Einführung in Neuronale Netze
Prinzipien Künstlicher Neuronaler Netze

Modellierung von Neuronen

Ein künstliches neuronales Netz besteht aus stark idealisierten Neuronen. Wie ihr biologisches Vorbild bestehen sie aus drei Komponenten: dem Zellkörper, den Dendriten und einem Axon. Die Dendriten summieren die Eingabe des Netzes in die Zelle auf, das Axon leitet die Ausgabe der Zelle an die Dendriten nachfolgender Synapsen weiter.

Die Stärke der Synapsen wird durch einen numerischen Wert, das Verbindungsgewicht, dargestellt. Daher läßt sich die Verbindung zwischen Neuronen als direkte gewichtete Verbindung zwischen den beiden Zellen i und j darstellen.

Als nächstes soll die Frage beantwortet werden, aus welchen Komponenten neuronale Netze bestehen.

Komponente		Eigenschaften und Aufgaben
Zellen (Neuronen)		
	Aktivierungszustand Z	gibt den Aktivierungsgrad der Zelle an
	Aktivierungsfunktion f_{act}	gibt an, wie sich ein neuer Aktivierungszustand $Z_{neu,j}$ des Neurons j aus der alten Aktivierung $Z_{alt,j}$ und der Netzeingabe $net_j(t)$ sowie dem Schwellenwert des Neurons j ergibt.
	Ausgabefunktion f_{out}	bestimmt aus der Aktivierung die Ausgabe des Neurons
Verbindungsnetzwerk der Zellen		Ein neuronales Netz kann als gerichteter, gewichteter Graph angesehen werden. Die Kanten stellen die Verbindungen zwischen den Neuronen dar. $w_{i,j}$ ist das Gewicht (weight) der Verbindung von Neuron i nach

		Neuron j, die Matrix W aller Verbindungen heißt Gewichtsmatrix.
Propagierungsfunktion		gibt an, wie sich die Netzeingabe eines Neurons aus den Ausgaben der anderen Neuronen und den Verbindungsgewichten berechnet. Dabei handelt es sich um die gewichtete Summe der Ausgaben der Vorgängerzellen.
Lernregel		Algorithmus, nach dem das Netz lernt, für eine vorgegebene Eingabe eine gewünschte Ausgabe zu produzieren. Durch die wiederholte Eingabe von Trainingsmustern wird die Stärke der Verbindungen zwischen den Neuronen modifiziert. Dabei wird versucht, den Fehler zwischen erwarteter und tatsächlicher Ausgabe des Netzes zu minimieren. Lernverfahren sind die interessanteste Komponente der neuronalen Netze.

Formal kann man ein einzelnes Neuron folgendermaßen beschreiben:

Definition:
Ein künstliches Neuron *ist ein Tupel (x, w, f_a, f_o, o) bestehend aus einem Eingabevektor x = (x_1, ..., x_n), einem Gewichtsvektor w = (w_1, ..., w_n), einer Aktivierungsfunktion f_a mit f_a: IR^n x IR^n ---> IR und einer Ausgabefunktion f_o, für die f_o: IR --> IR gilt. Dabei wird durch $f_o(f_a(x, w))=o$ der Ausgabewert des Neurons erzeugt, der an die nachfolgenden Neuronen über die Axonkollaterale weitergeleitet wird.*

Faßt man Aktivierungs- und Ausgabefunktion zu einer Transfer-Funktion zusammen, verwendet man häufig die gewichtete Summe:

$$f_a(\vec{x}, \vec{w}) = \sum_{k=i}^{n} x_i w_i$$

Außerdem kann man dem einzelnen Neuron, analog zur biologischen Nervenzelle, einen inneren Aktivierungszustand zuordnen. Die Definiton für ein Neuron kann dann auch so aussehen:

Definition:
Ein künstliches Neuron *ist ein Tupel (x, w, f_a, Z, f_o, o) bestehend aus:*

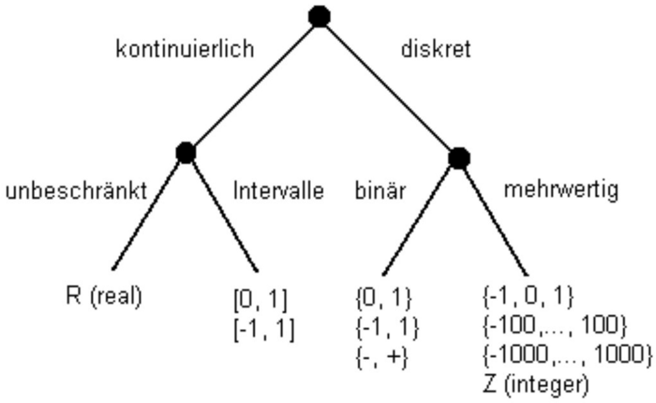
1. Eingabevektor x
2. Gewichtsvektor w
3. Aktivierungsfunktion f_a mit f_a: IR^n x IR^n ---> IR
4. Zustand Z
5. Ausgabefunktion f_o: IR --> IR

Hierbei ist die Ausgabefunktion f_o eine Funktion in Abhängigkeit von Z.

Der Aktivierungszustand Z hängt vom alten Zustand und der Veränderung der Aktivierungsfunktion ab und kann beispielsweise durch

$$Z^{neu} = Z^{alt} + f_a(\vec{x}, \vec{w})$$

ausgedrückt werden.
In der technischen Realisierung wird er unterschiedlich dargestellt. Man unterscheidet (quasi-)kontinuierliche und diskrete Wertebereiche, die sich wiederum in unterschiedliche Modelle aufteilen lassen:

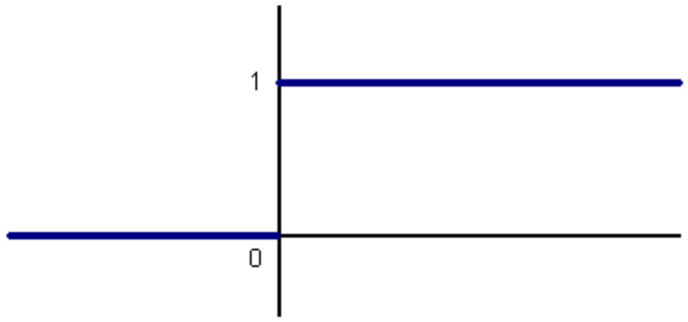


Bei kontinuierlichen Wertebereichen beschränken die meisten Modelle den Aktivierungszustand auf ein Intervall. Das liegt daran, daß es sich meistens um eine nichtlineare, häufig sigmoide, Aktivierungsfunktion und die Identität als Ausgabefunktion handelt. Dadurch wird die Ausgabe identisch mit der Aktivierung, und der Wertebereich der Aktivierungsfunktion gibt den Wertebereich des Aktivierungszustandes an.

Das ursprüngliche Hopfield-Modell und andere Modelle verwenden diskrete Aktivierungszustände. Diese werden dann auch in der Implementierung als binäre Werte gespeichert und verarbeitet.
Damit eine biologische Nervenzelle "feuert", also ein Aktionspotential ausgelöst wird, muß ein bestimmter Schwellenwert S überschritten werden.

Auch für künstliche Neuronen gibt es eine Schwellenwertfunktion:

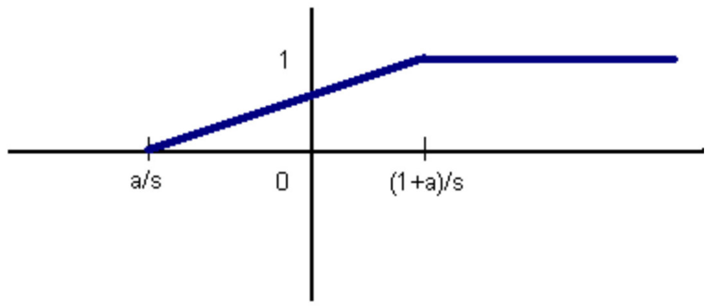
$$f_o\left(\sum_{k=1}^{n} x_k w_k\right) = \begin{cases} 1: falls \sum_{k=1}^{n} x_k w_k \geq S \\ 0: sonst \end{cases}$$



Da diese Art der Ausgabefunktion nicht die Intensität der aufeinanderfolgenden Aktionspotentiale eines biologisches Neurons simuliert, werden lineare Ausgabefunktionen verwendet. Der zeitliche Abstand, in dem die Aktionspotentiale durch die biologische Nervenzelle weitergereicht werden, ist nach unten beschränkt. Daher sollte auch im künstlichen Neuronenmodell eine beschränkte Ausgabefunktion Verwendung finden.

Diese Ausgabefunktionen lassen sich durch semilineare Funktionen der Art

$$f_o\left(\sum_{k=1}^{n} x_k w_k\right) = \begin{cases} 1 & : falls \sum_{k=1}^{n} x_k w_k \geq \frac{1+a}{s} \\ s\left(\sum_{k=1}^{n} x_k w_k\right) - a & : falls \frac{a}{s} \leq \sum_{k=1}^{n} x_k w_k < \frac{1+a}{s} \\ 0 & : sonst \end{cases}$$



beschreiben.

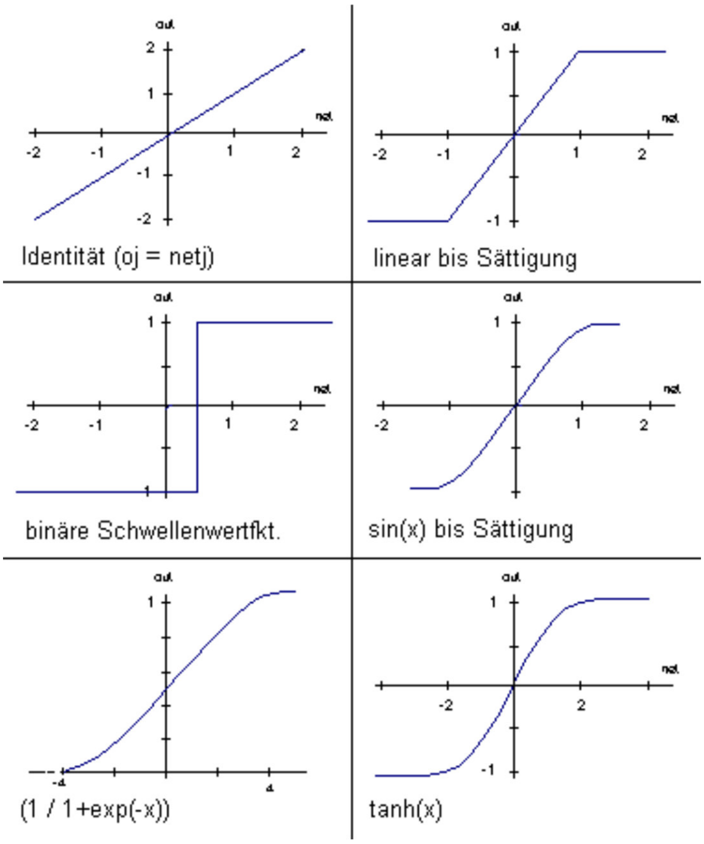
Sinnvoller ist es aber, die Aktivierung beziehungsweise die Ausgabe durch glattere, also differenzierbare Funktionen zu beschreiben. Ein Beispiel für differenzierbare und beschränkte Funktionen sind die s-förmigen oder sigmoiden Funktionen:

Definition:
Eine Funktion s_c: IR --> [0, 1] heißt ***sigmoide oder s-förmige Funktion****, wenn sie monoton wachsend und differenzierbar ist und wenn*

$$\lim_{\lambda \to \infty} s_c(\lambda) = k_1 \text{ und } \lim_{\lambda \to -\infty} s_c(\lambda) = k_2 \text{ mit } k_2 < k_1$$

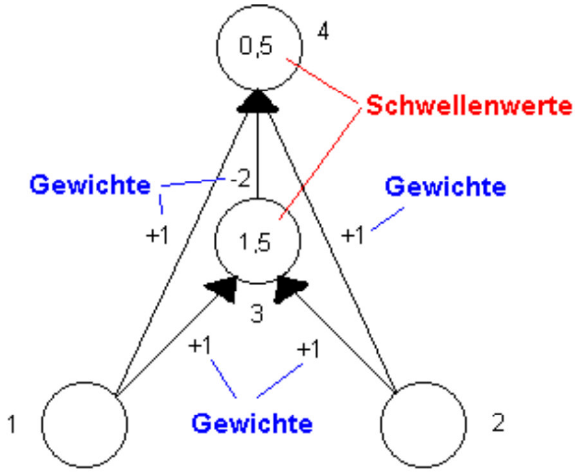
gelten.

Die folgende Abbildung enthält einige gebräuchliche Aktivierungs- bzw. Ausgabefunktionen, wobei die beiden letzteren am häufigsten verwendet werden.



Modellierung von Neuronen
1. Beispiel eines Netzes: XOR-Netz mit 4 Zellen

Als typisches Beispiel für ein kleines neuronales Netz soll hier das XOR-Netz mit 4 Zellen vorgestellt werden. Dieses neuronale Netz kann leicht durchgerechnet werden. Es benötigt mindestens eine Schicht verdeckter Neuronen, um zu funktionieren.



XOR-Netz mit 4 Zellen

Für die Aktivierungen der Neuronen werden lediglich die binären Werte 0 und 1 verwendet. Die Netzeingabe wird durch folgende Propagierungsfunktion berechnet:

$$net_j(t) = \sum_i o_i(t) w_{ij}$$

Als Aktivierungsfunktion wird eine binäre Schwellenwertfunktion benutzt.

$$a_j(t) = \begin{cases} 1 : \text{falls } net_j(t) \geq S_j \\ 0 : \text{sonst} \end{cases}$$

Die Ausgabefunktion der Neuronen ist die Identität.

Anhand der Tabelle läßt sich erkennen, daß für alle möglichen Eingaben jeweils der Wert als Ausgabe von Neuron 4 berechnet wird, der einem binären XOR entspricht.

Ausgabe Neuron1: o_1	**Ausgabe Neuron2: o_2**	**Eingabe Neuron3: net_3**	**Schwellenwert Neuron3: S_3**	**Ausgabe Neuron3: o_3**	**Eingabe Neuron4: net_4**	**Schwellenwert Neuron4: S_4**	**Ausgabe Neuron4: o_4**	**XOR**
0	0	net_3 = 0*1+0*1 = 0	1,5	0	net_4 = 0*1+0*1+0(-2) = 0	0,5	0	0
0	1	net_3 = 0*1+1*1 = 1	1,5	0	net_4 = 0*1+1*1+0(-2) = 1	0,5	1	1
1	0	net_3 = 1*1+0*1 = 1	1,5	0	net_4 = 1*1+0*1+0(-2) = 1	0,5	1	1
1	1	net_3 = 1*1+1*1 = 2	1,5	1	net_4 = 1*1+1*1+1(-2) = 0	0,5	0	0

Modellierung von Neuronen
2. Vergleich von biologischen und künstlichen Neuronen

Die Neuronen künstlicher neuronaler Netze sind im Vergleich zu ihren natürlichen Vorbildern sehr stark idealisiert. Trotzdem gibt es viele **Gemeinsamkeiten:**

- Die massive Parallelität der Neuronen

- Relativ einfache Elemente: Neuronen verarbeiten die Aktivierungen der Vorgängerneuronen und die Stärke der Verbindung zu einer Ausgabe.

- Die Neuronen sind durch gewichtete Verbindungen (biologisch: Synapsen) miteinander verbunden.

- Die Verbindungsgewichte bei künstlichen Neuronen sind modifizierbar. Das entspricht der Plastizität der Synapsen beim biologischen Vorbild.

- Ein Neuron ist mit sehr vielen anderen Neuronen verbunden (hohe Konnektivität).

Durch die vereinfachte Modellierung der Neuronen kommt es zu folgenden **Unterschieden** zwischen biologischen und künstlichen Neuronen:

Künstliches neuronales Netz	**Biologisches Vorbild**
- viel geringere Anzahl der Neuronen (10^2 - 10^4)	- ca. 10^{11} Neuronen
- viel geringere Anzahl von Verbindungen	- höhere Anzahl an Verbindungen zwischen den Neuronen
- Stärke einer Synapse wird ausschließlich durch das Gewicht bestimmt	- Einfluß verschiedener Neurotransmitter auf die Stärke einer Synapse
- numerischer Aktivierungswert (Amplitudenmodulation)	- impulscodierte Informationsübertragung (Frequenzmodulation)
- zeitliche Vorgänge der Nervenleitung werden vernachlässigt	- verzögerte Aktivierung

Sicherlich gibt es noch weitere Unterschiede zwischen künstlichen und biologischen Neuronen. Da jedoch Anpassungen an das biologische Vorbild wesentlich erhöhte Simulationsanforderungen mit sich bringen und sich viele Probleme mit den relativ biologiefernen künstlichen neuronalen Netzen sehr gut lösen lassen, haben biologisch adäquatere Simulationen noch keine Vorteile gezeigt.

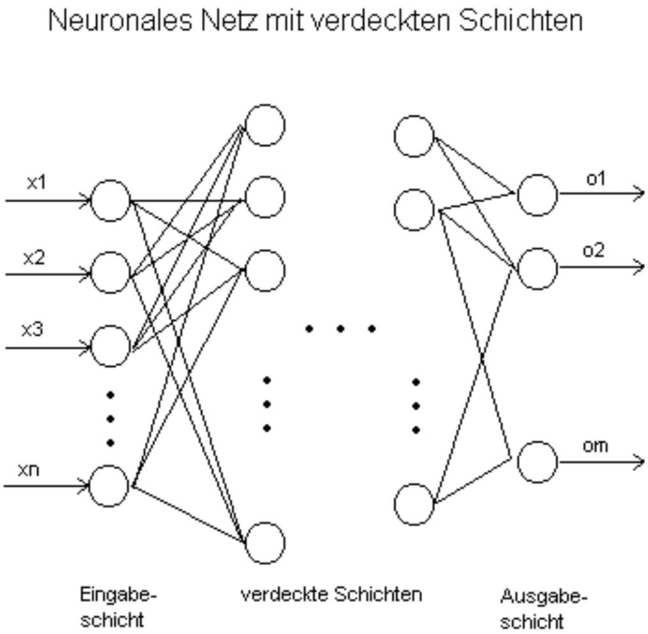
Modellierung Künstlicher Neuronaler Netze

Verbindet man nun mehrere Neuronen miteinander, so erhält man ein neuronales Netz.

Definition:
Ein Neuronales Netz *ist ein Paar (N, V) mit einer Menge N von Neuronen und einer Menge V von Verbindungen. Es besitzt die Struktur eines gerichteten Graphen, für den die folgenden Einschränkungen und Zusätze gelten:*

i. *Die Knoten des Graphen heißen Neuronen.*
ii. *Die Kanten heißen Verbindungen.*
iii. *Jedes Neuron kann eine beliebige Menge von Verbindungen empfangen, über die es seine Eingabe erhält.*
iv. *Jedes Neuron kann genau eine Ausgabe über eine beliebige Menge von Verbindungen aussenden.*

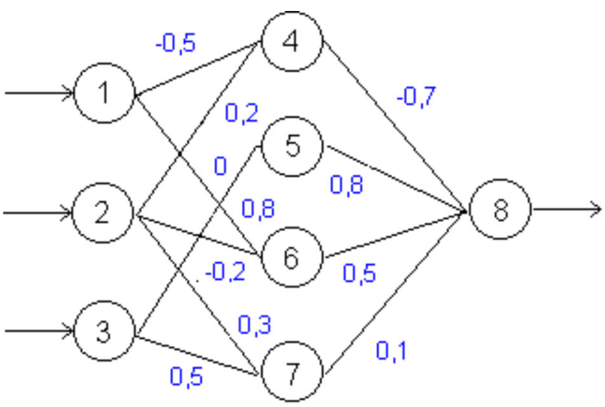
v. *Das Neuronale Netz erhält aus Verbindungen, die der "Außenwelt" entspringen, Eingaben und gibt seine Ausgaben über in der "Außenwelt" endende Verbindungen ab.*



Alle Verbindungen, die von anderen Neuronen zu einem einzelnen Neuron *j* gehen, ergeben den Eingabevektor x_j von *j*. Da bei den meisten neuronalen Netzen die Eingabe gewichtet wird, kann man die Verbindungsstruktur (Topologie) in Form einer Matrix beschreiben. Zeilen und Spalten identifiziert man mit den Neuronen; in den Kreuzungspunkt schreibt man das Gewicht der Verbindung.
In der Schreibweise $W = [w_{i,j}]$ *gilt dann:*

- $w_{i,j} = 0$: keine Verbindung von Neuron *i* zu Neuron *j*
- $w_{i,j} < 0$: hemmende Verbindung der Stärke $|w_{i,j}|$
- $w_{i,j} > 0$: anregende Verbindung der Stärke $|w_{i,j}|$

Dieses neuronale Netz kann in seiner Verbindungsstruktur dargestellt werden durch:



	1	2	3	4	5	6	7	8
1				-0,5		0,8		
2				0,2		-0,2	0,3	
3					0		0,5	
4								-0,7
5								0,8
6								0,5
7								0,1
8								

Alle freien Felder werden mit Null besetzt.

Modellierung Künstlicher Neuronaler Netze
1. Topologien

Neuronale Netze lassen sich gemäß der folgenden Topologien klassifizieren:

1. Netze ohne Rückkopplung (feedforward-Netze)
2. Netze mit Rückkopplungen (rekurrente Netze)

1. Netze ohne Rückkopplung (feedforward-Netze)

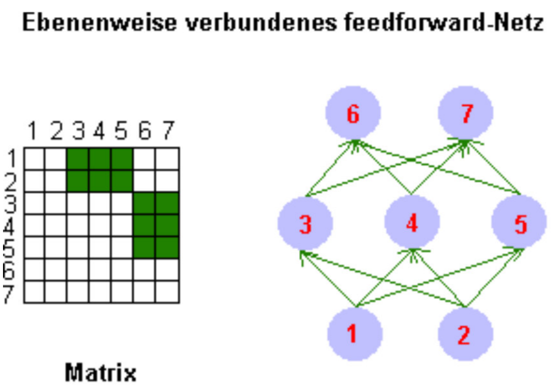
Bei Netzen ohne Rückkopplungen existiert kein Pfad, der von einem Neuron direkt oder über zwischengeschaltete Neuronen wieder zurück zu diesem Neuron führt. Daten werden also nur in eine Richtung weitergegeben.

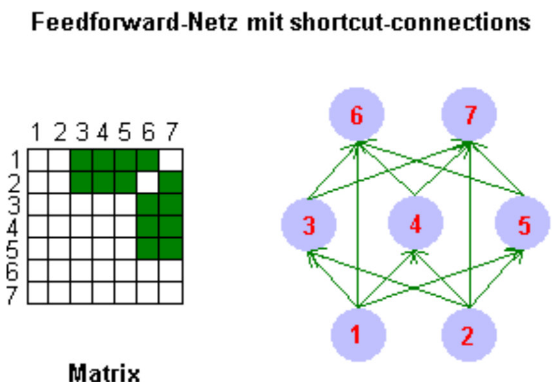
Mathematisch ist diese Topologie ein azyklischer Graph. In der Matrixdarstellung ist nur die obere Dreiecksmatrix mit Werten ungleich Null besetzt, da kein Neuron eine Verbindung zu einem dichter an der Eingabeschicht liegenden Neuron haben kann.

Man unterscheidet außerdem ebenenweise verbundene und allgemeine feedforward-Netze.

Ebenenweise verbundene feedforward-Netze sind in mehrere Schichten eingeteilt, wobei es nur Verbindungen von einer Schicht zur nächsten gibt. Man spricht von vollständig verbundenen Netzen, falls jedes Neuron der Schicht U_i mit jedem Neuron der darauffolgenden Schicht U_{i+1} verbunden ist.

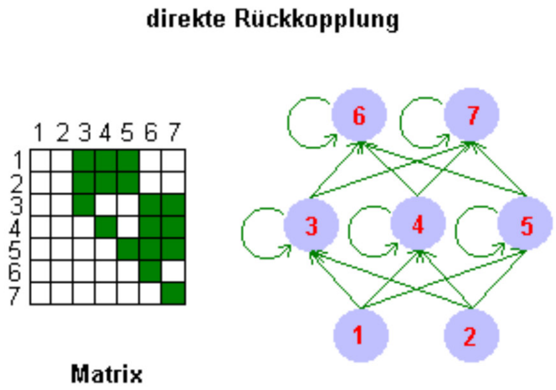
Allgemeine feedforward-Netze besitzen dagegen auch sogenannte shortcut connections, also Verbindungen zwischen Neuronen, die Ebenen überspringen.





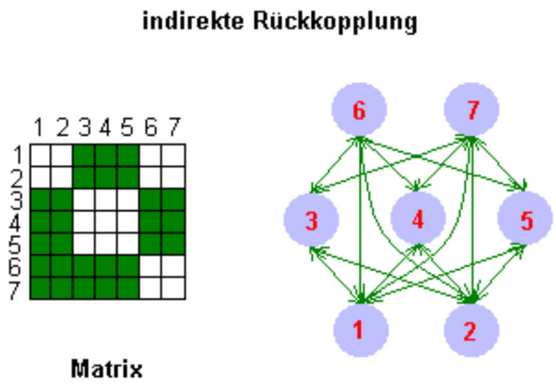
2. Netze mit Rückkopplungen (rekurrente Netze)

Netze mit Rückkopplungen unterteilt man meist folgendermaßen:



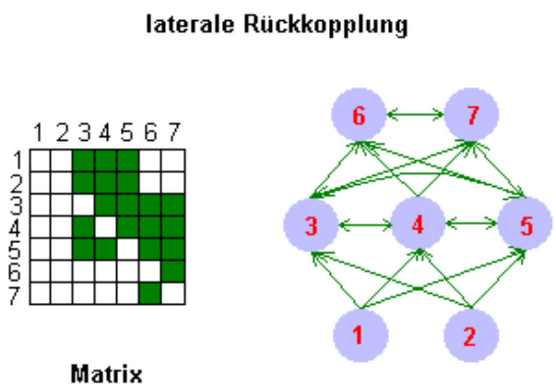
1. Netze mit direkten Rückkopplungen *(direct feedback)*

Die Neuronen haben eine Verbindung von ihrer Ausgabe zurück zur Eingabe und können dadurch ihre eigene Aktivierung verstärken oder abschwächen. Diese Verbindungen bewirken oft, daß Neuronen die Grenzzustände ihrer Aktivierungen annehmen, weil sie sich selbst verstärken oder hemmen.



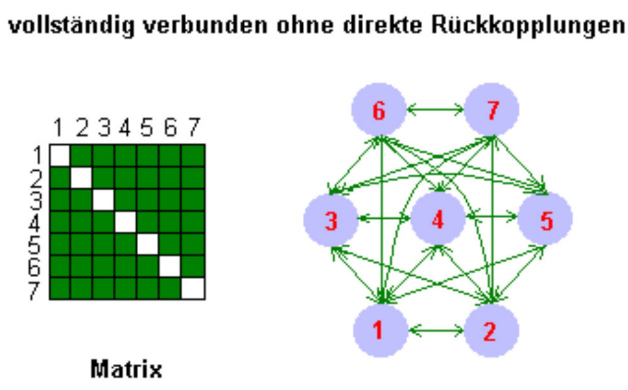
2. Netze mit indirekten Rückkopplungen *(indirect feedback)*

Diese Netze besitzen Rückkopplungen von Neuronen höherer Ebenen zu Neuronen niederer Ebenen. Dadurch erreicht man eine Aufmerksamkeitssteuerung auf bestimmte Bereiche von Eingabeneuronen oder auf bestimmte Eingabemerkmale durch das Netz.



3. Netze mit Rückkopplungen innerhalb einer Schicht *(lateral feedback)*

Netze mit Rückkopplungen innerhalb derselben Schicht werden oft für Aufgaben eingesetzt, bei denen nur ein Neuron einer Gruppe aktiv werden soll. Jedes Neuron hat dann hemmende Verbindungen zu den anderen Neuronen und oft noch eine aktivierende direkte Rückkopplung zu sich selbst. Das Neuron mit der stärksten Aktivierung, der Gewinner, hemmt dann die anderen Neuronen. Daher heißt eine solche Topologie auch *winner-takes-all-* Netzwerk.



4. vollständig verbundene Netze
Vollständig verbundene Netze haben Verbindungen zwischen allen Neuronen. Sie sind insbesondere als Hopfield- Netze bekannt geworden. Bei diesen muß allerdings auch die Verbindungsmatrix symmetrisch sein und die Diagonale darf nur Nullen enthalten.

Netze mit Rückkopplungen werden auch eingesetzt, um Zeitabhängigkeiten bei Daten, wie z.B. die Struktur einer Schwingung, modellieren zu können. Über die Rückkopplung erhält man als Netzeingabe nicht nur die neuen Daten, sondern auch wieder die bereits verarbeiteten alten Daten.

Modellierung Künstlicher Neuronaler Netze
2. Modellierung des Lernens

Ein neuronales Netz "lernt", indem es sich gemäß einer fest vorgegebenen Vorschrift, der Lernregel, selbst modifiziert. Prinzipiell kann der Lernprozeß bestehen aus:

- Entwicklung neuer Verbindungen
- Löschen existierender Verbindungen
- Modifikation der Verbindungsstärke (Veränderung der Gewichte)
- Modifikation des Schwellenwertes
- Modifikation der Aktivierungs- bzw. Ausgabefunktion
- Entwicklung neuer Zellen
- Löschen bestehender Zellen

Von diesen Möglichkeiten wird die dritte, also das Lernen durch Veränderung der Gewichte, am häufigsten verwendet. Erst in letzter Zeit haben Verfahren, die auch eine Veränderung der Topologie beinhalten an Bedeutung gewonnen.

Eine weitere Unterscheidungsmöglichkeit besteht in der Art des verwendeten Lernparadigmas. Hier lassen sich drei Arten unterscheiden:

1. Überwachtes Lernen *(supervised learning)*
Beim überwachten Lernen gibt ein externer Lehrer dem Netz zu jeder Eingabe die korrekte Ausgabe oder die Differenz der tatsächlichen zur korrekten Ausgabe an. Anhand dieser Differenz wird dann das Netz über die Lernregel modifiziert. Diese Technik setzt allerdings voraus, daß Trainingsdaten

2. Bestärkendes Lernen *(reinforcement learning)*
Im Gegensatz zum überwachten Lernen wird dem Netz hier lediglich mitgeteilt, ob seine Ausgabe korrekt

3. Unüberwachtes Lernen *(unsupervised learning)*
Hierbei gibt es überhaupt keinen externen Lehrer, daher heißt dieses Lernparadigma auch *self-organized learning*. Das Netz versucht ohne Beeinflussung von außen die

exisitieren, die aus Paaren von Ein- und Ausgabedaten bestehen.
Ein typisches überwachtes Lernverfahren wie z.B. Backpropagation durchläuft für alle Paare von Ein- und Ausgabemustern folgende Schritte:

1. Das Eingabemuster wird dem Netz durch entsprechende Aktivierung der Eingabeneuronen präsentiert.
2. Die angelegte Eingabe läuft vorwärts durch das Netz. Dadurch wird ein Ausgabemuster für die aktuelle Eingabe erzeugt.
3. Tatsächliche und korrekte Ausgabe werden verglichen und die Differenz berechnet.
4. Die Fehler laufen rückwärts von der Ausgabe- zur Eingabeschicht. Dabei werden die Verbindungsgewichte verändert, so daß der Fehler verringert wird.
5. Die Gewichte aller Neuronen werden um die vorher berechneten Werte verändert.

oder inkorrekt war. Das Netz erfährt nicht den exakten Wert des Unterschiedes.

präsentierten Daten in Ähnlichkeitsklassen aufzuteilen.

Übungsaufgaben

1. Versuchen Sie mit Hilfe der Simulation A eines Netzes mit 3 Neuronen die Schwellenwerte und Gewichte derart zu ändern, daß sich die logischen (booleschen) Funktionen AND und OR darstellen lassen.
2. Versuchen Sie nun, mit dem in Aufgabe 1 benutzten Applet die XOR-Funktion zu realisieren. Gelingt das?
3. Realisieren Sie die XOR-Funktion mit Hilfe des Netzes B mit 4 Neuronen. Wie müssen die Schwellenwerte und Gewichte eingestellt werden?

A.

B.

Zurück zum letzten Kapitel Zum nächsten Kapitel