LEHRSTUHL FÜR
ALLG. BWL UND WIRTSCHAFTSINFORMATIK
UNIV.-PROF. DR. HERBERT KARGL

Strecker, Stefan

Künstliche Neuronale Netze – Aufbau und Funktionsweise

ARBEITSPAPIERE WI
Nr. 10/1997

Schriftleitung:
Dr. rer. pol. Axel C. Schwickert

Information

Reihe: Arbeitspapiere WI

Herausgeber: Univ.-Prof. Dr. Axel C. Schwickert
Professur für BWL und Wirtschaftsinformatik
Justus-Liebig-Universität Gießen
Fachbereich Wirtschaftswissenschaften
Licher Straße 70
D – 35394 Gießen
Telefon (0 64 1) 99-22611
Telefax (0 64 1) 99-22619
eMail: Axel.Schwickert@wirtschaft.uni-giessen.de
http://wi.uni-giessen.de

Bis Ende des Jahres 2000 lag die Herausgeberschaft bei:

Lehrstuhl für Allg. BWL und Wirtschaftsinformatik
Johannes Gutenberg-Universität Mainz
Fachbereich Rechts- und Wirtschaftswissenschaften
Welderweg 9
D - 55099 Mainz

Ziele: Die Arbeitspapiere dieser Reihe sollen konsistente Überblicke zu den Grundlagen der Wirtschaftsinformatik geben und sich mit speziellen Themenbereichen tiefergehend befassen. Ziel ist die verständliche Vermittlung theoretischer Grundlagen und deren Transfer in praxisorientiertes Wissen.

Zielgruppen: Als Zielgruppen sehen wir Forschende, Lehrende und Lernende in der Disziplin Wirtschaftsinformatik sowie das IuK-Management und Praktiker in Unternehmen.

Quellen: Die Arbeitspapiere entstanden aus Forschungsarbeiten, Diplom-, Studien- und Projektarbeiten sowie Begleitmaterialien zu Lehr- und Vortragsveranstaltungen des Lehrstuhls für Allg. Betriebswirtschaftslehre und Wirtschaftsinformatik Univ. Prof. Dr. Herbert Kargl an der Johannes Gutenberg-Universität Mainz.

Hinweise: Wir nehmen Ihre Anregungen und Kritik zu den Arbeitspapieren aufmerksam zur Kenntnis und werden uns auf Wunsch mit Ihnen in Verbindung setzen.
Falls Sie selbst ein Arbeitspapier in der Reihe veröffentlichen möchten, nehmen Sie bitte mit dem Herausgeber (Gießen) unter obiger Adresse Kontakt auf.
Informationen über die bisher erschienenen Arbeitspapiere dieser Reihe und deren Bezug erhalten Sie auf dem Schlußblatt eines jeden Arbeitspapiers und auf der Web Site des Lehrstuhls unter der Adresse http://wi.uni-giessen.de

Arbeitspapiere WI Nr. 10/1997

Autor: Strecker, Stefan

Titel: Künstliche Neuronale Netze – Aufbau und Funktionsweise

Zitation: Strecker, Stefan: Künstliche Neuronale Netze – Aufbau und Funktionsweise, in: Arbeitspapiere WI, Nr. 10/1997, Hrsg.: Lehrstuhl für Allg. BWL und Wirtschaftsinformatik, Johannes Gutenberg-Universität: Mainz 1997.

Kurzfassung: Herkömmliche Computer erledigen exakt berechenbare, routinehafte Aufgaben schneller und zuverlässiger als der Mensch. Einige typisch menschliche Fähigkeiten (z. B. die Gesichtserkennung) stellen die konventionelle Informationsverarbeitung dagegen vor große Schwierigkeiten. Herkömmliche Algorithmen scheitern, sobald die vorausgesetzte Bildqualität nicht gegeben ist. Der Mensch erkennt dagegen Gesichter problemlos auch unter erschwerten Bedingungen (Dunkelheit, Nebel). Es liegt also nahe zu fragen, nach welchen Prinzipien das menschliche Gehirn organisiert ist und auf welche Weise es die sensorischen Informationen der Sinne verarbeitet. Vor diesem Hintergrund ist die Entwicklung Künstlicher Neuronaler Netze (KNN) zu sehen: KNN imitieren die Organisations- und Verarbeitungsprinzipien des menschlichen Gehirns. Aus betriebswirtschaftlicher Sicht stellen KNN neue Problemlösungsverfahren aus dem Forschungsgebiet der Künstlichen Intelligenz dar, die das ökonomische Modellierungsinstrumentarium erweitern und sich besonders für komplexe, nicht-konservative Aufgabenstellungen eignen. Gegenüber traditionellen Verfahren aus der Statistik und dem Operations Research zeichnen sich KNN durch Lernfähigkeit, Fehlertoleranz, Robustheit und Generalisierungsfähigkeit aus. Betriebliche Anwendungsfelder finden sich insbesondere in den Bereichen Prüfung und Beurteilung, Prognose, Klassenbildung und Optimierung. Der vorliegende Beitrag soll praxisorientiert einen Überblick über den Aufbau und die Funktionsweise von KNN geben und damit einen Einstieg in die Thematik ermöglichen. Ausgehend von den biologischen Grundlagen werden die statischen und dynamischen Kernkomponenten von KNN definiert und die prinzipiellen Informationsverarbeitungsprozesse erläutert. Ein Überblick über die typischen Eigenschaften von KNN bildet den Abschluß des Beitrags.

Schlüsselwörter: Künstliche Neuronale Netze, Konnektionismus, Künstliche Intelligenz

Inhaltsverzeichnis

1 Einleitung

Herkömmliche Computer erledigen exakt berechenbare und routinehaft-fixierte Aufgaben schneller und zuverlässiger als der Mensch. Einige typisch menschliche Fähigkeiten (z. B. die Gesichts- oder Stimmerkennung) stellen die konventionelle Informationsverarbeitung dagegen vor große Schwierigkeiten. Diese Aufgaben zeichnen sich durch ein hohes Maß an Undeterminiertheit, Vagheit und Unschärfe aus. Die Gesichtserkennung auf der Basis klassischer Algorithmen scheitert zum Beispiel, sobald die vorausgesetzte Bildqualität nicht gegeben ist. Der Mensch dagegen erkennt Gesichter problemlos auch unter erschwerten Bedingungen (Dunkelheit, Nebel). Es liegt also nahe, zu fragen, nach welchen Prinzipien das menschliche Gehirn organisiert ist und auf welche Weise es die sensorischen Informationen der Sinne verarbeitet. Vor diesem Hintergrund ist die Entwicklung Künstlicher Neuronaler Netze (KNN) zu sehen: KNN imitieren die Organisations- und Verarbeitungsprinzipien des menschlichen Gehirns.

Die Ausgangspunkte der Erforschung menschlicher Denk- und Wahrnehmungsprozesse sind vielfältig. Neurophysiologische Untersuchungen und psychologische Experimente führten zusammen mit Erkenntnissen der Neurologie und Neurobiologie zu immer exakteren Vorstellungen über die Arbeitsweise des menschlichen Gehirns. Letztlich mündeten die biologisch motivierten Theorien in mathematisch formulierte Modelle des Gehirns. Die Simulation der Modelle führte über die Schnittstelle „Computer“ zur Informatik und dort im Teilgebiet der Künstlichen Intelligenz (KI) zur Entstehung der interdisziplinären Forschungsrichtung des Konnektionismus, dessen Forschungsgegenstand und Instrument Künstliche Neuronale Netze sind.

Das biologische neuronale Netz „Gehirn“ besteht aus Milliarden von Nervenzellen (Neuronen), die miteinander verbunden die Sinneswahrnehmungen verarbeiten. Ein Künstliches Neuronales Netz wird definiert als

„[...] a system composed of many simple processing elements operating in parallel whose function is determined by network structure, connection strengths, and the processing performed at computing elements or nodes.“[1]

Hauptanwendungsgebiet der KNN ist die Mustererkennung. KNN können Muster in Daten finden, wesentliche Merkmale extrahieren und funktionale Zusammenhänge zwischen den Mustern approximieren. Ein typisches Mustererkennungsproblem ist die Klassifikation von Daten. Die Merkmale der Daten definieren einen Musterraum, den ein Klassifikator partitioniert. Der Klassifikator approximiert die funktionalen Zusammenhänge, die die Entscheidungsgrenzen in einem Musterraum definieren. In einfachen Fällen lassen sich Klassen (Teilräume) durch lineare Entscheidungsgrenzen separieren. Praxisnahe Anwendungen mit komplexen n-dimensionalen Musterräumen zeichnen sich dagegen typischerweise durch nicht-lineare Entscheidungsgrenzen aus, deren Approximation nicht trivial ist (vgl. Abbildung 1).[2]

1 DARPA Neural Network Study, Fairfax, VA: AFCEA International Press 1988, S. 60.

2 Vgl. Scherer, A.: Neuronale Netze - Grundlagen und Anwendungen, Braunschweig et. al.: Vieweg 1997, S. 22 f.

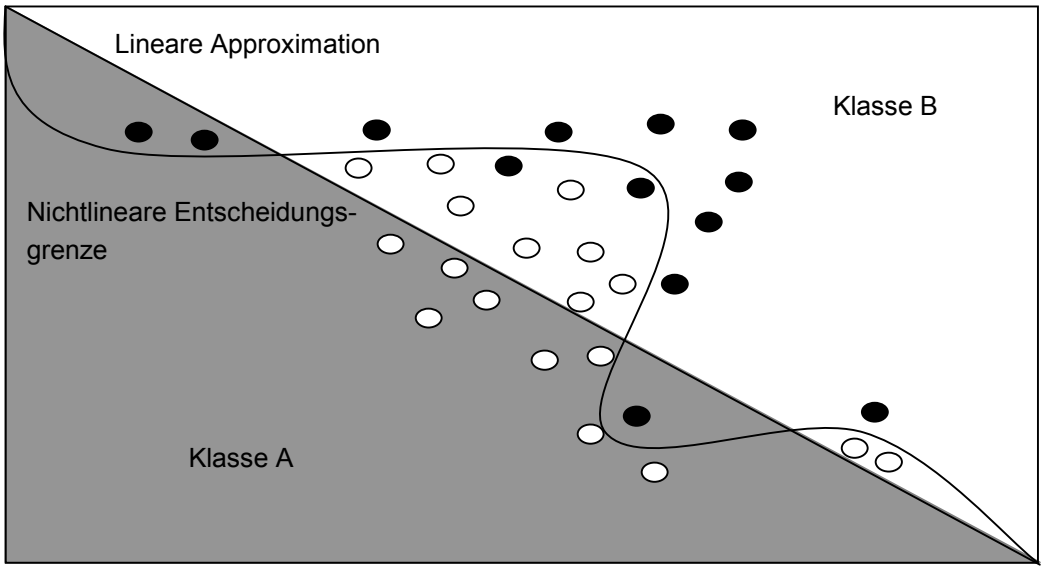

Abb. 1: Ein nichtlineares Klassifikationsproblem mit 2 Klassen

Weitere typische Aufgabenbereiche für KNN sind die Prognose, Klassenbildung, Funktionsapproximation und Optimierung. Ein traditionelles Anwendungsfeld sind die Ingenieurswissenschaften, in denen KNN z. B. zur Steuerung von Robotern eingesetzt werden. Zunehmend finden KNN auch Anwendung in betriebswirtschaftlichen Problemstellungen. In der betriebswirtschaftlichen Forschung werden KNN als alternative Problemlösungsverfahren zu Methoden aus der Statistik und dem Operations Research betrachtet, die das ökonomische Modellierungsinstrumentarium erweitern. Typischerweise messen empirisch-vergleichende Studien daher die Qualität der KNN an traditionellen Verfahren wie etwa der Multivariaten Diskriminanzanalyse (MDA) oder der Clusteranalyse.

Für den praktischen Einsatz in Unternehmen gibt es bereits zahlreiche Beispiele. Einige exemplarische betriebswirtschaftliche Anwendungen liegen in folgenden Bereichen vor:[3]

- **Prüfung und Beurteilung** (Musterklassifikation): Kreditwürdigkeitsprüfung, Insolvenzprüfung, Bilderkennung
- **Klassenbildung** (Clustering): Marktsegmentierung, Data Mining
- **Prognose** (Prediction): Kursprognosen, Absatzprognosen, Kostenprognosen
- **Optimierung**: Transportoptimierung (Travelling-Salesman-Problem), Reihenfolgeplanung

3 Vgl. Corsten, H.; May, C.: Anwendungsfelder Neuronaler Netze und ihre Umsetzung, in: Neuronale Netze in der Betriebswirtschaft – Anwendungen in Prognose, Klassifikation und Optimierung, Hrsg.: Corsten, H.; May, C., Wiesbaden: Gabler 1996, S. 3 und Scherer, A.: Neuronale Netze - Grundlagen und Anwendungen, a. a. O., S. 13.

2 Biologische Vorbilder

Grundbausteine der menschlichen Intelligenz bilden Nervenzellen (Neuronen) in der Hirnrinde (Neokortex).[4] Die Neuronen bestehen schematisch aus Dendriten (Zelleingängen), Zellkörper (Soma) mit Zellkern, Zellmembran, und einer Nervenfaser (Axon) mit Synapsen (Zellausgängen).

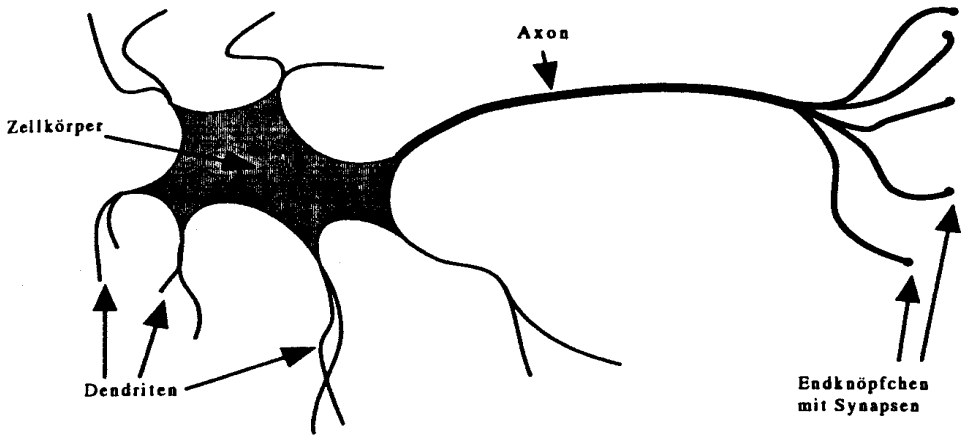

Abb. 2: Biologisches Neuron[5]

Die Dendriten dienen als Eingänge des Neurons. Synapsen sind Ausläufer des Axons und dienen der Signalübertragung (vgl. Abbildung 2). Im Durchschnitt ist ein Neuron über die Dendriten mit Synapsen von 14.000 anderen Neuronen verbunden. Maximal kann ein Neuron ca. 200.000 Ein- und ebensoviele Ausgangsverbindungen besitzen.[6] Das Axon transportiert ein elektrisches Signal, welches durch Ladungsunterschiede des elektrischen Potentials zwischen Zellinnerem und -umgebung (sog. Membranpotential) entsteht. Ein Neuron "feuert", d. h., es ist aktiv, wenn die Ladungsunterschiede ein Aktionspotential auslösen. Die Aktivität eines Neurons folgt dabei dem „Alles-oder-nichts"-Prinzip. Ein Neuron feuert oder es feuert nicht; Zwischenzustände existieren nicht. Der Informationsgehalt in einem Neuron drückt sich deshalb nicht in der Tatsache aus, daß ein Neuron aktiv ist, sondern in der Frequenz, mit der ein Neuron seinen Zustand verändert. Die Frequenz beträgt üblicherweise 250 Takte pro Sekunde (maximal bis zu 1000). Das elektrische Signal breitet sich dann ausgehend vom Zellmembran entlang dem Axon aus und endet in den Synapsen. Die Synapsen sind mit Dendriten anderer Neuronen verbunden , wobei keine direkte physikalische Verbindung, sondern ein mit Flüssigkeit gefüllter Zwischenraum zwischen Synapsen und Dendriten besteht. An

4 Vgl. Schumann, M.; Lohrbach, T. Retzko, R.: Einführung in Aufbau und Arbeitsweise Künstlicher Neuronaler Netze, Georg-August-Universität Göttingen, Abtlg. Wirtschaftsinformatik II, Arbeitspapier Nr. 1, Hrsg.: Schumann, M., Dezember 1991, S. 1.

5 Entnommen aus Krause, C.: Kreditwürdigkeitsprüfung mit Neuronalen Netzen, Düsseldorf: IDW 1993, S. 37.

6 Vgl. Schöneburg, E.; Hansen, N.; Gawelczyk, A.: Neuronale Netze, Haar bei München: Markt-u.-Technik 1990, S. 37.

den Synapsen führt die elektrische Ladung zu einer biochemischen Reaktion, die sogenannte Neurotransmitter freisetzt. Diese Trägersubstanzen werden von den Dendriten des Empfängerneurons aufgenommen und führen ihrerseits zu einer Reaktion. Ein Neurotransmitter wirkt dabei hemmend (inhibitorisch) oder verstärkend (exzitatorisch). Insgesamt sind 30 unterschiedliche Trägersubstanzen in menschlichen Synapsen bekannt. Nach dem heutigen Kenntnisstand sind Anpassungen der Synapsen für die Lernvorgänge des menschlichen Gehirns verantwortlich. Bei der Adaption verändert sich die Stärke der Verbindung zwischen Neuronen, indem Anzahl und Größe der Synapsen sowie weitere Aufbauparameter modifiziert werden.[7]

Im Neokortex des menschlichen Gehirns befinden sich ca. 10-100 Milliarden Neuronen. Bei durchschnittlich 14.000 Eingangs- und Ausgangsverbindungen ergeben sich rund 100-1.000 Billionen Verbindungen zwischen Neuronen, wobei nicht alle Neuronen miteinander verbunden sind. Biologische oder natürliche neuronale Netze entstehen durch räumliche Zusammenlagerung und Verbindung von Neuronen (vgl. Abbildung 3).[9] Schematisch kann der Aufbau des Gehirns als hierarchisch beschrieben werden. Bestimmte Funktionsbereiche sind in Mikrosäulen zusammengefaßt. Die Mikrosäulen bilden wiederum auf der nächsthöheren Ebene Rindenfelder, von denen es ca. 80 Stück, über den gesamten Neokortex verteilt, gibt. Mikrosäulen und Rindenfelder bilden verschiedene Aggregationen von biologischen neuronalen Netzen.

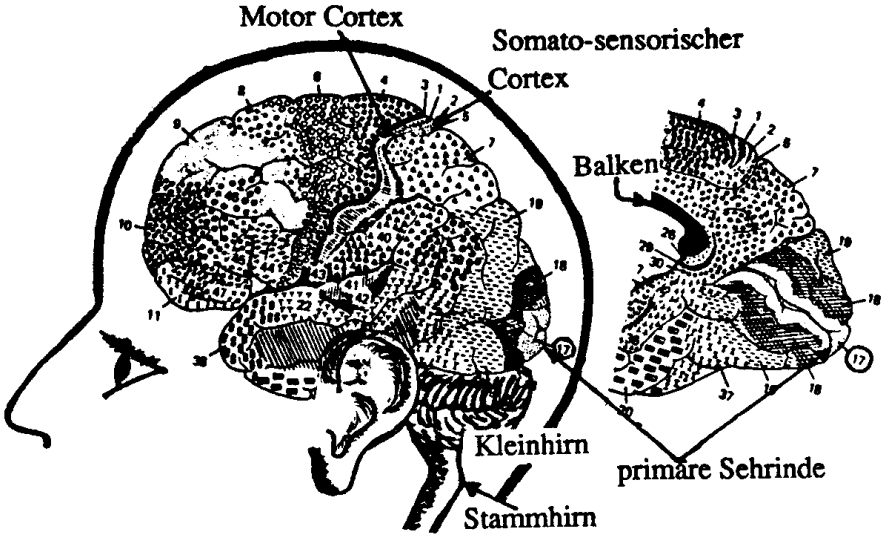

Abb. 3: Bestandteile der Neokortex[8]

7 Vgl. Schöneburg, E.; Hansen, N.; Gawelczyk, A.: Neuronale Netze, a. a. O., S. 40.

8 Entnommen aus Brause, R.: Neuronale Netze - eine Einführung in die Neuroinformatik, Stuttgart: Teubner 1991, S. 13.

9 Vgl. Schumann, M.; Lohrbach, T.; Bährs, P.: Versuche zur Kreditwürdigkeitsprüfung mit Künstlichen Neuronalen Netzen, Georg-August-Universität Göttingen, Abtlg. Wirtschaftsinformatik II, Arbeitspapier Nr. 2, Hrsg.: Schumann, M., Januar 1992, S. 3.

Die Leistungsfähigkeit des „Neurocomputers“ menschliches Gehirn ist beachtenswert.[10] Verglichen mit heutigen Computern ist die Verarbeitungsgeschwindigkeit um ein Vielfaches (ca. 10.000-fach) langsamer, dennoch sind relativ kurze Antwortzeiten bei Abruf gespeicherten Wissens möglich. Man führt die Leistungsfähigkeit auf die massiv-parallele Verarbeitungsorganisation des Gehirns zurück. Die komplexen Aufgaben des Gehirns müssen sich in Anbetracht der Antwortzeiten bei geringerer Verarbeitungsgeschwindigkeit mit relativ wenigen (ca. 100) Operationen durchführen lassen. Dies ist nur möglich, wenn das Wissen in parallelem und gleichzeitigem Zugriff für viele Operationen zur Verfügung steht. Daraus schließt man auf die dezentrale Wissensspeicherung und Wissensverarbeitung des Gehirns.

Die neurophysiologischen Erkenntnissen über den Aufbau und die Funktionsweise des Gehirns bilden die Grundlage für die Entwicklung mathematisch formulierter Neuronenmodelle, die in Anlehnung an das biologische Vorbild als „Künstliche Neuronale Netze“ bezeichnet werden. Im folgenden Abschnitt demonstriert ein Beispiel die grundlegenden Komponenten und die prinzipielle Funktionsweise eines einfachen KNN, um die Unterschiede zwischen biologischen und künstlichen neuronalen Netzen zu verdeutlichen.

3 Aufbau und Funktionsweise von KNN

3.1 Ein einführendes Beispiel: Die *inclusive or*-Operation

Die boolesche *inklusive oder*-Operation (*inklusive oder*-Operation oder verkürzt *oder*-Operation) ist ein typisches Beispiel für eine Klassifikation mit KNN.[11] Die Beispielaufgabe wird mit Hilfe eines „Perceptrons“ gelöst, ein KNN, das 1958 von Frank Rosenblatt zur Simulation der Netzhaut im menschlichen Auge entwickelt wurde.[12] Das Perzeptron gilt als das klassische „Ur-Neuronenmodell“.[13]

Gegeben sind zwei binäre Variablen X_1 und X_2. Die *oder*-Operation verknüpft die beiden Operanden gemäß Tabelle 1. Deutlich wird das Klassifikationsproblem in einer graphischen Darstellung. Die Ein- und Ausgabewerte lassen sich als Koordinaten in einem 2D-Raum abbilden und je einer Klasse zuordnen (vgl. Abbildung 4).

10 Vgl. Ritter, H.; Martinez, T.; Schulten, K.: Neuronale Netze - Eine Einführung in die Neuroinformatik selbstorganisierender Netzwerke, 2., erw. Aufl., Reading, MA et. al.: Addision-Wesley 1991, S. 5.

11 Das Beispiel wird aufgrund seiner Einfachheit als Spielzeugbeispiel („toy problem“) kritisiert und entspricht nicht der Komplexität praxisrelevanter Problemstellungen. Vgl. Freeman, J. A.; Skapura, D. M.: Neural Networks - Algorithms, Applications and Programming Techniques, 2., korr. Aufl., Reading, MA: Addison-Wesley 1992, S. 29.

12 Photo-Perzeptronen sind Bestandteile der Netzhaut (Retina) im menschlichen Auge und verarbeiten die Stimuli der optischen Sinneswahrnehmung. Vgl. Freeman, J. A.; Skapura, D. M.: Neural Networks - Algorithms, Applications and Programming Techniques, a. a. O., S. 22.

13 Vgl. Kemke, C.: Der neuere Konnektionismus, in: Informatik Spektrum, 11/1988, S. 143.

Das *oder*-Problem ist gelöst, wenn eine Gerade gefunden ist, die die beiden Klassen Null bzw. Eins voneinander trennt. Die Operation läßt sich daher als linear separierbares Klassifikationsproblem auffassen. Die Entscheidungsgrenze kann durch eine lineare Funktion (Geradengleichung) dargestellt werden.

	Eingabe		oder-Operation	Gewünschte Ausgabe gem. *oder*-Operation
Fall	X1	X2	X1 $\vee$ X2	
1	0	0	$0 \vee 0$	0
2	0	1	$0 \vee 1$	1
3	1	0	$1 \vee 0$	1
4	1	1	$1 \vee 1$	1

Tab. 1: Die Ein- und Ausgabewerte der *oder*-Operation

Ein KNN löst diese Aufgabe, indem es zu den vier dargestellten Eingabekonstellationen die gewünschte Ausgabe lernt. Die Muster 0-0, 0-1 usw. werden also der jeweiligen Musterklasse Null bzw. Eins zugeordnet.

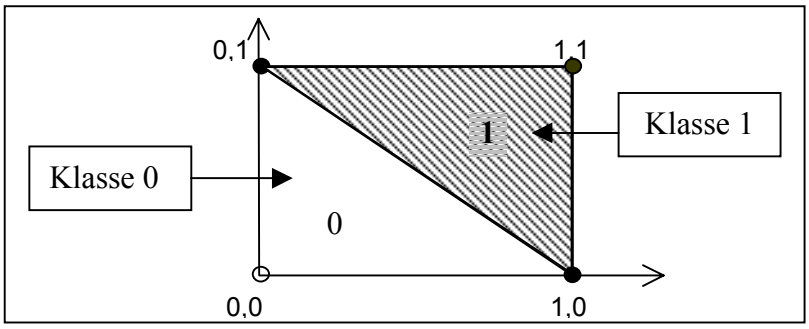

Abb. 4: Eine graphische Darstellung der booleschen *oder*-Operation

Dazu konstruiert man ein Perzeptron mit zwei Eingabeeinheiten ("processing elements") und einer Ausgabeeinheit. Die Einheiten sind in zwei, aufeinanderfolgenden Schichten angeordnet. Jede Eingabeeinheit x_1 und x_2 spiegelt eine der binären Variablen X_1 und X_2 wieder. In den Eingabeeinheiten finden keine Berechnungen statt; sie geben den Eingabewert an die Ausgabeeinheit weiter. Dazu sind sie über gewichtete Verbindungen (Gewichte w_1, w_2) mit der Ausgabeeinheit verbunden.

Die Ausgabeeinheit verbindet die beiden Eingabewerte mit den jeweiligen Verbindungsgewichten zur Netzeingabe mittels der Funktion $net = w_1 x_1 + w_2 x_2$. Der tatsächliche Ausgabewert wird durch einen Schwellenwert $\Theta = 0{,}5$ bestimmt, der dafür sorgt, daß der Ausgabewert Y, d. h. die Musterklasse, nur die Werte Null bzw. Eins annimmt. Für $net < \Theta$ ist $Y = 0$ und für $net \geq \Theta$ ist $Y = 1$. Der Schwellenwert (Threshold) ist notwendig, da in diesem Beispiel nur die beiden Klassen Null bzw. Eins zu separieren sind. Abbildung 5 veranschaulicht die Zusammenhänge graphisch.

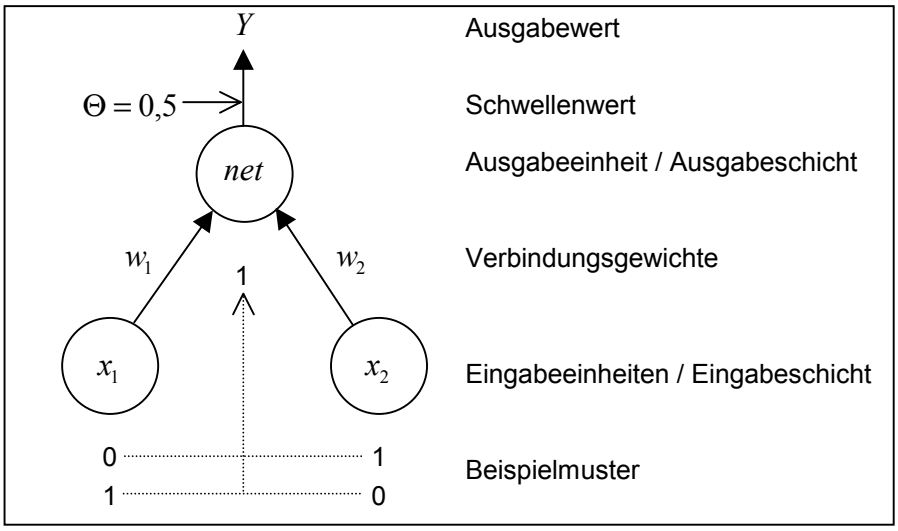

Abb. 5: Ein einfaches KNN: Das Perzeptron

Nach der Konstruktion des KNN folgt der Lernprozeß. Während des Lernens werden die Verbindungsgewichte so eingestellt, daß der Ausgabewert die gewünschte Musterklasse zuordnet. In einem iterativen Prozeß werden die vier Eingabemuster nacheinander an das Netz angelegt (vgl. Tabelle 2). Im ersten Lernschritt (Zeile 1) produziert das KNN für das Eingabemuster 0-0 die korrekte Klassifikation Null. Eine Anpassung der Verbindungsgewichte ist nicht notwendig. Für das Eingabemuster 0-1 errechnet das KNN bei der gegebenen Gewichtskonfiguration eine Netzeingabe von $net = 0 \cdot 0{,}1 + 1 \cdot 0{,}3 = 0{,}3$. Der Ausgabewert Y beträgt Null, da die Netzeingabe kleiner als der Schwellenwert ist ($net = 0{,}3 < 0{,}5$). Die korrekte Musterklasse Z (das Ergebnis der *oder*-Operation) für das Eingabemuster 0-1 ist Eins, d. h., das KNN konnte das Muster nicht korrekt klassifizieren. Der Fehler zwischen gewünschten und errechnetem Output ist $\Delta = Z - Y = 1 - 0 = 1$. Eine Adaption der Gewichte durch den Lernprozeß ist notwendig, um den Ausgabewert der korrekten Musterklasse anzunähern. Der Lernprozeß modifiziert die aktiven Verbindungsgewichte, d. h. diejenigen, die mit einem positiven Eingabewert verbunden sind. Für das Eingabemuster 0-1 wird also das Gewicht w_2 adaptiert, da die Eingabeeinheit $x_2 = 1$ ist. Die Änderung der Gewichte erfolgt nach einer Lernregel, die sich schrittweise den optimalen Werten nähert. Die Schrittweite und damit die Geschwindigkeit des Lernvorgangs steuert dabei die Lernrate δ. Mit den aktuellen Beispielwerten ergibt sich nach der Lernregel des vereinfachten Perzeptrons[14]

$$w_i(t+1) = w_i(t) + \delta \cdot \Delta \cdot X_i = 0{,}3 + 0{,}2 \cdot 1 \cdot 1 = 0{,}5$$

ein neues Verbindungsgewicht von $w_2 = 0{,}5$. Der Index t kennzeichnet die diskreten Lernzeitpunkte. Die Werte nach einem Lernschritt werden mit $t + 1$ und die Werte vor einem Lernschritt mit t indiziert. Wie bei iterativem Weiterrechnen duetlich wird, hat

14 Die vereinfachte Version wurde zuerst von Minsky und Papert vorgestellt und unterscheidet sich von dem ursprünglichen Perzeptron nach Rosenblatt. Vgl. Minsky, M.; Papert, S.: Perceptrons: Expanded Edition, 2. Aufl., Cambridge, MA: MIT Press 1988.

sich die Netzausgabe dem gewünschten Ausgabemuster angenähert. Der iterative Lernprozeß wird für alle Eingabemuster solange wiederholt, bis kein Gewicht mehr adaptiert werden muß, um die korrekte Musterklasse zuzuordnen. Im Beispiel ist dies nach vier Iterationen der Fall. Nach dem Abschluß des Lernvorgangs ist das KNN in der Lage, die Eingabemuster korrekt zu klassifizieren und die *inklusive oder*-Operation auszuführen.[15]

Lern-rate	Threshold		Initial $w_1(0)$		Initial $w_2(0)$					
$\delta = 0{,}2$	$\Theta = 0{,}5$		0.1		0.3					
				Initial					Final	
Zeile	x_1	x_2	Z	$w_1(t)$	$w_2(t)$	*net*	Y	Δ	$w_1(t+1)$	$w_2(t+1)$
1	0.00	0.00	0.00	0.10	0.30	0.00	0.00	0.00	0.10	0.30
2	0.00	1.00	1.00	0.10	0.30	0.30	0.00	1.00	0.10	0.50
3	1.00	0.00	1.00	0.10	0.50	0.10	0.00	1.00	0.30	0.50
4	1.00	1.00	1.00	0.30	0.50	0.80	1.00	0.00	0.30	0.50
5	0.00	0.00	0.00	0.30	0.50	0.00	0.00	0.00	0.30	0.50
6	0.00	1.00	1.00	0.30	0.50	0.50	0.00	1.00	0.30	0.70
7	1.00	0.00	1.00	0.30	0.70	0.30	0.00	1.00	0.50	0.70
8	1.00	1.00	1.00	0.50	0.70	1.20	1.00	0.00	0.50	0.70
9	0.00	0.00	0.00	0.50	0.70	0.00	0.00	0.00	0.50	0.70
10	0.00	1.00	1.00	0.50	0.70	0.70	1.00	0.00	0.50	0.70
11	1.00	0.00	1.00	0.50	0.70	0.50	0.00	1.00	0.70	0.70
12	1.00	1.00	1.00	0.70	0.70	1.40	1.00	0.00	0.70	0.70
13	0.00	0.00	0.00	0.70	0.70	0.00	0.00	0.00	0.70	0.70
14	0.00	1.00	1.00	0.70	0.70	0.70	1.00	0.00	0.70	0.70
15	1.00	0.00	1.00	0.70	0.70	0.70	1.00	0.00	0.70	0.70
16	1.00	1.00	1.00	0.70	0.70	1.40	1.00	0.00	0.70	0.70

Tab. 2: Der Lernprozeß im Überblick

Nachdem im vorangegangenen Beispiel die Elemente und der Ablauf eines KNN-Modells (Netzwerkmodells, Netzwerktyps) begrifflich vorgestellt wurden, soll nun eine inhaltliche Präzisierung dieser Begriffe erfolgen. Dabei konzentriert sich die vorliegende Arbeit auf die wesentlichen Kernkomponenten und die prinzipiellen Verarbeitungsabläufe.

15 Vgl. Klimasauskas, C. C.: Applying Neural Networks, in: Neural Networks in Finance and Investing, Hrsg.: Trippi, R.; Turban, E., Burr Ridge, IL et al.: Irwin 1993, S. 47 ff. und Medsker, L.; Turban, E.; Trippi, R. R.: Neural Networks Fundamentals for Financial Analysts, in: Neural Networks in Finance and Investing, a. a. O., S. 17 ff.

3.2 Kernkomponenten und grundsätzliche Verarbeitungsabläufe in KNN

Künstliche Neuronale Netze weisen Kernkomponenten oder Grundbausteine auf, die sich in allen Netzwerktypen wiederfinden.[16] Die statischen Kernkomponenten geben KNN die räumliche Gestalt bzw. Struktur:

- Verarbeitungseinheiten (processing elements)
- Verbindungen zwischen Verarbeitungseinheiten
- die Netzwerktopologie

Die dynamischen Kernkomponenten beschreiben die Informationsverarbeitung in KNN. Die grundsätzlichen Verarbeitungsabläufe (dynamischen Kernkomponenten) des Informationsverarbeitungsprozesses umfassen die Phasen:

- Lernphase
- Verarbeitungsphase

3.3 Ein Praxisbeispiel: Die Kreditwürdigkeitsprüfung

Das Zusammenspiel der Komponenten läßt sich am Beispiel der Kreditwürdigkeitsprüfung kurz verdeutlichen. Die Kreditwürdigkeitsprüfung ist der zentrale Bestandteil im Kreditgewährungsprozeß bei Kreditinstituten. Ziel ist es, Kreditanträge nach der Bonität des Antragstellers in Bonitätsklassen einzuteilen, um daran das Kreditrisiko und damit die Kreditvergabe zu beurteilen. Die Einstufung der Kreditanträge in Bonitätsklassen ist u. a. mit KNN möglich.

Die Kreditmerkmale des Antragstellers (Alter, Beruf, Einkommen, ...) werden dazu in numerischer Form als Eingabemuster kodiert. Zum Beispiel ließen sich Altersgruppen in zwei binären Eingabeeinheiten mit 0-0 für die Gruppe der 18 bis 25-jährigen Antragsteller, 0-1 für 26 bis 32-jährige Antragsteller usw. darstellen. Die Bonitätsklasse läßt sich als einzelnes, binäres Ausgabemuster mit den Ausprägungen 0 für schlechte Bonität und 1 für gute Bonität abbilden. Ein KNN für eine derartige Aufgabe könnte eine Struktur (Topologie) aufweisen, wie sie in Abbildung 6 dargestellt ist.

Das KNN besitzt eine Verarbeitungseinheit in der sog. Ausgabeschicht, die die Bonitätsklasse kodiert. In der Eingabeschicht weist das KNN soviele Einheiten auf, wie für die Kodierung der Kreditmerkmale notwendig ist. In der Darstellung sind beispielhaft als Ausschnitt die Kodierung der Altersgruppen in zwei Einheiten dargestellt. Darüber hinaus tritt in der sog. versteckten Schicht eine Anzahl von Verarbeitungseinheiten auf.

Für die Lernphase wird eine Menge von „alten“, abgeschlossenen Kreditfällen als Trainingsdatensatz benutzt. Der Datensatz enthält die Kreditmerkmale alter Kreditanträge als Eingabemuster sowie die dazugehörige, bekannte Bonität des Kunden als korrespon-

15 Vgl. Scherer, A.: Neuronale Netze - Grundlagen und Anwendungen, a. a. O., S. 45.

dierendes Ausgabemuster. Der Trainingsdatensatz enthält also Paare korrespondierender Eingabe-/Ausgabemuster.

Unterstellt man, daß sich in diesen abgewickelten Kreditanträgen auffällige Merkmalskombinationen für Antragsteller mit einer später offenbarten, guten Bonität ebenso wie Merkmalszüge für Kreditanträge illiquid gewordener Antragsteller befinden, dann soll das KNN diese typischen Merkmalskombinationen in seiner Topologie mathematisch abbilden und zukünftige Kreditanträge korrekt einer der beiden Bonitätsklassen zuordnen. Das KNN soll also eine Klassifikation der Eingabemuster (zukünftige Kreditanträge) zu einer a priori bekannten Musterklasse (Bonitätsklasse) vornehmen. Dazu werden dem KNN in der Lernphase die entsprechend kodierten Merkmale (Alter, Beruf, Einkommen, ...) abgeschlossener Kreditfälle als Eingabemuster präsentiert und jedem Eingabemuster die bekannte Bonitätsklasse des Antragstellers gegenübergestellt. Das KNN errechnet während des Lernens für jeden Datensatz solange Ausgabemuster (d. h. eine Bonitätsklasse), bis die berechnete mit der bekannten, korrekten Bonitätsklasse übereinstimmt. Hat das KNN die Zuordnung einer großen Menge abgeschlossener Kreditfälle zur jeweiligen Bonitätsklasse gelernt, unterstellt man dem KNN eine Vorhersagefähigkeit für unbekannte, zukünftige Kreditanträge, die es dann selbständig einer Bonitätsklasse zuordnet und so den Kreditgewährungsprozeß unterstützt.

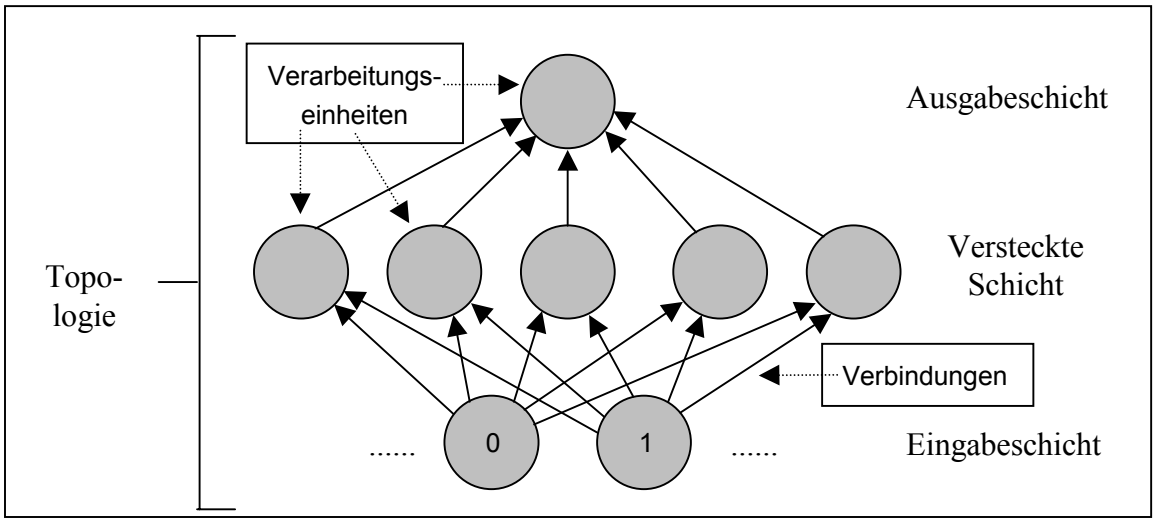

Abb. 6: Eine Beispieltopologie

3.4 Verarbeitungseinheiten

Einen Grundbaustein Künstlicher Neuronaler Netze bildet in Analogie zu den biologischen Vorbildern das künstliche Neuron. Gängiger als der Begriff „künstliches Neuron" sind jedoch die Synonyme Verarbeitungseinheit, Prozessorelement (processing element, PE), Knoten (node) und Einheit (unit).[17]

17 Die Bezeichnung „Neuron" wird in der Literatur auch als „Hype" (Übersteigerung) bezeichnet. Vgl. Hecht-Nielson, R.: Neurocomputing, Reading, MA et. al.: Addison-Wesley 1990, S. 13 und S. 23.

Prozessorelemente verarbeiten *n* numerische Eingangswerte und generieren einen numerischen Ausgabewert. Der Ausgabewert wird über *m* Ausgangsleitungen an verbundene Einheiten oder an die Systemumgebung (als Antwort des Netzwerks) weitergeleitet. Eingangswerte sind entweder Ausgabewerte verbundener Einheiten oder Eingabemuster aus der Systemumgebung. Die Systemumgebung bezeichnet dabei die netzwerkexterne Umwelt, die die Eingabedaten an das Netzwerk anlegt und die Netzwerkausgabe entgegennimmt.[18]

Eine Verarbeitungseinheit besteht im wesentlichen aus den Teilkomponenten (vgl. Abbildung 7):

- Aktivierungszustand (Aktivität)
  Die Aktivität beschreibt, ob ein PE aktiv oder inaktiv ist. Jede Einheit besitzt zu jedem Zeitpunkt einen definierten Aktivierungszustand. Technisch gesehen ist der Aktivierungszustand das Endresultat der Aktivierungsfunktion. Die Gesamtheit der Aktivierungszustände aller Einheiten in einem KNN wird als Netzaktivität bezeichnet.[19]
- Propagierungsfunktion (Eingabefunktion)
  Die Propagierungsfunktion oder Eingabefunktion kombiniert die Eingangswerte eines PE mit ihren entsprechenden Verbindungsgewichten zu einem einzigen Eingabewert, der Netzeingabe.[20]
- Aktivierungsfunktion
  Die Aktivierungsfunktion berechnet aus dem aktuellen Aktivierungszustand und der Netzeingabe den neuen Aktivierungszustand der Verarbeitungseinheit.
- Ausgabefunktion
  Die Ausgabefunktion ist in vielen Netzwerkmodellen ohne Funktion, d. h., der neuberechnete Aktivierungszustand wird ohne Veränderung direkt an die verbundenen Einheiten weitergeleitet.[21] In bestimmten Netzwerkmodellen berechnet die Ausgabefunktion z. B., ob eine Verarbeitungseinheit ihren Aktivierungszustand an die verbundenen Einheiten weiterleiten darf.

Aufgabe einer Verarbeitungseinheit ist die Berechnung eines neuen Aktivierungszustands aus dem aktuellen Zustand und der Netzeingabe sowie die Weitergabe des neuen Zustands an die verbundenen Verarbeitungseinheiten.

Die Informationsverarbeitung der Verarbeitungseinheiten ist durch die Merkmale „Einfachheit“ und „Autonomie“ gekennzeichnet. Verarbeitungseinheiten führen keine komplexen Aufgaben aus, sondern beschränken sich auf wenige, einfache Operationen, die sie unabhängig voneinander, parallel ausführen können.

---

18 Vgl. Hecht-Nielson, R.: Neurocomputing, a. a. O., S. 22.

19 Vgl. Kratzer, K.-P.: Neuronale Netze - Grundlagen und Anwendungen, 2., durchges. Aufl., München et. al.: Hanser, S. 23 f.

20 Vgl. Kemke, C.: Der neuere Konnektionismus, a. a. O., S. 150.

21 Vgl. Schumann, M.; Lohrbach, T. Retzko, R.: Einführung in Aufbau und Arbeitsweise Künstlicher Neuronaler Netze, a. a. O., S. 25.

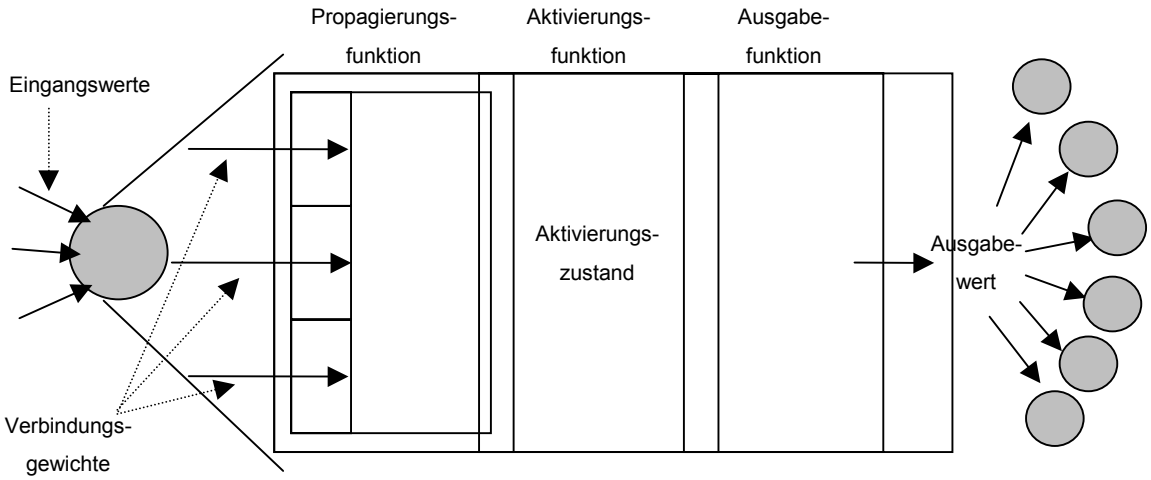

Abb. 7: Schema einer Verarbeitungseinheit

In bestimmten Netzwerkmodellen werden Verarbeitungseinheiten nach Funktion und Position in einem KNN unterschieden:

- Eingabeeinheiten (input units)
  Eingabeeinheiten verarbeiten die Eingabemuster, die an das Netzwerk angelegt werden. In den meisten Netzwerkmodellen dienen sie lediglich als Zwischenspeicher für die Eingabedaten und führen selbst keine Berechnungen aus.
- Ausgabeeinheiten (output units)
  Ausgabeeinheiten dienen zur Aufbereitung einer sinnvoll und einfach interpretierbaren Netzausgabe und damit ebenfalls als Zwischenspeicher.
- Versteckte Einheiten (hidden units)
  Versteckte Einheiten sind interne Verarbeitungseinheiten, die von außen nicht manipulierbar sind. In ihnen finden die eigentlichen Informationsverarbeitungsprozesse statt.

Darüber hinaus unterscheidet man Verarbeitungseinheiten nach ihrem funktionalen Aufbau, d. h., nach den mathematischen Funktionen der einzelnen Teilkomponenten (Propagierungs-, Aktivierungs- und Ausgabefunktion). Durch Kombination unterschiedlicher mathematischer Funktionen für die genannten Teilkomponenten entstehen unterschiedliche Typen von Verarbeitungseinheiten. Zum Beispiel demonstrierte das Perzeptron in Abschnitt 3.1 die Arbeitsweise einer simplen linearen Schwellenwerteinheit (linear threshold unit): die Propagierungsfunktion summiert alle gewichteten Eingabewerte auf; die Aktivierungsfunktion ist eine lineare Schwellenwertfunktion (vgl. Abbildung 8). Weiterentwickelte KNN-Modelle bedienen sich komplexerer Typen von Verarbeitungseinheiten. Einen Überblick gibt z. B. Hoffmann.[22]

22 Vgl. Hoffmann, N.: Kleines Handbuch neuronale Netze - anwendungsorientiertes Wissen zum Lernen und Nachschlagen, Braunschweig, et. al.: Vieweg, 1993, S. 32 f.

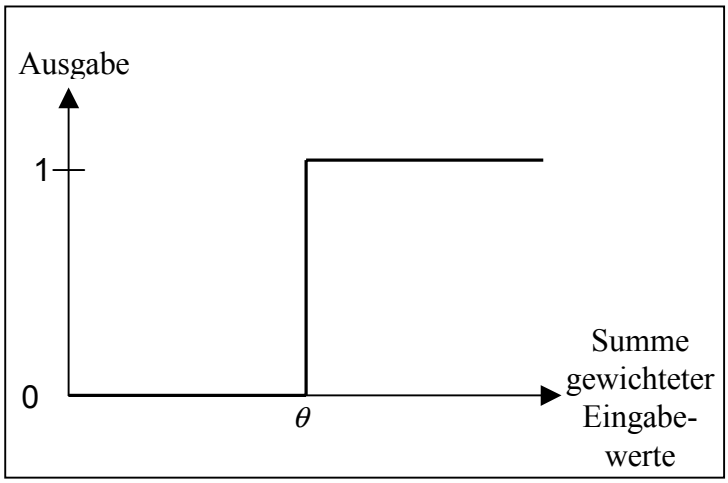

Abb. 8: Graph einer linearen Schwellenwertfunktion

3.5 Verbindungen und Netzwerktopologie

Künstliche Neuronale Netze entstehen durch die geordnete und zielgerichtete Verbindung vieler Verarbeitungseinheiten zu einem Netzwerk. Die räumliche Anordnung der Verarbeitungseinheiten wird dabei als (Netzwerk-)Topologie, Netzwerkstruktur oder Netzwerkarchitektur bezeichnet. Die Topologie eines KNN ist durch die Art und Anzahl der Verarbeitungseinheiten sowie deren Verbindungen untereinander spezifiziert.[23]

Die Verbindungen zwischen zwei Verarbeitungseinheiten sind durch einen Gewichtswert (Verbindungsgewicht) realisiert, über den die Einheiten miteinander kommunizieren. Verbindungsgewichte sind von zentraler Bedeutung, da sie zusammen mit den Aktivierungszuständen das gelernte, verteilte Wissen, d. h. die „künstliche Intelligenz“ des KNN repräsentieren.

Vorrangige Aufgabe der Verbindungen ist die Festlegung der räumlichen Struktur eines KNN. In bestimmten Netzwerkmodellen stellt dabei die Gruppierung funktionsgleicher Verarbeitungseinheiten zu sog. Schichten eine weitere, grundlegende Architekturkomponente dar. Unter einer Schicht (layer, slab) versteht man dabei eine Anzahl von Verarbeitungseinheiten, die eine gleichartige Verhaltensweise hinsichtlich der Datenverarbeitung aufweisen und in der Netzwerkstruktur eine identische Funktion einnehmen.[24] Verbindungen zwischen Prozessorelementen sind deshalb grundsätzlich in Verbindungen zwischen verschiedenen Schichten (inter-layer-Verbindungen) und Verbindungen innerhalb einer Schicht (intra-layer-Verbindungen) zu unterscheiden.

23 Grafisch wird die Netzwerktopologie als gerichteter, bewerteter Graph (Netzwerkgraph) veranschaulicht, in dem die Verarbeitungseinheiten als Knoten und die Verbindungen als Kanten dargestellt werden.

24 Vgl. Kratzer, K.-P.: Neuronale Netze - Grundlagen und Anwendungen, a. a. O., S. 27 und Maren, A. J.; Harston, C.; Pap, R.: Handbook of Neural Computing Applications, a. a. O., S. 46 und Hecht-Nielson, R.: Neurocomputing, a. a. O., S. 23 und Schöneburg, E.; Hansen, N.; Gawelczyk, A.: Neuronale Netze, a. a. O., S. 218.

Zudem regeln die Verbindungen den Informationsfluß in einem KNN. Grundsätzlich ist ein Datenaustausch nur zwischen zwei verbundenen Verarbeitungseinheiten möglich. Der Datentransport wird dabei durch die Ausrichtung und Stärke der Verbindungen gesteuert. Nach der Ausrichtung einer Verbindung unterscheidet man gerichtete (unidirektionalen) und ungerichtete (bidirektionalen) Verbindungen. Unidirektionale Verbindungen definieren eine eindeutige Richtung des Informationsflusses, wohingegen bidirektionale Verbindungen wechselseitigen, rückgekoppelten Informationsfluß zulassen. Die Stärke der Verbindungen wird nach exzitatorischen (verstärkenden) oder inhibitorischen (hemmenden) Wirkung auf verbundene Einheiten unterschieden.

Die Verbindungsausrichtung führt zu zwei unterschiedlichen Ablaufarten in der Verarbeitungsphase von KNN (vgl. Abschnitt 3.7): die vorwärtsgerichtete oder vorwärtsbetriebene Informationsverarbeitung (Feed-Forward, FF) und die rückgekoppelte Informationsverarbeitung (Feed-Backward, FB).

- **Vorwärtsgerichtete Informationsverarbeitung (Feed-Forward, FF)**
  Mit Feed-Forward wird ein gerichteter, d. h. nur in eine Richtung bestehender, Informationsfluß bezeichnet. Die Eingabedaten werden entlang der Verbindungen von der Eingabeschicht zur Ausgabeschicht in einer durch die Netzwerkarchitektur festgelegten Anzahl von Verarbeitungsschritten weiterverarbeitet.[25]
  FF-Netzwerke entstehen durch gerichtete, unidirektionale Inter-Layer-Verbindungen zwischen PE aufeinanderfolgender Schichten. Es bestehen keine Verbindungen zwischen PE innerhalb einer Schicht oder zu vorgeschalteten Schichten. Das KNN ist rückkopplungsfrei.[26] Mathematisch entspricht diese Struktur einem azyklischen Graphen.[27]

- **Rückgekoppelte Informationsverarbeitung (Feed-Backward, FB)**
  Die rückgekoppelte Informationsverarbeitung ist durch einen ungerichteten Informationsfluß gekennzeichnet, der durch Rückkopplungen zwischen Verarbeitungseinheiten entsteht.[28] Die Rückkopplungen basieren auf bidirektionalen Verbindungen zwischen Einheiten innerhalb einer Schicht (lateral feedback), unidirektionalen Verbindungen eines PE mit sich selbst (self-feedback, direct feedback) oder bidirektionalen Verbindungen zwischen PE einer nachgelagerten mit PE einer vorgeschalteten Schicht in einer schichtweisen Architektur (indirect feedback).[29]

25 Vgl. Nauck, D.; Klawonn, F.; Kruse, R.: Neuronale Netze und Fuzzy-Systeme, 2., überarb. und erw. Aufl., Braunschweig, Wiesbaden: Vieweg 1996, S. 137 und Hertz, J.; Krogh, A.; Palmer, R. G.: Introduction to the Theory of Neural Computation, Lecture Notes, Volume I, Santa Fe Institute Studies in the Science of Complexity, Reading, MA et. al.: Addison-Wesley 1991, S. 99 und S. 137.

26 Vgl. Schöneburg, E.; Hansen, N.; Gawelczyk, A.: Neuronale Netze, a. a. O., S. 108.

27 Vgl. Zell, A.: Simulation Neuronaler Netze, 1., unveränderter Nachdruck 1996, Reading, MA et. al.: Addison-Wesley 1994, S. 75.

28 Vgl. Hertz, J.; Krogh, A.; Palmer, R. G.: Introduction to the Theory of Neural Computation, a. a. O., S. 163.

29 Vgl. Scherer, A.: Neuronale Netze - Grundlagen und Anwendungen, a. a. O., S. 55.

Rückgekoppelte Verbindungen führen zu Schleifen (feedback loops) im Datenfluß,[30] d. h., der Ausgabewert einer Verarbeitungseinheit wird als Eingabewert desselben PE oder eines PE der gleichen bzw. einer vorgeschalteten Schicht wiederverwendet. Die Einheiten in einer rückgekoppelten Topologie beeinflussen sich deshalb gegenseitig.[31]

Topologien lassen sich anhand der Netzwerkstruktur und Verarbeitungsrichtung in vier Gruppen eingeteilen:

1. **Einschichtige, vorwärtsbetriebene Topologien (Single-layer feed forward networks)**
   Die ältesten KNN besitzen die einfachste Netzwerkstruktur: einschichtige Topologien. Die Topologien basieren auf unidirektionalen Verbindungen, die eine Eingabeschicht vollständig mit einer Ausgabeschicht verbinden. Durch die gerichteten Verbindungen ist ein Datenfluß nur in eine Richtung, d. h. vorwärtsbetrieben von der Eingabe- zur Ausgabeschicht möglich. Die Prozessorelemente in einschichtigen, vorwärtsbetriebenen Topologien besitzen i. d. R. einfache Aktivierungsfunktionen wie z. B. die lineare Schwellenwerteinheit in Abschnitt 3.1. Typische Netzwerkmodelle dieser Gruppe sind das Perzeptron und das ADALINE (ADAptive LINear Element oder ADAptive LInear NEuron).[32] Diese Modelle sind auf die Lösung linearer Problemstellungen beschränkt und kamen u. a. zur Dämpfung des Echos als Filter in Telefonnetzen zum Einsatz (ADALINE).

2. **Mehrschichtige, vorwärtsbetriebene Topologien (Multilayer feed forward networks)**
   Neue Lernalgorithmen ermöglichten Ende der 80er Jahre eine Erweiterung der einschichtigen, vorwärtsbetriebenen Topologien um versteckte, „von außen“ nicht zugängliche Schichten (hidden layer), mit deren Hilfe auch nichtlineare Probleme gelöst werden können. Die versteckten Schichten erweitern die mathematischen Lösungsfähigkeiten auf höherdimensionale Datenräume.[33] Bekanntester Vertreter dieser Gruppe von Topologien ist das Multilayer-Perzeptron, das eine Erweiterung des Perzeptrons von F. Rosenblatt darstellt.[34]

   Mehrschichtige, vorwärtsbetriebene Topologien besitzen einen hierarchischen Aufbau; d. h., auf die Eingabeschicht folgen eine oder mehrere versteckte Schichten, die wiederum durch eine Ausgabeschicht abgeschlossen werden. Üblicherweise bestehen nur inter-layer-Verbindungen zwischen PE direkt aufeinander folgender Schich-

30 Vgl. Haykin, S.: Neural Networks - A Comprehensive Foundation, London et. al.: Prentice-Hall 1994, S. 15.

31 Vgl. Schöneburg, E.; Hansen, N.; Gawelczyk, A.: Neuronale Netze, a. a. O., S. 108.

32 Das Perzeptron geht auf F. Rosenblatt (1958) zurück. Das ADALINE wurde zwischen 1956 und 1960 von Widrow und Hoff entwickelt. Vgl. Strecker, S., Schwickert, A.: Künstliche Neuronale Netze - Einordnung, Klassifikation und Abgrenzung aus betriebswirtschaftlicher Sicht, in: Arbeitspapiere WI, Nr. 4/97, Hrsg.: Lehrstuhl für Allg. BWL und Wirtschaftsinformatik, Johannes Gutenberg-Universität: Mainz 1997, S. 9.

33 Vgl. Nauck, D.; Klawonn, F.; Kruse, R.: Neuronale Netze und Fuzzy-Systeme, a. a. O., S. 71.

34 Vgl. Hertz, J.; Krogh, A.; Palmer, R. G.: Introduction to the Theory of Neural Computation, a. a. O., S. 115.

ten. Die unidirektionalen Verbindungen verbinden die PE zweier Schichten vollständig oder teilweise miteinander. Der Datenfluß ist von der Eingabe- zur Ausgabeschicht gerichtet und vorwärtsbetrieben (vgl. Abbildung 9). Mehrschichtige, vorwärtsbetriebene Topologien gelten als besonders flexibel, da sie beliebige mathematische Funktionen approximieren können (sog. universelle Approximatoren). Die mathematische Leistungsfähigkeit hängt allerdings direkt von der gewählten Topologie ab, für die es keine problemspezifischen Konstruktionsanleitungen gibt. Problematisch ist daher die Festlegung der optimalen Anzahl von PE und Schichten für ein gegebenes Problem. Die Entwicklung ist deshalb häufig mit einem „Trial and Error"-Prozeß verbunden.[35]

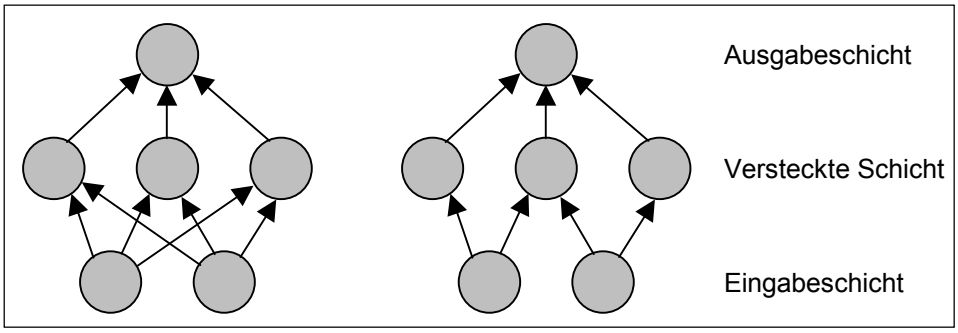

Abb. 9: Eine vollständig verbundene und eine teilweise verbundene mehrschichtige, vorwärtsbetriebene Topologie

3. Topologien mit direkten und indirekten Rückkopplungen

Rückgekoppelte Topologien weisen mindestens eine Schleife (feedback loop) im Informationsfluß auf.[36] Rückkopplungen führen zu Zyklen im Netzwerkgraphen und dementsprechend zu Endlosschleifen im Informationsfluß.[37] Die Ausgabewerte der rückgekoppelten Einheiten werden immer wieder in den Verarbeitungsprozeß zurückgeführt. Aus diesem Grund unterscheiden sich rückgekoppelte Topologien hinsichtlich der Informationsverarbeitung grundlegend von vorwärtsgerichteten Topologien.

Typischerweise minimieren rückgekoppelte Topologien im Verarbeitungsprozeß eine physikalische Energiefunktion: Ein Eingabemuster wird an das Netzwerk angelegt und die Aktivierungszustände der PE solange neu berechnet bis das Netzwerk einen stabilen Zustand (Ruhezustand) in einem Minimum der Energiefunktion erreicht. Dies ist der Fall, wenn bei wiederholter Präsentation desselben Eingabemusters die Aktivierungszustände aller PE konstant bleiben.

35 Vgl. Nauck, D.; Klawonn, F.; Kruse, R.: Neuronale Netze und Fuzzy-Systeme, a. a. O., S. 91.

36 Vgl. Haykin, S.: Neural Networks - A Comprehensive Foundation, a. a. O., S. 20.

37 Vgl. Zell, A.: Simulation Neuronaler Netze, a. a. O., S. 145.

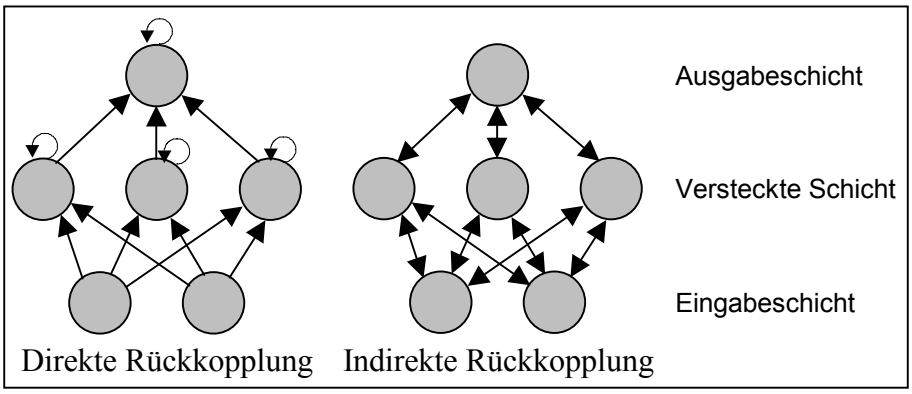

Abb. 10: Topologien mit direkten und indirekten Rückkopplungen

Drei Topologievarianten sind in dieser Gruppe besonders hervorzuheben: Topologien mit direkter Rückkopplung, mehrschichtige Topologien mit indirekter Rückkopplung und die vollständig verbundene Topologie (vgl. Abbildung 10 und 11):[38]

- Topologien mit direkter Rückkopplung (direct feedback)
  Die direkte Rückkopplung, d. h. die Verbindung einer Verarbeitungseinheit mit sich selbst, führt dazu, daß die Einheit ihren eigenen Aktivierungszustand verstärkt oder vermindert.
- Mehrschichtige Topologien mit indirekten Rückkopplungen (indirect feedback)
  In mehrschichtigen Topologien werden indirekte Rückkopplungen dazu benutzt, um bestimmte Merkmale in den Eingabedaten besonders hervorzuheben.

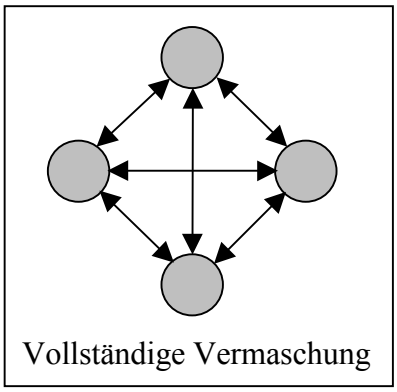

Abb. 11: Eine vollständig verbundene Topologie

- Vollständig verbundene Topologie (vollständige Vermaschung)
  Die vollständige Vermaschung ist ein Spezialfall der indirekten Rückkopplung. Alle Verarbeitungseinheiten innerhalb einer Schicht sind vollständig bidirektional und gewichtssymmetrisch miteinander verbunden, d. h., das Ver-

38 Vgl. Zell, A.: Simulation Neuronaler Netze, a. a. O., S. 78.

bindungsgewicht von PE i nach PE j entspricht dem Verbindungsgewicht von PE j nach i.[39] Im Gegensatz zur indirekten Rückkopplung besitzen vollständig vermaschte Topologien nur eine Schicht, in der zwischen allen Einheiten (auch Eingabeeinheiten) indirekte Rückkopplungen vorliegen. Indirekte rückgekoppelte Netzwerke weisen dagegen eine geschichtete Architektur ohne intra-layer-Verbindungen auf.[40]

4. **Topologien mit lateralen Rückkopplungen und Gitterstrukturen (lateral feedback and lattice structures)**
   In Gitterstrukturen sind die Einheiten der Ausgabeschicht geometrisch als ein-, zwei- oder höher dimensionale Gitter in Form einer Geraden, eines Rechtecks, Quaders oder Hyperquaders angeordnet.[41] Die Eingabeschicht ist vollständig, unidirektional mit der Gitterstruktur verbunden. Innerhalb der Gitterstruktur selbst liegen lateral rückgekoppelte Verbindungen vor, die in Abhängigkeit der lateralen Distanz zweier geometrisch benachbarter PE exzitatorische bzw. inhibitorische Rückkopplungen erzeugen und die Aktivierung betroffener Einheiten verstärken bzw. hemmen (sog. laterale Inhibition). Die Einheiten in der Gitterschicht weisen zudem i. d. R. direkte, exzitatorische Rückkopplungen auf (vgl. Abbildung 12).[42]

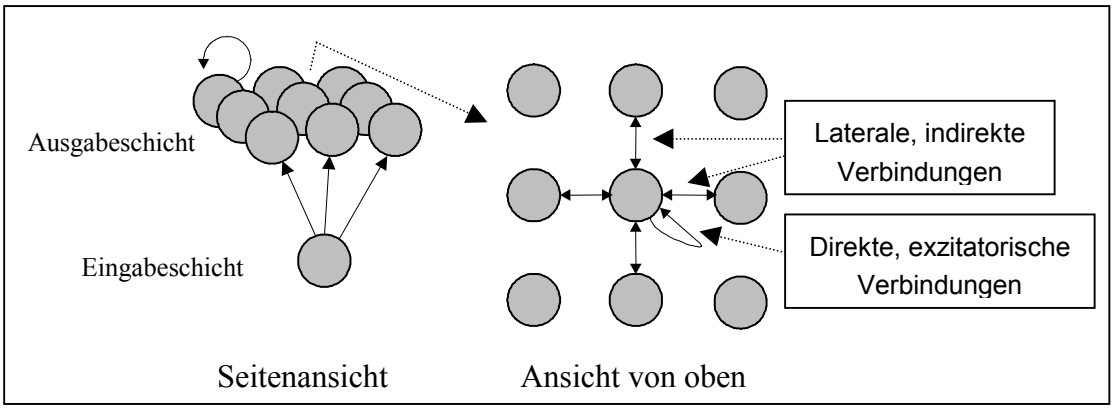

Abb. 12: Topologie mit lateralen Rückkopplungen und Gitterstruktur

Anhand der Ordnung der Eingabeschicht werden zwei unterschiedliche Topologien mit Gitterstrukturen unterschieden: Gitterstrukturen mit wenigen, geometrisch ungeordneten Eingabeeinheiten[43] und Topologien, deren Eingabeschicht bereits in einer geometrisch geordneten, zweidimensionalen Struktur vorliegt.[44]

39 Vgl. Scherer, A.: Neuronale Netze - Grundlagen und Anwendungen, a. a. O., S. 54.

40 Vgl. Zell, A.: Simulation Neuronaler Netze, a. a. O., S. 78 f.

41 Vgl. Nauck, D.; Klawonn, F.; Kruse, R.: Neuronale Netze und Fuzzy-Systeme, a. a. O., S. 125.

42 Vgl. Hertz, J.; Krogh, A.; Palmer, R. G.: Introduction to the Theory of Neural Computation, a. a. O., S. 233 f. und Haykin, S.: Neural Networks - A Comprehensive Foundation, a. a. O., S. 23 f. und S. 401 ff.

43 Vgl. Kohonen, T.: Self-organized formation of topologically correct feature maps, Biological Cybernetics, 43, 1988, S. 59-69.

44 Vgl. Willshaw, D. J.; von der Malsburg, C.: How Patterned Neural Connections Can Be Set Up By Self-Organization, Proceedings of the Royal Society of London, Series B 194, 1976, S. 431-445.

Die geometrische Anordnung der Verarbeitungseinheiten in einer Gitterstruktur bildet die Basis für eine Gruppe von Netzwerkmodellen, die als selbstorganisierende Karten bezeichnet werden. Selbstorganisation beschreibt die Fähigkeit, Klassen in den Eingabedaten selbständig abbilden zu können (Clustering). Der Begriff „Karte“ bezieht sich auf die Abbildung der Topologie des Eingabedatenraums in der Netzwerkstruktur. Selbstorganisierende Karten werden deshalb auch als topologie-erhaltende Karten (topology preserving maps), sensorische Karten oder topographische Karten bezeichnet.[45] Die Anwendungsgebiete selbstorganisierender Karten liegen in der Robotersteuerung, Spracherkennung und Optimierung (z. B. Travelling-Salesman-Problem).

3.6 Lernphase

Lernen in KNN ist ein Prozeß, in dem einem KNN nacheinander Beispielmuster aus der Problemstellung präsentiert werden und sich die Verbindungsgewichte gemäß eines Lernalgorithmus selbständig so adaptieren, daß das KNN die gewünschte Aufgabe lösen kann. Der Vorgang des Lernens wird häufig auch als Trainieren des Netzwerks bezeichnet.[46] Die Beispielmuster werden dementsprechend als Trainingsmuster oder Trainingsdaten aus einer Trainingsmenge von Mustern aufgefaßt.[47]

Das Training eines KNN entspricht im übertragenen Sinne dem Programmiervorgang eines konventionellen IV-Systems. Im Gegensatz zur klassischen Programmierung ist das entscheidendes Merkmal des Lernprozesses die selbständige Lernfähigkeit, d. h. „eine gegebene Aufgabe (weitgehend) selbständig aus Beispielen“[48] zu lösen, so daß aufwendige Software-Design- und Programmiertätigkeiten für einen Problemlösungsalgorithmus entfallen.[49]

Ziel des Lernprozesses ist es, ein KNN so zu trainieren, daß es unbekannte, nicht gelernte Eingabemuster „korrekt“ verarbeiten kann. Eingabemuster, die ähnliche Merkmale aufweisen, sollen erkannt und einer ähnlichen Ausgabe zugeordnet werden. Die Korrektheit der Verarbeitung ist vor der jeweiligen Problemstellung zu betrachten: Im Fall der Kreditwürdigkeitsprüfung sollen zukünftige, d. h. dem KNN unbekannte Kreditanträge derjenigen Bonitätsklasse zugeordnet werden, die in der Vergangenheit ähnliche Kreditmerkmale aufwies und sich damit eine Prognose über die Zahlungsfähigkeit des Antragstellers in der Zukunft ergeben. Diese Eigenschaft von KNN wird als Generalisierungsfähigkeit bezeichnet und ist mit einer mathematischen Interpolation vergleichbar.

45 Vgl. Hertz, J.; Krogh, A.; Palmer, R. G.: Introduction to the Theory of Neural Computation, a. a. O., S. 232.

46 Vgl. Hertz, J.; Krogh, A.; Palmer, R. G.: Introduction to the Theory of Neural Computation, a. a. O., S. 10.

47 Vgl. Zell, A.: Simulation Neuronaler Netze, a. a. O., S. 73 und S. 93.

48 Vgl. Zell, A.: Simulation Neuronaler Netze, a. a. O., S. 83.

49 Vgl. Hertz, J.; Krogh, A.; Palmer, R. G.: Introduction to the Theory of Neural Computation, a. a. O., S. 10.

Der Ablauf des Lernprozesses läßt sich allgemein in folgende Teilschritte zerlegen:

1. Im Ausgangszustand sind die Verbindungsgewichte des KNN mit Zufalls- oder Experimentalwerten belegt.
2. Eine Menge von Trainingsmustern wird in beliebiger Reihenfolge an das Netzwerk angelegt.
3. Für jedes Muster berechnet das KNN eine Ausgabe mit der vorbelegten, momentanen Gewichtskonfiguration.
4. Die Gewichte werden nach einem Lernalgorithmus angepaßt.
5. Der Prozeß endet, wenn die Ausgabe einem Zielkriterium genügt (z. B. ein abgeschlossener Kreditfall der richtigen Bonitätsklasse zugeordnet wurde) oder eine vorgegebene Anzahl von Lernschritten erreicht ist.

Der Aufbau und Ablauf des Lernens ist abhängig von der Lernaufgabe, d. h. der zu lösenden Aufgabenstellung und damit letztlich vom Anwendungszweck des KNN. Man unterscheidet feste und freie Lernaufgaben:[50]

- Feste Lernaufgaben sind durch eine Trainingsmenge von paarweise korrespondierenden Ein- und Ausgabemustern gekennzeichnet. Das KNN soll zu jedem beliebigen Eingabemuster die zugehörige Ausgabe erlernen.
- Freie Lernaufgaben sind dagegen durch eine Trainingsmenge von Eingabemustern ohne korrespondierende Ausgabe gekennzeichnet. Das KNN soll selbständig Ausgabewerte ermitteln und ähnliche Eingabemuster auch ähnlichen Ausgaben zuordnen.

Die Erfüllung fester und freier Lernaufgaben wird durch den Lernalgorithmus realisiert, der den Kern des Lernprozesses bildet. Der Lernalgorithmus legt Rechenvorschriften zur Adaption des KNN fest und soll erreichen, daß das Netzwerk die Lernaufgabe verallgemeinert, d. h. eine hohe Generalisierungsleistung erzielt. Lernalgorithmen werden grundsätzlich in überwachte Lernalgorithmen bei fester Lernaufgabe (überwachtes Lernen) und unüberwachte Lernalgorithmen bei freier Lernaufgabe (unüberwachtes Lernen) unterschieden:

- **Überwachtes Lernen (Supervised learning)**[51]
  Überwachtes Lernen wird häufig als „Lernen mit Lehrer“ bezeichnet, da zu jedem Eingabemuster in der Trainingsmenge ein bekanntes, korrespondierendes Ausgabemuster vorliegt und der „Lernzustand“ des KNN damit überwacht und gesteuert werden kann.[52]

50 Vgl. Nauck, D.; Klawonn, F.; Kruse, R.: Neuronale Netze und Fuzzy-Systeme, a. a. O., S. 29.

51 Eine weitere Form des überwachten Lernens ist das sog. Reinforcement Learning oder „Lernen mit Kritiker“. Vgl. Hertz, J.; Krogh, A.; Palmer, R. G.: Introduction to the Theory of Neural Computation, a. a. O., S. 188.

52 Vgl. Hertz, J.; Krogh, A.; Palmer, R. G.: Introduction to the Theory of Neural Computation, a. a. O., S. 89.

Die paarweise korrespondierenden Eingabe-/Ausgabemuster repräsentieren vorhandenes Wissen über die zu lösende Aufgabe und damit über die Umwelt des KNN. Anschaulich läßt sich dies an der Aktienkursprognose demonstrieren. Fundamentalanalytische oder technische Kapitalmarktdaten bilden die Eingabemuster und historische Aktienkurse die korrespondierenden Ausgabemuster in der Trainingsmenge. Das KNN soll die zugrundliegende Dynamik des Kapitalmarkts approximieren; sprich, das KNN soll die zukünftige Kursentwicklung berechnen.

Dazu wird ein Eingabemuster gleichzeitig mit dem korrespondierenden Ausgabemuster an das KNN angelegt. Im Aktienkursbeispiel werden die kodierten Kapitalmarkt- und Kursdaten eingelesen. Die Netzausgabe für das Eingabemuster wird berechnet und mit dem korrespondierenden Ausgabemuster (im Beispiel dem Kurs oder Trend) verglichen. Aus der Differenz zwischen der vom KNN berechneten Netzausgabe und dem bekannten Ausgabemuster (historischer Kurswert) ergibt sich der Fehler des KNN, der durch graduelle Anpassung der Verbindungsgewichte minimiert wird. Anschließend wird das nächste Eingabe-/Ausgabepaar (Analyse- und Kursdaten vom nächsten Tag) an das KNN angelegt. Dieser iterative Prozeß wird für alle Paare in der Trainingsmenge solange wiederholt, bis das KNN die Eingabe-/Ausgabepaare korrekt zuordnen kann. Anhand unbekannter, nicht gelernter Eingabemuster, deren Ausgabewerte bekannt sind (historische Kurswerte, die dem KNN in der Lernphase nicht präsentiert wurden), wird die Generalisierungsleistung des KNN überprüft.[53]

Anwendung finden überwachte Lernalgorithmen bei Musterklassifikationsaufgaben (z. B. bei der Kreditwürdigkeitsprüfung) oder bei der Funktionsapproximation (z. B. bei Kursprognosen).[54]

- **Unüberwachtes Lernen (Unsupervised learning, Self-supervised learning)**
  Unüberwachtes Lernen wird auch als „Lernen ohne Lehrer“ oder entdeckendes Lernen bezeichnet, da die Trainingsmenge keine paarweise korrespondierenden Eingabe-/Ausgabemuster, sondern nur Eingabemuster enthält und das KNN keine Rückmeldung darüber erhält, ob es Eingabemuster korrekt klassifiziert.[55] Eine durch korrespondierende Ausgabemuster extern gesteuerte Fehlerermittlung ist nicht möglich. Entdeckende Lernalgorithmen enthalten deshalb ein aufgabenunabhängiges Fehlermaß.[56]

  Unüberwachte Lernalgorithmen sollen selbständig statistische Eigenschaften aus den Eingabedaten extrahieren, d. h. Muster, Merkmale, Regelmäßigkeiten, Korrelationen und Klassen identifizieren und in der Netzwerkstruktur abbil-

53 Vgl. Zell, A.: Simulation Neuronaler Netze, a. a. O., S. 93.

54 Vgl. Nauck, D.; Klawonn, F.; Kruse, R.: Neuronale Netze und Fuzzy-Systeme, a. a. O., S. 29.

55 Vgl. Schöneburg, E.; Hansen, N.; Gawelczyk, A.: Neuronale Netze, a. a. O., S. 29.

56 Vgl. Haykin, S.: Neural Networks - A Comprehensive Foundation, a. a. O., S. 65 und Nauck, D.; Klawonn, F.; Kruse, R.: Neuronale Netze und Fuzzy-Systeme, a. a. O., S. 33.

den. Der Lernalgorithmus bedarf daher Mechanismen zur Selbstorganisation, die die Eigenschaften der Eingabedaten in den Verarbeitungseinheiten und Verbindungen abbilden. Die Topologie bestimmt, welche Mustermerkmale intern repräsentiert werden können.

Allgemeine Voraussetzung für den erfolgreichen Einsatz von unüberwachten Lernalgorithmen sind signifikante Redundanzen in den Eingabemustern, ohne die eine Klassifikation nicht möglich ist.[57]

Eine typische betriebswirtschaftliche Anwendung für unüberwacht lernende KNN ist die Marktsegmentierung

Typische Aufgabengebiete von unüberwachten Lernalgorithmen sind die Klassenbildung und die Klassifizierung (z. B. zur Marktsegmentierung) sowie die Optimierung (z. B. das Travelling-Salesman-Problem).

Eine andere Typisierung von Lernalgorithmen stellt daher die zugrundliegenden Aufgabentypen (Lernparadigmen) nach ihrer Zuordnung von Eingabe- und Ausgabemuster in den Mittelpunkt:[58]

- Pattern Association (Synonyme: Musterassoziation, Heteroassoziation)
  Musterassoziation bedeutet ganz allgemein die Zuordnung von Eingabe- zu einem Ausgabemuster. Unter Heteroassoziation versteht man speziell die Zuordnung eines Eingabemusters zu einem davon unterschiedlichen Ausgabemuster.
  Dieses Lernparadigma wird in der Funktionsapproximation (Approximation), Prognose (Prediction) sowie in regelungstechnischen Anwendungen (Neurocontrol) angewendet.
- Auto-Association (Autoassoziation)
  Die Autoassoziation unterscheidet sich von der Musterassoziation durch identische Eingabe-/Ausgabemuster, d. h., das KNN soll ein Eingabemuster mit sich selbst assoziieren und wiedererkennen. Anwendungsbereiche sind die Bilderkennung (z. B. die Gesichtserkennung in Zugangssystemen) und inhaltsbasierte Speicherung (inhaltsadressierbare Speicher). Das KNN soll verrauschte oder unvollständige Muster vervollständigen und gelernten Mustern zuordnen (Mustervervollständigung, Pattern Completion).
- Pattern Classification (Mustererkennung, Musterklassifizierung)
  Bei der Mustererkennung sollen mehrere zusammengehörende Eingabemuster einer von wenigen disjunkten, a priori bekannten Musterklassen zugeordnet werden. Anwendungsgebiet sind alle Arten von Klassifikationsaufgaben (z. B. die Kreditwürdigkeitsprüfung).

57 Vgl. Hertz, J.; Krogh, A.; Palmer, R. G.: Introduction to the Theory of Neural Computation, a. a. O., S. 197.

58 Vgl. Haykin, S.: Neural Networks - A Comprehensive Foundation, a. a. O., S. 66 ff. und Nauck, D.; Klawonn, F.; Kruse, R.: Neuronale Netze und Fuzzy-Systeme, a. a. O., S. 34 f.

- Regularity Detection (Ähnlichkeitserkennung, Kategorisierung, Klassenbildung, Klassenentdeckung)
  Die Ähnlichkeitserkennung unterscheidet sich von der Mustererkennung dadurch, daß die Musterklassen a priori nicht bekannt sind. Das KNN soll für Eingabemuster, die statistisch ähnliche Eigenschaften aufweisen, selbständig eine Musterklasse bilden. Neben Klassifikationsaufgaben (z. B. Marktsegmentierung) stellt die Datenanalyse (z. B. für das Data Mining) ein weiteres Anwendungsgebiet dar.
- Combinatorial Optimization Problems (kombinatorische Optimierungsprobleme)
  Dieser Aufgabentyp unterscheidet sich grundsätzlich von Assoziationsaufgaben. Das KNN soll ein Gleichungssystem minimieren und die optimale Lösung approximieren. Die Parameter des Gleichungssystems sind dabei in der Netzwerkstruktur kodiert und der Lernalgorithmus minimiert eine Fehlerfunktion.
  Anwendungsgebiet sind vor allem kombinatorische Optimierungsprobleme, bei denen der Rechenaufwand exponentiell statt polynomial zur Problemgröße wächst (sog. NP-vollständige Probleme wie z. B. das Travelling-Salesman-Problem).

3.7 Verarbeitungsphase

An die Lernphase eines KNN schließt sich die Verarbeitungs- oder Anwendungsphase an. Die Verarbeitungsphase entspricht übertragen auf den Software-Lebenszyklus der Nutzungszeit eines Software-Produkts.[59] In der Verarbeitungsphase werden nicht-gelernte Daten aus einem Anwendungsfall als Eingabemuster an das KNN angelegt. Die Verbindungsgewichte sind durch die Lernphase fixiert und repräsentieren das gelernte, verteilte Wissen des KNN. Das KNN generiert eine Netzausgabe aufgrund seines gelernten Wissens.

Schematisch laufen die Vorgänge in der Verarbeitungsphase in drei Schritten ab:

1. Fallspezifisches Muster anlegen
2. KNN berechnet Netzausgabe
3. Netzausgabe weiterverarbeiten

Am Beispiel der Aktienkursprognose werden die Vorgänge in der Verarbeitungsphase deutlich: Angenommen ein KNN wurde für eine kurzfristige, tageweise Kursprognose mit technischen Daten (Vortagskurs, Veränderungsrichtung etc.) trainiert. In der Verarbeitungsphase wird dem KNN ein Eingabemuster präsentiert, daß die aktuellen, heutigen Tagesdaten der Aktie enthält. Das KNN berechnet dazu einen Folgewert, der den morgigen Kurs der Aktie prognostiziert.

59 Vgl. Stahlknecht, P.: Einführung in die Wirtschaftsinformatik, 7., vollst. Überarb. und erw. Aufl., Berlin et. al.: Springer 1995, S. 242.

3.8 Konnektionistische Wissensrepräsentation und Wissensverarbeitung

Wissen ist ein komplexes Konstrukt, das im Kontext menschlicher und künstlicher Intelligenz kontrovers diskutiert wird. Pragmatisch läßt sich Wissen als deklaratives Wissen (Faktenwissen), prozedurales Wissen (Handlungswissen) und Metawissen (Wissen über Planung und Steuerung von Handlungen) auffassen. Für die maschinelle Wissensverarbeitung (knowledge processing) ist eine geeignete Wissensrepräsentation notwendig. Unter Wissensrepräsentation (knowledge representation) versteht man allgemein die ziel- bzw. problembezogene Darstellung und „Kodierung von Wissen in geeigneten Datenstrukturen“[60]. Die Wissensverarbeitung leitet aus bekanntem, gespeicherten Wissen in einem Schlußfolgerungsprozeß (Inferenz) neues Wissen ab. Die Inferenzmechanismen hängen dabei von der gewählten Wissensrepräsentation ab.[61]

Die konnektionistische Wissensrepräsentation basiert auf dem Prinzip der verteilten Repräsentation. Das heißt, Objekte der realen Welt (Fakten, Regeln, Ereignisse usw.) werden nicht explizit durch einzelne komplexe Entitäten, sondern implizit durch eine Gesamtheit vieler, einfacher Verarbeitungseinheiten repräsentiert. Das Wissen ist verteilt in den Verbindungsgewichten und den Aktivierungszuständen der Verarbeitungseinheiten dezentral gespeichert. Eine Zuordnung von Wissenselementen zu einzelnen Komponenten eines KNN ist daher nicht möglich. Die konnektionistische Wissensrepräsentation erlaubt keine Interpretation des intern kodierten Wissens. Eine Zuordnung von einzelnen Wissenselementen zu einzelnen KNN-Komponenten ist nicht möglich, weil das Wissen durch den Gesamtzustand aller Komponenten eines KNN beschrieben ist. Die Interpretation konnektionistischen Wissens hieße aber, sich alle Komponenten eines komplexen KNN ständig und umfassend vergegenwärtigen zu müssen. Lösungen eines KNN lassen sich deshalb nicht begründen und Lösungswege nicht aus der Netzwerkstruktur heraus erklären (fehlende Erklärungskomponente).[62]

Die konnektionistische Wissensverarbeitung wird als „parallel distributed processing“ (PDP) bezeichnet. Konnektionistische Systeme verarbeiten Wissen durch die Interaktion vieler einfacher Verarbeitungseinheiten, die über den Austausch verstärkender oder hemmender Signale parallel und gleichzeitig auf verteilt gespeicherte Elemente des konnektionistischen Wissens zugreifen.[63] Der Schlußfolgerungsprozeß entspricht einer Generalisierung; d. h., unbekannte Eingabemuster werden demjenigen Ausgabemuster zugeordnet, daß mit dem ähnlichsten gelernten Eingabemuster korrespondiert. Dieser Generalisierungsmechanismus führt zu einem inexakten, unscharfem (evidentiellen) Schließen gegenüber dem exakten, logischen Schließen regelbasierter Systeme (z. B. Expertensysteme). Konnektionistische Systeme sind deshalb in der Lage „softe“ Informationen, also unvollständiges und unscharfes Wissen zu verarbeiten. Eine Entschei-

60 Vgl. Nauck, D.; Klawonn, F.; Kruse, R.: Neuronale Netze und Fuzzy-Systeme, a. a. O., S. 186.

61 Vgl. Haykin, S.: Neural Networks - A Comprehensive Foundation, a. a. O., S. 99.

62 Vgl. Kemke, C.: Der neuere Konnektionismus, a. a. O., S. 144 und Nauck, D.; Klawonn, F.; Kruse, R.: Neuronale Netze und Fuzzy-Systeme, a. a. O., S. 186.

63 Vgl. Haykin, S.: Neural Networks - A Comprehensive Foundation, a. a. O., S. 34.

dungsfindung in KNN ist daher auch möglich, wenn nicht alle Randbedingungen vollständig erfüllt sind (z. B. fehlende oder unscharfe Mustermerkmale). Dies erlaubt die Behandlung von Ausnahmen, Grenzfällen und Abweichungen, die mit regelbasierten Systemen nicht problemlos zu verarbeiten sind.[64]

4 Netzwerkübergreifende Eigenschaften von KNN

4.1 Positive Eigenschaften von KNN

Aus der Struktur und der Informationsverarbeitung Künstlicher Neuronaler Netze ergeben sich Eigenschaften, die KNN gegenüber herkömmlichen Informationssystemen auszeichnen:[65]

- Selbständige Lernfähigkeit
  Künstliche Neuronale Netze werden durch Lernverfahren trainiert, mit denen sie ohne explizite formale Darstellung problembezogenes Wissen anhand von Beispielen aufnehmen können.[66] Dagegen benötigen regelbasierte Expertensysteme während der Wissensakquisition i. d. R. menschliche Experten, die ihre intuitives Erfahrungswissen in einem exakten Regelwerk explizit beschreiben müssen.
- Adaptivität
  Künstliche Neuronale Netze passen ihre Verbindungsgewichte selbständig an ein gegebenes Problem an und können veränderte Umweltbedingungen durch Nachlernen adaptieren. Sie lassen sich einfacher an Verändungen in der Problemstellung anpassen als herkömmliche Algorithmen.[67] Selbst nichtstationäre Probleme, in denen sich die Bedingungen mit der Zeit kontinuierlich verändern, können durch Gewichtsanpassung in Echtzeit gelöst werden.[68]
- Generalisierungsfähigkeit
  Das in den Trainingsdaten enthaltene Wissen führt auch für nicht gelernte Eingaben zu korrekten Entscheidungen und besitzt deshalb eine über die Trainingsmenge hinausgehende Allgemeingültigkeit. Problemlösungen sind auch für nicht gelernte, neue Eingaben möglich.[69]
- Hohe Performance durch Parallelverarbeitung
  KNN sind aufgrund der verteilten Wissensrepräsentation parallele Algorithmen und deshalb für eine Simulation auf massiv-parallelen Hardwarearchitekturen prädestiniert.[70] Insbesondere bei kognitiven Problemen (z. B. der Mustererkennung) errei-

64 Vgl. Kemke, C.: Der neuere Konnektionismus, a. a. O., S. 146.

65 Vgl. Corsten, H.; May, C.: Anwendungsfelder Neuronaler Netze und ihre Umsetzung, a. a. O., S. 4.

66 Vgl. Scherer, A.: Neuronale Netze - Grundlagen und Anwendungen, a. a. O., S. 5 und Nauck, D.; Klawonn, F.; Kruse, R.: Neuronale Netze und Fuzzy-Systeme, a. a. O., S. 35.

67 Vgl. Zell, A.: Simulation Neuronaler Netze, a. a. O., S. 27.

68 Vgl. Haykin, S.: Neural Networks - A Comprehensive Foundation, a. a. O., S. 5.

69 Vgl. Scherer, A.: Neuronale Netze - Grundlagen und Anwendungen, a. a. O., S. 5 und Corsten, H.; May, C.: Anwendungsfelder Neuronaler Netze und ihre Umsetzung, a. a. O., S. 4.

70 Vgl. Zell, A.: Simulation Neuronaler Netze, a. a. O., S. 27.

chen die Verfahren eine hohe Verarbeitungsgeschwindigkeit, die sie auch für Echtzeitanwendungen (z. B. in der Robotik) nutzbar macht.[71]

- Fehlertoleranz bei Fehlfunktionen
  Künstliche Neuronale Netze reagieren bei entsprechender Netzwerkarchitektur tolerant auf den Ausfall eines oder weniger Bausteine (z. B. durch Fehlfunktionen oder die Entfernung von Verbindungen und Verarbeitungseinheiten), da die verteilte Wissensrepräsentation graduelle Informationsverluste kompensieren kann.[72] Daraus ergibt sich die Robustheit gegenüber Datenmängeln.
- Robustheit gegenüber Störungen und Datenmängeln
  Die Leistungsfähigkeit nimmt auch bei inkorrekten, verrauschten oder widersprüchlichen Eingabedaten in der Verarbeitungsphase nur leicht ab.[73]
- „graceful degradation“
  Aufgrund der Fehlertoleranz und Robustheit nimmt die Leistungsfähigkeit bei teilweisem Ausfall und inkorrekten Eingabedaten in kleinen Schritten ab und endet nicht abrupt wie in herkömmlichen Informationssystemen.[74] Diese schrittweise Verminderung der Leistungsfähigkeit wird in der Systemtheorie als „graceful degradation“ bezeichnet. Ab einem bestimmten Ausmaß von Störungen (z. B. bei zu verrauschten Eingabedaten) bricht jedoch die Qualität der Ergebnisse ein.[75]
- Assoziationsfähigkeit
  Assoziationsfähigkeit ist die „... Eigenschaft Neuronaler Netze, Ähnlichkeiten zwischen gelernten Mustern und der Eingabe zu erkennen und dementsprechend eine sinnvolle Ausgabe zu liefern“[76]. Künstliche Neuronale Netze lassen sich gegenüber herkömmlichen Programmen (adreßbezogene Speicherung) aufgrund der Assoziationsfähigkeit als inhaltsbezogene Speicher (Assoziativspeicher) einsetzen.[77] Beispielsweise kann zu einer Zeichenkette ein Bild gespeichert werden und auf die Eingabe eines Teiltextes das entsprechende Bild mit dem Text assoziiert werden.
- Nichtlinearität
  Künstliche Neuronale Netze sind gegenüber linearen statistischen Verfahren (z. B. multivariate lineare Diskriminanzanalyse) in der Lage nichtlineare funktionale Zusammenhänge zu verarbeiten.[78]

---

71 Vgl. Zell, A.: Simulation Neuronaler Netze, a. a. O., S. 27.

72 Vgl. Schumann, M.; Lohrbach, T. Retzko, R.: Einführung in Aufbau und Arbeitsweise Künstlicher Neuronaler Netze, a. a. O., S. 10.

73 Vgl. Zell, A.: Simulation Neuronaler Netze, a. a. O., S. 27.

74 Vgl. Nauck, D.; Klawonn, F.; Kruse, R.: Neuronale Netze und Fuzzy-Systeme, a. a. O., S. 35.

75 Vgl. Ritter, H.; Martinez, T.; Schulten, K.: Neuronale Netze - Eine Einführung in die Neuroinformatik selbstorganisierender Netzwerke, a. a. O., S. 51.

76 Vgl. Schöneburg, E.; Hansen, N.; Gawelczyk, A.: Neuronale Netze, a. a. O., S. 215.

77 Vgl. Zell, A.: Simulation Neuronaler Netze, a. a. O., S. 27.

78 Vgl. Haykin, S.: Neural Networks - A Comprehensive Foundation, a. a. O., S. 4 und Goontilake, S.: Intelligent Systems in Finance and Business: An Overview, in: Intelligent Systems in Finance and Business, Hrsg.: Goontilake, S.; Treleaven, P., Chichester et. al.: John Wiley & Sons 1995, S. 10.

4.2 Negative Eigenschaften von KNN

Die charakteristischen Merkmale konnektionistischer Wissensverarbeitung erschweren den Einsatz von KNN in bestimmten betriebswirtschaftlichen Anwendungen:

- Komplizierte Analysierbarkeit und eingeschränkte Nachvollziehbarkeit
  Für den praktischen Einsatz erweist sich die komplizierte Analysierbarkeit und dadurch eingeschränkte Nachvollziehbarkeit von Entscheidungen eines KNN als Nachteil. Das Wissen in KNN läßt sich nicht oder nur eingeschränkt von außen interpretieren, deshalb ist das gelernte, implizite Modell nicht nachvollziehbar.[79]
- Fehlende Erklärungskomponente
  Die Entscheidungen, repräsentiert durch die Netzausgabe, fällt das KNN aufgrund seines gelernten Wissens ohne Angabe des Lösungswegs.[80]
- Akzeptanzprobleme
  In praktischen Anwendungen, in denen es auf eine Erklärung bzw. Rechtfertigung von Entscheidungen ankommt, führen die genannten Eigenschaften deshalb zu Akzeptanzproblemen.[81] Ein typisches Beispiel ist die Konsumentenkreditvergabe: Kunden eines Kreditinstitutes, deren Kreditanträge abgelehnt wurden, verlangen in der Regel nach einer Begründung. Konnektionistische Systeme können diese im Gegensatz z. B. zu Expertensystemen nicht liefern.[82]
- „Trial and Error"-Entwicklungsprozeß
  Der Entwicklungsprozeß eines KNN gestaltet sich experimentell-explorativ und entspricht bislang nicht einer ingenieurswissenschaftlich geplanten Vorgehensweise vergleichbar dem Software-Engineering.[83] Wesentliches Problem bei der Entwicklung ist das Fehlen einer allgemeingültigen Entwurfsmethode mit Handlungsanweisungen für die Entwicklung eines KNN.[84] Fehlende Entwurfsmethoden verursachen lange Entwicklungszeiten, d. h. letztlich hohe Kosten.[85] Als Lösungsvorschläge existieren Heuristiken und Faustregeln, die jedoch theoretischer Grundlage entbehren und meist aufgabenspezifische Erkenntnisse wiederspiegeln. Diese Erkenntnisse

79 Vgl. Nauck, D.; Klawonn, F.; Kruse, R.: Neuronale Netze und Fuzzy-Systeme, a. a. O., S. 35.

80 Vgl. Kurbel, K.; Pietsch, W.: Eine Beurteilung konnektionistischer Modelle auf der Grundlage ausgewählter Anwendungsprobleme und Vorschläge zur Erweiterung, in: WI 5/91, S. 361 und Corsten, H.; May, C.: Anwendungsfelder Neuronaler Netze und ihre Umsetzung, a. a. O., S. 4.

81 Vgl. Rehkugler, H.; Poddig, T.: Anwendungsperspektiven und Anwendungsprobleme von Künstlichen Neuronalen Netzwerken, a. a. O., S. 56.

82 Dieser Nachteil läßt sich jedoch durch Kombination eines KNN mit anderen KI-Verfahren (Fuzzy Logic, Genetische Algorithmen) in einem hybriden, intelligenten Entscheidungssystem lösen bzw. abmildern. Vgl. z. B. Nauck, D.; Klawonn, F.; Kruse, R.: Neuronale Netze und Fuzzy-Systeme, a. a. O.

83 Vgl. Kurbel, K.; Pietsch, W.: Eine Beurteilung konnektionistischer Modelle auf der Grundlage ausgewählter Anwendungsprobleme und Vorschläge zur Erweiterung, a. a. O., S. 361.

84 Vgl. Rehkugler, H.; Schmidt-von Rhein, A.: Kreditwürdigkeitsanalyse und -prognose für Privatkundenkredite mittels statistischer Methoden und Künstlicher Neuronaler Netze - Eine empirisch-vergleichende Studie, Otto-Friedrich-Universität Bamberg, Bamberger Betriebswirtschaftliche Beiträge, Nr. 93/1993, 1993, S. 45.

85 Vgl. Corsten, H.; May, C.: Anwendungsfelder Neuronaler Netze und ihre Umsetzung, a. a. O., S. 5.

sind daher selten auf andere Anwendungen übertragbar und besitzen keine Allgemeingültigkeit.[86] In der Praxis ist der Entwickler deshalb auf einen „Trial and Error"-Prozeß angewiesen, in dem Netzwerktopologien und Lernprozesse ausgewählt, getestet und optimiert werden bis eine akzeptable Lösung gefunden bzw. ein zeitliches Limit erreicht ist. Software-Qualität und -Zuverlässigkeit lassen sich daher nicht mit den Maßstäben der klassischen Software-Entwicklung messen.[87]

5 Hinweis auf weiterführende Darstellungen

In dieser Arbeit wurden die Grundlagen und Kernelemente konnektionistischer Modelle dargestellt. Abschließend soll darauf hingewiesen werden, daß die Darstellung wesentliche Details für die Entwicklung und den Einsatz Künstlicher Neuronaler Netze aus Vereinfachungsgründen vernachlässigt. Die vorgestellten Kernkomponenten bilden lediglich die „Essenz" bekannter Netzwerkmodelle. Mit diesem Grundgerüst ist noch kein KNN konstruierbar. Erst durch das inhaltliche Ausfüllen des Rahmens entstehen konkrete Netzwerkmodelle wie das dargestellte Perzeptron. Deshalb sei an dieser Stelle noch einmal auf die angegebene Literatur verwiesen, in der die genannten Netzwerkmodelle detailliert beschrieben sind. Besonders hervorzuheben ist auch die kommentierte Literaturliste in Kapitel 4 des Neural Network FAQ zum Thema „Books, data, etc....".[88]

86 Vgl. Schumann, M.; Lohrbach, T. Retzko, R.: Einführung in Aufbau und Arbeitsweise Künstlicher Neuronaler Netze, a. a. O., S. 16.

87 Vgl. dazu die Kritik in Kurbel, K.; Pietsch, W.: Eine Beurteilung konnektionistischer Modelle auf der Grundlage ausgewählter Anwendungsprobleme und Vorschläge zur Erweiterung, a. a. O., S. 360 ff.

88 Vgl. Sarle, W.S. (Hrsg.), Neural Network FAQ, part 4 of 7: Books, data, etc., periodic posting to the Usenet newsgroup comp.ai.neural-nets, 1997-06-03, URL: ftp://ftp.sas.com/pub/neural/FAQ4.html.

Literaturverzeichnis

Brause, R.: Neuronale Netze - eine Einführung in die Neuroinformatik, Stuttgart: Teubner 1991.

Corsten, H.; May, C.: Anwendungsfelder Neuronaler Netze und ihre Umsetzung, in: Neuronale Netze in der Betriebswirtschaft - Anwendungen in Prognose, Klassifikation und Optimierung, Hrsg.: Corsten, H., May, C., Wiesbaden: Gabler 1996.

DARPA Neural Network Study, Fairfax, VA: AFCEA International Press 1988.

Freeman, J. A.; Skapura, D. M.: Neural Networks - Algorithms, Applications and Programming Techniques, 2., korr. Aufl., Reading, MA et. al.: Addison-Wesley 1992.

Goontilake, S.: Intelligent Systems in Finance and Business: An Overview, in: Intelligent Systems in Finance and Business, Hrsg.: Goontilake, S.; Treleaven, P., Chichester et. al.: John Wiley & Sons 1995.

Haykin, S.: Neural Networks – A Comprehensive Foundation, London et. al.: Prentice-Hall 1994.

Hecht-Nielson, R.: Neurocomputing, Reading, MA et. al.: Addison-Wesley 1990.

Hertz, J.; Krogh, A.; Palmer, R. G.: Introduction to the Theory of Neural Computation, Lecture Notes, Volume I, Santa Fe Institute Studies in the Science of Complexity, Reading, MA et. al.: Addison-Wesley 1991.

Hoffmann, N.: Kleines Handbuch neuronale Netze - anwendungsorientiertes Wissen zum Lernen und Nachschlagen, Braunschweig, et. al.: Vieweg, 1993.

Kemke, C.: Der neuere Konnektionismus, in: Informatik Spektrum, 11/1988.

Klimasauskas, C. C.: Applying Neural Networks, in: Neural Networks in Finance and Investing, Hrsg.: Trippi, R.; Turban, E., Burr Ridge, IL et. al.: Irwin 1993.

Kratzer, K.-P.: Neuronale Netze - Grundlagen und Anwendungen, 2., durchges. Aufl., München et. al.: Hanser, 1993.

Krause, C.: Kreditwürdigkeitsprüfung mit Neuronalen Netzen, Düsseldorf: IDW 1993.

Kurbel, K.; Pietsch, W.: Eine Beurteilung konnektionistischer Modelle auf der Grundlage ausgewählter Anwendungsprobleme und Vorschläge zur Erweiterung, in: WI 5/91.

Maren, A. J.; Harston, C.; Pap, R.: Handbook of Neural Computing Applications, San Diego: Academic Press 1990.

Medsker, L.; Turban, E.; Trippi, R. R.: Neural Networks Fundamentals for Financial Analysts, in: Neural Networks in Finance and Investing, Hrsg.: Trippi, R.; Turban, E., Burr Ridge, IL et. al.: Irwin 1993.

Minsky, M.; Papert, S.: Perceptrons: Expanded Edition, 2., Aufl., Cambridge, MA: MIT Press 1988.

Nauck, D.; Klawonn, F.; Kruse, R.: Neuronale Netze und Fuzzy-Systeme, 2., überarb. und erw. Aufl., Braunschweig, Wiesbaden: Vieweg 1996.

Rehkugler, H.; Poddig, T.: Anwendungsperspektiven und Anwendungsprobleme von Künstlichen Neuronalen Netzwerken, in: Information Management, 2/1992.

Rehkugler, H.; Schmidt-von Rhein, A.: Kreditwürdigkeitsanalyse und -prognose für Privatkundenkredite mittels statistischer Methoden und Künstlicher Neuronaler Netze - Eine empirisch-vergleichende Studie, Otto-Friedrich-Universität Bamberg, Bamberger Betriebswirtschaftliche Beiträge, Nr. 93/1993, 1993.

Ritter, H.; Martinez, T.; Schulten, K.: Neuronale Netze - Eine Einführung in die Neuroinformatik selbstorganisierender Netzwerke, 2., erw. Aufl., Reading, MA et. al.: Addision-Wesley 1991.

Sarle, W.S. (Hrsg.), Neural Network FAQ, part 4 of 7: Books, data, etc., periodic posting to the Usenet newsgroup comp.ai.neural-nets, 1997-06-03, URL: ftp://ftp.sas.com/pub/neural/FAQ.html.

Scherer, A.: Neuronale Netze - Grundlagen und Anwendungen, Braunschweig et. al.: Vieweg 1997.

Schöneburg, E.; Hansen, N.; Gawelczyk, A.: Neuronale Netze, Haar bei München: Markt-u.-Technik 1990.

Schumann, M.; Lohrbach, T. Retzko, R.: Einführung in Aufbau und Arbeitsweise Künstlicher Neuronaler Netze, Georg-August-Universität Göttingen, Abtlg. Wirtschaftsinformatik II, Arbeitspapier Nr. 1, Hrsg.: Schumann, M., Dezember 1991.

Schumann, M.; Lohrbach, T.; Bährs, P.: Versuche zur Kreditwürdigkeitsprüfung mit Künstlichen Neuronalen Netzen, Georg-August-Universität Göttingen, Abtlg. Wirtschaftsinformatik II, Arbeitspapier Nr. 2, Hrsg.: Schumann, M., Januar 1992.

Stahlknecht, P.: Einführung in die Wirtschaftsinformatik, 7., vollst. Überarb. und erw. Aufl., Berlin et. al.: Springer 1995.

Strecker, S., Schwickert, A.: Künstliche Neuronale Netze - Einordnung, Klassifikation und Abgrenzung aus betriebswirtschaftlicher Sicht, in: Arbeitspapiere WI, Nr. 4/97, Hrsg.: Lehrstuhl für Allg. BWL und Wirtschaftsinformatik, Johannes Gutenberg-Universität: Mainz 1997.

Woltering, A.: Künstliche Neuronale Netze - Aufbau, Erstellung, Potentiale und Grenzen, in: Rechnergestützte Werkzeuge für das Management - Grundlagen, Methoden, Anwendungen, Hrsg.: Krallmann, H.; Papke, J.; Rieger, B., Berlin: Erich Schmidt 1992.

Zell, A.: Simulation Neuronaler Netze, 1., unveränderter Nachdruck 1996, Reading, MA et. al.: Addison-Wesley 1994.

Ziegler, U.: Neuronale Netze in betriebswirtschaftlichen Anwendungen - Möglichkeiten und potentielle Vorteile im Vergleich mit anderen entscheidungsunterstützenden Methoden, unveröffentlichte Diplomarbeit, Friedrich-Alexander-Universität Erlangen-Nürnberg, Fachbereich Wirtschafts- und Sozialwissenschaften, 1990.

Bisher erschienen

Stand: Dezember 2000 – Den aktuellen Stand der Reihe erfahren Sie über unsere Web Site unter http//wi.uni-giessen.de

Nr. 1/1996	Grundlagen des Client/Server-Konzepts	Schwickert/Grimbs
Nr. 2/1996	Wettbewerbs- und Organisationsrelevanz des Client/Server-Konzepts	Schwickert/Grimbs
Nr. 3/1996	Realisierungsaspekte des Client/Server-Konzepts	Schwickert/Grimbs
Nr. 4/1996	Der Geschäftsprozeß als formaler Prozeß - Definition, Eigenschaften, Arten	Schwickert/Fischer
Nr. 5/1996	Manuelle und elektronische Vorgangssteuerung	Schwickert/Rey
Nr. 6/1996	Das Internet im Unternehmen - Neue Chancen und Risiken	Schwickert/Ramp
Nr. 7/1996	HTML und Java im World Wide Web	Gröning/Schwickert
Nr. 8/1996	Electronic-Payment-Systeme im Internet	Schwickert/Franke
Nr. 9/1996	Von der Prozeßorientierung zum Workflow-Management - Teil 1: Grundgedanken, Kernelemente, Kritik	Maurer
Nr. 10/1996	Von der Prozeßorientierung zum Workflow- Management - Teil 2: Prozeßmanagement und Workfflow	Maurer
Nr. 11/1996	Informationelle Unhygiene im Internet	Schwickert/Dietrich/Klein
Nr. 12/1996	Towards the theory of Virtual Organisations: A description of their formation and figure	Appel/Behr
Nr. 1/1997	Der Wandel von der DV-Abteilung zum IT-Profitcenter: Mehr als eine Umorganisation	Kargl
Nr. 2/1997	Der Online-Markt - Abgrenzung, Bestandteile, Kenngrößen	Schwickert/Pörtner
Nr. 3/1997	Netzwerkmanagement, OSI Framework und Internet SNMP	Klein/Schwickert
Nr. 4/1997	Künstliche Neuronale Netze - Einordnung, Klassifikation und Abgrenzung aus betriebswirtschaftlicher Sicht	Strecker/Schwickert
Nr. 5/1997	Sachzielintegration bei Prozeßgestaltungsmaßnahmen	Delnef
Nr. 6/1997	HTML, Java, ActiveX - Strukturen und Zusammenhänge	Schwickert/Dandl
Nr. 7/1997	Lotus Notes als Plattform für die Informationsversorgung von Beratungsunternehmen	Appel/Schwaab
Nr. 8/1997	Web Site Engineering - Modelltheoretische und methodische Erfahrungen aus der Praxis	Schwickert
Nr. 9/1997	Kritische Anmerkungen zur Prozeßorientierung	Maurer/Schwickert
Nr. 10/1997	Künstliche Neuronale Netze - Aufbau und Funktionsweise	Strecker
Nr. 11/1997	Workflow-Management-Systeme in virtuellen Unternehmen	Maurer/Schramke
Nr. 12/1997	CORBA-basierte Workflow-Architekturen - Die objektorientierte Kernanwendung der Bausparkasse Mainz AG	Maurer
Nr. 1/1998	Ökonomische Analyse Elektronischer Märkte	Steyer
Nr. 2/1998	Demokratiepolitische Potentiale des Internet in Deutschland	Muzic/Schwickert
Nr. 3/1998	Geschäftsprozeß- und Funktionsorientierung - Ein Vergleich (Teil 1)	Delnef
Nr. 4/1998	Geschäftsprozeß- und Funktionsorientierung - Ein Vergleich (Teil 2)	Delnef
Nr. 5/1998	Betriebswirtschaftlich-organisatorische Aspekte der Telearbeit	Polak
Nr. 6/1998	Das Controlling des Outsourcings von IV-Leistungen	Jäger-Goy
Nr. 7/1998	Eine kritische Beurteilung des Outsourcings von IV-Leistungen	Jäger-Goy
Nr. 8/1998	Online-Monitoring - Gewinnung und Verwertung von Online-Daten	Guba/Gebert
Nr. 9/1998	GUI - Graphical User Interface	Maul
Nr. 10/1998	Institutionenökonomische Grundlagen und Implikationen für Electronic Business	Schwickert
Nr. 11/1998	Zur Charakterisierung des Konstrukts “Web Site”	Schwickert
Nr. 12/1998	Web Site Engineering - Ein Komponentenmodell	Schwickert
Nr. 1/1999	Requirements Engineering im Web Site Engineering – Einordnung und Grundlagen	Schwickert/Wild
Nr. 2/1999	Electronic Commerce auf lokalen Märkten	Schwickert/Lüders
Nr. 3/1999	Intranet-basiertes Workgroup Computing	Kunow/Schwickert
Nr. 4/1999	Web-Portale: Stand und Entwicklungstendenzen	Schumacher/Schwickert
Nr. 5/1999	Web Site Security	Schwickert/Häusler
Nr. 6/1999	Wissensmanagement - Grundlagen und IT-Instrumentarium	Gaßen
Nr. 7/1999	Web Site Controlling	Schwickert/Beiser
Nr. 8/1999	Web Site Promotion	Schwickert/Arnold
Nr. 9/1999	Dokumenten-Management-Systeme – Eine Einführung	Dandl
Nr. 10/1999	Sicherheit von eBusiness-Anwendungen – Eine Fallstudie	Harper/Schwickert
Nr. 11/1999	Innovative Führungsinstrumente für die Informationsverarbeitung	Jäger-Goy
Nr. 12/1999	Objektorientierte Prozeßmodellierung mit der UML und EPK	Dandl
Nr. 1/2000	Total Cost of Ownership (TCO) – Ein Überblick	Wild/Herges
Nr. 2/2000	Implikationen des Einsatzes der eXtensible Markup Language – Teil 1: XML-Grundlagen	Franke/Sulzbach
Nr. 3/2000	Implikationen des Einsatzes der eXtensible Markup Language – Teil 2: Der Einsatz im Unternehmen	Franke/Sulzbach
Nr. 4/2000	Web-Site-spezifisches Requirements Engineering – Ein Formalisierungsansatz	Wild/Schwickert
Nr. 5/2000	Elektronische Marktplätze – Formen, Beteiligte, Zutrittsbarrieren	Schwickert/Pfeiffer
Nr. 6/2000	Web Site Monitoring – Teil 1: Einordnung, Handlungsebenen, Adressaten	Schwickert/Wendt
Nr. 7/2000	Web Site Monitoring – Teil 2: Datenquellen, Web-Logfile-Analyse, Logfile-Analyzer	Schwickert/Wendt
Nr. 8/2000	Controlling-Kennzahlen für Web Sites	Schwickert/Wendt
Nr. 9/2000	eUniversity – Web-Site-Generierung und Content Management für Hochschuleinrichtungen	Schwickert/Ostheimer/Franke

Bestellung (bitte kopieren, ausfüllen, zusenden/zufaxen)

Adressat:

Professur für BWL und Wirtschaftsinformatik
Fachbereich Wirtschaftswissenschaften
Licher Straße 70
D – 35394 Gießen
Telefax: (0 641) 99-22619

Hiermit bestelle ich gegen Rechnung die angegebenen Arbeitspapiere zu einem Kostenbeitrag von DM 10,- pro Exemplar (MwSt. entfällt) zzgl. DM 5,- Versandkosten pro Sendung.

Nr.	An
1/1996	
2/1996	
3/1996	
4/1996	
5/1996	
6/1996	
7/1996	
8/1996	
9/1996	
10/1996	
11/1996	
12/1996	

Nr.	An
1/1997	
2/1997	
3/1997	
4/1997	
5/1997	
6/1997	
7/1997	
8/1997	
9/1997	
10/1997	
11/1997	
12/1997	

Nr.	Anz
1/1998	
2/1998	
3/1998	
4/1998	
5/1998	
6/1998	
7/1998	
8/1998	
9/1998	
10/1998	
11/1998	
12/1998	

Nr.	Anz
1/1999	
2/1999	
3/1999	
4/1999	
5/1999	
6/1999	
7/1999	
8/1999	
9/1999	
10/1999	
11/1999	
12/1999	

Nr.	Anz
1/2000	
2/2000	
3/2000	
4/2000	
5/2000	
6/2000	
7/2000	
8/2000	
9/2000	

Absender:

_____
Organisation

_____
Abteilung

_____
Nachname, Vorname

_____
Straße

_____
Plz/Ort

_____
Telefon Telefax eMail

_____
Ort, Datum Unterschrift