Marijana Zekic

Neuronale Netze in Vorhersage von Rentabilität der Anlagen

1. Methodologie der neuronalen Netze

1.1. Entstehen und Entwicklung neuronaler Netze

Neuronale Netze sind eine der Methoden künstlicher Intelligenz, die nach der Struktur des menschlichen Gehirns modelliert worden sind. Sie haben bis jetzt eine zweifache Funktion gehabt:

1) als Vereinfachung biologischer Modelle zum Testen von Hypothesen über die Funktionsweise des Gehirns. (Zahedi, F.,1993, S.508)
2) als technologische Systeme zur Verarbeitung komplexer Informationen in Erschaffung der Maschinenintelligenz.

Aus dieser zweifachen Funktion gehen auche zwei grundsätzliche Entwicklungslinien neuronaler Netze hervor. Die eine Funktion geht von einem biologischen Aspekt aus und bewertet die neuronalen Netze nach ihrer Modellierungs- und Erklärungsfähigkeit von Aktivitäten des natürlichen Gehirns, während die zweite Linie die neuronale Netze als technologische Erfindung versteht und sie nach ihrer Effizienz bei praktischer Anwendung zur Problemlösung bewertet. Verschiedene Entwicklungslinien dieser Methode haben zum Aufbau verschiedener Modelle von neuronalen Netzen geführt, dessen Ergebnis auch die zahlreichen Bezeichnungen für die Richtungen sind: connectionism, parallel distributed processing, neurocomputing, naturally intelligent systems und artificial neural networks. (Zahedi, F.,S 510).

1.2. Vorteile neuronaler Netze

Forschungsergebnisse zahlreicher Autoren haben einige positive Eigenschaften von neuronalen Netzen ausgewiesen, die dieser Methode den Vorrang vor den anderen Techniken geben:

- Fähigkeit, unzulängliche, ungewisse Daten zu analysieren und die Fähigkeit Probleme zu lösen, bei denen die Lösung nicht klar einzusehen ist
- Fähigkeit auf Vorinformationen zu lernen

Trotz vielen positiven Ergebnissen, kann der Gebrauch von neuronalen Netzen nicht generalisiert werden, so dass die Behauptung des Timothy Masters (Masters, T.,1993, S. 8), dass alle Probleme, die mit traditionellen Modellierungs- und statistischen Methoden gelöst werden können, in den meisten Fällen effektiver mit Hilfe von neuronalen Netzen lösbar sind, teilweise akzeptabel ist für die Lösung gewisser Probleme.

1.3. Struktur und Funktionsweise eines neuronalen Netzes

1.3.1. Neurone und ihre Verbindungen

Unter dem Begriff Neuron wird jedes Element eines Modells bezeichnet, mit dessen Hilfe eine Datenverarbeitung im neuronalen Netz durchgeführt wird. Neurone sind in einem Netz so verbunden, dass die Ausgangsinformation des einen Neurons die Eingangsinformation für das andere Neuron darstellt. Die Verbindung zwischen den Neuronen kann in nur eine oder in beide Richtungen verlaufen.
Ein wichtiges Element für die Verarbeitung mit neuronalen Netzen ist die Stärke der zwischen den Neuronen bestehenden Verbindung, die als Gewichtswert der Verbindung oder einfach als "Gewicht" bezeichnet wird. (z.B. W_{ji} bezeichnet den Verbindungsgewicht vom Neuron j zum Neuron i). Das Verbindungsgewicht in der Gegenrichtung, von Neruon i zum Neuron j, wird mit dem Symbol W_{ij} bezeichnet). Lernen basiert auf dem Prinzip der Gewichtsanpassung zwischen den Neuronen in einem Netz.

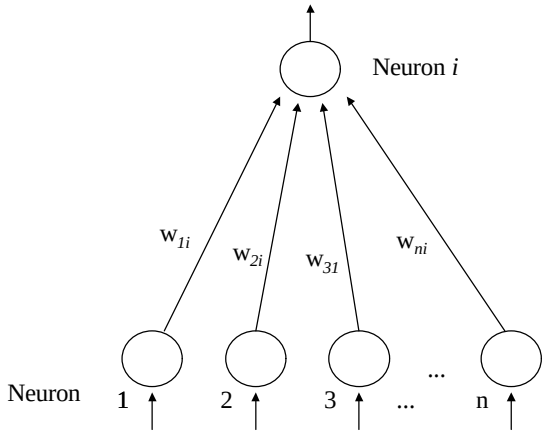

Bild 1. Illustration einer Verbindung zwischen Neuronen im neuronalen Netz

Neurone sind in Schichten gruppiert (layers, slabs). Drei Haupttypen der Schichten sind zu unterscheiden:

- Eingangsschicht,
- Zwischenschichten oder verborgene Schichten und
- Ausgangsschicht des Neurons

Das häufigste neuronale Netz besteht aus drei Schichten von Neuronen (Eingabeschicht, verborgene Zwischenschicht und Ausgabeschicht). Die Neurone der Eingabeschicht sind die Eingangsvariabeln ins Modell, die Informationen aus der äusseren Umgebung enthalten. Diese Informationen werden von Neuronen der Zwischenschicht empfangen, zur Ausgabeschicht weitergeleitet, wonach bei einem wiederholten Informationstransfer eine Korrektur der Gewichte durchgeführt wird. Die Korrektur wird durch viele Iterationen an den gewünschten, in Lerndaten gegebenen, Ausgang geführt. Endlich, nachdem das gewünschte Ergebnis im Lernprozess erreicht wird, wird die Ausgabe (output) dem Benutzer gezeigt.

Zwei Neurone können mit einer fördernden (excitatory) und einer hemmenden (inhibitory) Verbindung miteinander verbunden sein.

Der Entwurfsprozess eines neuronalen Netzes besteht aus folg. Phasen: (Zahedi, F.,1993, S. 515)

1. Anordnung der Neurone in verschiedenen Schichten
2. Bestimmung über die Art der Verbindung zwischen Neuronen verschiedener Schichten, sowie zwischen Neuronen derselben Schicht
3. Bestimmung über die Weise, auf die Neurone Eingaben empfangen und Ausgaben herstellen
4. Bestimmung der Verbindungsstärke im Netz

Verbindungstypen zwischen Neuronenschichten (inter-layer): (Zahedi, F.,1993, S. 516)

- vollständige Verbindung (fully connected) - jedes Neuron der ersten Schicht ist mit jedem Neuron der zweiten Schicht verbunden
- Teilverbindung (partially connected) - Neuron der ersten Schicht muss nicht unbedingt mit jedem Neuron der zweiten Schicht verbunden sein.
- Vorwärtsverbindung (feed-forward) - die Verbindung verläuft in eine Richtung,
- Neurone der ersten Schicht senden ihre Ausgabe an die Neurone der zweiten Schicht, nehmen aber keine Rückgabe an.
- Vor- und Rückwärtsverbindung (bi-directional) - ausser der Vorwärtsverbindung besteht auch die Rückwärtsverbindung, wobei Neurone der zweiten Schicht ihre Ausgabe an die Neurone der ersten Schicht senden.
- Hyerarchische Verbindung (hierarhical) - Neurone der einen Schicht sind nur mit Neuronen der nächsten Schicht verbunden.
- Resonanz (resonance) - ist eine zwei-Richtungsverbindung, in der die Neurone, die Informationen immer weiter senden zwischen den Schichten, bis eine gewisse Bedingung erfüllt wird.

Beispiele bekannter Netze mit Verbindungen zwischen Neuronenschichten (inter-layer):

- **Perceptron** (Frank Resenblat, 1975o) - das erste neuronale Netz, zweischichtig, vollständige Verbindung
- **ADALINE** (Bernard Widrow, Marcian E. Hoff, 1962) - zweischichtige, vollständige Verbindung
- **Backpropagation** (Paul Werbos, 1974, von Rumelhart, Hinton, Williams erweiter - das erste dreischichtige Netz mit einer oder mehreren verborgenen Schichten, die Verbindung zwischen den verborgenen Schichten ist hierarchisch (sieh Abb 3).
- **ART** (Adaptive Resonance Theory) (Stevens Grosberg, 1976) - resonante Verbindung, dreischichtiges Netz.
- **Feedforward Counterpropagation** (Robert Hecht-Nielsen, 1987) - die Struktur ähnelt dem backpropagation Netz, dreischichtig, aber nicht hierarchisch, es besteht auch eine Verbindung innerhalb einer Neuronenschicht.

- **Full Counterpropagation** - dreischichtig nichthierarchisch, aber in beiden Richtungen

Die Verbindungen innerhalb einer Neuronenschicht (intra-layer) können sein:

1. ***recurrent Verbindung*** - Neurone sind völlig oder teilweise innerhalb einer Schicht verbunden. Die Verbindung besteht darin, dass nach der Übernahme der Eingabe aus einer anderen Schicht, die Neurone ihre Ausgaben solange untereinander kommunizieren, bis ein stabiler Zustand erreicht wird. Erst dann dürfen sie die Ausgabe in eine andere Schicht senden.

Beispiele für die intra-layer Netze mit einer recurrent Verbindung:

- **Hopfields Netz** (John Hopfield, 1982) - zweischichtig, mit vollständiger Verbindung, Neurone der Ausgabeschicht sind untereinander mit einer recurrent intra-layer Verbindung verbunden.
- **recurrent backpropagation Netz** (David Rumelhart, Geoffrey Hinton, Ronald Williams, 1986) - eine recurrent intra-layer Verbindung, aber einschichtig, wobei ein Teil der Neurone Eingaben empfängt, und der Rest völlig durch einen recurrent intra-layer Kontakt verbunden ist.

2. ***on-center/off-surround*** - Verbindung, wobei ein Neuron in einer Schicht eine fördernde Verbindung zu sich und zu den benachbarten Neruonen, und eine hemmende Verbindung zu den anderen Neuronen in der Schicht hat.

Beispiele von Netzen mit solcher Verbindung:

- **ART1, ART2, ART3** (S. Grosberg, 1960) - resonante on-center/off-surround Verbindung
- **Kohonens selbstorganisierendes Netz** (Teuvo Kohonen, 1982)
- **Counterpropagation Netze**
- **Netze mit konkurrierendem Lernen** (Competitive Learning)

1.4. Eingabe und Ausgabe im neuronalen Netz

1.4.1. Eingabe ins Neurone (Input)

Wie verarbeiten die Neurone Informationen, die sie empfangen? Ist die Ausgabe, die ein Neuron *j* dem Neuron *i* sendet, als Ausgabe *j* bezeichnet, dann wird die Eingabe ins Neuron *i* nach der Formel ermittelt:

$$input_i = \sum(\text{aller j Neurone, die mit dem Neuron i verbunden sind}) \cdot w_{ij} \cdot output_j \quad (1)$$

Anders gesagt, das $input_i$ eines Neurons *i* ist der Wert aller gemessenen Signale, die in dieses Neuron gelangen. Ausser dieser Standardnetzeingabe, gibt es noch zwei Arten spezifischer Eingaben im Netz:

- $extinput_i$ (äussere Eingabe) - Eingabe, die ein Neuron aus der äusseren Umgebung empfängt.
- $bias_i$ - bias Wert, der in manchen Netzen zur Kontrolle der Aktivierung der Neuronen gebraucht wird.

Eingabewerte können im Interval 0 bis 1 normalisiert werden, um ihren zu grossen Anstieg zu vermeiden. Deswegen ist es notwendig, eine Normalisierung bei den meisten neuronalen Netzen durchzuführen (beim Kohonens Netz z.B. ist es obligatorisch).

1.4.2. Ausgabe von Neuronen (Output)

Ermittlung der Ausgabewerte erfolgt nach der sogenannten Übertragungsfunktion (transfer function). Einige der am häufigsten gebrauchten Übertragungsfunktionen sind:

- step Funktion
- signum Funktion
- sigmoide Funktion
- hyperbolisch-tangente Funktion
- lineare Schwellenwertfunktion (treshold linear function)

1. Bei der step Funktion wird die Ausgabe nach der folg. Formel ermittelt

$$output_i = \begin{cases} 0 \text{ wenn inputi}_i \leq T \\ 1 \text{ wenn inputi}_i \rangle\ T \end{cases} \qquad (2)$$

wobei T der realer Wert und der Schwellenwert der Funktion ist.

2. Signum Funktion ist ein Sonderfall der step Funktion, wobei der Schwellenwert T=0 ist:

$$output_i = \begin{cases} 1 \text{ wenn } input_i \rangle\ 0 \\ 0 \text{ wenn } input_i = 0 \\ -1 \text{ wenn } input_i \langle\ 0 \end{cases} \qquad (3)$$

Ein Beispielnetz mit dieser Funktion ist das erste neuronale Netz Perceptron.

3. Sigmoide Funktion hat die Formel:

$$\text{output}_i = \frac{1}{1 + e^{g \cdot \text{input}_i}} \qquad (4)$$

wobei *g* den Funktionsanstieg darstellt und wird ermittelt als $g = 1/T$, wobei *T* der Schwellenwert der Funktion ist. Der Anstieg bestimmt die Neigung der Funktion um 0. Die Funktion ergibt kontinuierliche Werte im Interval von 0 bis 1 als Resultat. Sie wird z.B. bei der Backpropagation und Hopfields Netz verwendet und ist eine der am häufigsten verwendeten Funktionen in neuronalen Netzen. Die Graphik ist in der Abbildung 2. dargestellt.

4. Eine besondere Art der sigmoiden Funktion ist die hyperbolisch-tangente Funktion. Die Formel ist:

$$\text{output}_i = \frac{e^{u} - e^{-u}}{e^{u} + e^{-u}} \tag{5}$$

wobei $u = g \cdot \text{input}_i$.

Der Graph dieser Funktion ist der sigmoiden Funktion ähnlich, die sich im Intervall der Werte unterscheidet (zwischen -1 und 1)

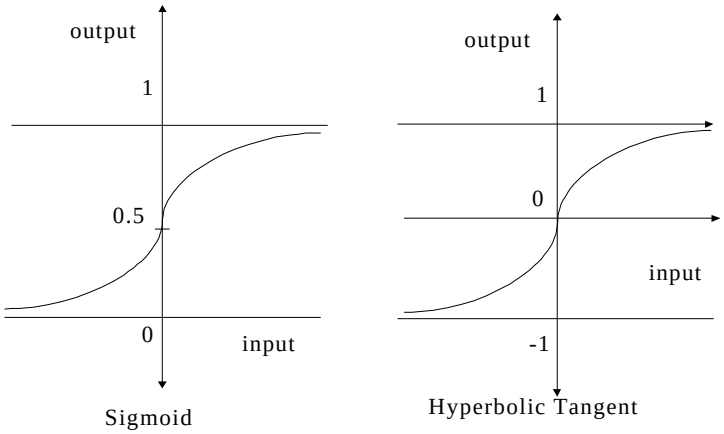

Abb. 2. Graph der sigmoiden und der hyperbolisch-tangenten Funktion

5. Die lineare Funktion hat folgende Form:

$$\text{output}_i = g \cdot \text{input}_i \tag{6}$$

6. Lineare Funktion mit einem Schwellenwert ist eine besondere Form der linearen Funktion

$$output_i = \begin{cases} 0 & \text{wenn } input_i \le T \\ input_i - T & \text{wenn } input_i > T \end{cases} \tag{7}$$

Der Ausgabewert des Neurons ist nur dann nicht gleich 0, wenn dei Eingabe den Schwellenwert *T* erreicht hat.

Die Auswahl der entsprechenden Funktion zur Ermittlung der Ausgabe des Neurons, hängt vor allem vom Design des neuronalen Netzes. Die Person, die das System entwickelt (Systemingenieur), wählt bei der Auswahl des Designs auch die Übertragungsfunktion, die in dieses Design eingebaut worden ist. Der Systemingenieur aber hat immer noch Einfluss auf die Schwellenwerthöhe (*T*) und den Anstieg (*g*). Durch Experimente an einem gewissen Problem scheidet die beste Funktion aus, die verwendet werden sollte.

1.5. Architekturen von neuronalen Netzen

Architekturen der neuronalen Netze unterscheiden sich nach folgenden Parametern:

- Typ des Lernens
- Verbindung zwischen den Ein- und Ausgabedaten
- Anzahl der Schichten
- Schiesssicherheit
- Typ der Verbindung
- Zeiteigenschaften
- Lernzeit

1.5.1. Lerntypen im neuronalen Netz

Mit "Lernen" des Netzes wird der Prozess der Gewinnung von Verbindungsgewichtswerten zwischen Neuronen (Zahedi, F.,1993, S.524) bezeichnet. Gewichte sind ein wichtiger Faktor, der den Eingabewert eines Neurons bestimmt und dadurch auch den Ausgabewert beinflusst.

Lernarten des Netzes:

- beaufsichtigt
- unbeaufsichtigt

Beim beaufsichtigten Lernen besteht der Lerndatensatz aus vergangenen Fällen, für die die Eingaben und Ausgaben, d.h. Resultate, bekannt sind. Das System bekommt ein genaues Ergebnis und es liegt an ihm, die nötigen Gewichte zu bestimmen, um an neuen Fällen eine richtige, unbekannte Ausgabe zu finden.
Da die Eingabedaten in Wirklichkeit unvollständig oder fehlerhaft sein können, besteht der Vorteil von neuronalen Netzen darin, dass sie eine genaue Ausgabe geben können, auch in Fällen, wenn die Eingabedaten nicht ganz mit den Daten identisch sind, an denen das System gelernt hat.

Im Gegenteil davon, sind beim beaufsichtigten Lernen die Ausgaben der Lernfälle nicht bekannt, im Gegensatz zur Eingabe. Diese Lernweise verwendet man oft bei Problemen der Mustererkennung und auf diesem Prinzip ist Kohonens selbstorganisierende Netz aufgebaut.

Beim Verwenden geht das neuronale Netz durch zwei Arbeitsphasen:

1) Lernphase (Trainingsphase)
2) Anwendungsphase (die Phase der Anwendung des neuronalen Netzes an neuen Problemen).

Bei manchen Netzen wird die Lernphase auch in der Anwendungsphase fortgesetzt (on-line Lernen), während bei anderen die Verbindungsgewichte zwischen Neuronen nach dem Abschluss der Lernphase fest sind und sich nicht mehr in der Anwendungsphase ändern.

Lerngleichungen

Eine Lerngleichung des neuronalen Netzes ist eine Formel, die vom System zur Anpassung der Verbindungsgewichte zwischen Neuronen gebraucht wird. Das Lernen verläuft nach vier Haupttypen von Gleichungen oder vier Regeln:

1. Hebbs Regel
2. Delta Regel
3. Verallgemeinerte Delta Regel
4. Kohonens Regel

1. Hebbs Regel

Nach der Hebbs Regel kann die Lerngleichung durch den folgenden Ausdruck beschrieben werden:
"Das Zwischengewicht vom Neuron j zum Neuron i vergrössert sich um die Produktgrösse von zwei Ausgaben" (Zahedi, F., 1993, S.524). (d.h. die Ausgabe des Neurons *j* und Ausgabe des Neurons *i*), oder in der Formel:

$$w_{ji}^{neu} - w_{ji}^{alt} = a \cdot output_j - output_i \qquad (8)$$

Dabei sind:
w_{ji} - Verbindungsgewicht vom Neuron *j* zum Neuron *i*
$output_j$ - Ausgang des Neurons *j*
$output_i$ - Ausgang des Neurons *i*
α - "Lernparameter" oder Lernkoeffizient.

Der Lernparameter α stellt die Lerngeschwindigkeit des Netzes dar, mit Werten in einem Intervall von 0 bis 1. Zu hohe Werte dieses Parameters sind nicht günstig, weil α umgekehrt proportional zur Generalisierungsmöglichkeit des Netzes ist.

Grossberg und Carpenter haben bei ihren Netzen die etwas modifizierte Hebbs Regel gebraucht, wobei sie statt der Ausgabe des Neurons i die Eingabe des Neurons i verwendet haben. Die verallgemeinerte Hebbs Regel enthält in derselben Formel, statt

Ausgabe des Neurons i, die Formulierung gewünschte Ausgabe des Neurons *i* (desoutput).
Eine wichtige Voraussetzung für das Lernen nach dieser Regel ist, dass die Daten in einzelnen Fällen in keiner Korrelation zueinander stehen (d.h. dass sie orthogonal sind), was im Gegenfall-einen der eingebauten Korrelation proportionalen Fehler in der Ausgabe verursachen würde.

2. Delta Regel

Die Delta Regel ist auch als Widrow/Hoffs Regel bekannt, oder als die Regel der kleinsten Durchschnittsquadrate. Sie versucht, die Zielfunktion durch Bestimmung der Gewichtswerte zu optimieren. Das Ziel ist, die Summe der Fehlerquadrate zu minimalisieren. Der Fehler ist als die Differenz zwischen der Istausgaben und der Sollausgabe eines Neurons für die aufgegebene Eingabedaten definiert.

Die Gleichung für diese Regel ist:

$$w_{ji}^{neu} - w_{ji}^{alt} = a \cdot output_j \cdot Fehler_i \qquad (9)$$

Dabei ist:

$$Fehler_i = output_i^{alt} - desoutput_i \qquad (10)$$

3. Verallgemeinerte Delta Regel

Die Regel ist durch die Einführung der Ausgabederivation des Neurons i in die Formel für die Delta Regel hergeleitet worden:

$$w_{ji}^{neu} - w_{ji}^{alt} = a \cdot output_j \cdot Fehler_i \cdot f'(input_i) \qquad (11)$$

Die Regel wird bei nichtlinearen Lernfunktionen verwendet.

4. Kohonens Regel

Diese Regle benutzt Kohonen in seinem selbstorganisierenden Netz. Kohonens neuronales Netz lernt nicht an bekannten Ausgaben, so dass solches Lernen auf einer unterschiedlichen Regel basiert als die vorhergehenden:

$$w_{ji}^{neu} - w_{ji}^{alt} = a \cdot (extinput_i^{neu} - w_{ji}) \qquad (12)$$

wo extinput i = input, das das Neuron i aus der äusseren Umgebung empfängt.

Probleme, die im neuronalen Netz beim Lernen auftauchen können:

- **overfitting** - wenn die Anzahl der Gewichte zu hoch ist, enthält die aproximierende Funktion eine Überzahl von irrelevanten Informatinen und die Generalisierungsfähigkeit verkleinert sich.
- **Lokales Minimum** - wenn auch der kleinste Fehler in einem bestimmten Funktionsteil gefunden wird, gelangt die Funktion auch ins lokale Minimum.

Damit das neuronale Netz an Daten lernen und generalisieren könnte, muss unbedingt die Voraussetzung bestehen, dass "die Anzahl der Trainingsmuster höher oder gleich der Anzahl der Freiheitsgraden im Netz, d.h. der Gewichtsanzahl ist" (Masters, T., 1993).

1.5.2. Verbindung zwischen Eingabe- und Ausgabedaten

Die Verbindung zwischen den Eingabe- und Ausgabedaten kann sein:

- Autoassoziative - Der Eingangsvektor ist dem Ausgangsvektor gleich (der Fall bei Mustererkennung, wo man das Ziel hat am Ausgang die gleichen Daten zu bekommen wie am Eingang)
- Heteroassoziative - Ausgangsvektor unterscheidet sich vom Eingangsvektor

1.5.3. Andere Parameter zur Auswahl der Architektur des neuronalen Netzes

- Anzahl der Schichten: zweischichtige und mehrschichtige Architekturen
- Schiesssicherheit:
    a) deterministische Netze - wenn das Neuron ein bestimmtes Aktivierungsniveau erreicht, sendet es Signale an andere Neurone
    b) stochastische Netze - Abschiessen ist nicht sicher und erfolgt nach Wahrscheinlichkeitsdistribution (Beispiel: Boltzmanmaschine)
- Art der Verbindung:
    a) cross-bar Netze - die Verbindung zwischen zwei Schichten mit feedback
    b) hierarchisch gegliederte Netze - jede Schicht verarbeitet eine Eigenschaft der Eingabedaten (die unteren Schichten einfachere Eigenschaften und die höheren kombinieren einfachere und verarbeiten komplizierte Informationen).
  (Beispiel: Neocognitron Netz, Combinatorial Hypercompression)
- Zeiteigenschaften:

a) statische Netze (empfangen Eingaben mit einem Mal)
b) dynamische Netze (empfangen Eingaben in Zeitabschnitten, auch Raum-Zeitnetze genannt)

- Zeitliche Einteilung des Lernens:

a) batch Lernen - das Netz lernt nur in der Trainingsphase
b) on-line Lernen - das Netz lernt auch in der Anwendungsphase

Einige von den genannten Parametern für die Auswahl der Architektur nach charakteristischen Netzen sind in der Tabelle 1 abgebildet.

Tabelle 1. Architekturen der neuronalen Netze

	Characteristics			
	Type of connection		Learning	
Arhitecture	Inter - layer	Intra - layer	Type of learning	Learning Equation
Two-layered:				
Perceptron	Fully Connected	-	Supervised	$w_{ji}^{new} = w_{ji}^{old} + a(desoutput_i - output_i)$
ADALINE / MADALINE	Fully Connected	-	Unsupervised	$\frac{1}{n}\sum_{i=1}^{n}(desoutput_i - output_i)^2$ min!
Kohonen's	Fully Connected	On-center/ off-surround	Unsupervised	$w_{ji}^{new} - w_{ji}^{old} = a(extinput - w_{ji}^{old})$
Hopfield's	Fully Connected	Recurrent Cross-bar	Unsupervised	$E = -\frac{1}{2}\sum_{j\neq i}^{n}\sum_{i=1}^{n} w_{ji} output_j output_i$ min!
Multi-layered:				
Backpropagation	Hierarchical	Recurrent	Supervised	$w_{ik}^{new} - w_{ik}^{old} = a \cdot output_i \cdot error_k$
Counter-Propagation	Fully Connected Non-hierarchical	Recurrent Cross-bar	Supervised	$w_{ji}^{new} - w_{ji}^{old} = a(extinput - w_{ji}^{old})$ izme|u 1.i 2. sloja
Recurrent Backpropagation	Fully Connected	Recurrent Cross-bar	Unsupervised	$w_{ji}^{new} = w_{ji}^{old} + 2(desoutput_i^t - output_i^t)r_{ji}^t$ where $r_{ji}^t = f'(input_i^t)(output_j^{t-1} + \sum w_{ki}^{old})r_{ki}^{t-1}$
ART mre`e	Resonance Fully Connected	On-center/ Off-surround	Unsupervised	Described in Carpenter, 1987 (in Zahedi, F., 1993)

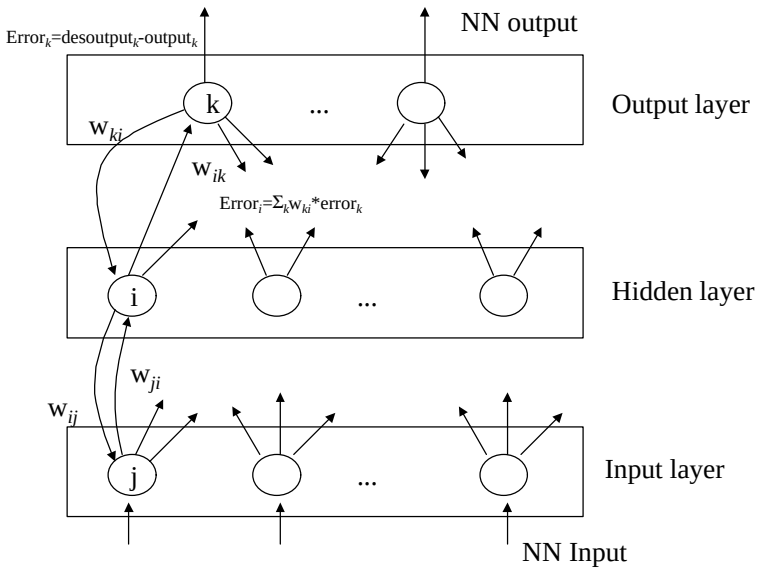

Abb. 3. Grundstruktur des Backpropagation neuronalen Netzes

2. Neuronale Netze in Finanzanlagen

Neuronale Netze finden ihre Anwendung bei Problemlösungen im Finanz- und Investitionsbereich. Unter den häufigsten sind:

- Vorhersage von Rentabilität/Bankrott des Unternehmens
- verschiedene Risikokalkulationen (beim Gewähren hypothekarischer und anderer Kredite, Obligationskurse, u.ä.)
- Vorhersage auf dem Warenmarkt und auf der Wertpapierbörse (Aktienkurse, Warenverkauf, u. ä.)
- Finanzielle Prognosen (Vorhersagen von Zeitreihen)

Fall 1. Chase Manhattan Bank

In der Bank ist ein intelligentes, auf der Integration neuronaler Netze und Expertensysteme basierendes System eingeführt, das die Prognose der Kreditwürdigkeit des Kreditnehmers prüft (der Bürger und von Privatkoporationen in den USA). Neuronale Netze sind zur Vorhersage von Ergebnissen verwendet worden, während die Expertensysteme die Rolle haben, die Ergebnisse zu erläutern und ein Schlussurteil zu fällen. Nach der Einführung hat das System wesentlich Verluste bei gegebenen Krediten verringert und es wird als eine der grössten und erfolgreichsten Anwendungen der künstlichen Intelligenz in den USA angesehen.(Marose, 1990)

Fall 2. Vorhersage von Bankrott des Unternehmens für Peat Marwick

Amerikanische finanzielle, kommerzielle und andere Firmen berichten, dass in einem Jahr (1990) etwa 725.000 Unternehmen bankrott gehen. Die Möglichkeit der Vorhersage ist von grosser Bedeutung für die Stabilität der Wirtschaft. Die Entwicklungsgruppe NeuralWare hat mit Genauigkeit von 90% durch Anwendung ihres Systems prognostiziert, welche von den Banken überleben und welche bis zum Ende des betrachteten Jahres bankrott gehen. Das entwickelte neuronale Netz verwendet Backpropagation Algorithmus, normalisierte, kumulative Delta Lernregel, hyperbolisch-tangentiale Übertragungsfunktion und die Eingangsvariabeln ins Modell sind verschiedene, in der Peat Marwick Gesellschaft kreierte Koeffiziente und Indizes. Die Gruppe Odom und Sharda hat nach Forschung festgestellt, dass die Präzision der Vorhersage mit Hilfe statistischer diskriminanter Analysen beim selben Problem 59.26% ist. (Zahedi, 1993)

Nach bisherigen Untersuchungen in der Anwendung neuronaler Netze in diesem Bereich können folgende Schlussfolgerungen gezogen werden:

- Beim Vergleich neuronaler Netze mit der regressiven Analyse hat sich herausgestellt, dass die neuronalen Netze ein höhere Genauigkeit in Vorhersagen haben als multiple regression, und der Quadratfehler ist geringer bei neuronalen Netzen (Marqüy, Hill, Worthley, Remus, 1991, Dutta, Shekar, 1988). Backpropagation Algorithmus ist benutzt worden.

- In Vorhersage von Zeitreihen haben Backpropagation neuronaler Netze einen geringeren MAPE (Median Absolute Percent Error) Fehler bewiesen als die statistische Box-Jenkins Methode.
- Die am häufigsten benutzten Architekturen des neuronalen Netzes im Finanzwesen und in Anlagen sind das Backpropagation, Recurrent backpropagation und das Hopfields Netz.
- Es bestehen auch negative Erfahrungen in der Anwendung neuronaler Netze bei manchen spezifischen Problemdomänen (Fishwick (in Sharda, Patil, 1992), wo die neuronalen Netze keine grösseree Genauigkeit erwiesen haben als z.B. die einfachen Regressionen.
- Die Möglichkeiten der Anwendung neuronaler Netze sind noch nicht erforscht und ausgenutzt genug und vor allem die Integrationsmöglichkeit mit anderen Methoden der künstlichen Intelligenz (Expertensysteme, Mustererkennung, Verarbeitung der natürlichen Sprache, u.a.)

Beispiel 1. Backpropagation Netz in Profitvorhersage

Profitvorhersage kann als eine einfache Zeitreihe angesehen werden. Die Anwendung neuronaler Netze bei Zeitreihen ist besonders vorteilhaft im Falle von Definierungsschwierigkeiten eines genauen prognostischen Modells, sowie bei Ungewissheit und kaotischen Zuständen der Daten.Die einfachste Struktur des neuronalen Netzes zur Vorhersage von Zeitreihen ist in der Form der schwarzen Box, mit einem oder mehreren Eingabeneuronen und einem Ausgabeneuron. Vom Netz wird erwartet, den Wert eines Punktes im voraus vorherzusehen. Die Muster besteht aus einem Punkt der jetzigen Reihe, mehreren Punkte für vorhergehende Reihen und einem Punkt für die kommende Reihe. Das Modell kann tabellarisch dargestellt werden.

Tabelle 2. Modell einer Zeitreihe zur Profitvorhersage

Eingabedaten (inputs) Vorgegebener Profit	Ausgabedaten (outputs) Vorhergesagter Profit
p_1 p_2 ... p_m	p_{m+1}
p_2 p_3 ... p_{m+1}	p_{m+2}
...	
...	
p_{n-m} $p_{n-(m-1)}$ ... p_{n-1}	p_n

p_i - Profit im Punkt i
m - Anzahl von Eingabeneuronen in jedem Trainingssatz
n - Nummer des zuletzt beobachteten Punktes im Trainingssatz

Im Experiment ist die Einheitsperiode von einer Woche beobachtet worden. Zum Trainieren, bzw. Lernen des Netzes sind zwei Datenreihen benutzt worden: die eine Reihe besteht aus 60 Punkten (Wochen) (m=9, n=6=), während in der zweiten Reihe eine Serie von 110 Punkten (Wochen) (m=9, n=110) beobachtet wurde, und die Resultate

sind verglichen worden. Für die Testphase des Netzes wurde eine zusätzliche Reihe von 20 Punkte (Wochen) benutzt.

Datenvorbereitung umfasst Dateneingabe in den ASCII Ordner, in einer tabellaren Anordnung nach der Tabelle 3.

Tabelle 3. Ein Musterteil der Eingabedaten fürs neuronale Netz

```
51 9 1
 7929.2925  2488.71    4092.7275  4313.9175   4078.785   7742.865   10861.3875  2975.21    2979.67    11190.015
 2488.71    4092.7275  4313.9175  4078.785    7742.865   10861.3875 2975.21     2979.67    11190.015  9080.655
 4092.7275  4313.9175  4078.785   7742.865    10861.3875 2975.21    2979.67     11190.015  9080.655   3543.915
 4313.9175  4078.785   7742.865   10861.3875  2975.21    2979.67    11190.015   9080.655   3543.915   13563.1725
 4078.785   7742.865   10861.3875 2975.21     2979.67    11190.015  9080.655    3543.915   13563.1725 11497.1125
 7742.865   10861.3875 2975.21    2979.67     11190.015  9080.655   3543.915    13563.1725 11497.1125 5847.0525
10861.387   2975.21    2979.67    11190.015   9080.655   3543.915   13563.1725  11497.1125 5847.0525  9039
 2975.21    2979.67    11190.015  9080.655    3543.915   13563.1725 11497.1125  5847.0525  9039       6486.885
 2979.67    11190.015  9080.655   3543.915    13563.1725 11497.1125 5847.0525   9039       6486.885   5367.5475
11190.015   9080.655   3543.915   13563.1725  11497.1125 5847.0525   9039       6486.885   5367.5475  9507.8475
 9080.655   3543.915   13563.1725 11497.1125  5847.0525  9039       6486.885    5367.5475  9507.8475  9776.7075
 3543.915   13563.1725 11497.1125 5847.0525   9039       6486.885   5367.5475   9507.8475  9776.7075  7156.005
13563.172   11497.1125 5847.0525  9039        6486.885   5367.5475  9507.8475   9776.7075  7156.005   8416.785
11497.112   5847.0525  9039       6486.885    5367.5475  9507.8475  9776.7075   7156.005   8416.785   2682.405
 5847.052   9039       6486.885   5367.5475   9507.8475  9776.7075  7156.005    8416.785   2682.405   1416.1125
...
```

Nach einer Analyse der Trendeinflüsse auf die Daten (Im Falle des Bestehens der Einflüsse ist es nötig eine Elimination des Trends durchzuführen, damit das neuronale Netz die empfindlicheren Einflüsse entdeckt), und der Einflüsse saisonbedingter Abweichungen (die in den meisten Fällen beseitigt werden sollten), sind die Daten ins Softwarepaket WinNN einzugeben. Eine backpropagation Architektur ist ausgewählt worden, mit drei Neuronschichten und der Delta Lernregel (sieh Tabelle 1). Lernparameter η, α sind bestimmt, Eingabegeräusch (input noise) und Gewichtsgeräusch (weight noise) und ein so designiertes Netz wurde in Ablauf gesetzt. Nach vielen Iterationen endet die Trainingsphase beim erreichten Fehler 0.01. In der Testphase zeigt der Fehler die Effizienz des Netzes in einer realen Situation, aufgrund zukünftiger Daten. Das Verfahren ist wiederholt worden für eine andere Anzahl von Neuronen in der verborgenen Zwischenschicht und für unterschiedliche Werte der Lernparameter, sowie an verschiedenen Grössen von Datenreihen. Ergebnisse sind in der Tabelle 3 abgebildet.

Tabelle 3. Ergebnisse

Anzahl von der Schichten	Anzahl von Neuronen in einzelnen Schichten (Eingabeschicht - Zwischenschicht - Ausgabeschicht)	Lernparameter	Anzahl Datensa-etzen	RMS[1] test
3	9-5-1	μ=auto, a=0.5, input noise=0,	91	0.45721

[1]RMS Test stellt den Wurzelwert des Durchschnittsfehlerquadrats (Root Mean Square Error). Die Formel zur Ermittlung des RMS Fehlers:

$$RMS\ \ ERR = \sqrt{\frac{1}{n}\sum_{i=0}^{n-1}(t_i - o_i)^2}$$

		weight noise=0		
3	9-5-1	μ=0.2, a=0.5, input noise=0.1, weight noise=0.01	91	0.80154
3	9-9-1	μ=auto, a=0.5, input noise=0.1, weight noise=0.01	51	0.02154
3	9-9-1	μ=0.2, a=0.5, input noise=0, weight noise=0	51	0.00827

Dabei ist η ein Lernparameter, der das Lernen verbessert durch Korrektur der Gewichtswerte nach dem ermittelten Fehlerwert (automatische Veränderung η schwankt zufällig zwischen 0.96 und 1.02), α ist Momentum, Eingangsgeräusch wirft zufällig ein Geräusch ein in jedes Eingabeneuron, was das Netz weniger empfindlich auf Veränderungen der Eingabewerte macht und in Vermeidung des lokalen Minimums hilft. Gewichtsgeräusch wirft zufällig ein Geräusch in jedes Gewicht ein.

Nach dem RMS Fehler und den Ergebnissen in der Tabelle 3, erweist das Netz mit 51 Datensätzen, Neuronstruktur 9-9-1, mit dem Lernparameter η=0.2, α=0.5, input noise=0, weight noise=0 und mit der signum Übertragungsfunktion, die höchste Effizienz.

Als Richtlinien für weitere Forschungen wird das Benutzen der zuusätzlichen Massnahmen zum Bewerten der Effizienz des Netzes (z.B. absoluter Durchschnittsfehler, maximaler Absolutfehler, Medialfehler u.a.) vorgeschlagen, sowie der Gebrauch des Modells der multiplen Zeitreihe, das den realen Erscheinungen in der Wirtschaft näher steht.

Beispiel 2. Vorhersage von Aktienpreisen mit Hilfe neuronaler Netze

Die Forschung (Schöneburg E., 1990) hat gezeigt, dass die statistischen Prognosetechniken befriedigende Ergebnisse geben, in Fällen, wenn mit Aktien in längeren Zeitabständen gehandelt wird (z.B. wöchentlich oder monatlich) mit dem Ziel des maximalen Preisanstiegs. Das Problem indessen, vor dem die Banken, Finanzinstitutionen, grosse Investoren und Broker oft stehen, ist, wie man in möglichst kurzer Zeit kauft und mit relativ kleinem Profit möglichst viele Aktien wiederverkauft. Unter solchen Bedingungen sind die Preisschwankungen zu gering, als dass man sie mit statistischen Prozeduren erkennen könnte, was eine Möglichkeit für die Anwendung der künstlichen Intelligenz, besonders der neuronalen Netze bietet. Mit ihrer Erkennungsfähigkeit der impliziten Abhängigkeiten zwischen Daten und mit der Flexibilität in Vorhersage, können sogar die einfachen Strukturen neuronaler Netze erfolgreiche Prognosen geben.

Schöneburg analysiert einige Grundtypen der Netze:MADALINE, Perceptron und Backpropagation, mit dem Ziel der kurzfristigen Vorhersage von Anstieg und Sturz der Aktienpreise, und vom absoluten Aktienwert für den folgenden Tag. Es sind Daten der drei grossen deutschen Aktiengesellschaften benutzt worden: BASF, COMMERZBANK

und DAIMLER-BENZ. Der Trainingssatz enthällt Daten für eine Zeitperiode von 42 Tagen, während sich die Vorhersagen auf höchstens 56 Tage beziehen.

Die Eingangsvariabeln im Modell sind:

- K - der gegenwärtige Aktientagespreis
- V V - absolute Preisschwankung im Verhältnis zum Vortag
- RV - Schwankungsrichtung V V (Anstieg/Absinken)
- RG - Schwankungsrichtung im Verhältnis zu zwei vorhergehenden Tagen
- G - die grössten Schwankungen im Verhältnis zum Vortag (vece od 1% des Aktienpreises)
- Preise in den letzten 10 Tagen (fürs Backpropagation Netz)

Forschungsergebnisse:

1. Beim ADALINE Netz wurde Anstieg oder Absinken der Aktien für die folgenden 19 Tage vorhergesehen. Bei den Aktien von BASF wurde nach 2500 Iterationen Genauigkeit von 79% erreicht. Aktienkurse der COMMERZBANK wurden nach genau so vielen Iterationen mit maximaler Genauigkeit von 74% vorhergesehen, während bei DAIMLER-BENZ Aktien die niedrigste Genauigkeit (58%) erreicht wurde. Es wurde die Abhängigkeit zwischen Vorhersagefähigkeit und der Zeitentfernung, für die die Vorhersage gemacht wird, gemessen. Der Genauigkeitstrend der Vorhersage hat eine absinkende Richtung im Verhältnis zur Zeitentfernung.

2. MADALINE Netz ist mit unterschiedlicher Struktur im Bezug auf die Anzahl der eingebauten ADALINE Prozessoren in der Zwischenschicht trainiert worden. Optimale Ergebnisse wurden auf Basis von 17 ADALINE Prozessoren erreicht. Genauigkeit der Vorhersage für BASF ist 68%, für die COMMERZBANK 74%, und für DAIMLER-BENZ 63%.

3. Mit dem Perceptron Netz wurde die geringste gesamte Genauigkeit der Vorhersage erreicht, mit einem überraschend hohen Genauigkeitsprozent von 68% für DAIMLER-BENZ Aktien, die alle andere Netzarten am schwächsten prognostiziert haben. Es sind Lernparameter η=0.01 und α=0.05 gebraucht worden.

4. Beim Backpropagation Netz sind verschiedene Strukturen und Übertragungsfunktionen getestet worden und die besten Ergebnisse wurden mit 4 verborgenen Neuronschichten, durch den Gebrauch der linearen Funktion in der Eingabeschicht, der sigmoiden und sinus Funktion in verborgenen Schichten und der sigmoiden in der Ausbageschicht erreicht. Als Lernregel hat man die Delta Regel und die cum - Delta Regel kombiniert. Lernparameter betrugen η =0.6 und α=0.9. Das Netz wies Vorhersage mit zeitlicher Verschiebung von einem Tag auf und entdeckte die Heuristik, mit deren Hilfe man sehr effektiv den Aktienwert für den heutigen Tag vorhersagen kann, indem man den Wert nimmt, den das Netz für den folgenden Tag vorhergesagt hat.

Schöneburg verweist auf weitere Forschung mit zusätzlichen Eingabevariabeln (z.B. Daten über die Aktienpreise anderer Firmen mit dergleichen Tätigkeit oder Branche), auf das Erforschen anderer Netztypen, insbesondere des Kohonens Netzes, das Herausfinden der günstigsten Netzarchitektur zur Vorhersage der Aktienpreise, sowie auf die statistische Analyse der Relevanz der ermittelten Ergebnisse.

Die angeführten Beispiele illustrieren nur einige von den zahlreichen Anwendungsmöglichkeiten neuronaler Netze im Finanzanlagebereich. Die neuere Forschung richtet sich auf holographische neuronale Netze, Anwendung eines stufenartigen (conjugate gradient) Algorithmus mit dem Ziel der Fehlerverringerung, und anderer, meistens kombinierter Verfahren zur Verbesserung der Effizienz. Integriert mit Expertensystemen und anderen Methoden der künstlichen Intelligenz, ist diese Methode sicherlich ein unumgänglicher Teil intelligenter Systeme für die Entscheidungsunterstüzung.

Literatur:

1. Badiru, A., B., The Role of Artificial Intelligence and Expert Systems in New Technologies, in Madu, C. N., ***Management of New Technologies for Global Competitiveness***, Quorum Books, Westport, Connecticut, London, 1993., pp. 301-317.
2. Canarelli, P., Analyzing the Past and Managing the Future using Neural Networks, Futures, Vol.27, No. 3, 1995, pp. 325-338
3. Dutta, S., Shekar, S., Bond Rating: A Non-Conservative Application of Neural Networks, Proceedings of the IEEE International Conference on Neural Networks, July, 1988, pp. II443-450.
4. Hawley, D.D. Johnson, H., raina, D., Arificial Neural Systems: A New Tool for Financial Decision Making, Finacial Analyst Journal, November-December, 1990, pp. 63-72.
5. Jerome, T., Connor, R., Douglas, M., Atlas, L.E., Recurrent Neural Networks and Robust Time Series Prediction, IEEE Transactions on Neural Networks, Vol. 3, No. 2, March 1994., pp. 240-254.
6. Kröse, B. J.A, Van der Smagt, Patrick, P., An Introduction to Neural Networks, Fourth Edition, University of Amsterdam, Faculty of Mathematics and Computer Science, September, 1991.
7. Kryzanovski, L., Galler, M., Wright, D.W., Using Artificial Neural Networks to Pick Stocks, Financial Analyst Journal, July-August, 1993, pp. 21-27.
8. Marose, R., A., A Financial Neural Network Application, AI Expert, May, 1990, pp. 50-53.
9. Marqüz, L., Hill, T., Worthley, R., Remus, W., Neural Network Models as an Altenative to Regression, in Trippi, R.R., Turban, E., Eds., Neural Networks in Finance and Investing, Probus Publishing Company, 1993, pp. 435-450.
10. Masters, T., Practical Neural Network Recipes in C++, Academic Press, Inc., 1993.
11. Mi{ljen~evi}, D., Mar{i}, I., Umjetna inteligencija, [kolska knjiga, Zagreb, 1991.
12. Schöneburg, E., StockPrice Prediction Using Neural Networks: A Project Report, Neurocomputing, 2, 1990, pp. 17-27.

13.Sharda, R., Patil, R.B., A Connectionist Approach to time Series Prediction: An Empirical Test, in Trippi, R.R., Turban, E., Eds., Neural Networks in Finance and Investing, Probus Publishing Company, 1993, pp. 451-464
14.Tanimoto, S.L., The Elements of Artificial Intelligence, An Intrduction Using LISP, Computer Science Press, 1987.
15.Trippi, R.R., Turban, E., Neural Networks in Finance and Investing, Probus Publishing Company, 1993.
16.Wong, F.S., Time Series Forecasting Using Backpropagation Neural Networks,Neurocomputing, 2, 1990/91, pp. 147-159.
17.Zahedi, F., Intelligent Systems for Business, Expert Systems with Neural Networks, Wadsworth Publishing Co., 1993
18.Zaiyong, T., de Almeida, C., Fishwick, P.A., Time Series Forecasting Using Neural Networks vs. Box-Jenkins Methodology, Simulation 57:5, 1991, pp. .303-310.
19.Zeidenberg, M., Neural Network Models in Artificial Intelligence, Ellis Horwood Limited, 1990.