Computer lernen nach dem Vorbild des Gehirns

Entwicklung eines künstlichen neuronalen Netzwerkes

F. G. Hüske, W. Baumgartner, I. Heil und J. Bohrmann

Bekanntermaßen geht das Lernen für ein „Ausreichend“ in einer Prüfung sehr viel schneller als für ein „Sehr gut“. Tatsächlich dauert nur das Training länger, nicht die eigentliche Lösung der Aufgabe. Dies zeigt sich auch bei künstlichen neuronalen Netzwerken.

Stichwörter: künstliches neuronales Netzwerk, künstliches Neuron, Lernen, künstliche Intelligenz, Bildverarbeitung

1 Einleitung

„Das menschliche Gehirn zu verstehen, stellt die größte wissenschaftliche und intellektuelle Herausforderung dar, der sich Menschen jemals gegenübersehen.“ [1]

Die Modellierung natürlicher neuronaler Systeme durch so genannte künstliche neuronale Netzwerke (KNN) ist eine mit der Entwicklung des Computers entstandene Vorgehensweise, die zum Verständnis des menschlichen Gehirns beitragen kann. KNN bieten die Möglichkeit, das Verhalten kleiner, idealisierter Versionen eines natürlichen neuronalen Netzwerks zu simulieren und daraus Rückschlüsse auf die Funktionsweise eines komplexen Gehirns zu ziehen. „Wissenschaftler [...] haben ein simuliertes Gehirn aus 1000 Computern gebaut [...] Das System erwies sich als besonders gut beim Erkennen von Menschen und Katzen [...] Entgegen der landläufigen Meinung zeigen unsere experimentellen Ergebnisse, dass es möglich ist, ein Gesichtserkennungssystem anzulernen, ohne zu kennzeichnen, ob auf einem Bild ein Gesicht zu sehen ist oder nicht [...]“ [2]

KNN werden heute bereits in vielen Bereichen des täglichen Lebens eingesetzt, ohne dass wir davon Notiz nehmen, beispielsweise bei der Bildverarbeitung, bei Sprach- und Schriftenkennung, in der Sicherheitstechnik und Medizin. Die Anwendungsgebiete von KNN sind somit äußerst vielfältig und lassen sich in drei Bereiche einteilen: (1) Anwendungen, die dem Verständnis natürlicher neuronaler Netzwerke dienen (d. h., dem Verständnis von Gehirnen und ihren Funktionen), (2) theoretische Anwendungen in der Informatik und (3) die genannten praktischen Anwendungen.

Selbstverständlich lässt sich das menschliche Gehirn durch Simulationen mit KNN bislang nicht in Gänze verstehen, doch lassen sich einige seiner Eigenschaften damit recht gut beschreiben. Eine dieser Eigenschaften ist das Lernen. Zunächst sollen hier Aufbau und Funktionsweise von KNN und deren Verwandtschaft mit natürlichen Vorbildern erläutert werden. Der Themenbereich lässt sich auf diese Weise gut in den Biologieunter-


Literatur
Signalwetterleitung ...

in einem natürlichen Neuron

Aufnahme von Aktionspotenzialen über synaptische Endknöpfchen (etc.)


Verrechnung aller eingehenden Signale zum entsprechenden Membranpotenzial im Zellkörper

Vergleich des neuen Membranpotenzials mit dem Schwellenpotenzial im Axonhügel

1. Möglichkeit:
Membranpotenzial > Schwellenpotenzial
→ Aktionspotenziale ausgelöst

2. Möglichkeit:
Membranpotenzial ≤ Schwellenpotenzial
→ keine Aktionspotenziale ausgelöst

in einem künstlichen Neuron

Aufnahme der Eingabevariablen (0 bzw. 1)

Summation der gewichteten Eingabevariablen durch die Inputfunktion

Berechnung der Aktivität durch die Transferfunktion

Vergleich des Netznets mit dem Schwellenwert durch die Outputfunktion

1. Möglichkeit:
Netzinput > Schwellenwert
→ Outputfunktion aktiviert („1“)

2. Möglichkeit:
Netzinput ≤ Schwellenwert
→ Outputfunktion inaktiv („0“)

Tab. 1: Vergleich von natürlichen und künstlichen Neuronen im Hinblick auf die Signalwetterleitung

richt der Qualifikationsphase (Q1 bzw. Q2) einbinden und auch mit dem Mathematik- und Informatikunterricht verknüpfen. Anschließend werden die Erstellung eines Lernprogramms und seine Arbeitsweise vorgestellt.

2. Der Aufbau künstlicher neuronaler Netze

KNN sind netzwerkartige Verkettungen „künstlicher Neuronen“. Sie sind Forschungsgegenstand der Neuroinformatik und ein Zweig der künstlichen Intelligenz. Es handelt sich dabei um zahlreiche kleine Einheiten, welche hochgradig miteinander verknüpft sind. Die einzelnen Einheiten oder Neuronen können ihre Verbindungen untereinander verändern. Dadurch sind KNN adaptiv, d.h. sie „lernen“. Obwohl die Neuronen für sich genommen lediglich simple Aufgaben erfüllen können, ist ein KNN in der Lage, hochkomplexe Aufgaben zu lösen.


3. Das künstliche Neuron


Beim künstlichen wie natürlichen Neuronen der beschriebenen Art werden ankommende, durch die synaptische Stärke „gewichtete“ Signale zunächst aufsummiert. Aus dieser Summe wird der Aktivitätszustand des Neurons bestimmt. Dieser Aktivitätszustand wird anschließend mit einem Schwellenwert verglichen. Wird der Schwellenwert überschritten („überkritische Aktivität oder Erregung“), so wird ein Ausgangssignal (eine Aktionspotenzial) ausgelöst. Dieses Ausgangssignal wird auf andere Neuronen übertragen. Dieses „Feuern“ wird in digitalen künstlichen Neuronen durch die Zählerwerte 1 und 0 reprä-

sientiert, wobei 1 einem Aktionspotenzial und 0 keinem Aktionspotenzial entspricht. Genau wie sein natürliches Pendant, das künstliche Neuron seinen Input über Verbindungen zu anderen Neuronen nimmt, können auch sensorische Neurone sein, die Messdaten generieren.

4 Mathematische Übersetzung der biologischen Signalweiterleitung


\[
\begin{align*}
\text{Inputfunktion } \text{net} & = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{pmatrix} \\
\text{Transferfunktion } a & = \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_M \end{pmatrix} \\
\text{Outputfunktion } o & = \begin{pmatrix} o_1 \\ o_2 \\ \vdots \\ o_M \end{pmatrix}
\end{align*}
\]

Die drei genannten Prozesse entsprechen den natürlichen Vorgängen.

Analogie: Die Inputfunktion entspricht der Signalaufnahme an den Dendriten, die Transferfunktion entspricht der Integration der Signale am Soma, und die Outputfunktion entspricht der Gene-

rierung des Aktionspotenzials am Axonhügel bei überkritischer Aktivität des Neurons.

Durch Wahl einer geeigneten Netzwerktopographie sowie geeigneter Input-, Transfer- und Outputfunktionen kann ein für unterschiedliche Zwecke geeignetes Netzwerk geschaffen werden. Im vorliegenden Fall wird als Inputfunktion die gewichtete Summe gewählt, wobei sich der Netzinput für das i-te Neuron wie folgt errechnet:

\[
\text{net}_i = \sum_{j=1}^{N} x_j \cdot w_{ij}
\]

Die Gewichte \( w_{ij} \) entsprechen der synaptischen Stärke, also dem Einfluss des Neurons \( i \) der vorhergehenden Schicht auf das Neuron \( j \) der betrachteten Schicht. Der Vektor des Netzinputs kann nun über eine Multiplikation der Gewichtsmatrix \( W \) mit dem Inputvektor \( x \) ermittelt werden:

\[
\text{net} = W \cdot x = \begin{pmatrix} w_{11} & w_{12} & \cdots & w_{1N} \\ w_{21} & w_{22} & \cdots & w_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ w_{M1} & w_{M2} & \cdots & w_{MN} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{pmatrix} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{pmatrix}
\]

Im vorliegenden Fall eines einfachen KNN wird als Transferfunktion die Identität verwendet, d.h.:

\[
a = \text{net}
\]

Schließlich wird die Aktivität \( a \) mit einer Schwelle \( \lambda \) verglichen und, wenn die Aktivität überkritisch \( (a > \lambda) \) ist, der Output \( o \) auf 1 gesetzt, ansonsten bleibt er 0:

\[
o = \begin{cases} 1 & \text{falls } a > \lambda \\ 0 & \text{falls } a \leq \lambda \end{cases}
\]

Diese binäre Schwellenwertfunktion kommt der Funktionsweise des natürlichen Vorbildes sehr nahe.

Analogie: Im biologischen Neuron werden aktivierende (exzitatorische) und hemmende (inhibitorische) Signale aufsummiert. Überschreiten diese aufsummierten Signale einen Schwellenwert, so löst das Neuron Aktionspotenziale aus, d.h., es "feuert". Die Auslösung eines Aktionspotenzials (am Axonhügel) erfolgt nach dem Alles-oder-Nichts-Prinzip, was bedeutet, dass nur zwei Zustände für das Neuron möglich sind: es "feuert", oder es "feuert nicht" (z. B. [6]).

Obwohl ein einzelnes künstliches Neuron eine relativ einfache Verarbeitungseinheit darstellt, führt die Verknüpfung solcher Prozessorelemente zu höchst komplexen und leistungsfähigen Strukturen, die sich in ihrer Funktionsweise den hochgradig vernetzten natürlichen Neuronen im menschlichen Gehirn annähern.


5 Funktionsweise künstlicher neuronaler Netzwerke

5.1 Arbeitsweise eines einlagenigen feed-forward-Netzes

An einem einfachen Beispiel wird nun die Arbeitsweise eines KNN erläutert. Ein einlagiges feed-forward-Netz aus zwei Eingängen \( (x_1 \) und \( x_2 \)) soll Folgendes realisieren: Es sollen sich zwei Ausgänge ergeben, wobei der erste Ausgang \( (o_1) \) eine logische UND-Verknüpfung (AND) und der zweite Ausgang \( (o_2) \) eine logische NICHT-ODER-Verknüpfung (NOR) darstellt. Die folgende Wahrheitstabelle soll diesen Sachverhalt darstellen:

<table>
<thead>
<tr>
<th>( x_1 )</th>
<th>( x_2 )</th>
<th>( o_1 )</th>
<th>( o_2 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Da zwei Ausgänge benötigt werden, muss das KNN aus zwei Neuronen bestehen. Wir wollen binäre Neuronen verwenden, die als Output nur 0 oder 1 ausgeben können. Der Schwellenwert \( \lambda \) soll 1 sein. Nun müssen wir einen biologisch inspirierten "Trick" anwenden, der weiter unten begründet wird. Wie in der Natur können auch in KNN sog. "Schrittmacherneuronen" verwendet werden; dies sind Neuronen, die im Grundzustand aktiv sind. Man bezeichnet ein solches Neuron, das permanent 1 ausgibt, als BIAS. Im vorliegenden Fall wird nur ein BIAS benötigt, der als normaler Input fungiert. Das in Abbildung 2 dargestellte Netz realisiert die oben geforderte logische Verknüpfung. Selbstverständlich gibt es viele äquivalente Realisierungsmöglichkeiten.

Betrachten wir nun die möglichen Fälle nach obiger Wahrheitstabelle für das KNN in Abbildung 2:

1. \( x = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \Rightarrow \text{net} = W \cdot x \)
Abb. 2: Ein einfaches KNN, welches die Aufgabe des Beispiels in Abschnitt 5.1 erfüllt. Das erste Neuron liefert als Output $o_1$ die logische UND-Verknüpfung (AND) der beiden Eingänge, während das zweite Neuron mit $o_2$ als Output die logische NICHT-ODER-Verknüpfung (NOR) liefert.

$\begin{bmatrix} 0.7 & 0.7 & 0 \\ -0.8 & -0.8 & 1.3 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$

$\Rightarrow o = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$

2. $x = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \Rightarrow net = W \cdot x$

$= \begin{bmatrix} 0.7 & 0.7 & 0 \\ -0.8 & -0.8 & 1.3 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0.5 \\ -0.3 \end{bmatrix}$

$\Rightarrow o = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$

3. $x = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \Rightarrow net = W \cdot x$

$= \begin{bmatrix} 0.7 & 0.7 & 0 \\ -0.8 & -0.8 & 1.3 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0.7 \\ 0.5 \end{bmatrix}$

$\Rightarrow o = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$

Das Netzwerk tut also genau das, was wir erwartet haben. Generell sind KNN in der Lage, aus Inputvektoren gewünschte Outputvektoren zu generieren. Allerdings gibt es dabei ein Problem: Wie findet man die Gewichte $w_i$? Nur in Ausnahmefällen, wie in unserem sehr einfachen Beispiel, kann man sich die Gewichtsmatrix durch Kombinieren herleiten. Wenn man beispielsweise Bildverarbeitung betreibt möchte, also eine Analyse komplexer Muster, ist das Finden der Gewichtsmatrix nicht trivial. Hier kommt die große Stärke der KNN zum Tragen: sie können lernen.

5.2 Das künstliche neuronale Netz lernt

"Lernen eines Systems besteht darin, dass es entsprechend früheren Erfolgen oder Misserfolgen (Erfahrungen) das interne Modell der Außenwelt verbessert". [3]


Wir unterscheiden, insbesondere im Bereich der KNN, drei wesentliche Arten von Lernen:

1. Überwachtes Lernen: Beim überwachten Lernen wird durch Vergleich von Soll- und Ist-Output für die zu trainierenden Eingabemuster auf die vorzunehmenden Änderungen der Gewichte geschlossen. Lernregeln sind beispielsweise „Delta-Regel“ (s. Arbeitsmaterial 2 in der Online-Ergänzung) oder „Backpropagation“.

2. Bewertendes (bestärkendes) Lernen: Es wird dem KNN nur die Qualität des Ergebnisses mitgeteilt (gut, weniger gut, schlecht, ...), ohne den Fehler exakt zu beschreiben.

3. Unüberwachtes Lernen: Das unüberwachte Lernen erfolgt ausschließlich

Abb. 4: Auszug aus der Online-Ergänzung: Beispiel für ein KNN als Autoassoziativspeicher, dessen Zweck es ist, gestörte oder unvollständige Daten korrekt zu erkennen. Das stark verrauschte Muster der Ziffer 3 wurde einem (kurz) trainierten Perzepton und einem (lang) trainierten Adaline als Input gegeben. Während Adaline das Muster korrekt rekonstruiert, finden sich in der Rekonstruktion des Perzepton einigen gravierenden Fehler.

Die effizienteste Lernmethode für ein KNN – aber auch die aufwändigste für den Trainer – ist das überwachte Lernen. Der hohe Aufwand liegt in der Tatsache begründet, dass für alle zu lernenden Muster die exakten Soll-Ergebnisse bekannt sein müssen. Hier reicht es nicht aus, das Ergebnis als gut oder schlecht zu bewerten, wie dies beim bewertenden Lernen der Fall ist.


So lange, bis alle Muster richtig erkannt werden:

1) Legen ein Muster an das Netz an.
2a) Wenn der tatsächliche Output (Ist-Output) o dem gewünschten Output (Soll-Output) z entspricht, gehe zu Schritt 1, ansonsten
2b) modifiziere die Gewichte aller falsch klassifizierenden Elemente nach der Formel:

\[ w_j \left( \text{neu} \right) = w_j \left( \text{alt} \right) + \eta \, \sigma \, (z_i - o_i) = w_j + \eta \, \sigma \, \delta \]

3) Gehe zu Schritt 1.

Hierbei ist \( \sigma \) – eine Konstante größer Null, aber deutlich kleiner als die Schwelle \( \lambda \) – die so genannte Lernrate und \( \delta = (z_i - o_i) \) der Fehler, d. h., die Abweichung von Soll- und Ist-Output. Dieses \( \delta \) (Delta) ist namensgebend für den Lernalgorithmus, der das Vorzeichen der Gewichtsänderung bestimmt. Inhaltlich bedeutet dies, dass man ein Trainingsmuster an das Netz anlegt und den Soll-Output mit dem Ist-Output vergleicht. Sind Soll- und Ist-Output identisch, so werden die Gewichte nicht modifiziert. Ist der Soll-Output 1 und der Ist-Output 0, dann ist offensichtlich die Aktivität zu niedrig. Alle Gewichte von aktiven Eingängen, also Eingängen des Wertes 1, werden um die Lernrate erhöht, wohingegen inaktive Eingänge (0) nicht verändert werden. Ist der Soll-Output 0 und der Ist-Output 1, so werden alle Gewichte aktiver Eingänge um \( \sigma \) erniedrigt, und die Aktivität wird gesenkt. Dieser Vorgang wird für alle zu lernenden Muster wiederholt, bis alle fehlerfrei erkannt werden.

Die hier dargestellte Delta-Lernregel („Perzepton-Lernregel“) kann modifiziert werden, indem man nicht den Ist-Output mit dem Soll-Output, sondern die Ist-Aktivität mit dem Soll-Output vergleicht („Adaline-Lernregel“). Das Perzepton beendet das Lernen, sobald das Muster „gerade so“ richtig erkannt wird. Das Adaline verwendet prinzipiell denselben Lernalgorithmus, berechnet aber den Fehler \( \delta = (z_i - o_i) \), d. h., die Abweichung der Ist-Aktivität \( a \) vom Soll-Output \( z \). Dabei ist es wichtig, dass die Schwelle \( \lambda \) genau zwischen den Werten für „aktiv“ und „inaktiv“ liegt. Bei 0 bzw. 1 für inaktiv bzw. aktiv muss \( \lambda = 0,5 \) sein (bei -1 bzw. 1 gilt \( \lambda = 0 \)). Damit ist sichergestellt, dass die Muster nicht nur „gerade so“ erkannt, sondern sozusagen „verinnerlicht“ werden.

Gemäß dieser Definition verändert ein Perzepton oder ein Adaline seine Ge-

Im Gegensatz zu einem Computer kann das menschliche Gehirn sehr gut generi- sieren. Wenn beispielsweise eine uns bekannte Person neuerdings eine Brille trägt oder sich einen Schnurrbart hat wachsen lassen, so ist es für uns ein Leichtes, diese Person wieder zu erkennen. Einen Com- puter hingegen, welcher ein fest vorgege- benes Programm verwendet und nicht, wie hier thematisiert, mit einem Lernpro- gramm arbeitet, stellt diese Tatsache vor große bis unlösbare Probleme.

Der Lerneffekt des KNN beruht aus- schließlich auf der Veränderung der Ge- wichtungen zwischen den einzelnen künstlichen Neuronen. Im natürlichen Netzwerk entspricht dies der Veränderung der Synapsen zwischen einzelnen Nervenzellen. Mit Hilfe der Lernregel werden die Gewichtungen so lange angepasst, bis das Netzwerk auf eine bestimmte Eingabe (z.B. das Bild eines Afelp) die gewünschte Ausgabe generiert („Dies ist ein Afelp“).


6 Programmierung eines einschichtigen feed-forward-Netzes


7 Bemerkungen zum Unterricht

Wichtig für jede Art von Programmieren ist es, immer auf einem Blatt Papier da- mit zu beginnen. Das Programm sollte zunächst in Form von Befehlszusätzen ent- worf en werden, wie: „Wenn (if) … der Fall ist, dann (then) …, Befehl beendet (end)“ oder „Solange (while) … noch nicht er- reicht wurde, wiederhole Befehle …, Be- fehl beendet (end)“. So kann man sich darüber klar werden, welche Befehle man benötigt. Die Abfolge der Befehle kann gedanklich überprüft und schließlich in die Skriptsprache übersetzt werden. Diese Übersetzung ist mit etwas Übung relativ einfach, da für das gesamte Programm le- diglich vier „Vokabeln“ (function, if, while und end) benötigt werden (abzüglich der vernachlässigbaren Kontrollstrukturen).


Als nächstes sollten die Lernenden eine exakte Beschreibung der mathemati- schen Abläufe in einem künstlichen Neu- ren entwickeln und aus solchen Neurono- nen ein kleines KNN erstellen. Beispiele für KNN könnten die mathematisch- logischen Schaltungen UND (AND), ODER (OR), NICHT-ODER (NOR), NICHT-UND (NAND) sein [14]. An solchen Schaltungen können die in einem einlagigen Perzep- tron ablaufenden Prozesse mit zwei bis drei Eingangsneuronen und einem Aus- gangsnuron simuliert werden (s. Arbeits- material 2 in der Online-Ergänzung).


Das künstliche Neuron

1. Aufbau eines künstlichen Neurons


Im Gegensatz zum natürlichen Neuron handelt es sich bei einem künstlichen Neuron also nicht um eine materielle Einheit. Vielmehr kann ein künstliches Neuron als Recheneinheit eines Computerprogramms angesehen werden, das wiederum als Teil eines KNN mit anderen Recheneinheiten verbunden ist. Ein KNN ist demnach ein Gedankenmodell, das ein natürliches Nervensystem in vereinfachender Weise repräsentiert. Indem man dessen Funktionsweise am Rechner simuliert, versucht man zum Beispiel auch das menschliche Gehirn besser zu verstehen.

2. Signalweiterleitung in einem künstlichen Neuron
Im künstlichen neuronalen Netzwerk werden Signale in Form von Zahlenwerten weitergeleitet. Für eine solche Weiterleitung wird zunächst eine Eingabe dargeboten. Man wählt hierzu die Variablen \( x_1, x_2, x_3, \ldots \), \( x_n \). Die EingabevARIABLEN repräsentieren Signale von anderen künstlichen Neuronen und können die Zahlenwerte „1“ (es gibt ein Signal) oder „0“ (es gibt kein Signal) annehmen. Nach Multiplikation dieser Eingabeveriablen mit der jeweils zugewiesenen Gewichtung \( w_{1i}, w_{2i}, w_{3i}, \ldots, w_{ni} \) ergibt sich eine entsprechende gewichtete Eingabe, zum Beispiel \( x_1 \cdot w_{1i} \) oder \( x_2 \cdot w_{2i} \), in das Neuron \( i \).

Die weitere Verarbeitung der gewichteten Eingaben erfolgt in zwei Schritten. Zunächst werden deren Einzelwerte durch die Inputfunktion \( \Sigma \) (griech., Sigma) aufsummiert. Durch diese Addition ergibt sich der sog. Netzinput \( \text{net} \). Der berechnete Wert dieses Netzinputs wird durch die Transferfunktion \( \varphi \) (griech., Phi) zur Aktivität, und diese wird mit einem vorgegebenen Schwellenwert \( \lambda \) (griech., Lambda) verglichen. Der Schwellenwert hat in der Regel den Wert „1“; ist bei diesem Vergleich die Aktivität größer als \( \lambda \), wird das künstliche Neuron \( i \) durch die Outputfunktion „aktiviert“ und gibt als Output \( o_i \) den Wert „1“ aus. Ist sie kleiner (oder gleich), bleibt das künstliche Neuron \( i \) „inaktiv“ und gibt den Wert „0“ aus.

Ein Ausgabesignal kann wiederum als Eingabesignal für ein folgendes künstliches Neuron dienen, in welchem die Informationsaufnahme, -verarbeitung und -weiterleitung in der zuvor beschriebenen Weise erfolgt.

Aufgaben:
- Beschreiben Sie die Abbildung des künstlichen Neurons. Verwenden Sie hierzu die im Text durch Fettdruck hervorgehobenen Begriffe (→ Kästchen). Ergänzen Sie außerdem die fehlenden Variablen (→ Kreise).
- Vergleichen Sie tabellarisch die Signalweiterleitung in einem natürlichen und in einem künstlichen Neuron (Lösung siehe Tab. 1 im Beitrag).
Natürliches Neuron:

Komponenten des natürlichen Neurons:
- Synaptische Endknöpfchen
- Synapsen (an Dendriten)
- Zellkörper
- Axonhügel
- Axon

Aufgaben:
- Eingabe (Input)
- Verarbeitung
- Ausgabe (Output)

Komponenten des künstlichen Neurons:
- Eingabevariablen
- Gewichte
- Inputfunktion
- Transferfunktion
- Outputfunktion

Künstliches Neuron:

\[ \sum \frac{X_i \cdot w_{ij}}{\text{Netzinput}} \]

Funktion bei der Signalweiterleitung:
- Input von Eingabevariablen (Wert „1“ oder „0“)
- Multiplikation der Eingaben mit Gewichten
- Addition der gewichteten Eingaben
- Berechnung der Aktivität
- Vergleich von Aktivität und Schwellenwert und Output von Wert „1“ oder „0“
Wissen eigenständig erarbeiten wollen, die Internet-Tutorials zur selbstständigen Erarbeitung nutzen. In dieser Phase sollen sich die Lernenden mit dem Programm Octave und der Möglichkeit, einfache mathematische Formeln damit zu berechnen, vertraut machen.


Literatur
(19.05.2014)

Anschrift der Verfasser
Finn Gerrit Hüske, Dr. Ingelburg Heil und Prof. Dr. Johannes Bohrmann, RWTH Aachen, Insti tut für Biologie II, Worringerweg 3, 52056 Aachen, E-Mail: bohrmann@bio2.rwth-aachen.de und Prof. Dr. Werner Baumgartner, Universität Linz, Institut für Medizin- und Biomechatronik, Altenbergerstraße 69, A-4040 Linz, E-Mail: werner.baumgartner@jku.at

PCR und Sequenzierung in der humangenetischen Diagnostik
Ein reales Fallbeispiel zur Mukoviszidose
K. Eggermann, J. Deerberg, S. Mehta und T. Eggermann

In der humangenetischen Diagnostik gehören Experimente wie die PCR und die Sequenzi erung nach Sanger zum Laboralltag. Originaldaten zu einem realen Fallbeispiel stellen diese Methoden in einen anwendungsbezogenen Kontext.

Stichwörter: Humangenetik, PCR, Sequenzierung, ARMS-Test, Mukoviszidose, Cystische Fibrose, CFTR-Gen

1 Einleitung
Die Polymerase-Ketten-Reaktion (Polymerase Chain Reaction, PCR) und die DNA-Sequenzierung nach Sanger gehören zu den grundlegenden Methoden in der Diagnostik von Erkrankungen. Dabei ist nicht nur das Verständnis der theoretischen Hintergründe von Bedeutung, sondern auch die Möglichkeit, die Techniken und Experimente an die entsprechende Fragestellung anzupassen. Insbesondere gilt dies für die PCR, die durch enorme Verbesserung der Laborchemie (hitzestabile DNA-Polymerasen, z.B. Taq-DNA-Polymerase, Pfu-DNA-Polymerase; Puffersysteme) und des bioinformatischen Primerdesigns wesentlich zur Optimierung der Laborabläufe beigetragen hat.

In der humangenetischen Diagnostik sind die genannten Methoden aber nicht einzeln zu betrachten, sondern sie sind Teil einer Kette von Experiments und der Datenauswertung, die nur im Zusammenhang mit einem konkreten Fall sinnvolle Ergebnisse liefern (Abb. 1) und damit den Anwendungsbezug molekulargenetischer