TECHNISCHE UNIVERSITÄT
CHEMNITZ

Fakultät für Naturwissenschaften
Physik kognitiver Prozesse
Struktur und Funktion kognitiver Systeme

Fortgeschrittenenpraktikum II – Versuch 12: Neuronale Netzwerke

Ort: C60.143

Verantwortliche Arbeitsgruppen: PHKP (Prof. Einhäuser-Treyer), SFKS (Prof. Bendixen)

Betreuung: W. Einhäuser-Treyer und Mitarbeitende der Arbeitsgruppen

Vorbereitung

Ein Teil dieses Praktikumsversuchs wird mit der Programmiersprache Matlab (The Mathworks, Natick, MA, USA) durchgeführt. Sofern Sie noch keine Erfahrung mit dieser Programmiersprache haben, stellt Ihnen Ihr Betreuer gerne einführendes Videomaterial zur Verfügung. Als Studentin oder Student der TU Chemnitz können Sie Matlab ohne eigene Kosten im Rahmen einer universitätsweiten Lizenz auch auf Ihrem heimischen Rechner nutzen. Hinweise zur Installation stellt Ihnen Ihr Betreuer bei Bedarf ebenfalls gerne zur Verfügung.

Einführung

Seit den späten 50er Jahren des 20. Jahrhunderts werden Modelle neuronaler Netzwerke als Verfahren zum automatisierten Lernen von Strukturen und Mustern eingesetzt. Nach einer Trainingsphase mit bekannten Daten erkennen diese Modelle in zuvor unbekannten Daten die erlernten Muster wieder (z.B.: Rosenblatt, 1958; Fukushima, 1980). Die Struktur solcher Netzwerke ist von den Verarbeitungsprozessen des menschlichen Gehirns inspiriert. Dabei knüpfen jedoch sowohl die Ausgestaltung der einzelnen Recheneinheiten, die grob den menschlichen Nervenzellen entsprechen, als auch die eingesetzten Lernverfahren nur lose an das biologische Vorbild an. Während um den Jahrtausendwechsel andere Verfahren des maschinellen Lernens in den Vordergrund traten, erleben seit einigen Jahren neuronale Netzwerke, nun vor allem in Form sogenannter «tiefer Netzwerke» (engl. *deep neural networks* – DNNs), einen enormen Bedeutungszuwachs mit einem breiten Anwendungsfeld (für eine Übersicht siehe z.B. LeCun et al., 2015). Der Praktikumsversuch führt zunächst in die Grundlagen neuronaler Netzwerke ein. Im zweiten Teil erfolgt eine Einführung in die Verarbeitung natürlicher Reize und das Konzept der Spärlichkeit im Zusammenhang früher visueller Verarbeitung. Im dritten Teil beschäftigen wir uns mit der Klassifikation natürlicher Szenen durch ein modernes DNN-Modell, insbesondere mit dessen Fehlern und deren Ursachen.

Die ersten beiden Teile haben bewusst einführenden Charakter, während der dritte freier gehalten ist. In allen Teilen gibt es freie Aufgaben, bei denen Sie einfache Simulationsexperimente durchführen und die Sie analog zu realweltlichen Experimenten dokumentieren sollen (also Ihre Hypothesen/Fragestellungen, das methodische Vorgehen [z.B. Wahl der Stimuli, Implementierung, usw.], die Ergebnisse und deren Interpretation darstellen). In den einführenden Teilen gibt es zudem Verständnisfragen, deren Beantwortung Sie in Ihre Ausarbeitung jeweils einfließen lassen können, wenn Sie zu Ihrem Simulationsexperiment hinleiten.

Tag 1, Aufgabe 1: McCulloch-Pitts-Neuron (zum Aufwärmen, noch ohne Computer)

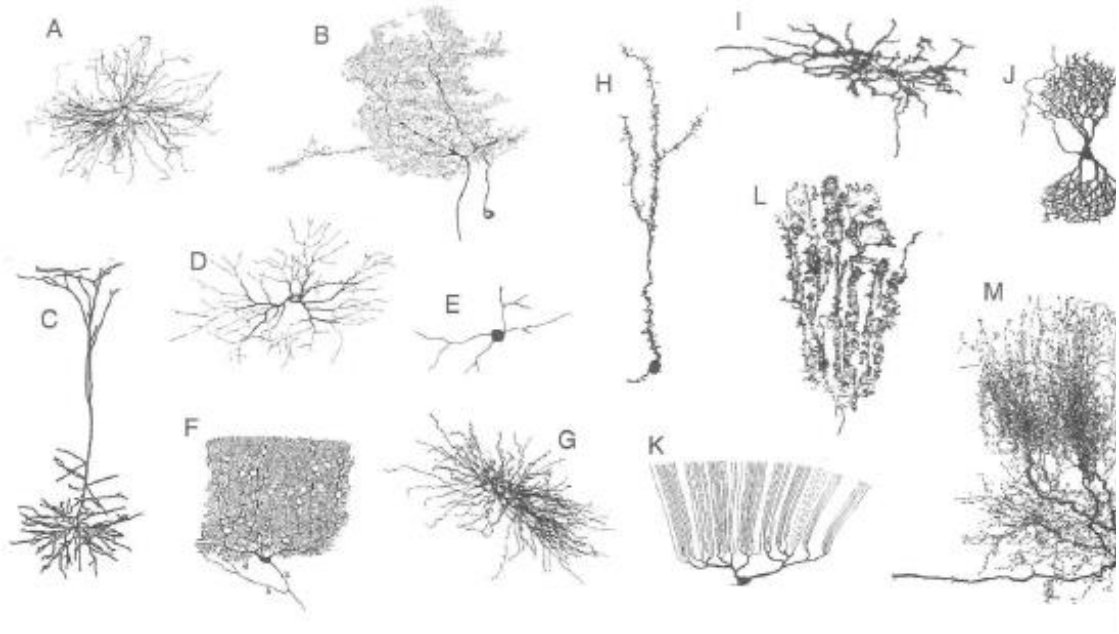

Abb. 1. Nervenzellen (Neurone). [Bildquelle: Koch (1998), S. 50]

Während eine reale Nervenzelle eine reichhaltige Struktur (man spricht von Morphologie) aufweist (Abb. 1), werden in Netzwerkmodellen häufig stark reduzierte Modelle, sogenannte Punkteneurone, verwendet. Diese summieren gewichtete Eingangssignale, prüfen, ob die Summe eine Schwelle überschreitet, und senden abhängig davon ein Signal oder nicht. Eine einfache Variante geht zurück auf die Arbeit von McCulloch und Pitts (1943).

Deren Neurone sind in Zeitschritt (t+1) aktiv, wenn in Zeitschritt t

- mehr als eine erregende (exzitatorische) Synapse aktiv ist **und**
- keine hemmende (inhibitorische) Synapse aktiv ist

Erregende Synapsen sind bei McCulloch & Pitts durch einen Punkt, hemmende durch eine Ellipse dargestellt, die Neurone selbst durch abgerundete Dreiecke (Abb. 2).

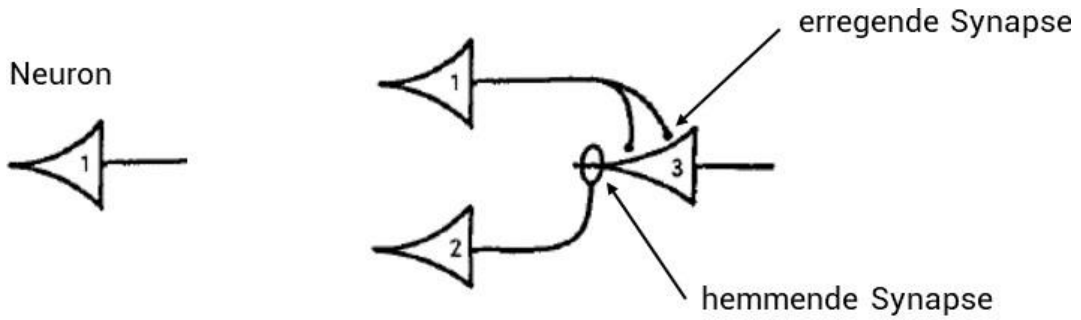

Abb. 2. Darstellung von Neuronen und Synapsen nach McCulloch & Pitts (1943).

Abbildung 3 zeigt das Beispiel einer logischen **UND**-Verknüpfung:

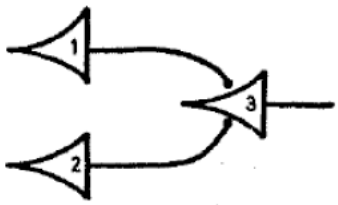

Abb. 3. Verschaltung dreier Neurone zu einer UND-Verknüpfung; Ausschnitt aus McCulloch & Pitts (1943), Abb. 1.

Neuron 3 ist zum Zeitpunkt t+1 genau dann aktiv, wenn Neuron 1 und Neuron 2 beide zum Zeitpunkt t aktiv waren. In diesem Fall summieren sich die Aktivitäten an den beiden Synapsen zu einem Wert größer als 1 (hier 2) und Neuron 3 wird aktiv. In allen anderen Fällen ist nur eine oder keine Synapse aktiv, die Summe bleibt kleiner oder gleich 1 und Neuron 3 bleibt inaktiv.

Man kann diesen Zusammenhang in einer sogenannten Wahrheitstabelle darstellen:

Eingang		**Ausgang**
Neuron 1 (zum Zeitpunkt t)	**Neuron 2 (zum Zeitpunkt t)**	**Neuron 3 (zum Zeitpunkt t+1)**
0	0	0
0	1	0
1	0	0
1	1	1

Dabei bedeutet eine «0», dass das Neuron inaktiv ist, eine «1» bedeutet, dass das Neuron aktiv ist. So lässt sich die berechnete Boolsche Funktion (hier das logische UND) darstellen.

McCulloch & Pitts zeigen in ihrer Arbeit, dass sich durch eine geeignete Verschaltung solcher Neurone jede Boolsche Funktion darstellen lässt. Abb. 4 ist der Originalarbeit von McCulloch & Pitts entnommen. ***Welche Funktion wird jeweils berechnet? Legen Sie dazu jeweils zunächst eine geeignete Wahrheitstabelle an und charakterisieren Sie die Funktion anhand dieser in Worten.***

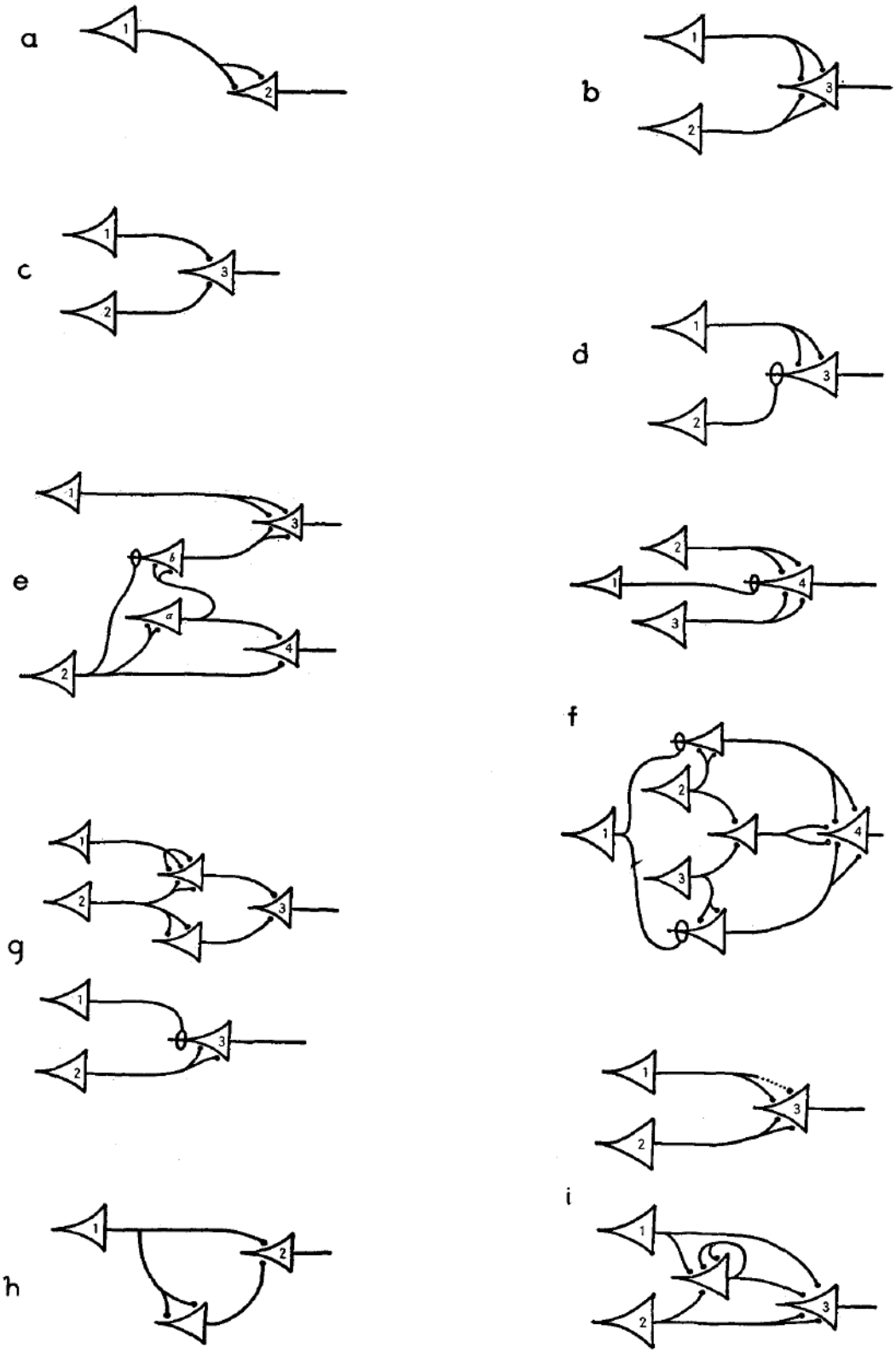

Abb. 4. Beispiele verschiedener mittels Modellneuronen implementierter Funktionen (McCulloch & Pitts (1943), Abb. 1).

Tag 1, Aufgabe 2: Perzeptron (Matlab)

In dem Modell von McCulloch und Pitts werden die Verbindungen zwischen Neuronen als unveränderlich angenommen. Ein wichtiger Schritt hin zu selbstlernenden Netzwerkmodellen war die Arbeit von Rosenblatt (1958), in der basierend auf Modellneuronen, die eine Verallgemeinerung des McCulloch-Pitts-Modells darstellen, erstmals veränderliche Verbindungen zum Lernen neuer Strukturen eingesetzt werden. Damit wird auf einer abstrakten Ebene ein wichtiger physiologischer Lernprozess, der ebenfalls auf der Veränderung von Verbindungen (im biologischen Fall sogenannte Synapsen) zwischen Nervenzellen beruht, im Modell abgebildet. Abgeleitet vom englischen Begriff für Wahrnehmung (*perception*) nennt Rosenblatt das Modell «Perzeptron» (engl. *perceptron*).

Zur Konstruktion eines einfachen Perzeptrons formulieren wir die Neurone als bestehend aus einem Vektor von Eingangssignalen x_i, einem Vektor von (synaptischen) Gewichten w_i und einer Schwelle θ. Überschreiten die gewichtet aufsummierten Eingangssignale die Schwelle, wird das Neuron aktiv (Ausgangssignal: +1), ansonsten bleibt es inaktiv (Ausgangssignal: -1)[1]. Das bedeutet, wir können die Aktivität des Neurons mit dem Skalarprodukt aus Eingangs- und Gewichtsvektor formulieren:

$$y = \begin{cases} +1 \; falls \sum_{i=1}^{n} w_i x_i - \theta > 0 \\ -1 \; sonst \end{cases}$$

Überlegen Sie sich an diesem Punkt:

- Inwiefern stellt diese Darstellung eine Verallgemeinerung des McCulloch-Pitts-Neurons dar, d.h. wie erhalten Sie deren Modell mit dieser Darstellung?
- Welche geometrische Struktur im n-dimensionalen Raum der Eingangssignale wird durch das Neuron dargestellt? Im 2- und 3-dimensionalen Fall (n=2 bzw. n=3) hilft die Anschauung.
- Welche Eigenschaften teilt solch ein Neuron mit dem biologischen Vorbild, welche nicht?

Das Training des Perzeptrons zur Klassifikation erfolgt nun anhand von Trainingsbeispielen, bei dem die korrekte Klassenzuordnung bekannt ist (man spricht von «überwachtem» Lernen). Formal sind Ihnen Paare von Eingangsvektoren **x** und zugehörigen Ausgangssignalen y^{true} gegeben und Sie müssen **w** und θ so bestimmen, dass alle diese Paare die obige Gleichung erfüllen.

- Unter welcher Bedingung ist das möglich?

Sofern eine solche Lösung existiert, kann sie durch die Perzeptron-Lernregel gefunden werden. Zunächst werden alle Gewichte zufällig initialisiert. Dann iterieren Sie über alle Trainingsbeispiele (**x,** y^{true}), worin y^{true} die tatsächliche (bekannte) Klasse zum Vektor **x** bezeichnet. Wenn das nach obiger Formel berechnete y dem y^{true} entspricht, lassen Sie die Gewichte unverändert, unterscheiden sie sich, ändern Sie den Gewichtsvektor um $\varepsilon(y^{true}-y)$**x**,

[1] Die Verwendung von +1 und -1 anstelle der sonst üblichen 1 und 0 vereinfacht an manchen Stellen die mathematische Darstellung, hat aber keine tiefere Bedeutung.

worin ε die Lernrate bezeichnet. D.h. die Änderung des Gewichtes hängt von der Richtung des Fehlers und dem Input ab (der Fall $y=y^{true}$ ist offensichtlich enthalten, in diesem Fall ist die Klammer 0 und keine Änderung geschieht).

Aufgaben:

- Implementieren Sie ein Perzeptron in Matlab für n=2. Ihr Betreuer stellt Ihnen dazu Code-Teile zur Verfügung, die Sie ergänzen können.
- Experimentieren Sie mit verschiedenen Beispielen, um folgende Fragen zu beantworten:
  - Welchen Einfluss hat die Lernrate auf das Lernverhalten?
  - Wenn eine Lösung existiert, ist diese eindeutig?
  - Was passiert, wenn keine Lösung existiert?
  - Hat die Reihenfolge der Trainingsdaten einen Einfluss auf das Ergebnis?
- Versuchen Sie, Ihre Beobachtungen theoretisch zu begründen.
- Erweitern Sie das Perzeptron auf mehr als 2 Eingangsdimensionen. Was beobachten Sie?

Tag 2, Aufgabe 1: Natürliche Szenen

In dieser Übung lernen Sie zunächst einige Eigenschaften von Bildern natürlicher Szenen sowie die Nützlichkeit der zweidimensionalen Fouriertransformation kennen. Durch eine Fouriertransformation kann ein Bild (allgemeiner ein 2-dimensionales Signal) in Phasen- und Amplitudenanteil zerlegt werden.

Vorgehen und Aufgabe:

- Machen Sie sich mit Matlabs Funktionen für die diskrete Fouriertransformation (fft2.m, ifft2.m, fftshift.m) sowie mit den Funktionen zum Einlesen (imread.m) und Darstellen von Bildern (image.m, imagesc.m) vertraut.
- Wählen Sie zwei beliebige, hinreichend unterschiedliche Bilder (z.B. eine Landschaftsszene und eine Innenraumszene) und schneiden Sie diese auf die gleiche Größe. Wir beschränken uns auf eine Graustufenabbildung (mit rgb2gray.m können Sie Farbbilder in Graustufenbilder umwandeln).
- Kombinieren Sie im Fourierraum das Phasenspektrum einer Szene mit dem Amplitudenspektrum der anderen und transformieren Sie die Kombination zurück. Was beobachten Sie und was schließen Sie daraus?
- Fügen Sie vor der Rücktransformation dem Phasen- und/oder dem Amplitudenspektrum Rauschen hinzu. Was beobachten Sie? Warum enthält das rücktransformierte «Bild» komplexe Zahlen? Was passiert, wenn Sie den Realteil oder den Betrag des rücktransformierten Bildes anzeigen?

Tag 2, Aufgabe 2: Spärlichkeit

In dieser Übung versuchen wir die Ergebnisse einer einflussreichen Arbeit (Olshausen & Field, 1996) zur sogenannten Spärlichkeitskodierung zu replizieren. Grundlegende Idee ist es, ein Bild I(x,y) bezüglich einer Basis darzustellen. Das Bild ist dann gegeben durch

$$I(x,y) = \sum_{i=1}^{n} a_i \phi_i(x,y)$$

Darin sind die a_i bildspezifische Koeffizienten und $\phi_i(x,y)$ eine Basis (für alle Bilder). Olshausen und Field fragen nun, wie die Basis zu wählen ist, damit die Koeffizienten eine sogenannte Spärlichkeitseigenschaft erfüllen. Intuitiv sagt diese Eigenschaft aus, dass für ein gegebenes Bild nur wenige Basisvektoren einen substantiellen Beitrag zum Bild leisten[2]. Dazu wird eine monotone nichtlineare Funktion S der Koeffizienten berechnet, wobei die Koeffizienten noch durch die Standardabweichung aller Pixelwerte skaliert wird: $S(a_i/\sigma)$. Um die Spärlichkeit zu maximieren, während gleichzeitig die Basis noch alle Bilder möglichst gut dargestellt werden sollen, wird der quadrierte Rekonstruktionsfehler für eine gegebene Basis betrachtet

$$\sum_{x,y} \left(I(x,y) - \sum_{i=1}^{n} a_i \phi_i(x,y) \right)^2$$

und dieser gemeinsam mit der Summe über die Spärlichkeit minimiert

$$\sum_{x,y} \left(I(x,y) - \sum_{i=1}^{n} a_i \phi_i(x,y) \right)^2 + \lambda \sum_{i} S\left(\frac{a_i}{\sigma}\right) \overset{!}{=} min$$

Der zentrale Befund von Olshausen & Field besagt, dass die resultierenden Basisvektoren, wenn man Sie im x,y-Raum betrachtet, lokalisierten orientierten Filtern entsprechen, die den sogenannten einfachen Zellen (Hubel & Wiesel, 1962) in der primären Sehrinde (V1) ähneln.

Aufgabe:

- Implementieren Sie durch Ergänzung der von uns bereitgestellten Code-Teile das Modell.
- Testen Sie mit dem von uns zur Verfügung gestellten Bilddatensatz verschiedene Funktionen S, z.B. $|x|$, $-\exp(-x^2)$, $\log(1+x^2)$. Was beobachten Sie für die Basisfunktionen? Wie interpretieren Sie dieses Ergebnis?
- Probieren Sie andere Funktionen aus, z.B. einfach S(x)=x? Was beobachten Sie nun? Wie interpretieren Sie dieses Ergebnis?
- Was geschieht, wenn Sie einen anderen Bilddatensatz wählen oder weniger Bilder aus dem ursprünglichen Datensatz?
- Was geschieht, wenn Sie n verkleinern?

[2] Etwas technischer ausgedrückt soll die Verteilung der Koeffizienten eine positive Kurtosis haben, also mehr «extreme» Werte besitzen als eine Normalverteilung gleicher Varianz.

Tag 3 DNNs

In dieser Übung beschäftigen wir uns mit tiefen neuronalen Netzwerken. Dazu werden Sie verschiedene Arten von Neuronalen Netzen trainieren und auch auf ein vortrainiertes Netz zurückgreifen.

Aufgaben

- Trainieren Sie ein Multi-Layer-Perceptron (MLP) zur MNIST-Klassifikation. Der MNIST-Datensatz (http://yann.lecun.com/exdb/mnist/) enthält handgeschriebene Ziffern und ist so etwas wie das "Hallo Welt" des Deep Learnings.
- In der Bildklassifikation spielen Convolutional Neural Networks (CNNs) eine wichtige Rolle. Machen Sie sich mit der grundlegenden Funktionsweise von CNNs vertraut: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53 und trainieren Sie ein CNN zur MNIST-Klassifikation.
- Untersuchen Sie, wie gut die Netzwerke diese Klassifikationsaufgabe erfüllen. Achten Sie darauf, dass Sie zur Gütebestimmung ein unverzerrtes Maß nutzen.
- Verwenden Sie ein vortrainiertes CNN und trainieren Sie es für eine komplexere Klassifikationsaufgabe (z.B. Hund/Katze oder Tier/Nichttier).
- Bestimmen Sie wieder die Güte des Klassifikationsergebnisses und lassen Sie sich einige Fehlklassifikationen ausgeben.
- Stellen Sie Hypothesen auf, welche Bildeigenschaften zu Fehlklassifikationen führen und welche nicht.
- Erstellen Sie neues Testbildmaterial, um Ihre Hypothesen zu testen, indem Sie gezielt Bildeingenschaften manipulieren. Führen Sie die Klassifikationsaufgabe erneut durch. Analysieren Sie, ob Sie Evidenz für oder gegen Ihre Hypothese finden.
- Diskutieren Sie Ihr Ergebnis auch in Hinblick auf die Frage der Repräsentativität des Trainingsmaterials.

Literatur

Fukushima, K. (1980). Neocognitron: a self organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics, 36*(4), 193-202.

Hubel, D. H., & Wiesel, T. N. (1962). Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of Physiology, 160*(1), 106-154.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature, 521*(7553), 436-444.

McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics, 5*(4), 115-133.

Koch, C. (1999). *Biophysics of Computation: Information Processing in Single Neurons*. Oxford University Press.

Olshausen, B. A., & Field, D. J. (1996). Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature, 381*(6583), 607-609.

Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review, 65*(6), 386-408.