Hopfield-Netze

1. Rückgekoppelte Netze

Bislang waren die Neuronen in Schichten angeordnet und (meist) auch nur von Schicht zu Schicht vernetzt. Dabei wanderten die Informationen jeweils in einer Richtung von der Eingabeschicht zur Ausgabeschicht. Bei natürlichen neuronalen Netzen können sich Neuronen aber auch wechselseitig beeinflussen. In Abb. 1 ist ein einfaches künstliches neuronales Netz dargestellt, in dem solche Rückkopplungen auftauchen. So erkennt man z. B. bei der Verbindung zwischen dem Neuron N0 und dem Neuron N1 zwei Synapsen, eine bei N1 und eine weitere bei N0. In Wirklichkeit handelt es sich hier also nicht um eine einzige, sondern um eine doppelte Verbindung mit den Gewichten $g_{0,1}$ und $g_{1,0}$. Wir werden im Folgenden nur solche rückgekoppelten Netze betrachten, bei denen die beiden Gewichte einer solchen doppelten Verbindung jeweils gleich sind. Deswegen findet man in Abb. 1 an den Verbindungslinien jeweils nur einen Gewichtswert.

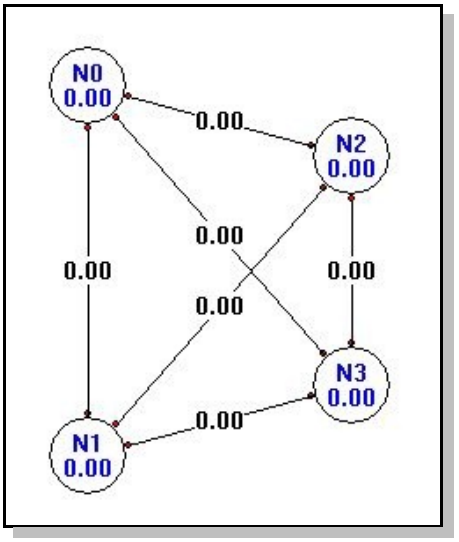


Abbildung 1

2. Muster und Hopfield-Netze (Netz32)

Solchen rückgekoppelten neuronalen Netze können Bildmuster eingeprägt werden; sie können diese dann aus Fragmenten rekonstruieren. Dazu ordnet man jedem Bildpunkt (Pixel) des Bildes ein Neuron zu. Jedes Neuron wird mit jedem *anderen* Neuron durch eine doppelte Verbindung verknüpft. Sinnvollerweise positioniert man diese Neuronen im Netz genauso wie die einzelnen Pixel. Bei größeren Mustern blendet man dabei meist in der Grafik die Verbindungen mit ihren Gewichten aus (vgl. Abb. 2).

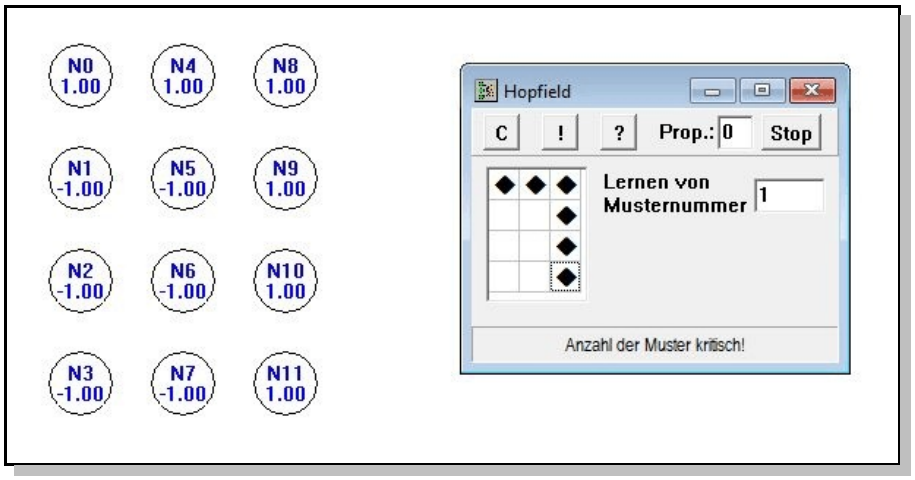


Abbildung 2

Alle Neuronen dienen hier sowohl zur Eingabe als auch zur Ausgabe. Ein schwarzes Pixel stellen wir durch ein aktives Neuron (d. h. mit dem Aktivierungsgrad +1) dar, ein weißes (nicht gesetztes) Pixel durch ein nicht aktiviertes Neuron (d. h. mit dem Aktivierungsgrad -1). Als Aktivierungsfunktion benutzen wir dementsprechend

$$f_{Hopfield}(i) = \begin{cases} +1 & \text{für } i \geq 0 \\ -1 & \text{sonst} \end{cases} .$$

Nach Hopfield besteht der Vorgang der Mustereinprägung darin, die Gewichte nun so einzustellen, dass das Netz sich beim gelernten Muster stabilisiert, wenn es - ausgehend von einem Musterfragment - schrittweise propagiert wird. Dies soll nun an einem einfachen Beispiel genauer untersucht werden.

3. Muster einprägen und rekonstruieren

Als Beispiel benutzen wir das Hopfield-Netz aus Abb. 3. Ihm wurde das rechts daneben dargestellte Muster eingeprägt. Dazu wurden die Gewichte gemäß der **Hebbschen Regel** bestimmt:

$g_{i,j}$ = + 1, wenn Neuron i und Neuron j denselben Pixelwert haben,
$g_{i,j}$ = - 1, wenn Neuron i und Neuron j einen unterschiedlichen Pixelwert haben,
$g_{i,i}$ = 0 (keine Verbindung).

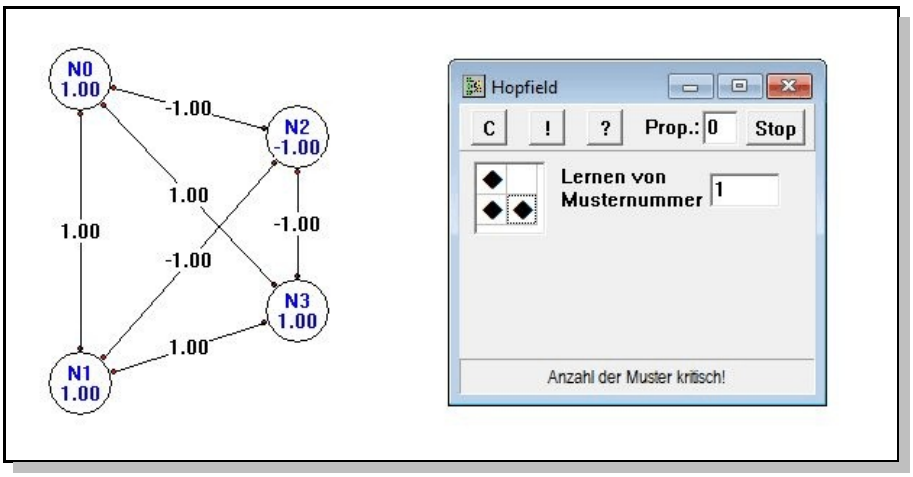


Abbildung 3

In unserem Fall sieht die Gewichtsmatrix G dann so aus: $G = \begin{pmatrix} 0 & +1 & -1 & +1 \\ +1 & 0 & -1 & +1 \\ -1 & -1 & 0 & -1 \\ +1 & +1 & -1 & 0 \end{pmatrix}$

Die Matrix ist wegen der Hebbschen Regel symmetrisch. Die Hauptdialgonalelemente sind alle 0.

Würde man bei dem Netz aus Abb. 3 eine Propagation durchführen, würde es sich nicht ändern: Es stellt einen stabilen Zustand dar. Davon kann man sich leicht durch eine einfache Rechnung überzeugen.

Was geschieht nun, wenn dem trainierten Netz ein anderes Muster präsentiert wird, z. B. das Fragment-Muster aus Abb. 4?

Abb. 4

Wir berechnen einmal den neuen Zustand von Neuron 0:

$$f(g_{1,0}\cdot a_1 + g_{2,0}\cdot a_1 + g_{3,0}\cdot a_3) = f((+1)\cdot(+1)) + (-1)\cdot(-1)) + (+1)\cdot(-1)) = f(+1) = +1$$

Das Neuron N0 hat nach der Propagation also denselben Zustand wie vorher. Gleiches gilt für das Neuron N1.

Anders verhält es sich beim Neuron N3: Die zugehörige Rechnung für den neuen Zustand ist:

$$f(g_{0,3}\cdot a_0 + g_{1,3}\cdot a_1 + g_{2,3}\cdot a_2) = f((+1)\cdot(+1)) + (+1)\cdot(+1)) + (-1)\cdot(-1)) = f(+3) = +1$$

Das Neuron ändert seinen Zustand von -1 auf +1; aus dem weißen Pixel wird ein schwarzes. Damit ist der stabile trainierte Zustand wieder hergestellt.

Unser Hopfield-Netz kann weitere Muster lernen. Die Vorgehensweise ist einfach: Zu dem neuen Muster wird wie oben gezeigt eine weitere Gewichtsmatrix bestimmt. Die erste und die zweite Gewichtsmatrix werden addiert: Die Summenmatrix $G = G_1 + G_2$ kann sowohl Fragmente des ersten als auch des zweiten Musters rekonstruieren. Das schauen wir uns der Einfachheit halber mit unserem Programm Netz32 an.

4. Hopfield-Netze mit Netz32

Das Netz aus Abb. 2 erzeugen wir, indem wir in der Werkzeugleiste die Schaltfläche <speziellen Netztyp wählen> anklicken. In dem Fenster, das sich öffnet, wählen wir die Option <rückgekoppelte Netze> und geben sowohl für die Zahl der horizontalen als auch der vertikalen Pixel jeweils die Zahl 2 ein und bestätigen dies mit <OK>. Nachdem nun das Hopfield-Netz im Netzfenster angezeigt worden ist, öffnen wir mit der Schaltfläche  das Hopfield-Fenster zum Einprägen und Rekonstruieren von Mustern. Hier geben wir als erstes das Muster aus Abb. 3 ein, indem wir die entsprechenden Felder mit der Maus anklicken; ein erneutes Anklicken löscht das gesetzte Pixel. Mit der C-Schaltfläche lässt sich das gesamte Muster löschen.

Damit unser Netz sich dieses Muster einprägt, betätigen wir nun die !-Schaltfläche. Wer mag kann sich die gelernten Gewichte anschauen, indem er im Anzeige-Menü die Darstellung der Gewichte und Verbindungspfeile aktiviert.

Gleich anschließend geben wir das Muster aus Abb. 5 ein und betätigen noch einmal die !-Schaltfläche. Auf die Bemerkung "Anzahl der Muster kritisch!" am unteren Rand des Hopfield-Fensters werden wir weiter unten noch zu sprechen kommen.

Abb. 5

Nun präsentieren wir Muster-Fragmente. Zunächst aktivieren wir nur die beiden linken Pixel und betätigen die ?-Schaltfläche. Schon nach einem einzigen Schritt ist das erste Muster rekonstruiert; die weiteren Schritte führen zu keiner Änderung - das Netz stellt jetzt einen stabilen Zustand dar. Gleiches stellt man fest, wenn als Fragment die unteren beiden Pixel aktiviert werden.

Nun präsentieren wir dem Netz auch auf ähnliche Weise Fragmente, und zwar werden einmal die beiden oberen Pixel aktiviert und einmal die beiden rechten. Diesmal wird unser zweites Muster aus Abb. 5 rekonstruiert. Bei anderen Fragmenten können Probleme auftreten:

- Ein gespeichertes Muster wird nicht erkannt.
- Ein zu einem eingeprägten Muster inverses Muster wird erkannt.

Generell gilt:

- Je mehr Neuronen das Netz besitzt, desto mehr Muster können gelernt werden. Für die Anzahl N_{Muster} der Muster, die in einem Hopfield-Netz recht zuverlässig gespeichert werden können, gilt die Faustregel $N_{Muster} = N_{Neuronen} \cdot 0{,}14$.
- Die Muster sollten in etwa gleich viele weiße wie schwarze Pixel beinhalten.
- Die Muster sollten sich nicht zu ähnlich sein.

5. Experimente mit 4×5-Mustern

Zunächst erstellen wir ein 4×5-Hopfield-Netz und lassen es die 4 Muster aus Abb. 6 lernen.

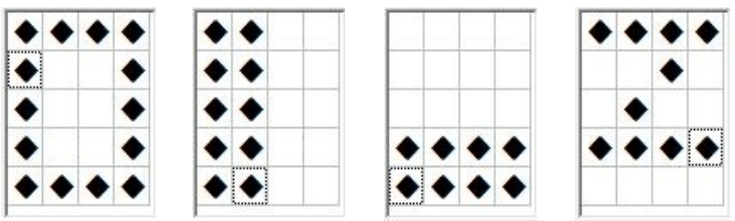


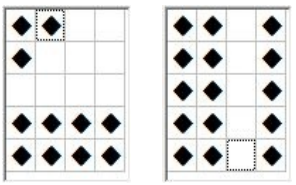
Abbildung 6

Da die Anzahl der Muster hier etwas größer als $4 \cdot 5 \cdot 0{,}14 = 2{,}8$ ist, wird unser Netz nicht unbedingt sehr zuverlässig arbeiten. Um so interessanter ist es, die Grenzen auszuloten.

Man stellt fest:

- Viele Fragmente werden zuverlässig in einem einzigen Schritt erkannt; dies gilt insbesondere, wenn sie dem Original ähneln und aus genügend vielen (aktivierten) Pixeln bestehen. Man beachte, dass ein "Fragment" durchaus auch einige aktivierte Pixel mehr haben kann als das Original.

- Bei manchen Fragmenten sind für die Rekonstruktion zwei oder mehr Schritte nötig. Beispiele für solche Fragmente sind in Abb. 7 angegeben.



Muster 1 Muster 2

Abbildung 7