REIHE INFORMATIK
21/95

Die technische Implementierung neuronaler Netzwerke

Reinhard Männer
Universität Mannheim
Seminargebäude A5
D-68131 Mannheim

Die technische Implementierung neuronaler Netzwerke

Reinhard Männer

Lehrstuhl für Informatik V, Universität Mannheim, und
Interdisziplinäres Zentrum für Wissenschaftliches Rechnen, Universität Heidelberg

Einleitung

Ebenso wie Digitalrechner sind neuronale Netzwerke, z.B. das Gehirn, informationsverarbeitende Systeme. Das ist aber auch schon fast die einzige Ähnlichkeit. Wie wir an uns selbst sehen, haben beide Arten von Systemen grundlegend verschiedene Fähigkeiten: Digitalrechner eignen sich ausgezeichnet zum schnellen und fast beliebig präzisen Rechnen mit Zahlen; dafür wurden sie ursprünglich auch konstruiert. Andere Anwendungen kamen im Laufe der Zeit hinzu, z.B. die Verwaltung sehr großer Informationsmengen in Datenbanken, d.h. Speicherung und gezieltes Wiederauffinden anhand einfacher Suchkriterien. Beides ist für den Menschen extrem mühsam. Hingegen bewältigt er mit Leichtigkeit Aufgaben, die selbst für die modernsten Parallelrechner noch unlösbar sind. Jedes Kind kann im Bruchteil einer Sekunde die Gesichter seiner Eltern von denen potentieller Feinde unterscheiden, weil dies einen evolutionären Vorteil darstellt. Neben den exzellenten Fähigkeiten zur Bildverarbeitung können wir mittels unseres neuronalen Netzwerks Sprache verstehen und den komplexen Muskelapparat unseres Körpers mit phantastischer Genauigkeit kontrollieren. Allerdings müssen wir die meisten dieser Fähigkeiten erlernen, während ein Digitalrechner seine Aufgaben erfüllt, sobald ein entsprechendes Programm entwickelt wurde.

Biologische neuronale Netzwerke

Die unterschiedlichen Fähigkeiten von neuronalen Netzwerken und Digitalrechnern beruhen auf ihrem gänzlich andersartigem Aufbau. Während ein Digitalrechner einen oder wenige (bis zu 10^4) Prozessoren besitzt, die eine Sequenz von vorgeschriebenen Instruktionen, das Programm, auf Daten ausführen, besteht ein biologisches neuronales Netzwerk wie das menschliche Gehirn aus sehr vielen (etwa 10^{11}) Neuronen. Jedes Neuron besitzt grob 1.000 Eingänge, die nach einer geeigneten Bewertung seinen Zustand beeinflussen können; dieser Zustand wird dann als Ausgangssignal zu anderen Neuronen weitergeleitet. Die Signale sind elektrische Potentialänderungen, die durch chemische Vorgänge verursacht werden; mit Zeitkonstanten von etwa 10 ms sind sie relativ langsam verglichen mit Digitalrechnern, die heute

z.T. mit Taktzeiten von <10 ns arbeiten. Die geringe Geschwindigkeit der verwendeten chemischen Technologie spielt jedoch bei der massiven Parallelität biologischer neuronaler Netzwerke keine Rolle.

Ein einzelnes Neuron errechnet sein Ausgangssignal als Funktion seiner bewerteten Eingangssignale. Die Bewertung entspricht der Stärke der jeweiligen Synapse, der Verbindungsstelle zwischen einem seiner Eingänge und dem Ausgang eines anderen Neurons. Deshalb hängt die Funktion des Gesamtsystems wesentlich von der Stärke der synaptischen Verbindungen ab, von denen es z.B. im Gehirn etwa 10^{14} gibt. Das Gehirn funktioniert also nur zufriedenstellend, wenn die Synapsenstärken geeignet eingestellt wurden. Könnte man diese Information mit 50 MBytes/s – der typischen Übertragungsgeschwindigkeit von Rechnern – an die richtige Stelle im Gehirn laden, würde dies alleine etwa einen Monat dauern. Statt dessen wird das Gehirn trainiert, es lernt den richtigen Zustand. Die Synapsenstärken werden immer dann geringfügig verändert, wenn zwei verbundene Neuronen etwas miteinander zu tun zu haben scheinen; nach der einfachsten Lernregel wird die synaptische Verbindung verstärkt, wenn Aus- und Eingangsneuron im selben Zustand sind. Ein einigermaßen vollständiges Training eines Gehirns ist ebenfalls eine zeitraubende Angelegenheit; beim Menschen dauert dies 10 bis 20 Jahre.

Die Anzahl von Synapsen bestimmt also wesentlich, was ein neuronales Netzwerk überhaupt lernen kann. Wie schnell es dagegen mit seinem Wissen etwas anfangen kann, hängt davon ab, wieviele Synapsenwerte pro Zeit durch die Neuronen verarbeitet werden können. In Abb. 1 sind – entsprechend einer vom Verteidigungsministerium der USA durchgeführten Studie [1] – einige biologische neuronale Netzwerke nach der Größe dieser Parameter eingetragen.

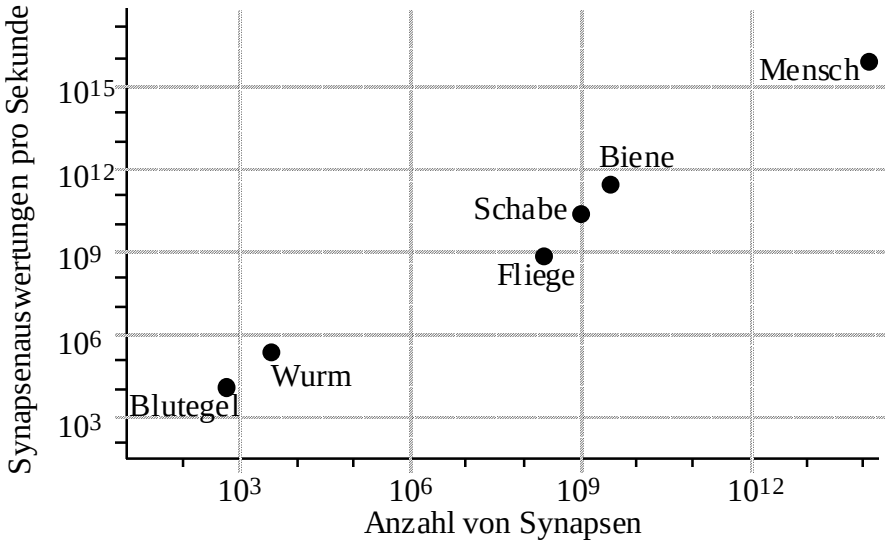

Abb. 1: Die Leistungsfähigkeit einiger biologischer neuronaler Netzwerke nach Verarbeitungsgeschwindigkeit und Speichergröße.

Künstliche neuronale Netzwerke

Die technische Nachbildung biologischer neuronaler Netzwerke verbindet sich mit der Hoffnung, in Bereichen große Fortschritte zu erzielen, in denen herkömmliche Rechner bislang scheiterten. Dies sind insbesondere die Verarbeitung von komplexen Bildern und akustischen Signalen und die Steuerung von komplexen Systemen. Weiter hat man die Erwartung, Probleme lösen zu können, bei denen z.B. ein menschlicher Experte die Lösung gelernt hat, also ein neuronales Netzwerk – sei es ein Mitarbeiter oder ein Gerät – trainieren kann, aber keinen Algorithmus angeben kann, der zur Lösung führt, also kein entsprechendes Programm schreiben kann. Systeme wie das Gehirn sind jedoch weder verstanden, noch kann eine entsprechende Zahl von Neuronen und Synapsen derzeit technisch nachgebildet werden. Es sind also extreme Vereinfachungen notwendig.

Das erste Beispiel für ein neuronales Netzwerk ist schon über 30 Jahre alt; es stammt von Karl Steinbuch [2]. Es soll die prinzipielle Wirkungsweise neuronaler Netzwerke demonstrieren. Später wird ihm u.a. eine moderne, hochintegrierte Version gegenübergestellt. Dabei wird offensichtlich werden, daß in diesen 30 Jahren nichts grundsätzlich Neues hinzugekommen ist.

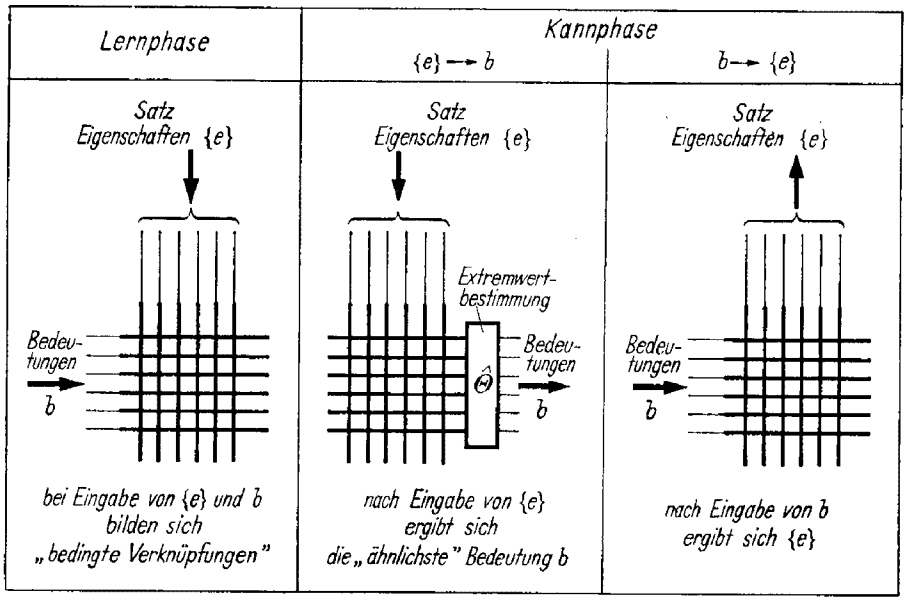

Abb. 2: Das Prinzip der Lernmatrix (Bild 104 aus [2]).

Steinbuch gab seinem Modell den Namen "Lernmatrix". Der Grund dafür ist in Abb. 2 zu sehen. Hier ist eine Reihe von horizontalen und vertikalen elektrischen Leitungen gekreuzt angeordnet; dies ist die Matrix. Steinbuch schlug vor, an den Kreuzungspunkten variable Widerstände anzubringen, die in einer Lern- oder Trainingsphase verändert werden. In

Anlehnung an die Pawlow´schen bedingten Reflexe nannte er sie "bedingte Verknüpfungen". Sie werden durch die Assoziation eines Satzes von "Eigenschaften" mit einer von mehreren "Bedeutungen" definiert, wobei beide durch elektrische Signale repräsentiert werden. Beispielsweise könnten die Eigenschaften die Werte der einzelnen Bildpunkte eines Fernsehbildes sein, das einen handgeschriebenen Buchstaben darstellt. Die zugehörige Bedeutung wäre das entsprechende Zeichen. Die Eigenschaften könnten jedoch auch die in einem Zeitintervall aufeinanderfolgenden Abtastwerte eines akustischen Signals sein, das gesprochene Sprache darstellt; die Bedeutung das zugehörige Wort; usw.

Das Trainieren einer Lernmatrix ist sehr einfach. Es werden Assoziationspaare an die Matrix angelegt, z.B. Eigenschaften an den vertikalen Leitungen und die zugehörigen Bedeutung an den horizontalen. Je nach Übereinstimmung der sich kreuzenden elektrischen Signale wird die bedingte Verknüpfung verstärkt. Dabei ist es durchaus erlaubt, daß fehlerhafte Kombinationen verwendet werden, solange die korrekten überwiegen. Wie in Abb. 3 gezeigt, addieren sich die korrekten Kombinationen am selben Kreuzungspunkt, während sich Störungen über die ganze Matrix verteilen.

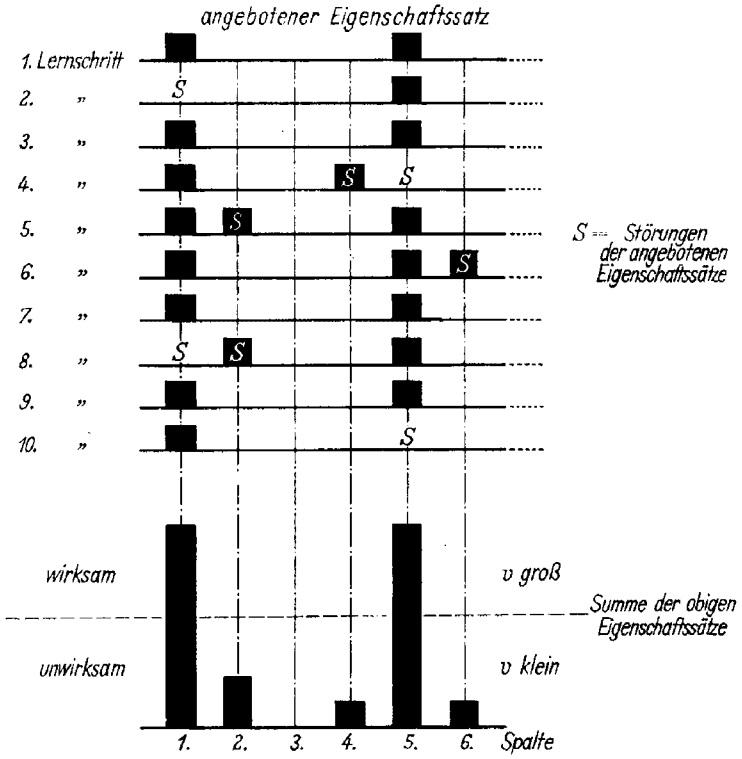

Abb. 3: Das Trainieren der Lernmatrix (Bild 105 aus [2]).

Ist das Training beendet, so wird die Lernmatrix in der Kann- oder Auswertephase dazu verwendet, einen unbekannten Satz von Eigenschaften mit früher gelernten zu assoziieren. Dies wird erreicht, indem der unbekannte Satz von Eigenschaften an die vertikalen Leitungen

angelegt wird. Die variablen Widerstände führen nun zu unterschiedlichen Strömen, die in die horizontalen Leitungen fließen. Auf jeder horizontalen Leitung werden die entsprechenden Ströme aufsummiert. Klarerweise ergibt sich dort die höchste Stromstärke, wo die höchste Übereinstimmung mit einem zuvor gelernten Muster auftritt. In Abb. 4 ist dargestellt, wie auf diese Weise ein Zeichen-Code in das entsprechende Zeichen umgesetzt wird. Mit jedem Eingangs-Bit e_i wird an eine von zwei vertikalen Leitungen Spannung gelegt: im Zustand "0" rechts, sonst links. Die Widerstände an den Kreuzungspunkten haben alle entweder einen festen Wert oder fehlen (dies entspricht dem Wert _). Das Bild zeigt, welche Erregung sich auf den horizontalen Leitungen ergibt, wenn einer der vier Codes eingegeben wird. Offensichtlich erlaubt eine Extremwertbestimmung die Zuordnung zum richtigen Zeichen. Wird ein unbekannter Code eingegeben, so wird die zu dem ähnlichsten Muster gehörige Bedeutung ausgegeben. Beispielsweise könnte dem Fernsehbild einer handgeschriebenen Postleitzahl – das ja nur aus einzelnen Bildpunkten besteht, mit denen ein Rechner gar nichts anfangen kann – die richtige Ziffernfolge zugewiesen werden, mit der dann eine Briefverteilungsanlage gesteuert werden kann. Die umgekehrte Betriebsweise, eine Bedeutung einzugeben und über die Widerstandsmatrix des trainierten Systems den zugehörigen Satz von Eigenschaften auszulesen, ist zwar möglich, hat jedoch keine praktische Bedeutung.

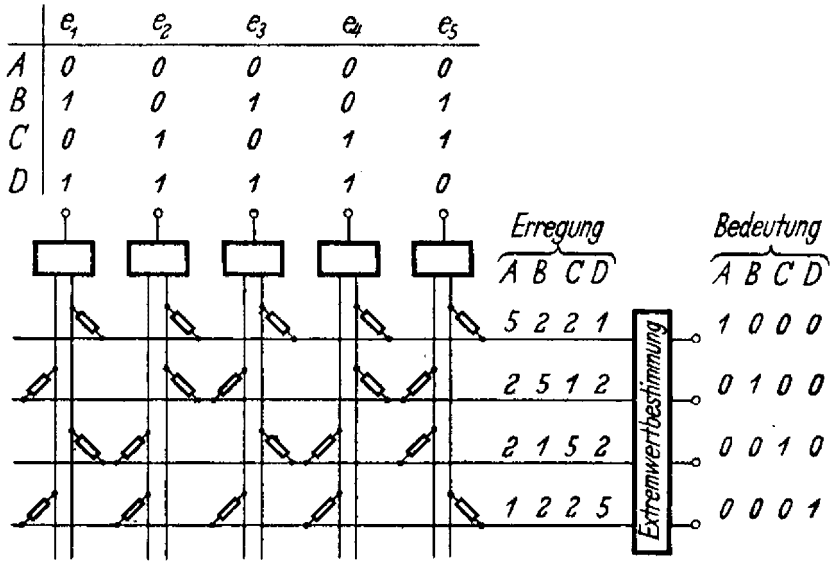

Abb. 4: Beispiel für die Anwendung einer Lernmatrix in der Auswertephase (Bild 107 aus [2]).

Bereits ein so simples System hat also bemerkenswerte Eigenschaften: Zunächst einmal kann es Muster "erkennen", worunter zu verstehen ist, daß ein beliebiges unbekanntes Muster dem ähnlichsten gelernten zuordnet wird. Wichtig dabei ist, daß es hier nicht notwendig ist, für diese Aufgabe ein Programm zu entwerfen. Daran sind bisher bei nur etwas komplizierteren

Mustererkennungsproblemen alle klassischen Ansätze, zu denen auch die traditionelle auf Regeln basierende Künstliche Intelligenz gezählt werden soll, gescheitert. Zudem ist es in weiten Grenzen egal, ob das System z.B. trainiert wird, Postleitzahlen zu erkennen, oder aus einer Reihe von Meßgrößen die optimalen Steuersignale für einen chemischen Reaktor abzuleiten. Ein weiterer wichtiges Gesichtspunkt ist die Fehlertoleranz des Systems. Es wurde oben bereits gezeigt, daß beim Training Fehler vorkommen dürfen; die schrittweise Anpassung der variablen Widerstände (der Synapsenstärken) sorgt dafür, daß das Netzwerk sich in der Auswertephase auf die wichtigen Bedeutungen konzentriert. Abweichungen der präsentierten von den gelernten Mustern spielen keine Rolle, solange nicht z.B. bei einem handgeschriebenen "E" der untere Querstrich nicht mehr sichtbar ist und deshalb ein "F" erkannt wird. Zu dieser Toleranz gegenüber Eingabefehlern kommt eine weitgehende Umempfindlichkeit gegenüber technischen Fehlern im System. Solche Fehler treten natürlich am wahrscheinlichsten in der Lernmatrix selbst auf, deren Größe ja immerhin mit dem Produkt aus der Zahl der vertikalen und horizontalen Leitungen wächst. Die hohe Zahl von Kreuzungspunkten hat jedoch auch ihre positiven Seiten: Ist z.B. einer davon defekt, so wird der Gesamtstrom auf der zugehörigen horizontalen Leitung, der ja über viele Kreuzungspunkte aufsummiert wird, nur unwesentlich verfälscht. Eine Lernmatrix oder andere Modelle neuronaler Netzwerke können also in höherem Maße miniaturisiert werden als digitale Schaltungen, die normalerweise hundertprozentig funktionieren müssen.

Um all diese attraktiven Eigenschaften auszunützen, geht es also zunächst darum, wie Widerstände realisiert werden könnten, deren Wert so variiert werden kann, wie es beim Training der Lernmatrix erforderlich ist. Dabei denkt jeder wohl erst an Potentiometer, wie sie z.B. zur Lautstärkeregelung beim Radio verwendet werden. Tatsächlich wurde ein System namens Perceptron, das in seinem Aufbau und seiner Wirkungsweise der Lernmatrix ähnelt und von Frank Rosenblatt, einem Pionier auf dem Gebiet der neuronalen Netzwerke, entwickelt wurde [3], unter Verwendung einer großen Zahl von motorgetriebenen Potentiometern aufgebaut (Abb. 5). Bernhard Widrow verwendete zuvor Bleistiftminen, die verschieden tief in einen Elektrolyten getaucht wurden [4].

Selbstverständlich war es notwendig, die Dimensionen zu verringern, einmal um größere und damit leistungsfähigere Systeme aufbauen zu können, zum anderen, um einen praktischen Einsatz zu ermöglichen. Ein erster Ansatz dazu war, Widrow´s Bleistiftminen zu integrieren: Elektroden und Elektrolyt wurden in einer Glasampulle untergebracht. Das entsprechende Bauelement, der "Memistor" (Abb. 6), wurde in den ersten kommerziellen Neurocomputern eingesetzt [5].

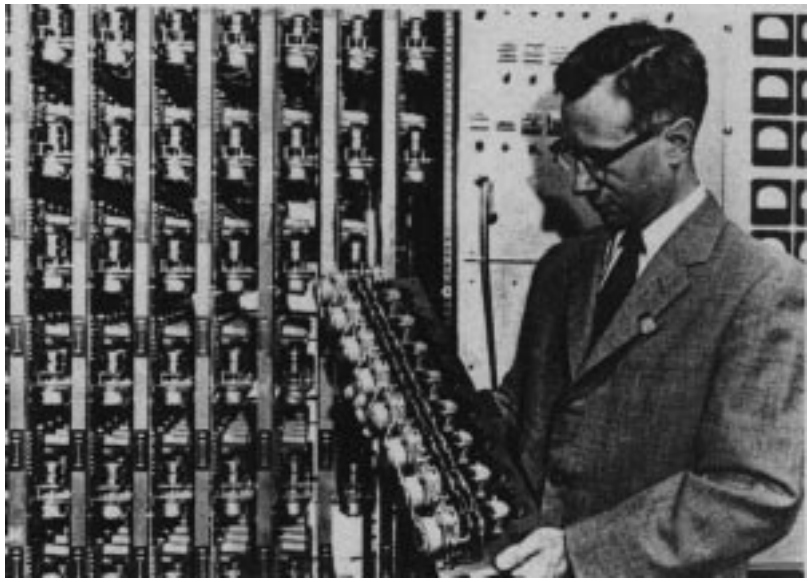

Abb. 5: Die erste Realisierung eines Perceptrons unter Verwendung von motorgetriebenen Potentiometern als variable Widerstände (Fig. 1.5 in [6]).

Abb. 6: Der Memistor, ein miniaturisierter variabler Widerstand (Fig. 8.12 aus [6]).

Offensichtlich waren diese Ansätze, neuronale Netzwerke direkt per Hardware zu realisieren, etwas unhandlich. Natürlich war störend, daß z.B. das Perceptron eine halbe Zimmerwand einnahm - aber das war für Rechner der damaligen Zeit nicht ungewöhnlich. Schlimmer war, daß ein solches in Hardware realisiertes System bestimmte, durch die Technologie und den praktischen Aufbau vorgegebene Eigenschaften besaß, die nur mit sehr großem Aufwand geändert werden konnten. Deshalb lag der Gedanke nahe, die Lernmatrix, das Perceptron, etc. nicht hardwaremäßig aufzubauen, sondern auf einem Universalrechner zu simulieren. Steinbuch schreibt [2]:

Selbstverständlich können Lernmatrizen auch mit Hilfe programmgesteuerter Rechenautomaten simuliert werden. Es zeigt sich jedoch, daß für typische Anwendungsfälle nicht nur der wesentlich höhere Aufwand des Rechenautomaten nachteilig ist, sondern auch die zu lange Rechenzeit. Beispielsweise braucht man zur Simulation einer Lernmatrix mit 50 Spalten und 50 Zeilen in der Kannphase 2.500 Multiplikationen, 2.450 Additionen und eine Maximumbestimmung. Hierzu brauchen auch schnelle elektronische Rechenautomaten mehrere Sekunden, während eine (viel billigere) Lernmatrix diese Aufgabe in wenigen Millisekunden löst. Das Ziel der technischen Entwicklung ist es, eine möglichst große Anzahl von Kreuzungspunkten auf kleinstem Raum und mit geringsten Kosten aufzubauen. Um dies zu erreichen, dürften besonders die elektro-chemischen Realisierungen interessant sein. Als unerreichbares

Ziel steht vor uns das menschliche Zentralnervensystem mit seiner ungeheueren Packungsdichte, mit Milliarden von Zellen und Milliarden von Synapsen im Litervolumen.

Davon abgesehen, daß sich Steinbuch bei der Zahl von Neuronen und Synapsen um bis zu fünf Größenordnungen irrte, hat sich über 30 Jahre später an der Beschreibung der Situation nicht viel geändert. Zwar kann jetzt eine solche Lernmatrix auf einem schnellen Arbeitsplatzrechner in etwa 100 µs simuliert werden; dafür sind aber die Modelle neuronaler Netzwerke komplizierter und die Netzwerke selbst wesentlich größer geworden. In vielen Fällen werden heutzutage sogenannte Multilayer-Perceptrons eingesetzt, die ein extrem vereinfachtes Beispiel für einen Ausschnitt aus dem Cortex darstellen. Sie bestehen aus mehreren Schichten von Neuronen (Abb. 7), einer Eingabeschicht $n_i^{(0)}$, einer (manchmal mehr als einer) verborgenen Schicht $n_j^{(1)}$ und einer Ausgabeschicht $n_k^{(2)}$. Innerhalb einer Schicht gibt es keine Verbindungen; benachbarte Schichten sind voll vernetzt, z.B. die ersten beiden Schichten mit den Verbindungsstärken $W_{i,j}^{(1)}$. Nach dem Anlegen eines Musters an die Eingabeschicht werden die Signale nur in Vorwärtsrichtung von Schicht zu Schicht verrechnet und weitergeleitet. Die Idee ist, daß die (erste) verborgene Schicht aus dem an der Eingabeschicht anliegenden Muster Eigenschaften extrahiert, anhand derer darauffolgende Schichten den Erkennungsvorgang durchführen können. Die verschiedenen Schichten können aus unterschiedlich vielen Neuronen aufgebaut sein; in der Regel nimmt ihre Zahl ab, weil von Schicht zu Schicht Teileigenschaften zu komplexeren Eigenschaften zusammengefaßt werden.

Die Ähnlichkeit zur Lernmatrix ist offensichtlich: Die Zustände der Neuronen in der Eingabeschicht entsprechen in Steinbuch´s Modell den Bedeutungen, die Zustände der Neuronen in der darauffolgenden Schicht den Eigenschaften und die Verbindungsstärken den variablen Widerständen. Steinbuch hat auch bereits "Multilayer-Lernmatrizen" vorgeschlagen; was hier neu ist, ist die Methode, das neuronale Netzwerk mittels Backpropagation [7] zu trainieren.

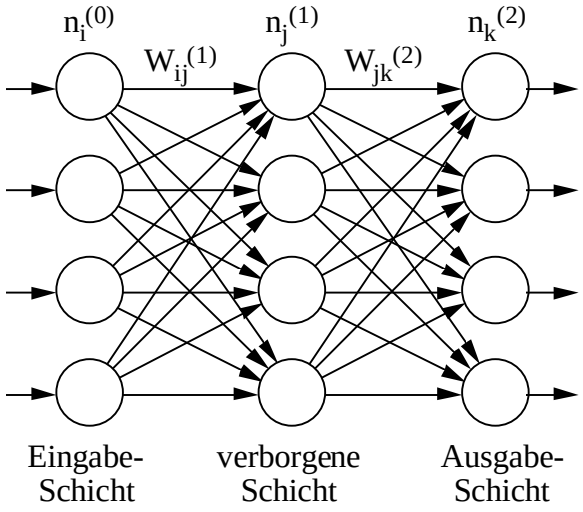

Abb. 7: Prinzipieller Aufbau eines Multilayer-Perceptrons.

Dabei wird so vorgegangen, daß dem Netzwerk zunächst an der Eingabeschicht ein Trainingsmuster präsentiert wird. Dieser Vektor $n_i^{(0)}$ muß nun mit der ersten synaptischen Matrix $W_{i,j}^{(1)}$ multipliziert werden, um das Erregungsmuster für die zweite Schicht $n_j^{(1)}$ zu ergeben. Nach Verarbeitung durch die Neuronen der zweiten Schicht erfolgt eine weitere Matrix-Vektor-Multiplikation, bevor durch die Ausgabeneuronen Signale erzeugt werden. Damit ist der erste Vorwärts-Durchlauf abgeschlossen. Solange aber das neuronale Netzwerk nicht ausreichend trainiert ist, werden die Ausgangssignale nicht dem entsprechen, was eigentlich gewünscht wird – z.B. wird ein handgeschriebener Buchstabe nach Anlegen der Bildpunkte, die ihn repräsentieren, nicht richtig erkannt. Deshalb müssen nun – das ist der Sinn der Trainingsphase – die synaptischen Matrizen so verändert werden, daß der Unterschied zwischen dem ausgegebenen und gewünschten Vektor verringert wird. Dies erfordert u.a. die Berechnung von Ableitungen und weiteren Matrix-Vektor-Produkten. Allerdings dürfen die Änderungen, die an den Synapsenwerten vorgenommen werden, nur relativ klein sein, sonst vergißt – bildlich gesprochen – das Netzwerk das, was es früher gelernt hat. Die schrittweise Anpassung des Netzwerks vom Ausgabevektor zurück zu der Eingabeschicht stellt den Rückwärts-Durchlauf dar. Leider ist es nun so, daß bei praktischen Anwendungen sehr viele Trainingsmuster präsentiert werden müssen und deshalb die Rechenzeiten, die auf einem schnellen Arbeitsplatzrechner für die Berechnung von Matrix-Vektor-Produkten benötigt werden, durchaus in der Gegend von Wochen betragen können.

Unglücklicherweise ist damit das Problem noch nicht erledigt; auch die Auswertephase kann sehr rechenaufwendig sein. Angenommen, es sollen Bilder der Größe 256×256 in Echtzeit, z.B. innerhalb von 0,1 Sekunden bearbeitet werden. In diesem Fall hätte die Eingabeschicht etwa 10^5 Neuronen. Hätte weiter die verborgene Schicht nur noch ein Zehntel so viele Neu-

ronen, so wäre die synaptische Matrix $W_{i,j}^{(1)}$ bereits 10^9 Elemente groß! Infolgedessen müßten für eine Echtzeitverarbeitung 10^{10} Multiplikationen und Additionen pro Sekunde berechnet werden können – eintausendmal mehr als ein großer Arbeitsplatzrechner vermag! Dazu kommt, daß die Speicherung der synaptischen Matrix etwa 1 GByte Hauptspeicher benötigt, etwa einhundert mal mehr als üblicherweise vorhanden. In Abb. 8, die ebenfalls an die erwähnte Studie [1] angelehnt ist, ist die benötigte Leistung bezüglich Speichergröße und Rechengeschwindigkeit für verschiedene Anwendungsgebiete aufgetragen: Anwendungen wie Robotersteuerung, Signalverarbeitung und Sprachverarbeitung können mit herkömmlichen Rechnern abgedeckt werden. Für die Echtzeitbildverarbeitung sind neue Konzepte nötig; an Systeme mit kognitiven Fähigkeiten, die dem Menschen vergleichbar sind, ist derzeit nicht zu denken.

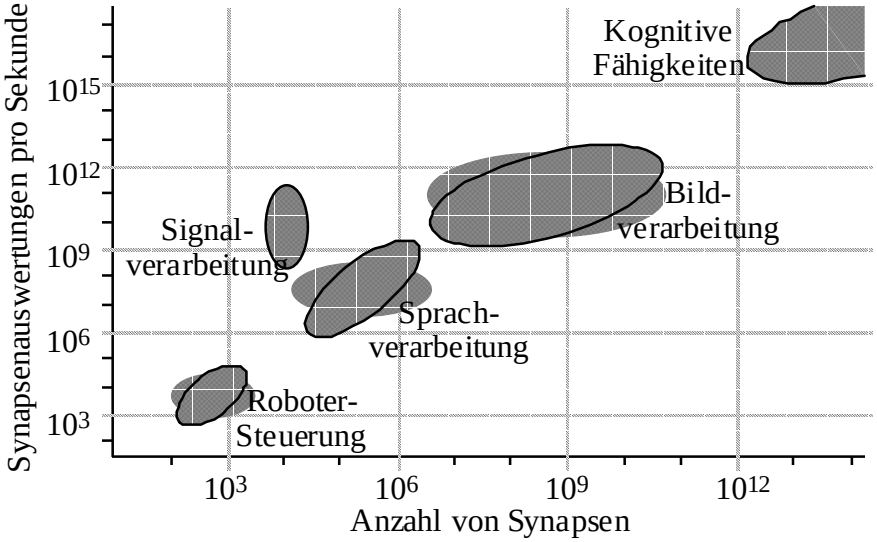

Abb. 8: Benötigte Leistung für verschiedene Anwendungsgebiete neuronaler Netzwerke.

Die systolische Berechnung von Matrix-Matrix-Produkten

Selbstverständlich liegt der Gedanke nahe, eine hohe Leistung dadurch zu erzielen, daß für eine bestimmte Anwendung das optimale neuronale Netzwerk ausgewählt und in möglichst hochintegrierter Form als ASIC (application specific integrated circuit) realisiert wird. Leider ist es aber sehr schwer, unter der Vielzahl von Modellen und möglichen Parametereinstellungen die richtige Auswahl zu treffen. In der Regel muß man erst einmal viele Möglichkeiten in einer Simulation testen, bevor man an die Herstellung von Spezial-Hardware denken kann. Die Simulation auf einem Universalrechner ist aber wegen der hohen Zahl von Operationen sehr zeitaufwendig (s.o.). Zur Beschleunigung liegt es nahe, Rechner zu bauen, die zwar nach wie vor frei programmierbar sind und deshalb alle Modelle neuronaler Netzwerke simulieren

können, die aber die zeitkritische Matrix-Matrix-Multiplikation besonders schnell ausführen können. Es wird also ein zusätzliches Rechenwerk benötigt, das für einen Universalrechner die Matrix-Matrix-Multiplikation übernimmt. Glücklicherweise ist ein solches Rechenwerk nicht allzu kompliziert, weil diese Operation sehr regulär ist. Zur Berechnung müssen alle Neuronenzustände (Index i) für alle Muster (Index p) mit allen synaptischen Gewichten W_{ij} multipliziert und geeignet aufaddiert werden:

$$h_{ip}(t) = \sum_{j=1}^{N} W_{ij} n_{jp}(t).$$

Dies gelingt am einfachsten, indem man eine Kette von m Prozessorelementen, hier Multiplizier-/Addier-Einheiten, parallel verwendet und die Operanden in einer speziellen Reihenfolge gleichzeitig in ein solches Rechenwerk eingibt.

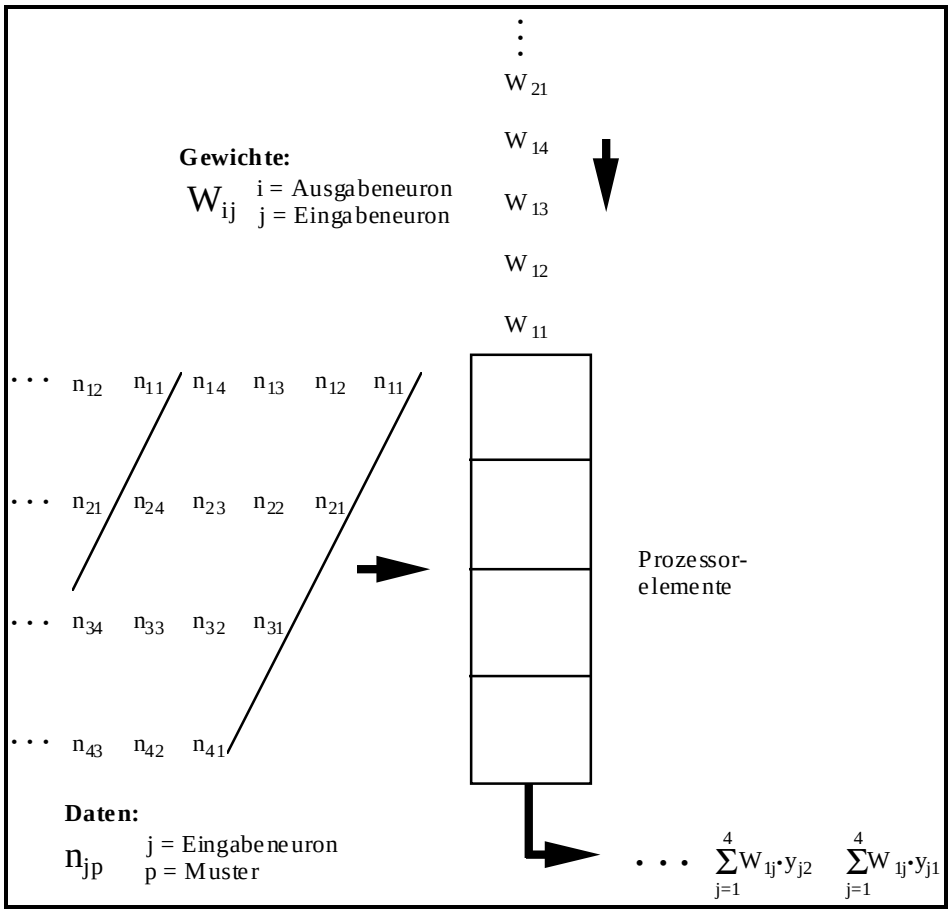

Abb. 9: Systolische Matrix-Matrix-Multiplikation.

Abb. 9 zeigt eine Anordnung für den Fall m=4. Hier werden im ersten Schritt der Ausdruck $W_{11}n_{11}$ berechnet, im zweiten die Ausdrücke $W_{11}n_{11}+W_{12}n_{21}$ und $W_{11}n_{12}$ usw. Nach vier Schritten ist die Kette gefüllt und es wird in jedem weiteren Schritt eine Summe der Form

$$\sum_{j=1}^{4} W_{ij} n_{ji}$$

ausgegeben. Diese Teilergebnisse müssen zwischengespeichert und zu den später berechneten Summen von j=5 bis j=8, von j=9 bis j=12 usw. addiert werden. Zur Erhöhung der Rechenleistung kann natürlich einerseits die Prozessorkette länger gemacht werden; andererseits können mehrere Ketten nebeneinander verwendet werden. So entsteht ein zweidimensionales Feld von lauter identischen, relativ einfachen Prozessorelementen. Da sowohl die synaptischen Gewichte, als auch die Eingangsdaten durch das Prozessorfeld geschoben werden, sind nur lokale Verbindungen zwischen den Prozessorelementen notwendig. Beides zusammen erleichtert eine Realisierung als integrierte Schaltung sehr: Das Prozessorelement mit seinen lokalen Verbindungen muß nur ein einziges Mal entworfen und kann dann so oft kopiert werden, daß die verfügbare Chip-Fläche optimal ausgenützt wird. Da die Prozessorketten sehr schnell gefüllt sind, ist die gesamte Rechenleistung der Anzahl der Prozessorelemente proportional. Eine solche Architektur bezeichnet man auch als systolisches Array [8].

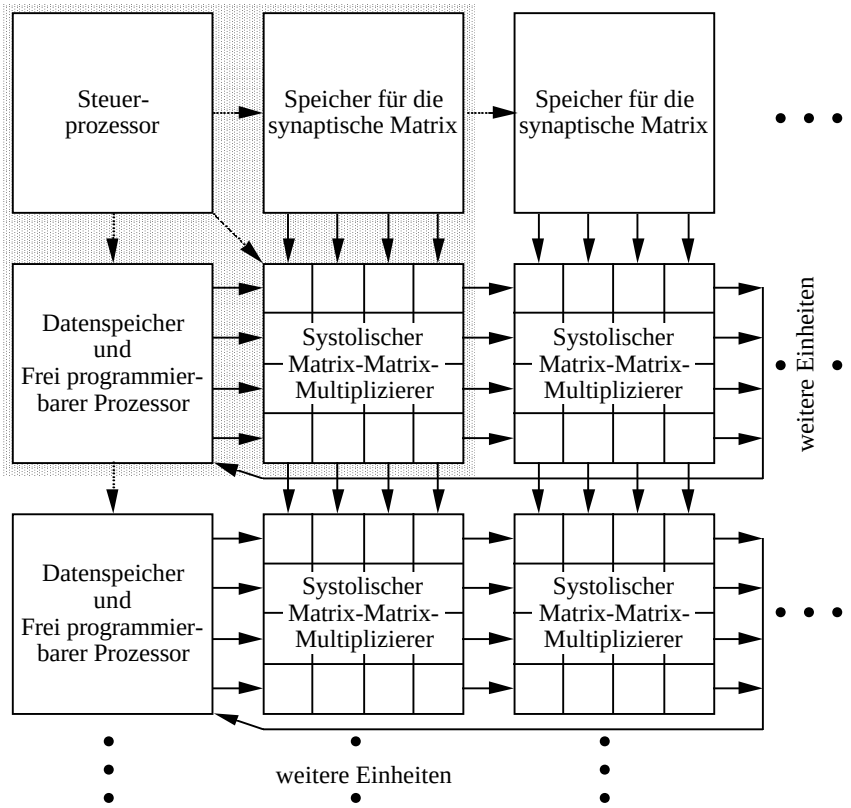

Abb. 10: Architektur des Neurocomputers SYNAPSE.

Neurocomputer

In unserem Fall berechnet ein systolisches Array allerdings nur die zeitkritische Matrix-Matrix-Multiplikation. Zur Simulation eines neuronalen Netzwerks sind jedoch eine ganze Reihe komplizierterer Operationen notwendig, die aber nicht so zeitkritisch sind. Diese können am besten von normalen Mikroprozessoren ausgeführt werden. Ein entsprechender Neurocomputer namens SYNAPSE wurde in einer Kooperation zwischen Siemens/München und der Universität Mannheim entwickelt [9]. Er hat den in Abb. 10 gezeigten Aufbau und setzt sich aus vier unterschiedlichen Einheiten zusammen: 1) einem systolischen Matrix-Matrix-Multiplizierer, 2) einem Speicher für die synaptische Matrix, 3) einer "Dateneinheit", die sowohl die Daten – die zu verarbeitenden Muster – als auch einen frei programmierbaren Prozessor enthält und 4) einem Steuerprozessor. Die Minimalkonfiguration besteht also aus vier Einheiten (siehe grau unterlegte Fläche in Abb. 10); bei größeren Anforderungen an die Rechengeschwindigkeit oder bei größerem Speicherbedarf kann das System in beiden Dimensionen erweitert werden. Dabei steigt seine Leistung proportional zur Zahl der verwendeten Prozessorelemente; es ist also skalierbar.

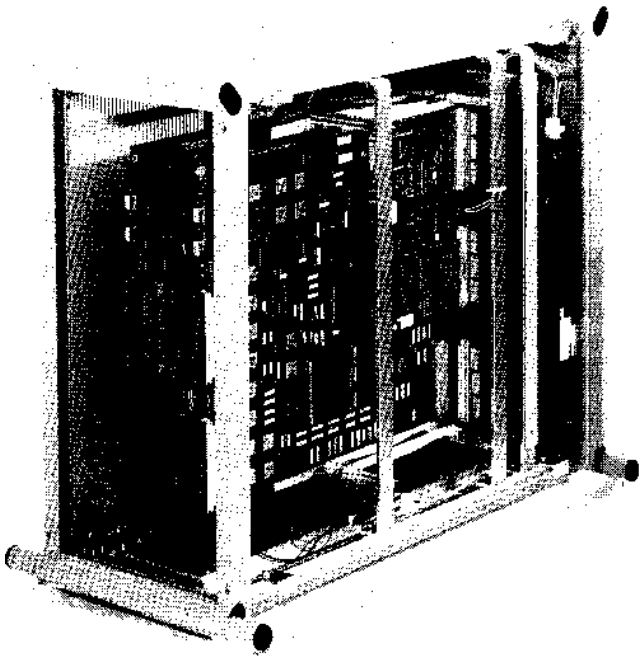

Abb. 11: Der Neurocomputer SYNAPSE-1, der von der Firma Siemens unter Mitarbeit der Universität Mannheim entwickelt wurde.

Allerdings wird diese simple Beschreibung der Komplexität des Systems in keiner Weise gerecht. Das zentrale Element ist der, grob vereinfacht als "systolischer Matrix-Matrix-Multiplizierer" bezeichnete, Neuro-Signalprozessor MA16. Er besteht aus 610.000 Transistoren, ist in 1µm CMOS-Technologie gefertigt und stellt die komplexeste digitale integrierte Schaltung dar, die je bei Siemens entwickelt wurde. Neben den erwähnten Matrix-Matrix-Multiplikatio-

nen kann er weitere Operationen ausführen, die für die Simulation neuronaler Netzwerke oder allgemein für die Signalverarbeitung benötigt werden. Ein einzelner Baustein hat bereits eine Rechenleistung von über $6 \cdot 10^8$ Multiplikationen und Additionen pro Sekunde. Eine Gewichtsspeicherkarte kann mit bis zu 0,5 GBytes bestückt werden. Die Dateneinheit ist selbst ein komplexer Rechner, der neben einem leistungsfähigen Mikroprozessor einen Spezialprozessor enthält, der auf die weniger zeitkritischen Operationen bei der Simulation neuronaler Netzwerke zugeschnitten ist. Zudem enthält sie mehrere Datenspeicher, die unabhängig von den Prozessoren durch eigene Steuereinheiten beschrieben oder gelesen werden können. Über sie können auch Daten in das System eingelesen werden. Der Datendurchsatz von dieser Einheit zum systolischen Prozessor ist gewaltig: Mit der Rate von 480 MBytes/s könnte z.B. dieser Artikel in 1 s etwa 10.000-mal übertragen werden. Die Steuereinheit enthält neben einem weiteren Mikroprozessor einen Adreßgenerator und einen frei mikroprogrammierbaren Generator für die Steuersignale der anderen Einheiten. Das System läßt sich auf verschiedenen Ebenen programmieren. Auf der untersten können direkt Eingriffe in das Mikroprogramm vorgenommen werden; dies ist natürlich nur für Experten ratsam. Auf der obersten Ebene steht ein komplettes Simulationspaket zur Verfügung, mit dem nur noch Parameter spezifiziert zu werden brauchen und die Bedienung interaktiv erfolgt.

Seit Herbst 1992 arbeitet ein System SYNAPSE-1. Bereits diese Minimalkonfiguration, die aus nur vier Karten besteht und ungefähr so groß ist wie ein Arbeitsplatzrechner (Abb. 11), hat eine beachtliche Leistung. Bei der Berechnung von neuronalen Netzwerken wurde eine etwa 8.000-mal höhere Geschwindigkeit gemessen als bei einem Rechner vom Typ SPARC-2 (ein schneller Standardrechner für Arbeitsplätze). Damit ist SYNAPSE-1 der derzeit schnellste Neurocomputer. Trägt man jedoch einen entsprechenden Datenpunkt in Abb. 1 ein, so stellt sich heraus, daß das System im wesentlichen die Leistung des biologischen neuronalen Netzwerks einer Fliege repräsentiert. Für manche mag das enttäuschend klingen; man sollte jedoch bedenken, welch komplexe Echtzeit-Bildverarbeitung die Fliege durchführt und welch komplizierte Flugmanöver gleichzeitig von ihr gesteuert werden.

Elektronische Implementierungen neuronaler Netzwerke

Möchte man noch leistungsfähiger werden, so muß man die flexible digitale Simulation durch kompakte, analoge Hardware ersetzen. Jetzt werden also nicht mehr Zahlen verarbeitet, sondern kontinuierliche Größen wie Ladungen, Spannungen, Ströme, Helligkeiten usw. Wie schon bei Steinbuch´s Lernmatrizen, stellen sich dabei zwei Probleme, die Realisierung der Matrix-Vektor-Multiplikation und die der synaptischen Matrix. Die analog-elektronische Realisierung der Matrix-Vektor-Multiplikation folgt dabei dem von der Mathematik vorgegebenen Schema (Abb. 12): Die Eingangssignale werden über entkoppelnde

Operationsverstärker auf vertikale Leitungen gegeben. Über variable Widerstände werden Ströme auf horizontale Leitungen geleitet und dort von anderen Operationsverstärkern aufsummiert. Dies war schon in Steinbuch´s Lernmatrix so realisiert.

Die vor 30 Jahren favorisierten elektro-chemischen Implementierungen variabler Widerstände sind jedoch heute viel zu unhandlich. Statt dessen greift man auf zwei bewährte Technologien zurück, die höchste Kompaktheit erlauben, Mikroelektronik und Optik. Gerade die Höchstintegrationstechnik macht es sehr einfach, variable Widerstände zu realisieren. Allerdings sind die Prozesse, mit denen normalerweise Mikroprozessoren oder Speicherbausteine hergestellt werden, nicht dafür geeignet, auf dem Chip Leiterbahnen durch Widerstandsmaterial zu verbinden. Statt dessen verwendet man Transistorstrukturen, die in MOS (metal oxid semiconductor) Technologie geradezu simpel sind: Eine Quelle (source) ist mit einer Senke (drain) durch einen Kanal verbunden, durch den Strom fließen kann. Wieviel, hängt davon ab, ob eine Steuerelektrode (gate), die isoliert über dem Kanal liegt, den Weg freigibt oder nicht. Da die entsprechenden Bauelemente spannungsgesteuert sind, genügt es, auf die Steuerelektrode den richtigen Betrag an Ladung zu bringen, der seinerseits dann den Widerstand des Kanals festlegt.

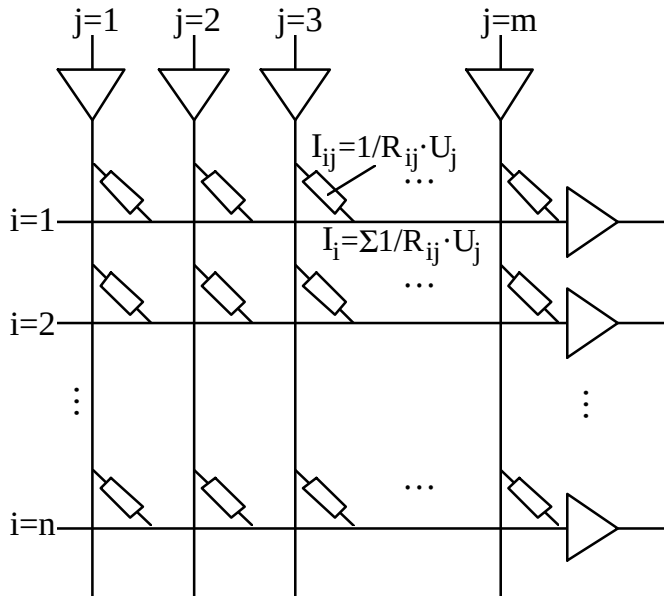

Abb. 12: Analog-elektronische Realisierung der Matrix-Vektor-Multiplikation.

Nachdem in den meisten Anwendungen neuronaler Netzwerke die Trainingsphase sehr viel seltener durchlaufen wird als die Auswertephase, wäre es sehr praktisch, wenn der Zustand der synaptischen Matrix nach dem Training automatisch erhalten bliebe. Genau dies leistet die "Floating-Gate"-Implementierung der variablen Widerstände. Hier werden zwei

Steuerelektroden verwendet (Abb. 13); das "floating gate" ist die Elektrode, die den Widerstandswert festlegt. Über ihr ist isoliert ein "control gate" aufgebracht, über das der Ladungszustand des floating gate verändert werden kann. Ist das control gate inaktiv, bleibt dieser Ladungszustand und damit der zugehörige Synapsenwert jahrzehntelang unverändert. Diese Standardtechnik wird auch in elektrisch löschbaren Festwertspeichern (EEPROM) eingesetzt.

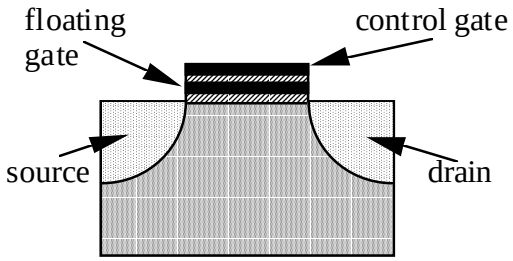

Abb. 13: Realisierung eines variablen Widerstandes mittels eines Floating-Gate-MOS-Transistors.

Da also Standardtechnologie verwendet werden kann, ist es relativ einfach möglich, ganze neuronale Netzwerke als Chip herzustellen. Ein Beispiel ist der Baustein ETANN (electrically trainable artificial neural network) der Firma INTEL. Er realisiert ein Multilayer-Perceptron, das aus drei Schichten (Eingabe-, verborgene und Ausgabeschicht) mit jeweils 64 Neuronen besteht. Zwischen je zwei Schichten ist eine volle Vernetzung mit 4.096 Synapsen realisiert. Es können also 64 analoge Eingangssignale an die erste synaptische Matrix $W_{i,j}^{(1)}$ angelegt und von dort an die zweite $W_{j,k}^{(2)}$ weitergeleitet werden. Die 64 analogen Ausgangssignale können extern weiterverwendet oder zur Realisierung von rückgekoppelten neuronalen Netzwerken, die hier nicht erwähnt wurden, an den Eingang der zweiten synaptischen Matrix zurückgeführt werden. Der Baustein ist also mit etwa 8.000 Synapsen zwar klein, aber relativ flexibel und weist die hohe Verarbeitungsgeschwindigkeit von 10^{10} Synapsenwerten/s auf. Trägt man einen entsprechenden Datenpunkt in Abb. 1 ein, so sieht man, daß eine einzige solche integrierte Schaltung gut für Signalverarbeitungsaufgaben geeignet ist. Abb. 14 läßt erkennen, daß ein großer Teil der Chipfläche für die beiden synaptischen Matrizen benötigt wird, die als dunkler Block in der Mitte zu erkennen sind.

Optische Implementierungen neuronaler Netzwerke

Trotz beachtlicher Leistung haben elektronische Implementierungen neuronaler Netzwerke einige gravierende Nachteile, die auf die elektrische Ladung der Teilchen zurückzuführen sind, deren Bewegung den Strom darstellt. In metallischen Leitern sind dies negativ geladene Elektronen; in Halbleitern können es auch positiv geladene "Löcher" sein – Stellen, an denen

ein Elektron fehlt. Elektrische Ladungen üben nun Kräfte aufeinander aus. Beispielsweise können die Ladungsträger, die auf ein floating gate aufgebracht werden, Ladungsträger aus dem Kanal zwischen source und drain verdrängen und somit den Widerstand des Kanals erhöhen. Leider stehen diesem erwünschten Effekt eine ganze Reihe von unerwünschten Nebenwirkungen gegenüber. Sollen z.B. Elektronen über eine Leitung transportiert werden, so stoßen sie sich gegenseitig ab und ziehen positive Ladungen der Umgebung an. Diese Umladungseffekte benötigen Zeit, weswegen sich elektrische Signale über Leitungen wesentlich langsamer als mit Lichtgeschwindigkeit ausbreiten. Dies führt letztlich auch dazu, daß die Schaltvorgänge elektronischer Bauelemente relativ lange dauern – um einen Transistor vom leitenden in den gesperrten Zustand oder umgekehrt zu bringen, muß die Steuerelektrode geladen oder entladen werden. Neben der begrenzten Signalübertragungs- und Schaltgeschwindigkeit führt die gegenseitig Beeinflussung von Ladungen aber auch dazu, daß ein Signal auf einer Leitung unerwünschterweise ein Signal auf einer daneben-liegenden Leitung induzieren kann – eine Störung, die die Dichte der integrierten Schaltung begrenzen kann. Schließlich kostet das unvermeidliche Verschieben von Ladungen Energie, die in Wärme umgewandelt wird; der Effekt ist umso stärker, je schneller die Umladevor-gänge durchgeführt werden, d.h. mit je höherer Taktrate ein solches System arbeitet. In vielen Fällen ist die Größe eines Chips oder seine Geschwindigkeit dadurch beschränkt, wieviel Wärme über das Gehäuse abgegeben werden kann. Aus diesem Grund sind höchstintegrierte Schaltungen auf zwei Dimensionen beschränkt: Alle Bauelemente und ihre Verbindungen sind in einigen wenigen Ebenen auf der Chipfläche untergebracht.

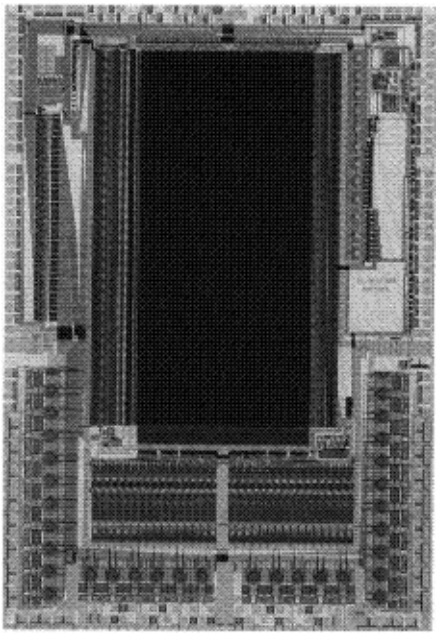

Abb. 14: Chip-Photo eines integrierten elektronischen neuronalen Netzwerks (ETANN, Firma INTEL).

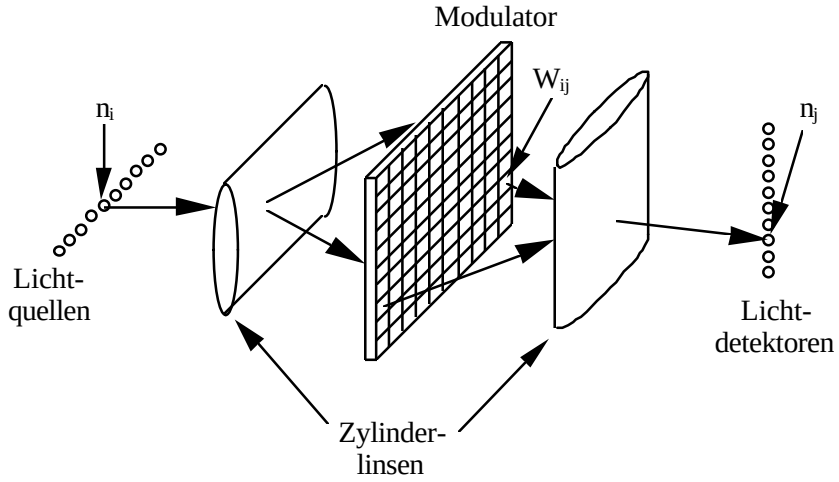

Abb. 15: Optischer Matrix-Vektor-Multiplizierer.

Damit liegt es nahe, an Alternativen zu denken, bei denen die Signalübertragung nicht durch geladene Teilchen erfolgt und somit die meisten dieser Nachteile nicht auftreten. Solche Teilchen sind die Photonen, aus denen Licht besteht. Ein optischer Matrix-Vektor-Multiplizierer könnte also aufgebaut werden, indem man das Schema aus Abb. 12 nimmt und dort ersetzt 1) die Eingangs-Operationsverstärker durch Lichtquellen, deren Lichtstärke das analoge Signal repräsentiert, 2) die Leitungen und Kontakte durch Lichtstrahlen und abbildende Elemente wie Linsen und Hologramme, 3) die Widerstände durch Licht-Modulatoren (d.h. Abschwächer) und 4) die Ausgangs-Operationsverstärker durch Lichtdetektoren. Damit gelangt man zu einer Anordnung, wie sie in Abb. 15 dargestellt ist.

Die entsprechende Anordnung könnte zwei Schichten eines Multilayer-Perceptrons repräsentieren. Die auf der linken Seite gezeigt Zeile von Lichtquellen stellt die Eingabeschicht dar. Das darauffolgende Abbildungselement, eine Zylinderlinse, weitet die punktförmigen Lichtquellen zu Linien auf, die jeweils eine Spalte der in der Mitte dargestellten Matrix überdecken. Dabei handelt es sich um einen Raummodulator, im einfachsten Fall um einen belichteten Film. Jedes Element der Matrix schwächt das auftreffende Licht um einen Faktor ab, der seiner Schwärzung entspricht; die Lichtstärke jedes Eingangsneurons wird also mit den Matrixelementen multipliziert. Der Raummodulator ist natürlich nichts anderes als die synaptische Matrix W_{ij}, die so viele Spalten enthält, wie Eingangsneuronen i, und so viele Zeilen, wie Ausgangsneuronen j vorhanden sind. Von der rechten Seite betrachtet, repräsentiert die beleuchtete Matrix die Erregungen, die an die Neuronen der Ausgangsschicht weitergeleitet werden. Dies wird erreicht, indem eine zweite Zylinderlinse die Matrix auf eine Zeile von Lichtdetektoren abbildet. Da diese Linse diesmal um 90° gedreht angeordnet ist, wird jeweils eine Zeile gemeinsam auf einen Detektor fokussiert; die Lichtstärken addieren sich dort.

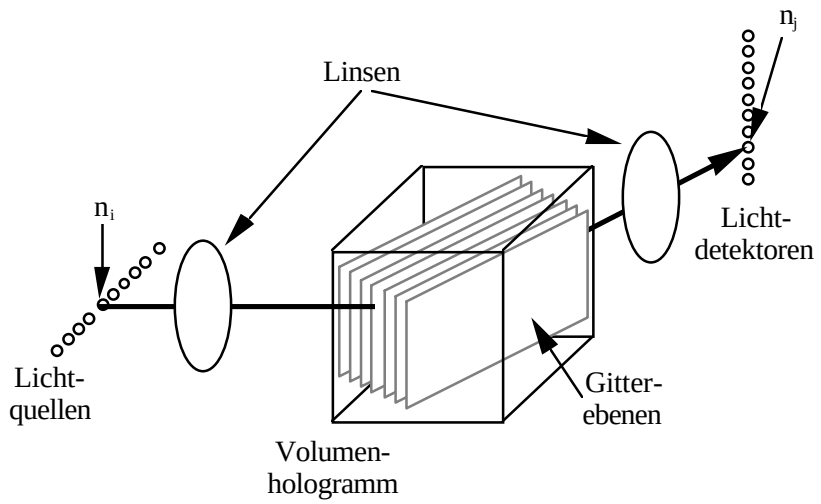

Abb. 16: Optischer Matrix-Vektor-Multiplizierer mit Volumenhologramm.

Eine etwas kompliziertere, aber auch leistungsfähigere Anordnung zeigt Abb. 16. Dort wurden beide Zylinderlinsen und die Modulationsmatrix ersetzt durch ein Volumenhologramm, das mittels der linken Linse mit parallelem Licht bestrahlt wird; das austretende parallele Licht wird durch die rechte Linse wieder auf die Lichtdetektoren abgebildet. Das Volumenhologramm arbeitet wie eine Reihe komplizierter Gitter, die das einfallende Licht sowohl modulieren, als auch geeignet ablenken. Würde nur ein Flächenhologramm verwendet, das einem einzelnen Gitter entspricht und z.B. auf einen Film belichtet werden kann, so wäre bereits bei einer Größe von 2,5×2,5 cm^2 eine vollständige Vernetzung von 10^4 Neuronen mit 10^8 Synapsen möglich; dies entspricht einer neuronalen Verarbeitung von Bildern der Größe 100×100 Bildpunkte. Noch eindrucksvoller ist die Leistungsfähigkeit von Volumenhologrammen, die in bestimmte Kristalle einbelichtet werden können: Bei einer Größe von 1 cm^3 ist eine vollständige Vernetzung von 10^6 Neuronen mit 10^{12} Synapsen möglich; dies entspricht einer neuronalen Verarbeitung von Bildern der Größe 1.000×1.000 Bildpunkten.

Zudem sind beide Anordnungen in der Lage, eine Matrix-Vektor-Multiplikation in der Zeit auszuführen, die das Licht zum Durchlaufen des Systems benötigt!

So betrachtet müßten optische Matrix-Vektor-Multiplizierer schon längst ihre elektronischen Alternativen verdrängt haben. Dies ist jedoch wegen gravierender Nachteile nicht der Fall. Zunächst sind die entsprechenden Lichtquellen, Modulatoren und Detektoren meist nur als Labormuster verfügbar, groß und teuer. Dies wird sich vermutlich in Zukunft ändern. Jedoch weisen optische Aufbauten Abbildungsfehler auf, die grundsätzlich nicht zu vermeiden sind. Zudem ist die analoge Auflösung der Systeme recht gering; während digitale Systeme mit 32 oder mehr Bits rechnen, kann dies der optische Aufbau nur mit etwa 5 Bits; seine Genauigkeit ist also mindestens um den Faktor $2^{27}Å10^8$ geringer! Dazu kommt eine relativ hohe

Empfindlichkeit gegenüber Materialfehlern. Zusammengenommen führen diese Effekte dazu, daß derzeit optische Matrix-Vektor-Multiplizierer für herkömmliche Anwendungen nutzlos sind.

Dies gilt jedoch nicht für ihren Einsatz in neuronalen Netzwerken. Wie früher gezeigt wurde, sind neuronale Netzwerke sehr tolerant gegenüber Fehlern aller Art. Insbesondere genügt in der Auswertephase eine sehr kleine Genauigkeit der Synapsenwerte. Statt Genauigkeit ist Geschwindigkeit gefragt, die der optische Aufbau bietet. Es ist deshalb zu erwarten, daß optische neuronale Netzwerke in immer stärkerem Maße eingesetzt werden.

Ausblick

Ausgehend von historischen Beispielen wurden drei Arten der technischen Realisierung neuronaler Netzwerke skizziert: 1) Neurocomputer, bei denen die zeitkritische Matrix-Vektor-Multiplikation durch spezielle Hardware unterstützt wird, die aber sonst frei programmierbare Rechner darstellen, 2) auf analoger Elektronik basierende integrierte Schaltungen, die sehr schnell, aber in ihren Ausbaufähigkeiten begrenzt sind, und 3) optische neuronale Netzwerke, die noch in ihren Kinderschuhen stecken, aber in der Zukunft sehr hohe Leistungen erwarten lassen. In Abb. 17 sind das System SYNAPSE-1 und der Baustein ETANN eingetragen. Um abzuschätzen, neuronale Netzwerke welcher Größe in Zukunft realisierbar sein sollten, kann das Joy´sche Gesetz verwendet werden, das eine Verdopplung der Leistungsfähigkeit von elektronischen Rechnern pro Jahr voraussagt. Die beiden im Bild eingetragenen Pfeile entsprechen der Entwicklung, die solche Systeme bis zum Jahr 2000 nehmen sollten. Ebenfalls eingetragen ist ein Punkt, der die projizierte Leistungsfähigkeit optischer Implementierungen andeuten soll. Wie aus der Abbildung zu entnehmen ist, werden in absehbarer Zeit elektronische Rechner nicht in den Bereich der kognitiven Fähigkeiten vorstoßen; für optische neuronale Netzwerke ist dies nicht vollständig ausgeschlossen.

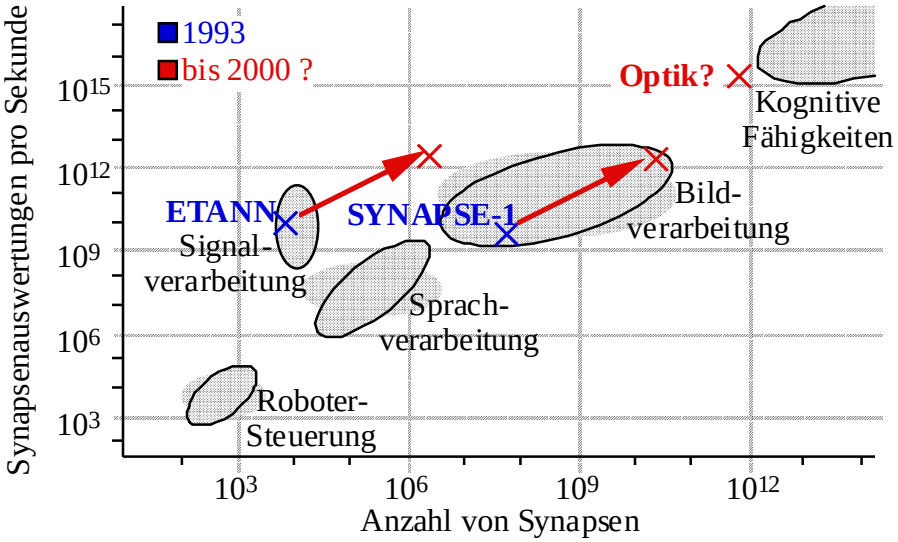

Abb. 17: Die zukünftige Leistungsfähigkeit von Neurocomputern, analog-elektronischer und optischer Implementierungen.

Danksagung

Das skizzierte System SYNAPSE beruht auf Überlegungen von U. Ramacher von Siemens/München. In seiner Gruppe wurden durch W. Raab, J. Anlauf, U. Hachmann, J. Beichter, N. Brüls, M. Weßeling und E. Sicheneder der Neuro-Signalprozessor mit der zugehörigen Prozessorkarte, der Gewichtsspeicher und die hier nicht erwähnte umfangreiche Software (Mikro-, System- und Anwendungsprogramme) entwickelt. J. Gläß und A. Wurz von der Universität Mannheim entwickelten den Steuerprozessor und die Dateneinheit. Hervorzuheben ist die ausgezeichnete Zusammenarbeit zwischen beiden Teams, ohne die die Realisierung von SYNAPSE in relativ kurzer Zeit nicht möglich gewesen wäre.

Literatur

[1] DARPA Neural Network Study, AFCEA International Press, Fairfax, VA (1990)

[2] K. Steinbuch: Automat und Mensch, Springer, Berlin (1965)

[3] F. Rosenblatt: The perceptron: A probabilistic model for information storage and organization in the brain; Psychol. Rev. 65 (1958) 386-408

[4] B. Widrow: ADALINE and MADALINE – 1963; Proc. IEEE 1st Int'l Conf. Neural Networks, San Diego, CA, Vol. I (1987) 145-157

[5] B. Widrow: Generalization and information storage in networks of ADALINE neurons; in: G.T. Yovitts (Ed.): Self-Organizing Systems, Spartan Books, Washington, DC (1962)

[6] R. Hecht-Nielsen: Neurocomputing; Addison-Wesley, Reading, MA (1989)

[7] D.E. Rumelhart, G.E. Hinton, R.J. Williams: Learning Internal Representation by Error Backpropagation; in: D.E. Rumelhart, J.L. McClelland (Eds.): Parallel Distributed Processing: Explorations in the Microstructures of Cognition, Vol. 1 (1986) 318-362

[8] H.T. Kung, C.E. Leiserson: Systolic arrays (for VLSI); Proc. SIAM Sparse Matrix Symposium (1978) 256-282

[9] U. Ramacher, W. Raab, J. Anlauf, U. Hachmann, J. Beichter, N. Brüls, M. Weßeling, E. Sicheneder, A. Wurz, J. Gläß, R. Männer: SYNAPSE-1: A High-Speed General Purpose Parallel Neurocomputer System; Proc. Parallel Computing Technologies ´93, Obninsk, Russia (1993) im Druck