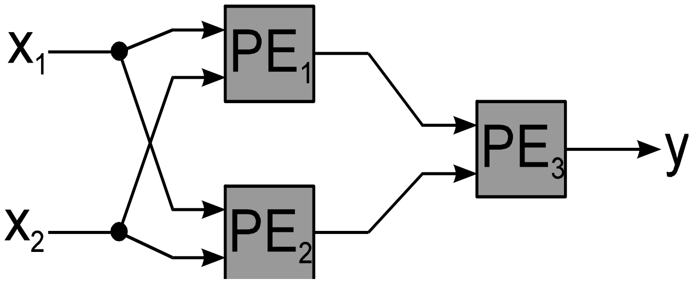
Frank Rosenblatt zeigte, dass ein einfaches Perzeptron mit zwei Eingabewerten und einem einzigen Ausgabeneuron zur Darstellung der einfachen logischen Operatoren AND, OR und NOT genutzt werden kann. Marvin Minsky und Seymour Papert wiesen jedoch 1969 nach, dass ein einlagiges Perzeptron den XOR-Operator nicht auflösen kann (Problem der linearen Separierbarkeit). Dies führte zu einem Stillstand in der Forschung der künstlichen neuronalen Netze (KNN).

XOR Eingang x_1, x_2	Summe PE_1	Ausgang PE_1	Summe PE_2	Ausgang PE_2	Summe PE_3	Ausgang y
1 , 1						
1 , 0						
0 , 1						
0 , 0						

Zu vervollständigende Tabelle

Künstliche Neuronale Netze

- Beispiel 2 -



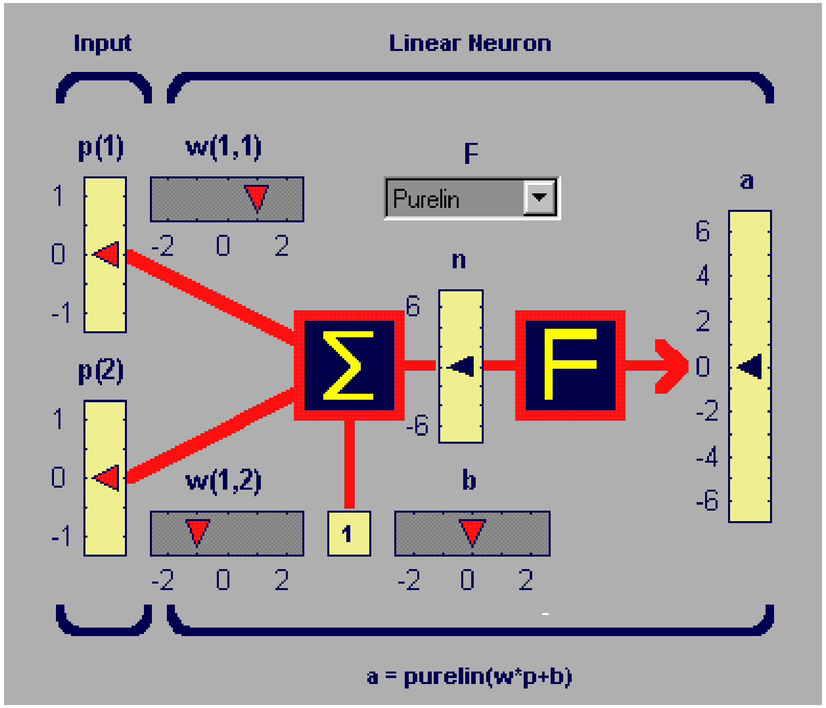
Exklusiv Oder (XOR)

PE1 hat die **Gewichte**:
-1 für w1 (Eingang x1) und **+1 für w2** (Eingang x2).
Das Ausgangssignal wird wie in Gleichung 1 beschrieben berechnet. Die Aktivierungsfunktion wird so gewählt, dass beim Überschreiben eines **Schwellwertes**, hier **0.5**, die PE als Ergebnis +1 liefert, andernfalls 0.
Für **PE2** gilt für die Gewichte:
+1 für w1 (Eingang x1) und -1 für w2 (Eingang x2),
Schwellwert ist ebenfalls 0.5 und das Ausgangssignal wird auch nach Gleichung 1 berechnet.
Für **PE3** gilt für die Gewichte:
+1 für w1 (Ausgang von PE1) und +1 für w2 (Ausgang von PE2),
Schwellwert ist ebenfalls 0.5 und das Ausgangssignal wird auch nach Gleichung 1 berechnet.

Eingang x_1, x_2	Summe PE_1	Ausgang PE_1	Summe PE_2	Ausgang PE_2	Summe PE_3	Ausgang y
1 , 1	0	0	0	0	0	0
1 , 0	-1	0	1	1	1	1
0 , 1	1	1	-1	0	1	1
0 , 0	0	0	0	0	0	0

Künstliche Neuronale Netze
- Prozesseinheit unter MatLab -

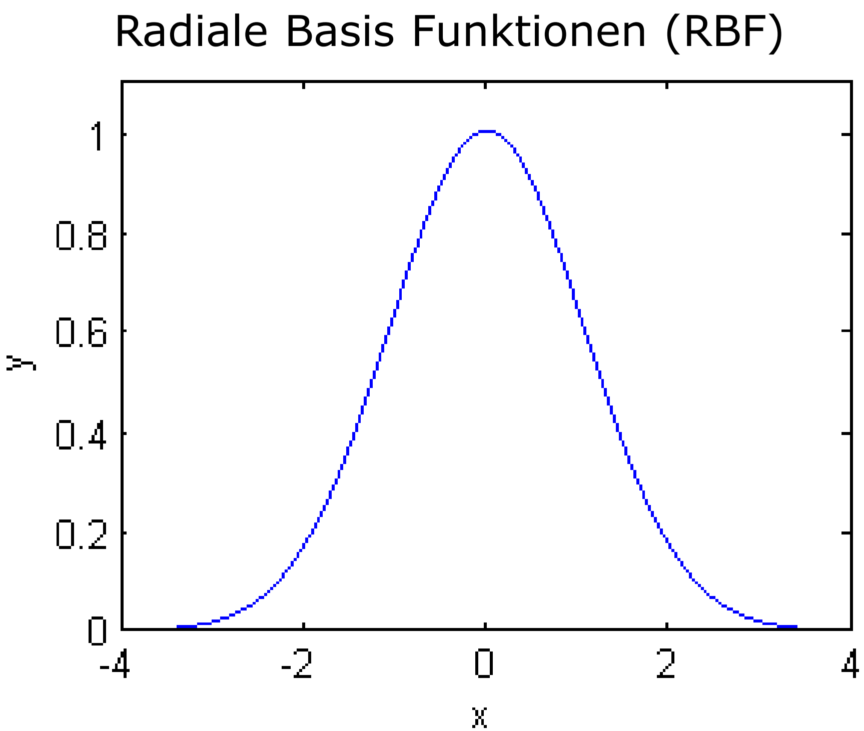
Demo Single Layer Perceptron (SLP) unter Matlab (NN)

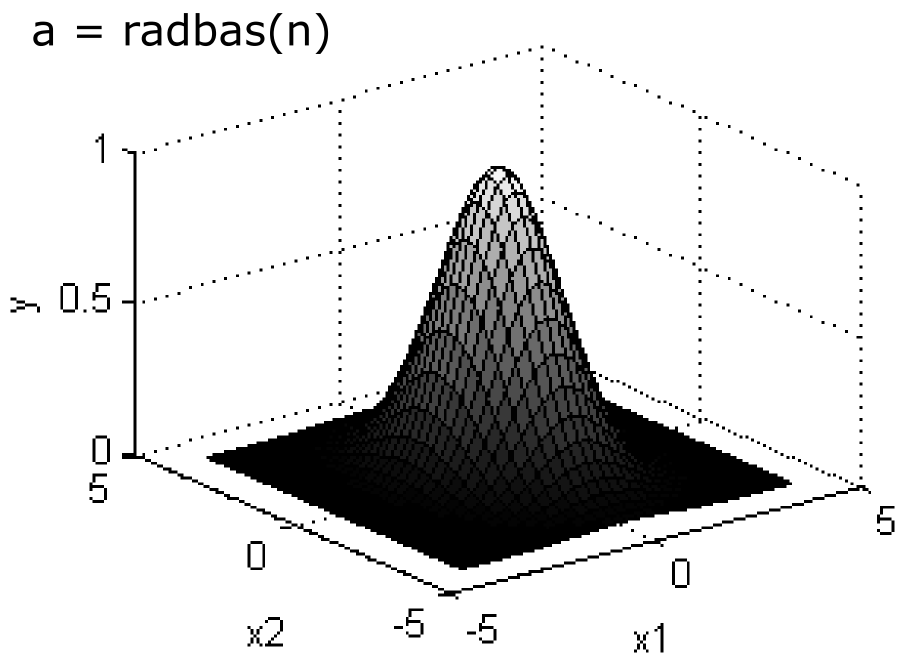


Zwischenergebnis n = w*p+b

Künstliche Neuronale Netze

- Radiale Basis Funktionen -





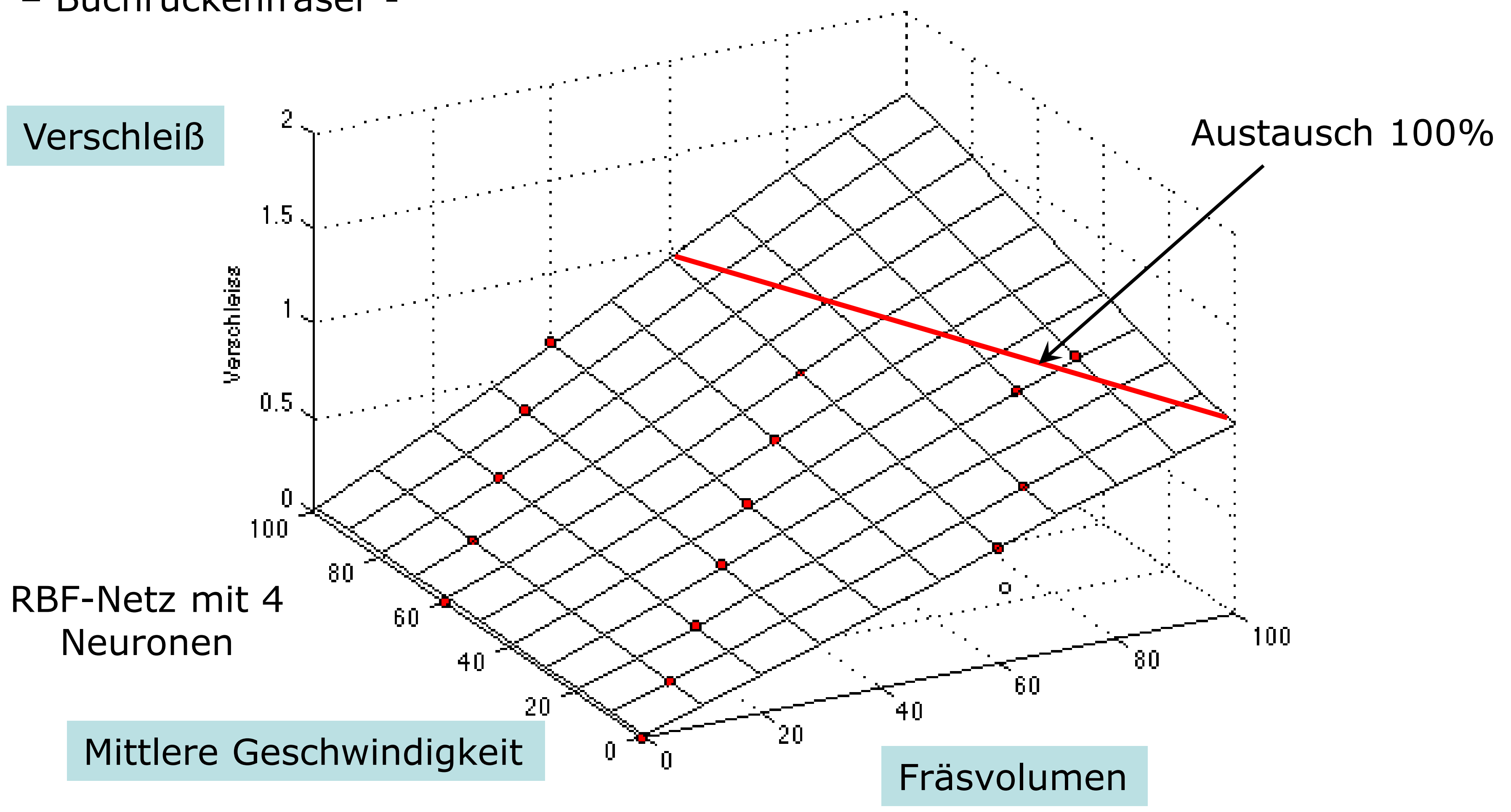
Zwei einzelne Gaußglocken, Ein- und Zweidimensional
In beiden Fällen ist das Zentrum der Gaußglocke im Nullpunkt
Der Abstand r zum Zentrum (0; 0) berechnet sich schlicht aus dem Satz des Pythagoras: r = Quadratwurzel aus $x_1^2 + x_2^2$.

RBF Netz : Aufsummieren der Gaußglocke

Künstliche Neuronale Netze

- Beispielanwendung – Papierfräser -

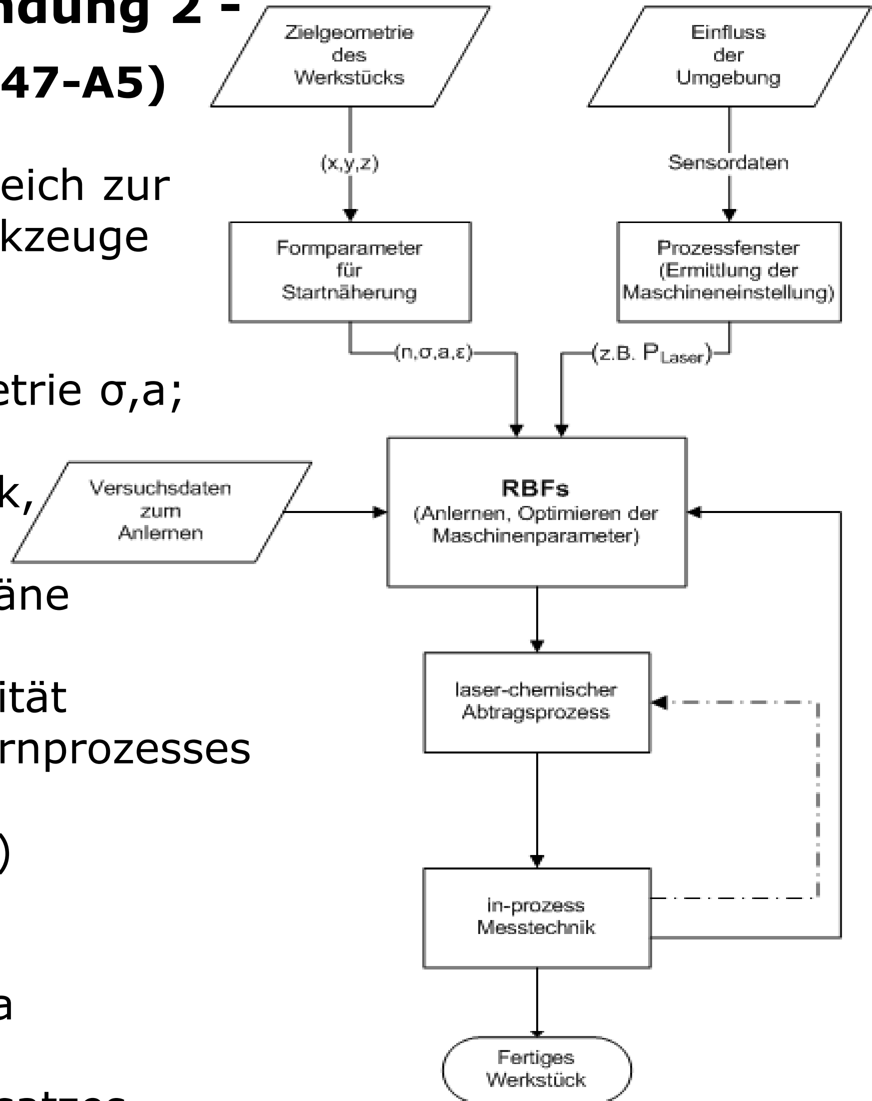
Digitale Maschinenakte: Verschleißprognose aufgrund der Prozessdaten, Beispiel – Buchrückenfräser -



Künstliche Neuronale Netze

- Beispielanwendung 2 -

Laser-chemische Mikroumformung (SFB747-A5)



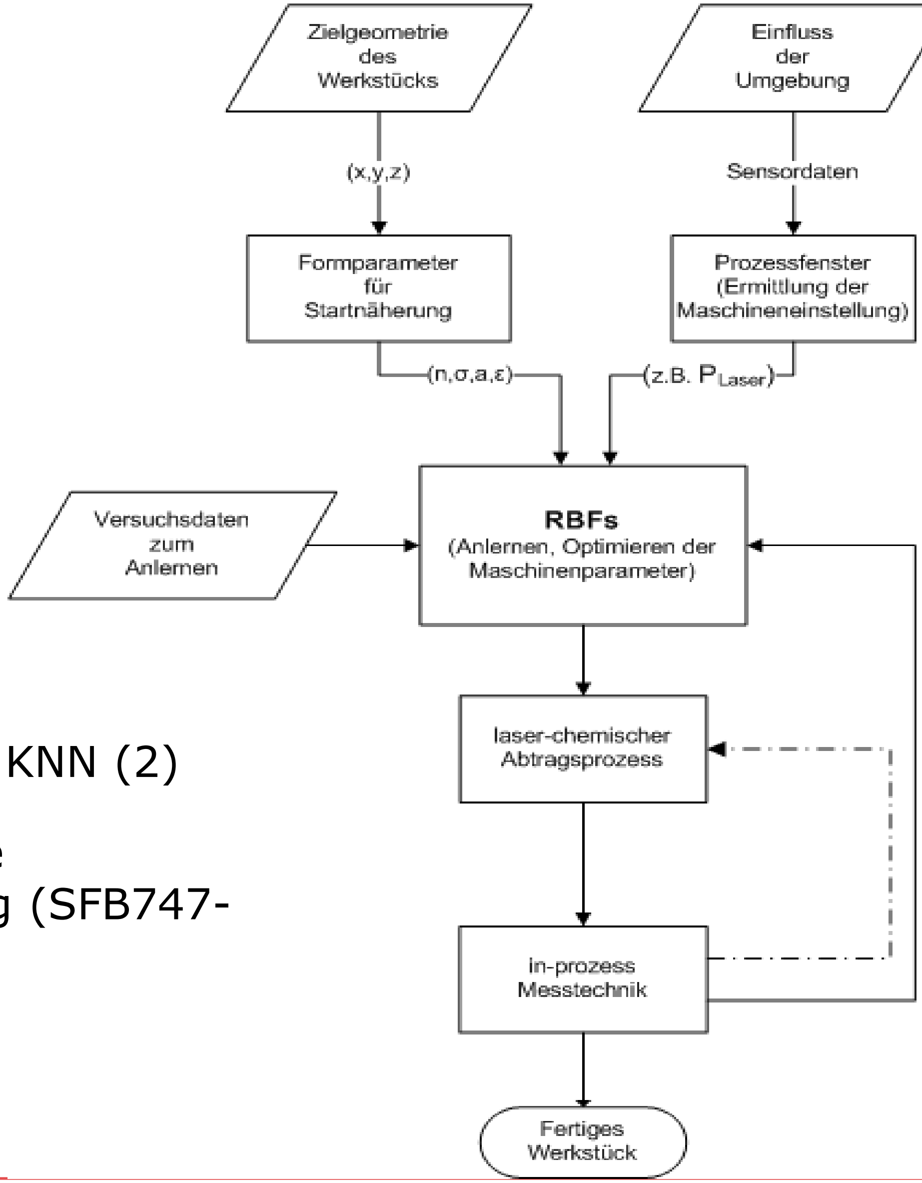
Ziel: Abtragen von Material im Mikrometerbereich zur Herstellung qualitativ hochwertiger Kleinstwerkzeuge

Motivation:
- Prozess unterliegt sehr vielen Parametern
  (Anzahl n der Abtragsbahnen; Abtragsgeometrie σ,a;
  Qualitätsanforderung, z.B. Rauheit ε
  Umgebungseinflüsse, z.B. Temperatur, Druck,
  Feuchtigkeit, etc.)
- Erlernen auf Basis teilfaktorieller Versuchspläne

Vorteile RBF:
- Hohe Datenverarbeitungsrate, große Flexibilität
- Schnellere und sicherere Konvergenz des Lernprozesses
- Hohe interne Transparenz
  (Aufbau und Funktionsweise nachvollziehbar)

Nachteile RBF:
- Vorinformationen erforderlich
  (Wissen über Prozessgrenzen erforderlich, da
  eingeschränkte Extrapolationsmöglichkeit)
- Größerer Stichprobenumfang des Lerndatensatzes

Künstliche Neuronale Netze



Anwendung von KNN (2)

Laser-chemische Mikroumformung (SFB747-A5)

Künstliche Neuronale Netze

- Anwendungsbeispiel 3 -

Modellierung und Steuerung der Produktion mit künstlichen Neuronalen Netzen

wachsende Produktvielfalt

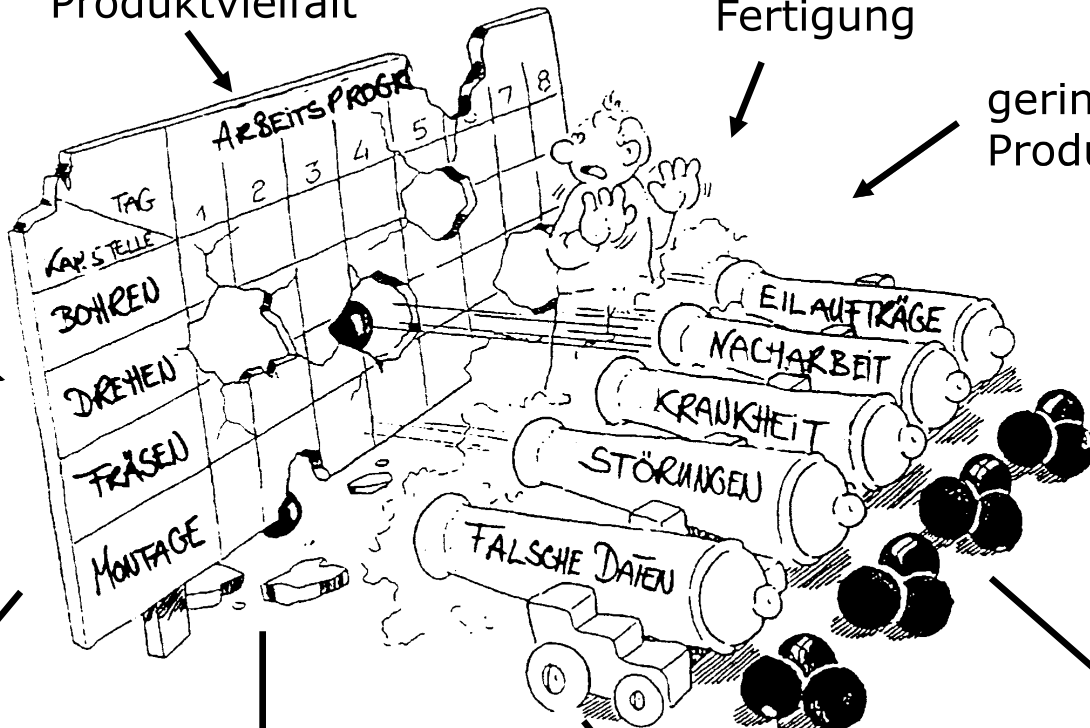
mehr kundenspezifische Fertigung

niedrige Auftrags-bestände

geringe Produktionstückzahlen

hohe Marktdynamik

kurze Liefertermine



wachsende Diskrepanzen zwischen ursprünglichem Plan und aktuellem Zustand

häufige Umplanungen

Selbstverstärkung kleiner Störungen

(Scherer) nach Tilo Hamann, BIBA ©

Künstliche Neuronale Netze

- Anwendung 3 – Durchlaufzeiten in der Fertigung -

Anwendung von KNN

Aufgabe

- Durchlaufzeit der Lose durch das Fertigungssystem und die Bestände an den Arbeitsplätzen werden geregelt

Erwartete Vorteile der KNN gegenüber konventioneller Regelung

- Lernfähigkeit, Parallelität, Verteilte Wissensrepräsentation
- Höhere Fehlertoleranz, Assoziative Speicherung von Informationen
- Robustheit gegen Störungen oder verrauschte Daten
- Aktive Repräsentation

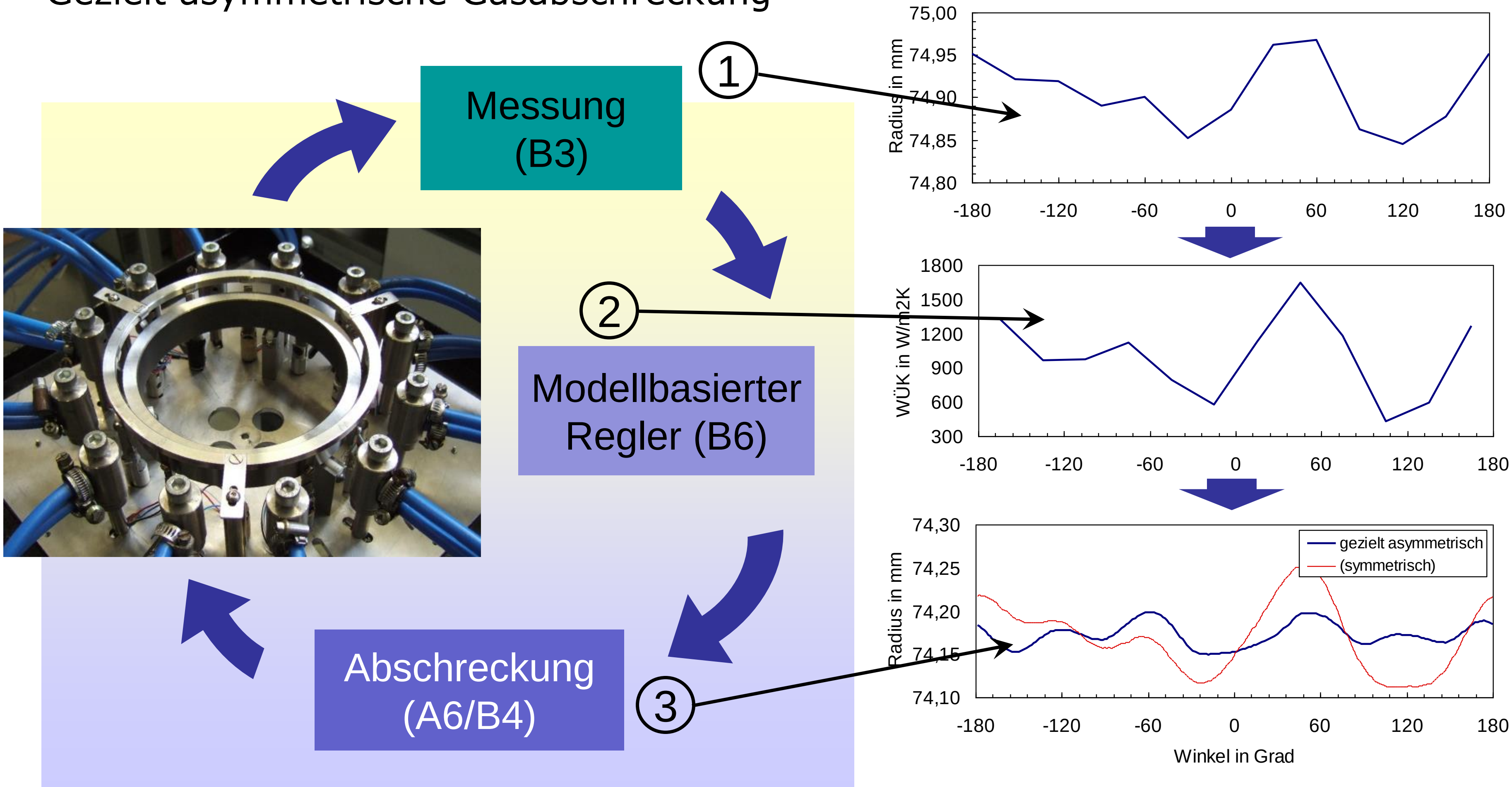
Erwartete Nachteile der KNN

- Wissenserwerb nur durch Lernen möglich
- keine Introspektion möglich (z.B. durch "Selbstbeobachtung")
- Logisches (sequentielles) Schließen ist schwer zu realisieren
- Lernen ist relativ langsam

Künstliche Neuronale Netze
- Anwendung 4 -Gasabschreckung -

Anwendung von KNN (4)

Gezielt asymmetrische Gasabschreckung



Künstliche Neuronale Netze

- Weitere Anwendungen und Einsatzgebiete -

Luftfahrt: Autopilot, Steuerungssysteme, Flugroutensimulation, Komponentensimulation, Fehlererkennung in Komponenten

Fahrzeugindustrie: Leitsysteme, Gewährleistungsanalyse, Motor- und Getriebediagnose, Montagefehlererkennung und Diagnose

Militär: Zielverfolgung, Objektseparierung, Steuerung und Regelung, Signal- und Bildverarbeitung in Sonar und Radaranwendungen

Elektonik: Schaltungs- und Chiplayout, Chip-Fehleranalyse, Sprachbe- und - verarbeitung (Digitale Telephonie), Bild- und Stimmerkennung, nichtlineare Modelle

Produktionstechnik: Prozess-Steuerung und Regelung, Produktdesign und Analyse, Prozess und Maschinendiagnose, Vorhersage, Wartung, Bildverarbeitung uvm.

Banken und Versicherungen: Dokumentenerkennung, Kreditvergabeprüfung
... **Unterhaltungsindustrie, Medizin, Öl- und Gasindustrie,** Robotic, Finanzindustr

Zukünftig (BIMAQ): Getriebediagnose in Windenergieanlagen

Lehrziele und Gliederung