

Dieser Vortrag soll zeigen, wie sich Genetische Algorithmen und Evolutionstrategien auf das Problem der Optimierung Neuronaler Netze anwenden lassen. Hierzu zunächst eine kurze Einführung in die Theorie dieser Netze.

## Neuronale Netze

Ein typisches Neuronales Netz besteht aus Prozessoren (den *Neuronen*), die durch Informationskanäle verbunden sind. Diese Verbindungen sind durch ihre *Gewichtung* gekennzeichnet, welche den Informationsfluss durch einen Kanal hemmen oder verstärken können. Durch die zeitliche Veränderung der Gewichte soll menschliches Lernen nachgebildet werden. Das hat den Vorteil, dass das System zum Lösen eines bestimmten Problems nicht explizit darauf hin programmiert werden muss. Dem Netzwerk werden lediglich Beispiele des Problems präsentiert und der jeweils gewünschte Output bei einem konkreten Input vorgegeben.

Durch verschiedene Lernalgorithmen passt das Netz seine Gewichte dem Problem entsprechend an und lernt so, Eingaben mit gewissen (gewünschten) Ausgaben zu korrelieren. Das Lernverhalten hängt allerdings entscheidend von der Topologie des Netzes ab. Damit verschiebt sich der Aufwand vom Programmieren hin zum Design eines günstigen Netzes. Allerdings ist es harte Arbeit, ein geeignetes oder gar optimales Netz für ein bestimmtes Problem zu finden, da bis jetzt keine Theorie für die Suche nach "guten" Netzen existiert.

## Der Lösungsansatz

Zunächst müssen die Netzwerktopologien formalisiert werden, da sich die Suche im Raum topologischer Darstellungen als sehr kompliziert erweist. Hierzu wechseln wir in den Raum mathematischer Formeln, die wir nur noch problemspezifisch definieren müssen, um dann die evolutionären Methoden auf sie anzuwenden.

Wir optimieren hierbei nicht nur die Struktur, sondern auch gleich die Gewichtung des Netzwerkes. Als Optimierungs- / Fitnesskriterium verwenden wir den mittleren quadratischen Fehler auf dem Trainingsdatensatz. Die durch den Algorithmus erzeugten Formeln entsprechen dann Neuronalen Netzen, die auf den Trainingsdatensatz hin optimiert, bzw. trainiert wurden.

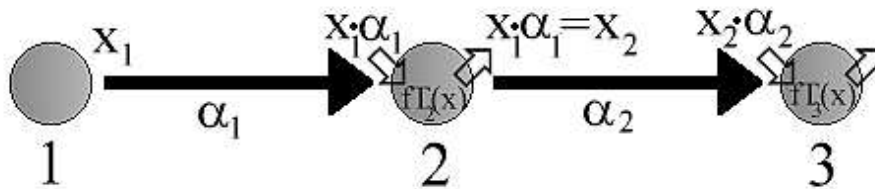
## Die formale Beschreibung der Netze

Das Alphabet von gültigen Formeln, die auch wirklich ein Netz repräsentieren, besteht aus folgenden Operatoren und Variablen:

$\alpha_1, \dots, \alpha_n$  (Gewichtung)  
 $x_1, \dots, x_m$  (Inputvariable)

$fT_i(x)$  (Die Transferfunktion des Neurons, hier meistens  $fT_i(x) = x$ )

Ein Beispiel:



Eine einfache Kette von zwei Neuronen. Hier ist  $fT_i(x) = x$

Die folgende Grammatik bestimmt die *akzeptablen* Formeln:

1. Wenn  $\alpha_i$  Gewicht und  $x_j$  Inputvariable sind, dann heißen alle Ausdrücke der Form:  
 $\alpha_i x_j + \dots + \alpha_k x_m$   
*basic expressions*.
2. Wenn  $\phi$  ein basic expression ist, dann wird  
 $fT_i(\phi)$   
*neuronaler Term* genannt.
3. Wenn  $\phi$  ein basic expression ist und  $\beta_1, \dots, \beta_n$  neuronale Terme sind, dann wird  
 $fT_i(\phi + \alpha_1 \beta_1 + \dots + \alpha_n \beta_n)$   
*terminaler Ausdruck* genannt.
4. Wenn  $\phi$  ein basic expression ist,  $\beta_1, \dots, \beta_n$  neuronale Terme sind und  $\Omega_1, \dots, \Omega_k$  terminale Ausdrücke sind, dann ist  
 $fT_i(\phi + \alpha_1 \beta_1 + \dots + \alpha_n \beta_n + \alpha_j \Omega_1 + \dots + \alpha_l \Omega_k)$   
 ebenfalls ein *terminaler Ausdruck*.

Basic Expressions befinden sich am Eingang eines Netzwerkes, die Eingangsneuronen liefern diese Signale. Neuronale Terme sind von Neuronen verarbeitete Signale, meistens gilt hier aber die Identität  $fT_i(x) = x$ , so dass die Neuronen einfach ihre Aktivierung weiterreichen. Terminale Ausdrücke beinhalten dementsprechend sowohl basic expressions als auch neuronale Terme. (die in unserem Fall auch nur basic expressions sind, da die Transferfunktion die Identitätsfunktion ist)

*Definition:*

Eine Folge von Termen  $\langle \Omega_1, \dots, \Omega_n \rangle$  ist eine *akzeptable* Formel, wenn alle  $\Omega_i$  terminale Ausdrücke sind und jedes  $x_j$  mindestens einmal in einem der  $\Omega_i$  auftritt. Dann kann man jedes der  $\Omega_i$  als ein Ausgangsneuron interpretieren und die Formel entspricht dann der Aktivierung und wie diese von den Eingangssignalen abhängt.

## Die Interpretation der Individuen

... gestattet sich relativ einfach. Die Aktivierung der Eingangsneuronen wird für die  $x_i$  eingesetzt und die Formeln werden von innen nach außen ausgewertet. Somit können wir alle erdenklichen Feed-Forward-Netztopologien auf unseren Formelraum abbilden.

Mit der Fitness haben wir ein Werkzeug an der Hand, mit dem wir die Evolution unserer Netze steuern können. Wie eingangs erwähnt, wird die Fitness vor allem vom mittleren quadratischen Fehler bestimmt. Ein Netz wird getestet, indem alle Muster eines Datensatzes eingegeben werden und der Netzoutput mit dem gewünschten Output verglichen wird. Dann hat die Fitnessfunktion einer Formel  $f$  folgende Gestalt:

$$fit(f) = \frac{1}{n} \sum_{i=1}^n (eval(f, x_i) - y_i)^2$$

wobei  $x_i$  der Eingangs- und  $y_i$  der dazu gehörige Ausgangsvektor sind und  $eval(f, x_i)$  die Formel  $f$  für den Eingangsvektor  $x_i$  auswertet.

Jetzt wäre es evtl. wünschenswert, eine möglichst kurze Formel, also ein einfaches Netz, zu bekommen. Hierzu können wir die Identitätsfunktion nutzen, denn diese wird bei der Evaluation einer Formel einfach weggelassen. Also können wir Individuen, bei denen die Identitätsfunktion öfter auftritt mit einer höheren Fitness belohnen. Falls die Anzahl dieser Vorkommen  $k$  ist, dann wäre folgende Fitnessfunktion denkbar:

$$fit(f) = \frac{1}{n} \sum_{i=1}^n (eval(f, x_i) - y_i)^2 + h(k)$$

wobei  $h(k) = \alpha \cdot k$  oder etwa  $h(k) = 2^k$  sein kann.

Weiterhin kann man, um flache Netzwerke, also mit wenig verdeckten Schichten, zu erzeugen, das wiederholte Anwenden der Transferfunktion mit einer schlechteren Fitness bestrafen. Will man hingegen die Zahl der Neuronen minimieren, muss man lediglich eine hohe Anzahl neuronaler Terme in den Formeln bestrafen. Dasselbe gilt für die Komplexität der Verbindungen, denn wenn man die Anzahl der Gewichtungen bestraft, werden die Gewinnernetze nicht so stark verschachtelt sein wie weniger fitte Individuen. Auf diese Weise kann nahezu jeder Optimierungsgesichtspunkt berücksichtigt werden.