2. Einführung in Deep Learning



- Feed Forward Netzwerke
- Convolutional Neural Networks
- Residual Neural Networks
- Graph Netzwerke
- Encoder-Decoder Architekturen
- Transfomers & Auto-Encoders
- Rekkurente Neuronale Netze
- LSTM
- Generative Adversarial Network (GAN)
- Diffusion Generative Models
- Training und Optimierung

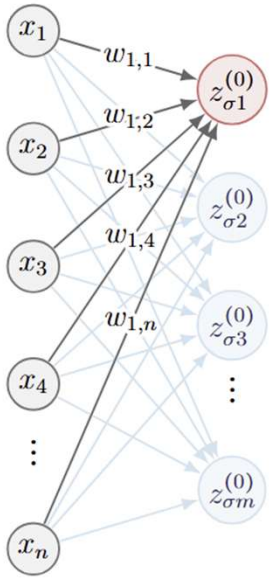
2.1 Fully-connected Neural Network



Ein Feedforward-Netz als parametrisierte Funktion

$$z^l(x) = W^l x + b^l$$

W^l: *Gewichtsmatrix* b^l: *Biasvektor*



$$\begin{pmatrix} z^{(0)}_{\sigma 1} \\ z^{(0)}_{\sigma 2} \\ \vdots \\ z^{(0)}_{\sigma m} \end{pmatrix} = \sigma \left[\begin{pmatrix} w_{1,1} & w_{1,2} & \dots & w_{1,n} \\ w_{2,1} & w_{2,2} & \dots & w_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{m,1} & w_{m,2} & \dots & w_{m,n} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} + \begin{pmatrix} b^{(0)}_1 \\ b^{(0)}_2 \\ \vdots \\ b^{(0)}_m \end{pmatrix} \right]$$

$$\mathbf{z}_\sigma{}^{(0)} = \sigma\left(\mathbf{W}^{(0)}\mathbf{x} + \mathbf{b}^{(0)}\right)$$

Verborgene Schichten mit Aktivierung

$$h^l = \sigma(W^l h^{l-1} + b^l)$$

Die L-1 verborgene Schicht eines feedforward Netzes:

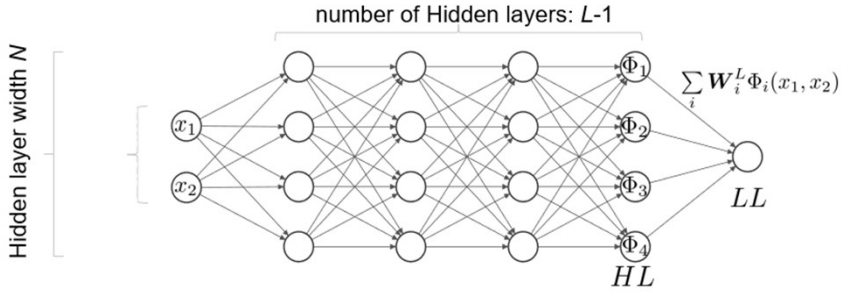
$$\mathcal{N}_h(x) = \sigma \circ z^{L-1} \circ \cdots \circ \sigma \circ z^1(x)$$

Für eine Regression, ein DNN hat typischerweise die Form:

$$\mathcal{N}(x;\theta) = z^L \circ \mathcal{N}_h(x)$$

Netzwerk Parameter:

$$\theta = \{\mathbf{W}^l, \mathbf{b}^l\}_{1 \le l \le L}$$



2.1 Motivation and Notwendigkeit für andere Neuronale Netze

Ursache: Diverse Datenformate und Lernaufgaben
Rettung: Verschiedene neuronale Netzarchitekturen



Datentyp: tabellarische Daten

Breast Cancer Wisconsin (Prognostic) Data Set

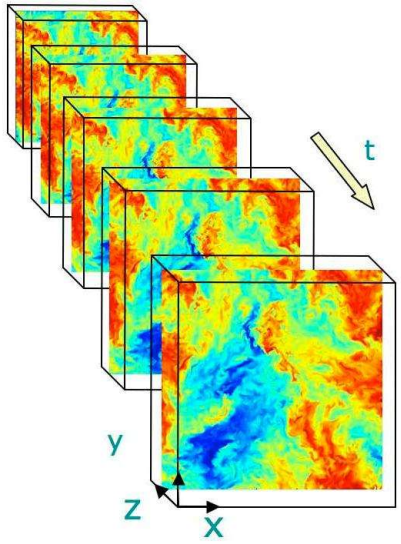
1) ID number
2) Outcome (R = recur, N = nonrecur)
3) Time (recurrence time if field 2 = R, disease-free time if field 2 = N)
4-33) Ten real-valued features are computed for each cell nucleus:

a) radius (mean of distances from center to points on the perimeter)
b) texture (standard deviation of gray-scale values)
c) perimeter
d) area
e) smoothness (local variation in radius lengths)
f) compactness (perimeter^2 / area - 1.0)
g) concavity (severity of concave portions of the contour)
h) concave points (number of concave portions of the contour)
i) symmetry
j) fractal dimension ("coastline approximation" - 1)"

UCI Machine Learning Repository

Datentyp: Bilddaten

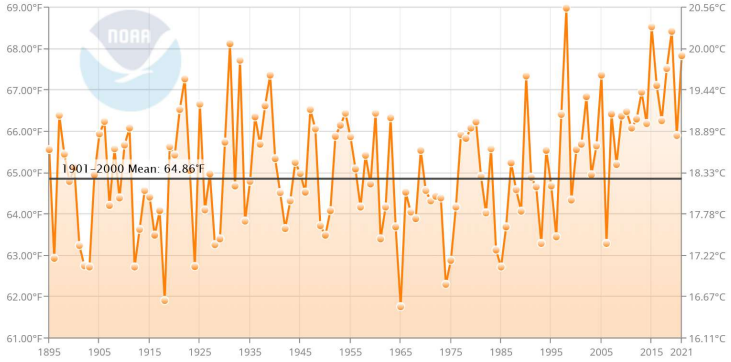
Velocity and pressure for fluid flow



Johns Hopkins Turbulence Databases

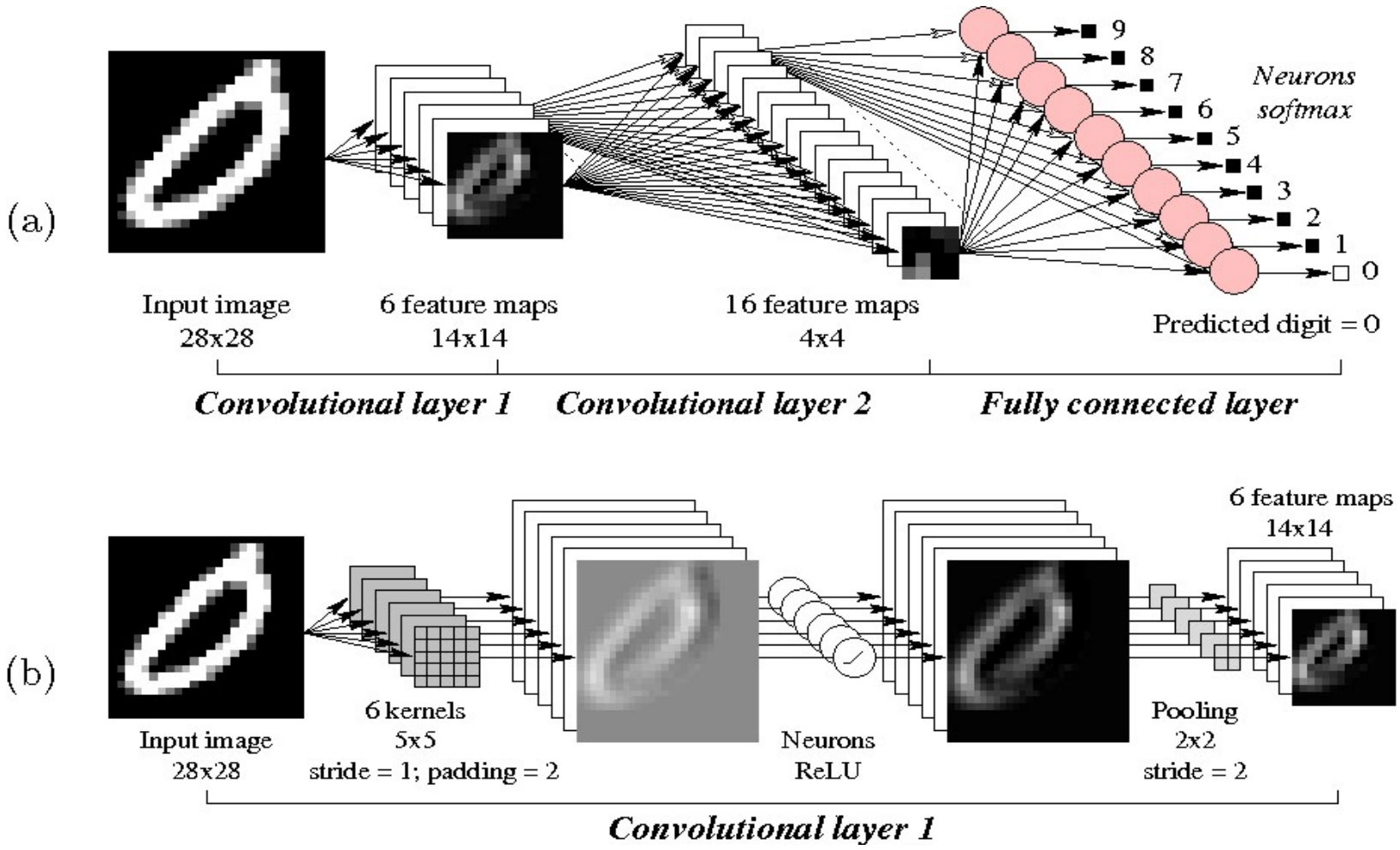
Datentyp: Zeitreihendaten

Contiguous U.S. Average Temperature

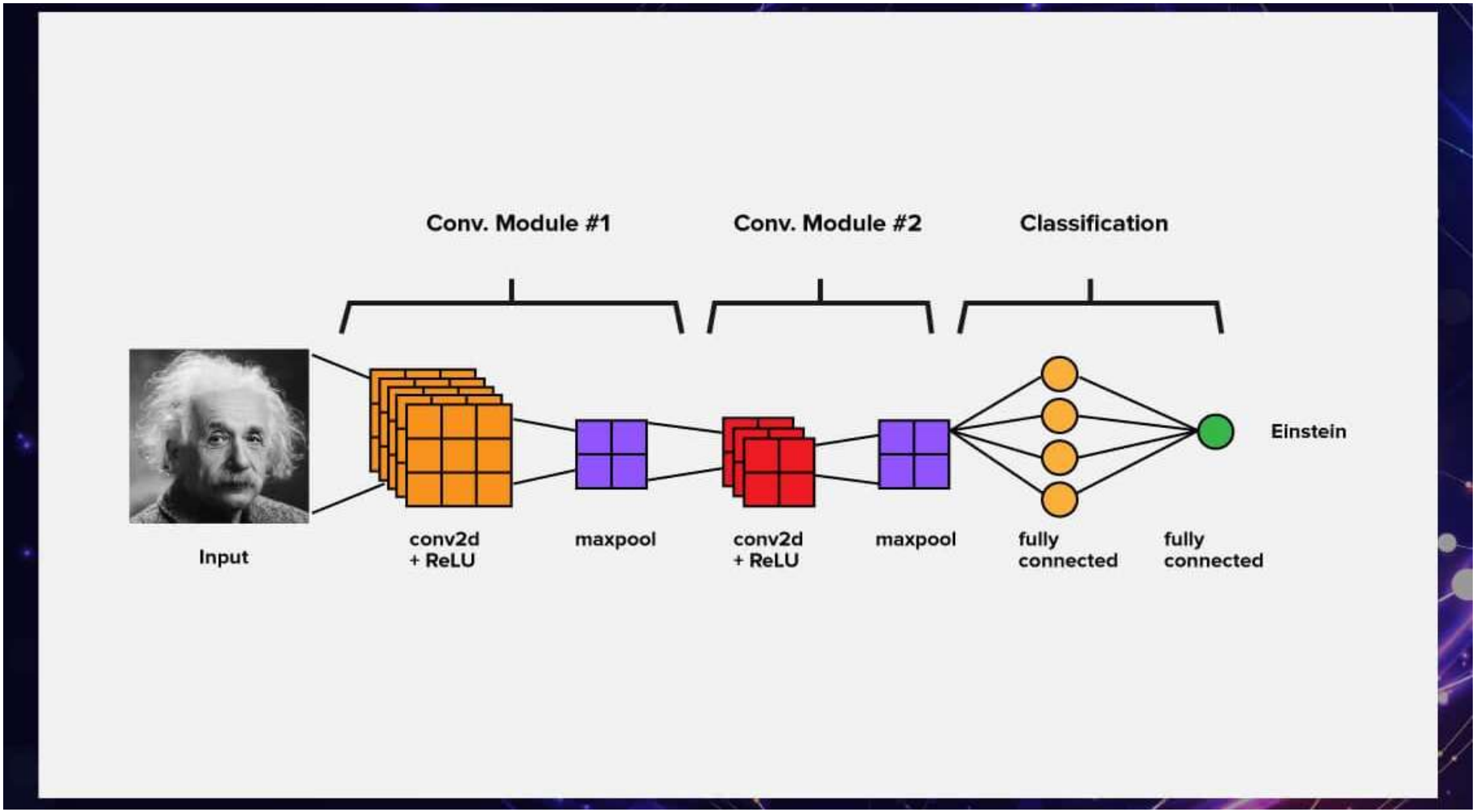


NOAA NATIONAL CENTERS FOR ENVIRONMENTAL INFORMATION
NATIONAL OCEANIC AND ATMOSPHERIC ADMINISTRATION

2.2 Convolutional Neural Networks (CNNs)



2.2 Convolutional Neural Networks (CNNs)

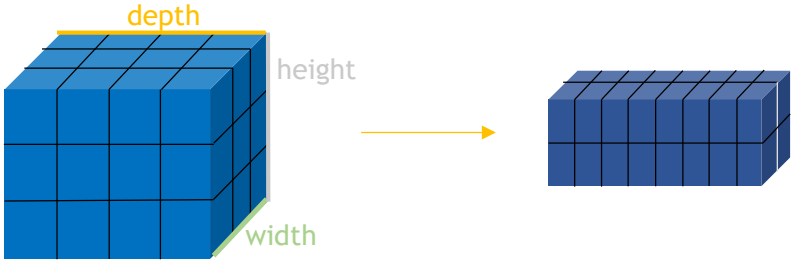


Faltung



- Input: Ein 3D Tensor (breite, höhe, tiefe); falls der Input ein Bild ist, Tiefe ist 3, rot, grün und blau Kanäle
- Output: Ein 3D Tensor mit höhe, tiefe und breite
- Eine Faltungsschicht: Faltung zwischen zwei Funktionen f und g ist definiert als

$$(f * g)(\mathbf{x}) = \int f(\mathbf{z}) g(\mathbf{x} - \mathbf{z}) d\mathbf{z}$$



- Für 1D diskreten Fall, Integral wird überführt in eine Summe

$$(f * g)(i) = \sum_a f(a) g(i - a)$$

- Für 2D

$$(f * g)(i, j) = \sum_a \sum_b f(a, b) g(i - a, j - b)$$

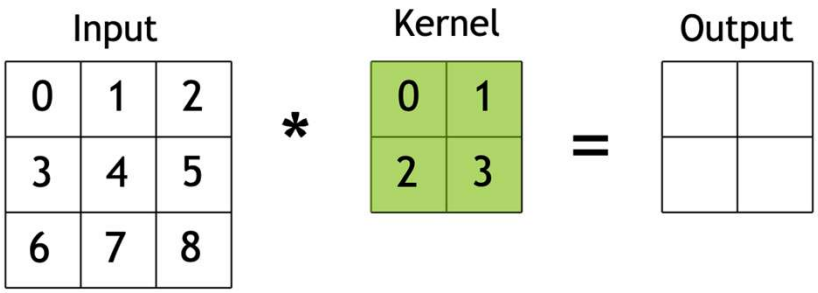
- Neu formulieren als Kreuzkorrelation

$$[\mathbf{H}]_{i,j} = u + \sum_{a=-\Delta}^{\Delta} \sum_{b=-\Delta}^{\Delta} [\mathbf{V}]_{a,b} [\mathbf{X}]_{i+a,j+b}$$

- Für multiple Kanäle

$$[\mathsf{H}]_{i,j,d} = \sum_{a=-\Delta}^{\Delta} \sum_{b=-\Delta}^{\Delta} \sum_c [\mathsf{V}]_{a,b,c,d} [\mathsf{X}]_{i+a,j+b,c}$$

Faltungsbeispiel



Kernel size: (2,2), Input size: (3,3)
⇒ No. of parameters: 2*2, Output size: (2,2)

Kernel size: (K, K), Input size: (W, H)
⇒ No. of parameters: K*K, Output size: (W-K+1,H-K+1)

FNN vs. CNN



FNN

$$T^l(x) = \boldsymbol{W}^l x + \boldsymbol{b}^l$$

$$[\mathbf{H}]_{i,j} = [\mathbf{U}]_{i,j} + \sum_k \sum_l [\mathbf{W}]_{i,j,k,l} [\mathbf{X}]_{k,l}$$
$$= [\mathbf{U}]_{i,j} + \sum_a \sum_b [\mathbf{V}]_{i,j,a,b} [\mathbf{X}]_{i+a,j+b}$$

CNN

$$[\mathbf{H}]_{i,j} = u + \sum_{a=-\Delta}^{\Delta} \sum_{b=-\Delta}^{\Delta} [\mathbf{V}]_{a,b} [\mathbf{X}]_{i+a,j+b}$$

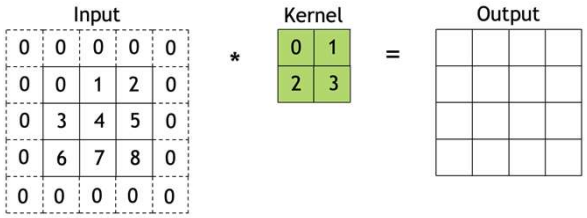
Eigenschaften eines CNN

- Effizienz: Deutlich weniger Parameter $(W\ x\ H)^2 \rightarrow K^2$
- Lokalität:
  1. Benachbarte Pixel stehen typischerweise in Beziehung zueinander
  2. Wir sollten nicht allzu weit entfernt von Punkt (i, j) suchen müssen, um $H(i, j)$ zu berechnen
- Translationsäquivarianz: Eine Verschiebung der Eingabe X sollte lediglich zu einer Verschiebung der verborgenen Darstellung H führen.

Padding



- Bei der Anwendung einer Faltungsschicht verlieren wir Pixel
- Hinzufügen von extra Pixeln um den Rand von unserem Input
- Padding type: konstant (0), Reflexion, etc.



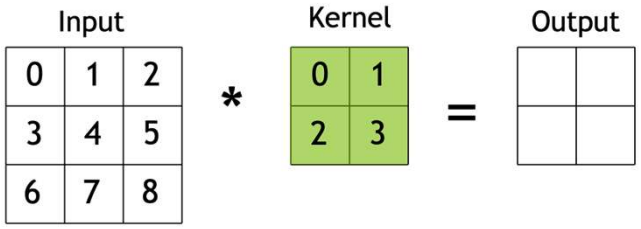
Kernel size: (2,2), Input size: (3,3), padding size: 1
$\Rightarrow$ Output size: (4, 4)

Kernel size: (K, K), Input size: (W, H), padding size: P
$\Rightarrow$ Output size: (W-K+2P+1,H-K+2P+1)

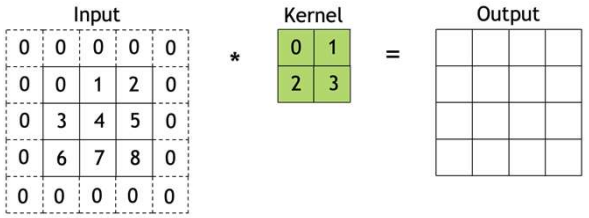
Padding



Convolution



Convolution + Padding

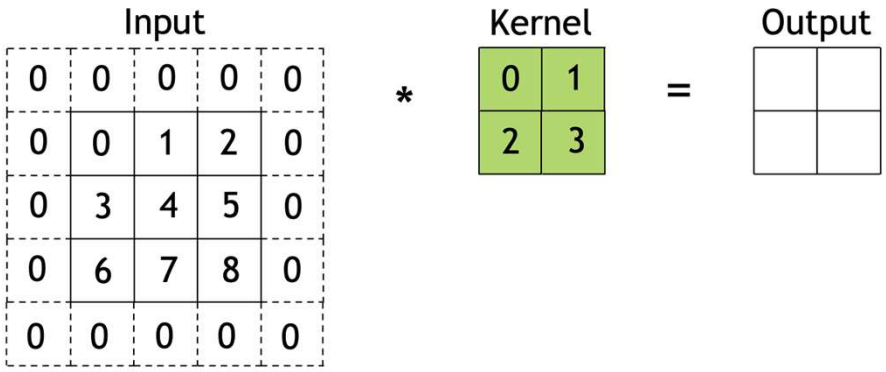


Stride



- Für erhöhte Recheneffizienz oder downsampling
- Das Fenster jeweils um mehr als ein Element verschieben

Kernel size: (2,2), Input size: (3,3), padding size: 1, stride: 2
$\Rightarrow$ Output size: (2, 2)
Kernel size: (K, K), Input size: (W, H), padding size: P, stride: S
$\Rightarrow$ Output size: $(\frac{W-K+2P}{S}+1, \frac{H-K+2P}{S}+1)$

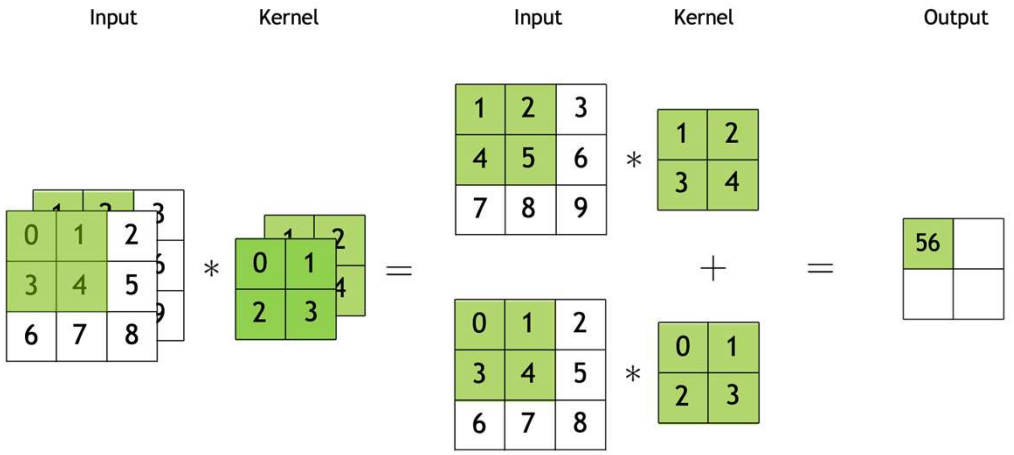


Multiple Input und multiple Output Kanäle



Multiple Input Kanäle

- Farbbilder haben die Standard RGB Kanäle, um den Anteil an rot, grün und blau anzugeben
- Input (3,3,2), Kernel: (2,2)
- 8 Parameter, Output: (2,2,1)
- Input: (W,H,D), Kernel: (K,K)
- K*K*D Parameter, Output: (W-K+1, H-K+1,1)



Multiple Output Kanäle

- Input: (W,H,D), N Kernel der Größe (K,K)
- K*K*D Parameter pro Kernel, Output: (W-K+1, H-K+1, N)

Zusammenfassung: Eine Faltungsschicht



- Eingabegröße: W_1, H_1, D_1
- Vier erforderliche Hyperparameter:

  Anzahl der Filter: N

  Filtergröße: K

  Schrittweite: S

  Breite der Randauffüllung: P

- Ausgabegröße: W_2, H_2, D_2

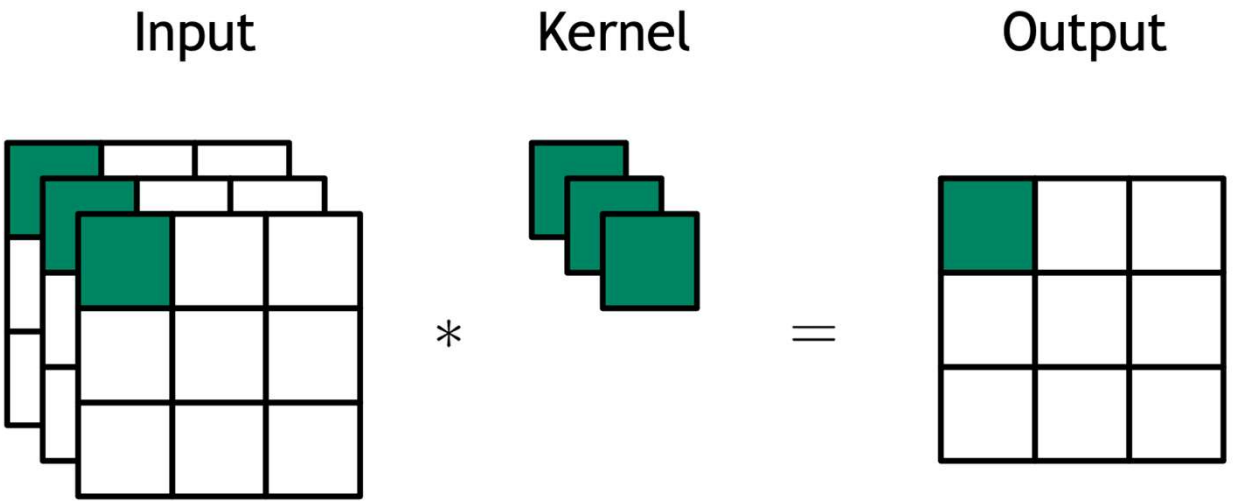
$$W_2 = \frac{W_1 - K + 2P}{S} + 1$$
$$H_2 = \frac{H_1 - K + 2P}{S} + 1$$
$$D_2 = N$$

Übliche Einstellungen:
N = Zweierpotenzen, z. B. 32, 64, 128, 512

- $K = 3, S = 1, P = 1$
- $K = 5, S = 1, P = 1$
- $K = 5, S = 2, P = ?$ $(passend\ wählen)$
- $K = 1, S = 1, P = 0$

- $K * K * D_1$ Parameter pro Filter. Insgesamt $K * K * D_1 * N$ Parameter und N Bias-Terme.
- Der d-te Ausgabekanal entsteht durch Faltung des d-ten Filters mit der Eingabe.

1x1 Faltungsschicht

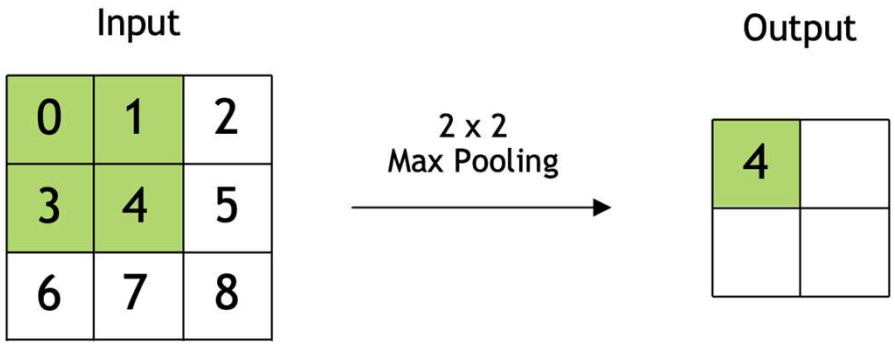


- K = 1
- Die Faltung erfolgt entlang der Kanaldimension.
- Eingabe und Ausgabe haben dieselbe räumliche Größe.
- Jedes Ausgabeelement entsteht durch eine Linearkombination der Elemente an derselben Position in den Eingabekanälen.
- Kann zur Dimensionsreduktion verwendet werden.

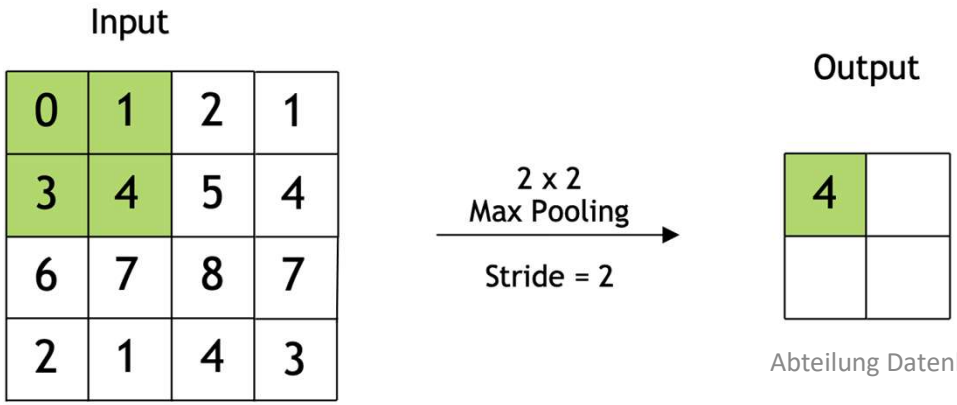
Pooling



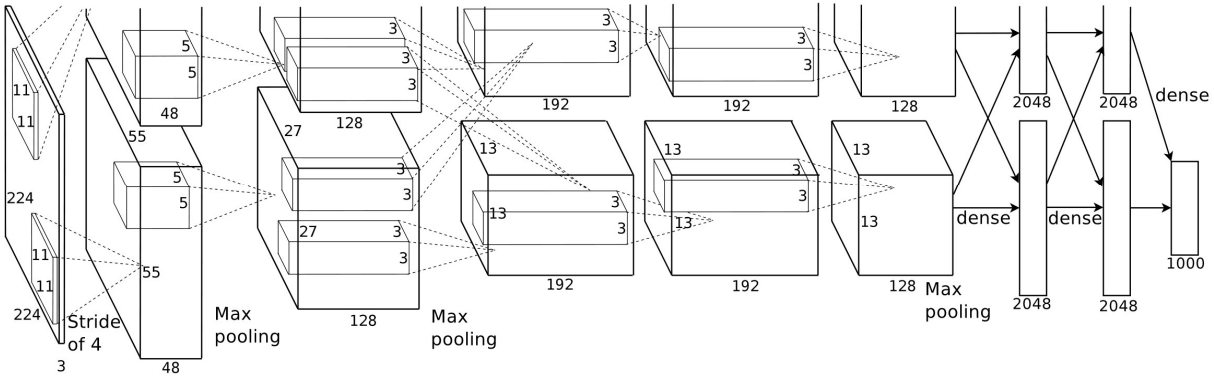
- Die räumliche Auflösung unserer verborgenen Darstellungen schrittweise verringern
- Maximum Pooling and Average Pooling
- Keine Parameter!

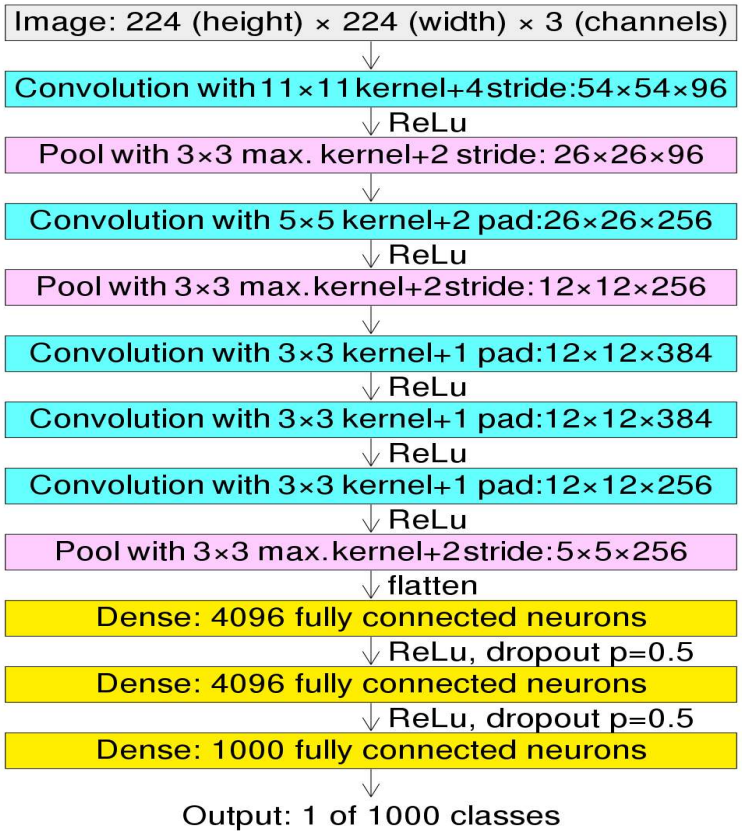


- Pooling kann Padding and Stride enthalten



Beispiel: AlexNet





AlexNet erreicht ein top-5 Fehler von 15.3%, mehr als 10.8% weniger als die des zweitplatzierten

Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems. 2012;25:1097-105.
https://en.wikipedia.org/wiki/AlexNet

CNN Kernel vs Finite Differenzen



Partielle Ableitungen

$$\frac{\partial \phi}{\partial x} = \frac{\phi(x+\delta x) - \phi(x-\delta x)}{2\delta x} + O(\delta x)^2 \qquad \frac{\partial}{\partial x} = \begin{bmatrix} -\frac{1}{2\delta x} & 0 & \frac{1}{2\delta x} \\ -\frac{1}{2\delta x} & 0 & \frac{1}{2\delta x} \\ -\frac{1}{2\delta x} & 0 & \frac{1}{2\delta x} \end{bmatrix}$$

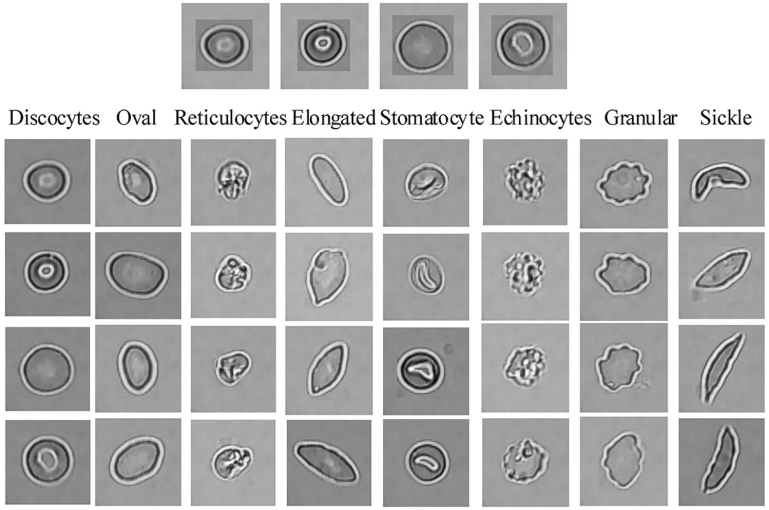
2D Laplacian (Δ)

Standard 5-Punkte
finite differenzen kernel

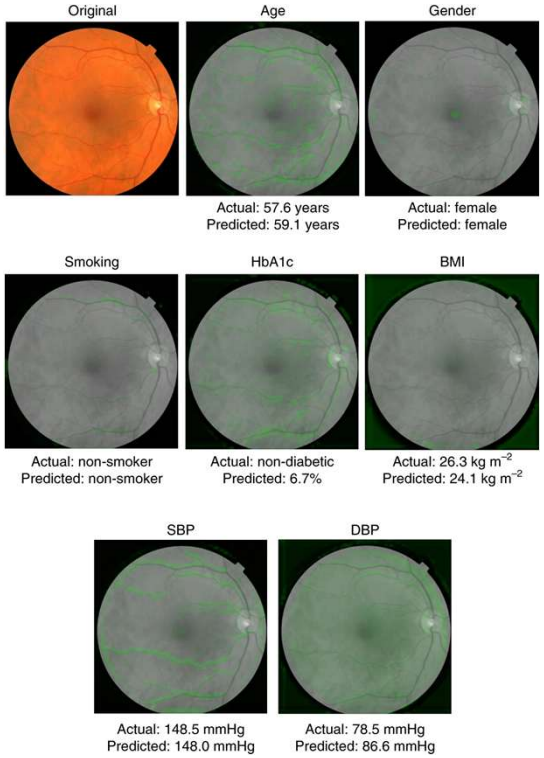
Lernbarer Kernel

$$\left.\frac{\partial^2 T}{\partial x^2}\right|_{i,j} + \left.\frac{\partial^2 T}{\partial y^2}\right|_{i,j} \rightarrow \frac{1}{h^2}\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \rightarrow \begin{bmatrix} k_{11} & k_{12} & k_{13} \\ k_{21} & k_{22} & k_{2,3} \\ k_{3,1} & k_{3,2} & k_{3,3} \end{bmatrix}$$

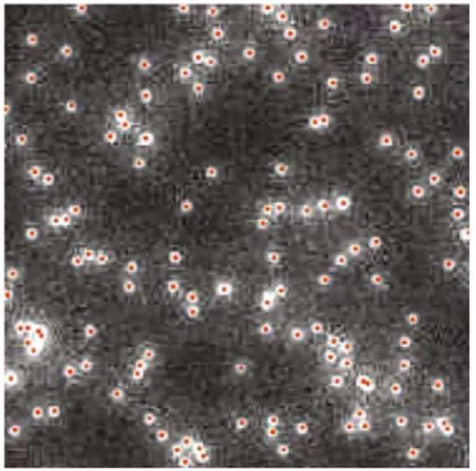
Anwendungen von CNNs in Wissenschaft und Engineering



PLOS Computational Biology, 2017



Nature Biomedical Engineering, 2018

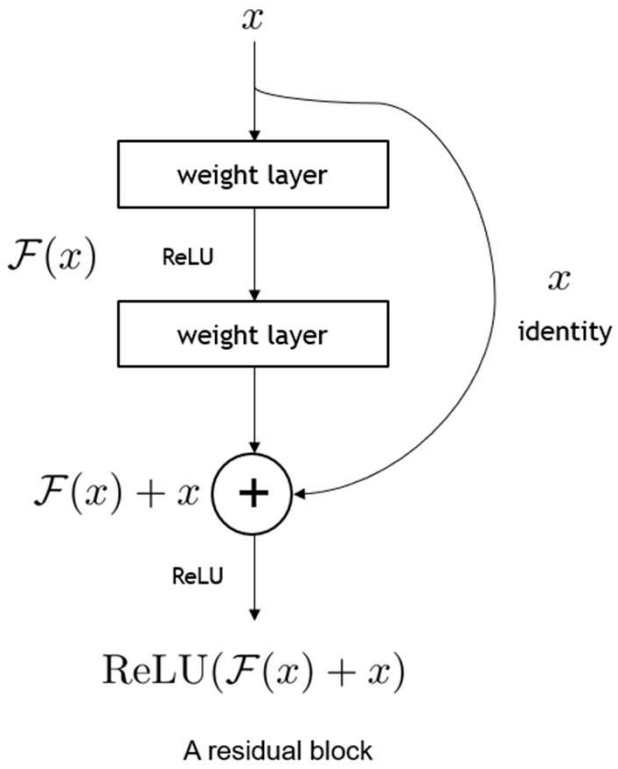


Nature Machine Intelligence, 2020

2.3 Resiudal Neural Network



- Tiefe Netzwerke sind schwer zu trainieren:
  Verschwindente Gradienten
- Kernidee: „Identitäts-Verbindung“,
  die eine oder mehrere Ebenen überspringt
- Weit verbreitet: einfach und mächtig

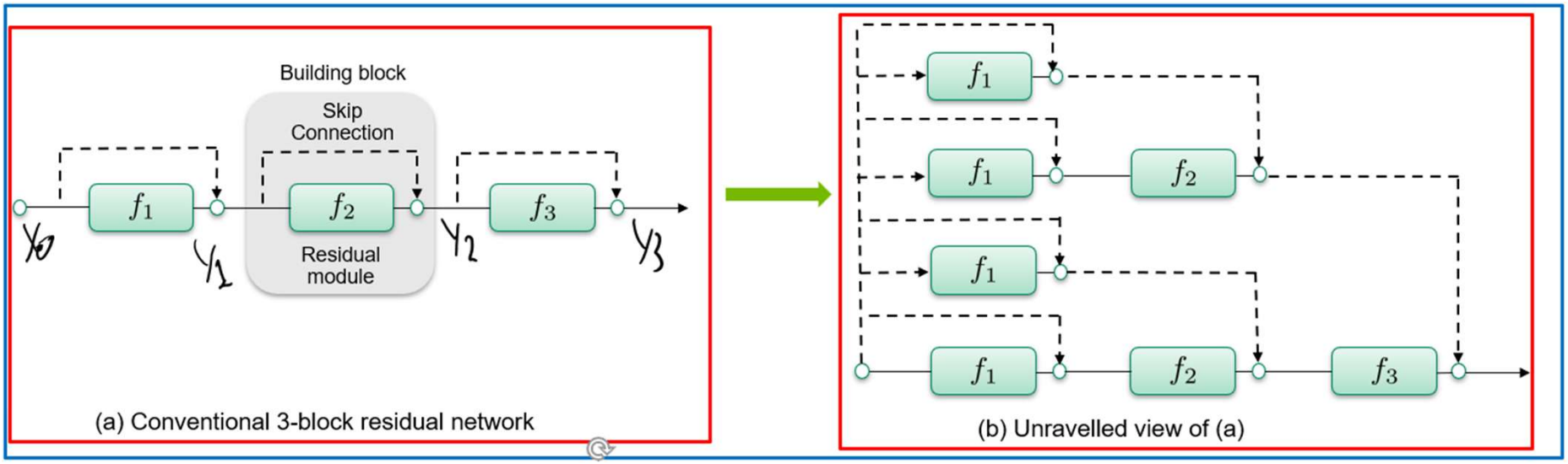


A residual block

He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. InProceedings of the IEEE conference on computer vision and pattern recognition 2016 (pp. 770-778).

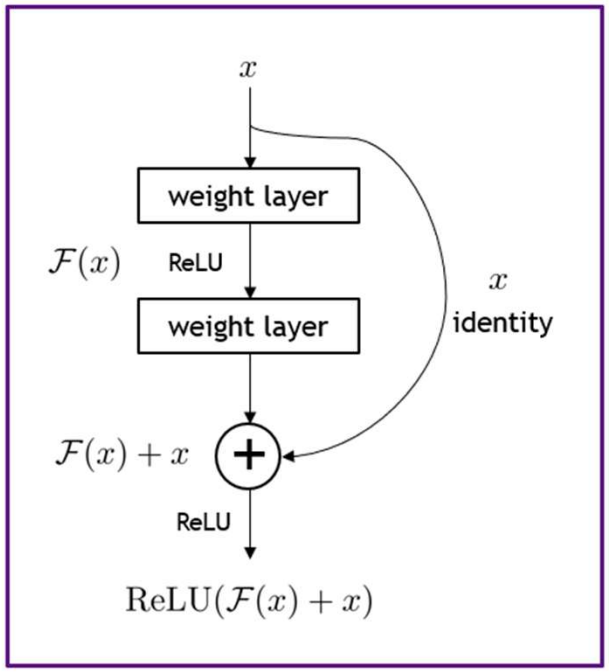
2.3 Residual Neural Network

$$
\begin{aligned}
y_3 &= y_2 + f_3(y_2) \\
&= [y_1 + f_2(y_1)] + f_3(y_1 + f_2(y_1)) \\
&= [y_0 + f_1(y_0) + f_2(y_0 + f_1(y_0))] + f_3(y_0 + f_1(y_0) + f_2(y_0 + f_1(y_0)))
\end{aligned}
$$



- Veit A, Wilber MJ, Belongie S. Residual networks behave like ensembles of relatively shallow networks. Advances in neural information processing systems. 2016;29:550-8.

Resnet = Forward Euler?



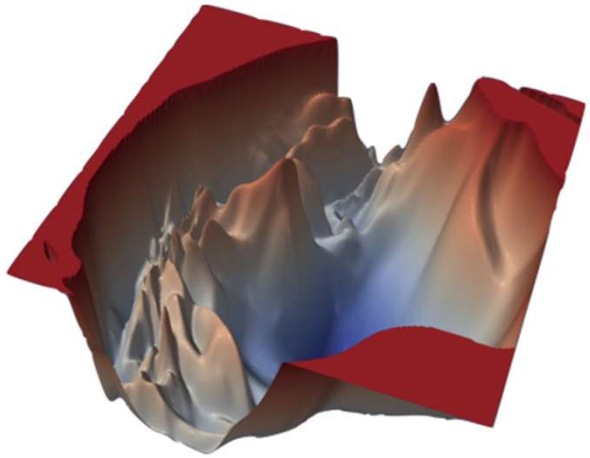
$$x_{n+1} = x_n + F(x_n)$$

$$u_{n+1} = u_n + \Delta t f(x_n) \quad \textbf{(Mass transport)}$$

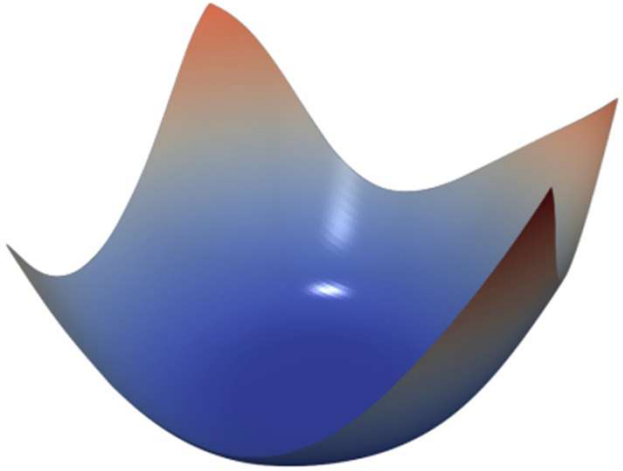
$$\frac{du}{dt} = f(u) \quad \textbf{(Dynamical system)}$$

Weinan E. A proposal on machine learning via dynamical systems. Communications in Mathematics and Statistics. 2017 Mar 1;5(1):1-1.

Optimization Landscape



ResNet-110, no skip connections



DenseNet-121 Layers, skip connections

Weinan E. A proposal on machine learning via dynamical systems. Communications in Mathematics and Statistics. 2017 Mar 1;5(1):1-1.

2.4 Graph Neural Networks: Motivation



Viele Daten besitzen eine Beziehungsstruktur

System	Knoten	Kanten
Molekül	Atome	Bindungen / räumliche Nähe
Verkehrsnetz	Kreuzungen / Sensoren	Straßenverbindungen
Physikalisches Netz	Gitterpunkte / Teilchen	Nachbarschaften / Wechselwirkung
Weitere Beispiele	Personen, Wörter, Bildregionen	Soziale, sprachliche oder räumliche Beziehungen

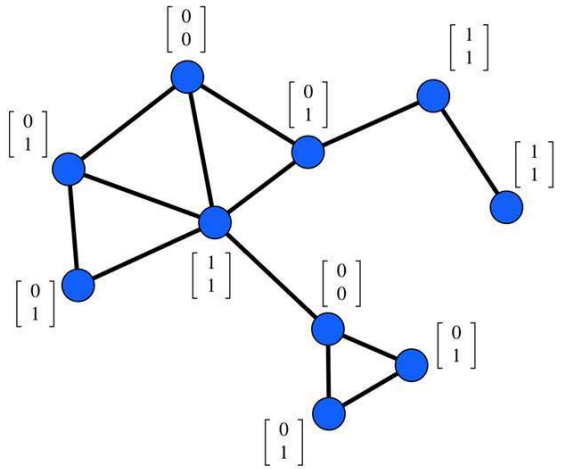
Ein GNN verarbeitet Merkmale zusammen mit der Struktur ihrer Beziehungen.

Graphen: Knoten und Kanten



$G = (V, E), \quad V = \{1, \ldots, N\}$

- V: Menge der Knoten, E: Menge der Kanten.
- Ungerichtet: Eine Kante verbindet zwei Knoten ohne Richtung.
- Gerichtet: Eine Kante besitzt eine Quell- und Zielrichtung.
- N(i): Nachbarschaft des Knotens i.



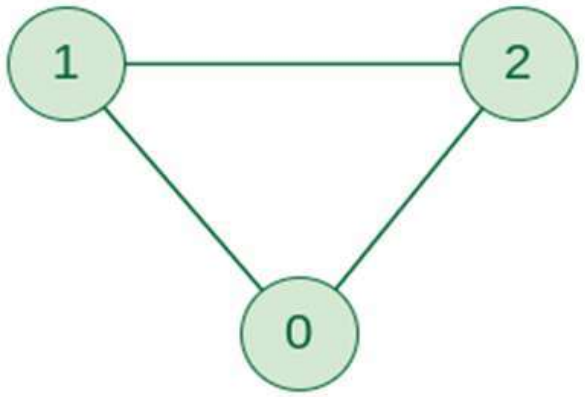
Nummerierung und Merkmale sind von der Topologie zu unterscheiden.

Für ungerichtete Graphen verwenden wir ungeordnete Knotenpaare {i, j}.

Adjazenzmatrix



Die Matrix beschreibt, welche Knoten verbunden sind



Ungerichteter Graph

A	0	1	2
0	0	1	1
1	1	0	1
2	1	1	0

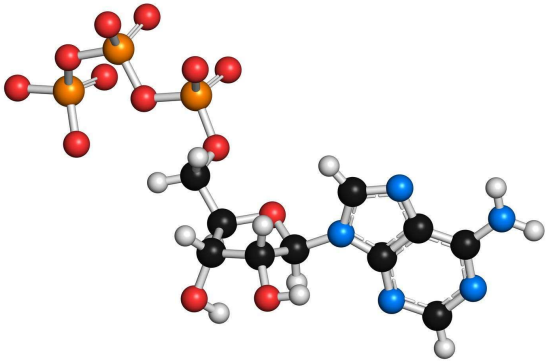
$A_{ij} = 1$ bei einer Kante, sonst $A_{ij} = 0$.

Ungerichtet: $A = A^T$. Ohne Selbstkanten: $A_{ii} = 0$.

Knoten- und Kantenmerkmale



Knotenmerkmale x_i

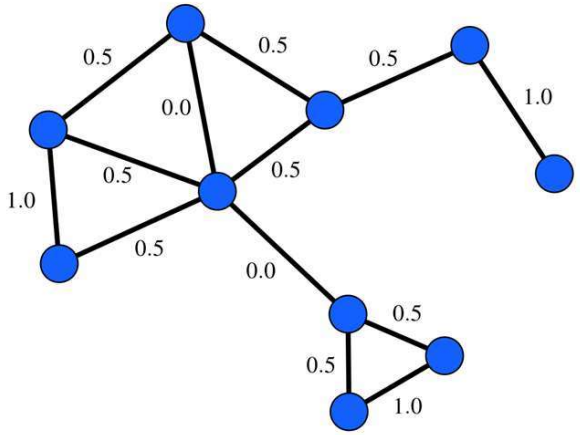


Beispiele: Temperatur, Geschwindigkeit,

Atomtyp oder Materialeigenschaft.

$X \in \mathbb{R}^{Nxf}$: ein Merkmalsvektor pro Knoten.

Kantenmerkmale e_{ij}



Beispiele: Abstand, Leitfähigkeit,

Bindungstyp oder relative Position.

Graph-Laplace-Operator



$$L = D - A, \qquad D_{ii} = \sum_j A_{ij}$$

Beispiel: drei Knoten, Kanten {1, 2} und {2, 3}

A

0	1	0
1	0	1
0	1	0

D

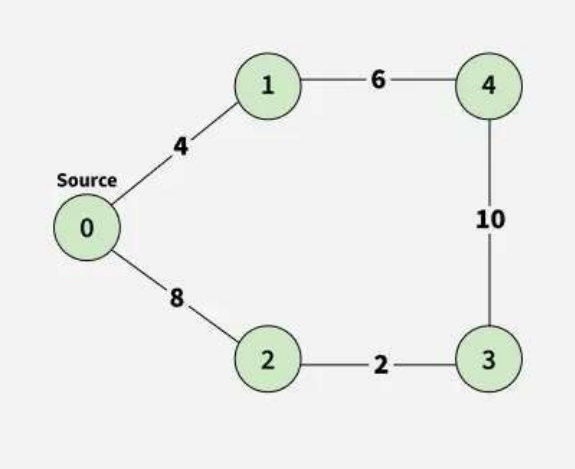
1	0	0
0	2	0
0	0	1

L = D − A

1	-1	0
-1	2	-1
0	-1	1

Klassische Graphalgorithmen

Aufgabe	Typisches Ziel
Kürzeste Wege / Distanzen	Günstigste Route, z. B. mit Dijkstra
Zyklen erkennen oder zählen	Rückkopplungen und geschlossene Wege finden
Communities und Link Prediction	Gruppen erkennen bzw. fehlende Kanten vorhersagen
Maximaler Fluss	Transport unter Kapazitätsbeschränkungen
Matching und Färbung	Knoten zuordnen bzw. konfliktfrei gruppieren



Bekannte Regeln und Zielfunktionen werden algorithmisch ausgewertet.

Warum Lernen auf Graphen?



Klassische Algorithmen und GNNs lösen unterschiedliche Aufgaben

- Viele klassische Graphprobleme besitzen effiziente exakte Algorithmen.
- Für manche Zielgrößen ist die passende Berechnungsregel unbekannt oder teuer.
- Ein GNN lernt diese Abbildung aus Beispielen und nutzt dabei die Graphstruktur.
- Nach dem Training kann es schnelle Vorhersagen liefern.

Beispiel: Molekülstruktur als Eingabe, gesuchte Materialeigenschaft als Ausgabe.

Vom CNN zum GNN



Eigenschaft	CNN	GNN
Datenstruktur	Regelmäßiges Gitter	Allgemeiner Graph
Nachbarschaft	Fester räumlicher Kernel	Durch Kanten vorgegeben
Anzahl der Nachbarn	Im Inneren typischerweise gleich	Kann von Knoten zu Knoten variieren
Geteilte Parameter	Gleicher Kernel an allen Positionen	Gleiche Nachrichten- und Updatefunktionen
Aggregation	Gewichtete lokale Summe	Z. B. Summe, Mittelwert oder Maximum

Ein GNN verallgemeinert die lokale Informationsverarbeitung auf unregelmäßige Strukturen.

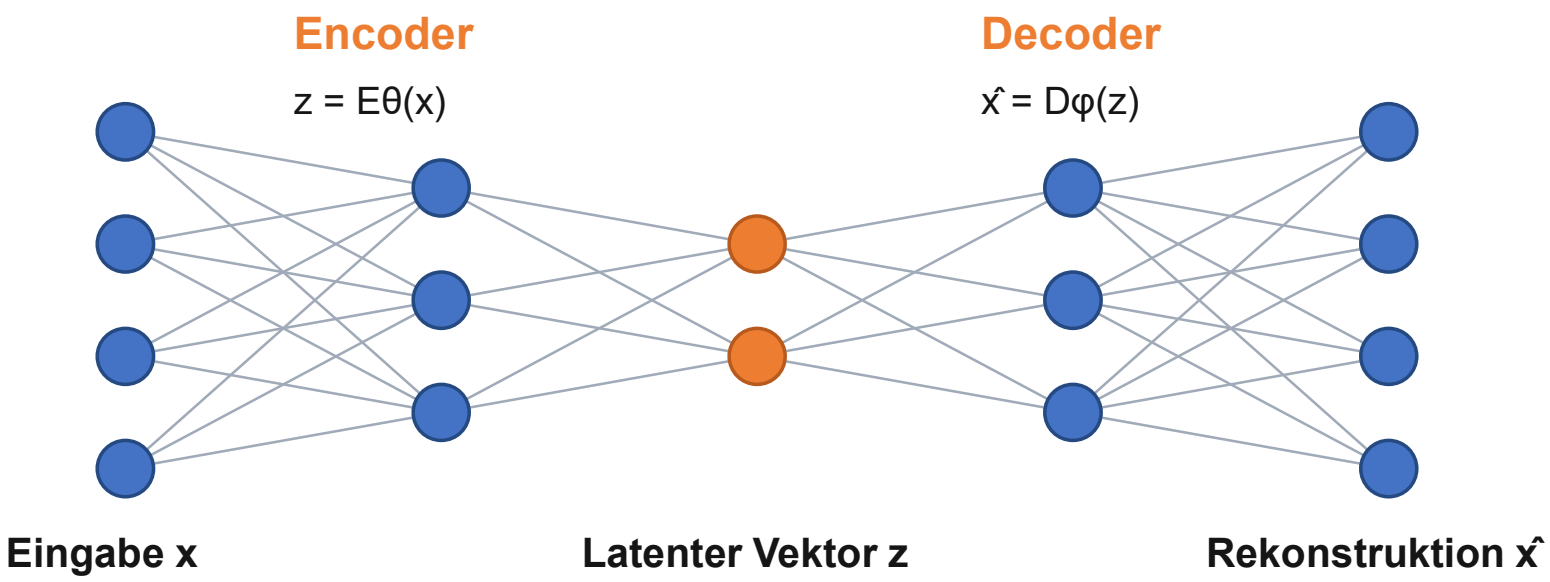
Vorhersagen auf Knoten, Kanten und Graphen



Ebene	Ausgabe	Beispiel
Knoten	Ein Wert / Vektor pro Knoten	Temperatur oder Geschwindigkeit an jedem Netzpunkt
Kante	Ein Wert / Vektor pro Kante	Fluss zwischen Zellen oder Existenz einer Verbindung
Gesamter Graph	Ein Wert / Vektor für das System	Molekülenergie oder Materialklasse

2.5 Autoencoder: Kompression und Rekonstruktion





- Rekonstruktion: Das Netzwerk lernt, seine Eingabe wiederherzustellen ($\hat{x} \approx x$).
- Dimensionsreduktion: Der Engpass verdichtet x zu einem latenten Code z.