Kapitel 11

Neuronale Netze

11.1 Einführung

Neuronale Netze sind der Versuch, ein System mit einer Funktionsweise aufzubauen, die in Teilaspekten der des menschlichen Gehirns ähnelt und auch für ähnliche Aufgaben geeignet ist. Ganz allgemein gesprochen sind sie Systeme aus miteinander verbundenen, zur Informationsverarbeitung fähigen Elementen, die Neuronen genannt werden. Diese Bezeichnung ist der Biologie entliehen, da im Gehirn die Neuronen (Nervenzellen) die informationsverarbeitenden Elemente sind. In einem künstlichen neuronalen Netz sind diese Neuronen als physikalische oder mathematische Modelle mit mehreren Ein- und Ausgängen realisiert, deren mathematisches Verhalten im Prinzip den biologischen Neuronen entspricht. Die wichtigsten Unterschiede zu den "normalen" Computern erkennt man, wenn man einen Vergleich zwischen den typischen Funktionsweisen von Computern und Gehirn zieht:

	Computer	Gehirn
Arbeitsprinzip	sequentiell	parallel
Berechnungen	digital/exakt	mit Fehlertoleranz/näherungsweise
Datenspeicherung	lokal	über das Netzwerk verteilt

Besonders interessant dabei ist die Möglichkeit, vom Prinzip her parallele Rechner mit hoher Rechenleistung bauen zu können.

Wir zeigen in Abb. 11.1 den Aufbau einer sogenannten Pyramidenzelle, deren Funktionsweise für Neuronen typisch ist, die aber nur einen von vielen Zelltypen im Gehirn darstellt. Die reizempfindlichen Bestandteile der Zelle sind der Zellkörper oder *Soma* und die *Dendriten*, lange, verzweigte Fortsätze des Zellkörpers. Diese stellen sozusagen die Eingangsleitungen des Neurons dar. Ein Ausgangssignal leitet die Zelle durch das *Axon* ab, welches

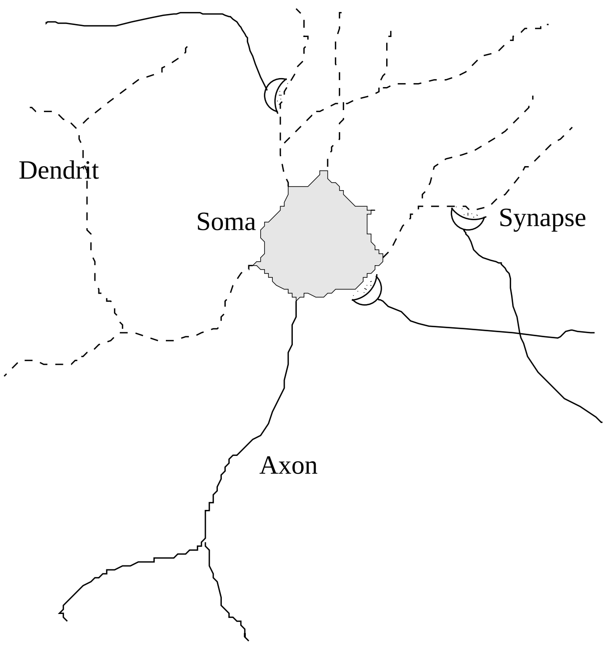


Abbildung 11.1: Struktur einer Pyramidenzelle. Dem Zellkörper (Soma) leiten Dendriten (gestrichelt) Reize zu, während Axonen (durchgezogen) den Erregungszustand vom Soma weg an andere Nerven weiterleiten. Der Kontakt zu anderen Nerven geschieht über Synapsen, in denen Erregungspotentiale die Auschüttung chemischer Botenstoffe bewirken. Diese lösen ihrerseits auf der dendritischen Seite elektrochemische Vorgänge aus, die das elektrochemische Potential lokal ändern. Diese lokale Störung wandert im Dendriten zum Soma, auf dem ebenfalls direkt Synapsen enden können. Die Nervenzelle feuert, wenn die "Summe" der einlaufenden Signale einen bestimmten Schwellwert überschreitet.

sich ebenfalls verzeigen kann und Verbindungen zu bis etwa 1000 anderen Nervenzellen herstellt.

Die Verbindung zwischen Axon einer Zelle und Dendrit oder Soma einer anderen Zelle wird durch Synapsen hergestellt. Je nach Typ der Synapse und Erregungszustand des Axons werden chemische Botenstoffe ausgeschüttet, die exzitatorische oder inhibitorische Wirkung auf die Zelle ausüben können. Diese äußert sich in einer lokalen Änderung der elektrochemischen Eigenschaften der Zellwand, die letztlich in einer lokalen Spannungsänderung resultiert, die im Dendriten propagieren kann. Bei mehreren aktiven Synapsen addiert (exhibitorische Synapse) oder subrahiert (inhibitorische Synapse) der Dendrit im wesentlichen die hereinkommenden Signale und leitet diese Information an das Soma weiter. Überschreitet dann das dortige Signal einen gewissen Schwellwert, so *feuert* das Neuron und gibt über das Axon ein Signal aus.

Die chemischen Eigenschaften der Synapsen können sich mit der Zeit abhängig von den Zuständen von Dendrit und Axon auf mehreren Zeitskalen ändern (*short term/long term potentiation*). Vereinfacht gesprochen, können Synapsen die einlaufenden Signale gewichtet an den Dendriten weitergeben. Dies ermöglicht die Speicherung von Informationen im

Netz durch die Eigenschaften der synaptischen Kontakte.

Ein solcher Lernmechanismus, dessen experimentelle Details allerdings weiterhin umstritten sind, wurde bereits 1949 durch den Psychologen *Hebb* in seinem Werk *Organization of Behaviour* als Hypothese verwendet. Diese *Hebbsche Hypothese* besagt, daß sich das synaptische Gewicht proportional zur korrelierten Aktivität vor und hinter der Synapse ändert. Das bedeutet, daß Synpasen, die aktiv waren und damit ein Feuern des Neurons bewirkt haben, ihre synaptischen Gewichte verstärken/erhöhen.

11.2 Modellneuronen

Während für die detaillierte Untersuchung von Einzelneuronen ein genaues Modell der Reizleitung und des Zeitverhaltens der Neuronen nötig ist, reichen für den Aufbau von Modellnetzen und der Untersuchung ihrer prinzipiellen Eigenschaften vereinfachte Neuronenmodelle aus. Eine Möglichkeit ist in Abb. 11.2 gezeigt: Über die Eingänge $x_1, ..., x_L$

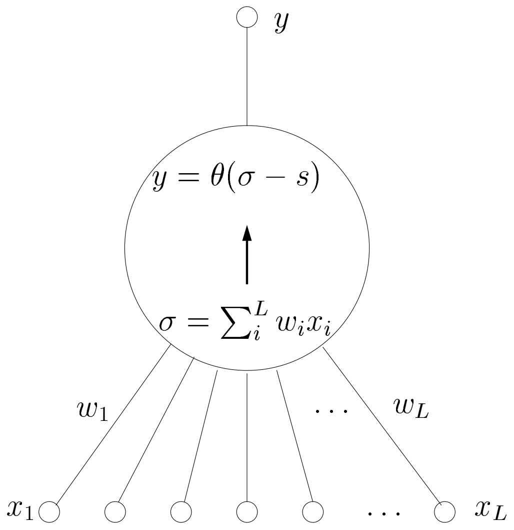


Abbildung 11.2: Ein Modellneuron mit Eingängen x_i, Gewichten w_i und einem Ausgang y. Das Ausgangssignal ergibt sich nach Gewichtung der Eingänge und Subtraktion eines Schwellwertes s durch Anwendung einer nichtlinearen Aktivierungsfunktion, hier der θ Funktion.

empfängt das Modellneuron Signale von außen. Diese werden an das Soma weitergeleitet und dabei mit einem Gewichtsfaktor $w_1, ..., w_L$ versehen. Im Soma wird nun die Summe über die Eingangssignale $\sigma = \sum_{i=1}^L w_i x_i$ gebildet. Das Ausgangssignal y der Zelle wird über eine nichtlineare Aktivierungsfunktion ermittelt, die z.B. die Form einer Sprungfunktion $y = \theta(\sigma - s)$ haben kann. Das resultierende Signal ist dann entweder "1" (Neuron feuert), wenn σ einen bestimmten Grenzwert s überschritten hat, oder "0" (Neuron feuert nicht), falls s nicht erreicht worden ist. Im Falle, daß kontinuierliche Ausgangssignale erwünscht sind, ersetzt man die Sprungfunktion durch eine stetig verlaufende Funktion, wie z.B.

$$f(x) = \frac{1}{1 + \exp(-x)}. \tag{11.1}$$

Beide Aktivierungsfunktionen sind in Abb. 11.3 gegenübergestellt.

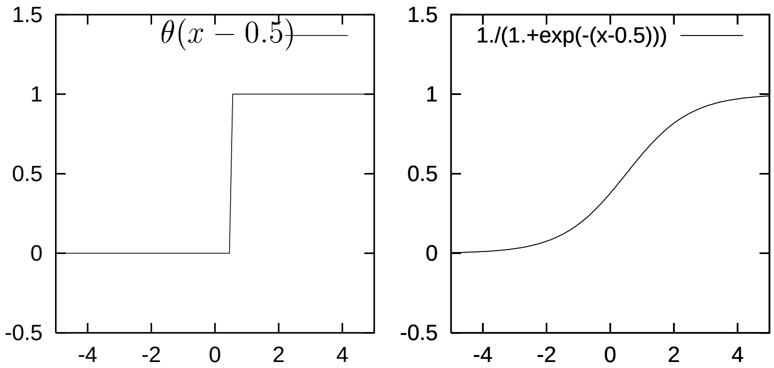


Abbildung 11.3: Mögliche Aktivierungsfunktionen sind die Heaviside-Funktion (links) und die Fermi-Funktion (rechts), hier dargestellt für Argumente mit einem positiven Schwellenwert von 1/2.

Mit Hilfe eines einzelnen Modellneurons lassen sich logische Grundoperationen wie die OR-Funktion oder die AND-Funktion realisieren. Ihre Wahrheitstabellen sind

OR	0	1
0	0	1
1	1	1

AND	0	1
0	0	0
1	0	1

Wir nehmen jetzt zwei Eingänge an und wählen als Aktivierungsfunktion die Sprungfunktion mit Schwellwert. Dann kann die OR-Funktion durch Wahl der Gewichtsfaktoren mit $w_{1,2} = 1$ und Schwellwert $s = 0.25$ dargestellt werden, für die AND-Funktion ist $w_{1,2} = 0.5$ und $s = 0.75$ eine geeignete Kombination.

Diese Neuronen, wie wir sie im folgenden weiterhin nennen werden, können auf verschiedene Weise miteinander verschaltet werden und ergeben dann Neuronale Netzwerke mit breit gestreuten, verschiedenen Eigenschaften und Anwendungen.

11.3 Das Perzeptron

Die einfachste Möglichkeit, ein solches Netz aufzubauen, ist das sogenannte *Perzeptron*, das eigentlich nur aus N unabhängigen Neuronen besteht, die jeweils die gleichen L Eingangssignale empfangen.

Diese N Eingangssignale können z.B. die Graustufen eines digitalisierten Bildes oder auch z.B. die Werte 0 oder 1 eines logischen Schaltkreises sein. Als Aufgabe stellen wir dem Perzeptron, für eine Menge von möglichen Eingangsmustern eine Zuordnung in Äquivalenzklassen vorzunehmen. Seien etwa die $x^{(\nu)}, \nu = 1 \ldots p$ die möglichen Eingangsmuster (etwa verschiedene Fotografien von verschiedenen Studenten in einem mit N Personen besetzten Hörsaal, jedes gerastert mit einer Auflösung von L Pixel) und sei für jedes ν bekannt, in welche von N Äquivalenzklassen (repräsentiert durch die eindeutige Identität des dargestellten Studenten) das Muster fällt, dann soll bei Präsentation eines Musters $x^{(\nu)}$ aus einer bestimmten Äquivalenzklasse ein bestimmtes der N Neuronen feuern und damit die Äquivalenzklasse identifizieren.

Das Neuron r soll feuern, wenn

$$\sigma_r = \sum_i^L w_{ri} x_i^{(\nu)} > 0 \tag{11.2}$$

ist. Hier haben wir angenommen, daß ein möglicher Schwellwert s der Aktivierungsfunktion verschwindet. Diesen können wir ohne Probleme in das obige Modell einbauen, wenn wir jedem Muster ein weiteres Signal $x_0^{(\nu)} = 1$ zuordnen, das mit jedem der N Neuronen über das Gewicht $w_{r0} = -s$ verbunden ist.

Es stellen sich nun zwei grundsätzliche Fragen: (i) Kann ein so spezifiziertes System überhaupt die Klassifizierungsaufgabe lösen, und (ii) falls ja, wie kann man die Gewichte w_{ri} ermitteln?

Die Antwort auf die erste Frage ist "manchmal." Um das einzusehen, nehmen wir an, daß wir Gewichte w_{ri} auf irgeneine Weise ermittelt hätten. Dann wird durch die Beziehung $\sum_i^L w_{ri} x_i = 0$ eine Hyperebene in dem L-dimensionalen Raum gegeben, der die Mustervektoren beherbergt. Damit jetzt das Neuron r alle Muster der Äquivalenzklasse $A(r)$ korrekt klassifiziert, muß diese Hyperebene so liegen, daß ausschließlich Muster aus $A(r)$ "oberhalb" liegen. In Abb. 11.4(a) wird diese Situation in $L = 2$ Dimensionen für 2 Äquivalenzklassen gezeigt. Der Teil (b) der Abbildung zeigt ein Beispiel, in dem keine lineare Separierbarkeit erreicht werden kann.

Im Falle, daß eine Klassifikation möglich ist, kann man ein Perzeptron mit der folgenden Lernregel dazu bringen, die Gewichtsfunktionen w_i korrekt zu erlernen. Falls eine Klassifizierung korrekt war, bleiben die Gewichte unverändert, sollte sie inkorrekt gewesen sein, so ändert man die Gewichte gemäß der sogenannten *Delta-Regel* an jedem Neuron r, das eine falsche Antwort gegeben hat,

$$\Delta w_{ri} = \epsilon (y_r^{soll} - y_r^{ist}) x_i. \tag{11.3}$$

Die positive Zahl ϵ, die üblicherweise $\ll 1$ gewählt wird, damit bereits gelernte Gewichte nicht zu sehr verändert werden, heißt *Lernrate.* Mit dieser Regel werden die Gewichte erhöht/erniedrigt, die aktive Eingänge mit falsch berechneten Ausgängen verbinden. War

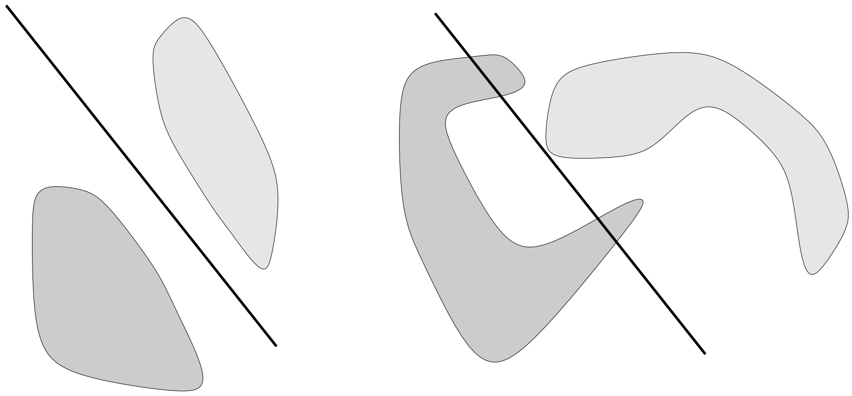


Abbildung 11.4: Muster in Äquivalenzklassen, die (a) linear separierbar und (b) nicht linear speparierbar sind.

das Ist-Ausgangssignal fälschlich 0, so wird das Gewicht erhöht, sonst erniedrigt. Die obige Regel läßt sich auch vektoriell schreiben,

$$\Delta \mathbf{W} = \epsilon(\mathbf{y}^{soll} - \mathbf{y}^{ist})\mathbf{x}^T, \tag{11.4}$$

wobei hier die Gewichte die Elemente einer Matrix geworden sind, und auf der rechten Seite der Spaltenvektor mit den Differenzen von Soll- und Ist-Ausgang mit dem Zeilenvektor bestehend aus den Eingangssignalen des präsentierten Musters multipliziert wird. Man macht sich durch kurzes Nachdenken klar, daß dies tatsächlich nur eine andere Schriebweise für die in einem Matrixschema angeordneten Gleichungen 11.3 ist.

Den Ablauf dieses Lernprozesses kann man an folgendem einfachen Beispiel erkennen, in dem ein einzelnes Neuron die Gewichte zur Realisierung der AND-Funktion lernt. Wir verwenden einen Schwellwert $s = 0.75$ und beginnen mit den falschen Gewichten $w_i = 1$. Einsetzen der verschiedenen Kombinationsmöglichkeiten in die Delta-Regel ergibt mit $\epsilon = 0.5$):

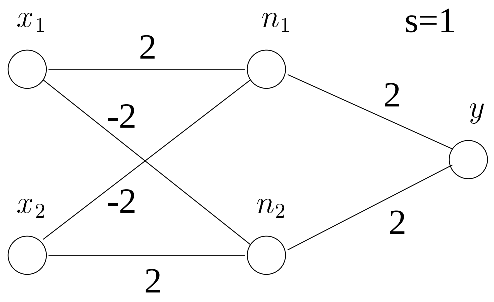
$$\begin{array}{lcll} \text{für } x_1 = 0,\, x_2 = 0 & \rightarrow & y = 0 \Rightarrow & \text{richtig} \\ \text{für } x_1 = 1,\, x_2 = 0 & \rightarrow & y = 1 \Rightarrow & \text{falsch!} \end{array}$$

Daher wenden wir die Delta-Regel an:

$$\begin{aligned} w_1' &= 1 + 0.5(0-1)\cdot 0 = 0.5 \\ w_2' &= 1 + 0.5(0-1)\cdot 0 = 1. \end{aligned}$$

Mit diesen neuen w_i präsentieren ein neues Eingangssignal

$$x_1 = 0,\, x_2 = 1 \rightarrow y = 1 \Rightarrow \text{falsch!}$$



x_1	x_2	n_1	n_2	y
0	0	0	0	0
0	1	0	1	1
1	0	1	0	1
1	1	0	0	0

Abbildung 11.5: Realisierung der XOR-Funktion mit Hilfe eines mehrschichtigen Netzes. Angedeutet sind die Gewichte, alle Schwellwerte betragen 1. Die Tabelle listet die Zustände der Zwischenschicht und den Ausgangszustand für die 4 möglichen Ausgangszustände auf. Die ersten zwei und die letzte Spalte geben die korrekten Werte der XOR-Funktion.

und wenden wieder die Delta-Regel an,

$$\begin{aligned} w_1'' &= 0.5 + 0.5 \cdot (0-1) \cdot 0 = 0.5 \\ w_2'' &= 1 + 0.5 \cdot (0-1) \cdot 1 = 0.5 \end{aligned}$$

Mit diesen Gewichten haben wir bereits oben gesehen, daß sich die richtigen Ausgangssignale ergeben.

Minsky und Papert haben 1969 gezeigt, daß dieser Lernprozeß immer konvergiert, wenn die Eingangsvektoren linear separierbar sind und als $\epsilon = 1/||\mathbf{x}||$ die inverse Norm des präsentierten Eingangsvektors gewählt wird. Der Beweis findet sich in [1].

11.4 Mehrschichtige Netze

Leider ist bereits die logische XOR-Funktion (wahr genau dann, wenn beide Eingangszustände *verschieden* sind) nicht linear separierbar. In solchen Fällen können mehrschichtige Netzwerke weiterhelfen.

Auch für mehrschichtige Netze läßt sich eine Lernregel angeben. Das Analog zur Delta-Regel bei den Perzeptron-Netzen ist hier die sogenannte *Backpropagation*-Methode. Man nutzt hier als Aktivierungsfunktion $f()$ eine stetig differenzierbare Funktion wie die sigmoide Fermi-Funktion aus dem Abschnitt 11.2.

Wir ordnen der Zwischenschicht die Aktivitäten $n_j = f(\sum_k w_{jk} x_k)$ zu. Analog gelte als Aktivität der Ausgabeschicht $y_i = f(\sum_j w_{ij} n_j)$. Die Gewichte der beiden Schichten wollen wir nur durch die Wahl der Indizes unterscheiden. Wir präsentieren dem Sytem jetzt Eingabemuster $\mathbf{x}^{(\nu)}$ und erwarten die zugehörigen Antworten $\mathbf{y}^{(\nu)}$ für $\nu = 1, \ldots, p$.

Ein Maß für den dabei auftretenden Fehler ist die Summe der Fehlerquadrate

$$E = \frac{1}{2}\sum_{\nu}^{p}\sum_{i}\left(y_i^{(\nu)} - y_i(\mathbf{x}^{(\nu)})\right)^2. \tag{11.5}$$

Dieser Ausdruck, der von den Gewichten der Zwischen- und Ausgabeschicht abhängt, soll jetzt durch ein Gradientenabstiegsverfahren (Hill Climbing) minimiert werden. Dazu ändert man alle Gewichte w_{jk} und w_{ij} wiederholt um

$$\Delta w_{ab} = -\epsilon \frac{\partial E}{\partial w_{ab}} \tag{11.6}$$

ab. Für hinreichend kleine ϵ bewegt man sich dadurch entlang der steilsten Abstiegsrichtung von E. Die Änderung von E bei Änderung aller Gewichte gemäß dieser Regel ist tatsächlich negativ,

$$\Delta E = -\epsilon \sum_{ab}\left(\frac{\partial E}{\partial w_{ab}}\right)^2 \leq 0. \tag{11.7}$$

Die in (11.6) benötigten Ableitungen erhält man durch die Anwendung der Kettenregel, zunächst für die Verbindungen von der Zwischen- zur Ausgabeschicht

$$\frac{\partial E}{\partial w_{ij}} = -\sum_{\nu}(y_i^{(\nu)} - y_i(\mathbf{x}^{(\nu)})f'(\sum_{j}^{\prime} w_{ij'}n_{j'})n_j \tag{11.8}$$

und in ähnlicher Weise für diejenigen von der Eingabe- zur Zwischenschicht,

$$\begin{aligned} \frac{\partial E}{\partial w_{jk}} &= -\sum_{\nu}\sum_{i}(y_i^{(\nu)} - y_i(\mathbf{x}^{(\nu)})f'(\sum_{j'} w_{ij'}n_{j'})w_{ij}\frac{\partial n_j}{\partial w_{jk}} \\ &= -\sum_{\nu}\sum_{i}(y_i^{(\nu)} - y_i(\mathbf{x}^{(\nu)})f'(\sum_{j'} w_{ij'}n_{j'})w_{ij} \\ &\quad \times f'(\sum_{k'} w_{jk'}x_{k'})x_k. \end{aligned} \tag{11.9}$$

Diese Ausdrücke lassen sich vereinfachen, wenn man für jeden Schritt nur nur ein einziges Muster und nicht die Summe in (11.8,11.9) verwendet. Das ist erlaubt, wenn wir sicherstellen, daß alle Muster gleich häufig präsentiert werden und wenn ϵ nicht zu groß ist. Verwenden wir außerdem noch für die Ableitungen der Fermi-Funktion $f()$ die Beziehung $f'() = f()(1 - f())$, und bezeichnen die Soll-Ist-Differenz bei Präsentation von $\mathbf{x}^{(\nu)}$ als $e^(\nu)$, dann finden wir

$$\Delta w_{ij} = \epsilon e_i^{(\nu)} n_j y_i(1 - y_i), \tag{11.10}$$

$$\Delta w_{jk} = \epsilon\left(\sum_i e_i^{(\nu)}\right)x_k y_i(1 - y_i)w_{ij}n_j(1 - n_j). \tag{11.11}$$

Man beachte die Ähnlichkeit der Regel (11.10) mit der Perzeptron-Delta-Regel, wenn man die Ableitung der Aktivitätsfunktion für einen Moment in ϵ absorbiert. Wir können daher auch jene Regel als Gradientenabstieg interpretieren.

Diese Lernregel ist zwar sehr allgemein, hat aber den gleichen Schönheitsfehler wie das Hill Climbing, daß wir im letzten Kapitel über Optimierungstechniken besprochen haben. Der Erfolg hängt davon ab, bei welchen Werten der Gewichte wir starten und ob wir ggfs. in einem lokalen Minimum steckenbleiben. Im Prinzip können wir hier das Wissen über Simulated Annealing anwenden, das wir uns dort erworben haben und den Lernalgorithmus entsprechend verbessern.

11.5 Rückgekoppelte Netze

Eine besonders wichtige Rolle für die Mustererkennung spielen Netzwerke, deren Ausgabemuster auf den Eingang zurückgekoppelt werden. Aufgabe des Netzes ist es, für ein verrauschtes Eingabemuster das ursprünglich gelernte unverrauschte Pendant zu ermitteln.

Ein geeignetes Netz wurde 1987 von John Hopfield vorgeschlagen. Es ähnelt in seiner Funktionsweise sehr den im vorigen Kapitel behandelten Spingläsern und es macht aus der Not, daß diese viele Zustände aufweisen, die energetisch niedrig liegen, aber geometrisch sehr unterschiedlich sind, eine Tugend. Wir wollen daher jetzt die Zustände $s_i = y_i = x_i = \pm 1$ zulassen, jeder Spin kann so mit einem Neuron identifiziert werden und ist sogleich Ein- als auch Ausgabewert. Die Gewichte w_{ij} zwischen den Neuronen/Spins entsprechen den (negativen, s.u.) Ising-Kopplungskonstanten J_{ij}. Wir fordern daher, daß die Kopplungen symmetrisch sind $w_{ij} = w_{ji}$, um eine Energiefunktion aufstellen zu können. In dieser sollen Selbstwechselwirkungen (d.h. ggf. von 0 verschiedene w_{ii}) wie im physikalischen Falle nicht gezählt werden,

$$E = -\sum_{i \neq j} w_{ij} s_i s_j. \tag{11.12}$$

Allerdings sind die w_{ij} langreichweitig, d.h. daß jeder Spin in der Regel mit jedem anderen verbunden ist.

Die Aktivierungsfunktion soll die Vorzeichenfunktion $\mathrm{sign}(x)$ sein, die positiven Argumenten oder Null die +1 und den negativen Argumenten die -1 zuordnet.

Wir definieren nun eine Dynamik des Netzes. An jedem Neuron i ermitteln wir die synaptische Aktivität/das magnetische Feld als Summe $\sum_j w_i j s_j$. Ist dieser Wert negativ, so soll der Spin i den Wert $\mathrm{sign}(\sum_j w_{ij} s_j) = -1$, anderenfalls den Wert +1 annehmen, sich also in Feldrichtung stellen. Denken wir uns die Gewichte als die Kopplungskonstanten $w_{ij} = J_{ij}$, dann entspricht das einer Energieabnahme des Systems. Diese Vorschrift wenden wir jetzt so häufig an, bis keine Änderung der Spin-Muster mehr eintritt, das System also in einen *Fixpunkt* der Dynamik gelaufen ist.

Es sollte klar sein, daß es sich dabei wieder um ein Gradientenabstiegsverfahren handelt und wir uns damit in ein Energieminimum begeben. Wir könnten diese Vorschrift auch als ein Hill Climbing mit deterministischer Zugauswahl, Nullpunkts-Monte-Carlo mit parallelem Spin-update, etc. bezeichnen.

Wie lernt nun so ein Netz? Bisher konnten wir mit überwachtem Lernen arbeiten, d.h. wir kannten die Ausgabe, die das Netz zu bestimmten Eingaben produzieren mußte und konnten die Gewichte entsprechend der Fehler des Netzes mit der Zeit modifizieren. Im Gegensatz dazu präpariert man im Hopfield-Modell die Gewichte von Anfang an so, daß das Netz durch seine Dynamik in diese gelernten Muster hineinläuft. Dazu setzt man

$$w_{ij} = \frac{1}{N} \sum_{\nu} x_i^{(\nu)} x_j^{(\nu)}, \tag{11.13}$$

wobei N die total Anzahl der Neuronen/Spins im System ist. Diese Kopplungen sind wie oben gefordert symmetrisch.

Weiterhin stellt jedes einzelnes Muster ein Energieminimum des Modelles dar, denn jedes gelernte Muster ist tatsächlich (fast) ein Fixpunkt,

$$\begin{aligned} \sum_j w_{ij} x_j^{\mu} &= \frac{1}{N} \sum_j \sum_{\nu} x_i^{(\nu)} x_j^{(\nu)} x_j^{\mu} \\ &= \frac{1}{N} \sum_j \left(x_i^{(\mu)} (x_j^{(\mu)})^2 + \sum_{\nu \neq \mu} x_i^{(\nu)} x_j^{(\nu)} x_j^{(\mu)} \right) \\ &= x^{(\mu)} + \frac{1}{N} \sum_{j,\nu\neq\mu} x_i^{(\nu)} x_j^{(\nu)} x_j^{(\mu)}. \end{aligned} \tag{11.14}$$

Die letzte Zeile zeigt, daß für große N und wenige gelernte Muster jedes dieser Muster ein Fixpunkt ist. Die verbleibende Summe läuft über $N(p-1)$ Summanden mit zufälligen[1] Werten ± 1. Diese Summe hat den Mittelwert 0, aber die Varianz $\sqrt{N(p-1)}$, so daß die als Rauschamplitude des zweiten Termes etwa $\sqrt{p-1)/N}$ beträgt. Das Modell verliert seine Klassifizierungskraft, wenn diese Rauschamplitude in die Nähe von 1 kommt, also die Anzahl der Muster einen nennenswerten Anteil der Anzahl der Neuronen ist. Genauere Untersuchungen zeigen, daß der kritische Übergang bei $p \approx 0.146N$ stattfindet.

11.6 Ausblick

Leider konnten wir das Thema neuronale Netze hier nicht erschöpfend behandeln. Nicht angesprochen wurden die Kohonen-Netze, die uns verstehen helfen, wie es zu einer selbst-

[1] Hier setzen wir voraus, daß die Muster selbst unkorreliert sind

organisierten Abbildung von externen Funktionen (Greifen, Hand, Fuß, Schmecken, ...) auf die Großhirnrinde kommt. Hier ist [1] eine gut lesbare Referenz zur Vertiefung. Hopfield-Netze lassen die Komplexität des Themas "assoziative Speicher" nur erahnen, und natürlich gibt es von allen Modellen viele Spielarten, vielleicht sollten hier asymmetrische Hopfield-Netze ($w_{ij} \neq w_{ji}$) und Netze für die zeitliche Mustererkennung Erwähnung finden. Die folgenden 2 Werke sind für einen Einstieg gut geeignet.

Weiterführende Literatur

[1] Ritter, Martinetz, Schulten, *Neuronale Netze*, Addison-Wesley, Bonn, 1991.

[2] Christopher M. Bishop, *Neural Networks for Pattern Recognition,* Clarendon Press, Oxford, 1994.