

AI | Theory

Tobias Grab

Vorstellung



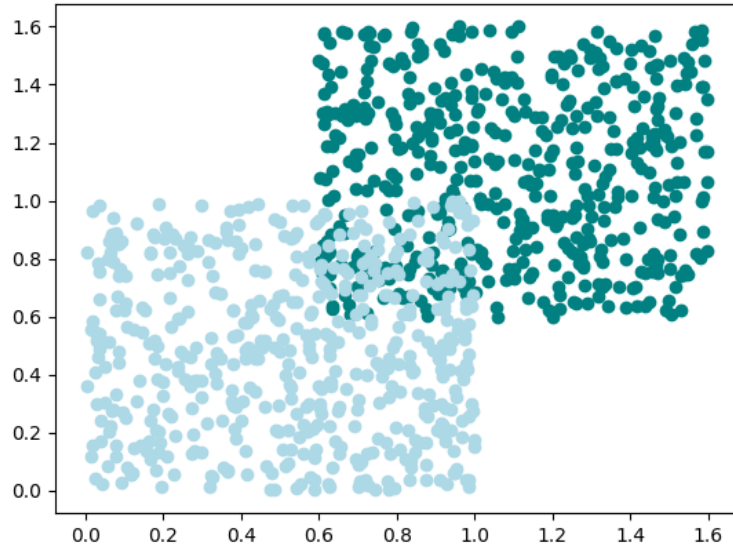
- **Semesterarbeit:** AI for Accessing Research-Based Knowledge in Nursing
- **Semesterarbeit:** Implicit Plastic Identifier
- **Masterarbeit:** Generative Design in Landscape Architecture
- **Innosuisse:** Schätzung der Entformungskräfte im Spritzguss
- **Innosuisse:** Machine Learning im Spritzguss
- **Innosuisse:** Nummernschild & Gesichtszensursystem
- Optical Quality Measurement and Prediction System
- Codelesesystem Matriq

Inhalt

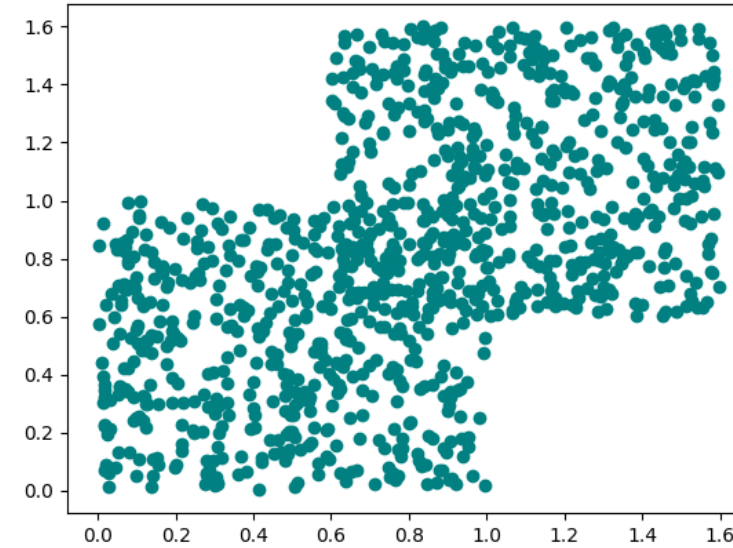
- Einführung
- Fully Connected Feedforward Neural Network
 - Inference & Backpropagation
- Verarbeitung von Bilddaten
- Verarbeitung von Sequenziellen Daten

Teilbereiche

Supervised Learning

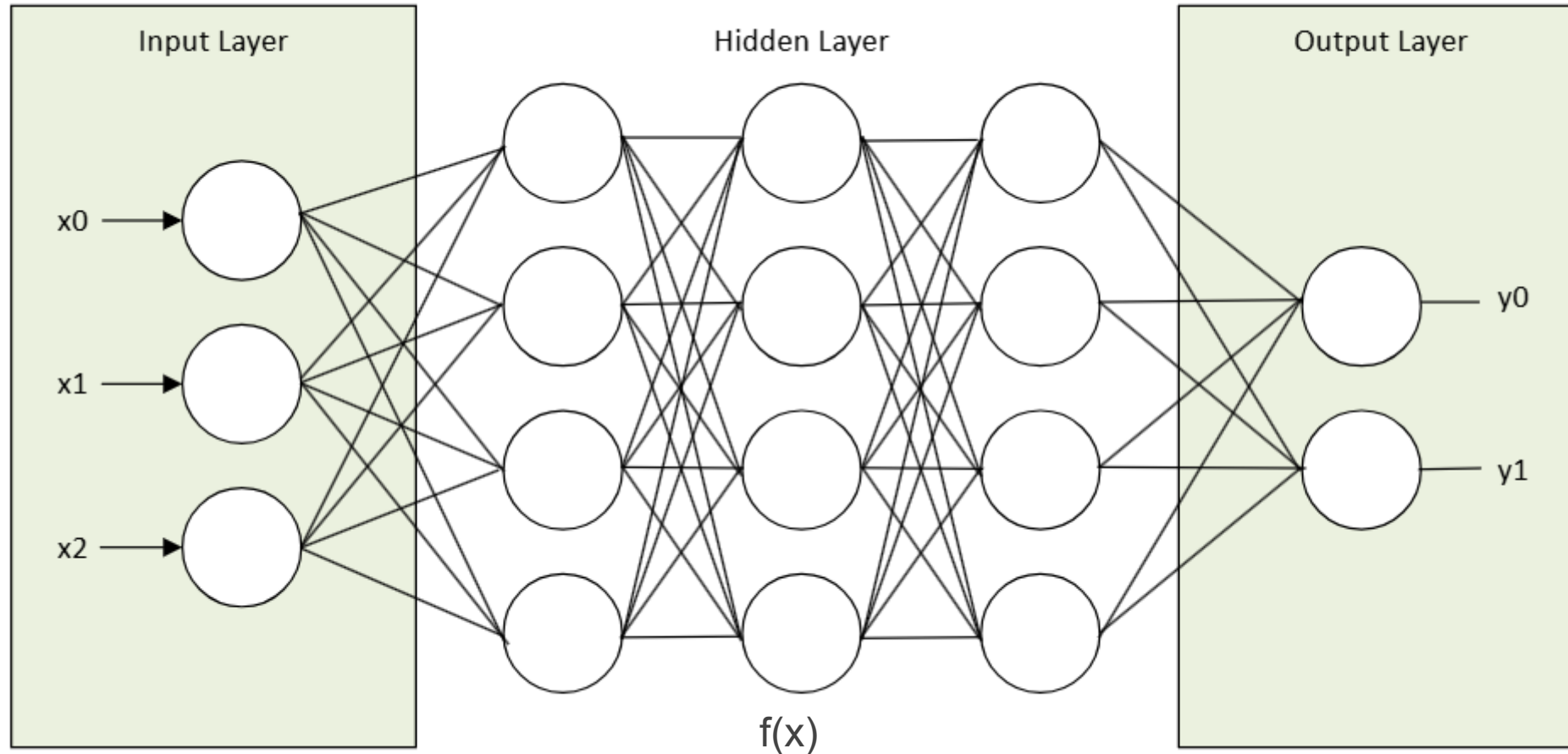


Unsupervised Learning

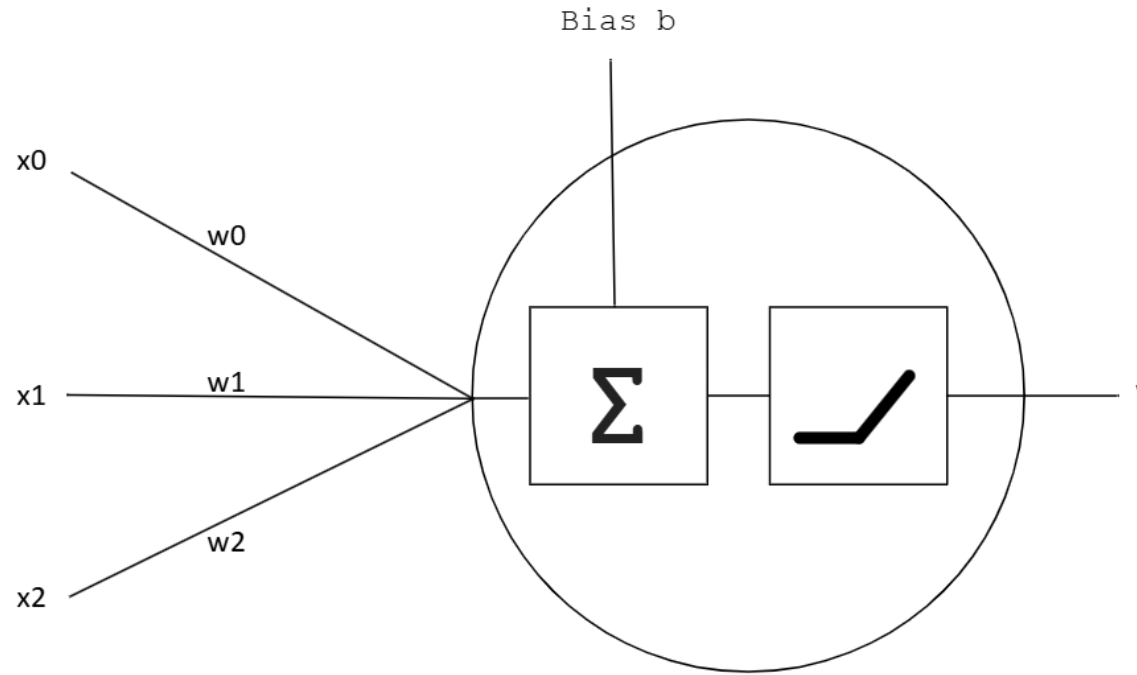


Reinforcement Learning

Fully Connected Feedforward Neural Networks



Neuron



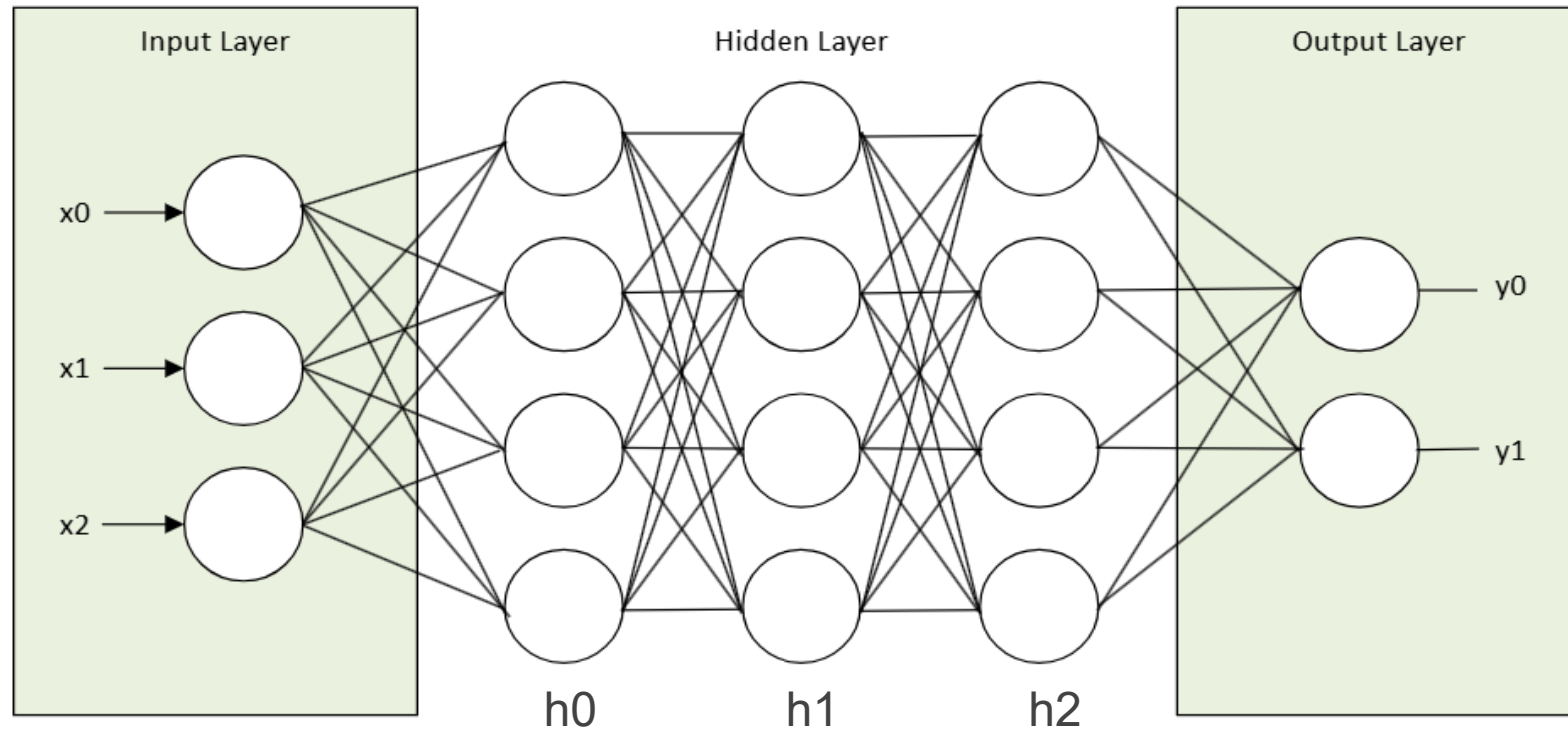
$$y = g \left(\left(\sum_{n=0}^2 x_n w_n \right) + b \right)$$

$$x = \begin{bmatrix} 1 \\ x_0 \\ x_1 \\ x_2 \end{bmatrix} \quad w = \begin{bmatrix} b \\ w_0 \\ w_1 \\ w_2 \end{bmatrix}$$

$$y = g(x^T w)$$

$$g(x) = x^+ = \max(0, x) = \frac{x + |x|}{2} = \begin{cases} x & \text{if } x > 0, \\ 0 & x \leq 0 \end{cases}$$

Inference



$$h_0 = g(W_0 x)$$

$$h_1 = g(W_1 h_0)$$

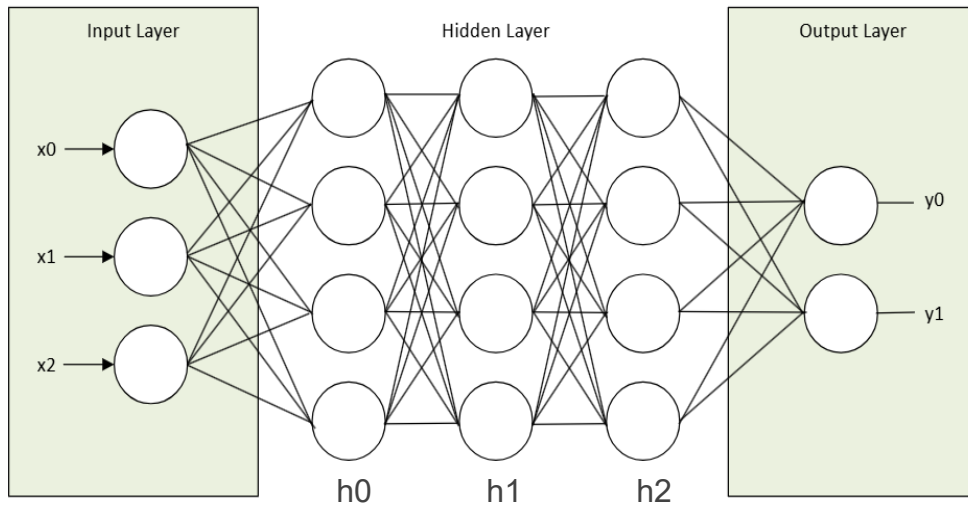
$$h_2 = g(W_2 h_1)$$

$$y = W_3 h_2$$

$$W_1, W_2 \in R^{4 \times 4}$$

$$W_0 \in R^{4 \times 3}, W_3 \in R^{2 \times 4}$$

Training / Backpropagation



Loss function

$$\mathcal{L} = \frac{1}{2} \sum_{i=1}^2 (y_i - \hat{y}_i)^2$$

We want

$$\begin{array}{cc} \frac{\delta \mathcal{L}}{\delta W_3} & \frac{\delta \mathcal{L}}{\delta W_1} \\ \frac{\delta \mathcal{L}}{\delta W_2} & \frac{\delta \mathcal{L}}{\delta W_0} \end{array}$$

$$W_i^{Neu} = W_i - \eta \frac{\delta \mathcal{L}}{\delta W_i}$$

Training / Backpropagation

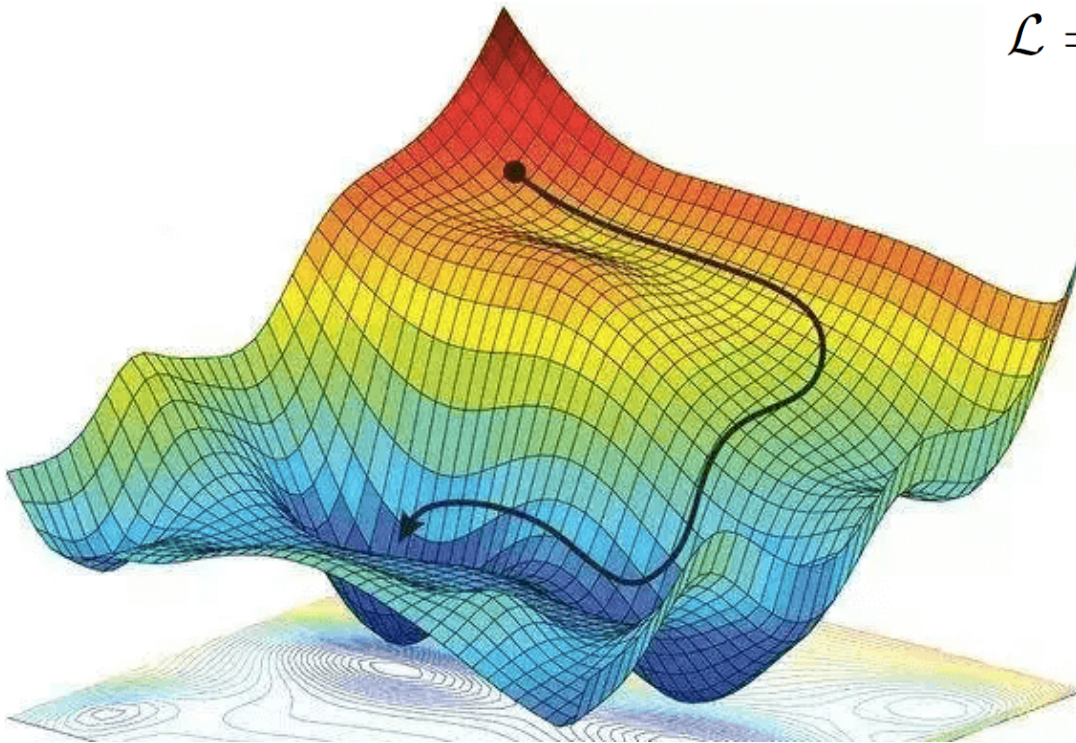
Loss function

$$\mathcal{L} = \frac{1}{2} \sum_{i=1}^2 (y_i - \hat{y}_i)^2$$

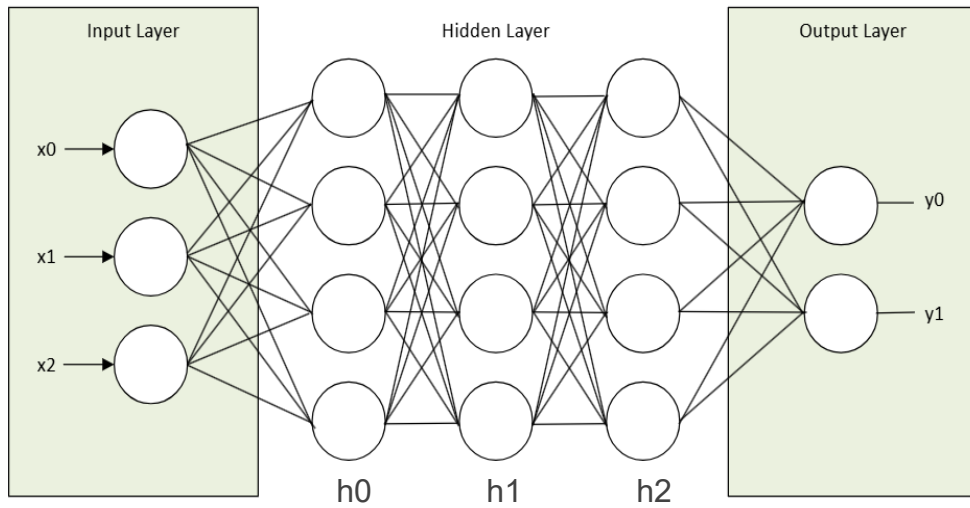
We want

$\frac{\delta \mathcal{L}}{\delta W_3}$	$\frac{\delta \mathcal{L}}{\delta W_1}$
$\frac{\delta \mathcal{L}}{\delta W_2}$	$\frac{\delta \mathcal{L}}{\delta W_0}$

$$W_i^{Neu} = W_i - \eta \frac{\delta \mathcal{L}}{\delta W_i}$$



Training / Backpropagation



Loss function

$$\mathcal{L} = \frac{1}{2} \sum_{i=1}^2 (y_i - \hat{y}_i)^2$$

$$\frac{\delta \mathcal{L}}{\delta \hat{y}} = y - \hat{y} :$$

Forward pass

$$h_0 = g(W_0 x)$$

$$h_1 = g(W_1 h_0)$$

$$h_2 = g(W_2 h_1)$$

$$y = W_3 h_2$$

$$\frac{\delta \mathcal{L}}{\delta W_3} = \frac{\delta \mathcal{L}}{\delta \hat{y}} \frac{\delta \hat{y}}{\delta W_3}$$

$$\frac{\delta \mathcal{L}}{\delta W_2} = \frac{\delta \mathcal{L}}{\delta \hat{y}} \frac{\delta \hat{y}}{\delta W_3} \frac{\delta W_3}{\delta W_2}$$

$$\frac{\delta \mathcal{L}}{\delta W_1} = \frac{\delta \mathcal{L}}{\delta \hat{y}} \frac{\delta \hat{y}}{\delta W_3} \frac{\delta W_3}{\delta W_2} \frac{\delta W_2}{\delta W_1}$$

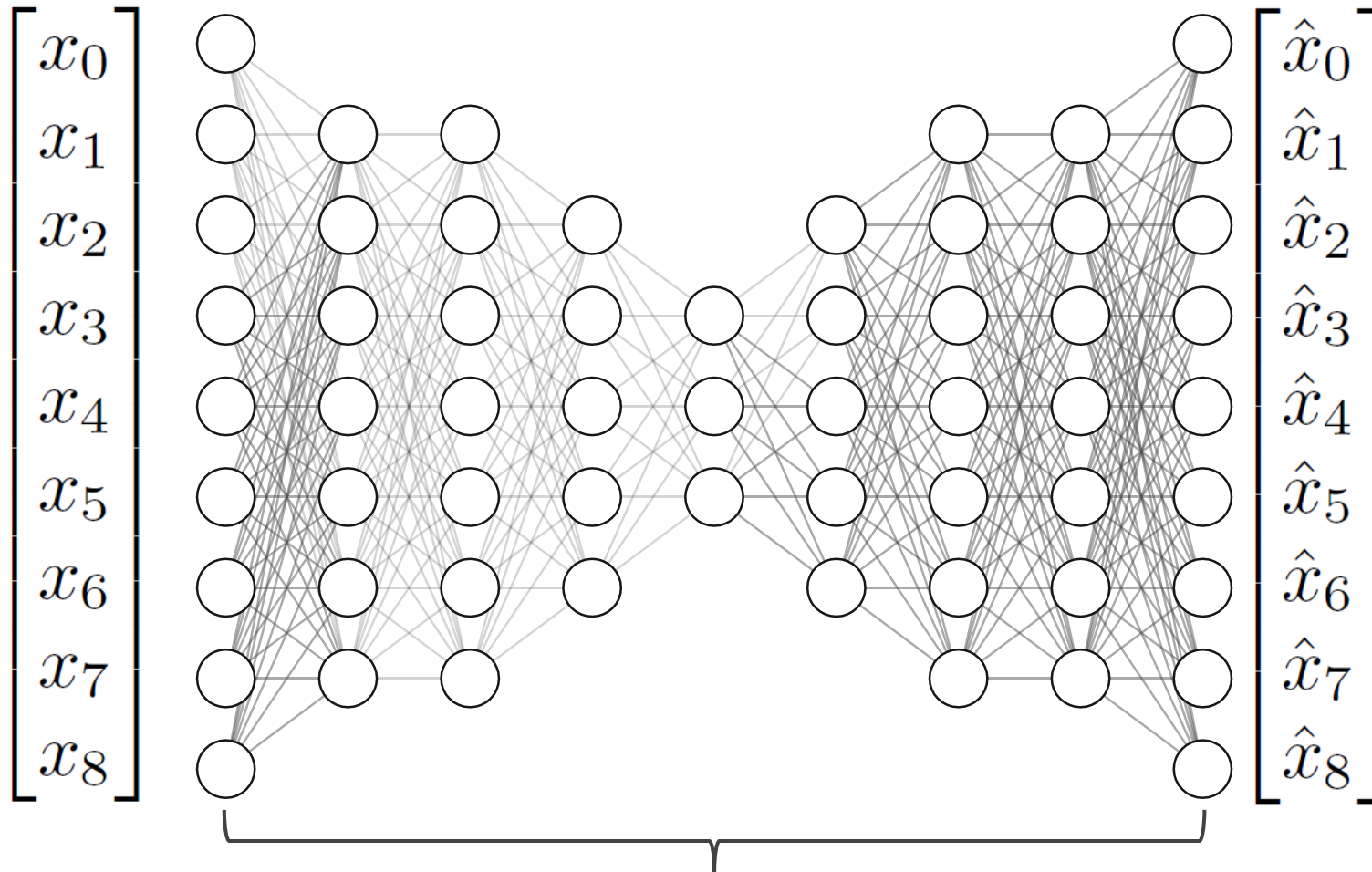
$$\frac{\delta \mathcal{L}}{\delta W_0} = \frac{\delta \mathcal{L}}{\delta \hat{y}} \frac{\delta \hat{y}}{\delta W_3} \frac{\delta W_3}{\delta W_2} \frac{\delta W_2}{\delta W_1} \frac{\delta W_1}{\delta W_0}$$

$$W_i^{Neu} = W_i - \eta \frac{\delta \mathcal{L}}{\delta W_i}$$

Praktik

- Training / Testdaten, Cross Validation
- Activation Function
- Normalisierung
- Initialization
- Dropout
- ADAM, AdaGrad, RMSProp
- Training in Batches, Stochastic Gradient Descent
- Batch Normalization
- Under- / Overfitting

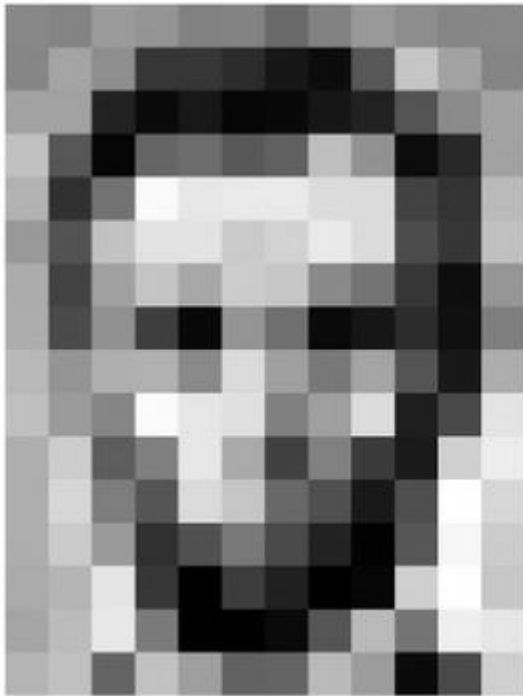
Unsupervised Learning: Autoencoder



$$\mathcal{L} = \sum_{i=0}^8 (x_i - \hat{x}_i)^2$$

Bilddaten

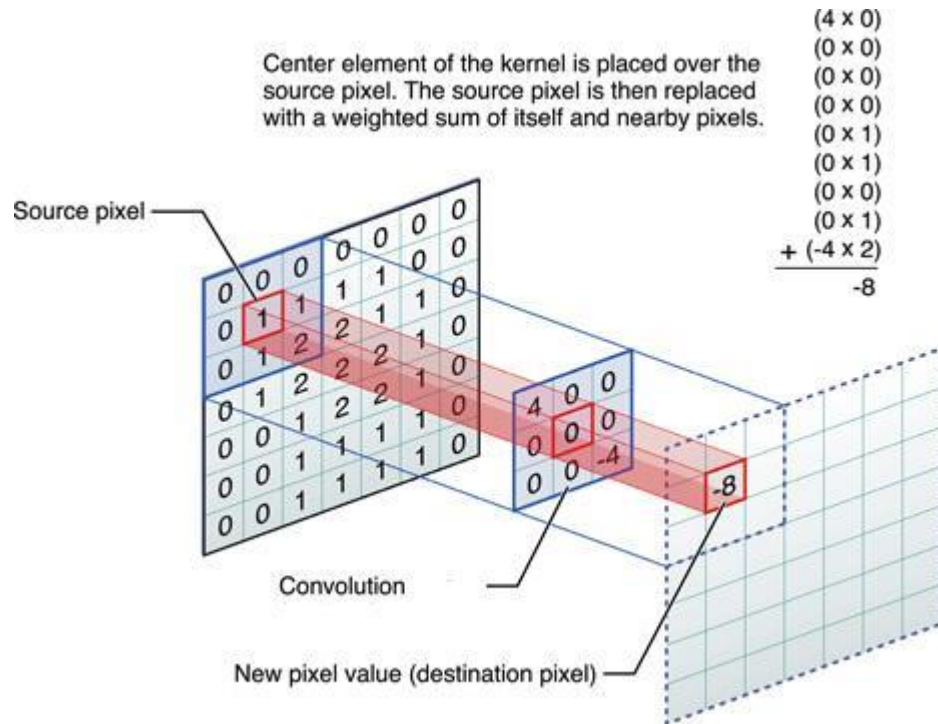
Bilddaten



157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	108	5	124	131	111	120	204	166	15	56	180
194	68	197	251	257	299	239	228	227	87	71	201
172	106	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	87	165	84	10	168	134	11	31	62	22	148
199	168	191	193	168	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	100	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

- Jedes Pixel hat einen Zahlenwert → Könnte also in ein NN gegeben werden.
- Probleme:
 - Zu viele Parameter
 - Positionsinformationen gehen verloren

Bildverarbeitung - Filter



- Filter wird über das Bild gezogen
- Entspricht mathematisch einer 2D Faltung → Multiplikation im Spektralbereich

Bildverarbeitung - Filter

Averaging



$$k = \frac{1}{25} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

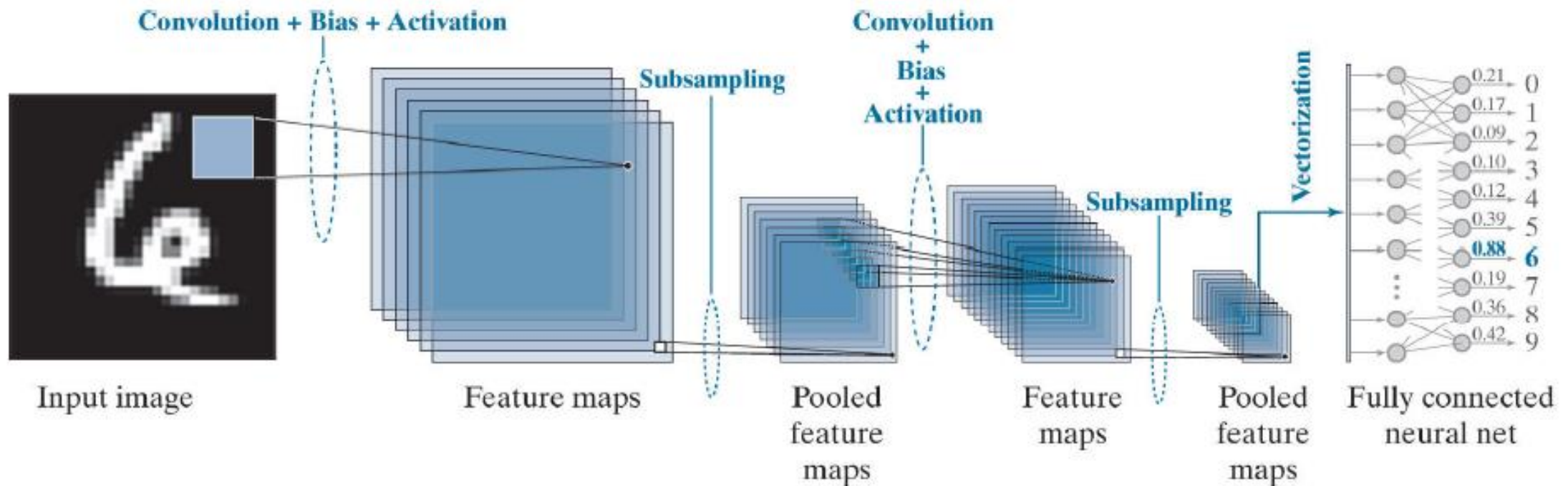
Bildverarbeitung - Filter

Approximation to laplacian filter



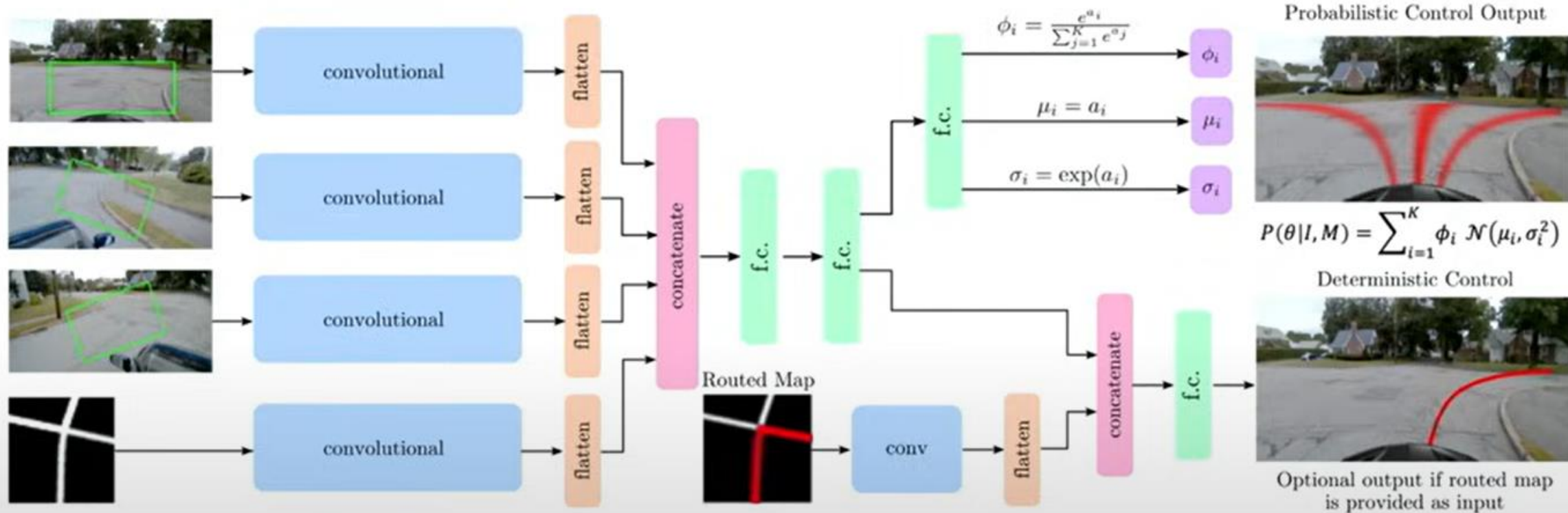
$$k = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Convolutional Neural Network - CNN

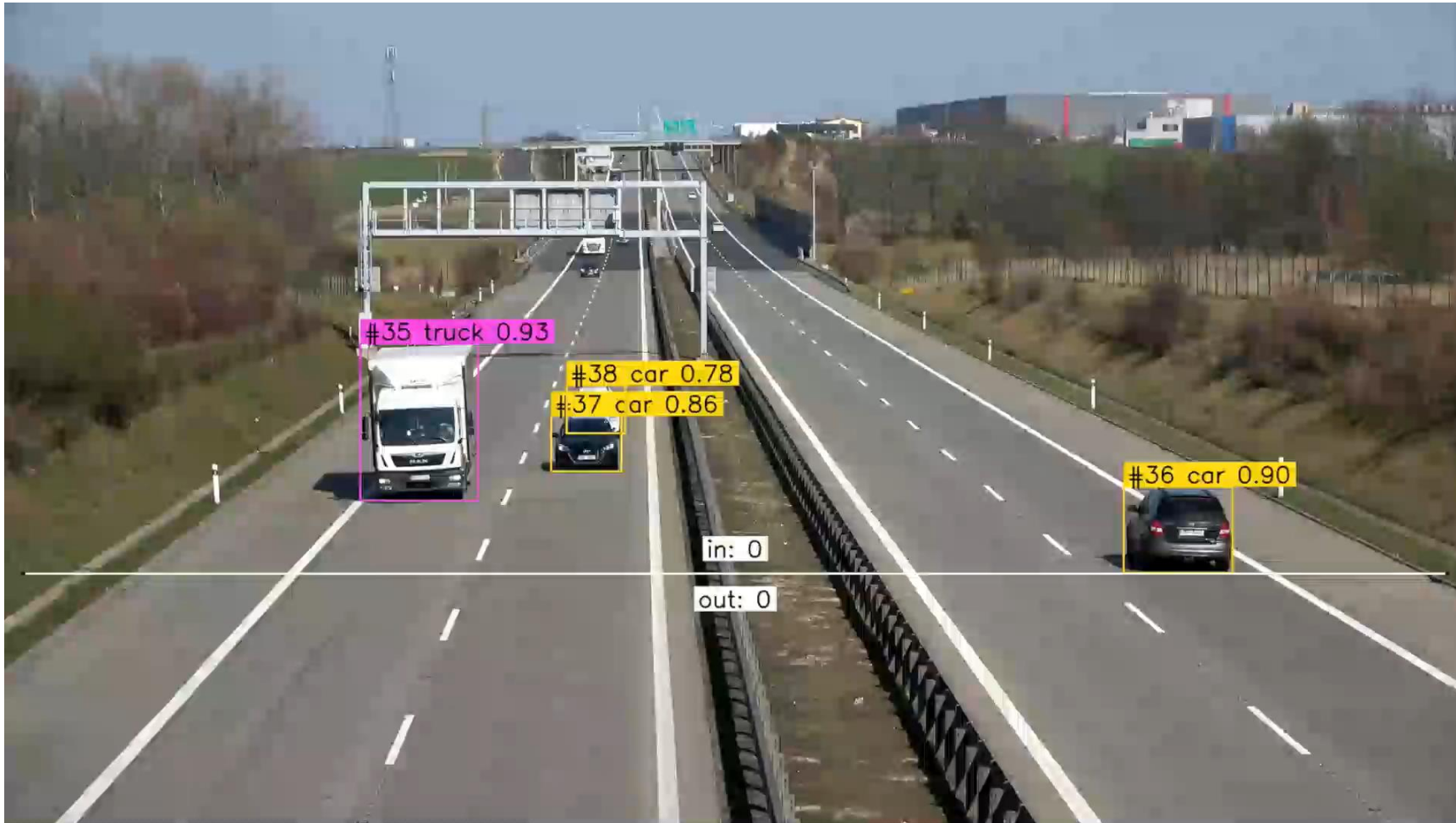


End-to-End Framework for Autonomous Navigation

Entire model is trained end-to-end **without any human labelling or annotations**



$$L = -\log(P(\theta|I, M))$$



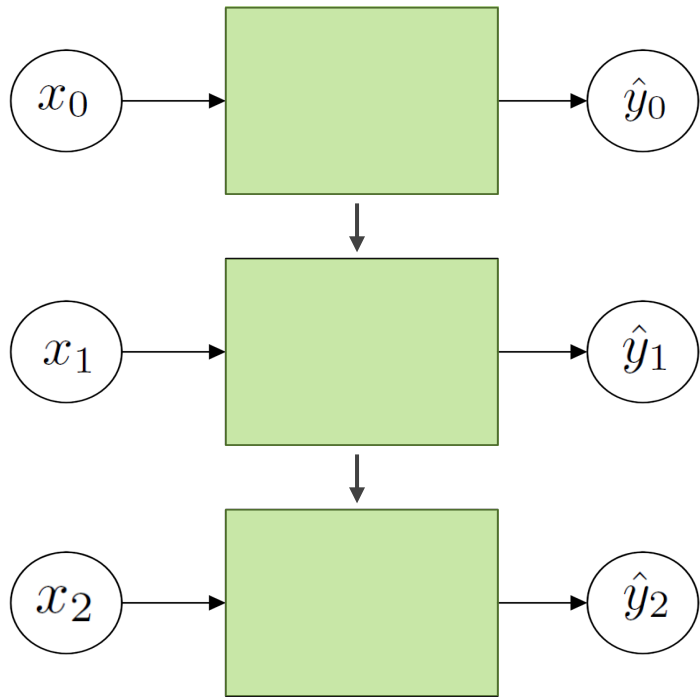
Sequentielle Daten

RNN, LSTM, GRU, Attention

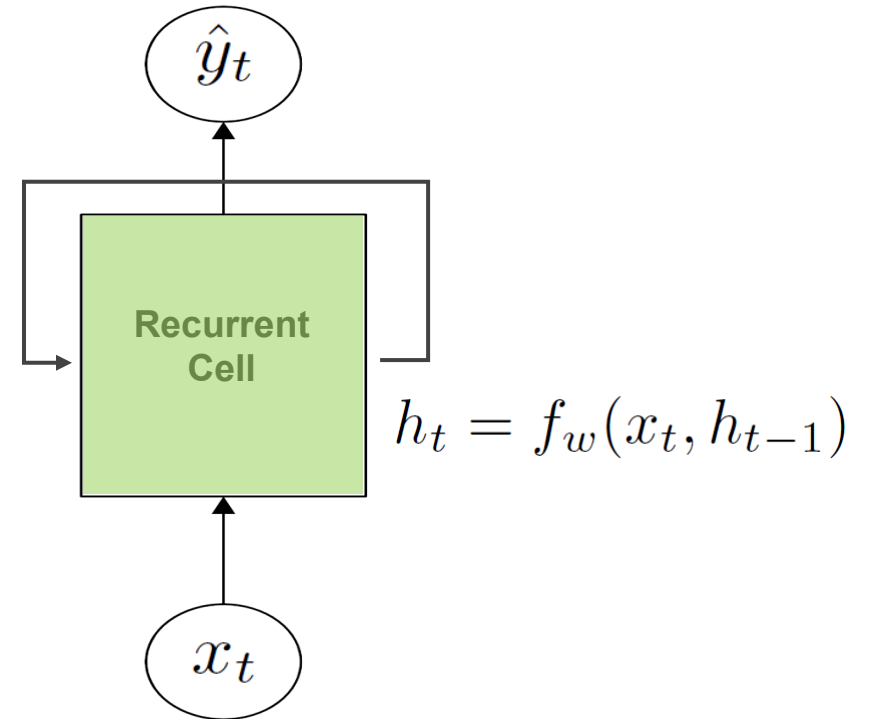
Sequenzielle Daten – Text, Audio...

- NN und CNN haben keinen Zeitfaktor.
- Die Netzwerke stellen lediglich eine Funktion $f(x)$ dar.
- Für sequenzielle Daten spielt der zeitliche Verlauf eine Rolle.
- Netzwerke brauchen eine Möglichkeit Daten sequenziell zu verarbeiten, bzw. zu speichern.
- Textdaten werden zuerst mit einem Tokenizer in Zahlen umgerechnet.

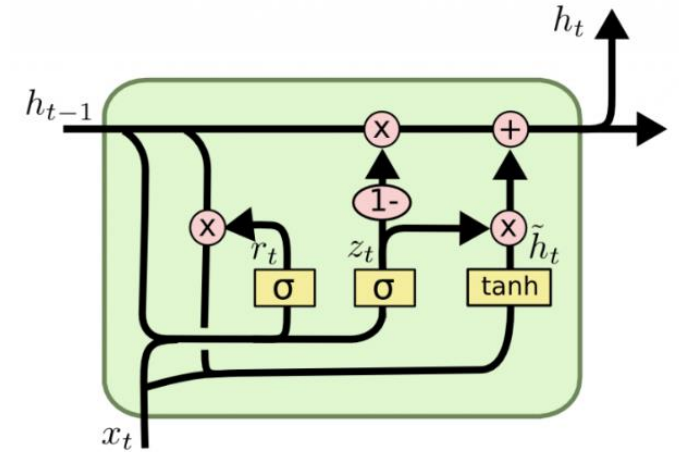
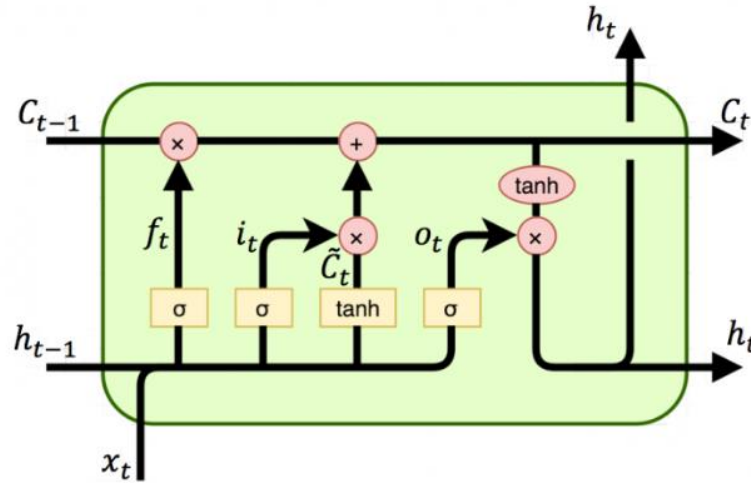
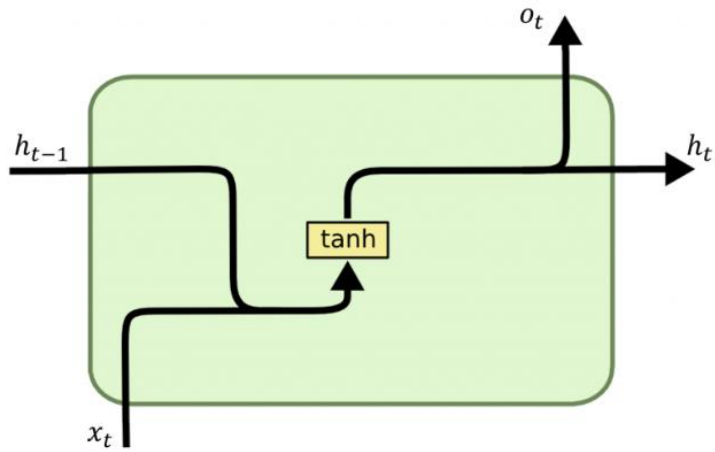
Recurrent Neural Networks



$$\hat{y}_t = f(x_t, h_{t-1})$$



RNN, LSTM, GRU



- Probleme:
 - Vanishing Gradients
 - Sequenzielle Berechnung, nicht parallelisierbar.

Attention is all you need

- Input Embeddings
- Positional Embeddings
- Attention

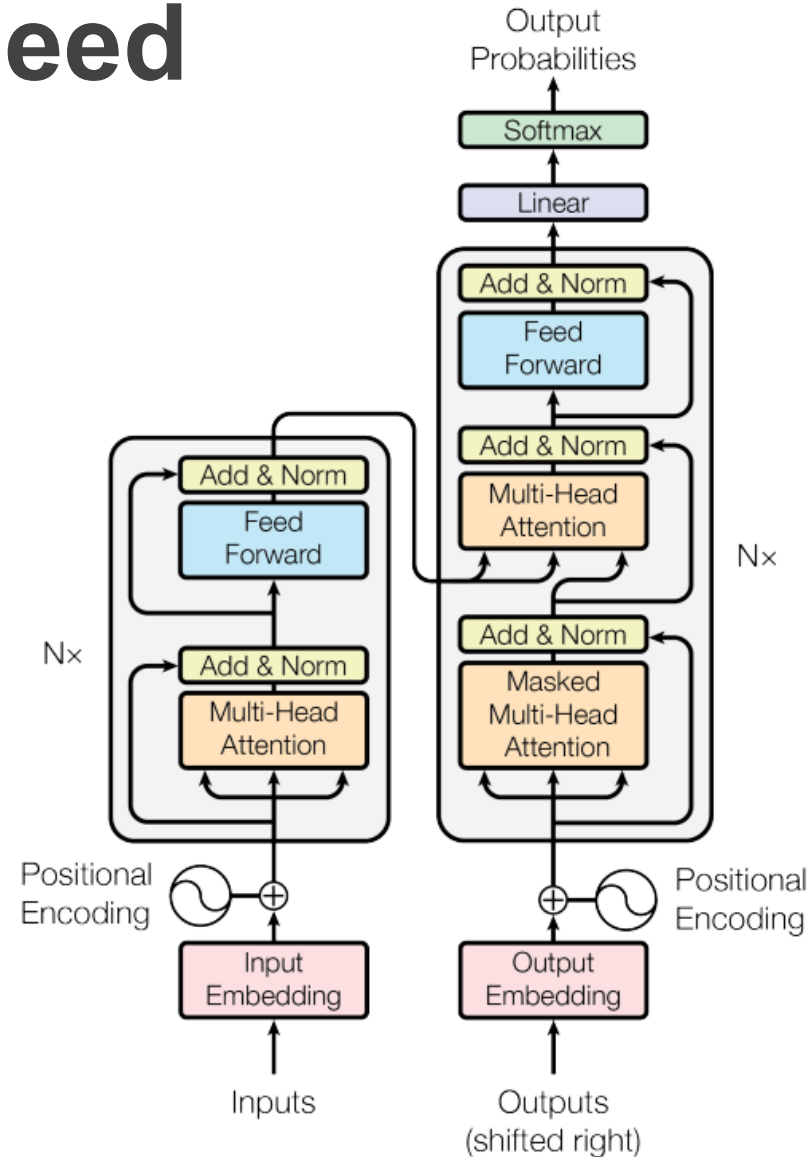
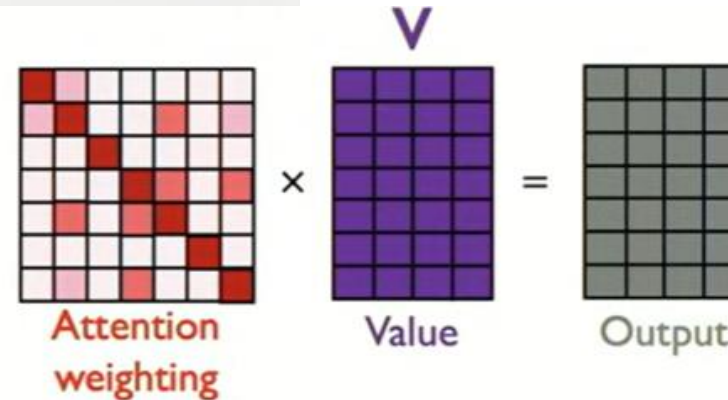
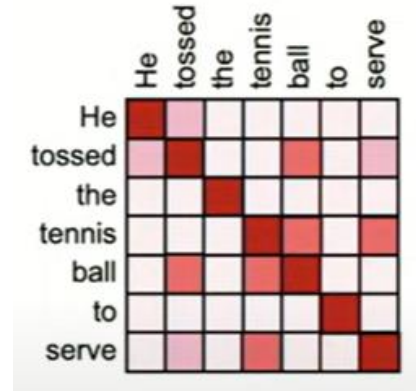
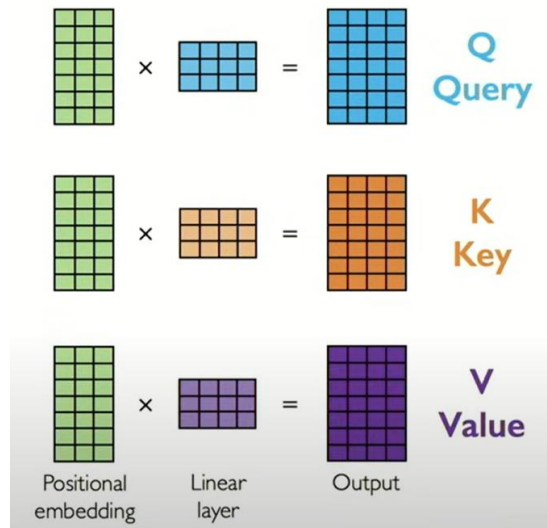


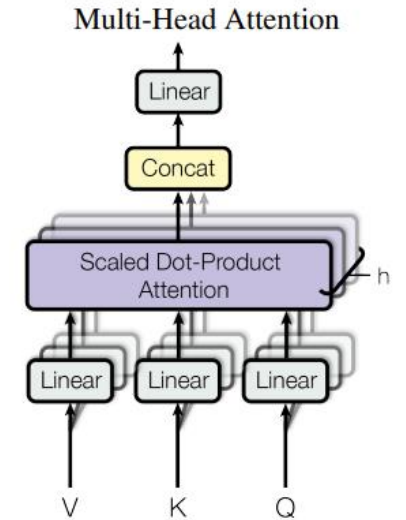
Figure 1: The Transformer - model architecture.

Attention is all you need

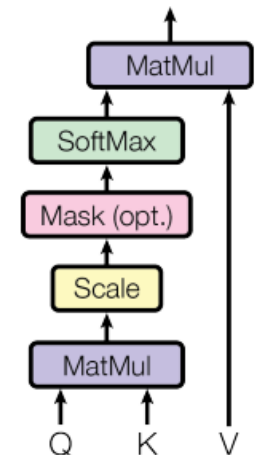
- Query und Key werden verglichen → Entsteht eine Attention Matrix
- Output wird mit der Value Matrix gewichtet



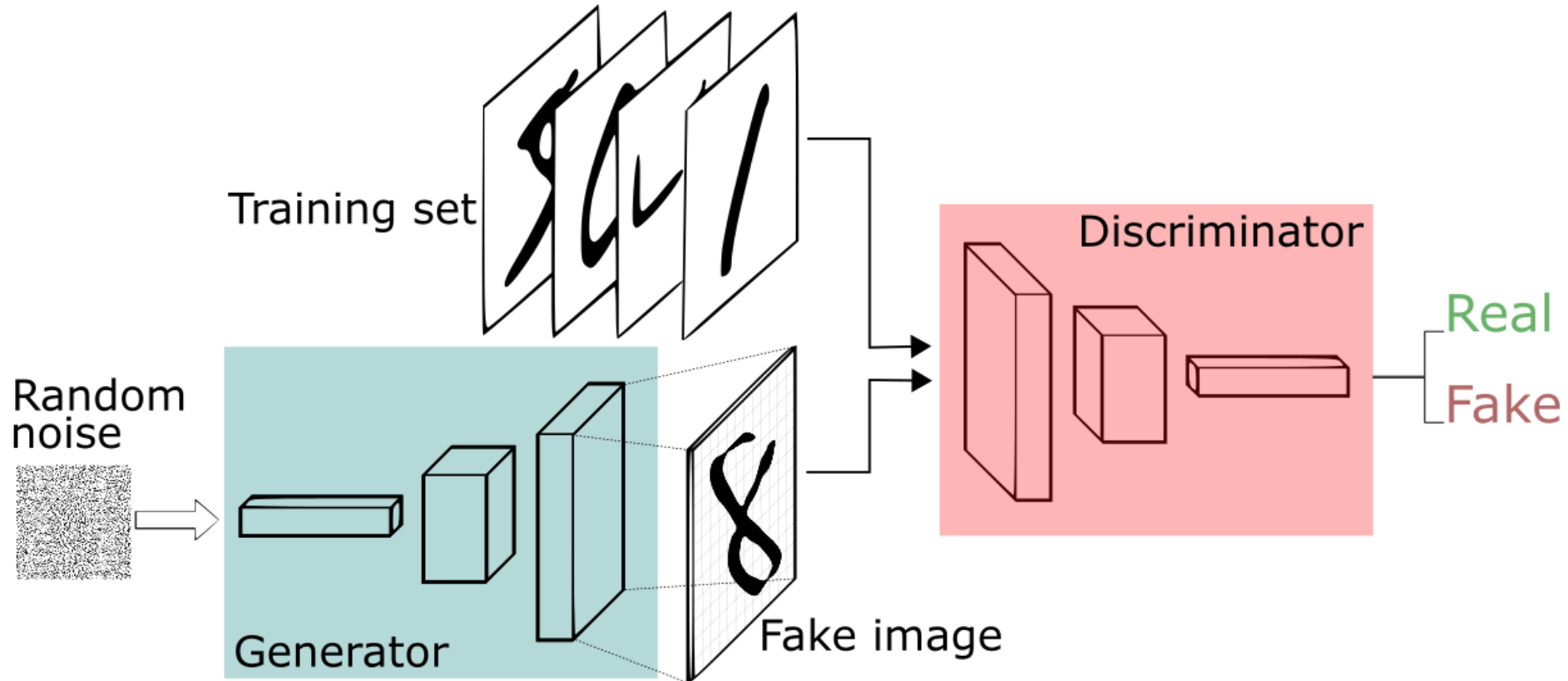
$$\text{softmax} \left(\frac{Q \cdot K^T}{\text{scaling}} \right) \cdot V = A(Q, K, V)$$



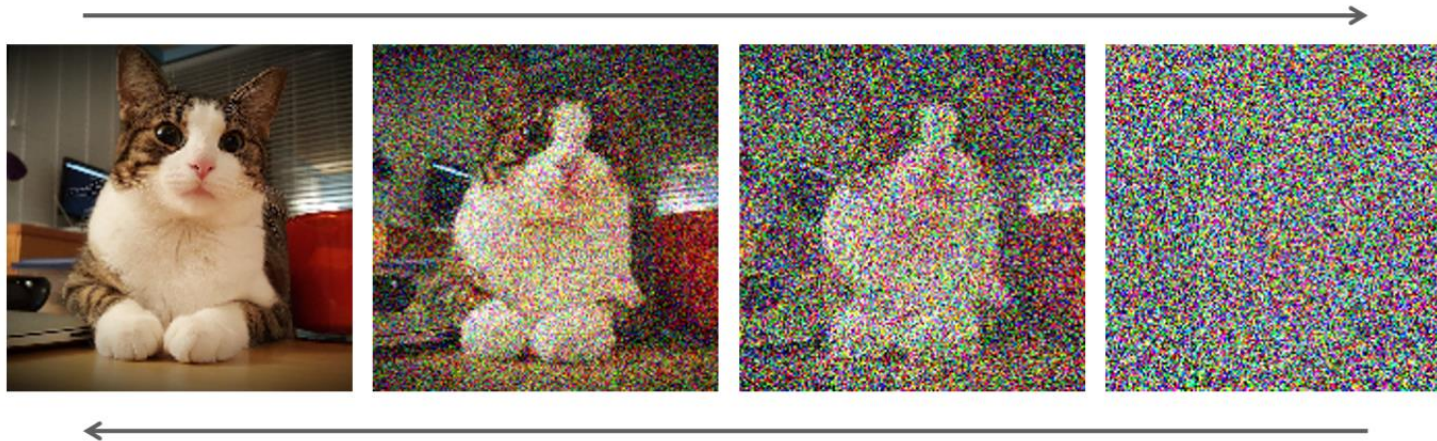
Scaled Dot-Product Attention



Zusatz: Generative Adversarial Networks



Zusatz: Diffusion Models



Fragen?

Quellen

- MIT Vorlesung: <https://www.youtube.com/watch?v=dqoEU9Ac3ek>
- Transformer Paper: <https://arxiv.org/pdf/1706.03762>
- RNN,LSTM, GRU Architectures: <http://dprogrammer.org/rnn-lstm-gru>