

Radiologie 2023 · 63:204–208
<https://doi.org/10.1007/s00117-022-01101-8>
 Angenommen: 25. November 2022
 Online publiziert: 28. Dezember 2022
 © The Author(s), under exclusive licence to Springer Medizin Verlag GmbH, ein Teil von Springer Nature 2022

Redaktion

F. Bamberg, Freiburg (Leitung)
 U. Attenberger, Bonn
 M. Eisenblätter, Bielefeld
 I. Molwitz, Hamburg
 M. Notohamiprodjo, München
 B. Sigl, Wien
 A. A. Tavakoli, Mannheim
 L. Ullrich, München



Einführung in Methoden neuronaler Netzwerke

Leon M. Bischoff^{1,2} · Julian A. Luetkens^{1,2}

¹ Klinik für Diagnostische und Interventionelle Radiologie, Universitätsklinikum Bonn, Venusberg-Campus 1, Bonn, Deutschland

² Quantitative Imaging Lab Bonn (QILaB), Universitätsklinikum Bonn, Bonn, Deutschland

Künstliche Intelligenz als zentrale Methodik in Gesellschaft und Medizin

Das Forschungsfeld der künstlichen Intelligenz hat sich in den letzten 10 Jahren zu einer zentralen Thematik nicht nur im wissenschaftlichen, sondern auch im gesellschaftlichen und politischen Diskurs entwickelt. Insbesondere in der Medizin erhofft man sich durch den Einsatz der künstlichen Intelligenz eine ressourcenschonende Automatisierung alltäglicher Routine unter Aufrechterhaltung eines hohen Qualitätsniveaus, welches neben der Diagnostik von Erkrankungen auch die gezielte Therapieplanung und Prognoseeinschätzung mit einschließt. Die Methodik, die hierbei in verschiedensten Variationen zum Einsatz kommt, ist jedoch auf den ersten Blick für viele Personen ohne direkten mathematischen Hintergrund nur schwer verständlich. Dieser Beitrag zielt deshalb darauf ab, die grundlegende Methodik von neuronalen Netzwerken als wichtigstem und am häufigsten angewandten Werkzeug der künstlichen Intelligenz zu erläutern.

Was ist ein neuronales Netzwerk?

Vereinfacht gesagt ist ein neuronales Netzwerk nichts anderes als ein riesiges mathematisches Gleichungssystem mit meh-

reren Tausend, oft sogar Millionen oder gar Milliarden Variablen. Die kleinste Einheit eines solchen Netzwerkes nennt man Perzeptron (■ Abb. 1a). Ein Perzeptron besteht aus vier verschiedenen Teilen:

1. Inputwerte (x): Dies sind die Werte, die das Perzeptron verarbeiten muss. Sie können prinzipiell jeden beliebigen Wert annehmen. Bei Algorithmen zur Bilderkennung stellen in der ersten Schicht des Netzwerkes beispielsweise die Pixelwerte des Bildes die Inputwerte dar.
2. Gewichte (w) und Bias (b): Dies sind die Variablen, die während des Trainingsprozesses fortlaufend optimiert werden. Während die Gewichte mit den Inputwerten multipliziert werden, wird der Bias zu diesen addiert.
3. Summierung des Inputs (Σ): Sämtliche Zwischenergebnisse, die aus der Verrechnung der Inputwerte mit den Gewichten und dem Bias entstanden sind, werden addiert. So wird sichergestellt, dass alle Inputwerte einen Einfluss auf das Endergebnis haben.
4. Aktivierungsfunktion (A): Um das Ergebnis der Addition zu skalieren und sehr hohe und niedrige Werte zu vermeiden, wird eine Aktivierungsfunktion angewandt. Eine der am häufigsten verwendeten ist beispielsweise die Sigmoidfunktion (■ Abb. 1b). Werden hohe positive Werte (> 0) in



QR-Code scannen & Beitrag online lesen

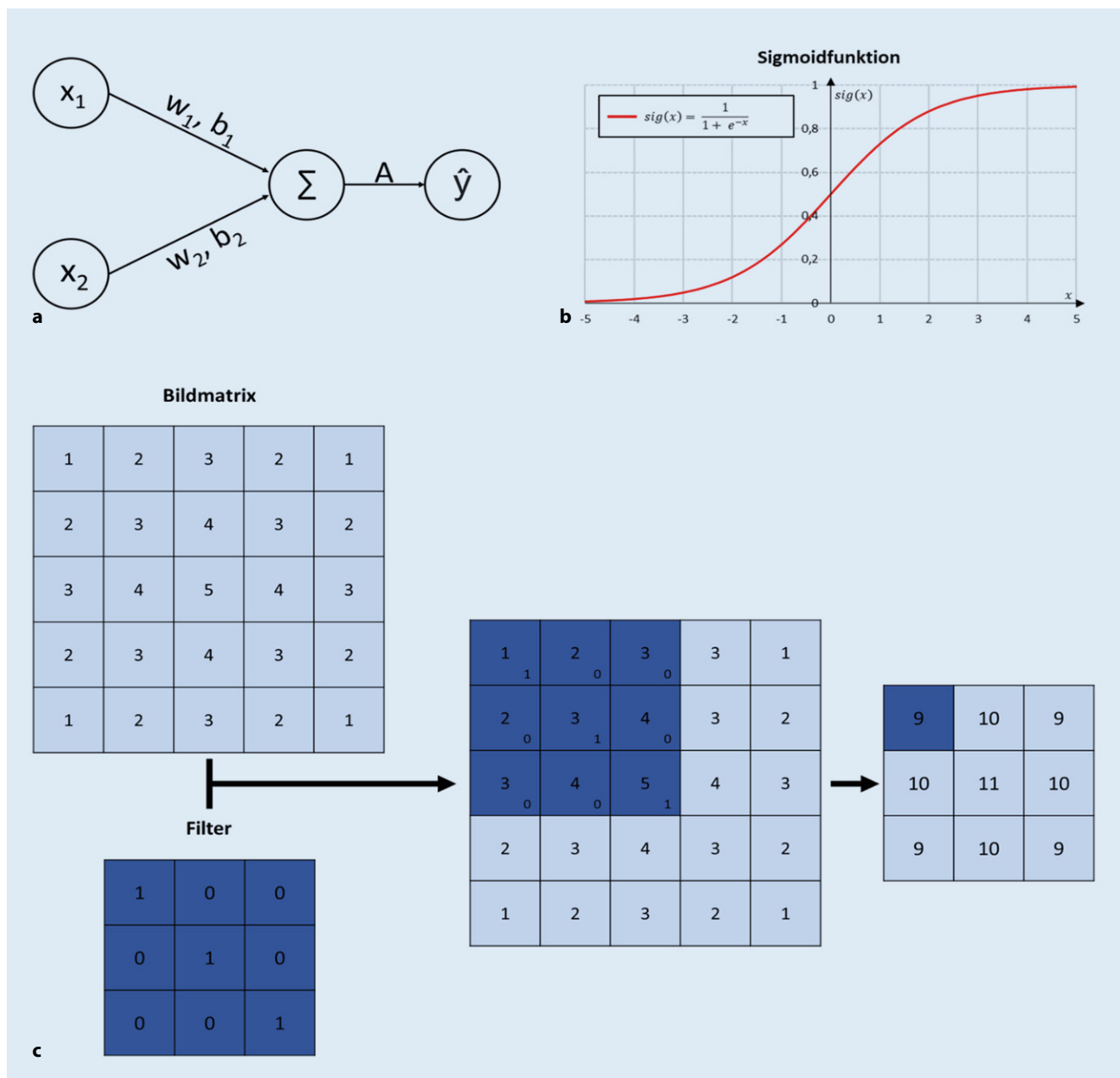


Abb. 1 ▲ Grundsätzliche Elemente neuronaler Netzwerke. **a** Aufbau eines Perzeptrons mit den Inputwerten (x), Gewichten (w), Bias (b), der Aktivierungsfunktion (A) und den Outputwerten ($\hat{y}$). **b** Graf der Sigmoidfunktion als Beispiel einer Aktivierungsfunktion. **c** Funktionsweise eines „convolutional neural networks“

diese Funktion eingegeben, nähert sich das Endergebnis asymptotisch dem Wert 1 an, bei negativen Werten (< 0) entsprechend dem Wert 0. So wird sichergestellt, dass das Ergebnis immer zwischen 0 und 1 liegt.

Zusammengefasst werden in einem Perzeptron viele verschiedene Inputwerte zu einem Outputwert ($\hat{y}$) verrechnet. Neuronale Netzwerke sind dabei in vielen hintereinander folgenden Schichten aufge-

baut. Die erste Schicht ist immer die Inputschicht, in die direkt die Zahlenwerte aus dem zu analysierenden Objekt eingegeben werden. Die letzte Schicht berechnet das finale Ergebnis (Abb. 2a). Alle dazwischenliegenden Schichten werden als versteckte Schichten bezeichnet und stellen im Wesentlichen eine Abstraktion der ursprünglichen Inputwerte in verschiedener Ausprägung dar. Neuronale Netzwerke unterscheiden sich dabei nicht nur in der Anzahl der Schichten, Perzeptrons und Va-

riablen, sondern auch in der Wahl der Aktivierungsfunktion und insbesondere der Perzeptronverschaltung [2].

Stark verbesserte Bilderkennung durch „convolutional neural networks“

Die automatische Bilderkennung ist eine der wichtigsten Anwendungsarten neuronaler Netzwerke, insbesondere in der Radiologie. Um Kontraste in Bildern so-

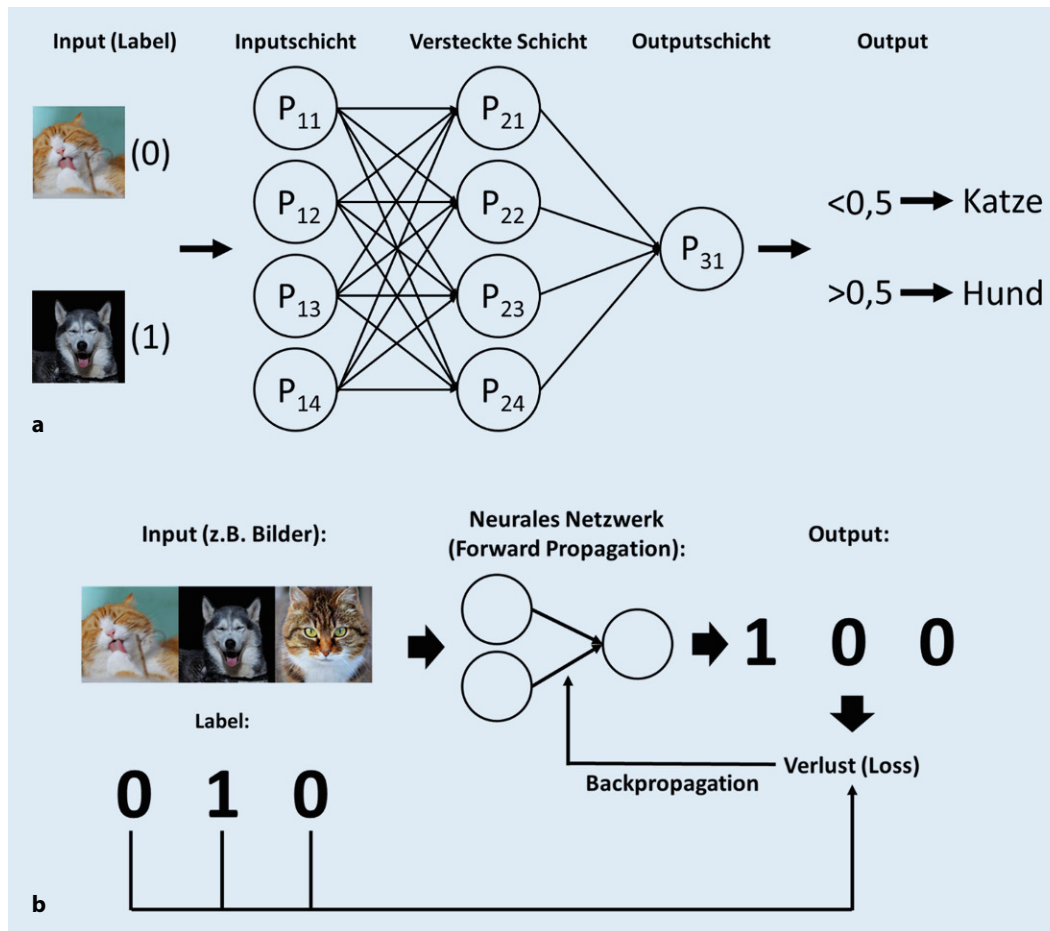


Abb. 2 ◀ Funktionsweise und Trainingsprozess neuronaler Netzwerke. **a** Das neuronale Netzwerk als Verschaltung mehrerer Perzeptrons (P) gibt eine Schätzung für jedes Bild ab (in diesem Fall Katze oder Hund). **b** Outputwerte und Label werden mittels der Verlustfunktion miteinander verglichen und abhängig hiervon in der Backpropagation die Variablen des Netzwerks angepasst. (Tierbilder unterliegen der Pixabay Lizenz; <https://pixabay.com/de/service/license/>)

wohl in horizontaler als auch in vertikaler Pixelrichtung besser zu erkennen, wurden sog. „convolutional neural networks“ (CNNs) entwickelt [1]. Die Inputwerte werden hierbei nicht separat voneinander betrachtet, sondern als Bildmatrix (Abb. 1c). Über diese Matrix bewegt sich nun ein Filter, beispielsweise mit einer Größe von 3×3 Feldern. Die einzelnen Felder dieses Filters stellen die Gewichte oder den Bias dar, die während des Trainingsprozesses angepasst werden. Jedes Gewicht des Filters wird mit dem jeweiligen Inputwert, über dem es sich befindet, multipliziert. Die Zwischenergebnisse eines Filters werden addiert und mittels einer Aktivierungsfunktion skaliert. Der Filter bewegt sich im Anschluss auf der Bildmatrix stets eine Position weiter, bis das Ende erreicht ist. Es entsteht eine neue, horizontal und vertikal minimal kleinere Bildmatrix, die eine bessere Abstraktion der Bildkontraste darstellt, als es in einem konventionellen neuronalen Netzwerk möglich wäre. Meist wird eine Kombination

von CNNs und konventionell verbundenen Schichten verwendet. Eine genaue Auflistung der wichtigsten Architekturen neuronaler Netzwerke und deren bevorzugte Anwendung findet sich in Tab. 1.

Minimierung der Verlustfunktion

Der zu analysierende Datensatz muss vor dem Training manuell annotiert werden. Im Falle des klassischen „Hunde-versus-Katzen“-Datensatzes bedeutet dies beispielsweise, dass alle Hunde als 1 und alle Katzen als 0 gelabelt werden [3]. Das neuronale Netzwerk erkennt nach abgeschlossenem Training bei einem Ergebnis der letzten Sigmoidfunktion von $>0,5$ einen Hund und bei $<0,5$ eine Katze (Abb. 2a). Damit ein neuronales Netzwerk abhängig von den initialen Inputwerten die richtigen finalen Outputwerte errechnet, muss es zuerst trainiert werden. Sämtliche Variablen des Netzwerks werden vor Beginn des Trainings zufällig initialisiert. Für jedes Bild des Da-

tensatzes, das im Anschluss das Netzwerk durchläuft, wird im folgenden Schritt ein Outputwert errechnet („forward propagation“). Der errechnete Wert wird mit dem ursprünglichen Label des Bildes verglichen (Abb. 2b); das gesamte Ziel des Trainings besteht in einer Minimierung des Unterschieds zwischen den beiden Größen. Dieser Vergleich wird durch die Verlustfunktion durchgeführt, beispielsweise mittels der binären Kreuzentropie:

$$L = -\frac{1}{N} \sum_{i=1}^N y_i \cdot \log(\hat{y}_i) + (1 - y_i) \cdot \log(1 - \hat{y}_i)$$

L: Verlust (Loss)
 N: Anzahl der Bilder
 y: Label
 $\hat{y}$: Outputwert

Zusammengefasst lässt sich diese Funktion so verstehen, dass abhängig vom Label jeweils einer der beiden Terme $y_i \cdot \log(\hat{y}_i)$ oder $(1 - y_i) \cdot \log(1 - \hat{y}_i)$

Tab. 1 Übersicht über die wichtigsten Architekturen neuronaler Netzwerke und deren Hauptanwendungen mit populären Beispielen		
Netzwerkarchitektur	Hauptanwendung	Beispiele
Convolutional neural network	Bildklassifikation	AlexNet, Inception, EfficientNet
	Objektdetektion	Faster R-CNN, Mask R-CNN, YOLO
Recurrent neural network	Computerlinguistik	LSTM, GRU
Transformer	Bildklassifikation, Objektdetektion, Computerlinguistik	GPT-2, BERT
<i>R-CNN</i> Region based recurrent neural network, <i>YOLO</i> You only look once, <i>LSTM</i> Long short-term memory, <i>GRU</i> Gated recurrent unit, <i>GPT-2</i> Generative Pre-trained Transformer, <i>BERT</i> Bidirectional Encoder Representations from Transformers		

Tab. 2 Überblick zu weiterführenden Internetseiten, die Methoden der künstlichen Intelligenz für Einsteiger, Fortgeschrittene und Profis erläutern. Beispielhaft sind ebenfalls vier Journals dargestellt, die regelmäßig neue Erkenntnisse und Methodiken zu künstlicher Intelligenz publizieren	
Internetseiten	Journals
www.medium.com www.towardsdatascience.com www.stackoverflow.com www.machinelearningmastery.com www.kaggle.com	Nature Machine Intelligence Machine Learning IEEE Transactions on Pattern Analysis and Machine Intelligence IEEE Transactions on Medical Imaging

gleich null ist und somit aus der Gleichung wegfällt. Der jeweils andere Term ist aufgrund des verwendeten Logarithmus bei $\log(1)$ gleich 0; dies wird nur erreicht, wenn der Outputwert sich nicht vom Label unterscheidet. Das Ergebnis aller Bilder, die zum gleichen Zeitpunkt das Netzwerk durchlaufen, wird gemittelt.

Optimierung des neuronalen Netzwerks mittels Backpropagation

Um eine Minimierung des Verlusts zu erreichen, müssen sämtliche Variablen des neuronalen Netzwerks fortlaufend angepasst werden. Dieser Prozess wird *Backpropagation* genannt [4]. Dies ist der letzte wichtige Baustein, um den generellen Trainingsprozess solcher Netzwerke besser verstehen zu können. Stellt man sich den errechneten Verlust als Funktion in Abhängigkeit einer einzelnen Variablen vor (der Verständlichkeit halber kann man sich diese Verlustverteilung als einfache quadratische Funktion vorstellen), so ist ersichtlich, dass das erwünschte Minimum durch eine Grafsteigung von 0 charakterisiert ist, wohingegen ein höherer Verlust mit einem positiven oder negativen Gradienten verbunden ist. Wird nun der Gradient der Verlustverteilung für diese Variable von dem ursprünglichen Wert der Varia-

ble subtrahiert, so nähert sich das Ergebnis dem Minimum des Verlusts an und wird als neuer Wert der Variable gesetzt. Die Lernrate α skaliert dabei den Gradienten, um große Schwankungen zu vermeiden:

$$w'_1 = w_1 - \alpha \cdot \frac{\partial L}{\partial w_1}$$

w'_1 : Optimiertes Gewicht
 w_1 : Ursprüngliches Gewicht
 α : Lernrate
 L : Verlust (Loss)

Dies wird für alle Variablen des Netzwerks separat berechnet, sodass nach Abschluss der Backpropagation ein optimiertes Gleichungssystem entsteht (■ Abb. 2b). Ein einzelner Durchgang wird dabei als Epoche bezeichnet, in der Realität wird ein Netzwerk meist über hunderte oder tausende Epochen trainiert.

Diese und weitere Methoden werden fortlaufend weiterentwickelt; für eine tiefergreifende Erläuterung verweisen wir auf die beispielhaft aufgelisteten Informationsquellen in ■ Tab. 2.

Übertragung der Eigenschaften eines neuronalen Netzwerks mittels Transferlernen

Eine der am häufigsten angewandten Trainingsstrategien ist das sog. Transferlernen [5]. Bei dieser Strategie wird sich zu Nutzen gemacht, dass Kontraste in Bildern eine generelle Eigenschaft und unabhängig vom abgebildeten Motiv sind. So sind Netzwerke, die bereits auf umfangreichen Datensätzen wie beispielsweise ImageNet trainiert wurden, ausgezeichnet in der Lage, die erlangte Fähigkeit der Differenzierung von Kontrasten nach erneutem kurzem Training auf andere Datensätze mit unterschiedlichen Bildern und Labeln zu übertragen. Dies führt nicht nur zu deutlich reduzierten Trainingszeiten, sondern erzielt auch oftmals bessere Ergebnisse.

Aufbau eines Datensatzes

Damit neuronale Netzwerke ihre *Magie* entfalten können, ist vorher eine sorgfältige Präparation des Datensatzes nötig. Meist wird der gesamte Datensatz zufällig in 3 Subdatensätze für unterschiedliche Zwecke aufgeteilt:

1. Trainingsdatensatz: Training des neuronalen Netzwerks
2. Validierungsdatensatz: Testung des neuronalen Netzwerkes *während* des Trainings auf bisher unbekannte Daten (z. B. nach jeder Epoche)
3. Testdatensatz: Testung des neuronalen Netzwerkes *nach* Abschluss des Trainings auf einen weiteren unbekannten Datensatz zur finalen Evaluation und Messung der Generalisierbarkeit

Die prozentuale Aufteilung der Datensätze ist frei wählbar, häufig verwendet wird beispielsweise ein Verhältnis von 70%/15%/15%.

Fazit

- Der Grundbaustein jedes neuronalen Netzwerks ist das Perzeptron, die zusammenschaltet ein großes Gleichungssystem ergeben.
- Neben der Struktur bilden die Verlustfunktion und die Backpropagation die wichtigsten methodischen Bausteine

ne, um ein neuronales Netzwerk zu trainieren.

- Für ein effizientes Training können Netzwerke mittels Transferlernen auf neue Datensätze angepasst werden.
- Eine Dreiteilung des Datensatzes für das Training, die Validierung und die abschließende Testung ermöglicht eine gute Generalisierbarkeit des neuronalen Netzwerks.

Korrespondenzadresse



Leon M. Bischoff

Klinik für Diagnostische und Interventionelle Radiologie, Universitätsklinikum Bonn, Venusberg-Campus 1
53127 Bonn, Deutschland
leon.bischoff@ukbonn.de

Einhaltung ethischer Richtlinien

Interessenkonflikt. L.M. Bischoff gibt an, dass kein Interessenkonflikt besteht. J.A. Luetkens ist als Referent für die Firma Philips Healthcare tätig und erhält Beraterhonorare von der Firma Bayer HealthCare.

Für diesen Beitrag wurden von den Autor/-innen keine Studien an Menschen oder Tieren durchgeführt. Für die aufgeführten Studien gelten die jeweils dort angegebenen ethischen Richtlinien.

Literatur

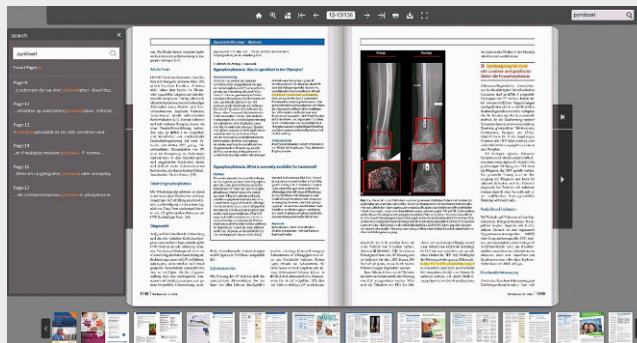
1. LeCun Y, Bottou L, Bengio Y et al (1998) Gradient-based learning applied to document recognition. *Proc IEEE* 86:2278–2324. <https://doi.org/10.1109/5.726791>
2. LeCun Y, Bengio Y, Hinton G (2015) Deep learning. *Nature* 521(7553):436–444. <https://doi.org/10.1038/nature14539>
3. Microsoft (2022) Kaggle cats and dogs dataset. <https://www.microsoft.com/en-us/download/details.aspx?id=54765>. Zugegriffen: 3. Okt. 2022
4. Rumelhart D, Hinton G, Williams R (1986) Learning representations by back-propagating errors. *Nature* 323:533–536. <https://doi.org/10.1038/323533a0>
5. Weiss K, Khoshgoftaar TM, Wang D (2016) A survey of transfer learning. *J Big Data* 3:9. <https://doi.org/10.1186/s40537-016-0043-6>



SpringerMedizin.de

Lesen Sie Ihre Fachzeitschrift auch als ePaper!

Als Abonnentin/Abonnent können Sie Ihre Zeitschrift in verschiedenen Formaten lesen. Wählen Sie je nach Vorliebe und Situation aus, ob Sie die Zeitschrift als Print-Ausgabe, in Form von einzelnen Beiträgen auf springermedizin.de oder aber als komplette, elektronische ePaper-Ausgabe lesen möchten.



Die ePaper sind die identische Form der gedruckten Ausgaben. Sie sind nutzbar auf verschiedenen Endgeräten wie PC, Tablet oder Smartphone

Das sind die Vorteile des ePapers:

- Das verlinkte Inhaltsverzeichnis führt Sie direkt zum gewünschten Beitrag.
- Eine Suchfunktion ermöglicht das Auffinden von Schlagworten innerhalb der Zeitschrift.
- Jede Ausgabe kann als PDF heruntergeladen und damit auch offline gelesen werden bzw. auch gespeichert oder ausgedruckt werden.
- Als Abonnentin/Abonnent haben Sie Zugang zu allen ePaper-Ausgaben ab 2016.

Sie finden die ePaper auf **SpringerMedizin.de** bei der jeweiligen Ausgabe Ihrer Fachzeitschrift. Klicken Sie auf den Button „**Ausgabe als ePaper lesen**“.