

Neural Networks

Math. Models Using Neural Networks

Dieter Kilsch

Technische Hochschule Bingen, FB 2 • Technik, Informatik und Wirtschaft

APL Germany, Solingen

November 26th, 2018

Convenience by Autonomous Vehicles

2007: Vision



The car manages itself,

the driver's mind

is free to enjoy live.

Convenience by Autonomous Vehicles

2007: Vision



© Die Zeit, 14.6.2007

The car manages itself,
the driver's mind
is free to enjoy live.

Convenience by Autonomous Vehicles

today: Vision ?



© Die Zeit, 14.6.2007

The car manages itself,
the driver's mind
is free to enjoy live.

Autonomous Cars Come True

“First Autonomous Car on Public Roads”

TU Braunschweig, <https://www.tu-braunschweig.de/presse/medien/presseinformationen?year=2010&pinr=133>

The car drives, the driver enjoys ...

Leonie

8.10.2010

Weltweit erstes automatisches Fahren im realen Stadtverkehr

Forschungsfahrzeug „Leonie“ fährt automatisch auf dem Braunschweiger Stadtring

Weltpremiere in Braunschweig: Erstmals fährt heute ein Fahrzeug automatisch im alltäglichen Stadtverkehr. Im Rahmen des Forschungsprojekts „Stadtpilot“ hat die Technische Universität Braunschweig in ihrem Kompetenzzentrum, dem niedersächsischen Forschungszentrum Fahrzeugtechnik, ein Forschungsfahrzeug entwickelt, das automatisch eine vorgegebene Strecke im regulären Verkehr fährt.

- 1 Analysis, Modelling and Solutions
- 2 Neurons and Neural Networks
- 3 Sabbatical Working Examples
- 4 Pattern Recognition
- 5 Neural Networks: Image and Speech Recognition
- 6 Conclusion

- 1 Analysis, Modelling and Solutions
 - What does Artificial Intelligence Mean?
 - Problem Solving Strategies
 - Objectives
- 2 Neurons and Neural Networks
- 3 Sabbatical Working Examples
- 4 Pattern Recognition
- 5 Neural Networks: Image and Speech Recognition
- 6 Conclusion

(Artificial) Intelligence

Intelligence

- Individual Intelligence
- Intelligence of a group
- Emotional Intelligence

Artificial Intelligence Alan Turing 1950: Turing Test

An engine has artificial intelligence if a human observer cannot decide if he deals with an engine or a human being.

(Artificial) Intelligence

Intelligence

- Individual Intelligence
- Intelligence of a group
- Emotional Intelligence

Artificial Intelligence Alan Turing 1950: Turing Test

An engine has artificial intelligence if a human observer cannot decide if he deals with an engine or a human being.

(Artificial) Intelligence

Intelligence

- Individual Intelligence
- Intelligence of a group
- Emotional Intelligence

Artificial Intelligence Alan Turing 1950: Turing Test

An engine has artificial intelligence if a human observer cannot decide if he deals with an engine or a human being.

Research Objective

Analyse intelligent achievements and make these methods computable.

Brain — Computer: a Comparison

Brain and standard computers

high performance w.r.t. to different tasks

Brain

- Highly parallel
- Fault tolerant
- Pattern recognition
- Generalization
- Self-organizing
- ca. 10^{11} neurons, reducing to 10^7
- Every neuron has ca. 10^4 connected neurons.

Computer

Brain — Computer: a Comparison

Brain and standard computers

high performance w.r.t. to different tasks

Brain

- Highly parallel
- Fault tolerant
- Pattern recognition
- Generalization
- Self-organizing
- ca. 10^{11} neurons, reducing to 10^7
- Every neuron has ca. 10^4 connected neurons.

Computer

- Precise
- Faultless storing
- Fast algorithmic calculations
- von Neumann architecture
- Nearly stand alone

Brain — Computer: a Comparison

Brain and standard computers

high performance w.r.t. to different tasks

Brain

- Highly parallel
- Fault tolerant
- Pattern recognition
- Generalization
- Self-organizing
- ca. 10^{11} neurons, reducing to 10^7
- Every neuron has ca. 10^4 connected neurons.

Computer

- Precise
- Faultless storing
- Fast algorithmic calculations
- von Neumann architecture
- Nearly stand alone

How to Solve a Problem?

Algorithm

- Intuitively build a model
- Deduce a numerical algorithm
- Put it into a program
- Use it respecting the preconditions

Expert System

Neural Network

How to Solve a Problem?

Algorithm

- Intuitively build a model
- Deduce a numerical algorithm
- Put it into a program
- Use it respecting the preconditions

Expert System

- Intuitively build a model
- Formulate rules
- Apply Rules
- may solve related problems

Neural Network

How to Solve a Problem?

Algorithm

- Intuitively build a model
- Deduce a numerical algorithm
- Put it into a program
- Use it respecting the preconditions

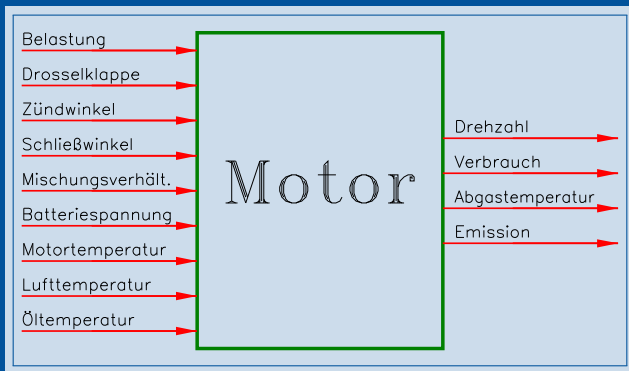
Expert System

- Intuitively build a model
- Formulate rules
- Apply Rules
- may solve related problems

Neural Network

- Intuitively build a model
- needs sampling points
- generalizes based on sampling data
- applies to related problems

Physical Performance: An Engine on a Test Bench

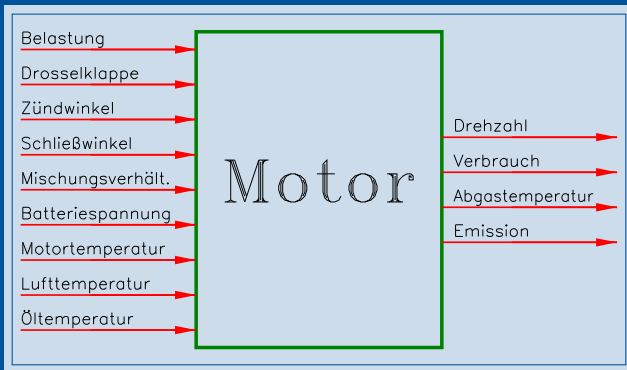


load, throttle valve, ignition angle, dwell angle, mixture, voltage, temperatures of engine, air and oil

→

rotational speed, consumption, temperature and amount of emission

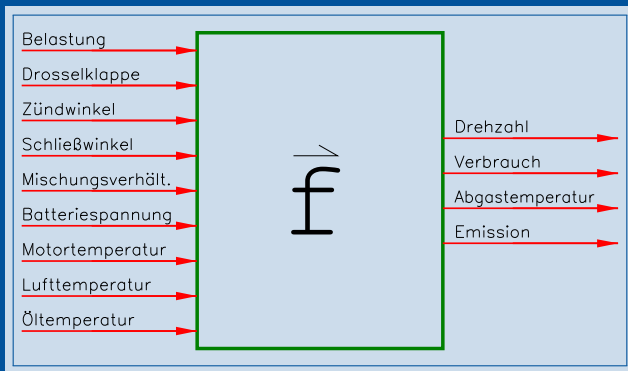
Physical Performance: An Engine on a Test Bench



Targets

- Create the optimal engine characteristic map.
- ... also regarding start situation.
- Reduce test bench time.

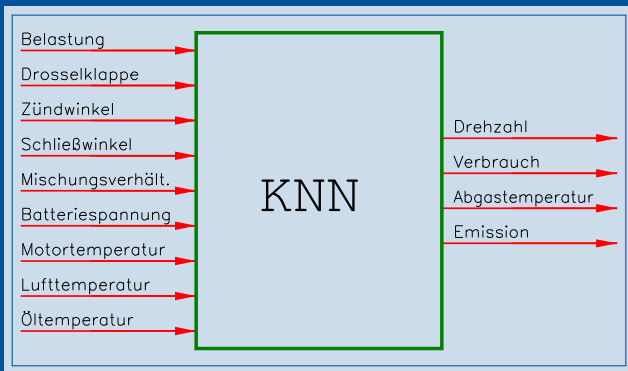
Mathematical Model of an Engine



Mathematical model: abstraction

- Look at the engine as a function
- Assumes functional dependencies (one-one)

Mathematical Model of an Engine

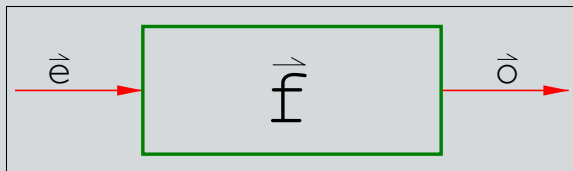


Models with artificial neural network

- Artificial neural networks should learn to “behave” like an engine.
- The knowledge must come from (measured) data.

Physical Performance: Mathematical Model

One-to-one relation: **All influencing factors are known.**

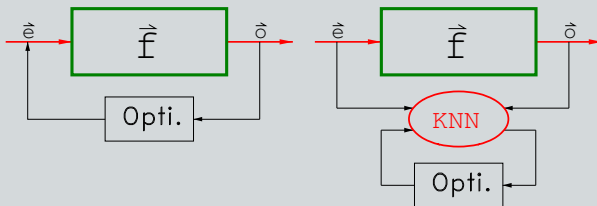


The output \vec{o} depends (functionally) on the input \vec{e} . This relation is described by a f :

$$\vec{o} = \vec{f}(\vec{e}) .$$

Application: Reduce Test Bench Time

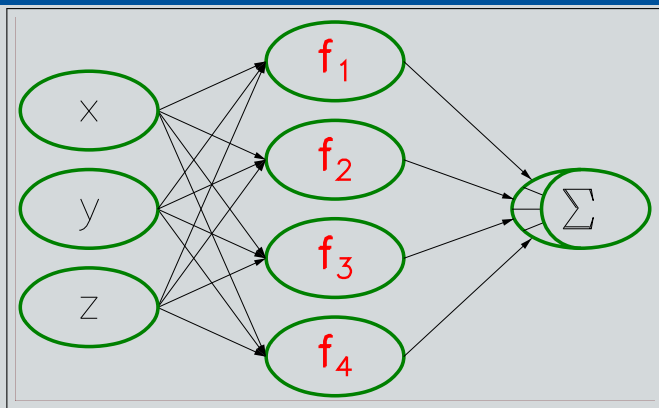
Optimization of characteristic maps using ANNs



- Feed the neural networks with the “knowledge” of several engines: measured data from test bench
- New engine: extending the knowledge base with a few data from test bench
- Optimize the characteristic map using the trained neural network

R. Stricker, BMW AG, 1996

Curve Fitting (Least Square Method, Regression)

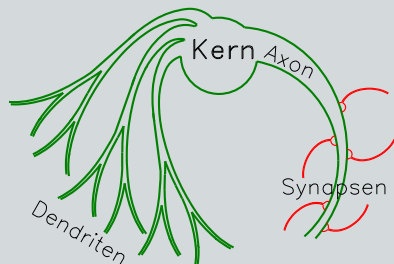


“Learning” only in the last layer

- 1 Analysis, Modelling and Solutions
- 2 **Neurons and Neural Networks**
 - NeuroScience
 - Artificial Neuron, Linear Separation
 - Neural Network Learning
 - Improving Learning
- 3 Sabbatical Working Examples
- 4 Pattern Recognition
- 5 Neural Networks: Image and Speech Recognition
- 6 Conclusion

The Biological Neuron

Principles of Operation

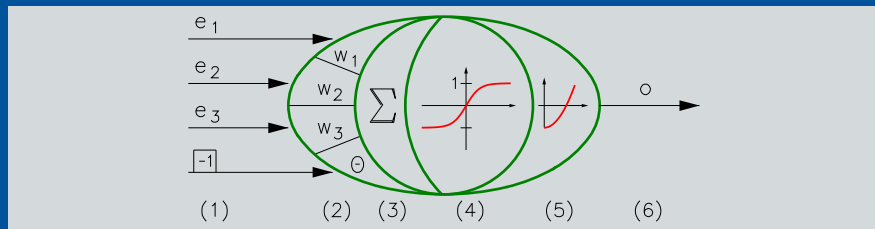


- 1 Impulse through the axon.
- 2 Synapses collect impulse.
- 3 Dendrites transmit it.
- 4 Nucleus gets impulse.
- 5 Overall impulse:
Excitation of the neuron.
- 6 Threshold target reached:
neuron sends impulse.

Learning:

Synapses, dendrites enhance their connection.

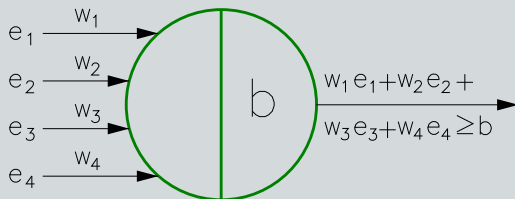
The Artificial Neuron



- 1 input (vector) $\vec{e} = (e_1, \dots, e_n)$, -1 to be used by threshold
- 2 weights and threshold $\vec{w} = (w_1, \dots, w_n)$ and θ
- 3 net (value), propagation $net = \langle \vec{e}, \vec{w} \rangle - \theta = \sum_{i=1}^n e_i w_i - \theta$
- 4 activation (primitive function), activity $a = a(\langle \vec{e}, \vec{w} \rangle - \theta)$
- 5 output function
- 6 output $o = o(a(\langle \vec{w}, \vec{e} \rangle - \theta))$

Weighted Threshold Units

Definition



$$0 \leq \left(\sum_{i=1}^n w_i e_i \right) - b = \langle \vec{w}, \vec{e} \rangle - b$$

Linearly Separable Sets

Hidden neurons separate linearly

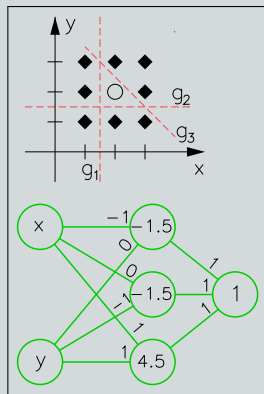
$$\begin{pmatrix} x \\ y \end{pmatrix} \begin{pmatrix} -1 \\ 0 \end{pmatrix} \geq \begin{pmatrix} 1.5 \\ 0 \end{pmatrix} \begin{pmatrix} -1 \\ 0 \end{pmatrix} = -1.5$$

$$\begin{pmatrix} x \\ y \end{pmatrix} \begin{pmatrix} 0 \\ -1 \end{pmatrix} \geq \begin{pmatrix} 0 \\ 1.5 \end{pmatrix} \begin{pmatrix} 0 \\ -1 \end{pmatrix} = -1.5$$

$$\begin{pmatrix} x \\ y \end{pmatrix} \begin{pmatrix} 1 \\ 1 \end{pmatrix} \geq \begin{pmatrix} 3 \\ 1.5 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = 4.5$$

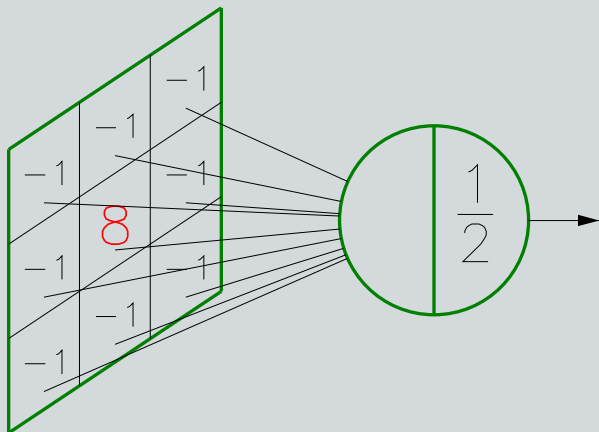
The output neuron gathers these results using the logical OR-function.

A positive answer ($o = 1$) signals that the element belongs to the outer region (positive region).



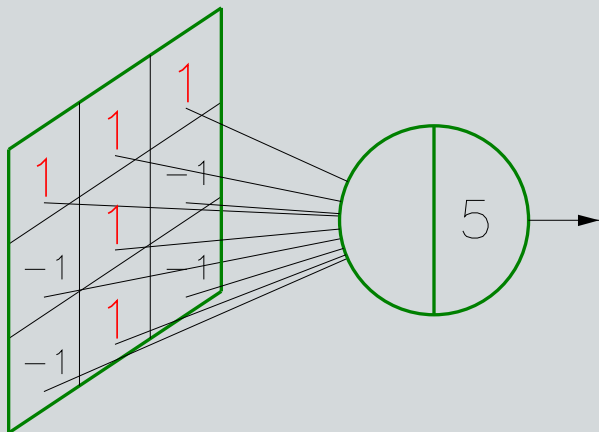
Boundary and Pattern Recognition

Matrix of sensors to recognize boundaries



Boundary and Pattern Recognition

Matrix of sensors to recognize symbols



Logical Functions, Pseudo Inverse

Example (AND OR XOR)

$$Y = \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ -1 & -1 & -1 & -1 \end{pmatrix};$$

$$Z = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

Logical Functions, Pseudo Inverse

Example (AND OR XOR: $Z = 0.5 \leq W \cdot Y$)

$$Y = \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ -1 & -1 & -1 & -1 \end{pmatrix}; \quad Z = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

$$YY^T = \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ -1 & -1 & -1 & -1 \end{pmatrix} \begin{pmatrix} 0 & 0 & -1 \\ 0 & 1 & -1 \\ 1 & 0 & -1 \\ 1 & 1 & -1 \end{pmatrix} = \begin{pmatrix} 2 & 1 & -2 \\ 1 & 2 & -2 \\ -2 & -2 & 4 \end{pmatrix}$$

$$W = ZY^T(YY^T)^{-1} = \begin{pmatrix} 0.5 & 0.5 & 0.25 \\ 0.5 & 0.5 & -0.25 \\ 0 & 0 & -0.5 \end{pmatrix}$$

Logical Functions, Pseudo Inverse

Example (AND OR XOR: $Z = 0.5 \leq W \cdot Y$)

$$Y = \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ -1 & -1 & -1 & -1 \end{pmatrix}; \quad Z = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

$$YY^T = \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ -1 & -1 & -1 & -1 \end{pmatrix} \begin{pmatrix} 0 & 0 & -1 \\ 0 & 1 & -1 \\ 1 & 0 & -1 \\ 1 & 1 & -1 \end{pmatrix} = \begin{pmatrix} 2 & 1 & -2 \\ 1 & 2 & -2 \\ -2 & -2 & 4 \end{pmatrix}$$

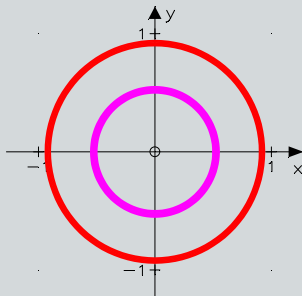
$$W = ZY^T(YY^T)^{-1} = \begin{pmatrix} 0.5 & 0.5 & 0.25 \\ 0.5 & 0.5 & -0.25 \\ 0 & 0 & -0.5 \end{pmatrix}$$

$$N = (\vec{n}_1, \dots, \vec{n}_4) = W \cdot Y = \begin{pmatrix} -0.25 & 0.25 & 0.25 & 0.75 \\ 0.25 & 0.75 & 0.75 & 1.25 \\ 0.50 & 0.50 & 0.50 & 0.50 \end{pmatrix}$$

Support Vector Machine

Not linearly separable data:

Two nested rings are not linearly separable.



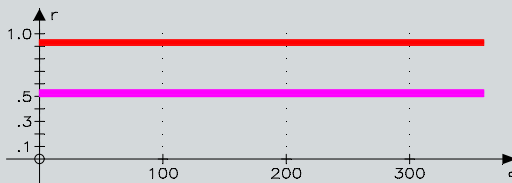
- 1 Transformation in polar coordinates
- 2 Transformation in higher dimensions, e.g. Gaussian

Support Vector Machine

Not linearly separable data: Transformation of data

Two nested rings are not linearly separable.

1 Transformation in polar coordinates



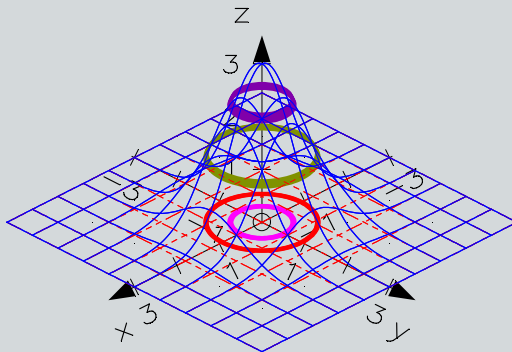
2 Transformation in higher dimensions, e.g. Gaussian

Support Vector Machine

Not linearly separable data: Transformation of data

Two nested rings are not linearly separable.

- 1 Transformation in polar coordinates
- 2 Transformation in higher dimensions, e.g. Gaussian



Support Vector Machine

Not linearly separable data: Transformation of data

Two nested rings are not linearly separable.

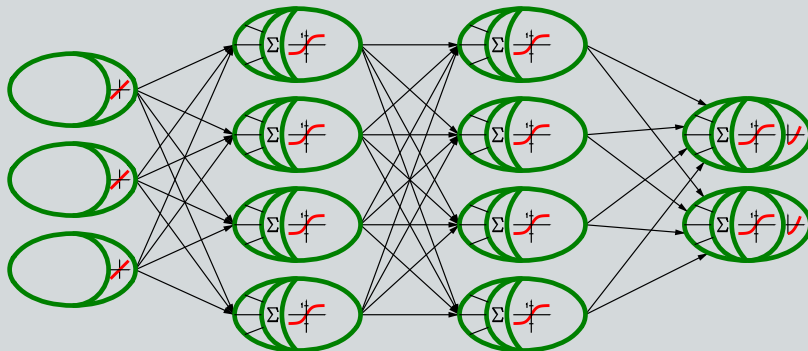
- 1 Transformation in polar coordinates
- 2 Transformation in higher dimensions, e.g. Gaussian

Support Vector Machines

Search for a transformation which allows a linear separation.

Multi-Layered Feed Forward Networks

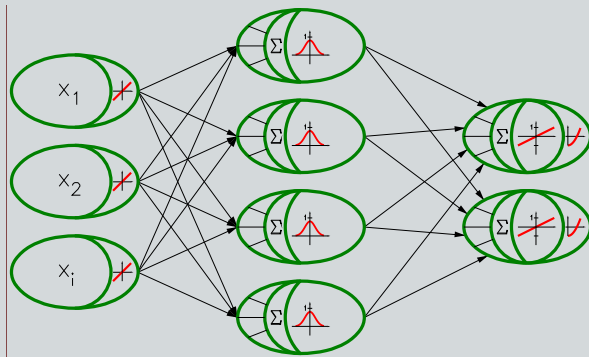
Feed forward network with topology 3-4-4-2



Learning: Change weights and threshold until the result satisfies.

Multi-Layered Feed Forward Networks

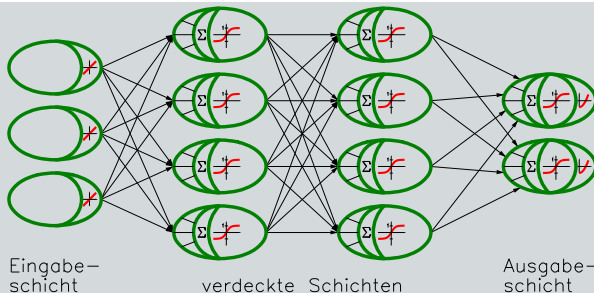
Feed forward network with topology 3-4-4-2



Learning: Change weights and threshold until the result satisfies.

Multi-Layered Feed Forward Networks

Feed forward network with topology 3-4-4-2



```

r←s Bpforw ein;anzs;aus;is;net
anzs←↑ρbpan◇aus←net←bpanP''0◇aus[1]←cQeinl''cφbpte      a Eing. trans.
is←1
DO4:→(anzs<is+is+1)/UNDO4      a Schleife über Schichten: Ausgaben
aus[is]←c1+1+*-bpap×>net[is]←c(is>bpbj)+[1](r>bpgw)+.×(r←is-1)◇aus◇→DO4
UNDO4:r←Q(anzs>aus)l''cφbpta
  
```


Topology of a Feed-Forward Network

Theorem (Kolmogorov, 1957)

Every vector-valued function $f : [0, 1]^n \rightarrow \mathbb{R}^m$ can be written as a 3-layer feed-forward network with n input neurons, $2n + 1$ hidden neurons and m output neurons. The activation functions depend on f and n .

Remark

- 1** *The proof shows the existence in a non-constructive way.*
- 2** *It does not give the activation functions.*
- 3** *The theorem has no direct practical impact.*

Topology of a Feed-Forward Network

Theorem (Kolmogorov, 1957)

Every vector-valued function $f : [0, 1]^n \rightarrow \mathbb{R}^m$ can be written as a 3-layer feed-forward network with n input neurons, $2n+1$ hidden neurons and m output neurons. The activation functions depend on f and n .

Corollary

For every continuous function $f : [-1, 1]^n \rightarrow [-1, 1]$ there are functions g and g_i ($i = 1, \dots, 2n+1$) in one argument and constants λ_j ($j = 1, \dots, n$) with

$$f(x_1, \dots, x_n) = \sum_{i=1}^{2n+1} g \left(\sum_{j=1}^n \lambda_j g_i(x_j) \right) .$$

Topology of a Feed-Forward Network

Theorem (Kolmogorov, 1957)

Every vector-valued function $f : [0, 1]^n \rightarrow \mathbb{R}^m$ can be written as a 3-layer feed-forward network with n input neurons, $2n + 1$ hidden neurons and m output neurons. The activation functions depend on f and n .

Theorem (Approximation with neural networks)

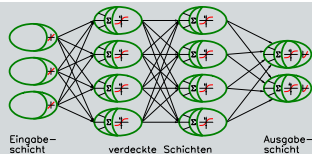
Every function allows an approximation by a neural network with one hidden layer.

Multi-Layered Feed Forward Networks

Input layer

- **Continuous input:**
Linear transformation into $[-1; 1]$
- **Discrete input:**
One neuron per value, transformed onto $-1, 1$

Multi-Layer Network

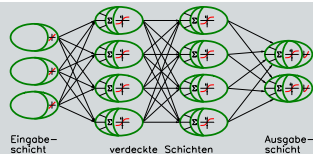


Multi-Layered Feed Forward Networks

Input layer

- **Continuous input:**
Linear transformation into $[-1; 1]$
- **Discrete input:**
One neuron per value, transformed onto $-1, 1$

Multi-Layer Network



Output layer using a tangential activity function

- Target activities should be equally distributed in the interval $[-0.6, 0.6]$!
- The inverse of the output function could be:

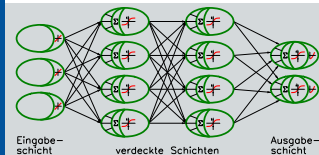
$$f(x) = \left\{ \begin{array}{ll} [m, M] & \rightarrow [-0.6, 0.6] \\ x & \mapsto -0.6 + 1.2 \left(\frac{x-m}{M-m} \right)^s ; \quad s > 0 \end{array} \right\}$$

Multi-Layered Feed Forward Networks

Input layer

- **Continuous input:**
Linear transformation into $[-1; 1]$
- **Discrete input:**
One neuron per value, transformed onto $-1, 1$

Multi-Layer Network



Output layer using a logarithmic activity function

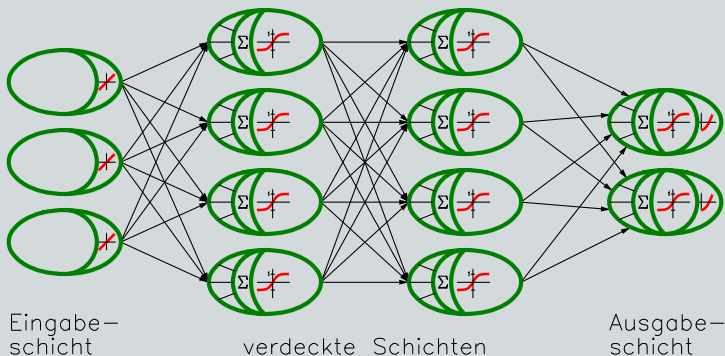
- Target activities should be equally distributed in the interval $[0.2, 0.8]$!
- The inverse of the output function could be:

$$f(x) = \left\{ \begin{array}{ll} [m, M] & \rightarrow [0.2, 0.8] \\ x & \mapsto 0.2 + 0.6 \left(\frac{x-m}{M-m} \right)^s ; \quad s > 0 \end{array} \right\}$$

Learning in Multi-Layered Networks

Target

Change the weights and thresholds in such a way that the errors in the training data get small.



Learning in Multi-Layered Networks

Target

Change the weights and thresholds in such a way that the errors in the training data get small.

Calculations

error:
$$E(\vec{w}) = \frac{1}{2} \sum_{i=1}^n \|\vec{z}_i - \vec{o}_i(w)\|^2$$

radient:
$$\vec{\text{grad}}_w E(\vec{w}) = \left(\frac{\partial E(\vec{w})}{\partial w_1}, \frac{\partial E(\vec{w})}{\partial w_2}, \dots, \frac{\partial E(\vec{w})}{\partial w_3} \right)$$

Learning in Multi-Layered Networks

Target

Change the weights and thresholds in such a way that the errors in the training data get small.

Calculations

error:
$$E(\vec{w}) = \frac{1}{2} \sum_{i=1}^n \|\vec{z}_i - \vec{o}_i(w)\|^2$$

radient:
$$\vec{\text{grad}}_w E(\vec{w}) = \left(\frac{\partial E(\vec{w})}{\partial w_1}, \frac{\partial E(\vec{w})}{\partial w_2}, \dots, \frac{\partial E(\vec{w})}{\partial w_3} \right)$$

Delta-Rule (Gradient descent)

$$\Delta \vec{w}^{(t)} = -\sigma \vec{\text{grad}}_w E(\vec{w}); \quad \vec{w}^{(t)} = \vec{w}^{(t-1)} + \Delta \vec{w}^{(t)} + \mu \Delta \vec{w}^{(t-1)}$$

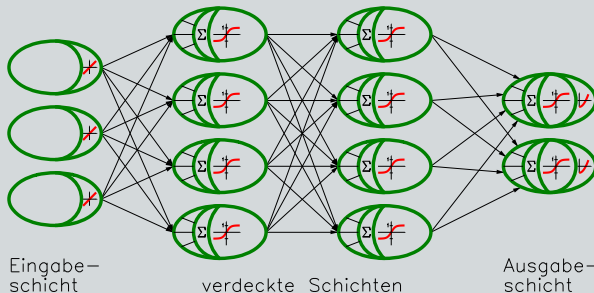
σ decreasing, z.B. von 0.9 auf 0.1, μ increasing, z.B. $\mu = 1 - \sigma$.

Error Back Propagation

Step-by-step error back propagation using the net error δ_i :

$$\vec{\delta}_i := \frac{\partial E}{\partial \vec{n}_i} = \frac{\partial E}{\partial \vec{n}_{i+1}} \cdot \frac{\partial \vec{n}_{i+1}}{\partial \vec{o}_i} \cdot \frac{\partial \vec{o}_i}{\partial \vec{n}_i} = \vec{\delta}_{i+1} \cdot W_{i+1} \cdot A'(\vec{n}_i)$$

$$\frac{\partial E}{\partial W_{i,rs}} = \frac{\partial E}{\partial \vec{n}_i} \cdot \frac{\partial \vec{n}_i}{\partial W_{i,rs}} = \vec{\delta}_i \cdot o_{i-1,s} \hat{e}_r = \delta_{i,r} o_{i-1,s}$$



Error Back Propagation

Step-by-step error back propagation using the net error δ_i :

$$\vec{\delta}_i := \frac{\partial E}{\partial \vec{n}_i} = \frac{\partial E}{\partial \vec{n}_{i+1}} \cdot \frac{\partial \vec{n}_{i+1}}{\partial \vec{o}_i} \cdot \frac{\partial \vec{o}_i}{\partial \vec{n}_i} = \vec{\delta}_{i+1} \cdot W_{i+1} \cdot A'(\vec{n}_i)$$

$$\frac{\partial E}{\partial W_{i,rs}} = \frac{\partial E}{\partial \vec{n}_i} \cdot \frac{\partial \vec{n}_i}{\partial W_{i,rs}} = \vec{\delta}_i \cdot o_{i-1,s} \hat{e}_r = \delta_{i,r} o_{i-1,s}$$

```
r←ziel Bpback aus;anzs;dgwa;err;is;lr
```

```
(aus lr)←aus
```

```
err←bpanρ**0
```

```
dgwa←dgw
```

```
is←anzs←↑bpan
```

```
r←anzs>aus
```

```
err[anzs]←←-2×bpap×(r×1-r)×ziel-r
```

```
DO:→(1>is←is-1)/UNDO
```

```
dgw[is]←←(-(1+is)×err)◦.×r←is>aus
```

```
±(is>1)/'err[is]←←bpap×(r×1-r)×(Qis>bpgw)+.×>err[1+is]' A Nettofehler
```

```
→DO
```

```
UNDO:bpgw←bpgw+lr×dgw+(1-lr)×dgwa
```

```
bpbi←bpbi+(dbi←-lr×err)+(1-lr)×dbi
```

```
r←anzs>err
```

```
A dgw global
```

```
A Anzahl Schichten
```

```
A Fehler letzte Schicht
```

```
A Nettofehler
```

```
A B.P. über alle Sch.
```

```
A ΔGewicht je Schicht
```

```
A Nettofehler
```

```
A Gewichte ändern
```

```
A Bias ändern
```

```
A Fehler in letzter Sch.
```

Learning in Multi-Layered Networks

Levenberg-Marquardt-Method

$$E(\vec{w}) = \frac{1}{2} \langle \vec{f}(\vec{w}), \vec{f}(\vec{w}) \rangle \quad \text{mit} \quad \vec{f}(\vec{w}) = \vec{z} - \vec{o}(\vec{w})$$

Learning in Multi-Layered Networks

Levenberg-Marquardt-Method

$$E(\vec{w}) = \frac{1}{2} \langle \vec{f}(\vec{w}), \vec{f}(\vec{w}) \rangle \quad \text{mit} \quad \vec{f}(\vec{w}) = \vec{z} - \vec{o}(\vec{w})$$

$$\vec{0} = E'(\vec{w}) = \overrightarrow{\text{grad}} E(\vec{w}) = f'^T(\vec{w}) \vec{f}(\vec{w})$$

Learning in Multi-Layered Networks

Levenberg-Marquardt-Method

$$E(\vec{w}) = \frac{1}{2} \langle \vec{f}(\vec{w}), \vec{f}(\vec{w}) \rangle \quad \text{mit} \quad \vec{f}(\vec{w}) = \vec{z} - \vec{o}(\vec{w})$$

$$\vec{0} = E'(\vec{w}) = \overrightarrow{\text{grad}} E(\vec{w}) = f'^T(\vec{w})\vec{f}(\vec{w})$$

$$\begin{aligned} E''(\vec{w}) &= \left(f'^T(\vec{w})\vec{f}(\vec{w}) \right)' = f''^T(\vec{w})\vec{f}(\vec{w}) + f'^T(\vec{w})f'(\vec{w}) \\ &= f'^T(\vec{w})f'(\vec{w}) \quad \text{für} \quad f''^T(\vec{w}) \quad \text{small!} \end{aligned}$$

Learning in Multi-Layered Networks

Levenberg-Marquardt-Method

$$E(\vec{w}) = \frac{1}{2} \langle \vec{f}(\vec{w}), \vec{f}(\vec{w}) \rangle \quad \text{mit} \quad \vec{f}(\vec{w}) = \vec{z} - \vec{o}(\vec{w})$$

$$\vec{0} = E'(\vec{w}) = \overrightarrow{\text{grad}} E(\vec{w}) = f'^T(\vec{w})\vec{f}(\vec{w})$$

$$\begin{aligned} E''(\vec{w}) &= \left(f'^T(\vec{w})\vec{f}(\vec{w}) \right)' = f''^T(\vec{w})\vec{f}(\vec{w}) + f'^T(\vec{w})f'(\vec{w}) \\ &= f'^T(\vec{w})f'(\vec{w}) \quad \text{für} \quad f''^T(\vec{w}) \quad \text{small!} \end{aligned}$$

$$\vec{w}_{k+1} = \vec{w}_k - E''(\vec{w})^{-1}E'(\vec{w})$$

$$\Delta \vec{w} = - \left(f'^T(\vec{w})f'(\vec{w}) \right)^{-1} f'^T(\vec{w})\vec{f}(\vec{w})$$

Learning in Multi-Layered Networks

Levenberg-Marquardt-Method

$$E(\vec{w}) = \frac{1}{2} \langle \vec{f}(\vec{w}), \vec{f}(\vec{w}) \rangle \quad \text{mit} \quad \vec{f}(\vec{w}) = \vec{z} - \vec{o}(\vec{w})$$

$$\vec{0} = E'(\vec{w}) = \overrightarrow{\text{grad}} E(\vec{w}) = f'^T(\vec{w})\vec{f}(\vec{w})$$

$$E''(\vec{w}) = f'^T(\vec{w})f'(\vec{w}) \quad \text{für} \quad f''^T(\vec{w}) \quad \text{small!}$$

$$\vec{w}_{k+1} = \vec{w}_k - E''(\vec{w})^{-1}E'(\vec{w})$$

$$\Delta\vec{w} = - \left(f'^T(\vec{w})f'(\vec{w}) \right)^{-1} f'^T(\vec{w})\vec{f}(\vec{w})$$

System of linear equations to be solved:

$$f'^T(\vec{w})f'(\vec{w})\Delta\vec{w} = -f'^T(\vec{w})\vec{f}(\vec{w})$$

Evaluation of a Trained Network

Error in training data

- maximal error
- mean error
- standard deviation

Evaluation of a Trained Network

Error in training data

- maximal error
- mean error
- standard deviation

Error in testing data

- 20%-40% of available data
- maximal and mean error
- standard deviation

Evaluation of a Trained Network

Error in training data

- maximal error
- mean error
- standard deviation

Insider

- Evaluation of forecasts

Error in testing data

- 20%-40% of available data
- maximal and mean error
- standard deviation

Evaluation of a Trained Network

Error in training data

- maximal error
- mean error
- standard deviation

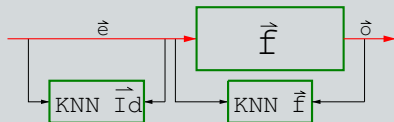
Insider

- Evaluation of forecasts

Error in testing data

- 20%-40% of available data
- maximal and mean error
- standard deviation

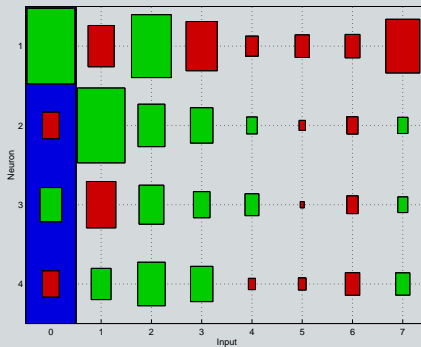
Auto correlation



Improving Learning

Reduction of input parameters

1 Weight in first layer



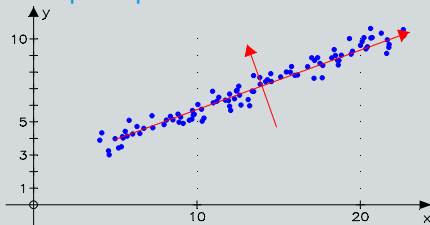
2 Transformation to the principal axes of the covariance matrix

3 Dimensional analysis (equations with physical units only)

Improving Learning

Reduction of input parameters

- 1 Weight in first layer
- 2 Transformation to the principal axes of the covariance matrix



- 3 Dimensional analysis (equations with physical units only)

Improving Learning

Reduction of input parameters

- 1 Weight in first layer
- 2 Transformation to the principal axes of the covariance matrix
- 3 Dimensional analysis (equations with physical units only)

$$u = \frac{5}{384} \frac{ql^4}{EI}$$

$$q[FL^{-1}], l[L], E[FL^{-2}], I[L^4] \quad \text{und} \quad u[L]$$

Improving Learning

Reduction of input parameters

- 1 Weight in first layer
- 2 Transformation to the principal axes of the covariance matrix
- 3 Dimensional analysis (equations with physical units only)

$$u = \frac{5}{384} \frac{ql^4}{EI}$$

$$\pi_1 = \frac{q}{EI} ; \quad \pi_2 = \frac{l^4}{l} ; \quad \pi_3 = \frac{u}{l}$$

Improving Learning

Reduction of input parameters

- 1 Weight in first layer
- 2 Transformation to the principal axes of the covariance matrix
- 3 Dimensional analysis (equations with physical units only)

$$u = \frac{5}{384} \frac{ql^4}{EI}$$

$$\pi_1 = \frac{q}{EI} ; \quad \pi_2 = \frac{l^4}{l} ; \quad \pi_3 = \frac{u}{l} = \frac{5}{384} \cdot \pi_1 \pi_2$$

Improving Learning

Improving quality of results

- 1 Decreasing learning rate, adjusted momentum

$$\begin{aligned}\Delta W_i^{(t)} &:= -\sigma \overrightarrow{\text{grad}}_{W_i} E = -\sigma (\vec{o}_{i-1} \cdot \vec{\delta}_i)^T \\ W_i^{(t)} &= W_i^{(t-1)} - \sigma \overrightarrow{\text{grad}}_{W_i} E + \mu \Delta W_i^{(t-1)}\end{aligned}$$

- 2 Learning – testing
- 3 “Drop out” of some neurons

Improving Learning

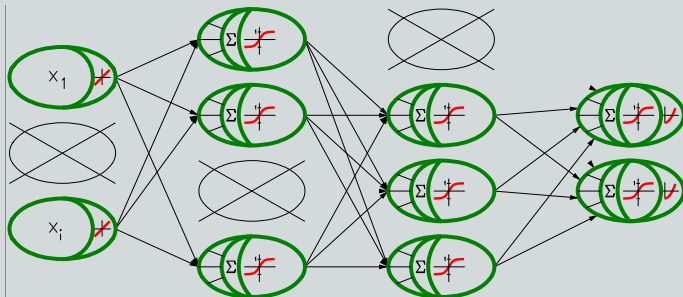
Improving quality of results

- 1 Decreasing learning rate, adjusted momentum
- 2 Learning – testing Stop learning once the error in testing data increases
 - 1 Smoother fitting fo the curve
 - 2 No overlearning
- 3 “Drop out” of some neurons

Improving Learning

Improving quality of results

- 1 Decreasing learning rate, adjusted momentum
- 2 Learning – testing
- 3 “Drop out” of some neurons



1 Analysis, Modelling and Solutions

2 Neurons and Neural Networks

3 Sabbatical Working Examples

- Crash-Tests
- Prediction of Accident Severity
- Learning Strategy
- Comfort in Cabriolet: Active Torsion Damping
- Active Torsion Damping using Neural networks
- Further Examples

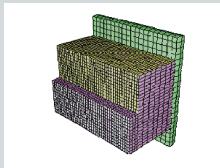
4 Pattern Recognition

5 Neural Networks: Image and Speech Recognition

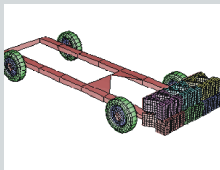
Predicting Impact on Passengers

Sabbatical 1995 (with M. Holzner, R. Stricker)

Barriers



Front crash



Seitenaufprall

Front crash



Foam and honeycomb barriers
(Fa. Fritzmeier)

Predicting Impact on Passengers

Sabbatical 1995 (with M. Holzner, R. Stricker)

Investigating usability of neural networks and fuzzy logic in crash predictions.

- Experimental series of crash tests 0°, 100% overlap, E36
- Target: predicting impact on passengers due to constructive changes
- Problems:
 - No knowledge on correlations
 - Small amount of data sets (90)

Predicting Impact on Passengers

Sabbatical 1995 (with M. Holzner, R. Stricker)

Tasks

- Input parameters and their domains:
 - **Car classification:** version (doors), cylinders, gearing, adjustable steering column?, model year
 - **Airbag:** modell year, exhaust port, ignition point, Young's modulus, volume, mass of explosive
 - **Test data:** place, speed, mass, dummy
 - **Results:** deformation of car, displacement of steering column
- Output parameters:
- Evaluation of neural network:

Predicting Impact on Passengers

Sabbatical 1995 (with M. Holzner, R. Stricker)

Tasks

- Input parameters and their domains:
 - **Car classification:** version (doors), cylinders, gearing, adjustable steering column?, model year
 - **Airbag:** modell year, exhaust port, ignition point, Young's modulus, volume, mass of explosive
 - **Test data:** place, speed, mass, dummy
 - **Results:** deformation of car, displacement of steering column
- Output parameters:
- Evaluation of neural network:

Predicting Impact on Passengers

Sabbatical 1995 (with M. Holzner, R. Stricker)

Tasks

- Input parameters and their domains:
 - **Car classification:** version (doors), cylinders, gearing, adjustable steering column?, model year
 - **Airbag:** modell year, exhaust port, ignition point, Young's modulus, volume, mass of explosive
 - **Test data:** place, speed, mass, dummy
 - **Results:** deformation of car, displacement of steering column
- Output parameters:
 - Dummies (driver and co-driver): HIC, acceleration of head and chest
- Evaluation of neural network:

Predicting Impact on Passengers

Sabbatical 1995 (with M. Holzner, R. Stricker)

Tasks

- Input parameters and their domains:
- Output parameters:
 - Dummies (driver and co-driver): HIC, acceleration of head and chest
- Evaluation of neural network:
 - 80% learning and 20% testing data: **Comparing standard deviation**
 - auto correlation
 - Discussing forecasts of neural network with car engineers.

Predicting Impact on Passengers

Sabbatical 1995 (with M. Holzner, R. Stricker)

Tasks

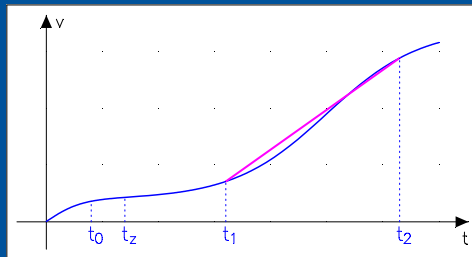
- Input parameters and their domains:
- Output parameters:
 - Dummies (driver and co-driver): HIC, acceleration of head and chest
- Evaluation of neural network:
 - 80% learning and 20% testing data: **Comparing standard deviation**
 - auto correlation
 - Discussing forecasts of neural network with car engineers.

Achievement

We could predict the results of a test.

Accident Severity

with A. Kuhn, J. Urbahn, BMW AG, 2000



t_0 : decision to fire airbag ...

t_z : ignition of airbag
($t_1 - t_z \approx 30$ ms)

t_1 : driver starts forward
displacement

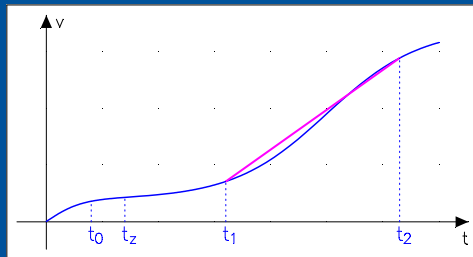
t_2 : acceleration decreases

Targets

- 1 predict the severity of the accident
- 2 help deciding which action to be taken
- 3 protect the passengers as good as possible

Accident Severity

with A. Kuhn, J. Urbahn, BMW AG, 2000



t_0 : decision to fire airbag ...

t_z : ignition of airbag
($t_1 - t_z \approx 30$ ms)

t_1 : driver starts forward
displacement

t_2 : acceleration decreases

Targets

- 1 predict the severity of the accident
- 2 help deciding which action to be taken
- 3 protect the passengers as good as possible

Targets of the Project

Accident severity: possible parameters

- 1 (mean) velocity of passengers (time, forward displacement)
- 2 mean acceleration of passengers

Data base

Data from parameter variations with Monte-Carlo method:

- 1 variation of relevant parameters and testing mode
- 2 FEM simulations using PamCrash
- 3 150 - 300 data sets for every 14 models

Targets of the Project

Accident severity: possible parameters

- 1 (mean) velocity of passengers (time, forward displacement)
- 2 mean acceleration of passengers

Data base

Data from parameter variations with Monte-Carlo method:

- 1 variation of relevant parameters and testing mode
- 2 FEM simulations using PamCrash
- 3 150 - 300 data sets for every 14 models

Targets of the Project

Accident severity: possible parameters

- 1 (mean) velocity of passengers (time, forward displacement)
- 2 mean acceleration of passengers

Data base

Data from parameter variations with Monte-Carlo method:

- 1 variation of relevant parameters and testing mode
- 2 FEM simulations using PamCrash
- 3 150 - 300 data sets for every 14 models

Data from some real crash tests

Targets of the Project

Accident severity: possible parameters

- 1 (mean) velocity of passengers (time, forward displacement)
- 2 mean acceleration of passengers

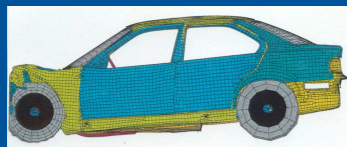
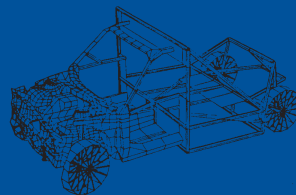
Data base

Data from parameter variations with Monte-Carlo method:

- 1 variation of relevant parameters and testing mode
- 2 FEM simulations using PamCrash
- 3 150 - 300 data sets for every 14 models

Data from some real crash tests

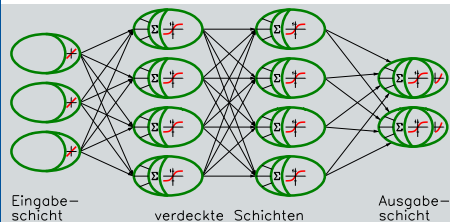
Made possible by



more computer power!

Using the Power of Neural Networks

3- or 4-layer networks



Input

- accelerations, velocities, displacements
- maximal and mean values

Output

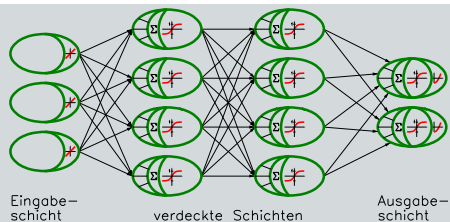
- 1 velocity
- 2 mean acceleration (impact to passengers)

Learning:

- activation function: tangential, piecewise parabola
- learning method: gradient descent, Levenberg-Marquardt

Using the Power of Neural Networks

3- or 4-layer networks



Input

- accelerations, velocities, displacements
- maximal and mean values

Output

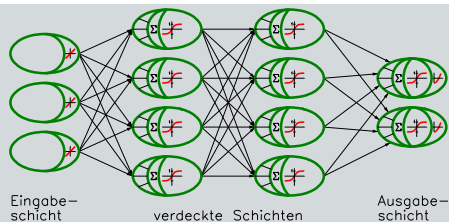
- 1 velocity
- 2 mean acceleration (impact to passengers)

Learning:

- activation function: tangential, piecewise parabola
- learning method: gradient descent, Levenberg-Marquardt

Using the Power of Neural Networks

3- or 4-layer networks



Input

- accelerations, velocities, displacements
- maximal and mean values

Output

- 1 velocity
- 2 mean acceleration (impact to passengers)

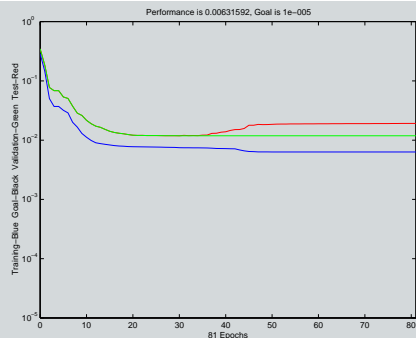
Learning:

- **activation function:** tangential, piecewise parabola
- **learning method:** gradient descent, Levenberg-Marquardt

Training the Networks: Learning Strategy

- random choice of 60% learning, 40% testing data
- stop training when the error in testing data increases

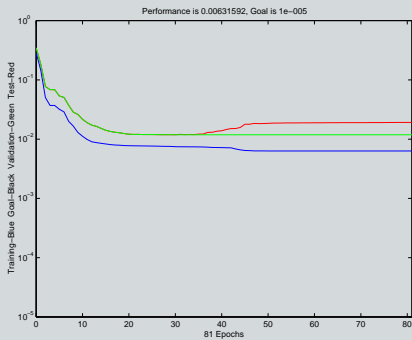
Mean learning error



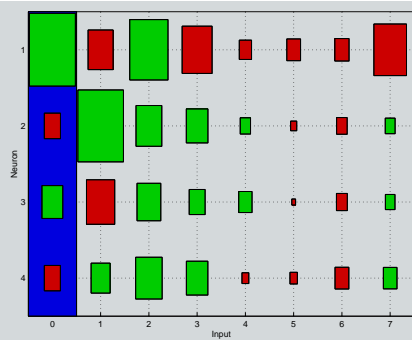
Training the Networks: Influence of the Input

- random choice of 60% learning, 40% testing data
- stop training when the error in testing data increases

Mean learning error

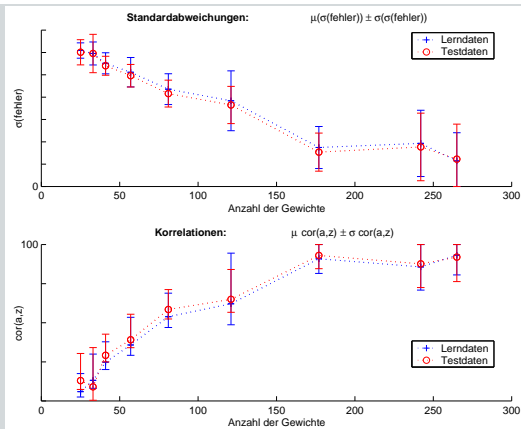


Weights in the first layer



Optimization

Statistics on the number of neurons (1 - 2 hidden layers)



Graphs:

- $\sigma(\sigma(o_i - z_i))$
- correlation

Expectation:

- $\sigma(\sigma)$ gets smaller up to saturation.
- error in learning data only a bit better than testing data

Results

Models

- 1 FEM simulation data gives a **good data base**.
- 2 **Usable topologies**: e.g.: 4-15-8-1, 4-33-1
- 3 **Usable parameter**: mean acceleration
- 4 **Usable input**:
mean accelerations and velocities

σ^2 -method allows:

- to choose a network of an appropriate size.
- to judge on the quality of the data.

Results

Models

- 1 FEM simulation data gives a **good data base**.
- 2 **Usable topologies**: e.g.: 4-15-8-1, 4-33-1
- 3 **Usable parameter**: mean acceleration
- 4 **Usable input**:
mean accelerations and velocities

σ^2 -method allows:

- to choose a network of an appropriate size.
- to judge on the quality of the data.

Active Damping of Torsion

 with Ch. Hornung, G. Pflanz, BMW AG, 2005

Problem of a Cabrio: lack of torsion stiffness M_x/d_y

Limousine: 100 %

Cabrio 7.3%

Origin of Unwanted Vibration

Vibration

Car vibration mainly caused by wheel resonance,

- ⇒ Vibration is transmitted by joints through the axes and spring strut,
- ⇒ Vibration is observed by passengers.

Origin of Unwanted Vibration

Vibration

Car vibration mainly caused by wheel resonance,

- ⇒ Vibration is transmitted by joints through the axes and spring strut,
- ⇒ Vibration is observed by passengers.

Origin of Unwanted Vibration

Vibration

- Car vibration mainly caused by wheel resonance,
- ⇒ Vibration is transmitted by joints through the axes and spring strut,
 - ⇒ Vibration is observed by passengers.

Active Damping: Actuators Produce Counter-Displacement

Sensors and Actuators

- Sensors realize a disturbance
 - Actuators produce opposite displacement
- ⇒ No displacement at the windscreen panel

Active Damping: Actuators Produce Counter-Displacement

Sensors and Actuators

- Sensors realize a disturbance
- Actuators produce opposite displacement

⇒ No displacement at the windscreen panel

Active Damping: Actuators Produce Counter-Displacement

Sensors and Actuators

- Sensors realize a disturbance
- Actuators produce opposite displacement

⇒ No displacement at the windscreen panel

Training

Models

- One or all velocities
- Time series up to 500 ms
- Combination of accelerations

Training of the neural networks

- At least 40% data for testing
- Gradient descent, Levenberg-Marquardt
- Termination: errors in testing data increase

Training

Models

- One or all velocities
- Time series up to 500 ms
- Combination of accelerations

Training of the neural networks

- At least 40% data for testing
- Gradient descent, Levenberg-Marquardt
- **Termination:**
errors in testing data increase

Training and Results

Models

- One or all velocities
- Time series up to 500 ms
- Combination of accelerations

Good results

- time series ca. 200 ms ,
- 2 and 4 input signals,
- both training methods
- small networks \Rightarrow strongly linear behaviour of car body and actuator

Training of the neural networks

- At least 40% data for testing
- Gradient descent, Levenberg-Marquardt
- **Termination:** errors in testing data increase

Validation

Validation

- Integration of trained network into a simulink-model
- Neural network gives slightly better results than a linear control

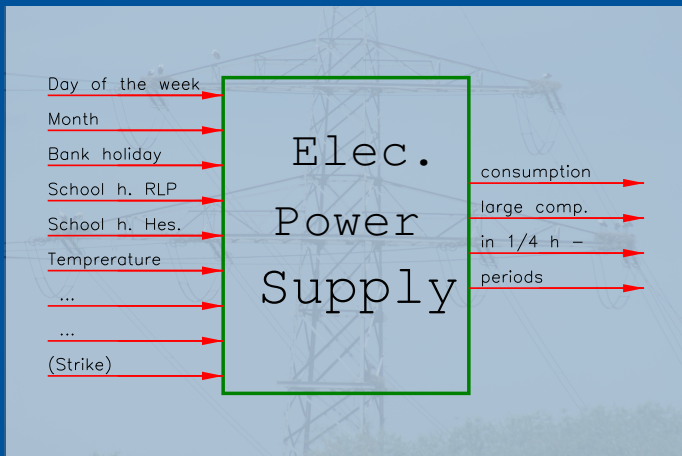
Validation

Validation

- Integration of trained network into a simulink-model
- Neural network gives slightly better results than a linear control

Prediction of Power Consumption

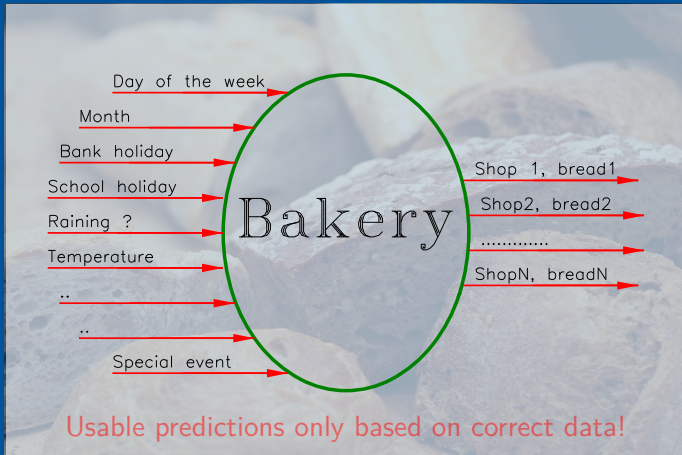
EWR, DA Th. Müller, 2005



Prediction of Sales Figures of Bread

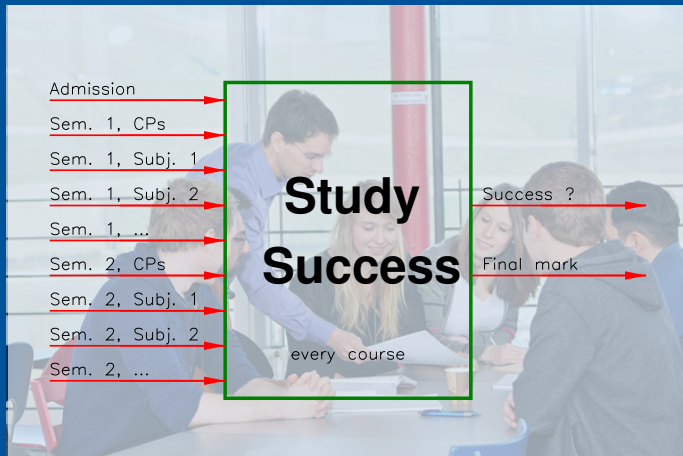


Prediction of Sales Figures of Bread



Prediction of Student Drop Out

TH Bingen (HSP III), 2017/18



Examples

Insolvency Detection

Based on annual reports a forecast on the risk of insolvency is given.

Examples

Controlling vehicles and robots

Neural Network controls a vehicle or robot. It is trained „on the job“.

Insolvency Detection

Based on annual reports a forecast on the risk of insolvency is given.



Examples

Controlling vehicles and robots

Neural Network controls a vehicle or robot. It is trained „on the job“.

Insolvency Detection

Based on annual reports a forecast on the risk of insolvency is given.

Forecast of share value

Forecast based on previous share values and economic data of the company.

Examples



Potential contract termination

Based on power consumption and client data companies that might terminate a contract are identified and get discount.

Insolvency Detection

Based on annual reports a forecast on the risk of insolvency is given.

Forecast of share value

Forecast based on previous share values and economic data of the company.

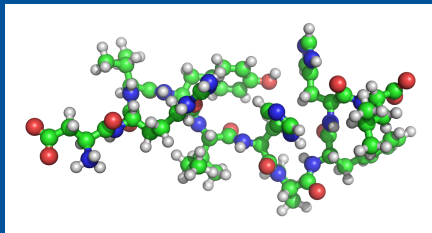
Examples

Chemical reactivity

Prediction of its reactivity from quantitative properties of a bonding.

Insolvency Detection

Based on annual reports a forecast on the risk of insolvency is given.



Forecast of share value

Forecast based on previous share values and economic data of the company.

Examples

Chemical reactivity

Prediction of its reactivity from quantitative properties of a bonding.

Origin of olive oil

The concentration of acids determines the origin (region) of Italian olive oil.

Potential contract termination

Based on power consumption and client data companies that might terminate a contract are identified and get discount.

Forecast of share value

Forecast based on previous share values and economic data of the company.

Examples

Chemical reactivity

Prediction of its reactivity from quantitative properties of a bonding.

Origin of olive oil

The concentration of acids determines the origin (region) of Italian olive oil.

Olfaktometer

Micro crystal system with six different piezo-electric crystal sensors: A neural network learns to recognize flavours.

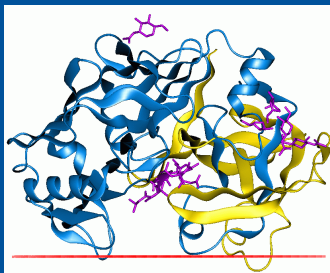
Forecast of share value

Forecast based on previous share values and economic data of the company.

Examples

Chemical reactivity

Prediction of its reactivity from quantitative properties of a bonding.



Olfaktometer

Micro crystal system with six different piezo-electric crystal sensors: A neural network learns to recognize flavours.

Structure of a protein

Conclusion from the primary structure of a protein to its secondary spacial structure.

Examples

Breaking torque

Determining the breaking torque from hydraulic pressure and velocity.



Olfaktometer

Micro crystal system with six different piezo-electric crystal sensors: A neural network learns to recognize flavours.

Structure of a protein

Conclusion from the primary structure of a protein to its secondary spatial structure.

Examples

Breaking torque

Determining the breaking torque from hydraulic pressure and velocity.

Neural stetoscope

A neural networks interprets the noise coming through a stethoscope and provides a diagnoses of a heart problem.

Olfaktometer

Micro crystal system with six different piezo-electric crystal sensors: A neural network learns to recognize flavours.



- 1 Analysis, Modelling and Solutions
- 2 Neurons and Neural Networks
- 3 Sabbatical Working Examples
- 4 **Pattern Recognition**
 - Kohonen Feature Maps: TSP
 - Counterpropagation
 - Hierarchical Classification
- 5 Neural Networks: Image and Speech Recognition
- 6 Conclusion

Kohonen Feature Maps

Our brain

- organizes itself,
- maps a sensation field onto the cortex,
- builds a map of sensation areas.

Transformation into a network architecture

An input changes the weights such that the Kohonen feature map learns the topological structure of the task and maps it.

Structure

- Input layer
- Topological layer: Every neuron has a fixed position.

Kohonen Feature Maps

Our brain

- organizes itself,
- maps a sensation field onto the cortex,
- builds a map of sensation areas.

Transformation into a network architecture

An input changes the weights such that the Kohonen feature map learns the topological structure of the task and maps it.

Structure

- Input layer
- Topological layer: Every neuron has a fixed position.

Kohonen Feature Maps

Our brain

- organizes itself,
- maps a sensation field onto the cortex,
- builds a map of sensation areas.

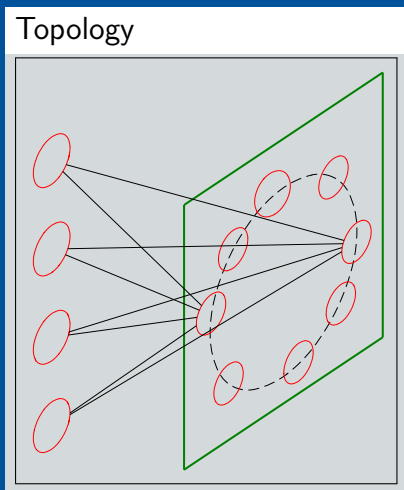
Transformation into a network architecture

An input changes the weights such that the Kohonen feature map learns the topological structure of the task and maps it.

Structure

- Input layer
- Topological layer: Every neuron has a fixed position.

Kohonen Feature Maps



Learning strategy

- 1 "Winner takes all"
- 2 Neurons in its neighbourhood update weights (decreasing radius)

Kohonen Feature Maps: Learning Strategy

Learning algorithm

1 Shrinking radius of neighbourhood:

$$\sigma(t) := \sigma_{\min}^{t/t_{\max}} \cdot \sigma_{\max}^{1-t/t_{\max}}$$

2 Decreasing learning rate:

$$\varepsilon(t) := \varepsilon_{\min}^{t/t_{\max}} \cdot \varepsilon_{\max}^{1-t/t_{\max}}$$

3 Weights change in neuron i closed zu winner j ($d(i, j) \leq \sigma(t)$):

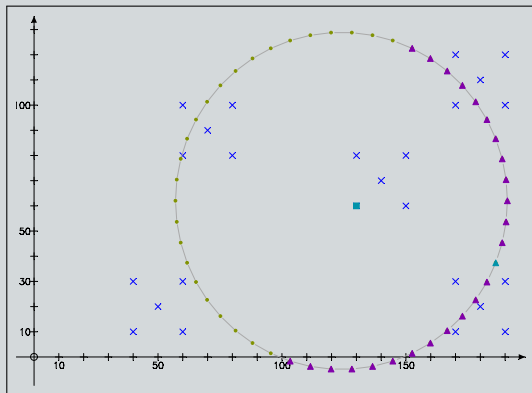
$$w_{ik}(t+1) = w_{ik}(t) + \varepsilon(t) \cdot e^{-d(i,j)^2/2\sigma(t)^2} \cdot (e_k - \vec{w}_{ik}(t))$$

4 At the end of learning:

Every input value will be identified sensation center.

Kohonen Feature Maps

Initialization

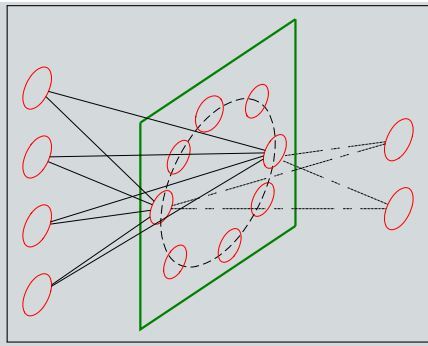




Schach6, Platine2

Counterpropagation Network

Topology

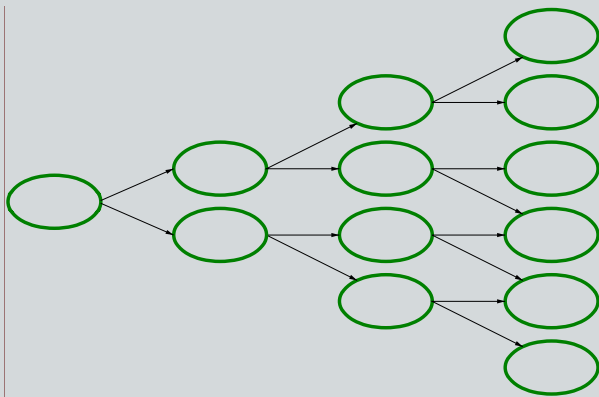


Learning strategy

- 1 “Winner takes all”
- 2 Neurons in its neighbourhood update weights (decreasing radius)

Hierarchical Classification

Refining pattern recognition



- 1 Analysis, Modelling and Solutions
- 2 Neurons and Neural Networks
- 3 Sabbatical Working Examples
- 4 Pattern Recognition
- 5 Neural Networks: Image and Speech Recognition**
 - Big Data and Deep Learning
 - Recurrent Neural Networks
 - Convolutional neural networks
- 6 Conclusion

Big Data and Deep Learning

The amount of data is huge

- images, sound, ultrasonic sound, laser - and radar signals
- videos of traffic situations, real and simulated
- glyphs and hand writing
- learning – productive situation

... and learning is deep

- Neural networks with million of neurons an billions of connections (weights)
- GPUs (graphical processing unit)

Big Data and Deep Learning

The amount of data is huge

- images, sound, ultrasonic sound, laser - and radar signals
- videos of traffic situations, real and simulated
- glyphs and hand writing
- learning – productive situation

... and learning is deep

- Neural networks with million of neurons an billions of connections (weights)
- GPUs (graphical processing unit)

Big Data and Deep Learning

The amount of data is huge

- images, sound, ultrasonic sound, laser - and radar signals
- videos of traffic situations, real and simulated
- glyphs and hand writing
- learning – productive situation

... and learning is deep

- Neural networks with million of neurons and billions of connections (weights)
- GPUs (graphical processing unit)

Big Data and Deep Learning

The amount of data is huge

- images, sound, ultrasonic sound, laser - and radar signals
- videos of traffic situations, real and simulated
- glyphs and hand writing
- learning – productive situation

... and learning is deep

- Neural networks with million of neurons and billions of connections (weights)
- GPUs (graphical processing unit)

Big Data and Deep Learning

The amount of data is huge

- images, sound, ultrasonic sound, laser - and radar signals
- videos of traffic situations, real and simulated
- glyphs and hand writing
- learning – productive situation

... and learning is deep

- Neural networks with million of neurons an billions of connections (weights)
- GPUs (graphical processing unit)

Big Data and Deep Learning

The amount of data is huge

- images, sound, ultrasonic sound, laser - and radar signals
- videos of traffic situations, real and simulated
- glyphs and hand writing
- learning – productive situation

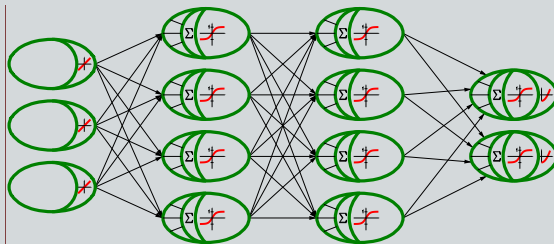
... and learning is deep

- Neural networks with million of neurons an billions of connections (weights)
- GPUs (graphical processing unit)

Big Data and Deep Learning

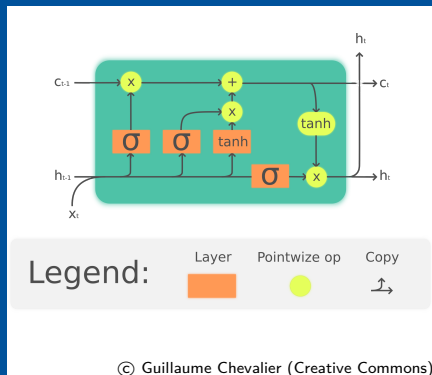
... and learning is deep

- Neural networks with million of neurons and billions of connections (weights)
- GPUs (graphical processing unit)



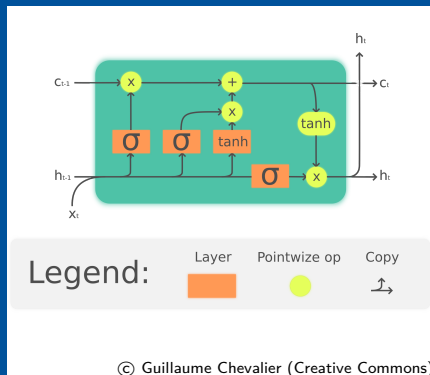
Hierarchical Feed forward network

Long Short Term Memory (Recurrent Network)



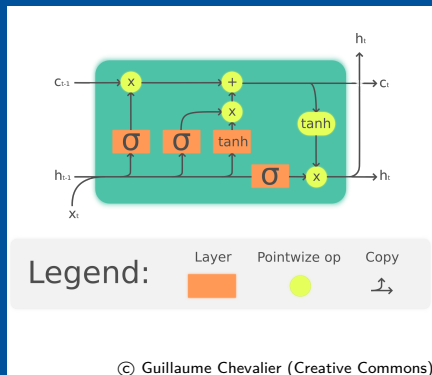
- 1 State and output of each neuron is stored.
- 2 State may be changed or deleted.
- 3 State controls output.

Long Short Term Memory (Recurrent Network)



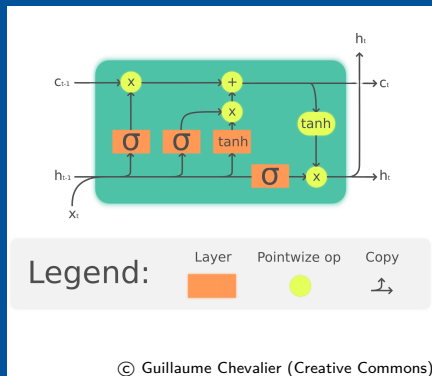
- 1 State and output of each neuron is stored.
- 2 State may be changed or deleted.
- 3 State controls output.

Long Short Term Memory (Recurrent Network)



- 1 State and output of each neuron is stored.
- 2 State may be changed or deleted.
- 3 State controls output.

Long Short Term Memory (Recurrent Network)



- 1 State and output of each neuron is stored.
- 2 State may be changed or deleted.
- 3 State controls output.

This allows a temporary memory.

Long Short Term Memory (Recurrent Network)

Success of complete LSTM

- 1 since 2003: Recognition of speech and glyphs
- 2 2007: Recognition of key words
- 3 Winner of competitions:
 - 2009: Recognition of French handwriting
 - 2014: Recognition of arabic handwriting
 - ...
- 4 Farsi, Chinese
- 5 Further Recognition of key words
- 6 Speech recognition with LSTM: newest version of google voice
- 7 Description of video clips

Long Short Term Memory (Recurrent Network)

Success of complete LSTM

- 1 since 2003: Recognition of speech and glyphs
- 2 2007: Recognition of key words
- 3 Winner of competitions:
 - 2009: Recognition of French handwriting
 - 2014: Recognition of arabic handwriting
 - ...
- 4 Farsi, Chinese
- 5 Further Recognition of key words
- 6 Speech recognition with LSTM: newest version of google voice
- 7 Description of video clips

Long Short Term Memory (Recurrent Network)

Success of complete LSTM

- 1 since 2003: Recognition of speech and glyphs
- 2 2007: Recognition of key words
- 3 Winner of competitions:
 - 2009: Recognition of French handwriting
 - 2014: Recognition of arabic handwriting
 - ...
- 4 Farsi, Chinese
- 5 Further Recognition of key words
- 6 Speech recognition with LSTM: newest version of google voice
- 7 Description of video clips

Long Short Term Memory (Recurrent Network)

Success of complete LSTM

- 1 since 2003: Recognition of speech and glyphs
- 2 2007: Recognition of key words
- 3 Winner of competitions:
 - 2009: Recognition of French handwriting
 - 2014: Recognition of arabic handwriting
 - ...
- 4 Farsi, Chinese
- 5 Further Recognition of key words
- 6 Speech recognition with LSTM: newest version of google voice
- 7 Description of video clips

Long Short Term Memory (Recurrent Network)

Success of complete LSTM

- 1 since 2003: Recognition of speech and glyphs
- 2 2007: Recognition of key words
- 3 Winner of competitions:
 - 2009: Recognition of French handwriting
 - 2014: Recognition of arabic handwriting
 - ...
- 4 Farsi, Chinese
- 5 Further Recognition of key words
- 6 Speech recognition with LSTM: newest version of google voice
- 7 Description of video clips

Long Short Term Memory (Recurrent Network)

Success of complete LSTM

- 1 since 2003: Recognition of speech and glyphs
- 2 2007: Recognition of key words
- 3 Winner of competitions:
 - 2009: Recognition of French handwriting
 - 2014: Recognition of arabic handwriting
 - ...
- 4 Farsi, Chinese
- 5 Further Recognition of key words
- 6 Speech recognition with LSTM: newest version of google voice
- 7 Description of video clips

Long Short Term Memory (Recurrent Network)

Success of complete LSTM

- 1 since 2003: Recognition of speech and glyphs
- 2 2007: Recognition of key words
- 3 Winner of competitions:
 - 2009: Recognition of French handwriting
 - 2014: Recognition of arabic handwriting
 - ...
- 4 Farsi, Chinese
- 5 Further Recognition of key words
- 6 Speech recognition with LSTM: newest version of google voice
- 7 Description of video clips

Long Short Term Memory (Recurrent Network)

Success of complete LSTM

- 1 since 2003: Recognition of speech and glyphs
- 2 2007: Recognition of key words
- 3 Winner of competitions:
 - 2009: Recognition of French handwriting
 - 2014: Recognition of arabic handwriting
 - ...
- 4 Farsi, Chinese
- 5 Further Recognition of key words
- 6 Speech recognition with LSTM: newest version of google voice
- 7 Description of video clips

Long Short Term Memory (Recurrent Network)

Success of complete LSTM

- 1 since 2003: Recognition of speech and glyphs
- 2 2007: Recognition of key words
- 3 Winner of competitions:
 - 2009: Recognition of French handwriting
 - 2014: Recognition of arabic handwriting
 - ...
- 4 Farsi, Chinese
- 5 Further Recognition of key words
- 6 **Speech recognition with LSTM: newest version of google voice**
- 7 Description of video clips

Long Short Term Memory (Recurrent Network)

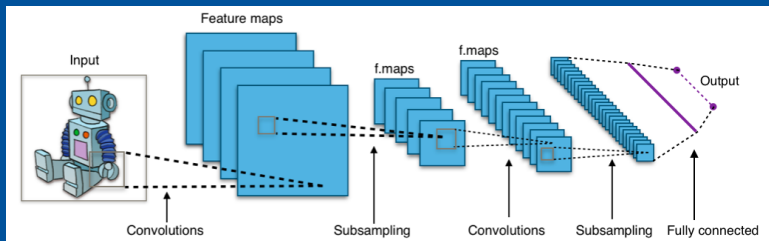
Success of complete LSTM

- 1 since 2003: Recognition of speech and glyphs
- 2 2007: Recognition of key words
- 3 Winner of competitions:
 - 2009: Recognition of French handwriting
 - 2014: Recognition of arabic handwriting
 - ...
- 4 Farsi, Chinese
- 5 Further Recognition of key words
- 6 Speech recognition with LSTM: newest version of google voice
- 7 [Description of video clips](#)

Long Short Term Memory (Recurrent Network)

Google, 2014: Zu Bildern beschreibenden Text hinzufügen: Bild in Vinyals,
Toshev, et al., "Show and Tell: Lessons Learned from the 2015 MSCOCO Image Captioning Challenge", S. 659 ..

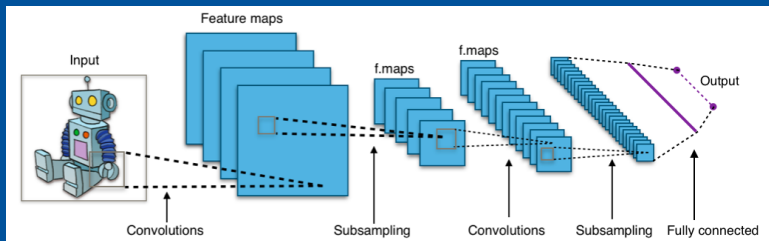
Hierarchical Neural Network (Convolutional NN)



© Aphex34 (Creative Commons)

- 1 Analyse parts of the image: recognise boundaries
- 2 Evaluate results: take most reasonable features
- 3 Analyse some features: recognise patterns
- 4 Evaluate results: take most reasonable pattern
- 5 Recognise situation (complete neural network)

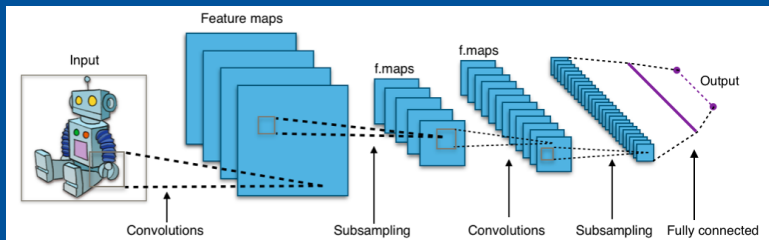
Hierarchical Neural Network (Convolutional NN)



© Aphex34 (Creative Commons)

- 1 Analyse parts of the image: recognise boundaries
- 2 Evaluate results: take most reasonable features
- 3 Analyse some features: recognise patterns
- 4 Evaluate results: take most reasonable pattern
- 5 Recognise situation (complete neural network)

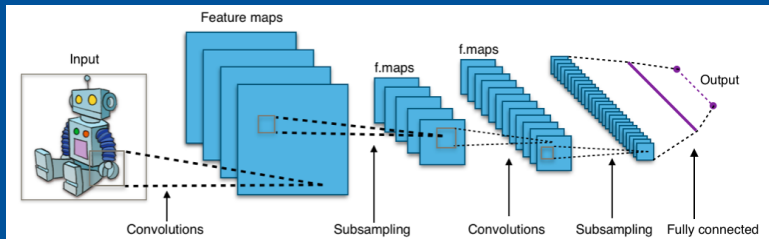
Hierarchical Neural Network (Convolutional NN)



© Aphex34 (Creative Commons)

- 1 Analyse parts of the image: recognise boundaries
- 2 Evaluate results: take most reasonable features
- 3 Analyse some features: recognise patterns
- 4 Evaluate results: take most reasonable pattern
- 5 Recognise situation (complete neural network)

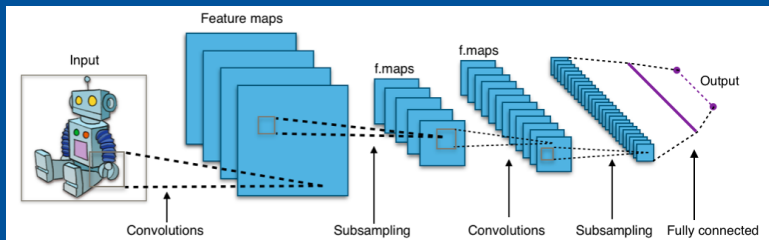
Hierarchical Neural Network (Convolutional NN)



© Aphex34 (Creative Commons)

- 1 Analyse parts of the image: recognise boundaries
- 2 Evaluate results: take most reasonable features
- 3 Analyse some features: recognise patterns
- 4 Evaluate results: take most reasonable pattern
- 5 Recognise situation (complete neural network)

Hierarchical Neural Network (Convolutional NN)



© Aphex34 (Creative Commons)

- 1 Analyse parts of the image: recognise boundaries
- 2 Evaluate results: take most reasonable features
- 3 Analyse some features: recognise patterns
- 4 Evaluate results: take most reasonable pattern
- 5 Recognise situation (complete neural network)

Big Data and Deep Learning

Very large CNN (convolutional neural network)

- Parts of the network recognise different features in parallel sections
- Features are gathered and used for further learning

Bild in ..

Big Data and Deep Learning

Very large CNN (convolutional neural network)

- Parts of the network recognise different features in parallel sections
- Features are gathered and used for further learning

Bild in ..

Big Data and Deep Learning

Very large CNN (convolutional neural network)

- Parts of the network recognise different features in parallel sections
- Features are gathered and used for further learning

Success of CNNs

- 2009: Highly improving speech recognition (Yann LeCun, Geoffrey Hinton, George Dahl; Toronto)
- 2012: ImageNet competition: After training with on million images, the NN must recognise the situation of the image: Dropping error rate: 25% ↓ 15%
- 2013: All competitors use this method.
- 2013: Merck: Who has a program which predicts the impact of 30 000 small molecules on 15 target molecules.
Winner: George Dahl using CNN, better by 15%.
- Since then new areas of applications:
recognition of languages, translation, weather forecast

Big Data and Deep Learning

Very large CNN (convolutional neural network)

- Parts of the network recognise different features in parallel sections
- Features are gathered and used for further learning

Success of CNNs

- 2009: Highly improving speech recognition (Yann LeCun, Geoffrey Hinton, George Dahl; Toronto)
- 2012: ImageNet competition: After training with on million images, the NN must recognise the situation of the image: Dropping error rate: 25% ↓ 15%
- 2013: All competitors use this method.
- 2013: Merck: Who has a program which predicts the impact of 30 000 small molecules on 15 target molecules.
Winner: George Dahl using CNN, better by 15%.
- Since then new areas of applications:
recognition of languages, translation, weather forecast

Big Data and Deep Learning

Very large CNN (convolutional neural network)

- Parts of the network recognise different features in parallel sections
- Features are gathered and used for further learning

Success of CNNs

- 2009: Highly improving speech recognition (Yann LeCun, Geoffrey Hinton, George Dahl; Toronto)
- 2012: ImageNet competition: After training with on million images, the NN must recognise the situation of the image: **Dropping error rate: 25% ↓ 15%**
- **2013: All competitors use this method.**
- 2013: Merck: Who has a program which predicts the impact of 30 000 small molecules on 15 target molecules.
Winner: George Dahl using CNN, better by 15%.
- Since then new areas of applications:
recognition of languages, translation, weather forecast

Big Data and Deep Learning

Very large CNN (convolutional neural network)

- Parts of the network recognise different features in parallel sections
- Features are gathered and used for further learning

Success of CNNs

- 2009: Highly improving speech recognition (Yann LeCun, Geoffrey Hinton, George Dahl; Toronto)
- 2012: ImageNet competition: After training with on million images, the NN must recognise the situation of the image: Dropping error rate: 25% ↓ 15%
- 2013: All competitors use this method.
- 2013: Merck: Who has a program which predicts the impact of 30 000 small molecules on 15 target molekules.
Winner: George Dahl using CNN, better by 15%.
- Since then new areas of applications:
recognition of languages, translation, weather forecast

Big Data and Deep Learning

Very large CNN (convolutional neural network)

- Parts of the network recognise different features in parallel sections
- Features are gathered and used for further learning

Success of CNNs

- 2009: Highly improving speech recognition (Yann LeCun, Geoffrey Hinton, George Dahl; Toronto)
- 2012: ImageNet competition: After training with on million images, the NN must recognise the situation of the image: **Dropping error rate: 25% ↓ 15%**
- 2013: **All competitors use this method.**
- 2013: Merck: Who has a program which predicts the impact of 30 000 small molecules on 15 target molekules.
Winner: George Dahl using CNN, better by 15%.
- **Since then new areas of applications:**
recognition of languages, translation, weather forecast

Uber-Accident

Autonomous Uber-car overlooks bicyclist

18.3.2018

- 1 Bicyclist crosses a road at night.
- 2 The car does not take her into account.
- 3 The driver realizes this situation too late.
- 4 The bicyclist dies in hospital.

Uber-Accident

Autonomous Uber-car overlooks bicyclist

18.3.2018

- 1 Bicyclist crosses a road at night.
- 2 The car does not take her into account.
- 3 The driver realizes this situation too late.
- 4 The bicyclist dies in hospital.

Uber-Accident

Autonomous Uber-car overlooks bicyclist

18.3.2018

- 1 Bicyclist crosses a road at night.
- 2 The car does not take her into account.
- 3 The driver realizes this situation too late.
- 4 The bicyclist dies in hospital.

Uber-Accident

Autonomous Uber-car overlooks bicyclist

18.3.2018

- 1 Bicyclist crosses a road at night.
- 2 The car does not take her into account.
- 3 The driver realizes this situation too late.
- 4 The bicyclist dies in hospital.

Uber-Accident

Autonomous Uber-car overlooks bicyclist

18.3.2018

Official responses

- Uber pays a compensation to the family of the victim.
- The governor of Arizona stops all licenses of all autonomous Uber cars.
- Nvidia stops all tests with its autonomous cars..

Uber-Accident

Autonomous Uber-car overlooks bicyclist

18.3.2018

Official responses

The Information

7.5.2018

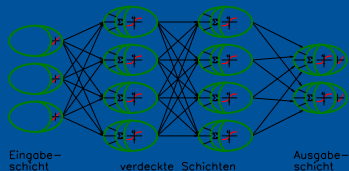
Uber has determined that the likely cause ... was a problem with the software that decides how the car should react to objects it detects, according to two people briefed about the matter.

The car's sensors detected the pedestrian, who was crossing the street with a bicycle, but Uber's software decided it didn't need to react right away. That's a result of how the software was tuned. Like other autonomous vehicle systems, Uber's software has the ability to ignore "false positives", or objects in its path that wouldn't actually be a problem for the vehicle, such as a plastic bag floating over a road. In this case, Uber executives believe the company's system was tuned so that it reacted less to such objects. But the tuning went too far, and the car didn't react fast enough, one of these people said.

Conclusion

Neuronal networks are able to

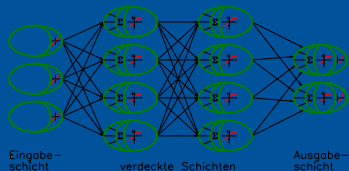
- learn and store know how of a system,
- map functional dependencies



Conclusion

Neuronal networks are able to

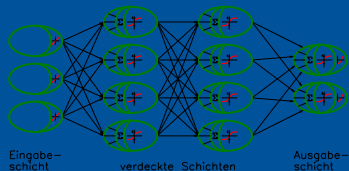
- learn and store know how of a system,
- map functional dependencies



Conclusion

Neuronal networks are able to

- learn and store know how of a system,
- map functional dependencies

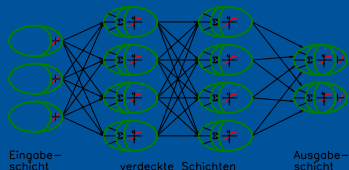


Conclusion

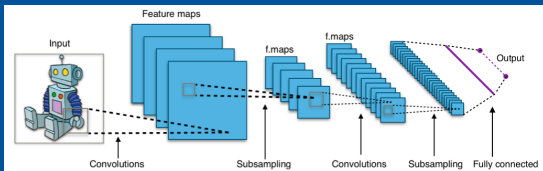
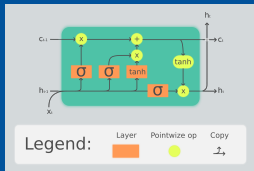
Neuronal networks are able to

- learn and store know how of a system,
- map functional dependencies

using a **smooth or balancing interpolation** between sampling points.



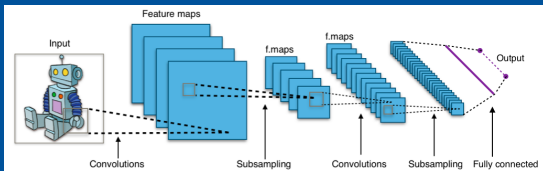
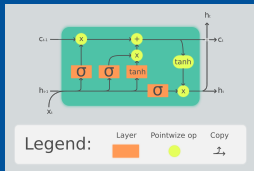
Conclusion



Big neural networks learn to recognise

- 1 hand writing,
- 2 speech,
- 3 situations and actions,
- 4 ...

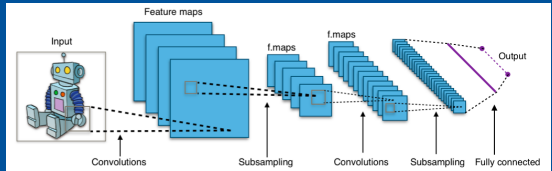
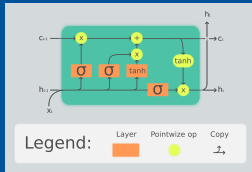
Conclusion



Big neural networks learn to recognise

- 1 hand writing,
- 2 speech,
- 3 situations and actions,
- 4 ...

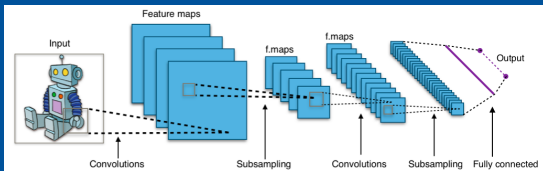
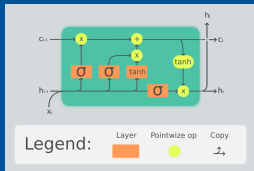
Conclusion



Big neural networks learn to recognise

- 1 hand writing,
- 2 speech,
- 3 situations and actions,
- 4 ...

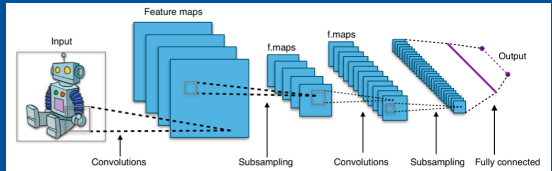
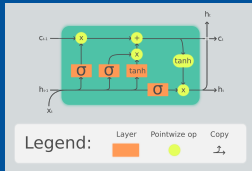
Conclusion



Big neural networks learn to recognise

- 1 hand writing,
- 2 speech,
- 3 situations and actions,
- 4 ...

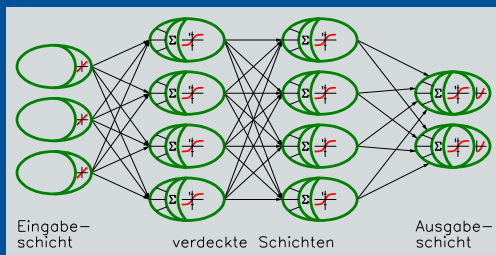
Conclusion



Big neural networks learn to recognise

- 1 hand writing,
- 2 speech,
- 3 situations and actions,
- 4 ...

They are able to support human beings in many areas!



Thank you for listening to my talk!

Literatur I



Fay, R. „Hierachische neuronale Netze“. Masterarbeit. Universität Ulm, 2002.
www.informatik.uni-ulm/.../papers/masterthesis_fay_2002.pdf (besucht 12.09.2018).



Grünter, T. **Deep Learning: Im „Kopf“ von künstlichen neuronalen Netzen.** 2016.
www.spektrum.de/video/im-kopf-von-kuenstlichen-neuronalen-netzen/1586616 (besucht 12.09.2018).



Helmig, C. **HERE HD Live Map für vernetzte Smart Cars.** HERE: www.here.com;
www.mobilegeeks.de/?? 1. Sep. 2016. https://www.youtube.com/watch?v=R81_wg6gswA
 (besucht 11.10.2017).



JAAI. **Convolutional Neural Networks – Aufbau, Funktion und Anwendungsgebiete.**
 27. Aug. 2015. jaai.de/convolutional-neural-networks-cnn-aufbau-funktion-und-anwendungsgebiete-1691/
 (besucht 12.09.2018).



Jones, N. “The Learning Machines”. In: **Nature** 505 (Jan. 9, 2014).
www.nvidia.com/content/tesla/pdf/machine-learning/nature-learning-machines.pdf (besucht 10/17/2017).



Khôi Nguyen, N. **AI detectives are cracking open the black box of deep learning.**
 July 6, 2017. <https://www.spektrum.de/video/im-kopf-von-kuenstlichen-neuronalen-netzen/1586616>
 (besucht 09/12/2018).

Literatur II



LeCunn, Y., Y. Bengio, and G. Hinton. "Deep Learning". In: **Nature** 521 (May 28, 2015). DOI: 10.1038/nature14539. www.cs.toronto.edu/~hinton/absps/NatureDeepReview.pdf (besucht 10/17/2017).



Olah, C. **Conv Nets: A Modular Perspective**. July 8, 2014. colah.github.io/posts/2014-07-Conv-Nets-Modular/ (besucht 09/12/2018).



– **.Understanding LSTM Networks**. Aug. 27, 2015. colah.github.io/posts/2015-08-Understanding-LSTMs/ (besucht 09/12/2018).



Schmidhuber, J. **Künstliche Intelligenz wird alles ändern**. 2016. www.youtube.com/watch?v=rafhHIQgd2A&t=690s (besucht 12.09.2018).



Vinyals, O., A. Toshev, et al. "Show and Tell: Lessons Learned from the 2015 MSCOCO Image Captioning Challenge". In: **IEEE Transactions on Pattern Analysis and Machine Intelligence** 39.4 (2017), pp. 652–663. www.computer.org/csdl/trans/tp/2017/04/07505636.pdf.



von Hugo, C. **Autonomes Fahren – Mehr als nur ein Hype**. (IAA 2017 Future Talk). 18. Sep. 2017. www.youtube.com/watch?time_continue=62&v=8GzE0toDc24 (besucht 11.10.2017).

Zur Vertiefung

Filme

- 1 Schmidhuber: Künstliche Intelligenz wird alles ändern
- 2 von Hugo: Autonomes Fahren – Mehr als nur ein Hype. (IAA 2017 Future Talk)
- 3 Helmig: HERE HD Live Map für vernetzte Smart Cars
- 4 Khoi Nguyen: AI detectives are cracking open the black box of deep learning