Antisysteme versus Neuronale Netze: Das Prinzip und die Anwendungsbeispiele

Professor, Dr.-Ing. Serge Zacher
Fachhochschule Wiesbaden
FB MND, Am Brückweg 26
D-65428 Rüsselsheim
Tel.: 06142/898-418, Fax: 06142/898-418, Email: s.zacher@web.de

1. Einführung: Vor- und Nachteile von neuronalen Netzwerk-Systemen

Eine der wichtigsten Eigenschaften der künstlichen neuronalen Netzen, die sich heute in allen technisch-wissenschaftlichen Bereichen etabliert haben, ist die sogenannte Lernfähigkeit.
Der Begriff des Neurons als ein logisches Schwellenwertelement mit mehreren Eingängen und einem Ausgang, das nur zwei Zustände annehmen kann, wurde erstmals von den Mathematikern *W.S. McCulloch* und *W. Pitts* [1] eingeführt. Die Jahrzehnte später erfundenen zahlreichen Neuronenmodelle stützten sich im wesentlichen auf dieses Grundmodell. Das Aufkommen von EDV hat die Möglichkeit eröffnet, die Funktion der einzelnen oder in ein Netz verbundenen Neuronen auch mit veränderlichen Parametern zu simulieren.

Die Lernfähigkeit eines Netzes besteht also darin, die eigenen Gewichte so einzustellen, daß der Fehler zwischen Ist- und Sollwert des Netzausgangs für eine bestimmte Klasse der Eingänge möglichst minimal wird. Grundlage dieses Lernverfahrens ist eine Hypothese des Psychologen D. Hebb [2], die besagt, daß das Lernen im Gehirn durch Änderung der Synapsenstärken erfolgt.
„Obwohl diese Hypothese experimentell bis heute nicht bestätigt werden konnte, hat sie - wenn auch oft in veränderter Form - Eingang in die Lernalgorithmen der neuronalen Netztheorien gefunden“, so *Kinnenbrock W.* ([3], Seite 23).
Die nachfolgenden Neuronenforscher aus verschiedenen Fachrichtungen haben entsprechende fachspezifische Analogien zur Netz-Topologie und -Algorithmen mitgebracht, wie z.B., J.J.Hopfield aus der magnetische Verhalten von Festkörpern, B.Widrow aus der Theorie der adaptiver Filtern oder der Nobelpreisträger L.Cooper aus der Physik. Aus der Hebb’sche Lernregel entstanden weitere Modifikationen, wie Delta-Regel (1958), Widrow-Hoff’s-Regel (1960), Oja’s-Regel (1982), sowie die Back- oder Counter-Propagation.

Trotz allen Verbesserungen ist die Lernregel immer noch in ihrer ursprünglichen Form als das Konvergenzziel der Itera-tionen geblieben. Die Eingabe- und Ausgabevektoren in Form einer Datei werden vom Benutzer dem Netz zugeordnet, das Netz konvergiert und „erkennt“ danach die neu angegebenen Eingangssituationen.

- Die Vorteile von Neuronalen Netze:
  a) die Fähigkeit zur Generalisierung
  b) die Lernfähigkeit
  c) die Fehlertoleranz
  d) die Selbsorganisation und die Adaptationsfähigkeit

„Auch neuronale Netzwerke, trotz ihrer vielen Vorteile und hohen Leistungsfähigkeit, Nachteile besitzen, über die nicht immer hinweggewiesen werden kann und die eine denkbare Einsatzmöglichkeit wieder zunichte machen können“, warnt *M. Seraphin* ([4], Seite 163).

- Die Nachteile von Neuronalen Netze:
  a) keine Nachvollziehbarkeit der Ergebnisse
  b) schlechte und nicht gesicherte Konvergierbarkeit des Netzes (z.B., die lokale Minimumstellen der Fehlerfunktion oder die fehlende Abbruchbedingungen des Lernverfahrens)
  c) die hohe Hard- bzw. Softwareanforderungen
  d) kein einheitliches Modell möglich; es fehlt an Entwurfsverfahren sowie an Vergleichsanalyse von verschiedenen KNN-Typen.

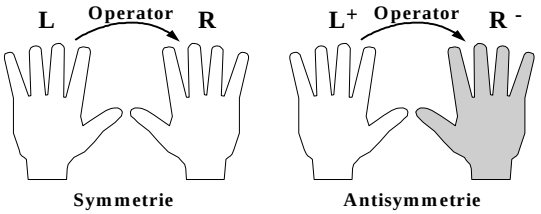
Im folgenden wird ein neues Verfahren des nichtiterativen „Lernen“ den konventionellen Neuronalen Netzen gegenübergestellt und als ein modifiziertes Neuronales Netz realisiert.

2. Antisymmetrie und Antisysteme

2.1. Theoretische Grundlagen

Analog zu den Energie- und Stoffbilanzen, gelten in der Natur auch die Symmetriebilanzen, mit denen mathematische Zusammenhänge in kompakter Form ausgedrückt werden können. Besonders reizend ist die Antisymmetrie, die aus genau den gleichen Symmetrieoperationen, nur mit entgegengesetzten Vorzeichen, besteht.

Das Prinzip der Antisymmetrie, das nach den gleichen Gruppenoperationen der Symmetrie jedoch mit Vertauschung von Variablen in Gegenrichtung funktioniert, ist seit 1929 bekannt. Ein Beispiel der Antisymmetrie [5] ist im Bild 1 gezeigt:



Symmetrie: zwei gleiche Handschuhe für die linke und rechte Hand

Antisymmetrie: ein weißer linker und ein schwarzer rechter Handschuh

Überzeugende Erfolge der Antisymmetrie liefert die Atomphysik, wo jedes Teilchen sein Antiteilchen besitzt (z.B. Elektron und Positron, Proton und Neutron usw.). Jedoch liegen in technischen Bereichen zunächst keine antisymmetrische Darstellung vor.

Bild 1: Beispiel der Antisymmetrie (Handschuhe)

Erste Untersuchungen [6] aus dem Jahr 1968 weisen auf eine Bilanz zwischen Eingangs- und Ausgangsgrößen eines technischen Systems und seines Antisystems hin. Später wurde der Begriff Antisystem als ein System mit gleichen jedoch transponierten Vektoren $\underline{W}^i$ mit Gegenrichtung definiert [7] (Bild 2).

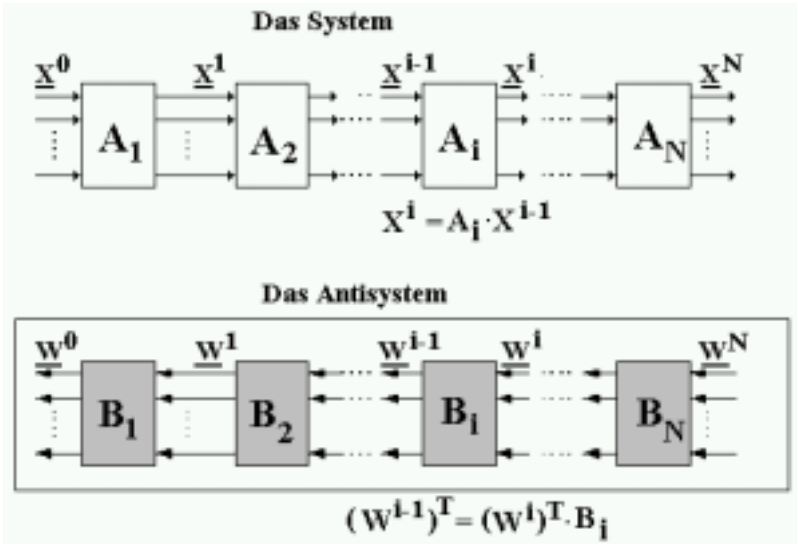


Bild 2: Das Prinzip von Antisystemen

Das Ziel der Antisystemtheorie ist es also, auch bei einem technischen System ein entsprechendes Antisystem zu definieren, um die Vorteile aus den Symmetriebilanzen zu nutzen.

Definition: Ein Antisystem ist System mit Bilanz von Skalarprodukten für alle Ein-/ Ausgangs- und Zwischenvariablen des Paares „System-Antisystem“:

$$(\underline{W}^0)^T \cdot \underline{X}^0 = (\underline{W}^i)^T \cdot \underline{X}^i = (\underline{W}^N)^T \cdot \underline{X}^N$$

Neue Darstellungen bringen neue Qualitäten: die dauerhaften Gewichtsänderungen der konventionellen Neuromodelle werden hier durch die Symmetriebilanzen ersetzt, die Rechenzeit sowie der Speicherbedarf werden wesentlich verringert. Es ist nicht nur möglich, die Antisysteme für konventionelle Einsatzgebiete von neuronalen Netzen, wie z.B., Bilderkennung, Bildkompression oder Fehlererkennung, effektiv einzusetzen, sondern auch ganz neue Anwendungsgebiete, wie z. B., Kryptographie oder Untersuchung von Magnetfeldern, zu eröffnen.

• Die Vorteile von Antisystemen	a) die Nachvollziehbarkeit der Ergebnisse b) „das Lernen“ ohne Iterationen c) keine Hard- bzw. Softwareanforderungen d) ein einheitliches Modell sowie das Entwurfsverfahren möglich

2.2. Ein einführendes Beispiel: N = 3 Blöcke; M = 2 Blöcke; n = 4 Komponenten.

Gegeben ist ein lineares Gleichungssystem:

$$\begin{aligned}
\underline{X}^0 &= \underline{X}^1 + A^1 \underline{Y}^1 \\
\underline{Y}^2 &= \underline{Y}^1 + B^1 \underline{X}^1 \\
\underline{X}^2 &= \underline{X}^1 + A^2 \underline{Y}^2 \\
\underline{Y}^3 &= \underline{Y}^2 + B^3 \underline{Y}^3 \\
\underline{X}^3 &= \underline{X}^2 + A^3 \underline{Y}^3
\end{aligned}$$

mit folgenden Werten $\underline{X}^0 = \begin{vmatrix} 100 \\ 100 \\ 100 \\ 100 \end{vmatrix}$; $\underline{X}^3 = \begin{vmatrix} 10 \\ 10 \\ 10 \\ 10 \end{vmatrix}$;

$$A^1 = \begin{vmatrix} -2 & 0 & 0 & 0 \\ 0 & -4 & 0 & 0 \\ 0 & 0 & -2 & 0 \\ 0 & 0 & 0 & -1 \end{vmatrix}; \qquad A^2 = \begin{vmatrix} -3 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -2 \end{vmatrix}; \qquad A^3 = \begin{vmatrix} -4 & 0 & 0 & 0 \\ 0 & -2 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{vmatrix};$$

$$B^1 = \begin{vmatrix} -0{,}5 & 0{,}5 & 0 & 0 \\ 0{,}5 & -0{,}7 & 0{,}2 & 0 \\ 0 & 0{,}2 & -0{,}8 & 0{,}6 \\ 0 & 0 & 0{,}3 & -0{,}3 \end{vmatrix}; \qquad B^2 = \begin{vmatrix} -0{,}3 & 0{,}3 & 0 & 0 \\ 0{,}3 & -0{,}4 & 0{,}1 & 0 \\ 0 & 0{,}1 & -0{,}3 & 0{,}2 \\ 0 & 0 & 0{,}7 & -0{,}7 \end{vmatrix};$$

Gesucht ist z.B., die 4.Komponente des Vektors $\underline{Y}^1$, die unbekannte Größe ist also Y_4^1.

Nach dem klassischen Verfahren lautet die Lösung: $\underline{Y}^1 = \mathbf{B}^{-1} (\underline{X}^3 - \mathbf{A}\,\underline{X}^0)$,
wobei $\mathbf{A} = A^3 A^2 B^2 B^2 + (A^2 + A^3) B^1 + A^3 B^2 + E$
$\mathbf{B} = A^3 + A^2 + A^1 + (B^1 + B^2) A^1 A^2 + A^3 A^2 B^2 + A^3 A^2 A^1 B^2 B^1$

Statt Matrizen **A** und **B** zu berechnen und danach die inverse Matrix $\mathbf{B}^{-1}$ zu bestimmen, kann man das gegebene lineare Gleichungssystem mit einem Antisystem ergänzen. Die Lösung der dabei entstandenen Gleichung $\underline{Y}^1 = \mathbf{G}^{-1} (\mathbf{W}\,\underline{X}^3 - \mathbf{H}\,\underline{X}^0)$ wird durch Symmetriebilanz zu einer Skalargleichung transformiert:

$$(\underline{w}^1)^T \underline{Y}^1 = (\underline{W}^3)^T \underline{X}^3 - (\underline{W}^0)^T \underline{X}^0$$

wobei $\underline{W}^i$ und $\underline{w}^i$ sind die Antisystemenvariablen. Rechnerisch bestimmt man die Werte

$$\underline{W}^3 = \begin{vmatrix} -0.0845 \\ -0{,}305 \\ -2{,}96 \\ -4{,}34 \end{vmatrix}; \qquad \underline{W}^0 = \begin{vmatrix} 0{,}353 \\ 0{,}278 \\ 2{,}875 \\ -11{,}11 \end{vmatrix}; \qquad \underline{w}^1 = \begin{vmatrix} 0 \\ 0 \\ 0 \\ -27{,}66 \end{vmatrix};$$

daraus folgt die Lösung: $Y_4^1 = 24{,}56$.

Die Zahl der notwendigen Rechenoperationen im Vergleich zu klassischen Methoden reduziert sich um $\dfrac{n + M}{0{,}5N + M}$ mal.

3. Kompromißlösung: Neuronale Netze mit Antineuronen

Die Prinzipien der Symmetrie und Antisymmetrie lassen sich mit den biologischen Prinzipien der Wirkung und Gegenwirkung zu verknüpfen und ein neuer neuronaler Netztyp zu entwickeln.

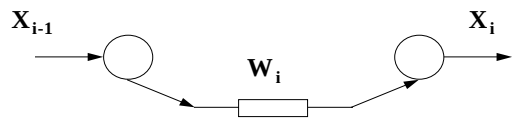


Bild 3: Neuronen

Das Netz besteht aus Neuronen $\mathbf{X_i}$, die mit Gewichten $\mathbf{W_i}$ die Signale in einer Richtung übertragen (Bild 3).
Die Übertragung der Antineuronen $\underline{\mathbf{a_i}}$ wird mit gleiche Gewichten $\underline{\mathbf{W_i}}$ aber in Gegenrichtung durchgeführt (Bild 4).

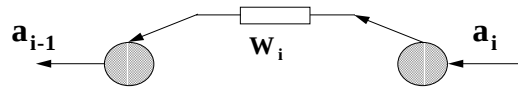


Bild 4: Antineuronen

Da zwischen den beiden Darstellungen nach der Antisystemtheorie eine Bilanz entsteht, erhält die Gesamtenergie des Netzes ihren minimalen Wert (Bild 5).

Ein Netz mit Antineuronen hat folgende Eigenschaften:

- Energiebilanz: Die Energie des Netzes **E** (die Summe der Einzelenergien jeden Paares $\mathbf{e_i}$) erreicht für bestimmte Netzzustände ihre Minima.
- Kompression: Die Netzzustände, die durch Neuronen-/Antineuroneneingänge $\underline{\mathbf{X}}_{\mathbf{i-1}}$ und $\underline{\mathbf{a}}_{\mathbf{i}}$ beschrieben sind, kann man durch Skalar-Energiewerte $\mathbf{e_{i-1}} = (\underline{W}_{i-1})^T \underline{X}_{i-1}$ und $\mathbf{e_i} = (\underline{W}_i)^T \underline{X}_i$ ersetzen.

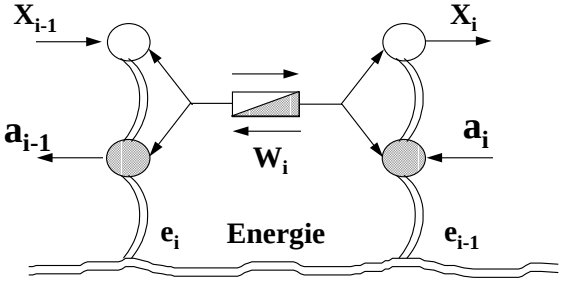


Bild 5: Energiebilanz eines Netzes mit Antineuronen

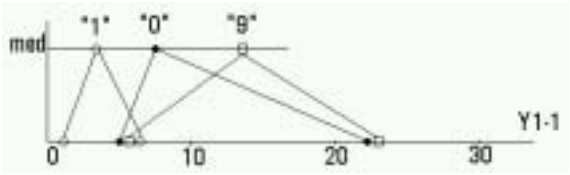
Auf dieser Eingenschaften beruht die Lernfähigkeit des Netzes. Die Gewichtungen von Antineuronen werden über Neuronen von Eingangsmustern übernommen. Die Gesamtenergie des Netzes wird nach dem Symmetriebilanz berechnet (das Lernverfahren) und mit der Musterenergie vergliechen.

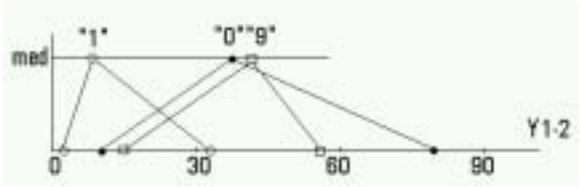
4. Konventionelle Anwendungsgebiete

4.1. Bilderkennung

Betrachtet wird die Handschriftlesung für die Belegerfassung [8]. Das Netz mit Antineuronen wird für die Erkennung von 10 Eingabemustern (Ziffern) trainiert. Eine 160 x 160 Pixel-Muster-Datei wird Stufenweise bis zum Skalarwerte Z1 und Z2 reduziert. Die Klassifizierung und Erkennung erfolgt nach einem nichtiterativen Lernprozess mit Hilfe von 5 Kriterien (als Beispiel sind unten nur 2 Kriterien gezeigt). Die Erkennungsrate für 160 Testdateien erreicht 90%.

Entscheidungskriterien





Skalar-Ebene

Z1 = 72,2 **Z2** = 72,2

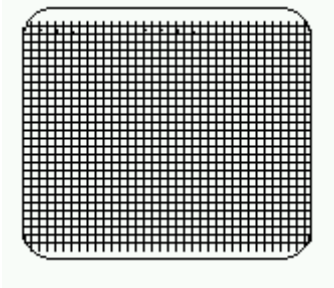
Vektoren - Ebene: 3 x 1 und 1 x 2

Y1 = 4,6 | 28,3 | 39,4 **Y2** = 65 / 6,8

Merkmalen-Ebene: 9 x 6 Elemente, 9 Blöcke

0.072727; 0.000000; 0.000000; 0.618182; **0.018182**; 0.000000; 0.915643; 0.034398**; 0.025306;**
0.012653; **0.012653; 0.177144;** 0.050613; 0.012653; **0.442302; 0.062749**; 0.652606; 0.000000**;**
0.000000; **0.296639**; 0.000000; 0.000000**; 0.224406; 0.951679**; 0.582046; 0.000000; 0.000000;
0.139185; 0.000000**;** 0.000000; 0.088766; 0.867769**; 0.092712; 0.018115**; 0.000000; 0.000000;
0.000000; 0.000000; -0.122884**;** 0.238783; 0.000000; 0.000000; **0.208383; 0.185232**; 0.000000;
0.000000**;** 0.000000; **-0.201908; 0.225175;** 0.000000; 0.000000; 0.597826; **1.045045; 2.840841**;

Pixel-Matrix-Ebene: 160 x 160 Pixel



Für die zur Verfügung gestellten Testdateien und die Anregungen beim Ausprobieren von Demo-Version des Handschriftleser-Programms CLEQS gilt ein Wort des Dankes an die Firma Gentriqs Software GmbH (Eltville/Martinsthal), ganz besonders an den Managing Direktor, Herrn Daniel Gens.

Beispiel: Eingabemuster: Die handgeschriebene Ziffer „**5**“

N1=	160 x 160	Pixel
N2=	9 x 6	Matrix- Elemente
Nb=	9	Block-Matrizen
Ny=	3	Vektor-Komponenten
Nz=	1	Skalarwert

Die 1.Stufe der Klassifizierung (nach Verletzung von Grenzbedingungen):
Mögliche Antworte sind: 0 2 4 5 6 7

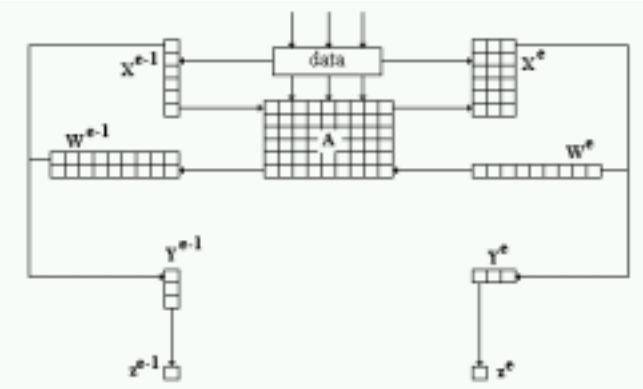
Die 2.Stufe der Klassifizierung (nach 5 Kriterien)

Muster	Abstände				
	Y1-1	Y1-2	Y1-3	Xnull	Z
0	**0,13**	**0,53**	1,15	6	0,06
2	0,24	1,36	2,23	10	1,86
4	1,20	2,01	2,50	11	208,94
5	0,49	0,94	**0,72**	**3**	**0,02**
6	1,27	2,30	1,18	11	1826,57
7	0,75	2,23	3,26	11	40,37
Minimale Abstände:	0	0	5	5	5

Die 3.Stufe der Klassifizierung (Erkennung)

1. Antwort: Ziffer „**5**“
2. Antwort: Ziffer „0“

4.2. Bildkompression



Eine Bildsequenz besteht aus $\mathbf{N_S}$ Bildmatrizen.
Jede (**n** x **m**) Bitmap-Matrix $\mathbf{A^e}$ soll erst komprimiert und danach mit der Nachfolgematrix vergliechen werden.
$\underline{\mathbf{X}}^{\mathbf{e-1}}$ und $\underline{\mathbf{W}}^{\mathbf{e}}$ sind Eingangsverktoren von System und Antisystem (Bild 6).

Die Unterschiede zwischen zwei Matrizen $\mathbf{A^e}$ und $\mathbf{A^{e+1}}$ lassen sich als folgendes bestimmen:

$$\Delta a_{ij}^{e} = \frac{\Delta z}{W_i^e \cdot X_j^{e-1}};$$ dabei wird auch die Position des geänderten Blocks lokalisiert.

Bild 6. Bildkompression mit Antisystemen

Beispiel:

Matrix $\mathbf{A^1}$:

A=	2,4	4,4	5,2	4,8	4,8	2,0	3,6	3,6	2,4
	5,6	4,8	4,0	5,6	3,2	5,6	5,6	4,0	4,0
	4,4	4,8	4,8	4,4	3,6	3,2	2,4	4,8	4,0
	3,6	2,0	4,8	4,0	4,0	4,8	4,4	2,8	3,6
	2,4	3,6	2,0	3,2	4,0	5,6	2,8	4,4	5,6
	3,2	5,2	4,8	5,6	3,2	3,6	3,6	4,8	2,8

Eingangsvektoren $\mathbf{X^0}$ und $\mathbf{W^1}$:

$\mathbf{W^1}$= | 6 | 5 | 4 | 3 | 2 | 1 |

$\mathbf{X^0}$=

1
2
3
4
5
6
7
8
9

Nach dem Lernen entstehen folgende Werte:

$\mathbf{X^1}$=

26,8	55,2	75,6
27,2	72,0	107,2
28,4	54,8	91,2
22,0	64,8	85,6
15,6	66,4	105,2
28,0	60,0	88,8

$\mathbf{W^0}$=

60,0	69,6	70,4	74,4	59,2	52,8	59,2	60,8	50,4
18,8	18,4	23,2	24,0	23,2	29,2	22,4	22,0	24,8

$\mathbf{Y^0}$=

2675,2
1068,4

$\mathbf{Y^1}$=

535,6	1297,6	1910,4

$\mathbf{z}$=

3743,6

Nun ändert sich ist das Element $\mathbf{a_{45}}$ der Matrix $\mathbf{A^2}$, z.B., $\mathbf{a_{45}}_{new}$ = 4,3.
Die komprimierte Werte der Matrix $\mathbf{A^2}$ sind:

$\mathbf{X^1}$=

26,8	55,2	75,6
27,2	72,0	107,2
28,4	54,8	91,2
22,0	**66,3**	85,6
15,6	66,4	105,2
28,0	60,0	88,8

$\mathbf{W^0}$=

60,0	69,6	70,4	74,4	59,2	52,8	59,2	60,8	50,4
18,8	18,4	23,2	24,0	**24,1**	29,2	22,4	22,0	24,8

$\mathbf{Y^0}$=

2675,2
1072,9

$\mathbf{Y^1}$=

535,6	**1302,1**	1910,4

$\mathbf{z}_{new}$ =

3748,1

Das Kriterium $z \neq z_{new}$ signalisiert, daß eine Änderung vorhanden ist. Aus dem weiteren Vergleich von $\mathbf{Y^1}$, $\mathbf{X^0}$, $\mathbf{X^1}$, $\mathbf{W^0}$, $\mathbf{W^1}$ ergibt sich die Blocknummer des Elements $\mathbf{a_{45}}$ und die Abweichung:
$\Delta a_{45} = \frac{4,5}{3 \cdot 5} = 0,3$. Das Element $\mathbf{a_{45}}$ der Matrix $\mathbf{A^2}$ läßt sich damit errechnet werden:

$$a_{45new} = a_{45old} + \Delta a_{45} = 4,0 + 0,3 = 4,3$$

5. Kryptographie: das neue Anwendungsgebiet für Neuronale Netze

Die Anwendung der Neuronalen Netze mit Antineuronen für die Datensicherheit wird anhand den folgenden Turbo-Pascal-Beispiel kurz erläutert. Analog den Public-Key-Verfahren [10], verschlüsselt der Sender eine Nachricht und übersendet sie dem Empfänger zusammen mit einem öffentlichen Schlüssel.
Der öffentlicher Schlüssel ist ein neuronales Netz, in dem das Paßwort nicht gespeichert wird, sondern bei jedem Start nach einem geheimen Schlüssel (gespeicherte Idee des Senders) hergestellt.

```
PROGRAM antineuron;
USES crt;                                              {------Verschlüsselter Text-----}
CONST  vtext : ARRAY[ 1..37 ] OF Integer = ( 75,101,20,64,31,38,77,33,100,50,110,161,65,
154,24,58,35,102,105,87,30,80,74,50,118,121,119,125,193,107,203,102,191,45,198,110,145 );
VAR   a1,b1,c1,d1,a2,b2,c2,d2,a3,b3,c3,d3, i,j,k : Integer;  z:Char;
        a : ARRAY[ 1..4 ] OF Integer;  g : ARRAY[ 1..7 ]  OF Longint;  x : ARRAY[ 0..8 ]  OF Longint;
BEGIN
  FOR i:= 1 TO 7 DO g[i]:= RANDOM(10)+1;
  Randomize;          x[1]:= RANDOM(10)+1;
  Randomize;          x[2]:= RANDOM(10)+5;
  Randomize;          x[3]:= RANDOM(10)+3;
  Randomize;          x[4]:= RANDOM(10)+2;
  FOR i:= 5 TO 8 DO  x[i]:=x[i-1] * g[i-1];
  WriteLn('  Eingänge des Neurons  ');
  FOR i:= 1 TO 8 DO WriteLn(' x[i]=  ',x[i] );       {-------------öffentlicher Schlüssel--------}
  WriteLn('  Die Gewichtungen eingeben:  ');
  FOR i:= 1 TO 7 DO  BEGIN
  Write(' g[',i,']= '); ReadLn(g[i]);END;              {----------------Das Lernen-----------------}
  Randomize;  x[1]:= RANDOM(10)+1;
  Randomize;  x[2]:= RANDOM(10)+5;
```

```
Randomize;  x[3]:= RANDOM(10)+3;
x[4]:= g[1]*x[1]+ g[2]*x[2]+ g[3]*x[3];
FOR i:= 5 TO 8 DO  x[i]:= x[i-1] * g[i-1];   WriteLn('------Geheimzahl eingeben------');
FOR i:= 1 TO 4 DO   BEGIN                    {-------geheimer Gruppenschlüssel------}
z:=UpCase( ReadKey ); a[i]:= Ord(z) - 48;
Write(#95); END; WriteLn(' Danke! '); ReadLn;{---------------  Authentisierung------------}
FOR i:= 1 TO 8 DO
  FOR j:= 1 TO 8 DO
    FOR k:= 1 TO 8 DO
IF x[a[1]] * x[a[1]+1]=x[a[2]] * x[i] +x[a[3]] * x[j] + x[a[4]] * x[k]  THEN
BEGIN                                                {----------------Entschlüsseln-----------}
  d1:=3*9+a[1]+1;   c1:=2*9+k;   b1:=1*9+j;   a1:=0*9+i;   {-------------------Benutzer A----------------}
  d2:=3*8+a[1]+1;   c2:=2*8+k;   b2:=1*8+j;   a2:=0*8+i;   {-------------------Benutzer B----------------}
  d3:=3*10+a[1]+3; c3:=2*10+k; b3:=1*10+j; a3:=0*10+i;  {-------------------Benutzer C----------------}
END;
WriteLn ('-----------------Klartext:---------');
WriteLn; Write('---Benutzer A:------');
FOR i:= 0 TO 3 DO
   Write ( chr (vtext [a1-i]+vtext [b1-i] + vtext [c1-i] - vtext [d1-i]) );
 WriteLn; Write('---Benutzer B:------');
FOR i:= 0 TO 4 DO
   Write ( chr (vtext [a2-i]+vtext[b2-i]+vtext [c2-i] - vtext [d2-i]) );
WriteLn; Write('---Benutzer C:-----');
FOR i:=0 TO 5 DO
Write ( chr (vtext [a3-i]+vtext[b3-i]+vtext [c3-i] - vtext [d3-i]) );
z:=ReadKey;   END.
```

Um den Lernprozess des Netzes zu starten, speist der Benutzer sein neuronales Netz mit den Zufallszahlen. Die Antwort des Netzes x[1],...x[8] weist ihn darauf hin, welche Gewichtungen g[1],..g[7] er nach seinem geheimen Schlüssel eingeben soll. Damit wird eine Geheimzahl hergestellt. Die Anzahl der möglichen Kombinationen ist praktisch unbegrenzt.
Die Benutzer A, B und C erhalten verschiedene Klartexte, z.B.:
der Benutzer A - den Klartext „INFO“,
der Benutzer B - den Klartext „Danke“,
der Benutzer C - den Klartext „Appell“.
Eine mehrstufige Identifizierung ist möglich: ein Schlüssel a[1],..a[4] für eine ganze Benutzergruppe und die individuellen Schlüssel für jeden Benutzer, z.B. a1,b1,c1,d1 für Benutzer A oder a2,b2,c2,d2 für Benutzer B und a3,b3,c3,d3 für Benutzer C.

6. Literatur:

[1] *McCulloch, W.S.* und *Pitts, W.*: A logical calculus of the ideas immanent in nervous activity. Bulletin of Mathematical Biophysics, 5, pp.115-133, 1943.

[2] *Hebb, D.*: The organisation of behavior, Wiley, New York, 1949.

[3] *Kinnenbrock, W.* Neuronale Netze. R.Oldenbourg Verlag München Wien, 1992.

[4] *Seraphin, M.:* Neuronale Netze und Fuzzy-Logik, Franzis-Verlag, 1994.

[5] *Zakharian, S.:* Symmetrie und Antisymmetrie in der Automatisierungstechnik. Kolloquium des Barkhausen-Lehrstuhls, TU Dresden, 27.06.1991.

[6] *Kattanek, S.* und *Sacharian S.:* Berechnung von Übertragungsfunktionen des Festbettreaktors durch Reduzierung des Signalflußbildes. Chem. Technik, XX, Heft 10, S.725-728, 1968.

[7] *Zakharian, S.*: Dynamics of heterogeneous systems. Materials of the seminar. The Institute for System Studies, Moscow, pp.199-210, 1984.

[8] *Zacher, S.* und *Gens,D.:* Handschriftlesung für die Belegerfassung mittels Neuronaler Netze. Anwendersymposium „Aktuelle Entwicklungen und Realisierungen der Bildverarbeitung“, Aachen, Tagungsband, S.77-83, 1997.

[9] *Zakharian, S.* und *Avetisian, A.*: Fault detection by compressed neural networks. IFAC- Simposium „Safeprocess’91“, Baden-Baden, September,1991.

[10] *Beutelspacher, A./ Schwenk, J. / Wolfenstetter, K.-D.:* Moderne Verfahren der Kryptographie. Vieweg, 1995.