

# Intelligente Systeme

- Referat -

Adrian Helberg

HAW Hamburg

21.02.2020

# Inhalt

- 1 Suchen
- 2 Lernen
- 3 Sequenzen
- 4 Quellen



# Problem des Handlungsreisenden

## Definition

Die Aufgabe besteht darin, eine Reihenfolge für den Besuch mehrerer Orte so zu wählen, dass keine Station außer der ersten mehr als einmal besucht wird, die gesamte Reisstrecke des Handlungsreisenden möglichst kurz und die erste Station gleich der letzten Station ist <sup>1</sup>

- Kombinatorisches Optimierungsproblem
- $10! = 3628800$  mögliche Lösungen
- Theoretische Informatik
- Festlegung der Reihenfolge der zu besuchenden Städte
- NP-vollständig

---

<sup>1</sup>[https://de.wikipedia.org/wiki/Problem\\_des\\_Handlungsreisenden](https://de.wikipedia.org/wiki/Problem_des_Handlungsreisenden)




# Genetischer Algorithmus

## Definition

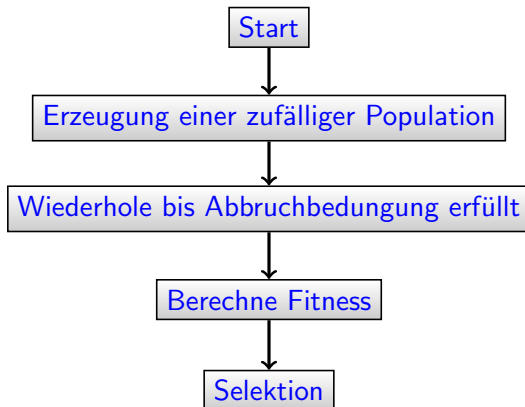
Evolutionäre Algorithmen (EA) sind eine Klasse von stochastischen, metaheuristischen Optimierungsverfahren, deren Funktionsweise von der Evolution natürlicher Lebewesen inspiriert ist <sup>2</sup>

- Optimierte, akzeptable Lösung
- Aufgabenstellung mit hoher kombinatorischen Komplexität
- Mengen an (immer besser werdenden) Lösungen
- Kein „Hängenbleiben“ an einem lokalen Optimum

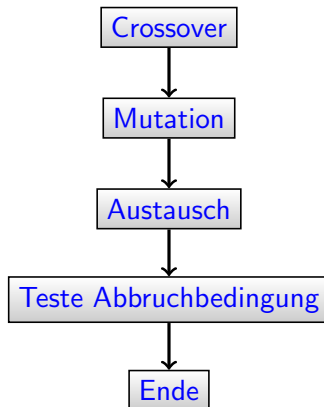
---

<sup>2</sup>[https://de.wikipedia.org/wiki/Evolutionärer\\_Algorithmus](https://de.wikipedia.org/wiki/Evolutionärer_Algorithmus)   

# Ablauf (1/2)



# Ablauf (2/2)



# Eigenschaften

- Chromosomen
- Fitness
- Selektion
- Kreuzung
- Mutation
- Austausch



# Realisierung mit Java

```
-----Genetic Algorithm Properties-----  
Number of Cities:    48  
Population Size:     500  
Max. Generation:    500  
k Value:             3  
Elitism Value:       1  
Force Uniqueness:    false  
Local Search Rate:   0.0  
Crossover Type:      UNIFORM_ORDER  
Crossover Rate:      90.0%  
Mutation Type:       INSERTION  
Mutation Rate:       4.0%
```

◀ ◻ ▶ ◀ ◻ ▶ ◀ ≡ ▶ ◀ ≡ ▶ ≡ ↺ 🔍 ↻

# Selbstorganisierende Karte

## Definition

Als Selbstorganisierende Karten [...] bezeichnet man eine Art von künstlichen neuronalen Netzen. [...] Ihr Funktionsprinzip beruht auf der biologischen Erkenntnis, dass viele Strukturen im Gehirn eine lineare oder planare Topologie aufweisen <sup>3</sup>

- Neuroinformatik
- *Kohonen-Karte*, *SOM*
- Unüberwachtes Lernen

---

<sup>3</sup>[https://de.wikipedia.org/wiki/Selbstorganisierende\\_Karte](https://de.wikipedia.org/wiki/Selbstorganisierende_Karte)

# Prinzip

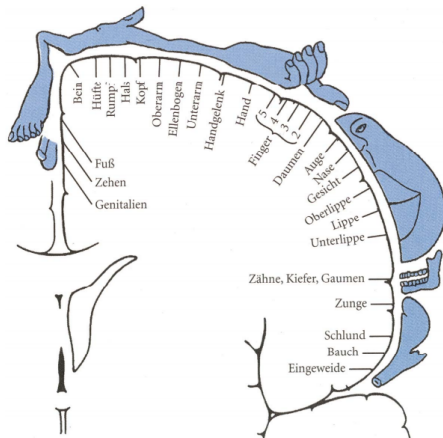


Abbildung: Topologische Abbildung des sensorischen Kortex

# Neuron

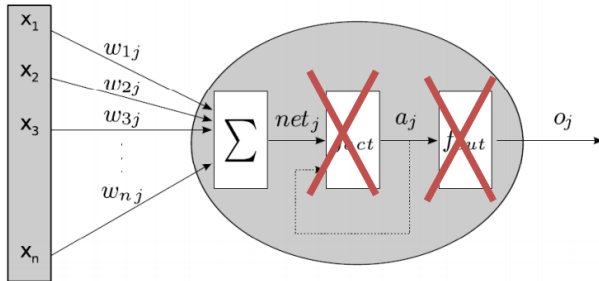


Abbildung: Neuron einer SOM

- Keine Aktivierungsfunktion
- Kein Mappingfunktion (Ooutput)

# Neuron

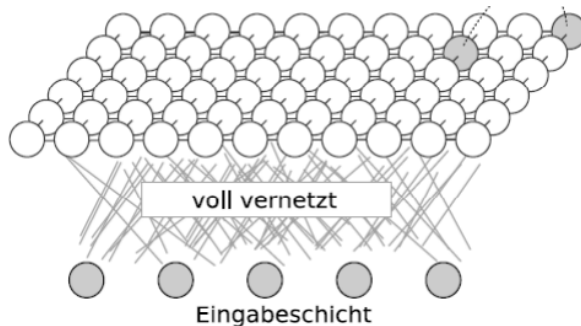


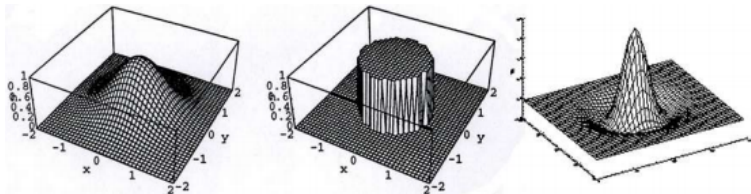
Abbildung: *Kohonen-Netz*

- Eingabeneuronen sind über die Gewichtsvektoren mit den Ausgabeneuronen vollvermascht

# Lernen

- Auswahl von Neuronen über die „Eignung“ (euklidische Norm)
- Abstand zwischen Eingabevektoren und Gewichtvektoren wird bestimmt
- Zufällige Auswahl bei gleicher Distanz
- *Besonderes*: „Sigerneuron“ **und** dessen Nachbarneuronen werden angepasst

# Distanzfunktionen



**Abbildung:** Gaussche Glockenfunktion, Zylinderfunktion und „Mexican-Hat“



# Anpassen

## Siegerneuron anpassen

$W_c = W_c + \Delta W_c$  und  $\Delta W_c = \eta * (x - W_c)$ , mit

- $c$  als Siegerneuron
- $W_c$  als Gewichtungsvektor
- Eingabevektor  $x$

Es gilt  $0 < \eta < 1$ .

# Anpassen

## Nachbarneuronen anpassen

$\Delta W_j = \eta * h_{cj} * (x - W_j)$ , mit

- $h_{cj}$  als Nachbarschaftsfunktion (Wie stark lernt das Neuron  $j$  mit bei Siegerneuron  $c$ )

Als Funktion des Abstandes  $z$  der Neuronen wird in der Praxis oft eine Zylinderfunktion gewählt:

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

# Lernmetapher

## „Kupferschmiede“

Ein Kupferschmied hämmert auf einer Kupferplatte. Hierbei bestimmt

- der Vektor  $x$ , wohin
- die Lernrate  $\eta$ , wie stark
- der Lernradius  $d$ , mit welcher Hammergröße und
- die Nachbarschaftsfunktion  $h_{cj}$ , mit welcher Hammerform geschlagen wird

# Anwendung - Fragestellung

## 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

Abbildung: Sind ähnliche Tiere eines Zoos benachbart?

# Anwendung - Resultierende Map

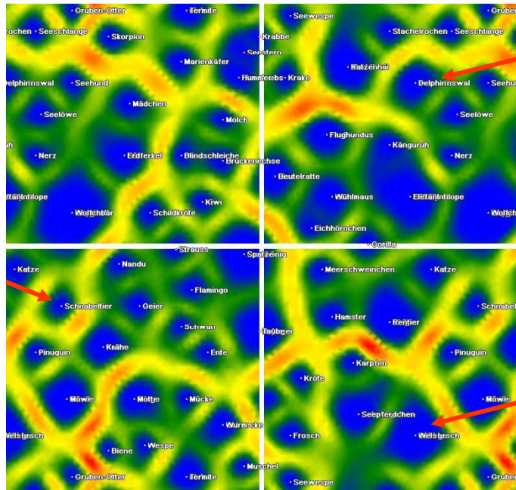


Abbildung: Reduktion mehrerer Dimensionen auf 2 (Karte)

# Anwendung - Anwendungsgebiete

- Dimensionsreduktion
- Optimierung
- Data Mining- Cluster
- Data Mining - Regeln
- Überwachung und Anomaliedetektion
- Kontextkarten

- ◀ ◻ ▶ ◀ ◻ ▶ ◀ ≡ ▶ ◀ ≡ ▶ ≡ ↺ 🔍 ↻

# Thema

## Verarbeitung mobiler Sensordaten

- Steigende Beliebtheit tragbarer Endgeräte
- Kontinuierliche Erfassung physiologischer und funktioneller Daten
- Anwendungen bei Apps für Sport, Gesundheit und Wohlbefinden
- Verarbeitung und Analyse von **Big Data**
- *Deep Learning*- Ansatz



# Thema

## *Human Activity Recognition (HAR)*

- Auswertung zeitlicher Reihendaten von Trägheitssensoren
- Identifikation ausgeführter Aktionen: Klassifizierung von Bewegungsabläufen
- Erkennung von Krankheitsausbrüchen (z.B. Parkinson) oder Wirksamkeit einer Behandlung

# Problem

- *Deep Learning* ist zur Zeit sehr erfolgreich bei Implementationen, die Hochleistungs-Rechencluster nutzen.
- Einsatz auf mobilen Endgeräten stellt sich als schwierig heraus, da nur geringe Ressourcen zur Verfügung stehen
- Gesucht ist ein optimiertes Verfahren: Bestes Ergebnis bei geringsten Ressourcen

# Technik

- Entwerfen der Klassifizierungsmethode für eine Zeitreihenanalyse
- Auswahl von Merkmalen der Klassen
- Oft durch den Einsatz von *Deep Believe Networks* (DBN), *Restricted Boltzman Machines* (RBM) oder *Convolutional Neural Networks* (CNN) umgesetzt
- Herausarbeiten von *Features* aus Rohdaten

# „Shallow Deep Learning“- Methode

## Prinzip

*Shallow Deep Learning* setzt sich aus *Deep Learning* und *Shallow Learning* zusammen und ist für *HAR* unterteilt in:

- Input
- Segmentextraktion
- Spektrogramm
- *Deep Learning* Modul
- *Shallo Features*
- Training und
- Evaluation

# Ablauf

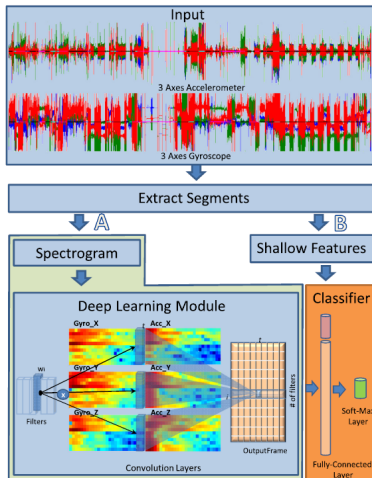


Abbildung: Schematischer Ablauf

# Prozess

## Input

Um eingehende Daten (Input) zu generieren wird auf verbaute Trägheitssensoren wie

- Beschleunigungsmesser und
- Gyroskope

zugegriffen. Zugriffe auf anderen Sensoren, wie Elektrokardiographie (EKG) oder Elektromyographie (EMG), sind ebenfalls möglich (Smartwatch)

# Prozess

## Segmente extrahieren

- Extraktion von  $n$  Segmenten aus den Rohdaten ( $n$  hängt von der Art der Anwendung ab)
- Segmente bei *HAR* werden auf 4 bis 10 Sekunden (*sliding window*) gesetzt
  - Genauigkeit der Erkennung der ausgeführten Aktivität soll maximal sein
  - Verschleierung der Grenzen zwischen verschiedenen Aktivitäten soll minimal sein

# Prozess

## Spektrogramm

- Segmente werden als Spektrogramm-Repräsentation an das *Deep Learning* Modul weitergereicht
- Herausarbeiten von Intensitätsunterschieden in den Trägheitspunkten des Spektrogramms, wie
  - Zeit- und Abtastrateninvarianz
  - Amplituden
  - Frequenz



# Prozess

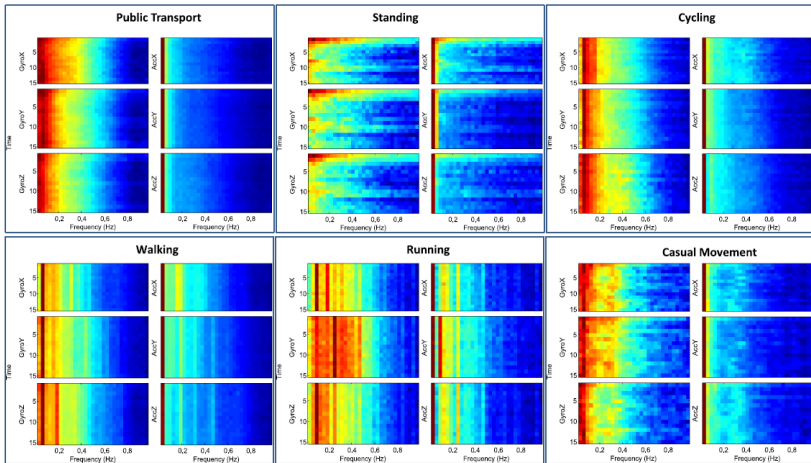


Abbildung: Spektrogramme aus verschiedenen Aktivitäten

# Prozess

## Shallow Features

Intensitätsunterschiede in den Spektrogrammen ergeben „flache“ Merkmale, die auf die Spektrogramme angewendet werden.

Beispiele sind

- Quadratischer Mittelwert
- Standardabweichung

## Klassifizierung

Sobald sowohl flache („shallow learned“), als auch tiefe („deep learned“) Eigenschaften berechnet wurden, können diese zu einem eindeutigen Vektor zusammengeführt und klassifiziert werden

# Prozess

## Training

Flache und tiefe Features werden zusammen in einem einheitlichen, neuronalem Netz trainiert. In jeder Schicht des neuronalen Netzes werden Fehler (Errors) zwischen den Soll- und Istwerten in einer *Backwards Propagation*-Routine genutzt, um die Gewichtungen in den *Hidden-Layers* anzupassen.

## Evaluation

Das vorgeschlagene *Shallow Deep Learning*-Verfahren könnte nun mit verschiedenen Datensätzen analysiert und evaluiert werden

# Prozess

Dataset	Description	# of Classes	Subjects	Samples	Sampling Rate
ActiveMiles	Daily activities collected by smartphone in uncontrolled environments	7	10	4,390,726	50 – 200 Hz
WISDM v1.1	Daily activities collected by smartphone in a laboratory	6	29	1,098,207	20 Hz
WISDM v2.0	Daily activities collected by smartphone in uncontrolled environments	6	563	2,980,765	20 Hz
Daphnet FoG	Freezing of gait episodes in Parkinson's patients	2	10	1,917,887	64 Hz
Skoda	Manipulative gestures performed in a car maintenance scenario	10	1	~ 701,440	98 Hz

Abbildung: Datensätze für eine Evaluierung

# Inhalt

- 1 Suchen
- 2 Lernen
- 3 Sequenzen
- 4 Quellen**

# Quellen I



User: Akaq.

Problem des handlungsreisenden.

[https://de.wikipedia.org/wiki/Problem\\_des\\_Handlungsreisenden](https://de.wikipedia.org/wiki/Problem_des_Handlungsreisenden), Dezember 2019.



Yacin Bessas.

Selbstorganisierende karten - proseminar neuronale netze.

<http://www.informatik.uni-ulm.de/ni/Lehre/WS04/ProSemNN/pdf/SOM.pdf>, April 2015.

# Quellen II



Sabrina Eimler, Stefan Geisler, and Philipp Mischewski.

Ethik im autonomen fahrzeug: Zum menschlichen verhalten in drohenden unfallsituationen.

In Raimund Dachzelt and Gerhard Weber, editors, *Mensch und Computer 2018 - Workshopband*, Bonn, 2018. Gesellschaft für Informatik e.V.



Julia Köppe.

Einer muss sterben - nur wer?

<https://www.spiegel.de/wissenschaft/technik/unfaelle-mit-selbstfahrenden-autos-wer-soll-leben-wer-soll-sterven-2018-10-11>, Oktober 2018.

# Quellen III



User: LazoCoder.

Implementation traveling salesman.

<https://github.com/LazoCoder/>

Genetic-Algorithm-for-the-Traveling-Salesman-Problem,  
Februar 2017.



Thomas Leske.

Trolley-problem.

<https://de.wikipedia.org/wiki/Trolley-Problem>,

Februar 2020.



User: NadirSH.

Evolutionärer algorithmus.

[https://de.wikipedia.org/wiki/Evolution%C3%A4rer\\_](https://de.wikipedia.org/wiki/Evolution%C3%A4rer_Algorithmus)  
Algorithmus, August 2019.



# Quellen IV



User: Orthographus.

Selbstorganisierende karten.

[https://de.wikipedia.org/wiki/](https://de.wikipedia.org/wiki/Selbstorganisierende_Karte)

[Selbstorganisierende\\_Karte](https://de.wikipedia.org/wiki/Selbstorganisierende_Karte), Januar 2020.



Daniele Ravi.

A deep learning approach to on-node sensor data analytics for mobile or wearable devices.

Download from [https](https://www.researchgate.net/publication/311947612_A_Deep_Learning_Approach_to_on-Node_Sensor_Data_Analytics_for_Mobile_or_Wearable_Devices):

[//www.researchgate.net/publication/311947612\\_A\\_](https://www.researchgate.net/publication/311947612_A_Deep_Learning_Approach_to_on-Node_Sensor_Data_Analytics_for_Mobile_or_Wearable_Devices)

[Deep\\_Learning\\_Approach\\_to\\_on-Node\\_Sensor\\_Data\\_](https://www.researchgate.net/publication/311947612_A_Deep_Learning_Approach_to_on-Node_Sensor_Data_Analytics_for_Mobile_or_Wearable_Devices)

[Analytics\\_for\\_Mobile\\_or\\_Wearable\\_Devices](https://www.researchgate.net/publication/311947612_A_Deep_Learning_Approach_to_on-Node_Sensor_Data_Analytics_for_Mobile_or_Wearable_Devices), Januar

2017.

# Danksagung

Danke für die Aufmerksamkeit!