Neuro-Fuzzy-Hybridsysteme

Jan-Hendrik Schleimer

5. August 2002

Zusammenfassung

Neuronale Netze und Fuzzysysteme sind für sich genommen schon sehr interessante Modelle um Informationen zu verarbeiten. Beide Systeme sind stark von der biologischen Wissensverarbeitung inspiriert. In gewisser Weise stehen die Ansätze aber auch konträr an zwei Polen der Modellierungsmethoden. Fuzzy Systeme stellen mit ihren Regeln formal formuliertes linguistisches Wissen eines Menschen bereit, auf dessen Basis Entscheidungen getroffen werden können, aber sie besitzen nicht die Fähigkeit sich anzupassen. Das Paradigma Neuronaler Netze ist es mit ihren Lernalgorithmen zu beliebigen Eingabe und Ausgabe Paaren ihre inneren Parameter so anzugleichen, das sie bei wiederholter Eingabe mit der gewünschten Ausgabe aufwarten können. Dabei sind sie nicht auf das Vorwissen menschlicher Experten, die schon Zusammenhänge in den Daten gefunden haben, angewiesen - sie könnte damit sogar gar nichts anfangen. Und umgekehrt kann ein Experte aus der Netzkonfiguration und den Verbindungsgewichten nicht viel sinnvolles lesen.
Hier sollen die Neuronalen-Fuzzy Hybrid Modelle Abhilfe schaffen in dem sie das beste aus beiden Welten verbinden. Es gibt schon eine Vielzahl an erfolgreichen Kreuzungsversuchen und viele sind sehr interessante Ansätze. Aber man kann auch folgendes in der Literatur lesen:

"Es gibt keine formale einheitliche Definition was ein Neuro-Fuzzy System ist. Man versteht darunter ein System, das in der ein oder anderen Art die beiden Anstze involviert und daraus Nutzen zieht."

Hier also der Versuch dennoch einiges wissenswertes über dieses Forschungsgebiet zusammen zu tragen.

Inhaltsverzeichnis

Kapitel 1

Fuzzy-Logik
Unter dem grauen Schatten

In diesem und in den darauf folgenden Kapiteln werden drei informationsverarbeitende Systeme vorgestellt, die sich deutlich von dem Gebiet der symbolischen künstlichen Intelligenz abheben, allerdings auch zum Forschungsbereich der KI gehören. Die beiden ersten Ansätzen: die Fuzzy-Systeme und die Neuronalen Netze, sind in ihren Eigenschaften verschieden. Neuronale Netze sind von ihrer Struktur her biologischen Neuronen nachempfunden, Fuzzy-Systeme hingegen sind dem menschlichen Schlussfolgerungsprozess nachmodelliert. Das letzte System ist nur ein Zusammengesetztes aus den beiden ersten, aber mit überraschenden Vorzügen.
Zuerst widmen wir uns also den Fuzzy-Systemen. Grundlage für diesen Ansatz ist ein erweiterter Mengenbegriff und darauf aufbauend formale Regeln die Informationen prozessieren.

"Daß von allem entweder immer die Bejahung oder Verneinung war ist, davon muss man wissen." [Aristotels OrganonIV]

"As far as the laws of mathematics refer to reality, they are not certain; as far as they are certain, they do not refer to reality." [Albert Einstein]

"The rolemodel of fuzzy logic is the human mind" [Lotfi A. Zadeh: Soft Computing and Fuzzy Logic. IEEE Software 11(6): 48-56 (1994)]

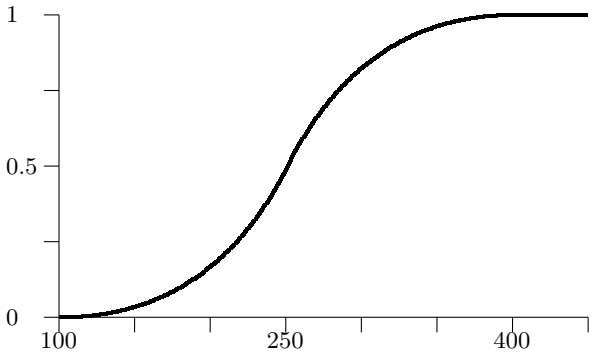
1.1 Was ist Fuzzy

Der alltägliche Gebrauch von Logik ist längst nicht so genau wie es Aristoteles in seinen Werken formuliert. Meißt sind Mengen in unserer Sprache mit Attributen verknüpft: die Menge der *schnellen* Autofahrer, die Menge der *langweiligen* Artikel oder die Menge der *dicken* Bücher. All diese Mengen haben ein ihre Elemente charakterisierendes Adjektiv (schnell, langweilig und dick).
Oft reden wir über eine solche Menge von Objekten, aber machen wir uns immer die Mühe genau darauf zu achten ob wirklich jedes Element genaustens der Definition der Menge entspricht oder ob

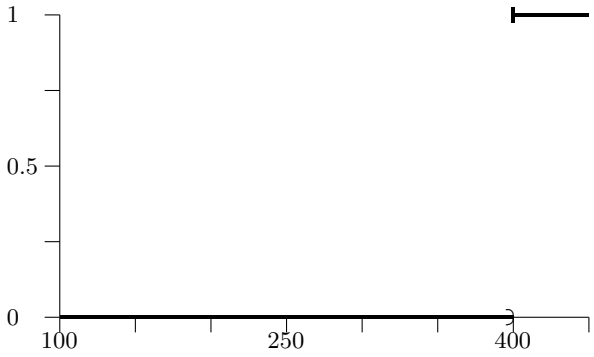
all die Dinge über die wir sprechen gleich stark zu den Mengen zu denen wir sie zusammenschließen gehöhren?
Unterhalten wir uns beispielsweise über *dicke* und *dünne* Bücher. Man könnte festlegen alle Bücher die mehr als 400 Seiten haben sind *dick*. Dennoch wäre man wohl geneigt auch bei einem Buch mit 390 Seiten noch von einem dicken Buch zu sprechen. Die Elemente aus unsere Mengen entsprechen also gar nicht exakt den Definition die sie zu Mitgliedern einer Menge machen, sonder nur mehr oder weniger - eben unscharf. Natürlich sind Bücher mit 350 Seiten wirklich nicht mehr ganz so dick wie diese mit 400, sie sind ein bisschen weniger (oder zu einem gewissen Grad) *dick*. Etwas mathematischer könnten wir also allen Büchern mit zunehmender Seitenzahl gradiel eine Zugehörigkeit zur Menge der *dicken Bücher* geben und genau so sind auch Fuzzy-Mengen definiert:

$$\begin{aligned} A &= \text{dicke Bücher} \\ \mu_A &: X \to [0,1] \end{aligned}$$

X ist hier die Übermenge, also in unserem Beispiel alle Bücher. Diese *Zugehörigkeitsfunktion* bildet in das Einheitsintervall ab, dabei sei 1 die komplette Zugeörigkeit zur Menge und 0 gar keine Zugehörigkeit, wie man es aus der klassischen Logik gewöhnt ist, allerdings sind auch alle Werte dazwischen erlaubt.
Die folgenden Graphen zeigen auf der einen Seite eine solche, im Beispiel sigmoide, stetige Zugehörigkeitsfunktion und auf der anderen Seite eine charakteristische (Indikator) Funktion, die die mengenzugehörigkeit in der klassischen Logik beschreibt.



[Abb.1] Fuzzy-Logik:
Zugehörigkeitsfunktion $\mu_A(x) : X \to [0,1]$



[Abb. 2] Klassische Logik: $I_A(x) = \begin{cases} 1 & : \quad x \in A \\ 0 & : \quad x \notin A \end{cases}$

Das Buch aus dem dieses Beispiel stammt [Demant93] hat seinersteits etwas über 150 Seiten und wäre somit nach unserer Definition vielleicht zu 6% *dick*. Als Zugehörigkeitsfunktionen kommen oft parametrisierte Dreiecksfunktionen (1.1) oder Gaußscheglockenkurven (1.2) zum Einsatz, denn es müssen keine nur monoton steigenden Funktionen sein wie in unserem Beispiel.

$$\mu_{m,d}(x) = \begin{cases} 1 - \left|\frac{m-x}{d}\right| & : \quad \text{falls } m-d \le x \le m+d \\ 0 & : \quad \text{falls } x < m-d \text{ oder } x > m+d \end{cases} \tag{1.1}$$

$$\mu_{m,d}(x) = e^{-a(x-m)^2} \qquad a > 0, m \in \mathbb{R} \tag{1.2}$$

Wobei m der Wert mit der maximalen Zugehörigkeit ist und d bei der Dreiecksfunktion die Distanz zum m angibt, in der die Zugehörigkeit 0 ist.
Besonderes Augenmerk sollte man noch dem Wert $\mu_A(x) = \frac{1}{2}$ der Zugehörigkeitsfunktion schenken. Elemente mit einem solchen Zugehörigkeitswert sind genauso Mitglieder der Menge wie auch Mitglieder der komplementären Menge, deren Definiton im nächsten Paragraphen nachgelesen werden kann.
Es sei noch erwähnt das man auch Zugehörigkeitsfunktionen in mehrdimensionaler Form mit $\mu_A(x^n) : X^n \to [0,1]^n$ betrachten kann. Man veranschaulicht die Elemente der Mengen als n-dimensionale Attributsverktoren deren Zugehörigkeitsgrade in einem gleich dimensionierten Einheitshyperkubus repräseniert sind. Die Endpunkte dieser Kuben sind wieder die gewöhnlichen Mengen. Man kann also die gewöhnlichen Mengen als einen Spezialfall der Fuzzymengen ansehen.

1.2 Fuzzy Mathematik

Der Beginn der Fuzzy Logik liegt schon in den 20er und 30er Jahren des letzten Jahrhunderts, mit dem Beginn der Quanten Theorie. Erst wurde nebst den bekannten WAHR(1) und FALSCH(0) noch ein weiterer Wert $\frac{1}{2}$ hinzugefügt und später dann das ganze Intervall $[0,1] \subset \mathbb{R}$.
Systematisch eingeführt wurde die Fuzzy-Logik 1965 von *Lotfi Zadeh*.

1.2.1 Mengenoperationen und Relationen

In der Einfürung haben wir gesehen das man Fuzzy Mengen über ihre Zugehörigkeitsfunktionen beschreiben kann. Mit Mengen alleine kann man aber noch nicht viel anfangen. Will man Sachverhalten modellieren, muss man auch grundlegende Mengenoperationen und Relationen auf den Mengen definieren.

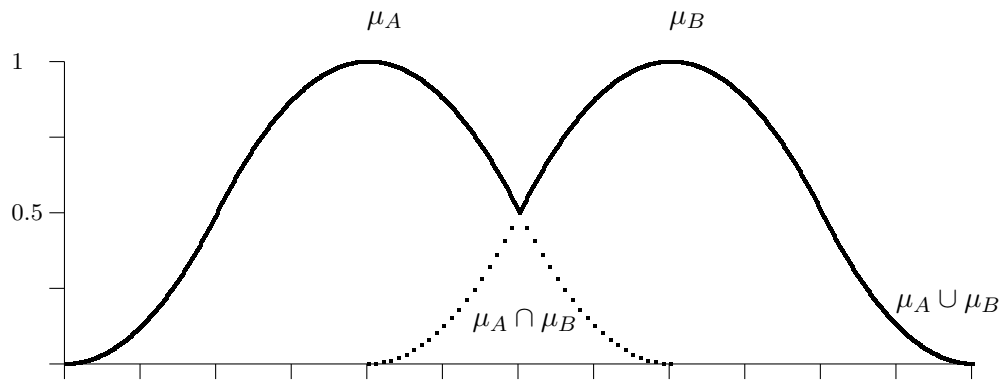
Die Vereinigungsmenge zweier Fuzzy-mengen μ_A und μ_B wird durch das punktweise Maximum der Zugehörigkeitsfunktionen definiert:

$$\mu_A \cup \mu_B := \max(\mu_A, \mu_B) \tag{1.3}$$

Und analog der Durchschnitt:

$$\mu_A \cap \mu_B := \min(\mu_A, \mu_B) \tag{1.4}$$

Mit min ist hier wieder das punktweise Minimum der beiden Zugehörigkeitsfuntionen gemeint. $\cap$ und $\cup$ sind Funktionen von $[0,1]^2 \to [0,1]$ man nennt sie auch t-Norm und t-Conorm. Die Bedeutung dieser beiden Formeln ist intuitiv und läßt sich graphisch leicht veranschaulichen (siehe [Abb. 3]).



[Abb.3] Schnitt und Vereinigung

Die neue Zugehörigkeitsfunktionen für die Vereinigung ist die komplette Durchgezogene Linie, mit einer Unstetigkeit Stelle in der Mitte und die gepunktete Linie ist die neue Zugehörigkeitsfunktion der Schnittfuzzymenge.

Zur Komplementbildung verwendet man folgende Definition:

$$\overline{\mu}_A \quad := \quad 1 - \mu_A \tag{1.5}$$

Geometrisch wäre dies eine Spiegelung des Funktionswertes an einer Parallelen im Abstand 0.5 zur Abszisse.
Es gelten folgende Regeln der bekannten Mengenlehre:

$$\mu_A \cap 1 = \mu_A \quad \text{Einselement} \tag{1.6}$$
$$\mu_A \cap \mu_B = \mu_B \cap \mu_A \quad \text{Kommutativität} \tag{1.7}$$
$$\mu_A \cup \mu_B = \mu_B \cup \mu_A \quad \text{"} \tag{1.8}$$
$$(\mu_A \cap \mu_B) \cap \mu_C = \mu_A \cap (\mu_B \cap \mu_C) \quad \text{Assoziativität} \tag{1.9}$$
$$(\mu_A \cup \mu_B) \cup \mu_C = \mu_A \cup (\mu_B \cup \mu_C) \quad \text{"} \tag{1.10}$$
$$\mu_A \cap (\mu_B \cup \mu_C) = (\mu_A \cap \mu_B) \cup (\mu_A \cap \mu_C) \quad \text{Distributivität} \tag{1.11}$$
$$\mu_A \cup (\mu_B \cap \mu_C) = (\mu_A \cup \mu_B) \cap (\mu_A \cup \mu_C) \quad \text{"} \tag{1.12}$$

Auch die DeMorganschen Sätze lassen sich für Fuzzy-regeln mit Hilfe obiger Regeln beweisen.

Zwei bekannte Eigenschaften klassischer Mengen sind bei Fuzzy-Mengen nicht gültig. Der berümte Ausschluss des dritten "Tertium non datur" gilt beispielsweise nicht - schade Aristoteles! Schneidet man eine unscharfe Menge mit ihrem komplement so muss die entstandene Menge nicht leer sein:

$$\mu_A \cap \overline{\mu}_A \neq \emptyset$$

umgekehrt ist auch die Vereinigung zweier komplementärer Mengen nicht gleich der Übermenge. $\mu_A \cup \overline{\mu}_A \neq X$.

[Abb. 3] zeigt aber auch das man auf einer Übermenge mehrere Fuzzy-mengen definieren kann indem man einfach verschiedene Zugehörigkeitsfunktionen wählt. Dies ist im Hinblick auf die Erstellung eines Fuzzy-Reglers wichtig, denn es sollte darauf geachtet werden das der gesamte Eingabebereich in Fuzzy-mengen partitioniert wird, um es dem System zu ermöglichen auf alle Eingaben zu reagieren, dazu aber später mehr.

1.2.2 Wie groß ist eine Fuzzy Menge?

Die Kardinalität einer Fuzzy-menge (σ-Count) zu bestimmen ist nicht ganz so einfach wie bei herkömmlichen Mengen. Nicht jedes Element gehört gleich stark zur Menge und somit darf auch nicht jedes Element die Kardinalität gleich erhöhen, wie es in der klassischen Mengenlehre der Fall ist. Der Zugehörigkeitswert muß also berücksichtigt werden um den Prinzipien der Fuzzy Logik treu zu bleiben. Die Kardinalität von Fuzzy-mengen gleicht der aus der Informatik bekannten *Hamming Norm* (ℓ^1).

$$\#(\mu_A) \quad = \quad \sum_{x \in X} \mu_A(x) = \ell^1 \tag{1.13}$$

Sie existiert für alle endlichen Mengen. Damit wäre die Menge der *dicken* Bürcher eines Antiquariats auch wirklich größer als die in einem Zeitschriften Laden, auch wenn die Zahl der Zeitschriften der der Bücher gleichen würde (also jeweils eine gleich mächtige Übermenge X zugrundeläge).

1.2.3 Untermengen

Da wir nun wissen wie man Fuzzy-mengen abzählt ist es möglich eine weitere Relation in der zwei Fuzzy-mengen zu einer stehen können zu definierten. Auch die Eigenschaft der Menge μ_A Untermenge von μ_B zu sein soll unscharf beziehungsweise graduell sein. Man drückt es durch den Quotienten aus der Mächtigkeit des Schnittes von μ_A und μ_B, und der Mächtigkeit von μ_A selbst, aus.

$$\mu_A \subset \mu_B \quad = \quad \frac{\#(\mu_A \cap \mu_B)}{\#(\mu_A)} \tag{1.14}$$

Man kann es sich auch als Prozentsatz der Elemente aus μ_A die auch im Schnitt mit μ_B liegen vorstellen. Diese Formel erinnert an die Definition von bedingter Wahrscheinlichkeit aus der Stochastik, wo das Eintreten von X bedingt von Y definiert ist als $P(X|Y) = \frac{P(X) \cap P(Y)}{P(X)}$.

1.3 Fuzzy Regeln

Wollen wir unserem Ziel, einem informationsverarbeitenden Fuzzy-System, näher kommen brauchen wir noch einen weiteren Schritt: etwas mit dem man Wissen beschreiben, festhalten und verarbeiten kann. Dazu können Regeln dienen. Bei einen Fuzzy-System sind es natürlich Fuzzy Regeln, mit denen man aus unscharfen Fakten und unscharfen Regeln unscharfe Konklusionen ziehen kann. Man nennt solche Regeln auch *linguistische Kontrollregeln* und man kann sie sich am besten als "Daumenregeln" vorstellen, die für einen ungefähren Sachverhalt eine ungenaue Reaktion empfehlen.

Man unterscheidet in der Praxis zwei Typen von Fuzzy Regeln.

Mamdani-Regeln: Sie haben folgende allgemeine Form:

$$R^{(i)}\text{: IF } \xi_1 \text{ is } A_1^{(i)} \text{ and ... and } \underbrace{\xi_n}_{\substack{\text{linguistische}\\ \text{Variable}}} \text{ is } \underbrace{A_n^{(i)}}_{\substack{\text{linguistischer}\\ \text{Term}}} \text{ THEN } \eta \text{ is } B^{(i)}$$

Es sind formalisierte Regeln, die an IF-Abfragen einer Programmiersprache erinnern, bestehend aus einer durch IF und THEN eingeschlossenen Prämisse und der dem THEN folgenden Konklusion.
A_1 bis A_n sind *linguistische Variablen*, sie werden mit den *linguistische Terme* ξ_1 bis ξ_n verglichen.
Was aber genau ist ein *linguistischer Term*? In der Literatur kann man die anschlieende Beschreibung finden: "Linguistische Variablen (Terme) dienen insbesondere dazu, linguistisch formuliertes Wissen (z.B. von Experten, wenn man ein Expertensystem modellieren möchte) mit all seinen Unschärfen angemessen in formale Sprachen zu übersetzen, wobei möglichst wenig des Reichtums menschlicher Sprache verloren gehen und gleichzeitig das Wissen mit rechnerbasierten Informationssystemen verarbeitet werden soll." [Maye93, 62;vgl. auch Zimm93,90]
Es handelt sich also um eine sprachliche Bezeichnung für technische Größen. Wie aber kann man diese Begriffe für einen Computer greifbar machen? Das ist der Punkt wo unsere Fuzzy-Mengen von vorher ins Spiel kommen.
Am einfachsten erklärt man es direkt an einem Beispiel. Will man den Experten Lokführer simulieren muss man eine ganzen Regelsatz zur Steuerung von Zügen zusammentragen. Wir greifen einfach eine solche Regel heraus:
"Wenn der Zug dem Ziel *nah* ist und die Geschwindigkeit *hoch* ist, dann ist *stark* zu bremsen."
Seien die unterstrichenen Worte Ziel, Geschwindigkeit und bremsen *linguistisch Variablen* und die kursiv gedruckten Worte *linguistische Terme*. Linguistische Terme sind also Attribute wie im Eingangsbeispiel das Merkmal *dick* der Bücher und können gleichsam durch Zugehörigkeitsfunktionen ausgedrücktwerden. Man ordnet beispielsweise über eine Zugehörigkeitsfunktion allen Geschwindigkeiten ihren Grad der Eigenschaft *hoch* zu sein zu. Folglich ergeben sich für alle n Paare aus Termen und Variablen n Zugehörigkeitswerte, all diese sind dann in der Prämisse UND-Verknüpft, was man als Schnitt-menge der n Fuzzy-mengen realisieren kann. Insgesamt ergibt sich dann ein Erfülltheitsgrad α_i der Prämisse, dem entsprechen dann auch die Konklusion generiert wird. Das heisst η_i bekommt einen zugehörigkeitsgrad (einen linguistischen Term) zugewiesen, der wiederum für ein bestimmtes Attribut steht.

Sugeno-Regeln Sie unterscheiden sich von den Mamdani-Regen nur in sofern, das ihre Konklusionen nicht mehr als unscharfer Werte angegeben werden sondern reellwertige Zahlen sind, die die Funktion f in Abhängigkeit der linguistischen Variablen generiert.

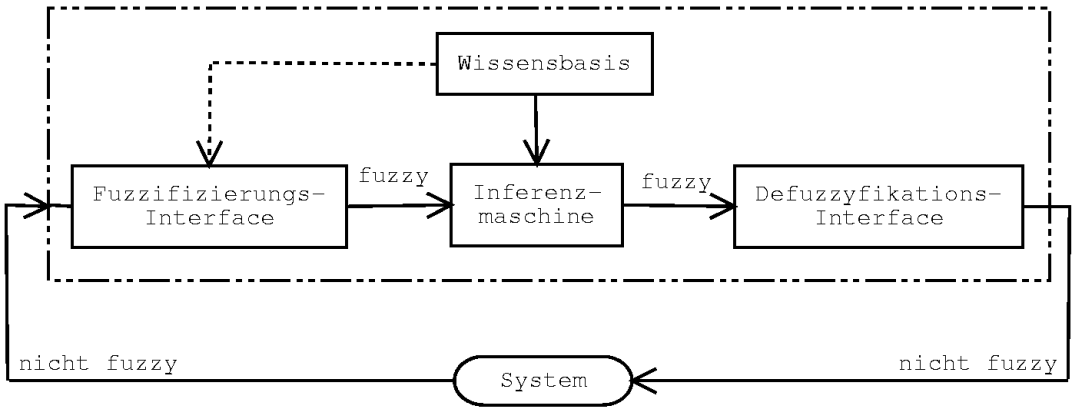
$$R^{(i)}\text{: IF } \xi_1 \text{ is } A_1^{(i)} \text{ and ... and } \underbrace{\xi_n}_{\substack{\text{linguistische}\\ \text{Variable}}} \text{ is } \underbrace{A_n^{(i)}}_{\substack{\text{linguistischer}\\ \text{Term}}} \text{ then } \eta = f_i(\xi_1, ..., \xi_n).$$

Dieser Typ Regeln ist für manche Aufgaben geeigneter, insbesondere wenn es um einen Regler geht, der numerische Stellgrößen bestimmen soll.

1.4 Fuzzy-Systeme

1.4.1 Am Modell

Das Ziel ist eine kognitive Modellierung eines Experten, zur Lösung komplexer Probleme. Das können einerseits die Reglung dynamischer Systeme (Inverse Pendel, Motoren, Atomkraftwerke) bedeuten, es kann sich aber auch Entscheidungsvorgänge und Beurteilungen handeln, wie z.B. der Einschätzung des Börsenverlaufs, Entscheidung über Kreditvergabe oder medizinische Diagnosevorschläge.
Dies kann man unter Zuhilfenahme von Fuzzy-Systeme bewerkstelligen. Wir stellen uns diese zunächst einmal als einen Kasten mit einer Eingabemöglichkeit und einer daraufhin erfolgenden Ausgabe vor. Handelt es sich um ein Regelungssystem, so kann man sich den Input und Output als *Ursache* und *Wirkung*, *Meßgröße* und *Stellgrösse* oder *Stimulus* und *Effekt* vorstellen. Simuliert man hingegen ein Expertensystem so ist es vielleicht eher mit *Frage* und *Antwort* zu vergleichen.
[Abb. 4] zeigt ein allgemeines Modell eines solchen System



[Abb. 4]

Modell eines Fuzzy-Systems

Jetzt schauen wir in den gestrichelten Kasten, in dem sich der Regler befindet, und folgen dem Eingabewert durch die Maschine. Zu erst kommt die scharfe Eingabe in das *Fuzzifizierungsinterface*. Dort werden den numerischen Meßwerten, linguistische Werte, beziehungsweise über Zugehörigkeitsfunktionen Zugehörigkeitswerte der Fuzzy-Mengen zugeordnet. Man kann also in Bezug auf die Fuzzy Regeln aus dem vorherigen Kapitel sagen, das hier die linguistischen Variablen der Prämisse mit Werten gefüllt werden. Für die Prämisse bestimmt dann im zweiten Schritt die *Inferenzmaschiene*, welche die Entscheidungslogik enthällt, einen Aktivierungsgrad α_i, indem sie die Zugehörigkeiten der einzelnen Meßwerte zu ihren Fuzzy-mengen konjuktiv verknüpt, und generiert in Abhängigkeit davon eine neue Stellgröße als Konklusion. Sie stützt sich auf Informationen aus der Wissensbasis, zum Beispiel die gültigen Wertebereiche für Meß- und Stellgrößen.
In [Abb. 4] ist ein Regler vom Typ des Mamdani-Regler visualisiert. Bei diesen ist, wie wir schon aus den zugrunde liegenden Regeln wissen, das Ergebnis der Entscheidungslogik unscharf. Es bedarf also eines dritten Schritts in dem aus diesen Fuzzy-Werten eine reellwertige Stellgröße generiert wird, denn das zu regelnde System kann in den seltensten Fällen mit Fuzzy-Werten arbeiten. Dies übernimmt das *Defuzzyfizierungsinterface*. Dabei gibt es verschiedene Vorgehensweisen [Nauck96]:

- Mean of Maximum (MOM)
  Man mittelt über alle Werte die für die, durch die Regel erzeugte Zugehörigkeitsfunktion der Konklusion maximalen Zugehörigkeitsgrad haben.

- Schwerpunktsmethode (Center-of-Graphity, COG)
  Mittelt über alle möglichen Ausgabewerte, unter Berücksichtigung ihrer Zugehörigkeitsgrade.

Bei einem Sugeno-Regler entfällt die Defuzzifizierung da die zugrunde liegenden Regeln schon eine scharfe Konklusion liefern.

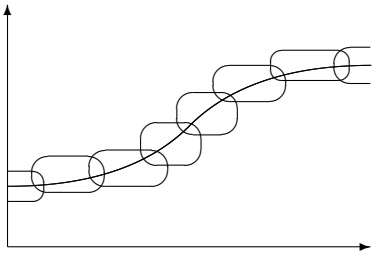
1.4.2 Die Wirklichkeit und ihre Beispiele

Das Interesse an Fuzzy-Systemen bekam sicher nochmals einen Aufschwung, nachdem sie erfolgreich in japanische Konsumprodukte (z.B. Waschmaschine, Camcorder, Autos) eingebaut wurden und seit dem auch wirtschaftlichen Einfluß haben. Es gibt noch ein ganze Reihe weiterer Gebiete in denen sie Einzug erhielten. Zum Beispiel in Ampelanlagen zur Verkehsrkontrolle in Großstädten, wo es kaum möglich ist ein schlüssiges mathematisches Modell zu finden, was alle Reaktionen eines Verkehrsleitsystems auf die jeweilige Verkehrssituation beschreiben kann. Bei einem Fuzzy-System benötigt man nur eine etwas größere Anzahl an auf Erfahrung basierenden Regeln.
Ein weiteres Beispiel welches zeigt das unsere Regel des Zugführes gar nicht so weit hergeholt war, ist das U-Bahn System der Stadt Sendai in Japan. Es wird durch ein System mit 59 Regeln gesteuert, dabei werden verschiedene Aufgaben wahrgenommen. Einerseits Beschleunigen auf gewünschte Fahrgeschwindigkeiten, zeitiges abbremsen in Bahnhöfen und Geschwindigkeitsanpassung bei der Fahrt durch Kurven, abhänig vom Neigungswinkel. Um all diese Regeln aufeinander abzustimmen brauchten die Fuzzy-Ingenieure 2 Jahre.

1.4.3 Bewertung von Fuzzy-Systemen

Wir haben nun einen kleinen Überblick über die Funktionsweise und das Einsatzgebiet von Fuzzy-Systemen, dies sollte es uns ermöglichen zu erkennen wo die Stärken dieses Ansatzes liegen und auch wo seine Schwächen sind.
Angedeutet am Beispiel der Verkehrskontrolle, sei hier noch einmal genannt das man zur Modellierung eines Reglers als Fuzzy-Maschine nicht über ein genaues mathematisches Modell verfügen muss. Natürlich ist es möglich fast alles mit einem riesigen Gebäude aus Differenzialgleichungen zu erschlagen, nur kann dies ja nach Art des Problems unwahrscheinlich kompliziert werden. In Gebieten, in denen menschlicher "common sense" gefragt ist, um eine geeignete Lösung zu erbringen, ist es besonders schwer ein mathematisches Modell zu entwerfen. Ein Fuzzy-System braucht nur eine Menge von Regeln, die eben gerade auf menschlichem Expertenwissen beruhen können. Aber dennoch ist es möglich ein jedes mathematisch fassbares, genauer gesagt durch eine stetige Funktion beschreibbares, Problem auch mit einer finiten Menge an Fuzzy-Regeln zu approximieren. Dies wurde von Bart Kosko in seinem *Fuzzy Approximation Theorem* (FAT) 1990 bewiesen. Am einfachsten ist es sich das FAT geometrisch zu veranschaulichen, da der formale Beweis sehr aufwendig ist. Möchte man eine 2 dimensionale Kurve im kartesischen Koordinatensystem approximieren, so stellt man sich eine Fuzzy-Regel mit einfacher Prämisse als Kreuzprodukt von dem Bereich den die Zugehörigkeitsfunktion der Prämisse abdeckt mit dem Bereich den die Zugehörigkeitsfunktion der Konklusion abdeckt vor, beides sind lineare Abschnitte auf Abszisse und Ordinate. Für das Kreuzprodukt ergibt sich dann ein Rechteck (beziehungsweise je nach Zugehörigkeitsfunktion, z.B. Dreiecksfunktionen eher eine Ellipse).

Mit geeigneten Regeln kann man diese Rechtecke dann über den Graphen legen und wenn man genügende Rechtecke nimmt und sie leicht überlappen lässt, dann kann die ganze Kurve damit abdecken werden. Die Genauigkeit nimmt natürlich zu wenn man kleinere Flächen nimmt, aber dies erhögt dann die Gesamtzahl der Regeln und das System wird komplexer.



[Abb. 5] FAM Fuzzymengen decken eine Kurve ab.

Dieses Theorem soll zeigen das man mit Fuzzy Systemen alles modellieren kann, wenn man nur geeignete Regeln hat. Existiert schon ein fundiertes Fachwissen, was bei vielen praktischen Phänomenen der Fall ist, so sind Fuzzy-Systeme eine gute Wahl, man braucht das Wissen nur noch in die Form linguistischer Kontrollregeln zu formulieren. Leider ist es aber als Konsequenz dessen nicht möglich ohne a-priori Wissen ein Fuzzy-System aufzubauen. Kann man kein Wissen akquirieren, so wird es auch keine Fuzzy-Regeln geben und so auch kein Fuzzy-System. Man kann also nur schon bekanntes Wissen modellieren und simulieren, nicht aber etwas neues Erlernen. Dieser Mangel an Lernfähigkeit und die daraus resultierende Unflexibilität im Sinne einer Nichtanpassungsfähigkeit sind ein grösses Manko dieses Ansatzes. Eine andere Gruppe von Modellierungansätzen geniessen allerdings den Rufe Lernfägikeit als eine ihrer haupten Eigenschaften zu haben, die künstlichen Neuronalen Netze. Diese Werden wir uns im folgenden Kapitel genauer anschauen.
Ein anderer Punkt der für eine Implementation als Fuzzy-System spricht ist das die Entscheidungen, die von einem System getroffen wurden, nachvollziehbar sind, denn sie beruhen ja auf den einzelnen Regeln der Regelbasis. Möchte man also wissen, warum um himmelswillen eine U-Bahn in mitten eines Tunnels einfach abbremst, so muss man sich anschauen welche Regel zu dieser Entscheidung gefürt hat, sozusagen auf welche Regel sich der Fuzzy-Zugführ gerade bezieht. Diese Transparenz kann auch im Falle medizinischer Diagnose- und Beratungssystemen von grösser Bedeutung sein.
Fuzzy-Systeme sind relativ leicht zu implementieren, zum einen was die Hardware angeht, denn betrachtet man sich noch einmal den Aufbau der Regeln, so erinnert man sich das die Einzelnen Mengen in der Prämisse UND-Verknüpft wurden. Das UND ist aber durch eine Vergleichtsoperation (min) definiert, welche als Hardware nicht allzu schwer zu realisieren ist. Auch andere Rechnungen die zur Evaluation der Regeln von Nöten sind, basieren auf modest schweren Rechenoperationen, wie zum Beispiel Schnitt oder Untermengen.
Was die Software angeht so gibt es bereits fertige Tools zur Modellierung, wie zum Beispiel fuzzyTECH der Inform GMBH. Das Programm erlaubt ein graphisches Designen der Fuzzy-Regen und generiert daraus c-Code.

Kapitel 2

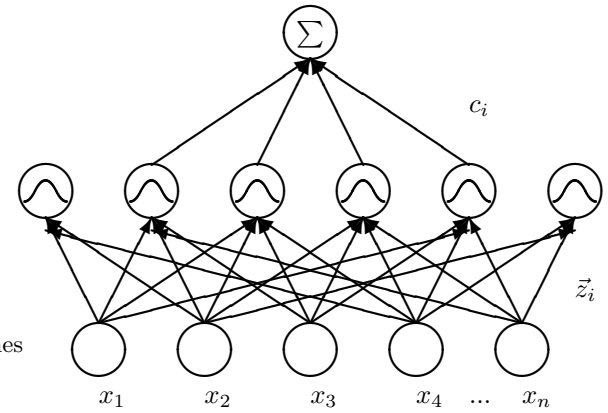
Neuronale Netze
Alles will gelernt sein

In diesem Kapitel soll nur ein spezieller Netzwerktyp beschrieben werden, welcher in Hinblick auf ein hybrides Modell von Bedeutung ist. Die darauffolegenden Beispiele sind dann wieder auf allgemeinere Netztypen bezogen und auch die sich anschließende Bewertung gilt allgemein Neuronalen Netzen (auch konnektionistische Systeme genannt).

2.1 Radiale-Basisfunktionen (RBF) Netze

2.1.1 Aufbau

RBF-Netze entsprechen vom Aufbau her einfachen feedforward[1] Netzen mit einer Schicht verdeckter Neuronen. Eingabe und verdeckte Schicht (hiddenlayer) sind maximal miteinander verbunden, sodass zu jedem Neuron der verdeckten Schicht der komplette Eingabevektor propagiert wird ([Abb. 6] aus [Zell94]). Das besondere an RBFNs sind ihre Aktivierungsfunktionen in der hiddenlayer. Es handelt sich nicht wie häufig um sigmoide oder binäre Schwellenwertfunktionen sondern es sind radialsymetrische Funktion wie z.b. Gaußfunktionen. Eine Aufgabe die ein solches Netz ausführen kann ist die Approximation mehrdimensionaler Funktionen, mit den radialen Funktionen der Neuronen in der mittleren Schicht als Stützstellen. Diese könne durch die Trainingsmuster vorgegeben sein, direkt berechnet werden oder durch spezielle Lernverfahren ermittelt werden.



[Abb. 6] Schema eines RBF Netzes

[1] alle Verbindungsvektoren sind nur in einer Richtung angeordnet. Es gibt keine rekurenten Verbindungen im Netz und auch keine Abkürzungen (shortcut-Verbindungen)

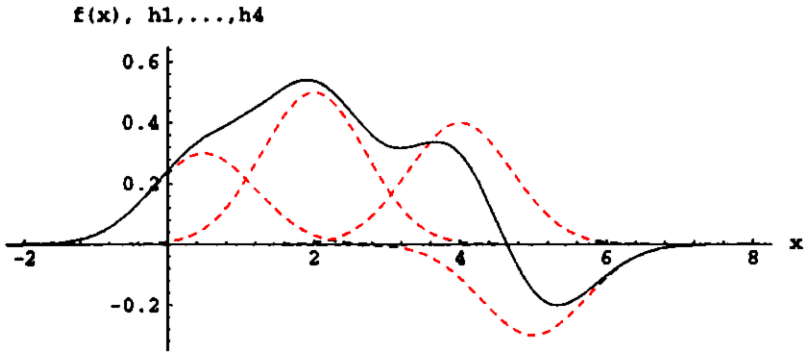
2.1.2 Mathematischer Hintergrund

Sei eine mehrdimensionale Funktion $f : \mathbb{R}^n \to \mathbb{R}$ durch ein RBFN zu approximieren. Bekannt sind n Stützvektoren der Funktion f, dann kann man die Funktion wie folgt approximieren:

$$f(\vec{x}) = \sum_{i=1}^{n} c_i \cdot \underbrace{e^{-\frac{||\vec{x}-\vec{z}_i||}{\sigma_i^2}}}_{\text{Gauß-Funk.}} \qquad \text{mit } \dim(\vec{x}) = n$$

Die e-Funktion ist eine einfache parametrisierte Gaußsche Glockenfunktion, mit $\vec{z}_i$ als Zentrum. Jedes dieser Zentren ist im RBFN als Gewichtsvektor einer der i verdeckten Neuronen repräsentiert. Der Faktor c_i jedes Summanden ist ein spezielles zu bestimmendes Gewicht, des Gewichtsvektor des Ausgabe Neurons, wessen Funktionalität im Aufsummieren der Stützfunktionen besteht.

Nach dem anlegen eines Vektor $\vec{x}$, wird er zuerst an alle Neuronen der zwischen Schicht propagiert, je kleiner der Abstand zum jeweiligen Zentrum $\vec{z}_i$, desto grösser die Ausgabe der Neuron dieser Schicht. Gleicht der Eingabevektor $\vec{x}$ einem der Gewichtsvektoren $\vec{z}_i$ so feuert nur dieses Neuron und bestimmt den Funtionswert $f(\vec{x})$. Wie eine solche Approximation aussehen könnte veranschaulicht die folgende Graphik aus [Zell94].



Das Bestimmen der Gewichte c_i geschieht durch das lösen eines linearen Gleichungssystems mit n Gleichungen und n Unbekannten. Es ist auch möglich ein RBFN mit mehreren Ausgabeneuronen zu bestücken, dann muss für jedes das LGS gelöst werden.

2.1.3 Nachtraining Radialer-Basisfunktion-Netze

Als Stützstellen verwendet man oft einfach die Vektoren aus der initialen Trainingsmenge, manchmal noch mit einem leiten "Rauschen" versehen um Ausreisser auszubügeln. Damit erzielt man aber nicht unbedingt ein optimales Ergebnis.
Ein Lernverfahren kann an mehreren Stellen eingreifen: die Stützstellen können verschoben werden, wodurch mach auch die Gewichte c_i neu berechnen muss; die Breite der Glockenkurven (Varianz) kann verändert werden und auch die Verbindungsgewichte können durch ein Gradientenabstiegsverfahren[2] modifiziert werden.
Zur Bestimmung der geeigneteren Stützstellen z_i verwendet man als Ähnlichkeitsmass zweier genormter Vektoren das Skalarprodukt (grösser je ähnlicher die Vektoren). Aus den Trainingsmustern werden willkürlich einige Vektoren als Zentrumsvektoren auserkoren und diese dann via Skalarprodukt mit allen Trainingsverktoren verglichen. Der Zentrumsvektor der dem aktuellen Lernvektor

[2]Das bekannteste Gradientenverfahren ist Backpropagation. Man tägt die Fehler eines Neuronalen Netzes als Funktion der Gewichte auf, und sucht dann nach einem globalen Minimum.

der Trainingsmenge am nächsten ist gewinnt und wird in seine Richtung verschoben. Nach mehrmaligem Wiederholen der Prozedur kommt man zu einer besseren Verteilung der Stützstellen.

2.2 Anwendungsbeispiele Neuronaler Netze

Die weiter unten dargestellten Anwendungsbeispiele sind relativ bekannt und wurden auch deshalb ausgewählt.

2.2.1 NETtalk

NETtalk wurde von Seijnowski und Rosenberg implementiert um eingegebene englische Texte vorzulesen. Es handelt sich um ein dreischichtiges Netzwerk mit 203 Eingabeeinheiten, von denen immer 29 Neuronen ein Zeichen kodieren, also 7 Zeichen pro Eingabe. Das Netz versucht herauszufinden wie man den mittleren Buchstaben der 7 ausspricht und hat dazu die 3 Vorgänger und 3 Nachfolger Buchstaben zur verfügung. Aus diesem Kontext leitet es die Aussprache ab. Die verdeckte Schicht besteht aus 80 Neuronen gefolgt von 29 Ausgabeeinheiten, die für artikulatorische Merkmale und die Betonung von Silbengrenzen stehen, aus denen sich das die Phoneme zusammen setzen. Verdeckte- und Ausgabeschicht sind maximal mit einander verbunden.
Das Netz bekam eine feste Lernaufgabe, die aus einer in Phoneme zerhackte Rede eines Kindes generiert wurde. Interessanterweise lernte das noch nicht vollständig trainierte Netz erst nur undeutlich vor sich hin zu brabbeln, so wie es kleine Kinder zu Beginn ihres Spracherwerbs tun. Erst gegen Ende konnte es dann in verständliches Englisch vorlesen. Das fertig trainierte Netz wies eine Aussprachegenauigkeit von 95% auf.

2.2.2 Finanzempfehlung

Das *NeuroForecasting Center* hat auf der Basis eines multilayer Perceptron ein Finanzenscheidungssystem entworfen und mit täglichen Wechselkurs-Daten aus dem Zeitraum 1984-1986 trainiert. Es soll Investitions- und Handelsstratiegien empfehlen.
Von 1986-1992 wurde jeden Tag nach den Vorschlägen des Netzes Investitionen getätigt. Man startete mit einem Finanzvolumen von 1 Mio. US-Dollar. Der jährliche Profit betrug 18%.

2.3 Bewertung von Neuronalen Netzen

Genau wie bei Fuzzy-Systemen ist es auch bei Neuronalen Netzen um ein Problem zu lösen nicht notwendig ein mathematisches Prozessmodell zu Grunde zu legen, es ist so gar möglich daß ein Netz eine Reaktion erlernt die in gar keinem logischen Zusammenhang mit seinem Input steht. Wie und was das Netz lernt hängt von den Trainingsdaten und dem jeweiligen Lernalgorithmus den man verwendet ab. So ist also ein Neuronales Netz im Gegensatz zu einem Fuzzy-System nicht auf linguistitsch formuliertes a-priori Wissen angewiesen. Dies ist ein wichtiger Vorteil gegenüber anderen Expertensystemen, die wegen nicht-Lernfähigkeit auch bei sich ändernder Umgebeung nicht mit einer Anpassung reagieren können, man müsste dazu die Architektur des Systems selbst verändern. Ein Neuronales Netz lernt unter Beibehaltung seiner Netztopologie (bis auf einige Ausnahmen z.B. Cascade-Correlation Lerning Architecture) in dem es nur die Gewichte der Verknüpungen zwischen den einzelnen Neuronen ändert. Dazu muss immer entweder eine *feste Lernaufgabe* zur Verfügung stehen, die ein offline-Lernen ermöglicht, oder der Regler muß mit Hilfe verstärkenden Lernens

die Regelaufgabe online durch Versuch und Irrtum erlernen dürfen (Was bei manchen Aufgaben von vornherein ausgeschlossen ist, man denke vielleicht an die Regulierung eines Atomkraftwerks). Man kann die bekannten Lernverfahren in 2 grosse Gruppen teilen. Das *überwachte Lernen* oder assoziatives Lernen (engl. supervised lerning) geschieht entweder unter Angabe der Güte der Reaktion des Neuronalen Netzes auf die ihm gestellt Aufgabe oder unter Angabe der gewünschten Reaktion selbst. Eine Menge aus Eingaben mit dazugehörigen, gewünschten Ausgaben nennt man Trainingsset. Die Abweichung des zunächst generierten vom gewünschten Ausgabevektors stellt das Fehlersignal dar, das es zu minimieren gilt. Traingssets dürfen nicht zu wenige Datenpunkte enthalten sonst kann das lernen erfolglos bleiben, dies ist unter umständen ein Nachteil. Gerade in der medizinischen Anwendung (Diagnostik) sind oft nur wenige Patienten-daten zu einem Krankheitsbild vorhanden, wenn es ein spezielleres Leiden ist. Oder auch bei industriellen Anwendungen kann erlangen eines Datensatzes sehr kostspielig sein.
Die zweite Gruppe von Lernverfahren ist das *unüberwachtes Lernen*. Es basiert auf Selbstorganisation (engl. self-organizing leraning), wobei dem Netzwerk lediglich der Eingabevektor päsentiert wird. Eine spezielle Art des unüberwachten Lernens stellt das Wettberwerbslernen dar, bei dem mehrere Ausgabeneuronen darum konkurrieren, den aktuellen Ausgabewert liefern zu dürfen.
Generell ist es nur für sehr einfach Netztypen möglich mathematisch zu beweisen das ihr Lernvorgang stabil konvergiert. Kompliziertere Netze können sozusagen unendlichlange lernen müssen.
Des weiteren sind Neuronale Netze in der Lage Gelerntes zu extrapolieren. Also auch auf Eingaben, die nicht trainiert wurden, mit einer sinnvollen Ausgabe zu reagieren (allerdings nur bis zu einem gewissen Grad). Gibt man einem Neuronalen Netz ein Muster, z.B.: [1,3,5,7,9,11,...] und trainiert es so das es bei erkennen dieser Zahlen eine 1 als Output gibt, sonst eine 0, so wird es auch in der Lage sein weitere ungerade Zahlen zu erkennen und dann ebenfalls so etwas wie 0.94 als Output ausgeben.
Eng damit verbunden ist auch die Fehlertolleranz konnektionistischer Systeme für ungenaue Eingaben.
Neuronale Netze arbeiten massiv parallel und sind dadurch robust gegenüber dem ausfallen einzelner Neuronen, eine Qualität die sie mit ihren biologischen Vorbildern gemein haben.
Ein grosser Nachteil ist die mangelnde Interpretierbarkeit der von einem Netz getroffenen Entscheidungen. Man kann sagen das sich Neuronale Netze wie eine Blackbox verhalten. Nach dem lernen kann man zu einem gewissen Input mit einem Output rechnen, aber es ist sehr schwierig nachzuvollziehen wie es zu dieser Reaktion kommt. Wenn man sich die Gewichtsmatrix der Verknüpungen ansieht, in der ja das gelernte Wissen des Netzes steckt, so wird man darin kaum eine logische Struktur oder strukturierte Information finden.
Und auch das Umgekehrte ist nicht möglich. Wenn man bereits weiss wie man von einer Eingabe zu einer Ausgabe kommt, so kann man dieses a-priori Wissen nicht einfach in das Netz stecken, man müsste auch das irgendwie in der Gewichtsmatrix kodieren, diese werden aber normalerweise randomisert initialisiert. Ein Neuronales Netz muss alles "from scratch" lernen.

An diesem Punkt fällt auf das einige der Nachteile der Neuronalen Netze und der Fuzzy-Systeme komplementär sind. Das fürte zu der Idee diese Systeme zu kombinieren, auf das sie sich gegenseitig ausgleichen. Dieser Ansatz wird in dem nachfolgende Kapitel beschrieben.

Kapitel 3

Neuro-Fuzzy Hybride, Das Ganze ist mehr als zwei Hälften

Die beiden vorhergehenden Kapitel schlossen mit einer kleinen Zusammenfassung der Vor- und Nachteile des jeweiligen Ansatzes. Dieses Kapitel behandelt einen Ansatz, der beide Modelle, Fuzzy-Systeme und Neuronale Netz, kombiniert und von dem man sich verschiedene Synergien erhofft. Daher seien an dieser Stelle noch einmal die Eigenschaften der beiden einzelnen Ansätze tabellarisch zusammengefasst.

Neuronale Netze	Fuzzy-Systeme
Vorteile	
• keine mathematisches Prozeßmodell notwendig • Kein Regelwissen Notwendig • Verschiedene Lernalgorithmen	• keine mathematisches Prozeßmodell notwendig • A-priori (Regel-) Wissen nutzbar • Einfache Interpretation und Implementierung
Nachteile	
• Black-Box-Verhalten • Kein Regelwissen extrahierbar • Heuristische Wahl der Netzparameter • Anpassung an veränderte Parameter ist evtl. schwierig und kann Wiederholung des Lernvorgangs erfodern • Kein a-priori Wissen verwendbar "learning from scrach" • Der Lernvorgang konvergiert nicht garantiert	• Regelwissen muß verfügbar sein • Nicht lernfähig • Keine formalen Methoden für Tuning • Semantische Probleme bei der Interpretation "getunter Regeln" • Anpassung an veränderte Parameter schwierig • Ein "Tuning"-Versuch kann erfolglos bleiben

Tabelle: Gegenüberstellung Neuronaler Regelung und Fuzzy-Reglung aus [Nauck96]

3.1 Verschiedene Klassen von Hybridsystemen

Es gibt eine viel Zahl verschiedener Ansätze wie man ein Neuro-Fuzzy Hybridsystem konstruiern kann. Einige davon unterscheiden sich grundlegend in ihrer Architektur. Diese Liste teilt sie in Kategorien ähnliche [YaTeo96] ein:

(**A**) kooperative Fuzzy-Systeme: Fuzzy-Systeme, die durch Neuronale Netze trainiert werden. Ein nachgeschaltetes Neuronales Netz kann z.B. lernen wie weit man einer Regel bei bestimmten Situationen vertrauen kann.

(**B**) normale neuronale Netze, die "Fuzzy-Neuronen" (z.B. min-/max-Neuronen) und "Fuzzy-Gewichte" benutzen. Die Struktur des Fuzzy-Systems lässt sich in der Netztopologie wieder erkennen.

(**C**) Neuronale Netze die mit fuzzy Lernmethoden trainiert werden. Es werden die Veränderungen der Gewichte zwischen den Neuronen bei jedem Lernschritt durch ein Fuzzy-System berechnet.

(**D**) Neuronales Netz und Fuzzy-System arbeiten unabhänig.

(**E**) topologische Konfiguration eines Neuronalen Netzen, mit mehr oder weniger komplexen Fuzzy-Systemen als Neuronen.

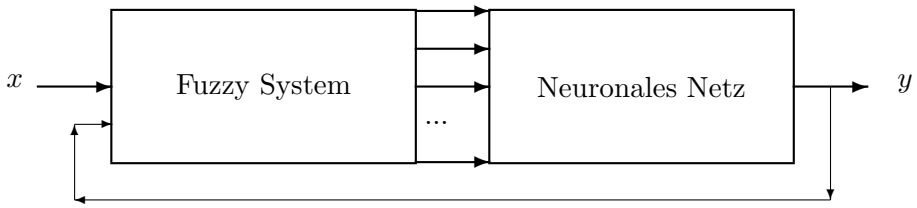
(**F**) Ein Mix aus klassischen Experten-Systemen und einer der obigen Ansätze.

Schauen wir uns nun die Ansätze A und B. E ist nur eine Weiterfürhurng des Ansatzes A und die anderen Klassen sind nicht so weit verbreitet.

3.1.1 Kooperative Systeme (A)

Arbeitet Ein Fuzzyregler in einer dynamischen Umgebung z.B. bei sich verändernden Temperaturen, so kann man mit einem adaptiven System sich an diese Situation immer neu anpassen.
Das Neuronale Netz und das Fuzzy-System existieren parallel neben einander. Dabei sind mehrere Kombinationsmöglichkeiten denkbar:

1) Einem Fuzzy System wird ein Neuronales Netz nachgeschaltet, was die Ausgabe des Gesamtsystems liefert (siehe [Abb. 7]).

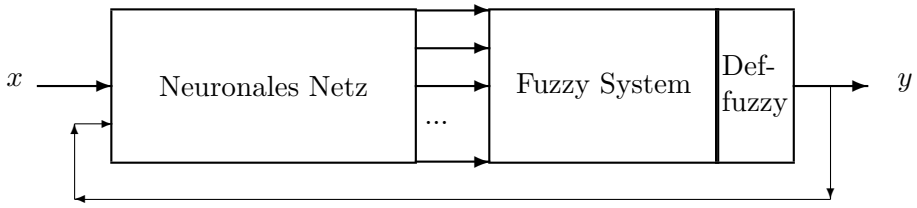


[Abb. 7] Kooperative Systeme Variante a)

Die Ausgaben des Fuzzy Systems werden sofort durch das Neuronale Netz nach bearbeitet. So kann man auf einem vorhandenen Basiswissen aufbauend (steckt im Fuzzy-System) ein nicht lineares System konstruieren, das sich dennoch an eine speziellere Gegebenheiten anpassen kann, die nicht 100% durch das Basiswissen abgedeckt werden. Dieses "Fein-Tunen" der Ausgabe übernimmt das Neuronale Netz. Welches Tuning bei welcher Eingabe erforderlich ist kann das Netz lernen. Das Fuzzy System muss keine defuzzyfizierte Ausgabe liefern, auch das kann das Neuronale Netz erledingen.
Die rekurente Verbindung in [Abb. 7] ist eine Erweiterung für den Fall das vorherige Ausgaben des Netzes y_{t-1} in die Berechnung der Ausgabe y_t mit einfliessen sollen.

Denkbar ist auch das das Neuronale Netz Veränderungen an der Regelbasis des Fuzzy-System vornimmt, dabei entweder die Regeln selbst verändert oder nur die Zugehörigkeitsfunktionen manipuliert, und so eine Anpassung erreicht, denn das Netz kann ja für jede Sitation die beste Modifikation des Fuzzy-System erlernen.

2) Eine andere Variante ist es das Neuronale Netz vor das Fuzzy System zu schalten.



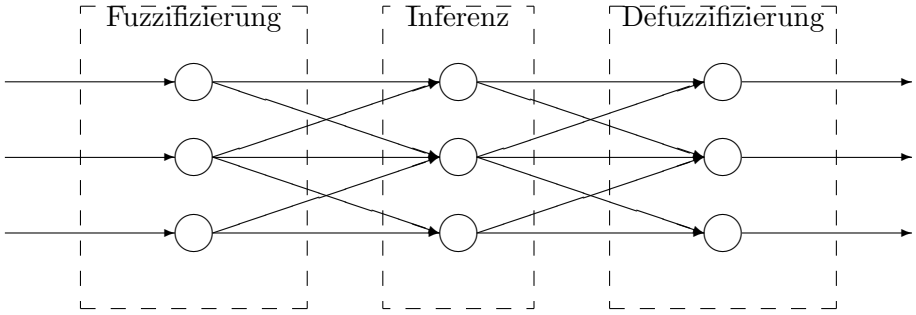
[Abb. 8] Kooperative Systeme Variante b)

So können Veränderungen in den Eingabedaten, die nicht mehr durch das Fuzzy-System prozzesiert werden können vorher vom Neuronelen Netz vorbearbeitet werden.

Es sind noch viele andere kooperative Modelle, die verschiedene Anordungen von Neuronalen Netzen und Fuzzy-Systemen beinhalten, denkbar.
Kommen wir aber nun zu einem inhaltlich anderen Ansatz.

3.1.2 Hybride Neuro-Fuzzy-Systeme

Damit sind hier Neuronale Netze gemeint die durch ihren Aufbau die Struktur und Funktionalität eines Fuzzy-Systems nachahmen. Der komplette Fuzzy-System wird so zusagen in neuronaler Architektur implementiert, das erlaubt es aber das ganze System noch auf neuronaler Hardware laufen zu lassen.
Was muss aber nun mit dem Neuronalen Netz geschehen damit es wie ein Fuzzy-System arbeitet? Zum Beispiel müssen die sigmoiden oder binären Aktivierungs-Funktion eines Neuron durch eine parametrierbare Funktion ersetzt werden, sodass eine Simulation der einer Zugehörigkeitsfunktion im Sinne der Fuzzy-Logik möglich ist. Solche Fuzzy Neuronen können beliebige t-Normen als Aktivierungsfunktionen tragen, um Informationen "fuzzy" zu integrieren. Beispiele wären die min- und max-Funktionen, die man auch vom *winner-take-all Prinzip*[1] in anderen Neuronalen Netzen kennt. Modellhaft kann man sich die dreiteilige Struktur wie in [Abb. 9] gezeigt auf das Netz projiziert vorstellen.



[Abb.9] Fuzzy-System in neuronaler Architektur

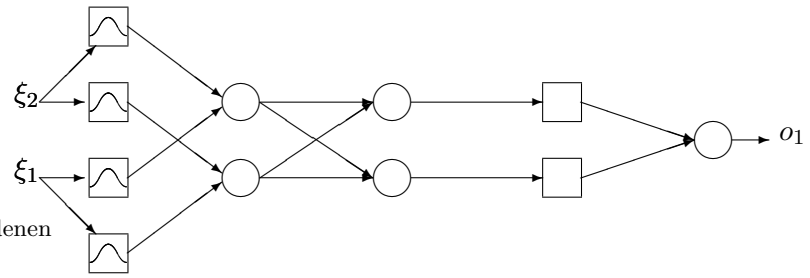
[1] Auf eine bestimmte Eingabe wird immer nur ein Neuron aktiv. Beispiele dafür sind Kohonen-Schichten.

Es ist wichtig darauf zu achten das nicht die Interpretierbarkeit verlorenen geht. Ein Rückübersetzung in Fuzzy Regeln kann durch einhalten der Fuzzy-Schritte gewhrleistet werden. Genauer:
Die einzelnen Neurone der Fuzzyfizierungsschicht represäntieren eine Fuzzy-Variable (Term) und haben die Aufgabe den Eingangswerten mit Hilfe einer Schwellenwertfunktion in einen Zugehörigkeitsgrad für einen linguistischen Term umzuwandeln.
Die Zwischenschicht ist für die Interferenz zuständig. Jedes Neuron stellt eine Regel der Regelbasis dar. Die Verbindungen zur Fuzzyfizierungsschicht stellen die Prämissen der Regel da, und die Verbindungen zur Defuzzifizierungsschicht sind die Schlussfolgerungen der einzelnen Regeln.
Die letzte Schicht übernimmt dann die Defuzifizierungsaufgabe, indem sie aus den Erfülltheitsgraden der Regeln einen korrekten Ausgangswert berechnet.
Bei komplizierteren Inferenzmethoden oder Defuzzifizierungs-Vorgängen treten an die Stelle der einfachen Schichten unter Umständen kompliziertere Subnetze.

Diese grobe Beschreibung eines Hybrid-Systems sollen anhand der nun präsentierten Beispiele *ANFIS* und *Fuzzy RuleNet* illustriert werden.

3.1.3 Beispiel: ANFIS

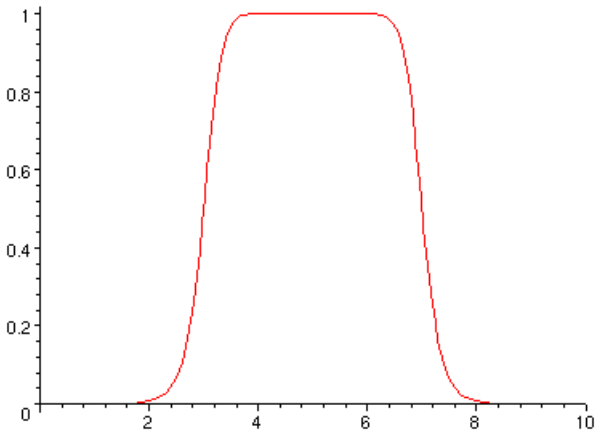
Das Akronym steht für Adaptive-Network-based Fuzzy Inferenz System. Es wurde von Jang in seiner Dissertation ausfürlich beschrieben [Jang92]. ANFIS ist ein Netz mit sechs feedforward Schichten, wobei keine Verbindung besonders gewichtet wird - die Logik steckt einzig und alleine in den Neuronen und der Verknüpungstopologie. Das Neuronale Netz kann die Funktionalität eines Surgeno-Reglers bereitstellen.



[Abb. 10] Aufbau der verschiedenen Schichten von ANFIS

Die Knoten der ersten Schicht Neuronen (in [Abb. 10] Vierecke) ordnen den Eingabewerten ξ eine Zugehörigkeit zu einem linguistischen Ausdruck zu. Als Aktivierungsfunktionen eignen sich beispielsweise

$$\mu_A(\xi) = \frac{1}{1 + \left(\left(\frac{\xi - c}{a}\right)^2\right)^b}$$



[Abb. 11] Graph für *a=2, b=6, c=5*

Mit drei Parametern *a, b, c.* Dabei ist a die Breite der Zugehörigkeitsfunktion über den Eingabewerten, sozusagen wie viele Elemente noch mit $\mu > 0$ zur Menge gehören; c verschiebt μ horizontal auf dem Eingabeintervall und b ist die Steigung der Funktion, also zu welchem Grad Elemente am "Rand" der Menge dazu gehören. Jeder Knoten bekommt nur Informationen von einer Eingabeeinheit und steht für genau eine linguistische Eigenschaft (A_i). Die Parameter *a, b* und *c* können von einem Lernenverfahren bestimmt werden.
In der zweiten Schicht gibt es für jede Fuzzy-Regel genau ein Neuron, welche die Regelprämisse bilden, in dem es die einmündenden Zugehörigkeitswerte auf multipliziert (auch ein fuzzy AND bzw. Schnitt sind denkbar).
Die mittlere Neuronenschicht hat die gleiche Dimension wie ihre, mit ihr maximal verbundene, Vorgängerschicht. Sie kalkuliert den Erfülltheitsgrad $\overline{w}_1$ der Prämise in Abhängingkeit aller anderen Regeln.

$$\overline{w}_1 = \frac{w_1}{w_1 + w_2}$$

Die vierte Schicht (hier viereckige Neuronen) hat zusätlich zu einer Verbindung zu genau einem Neuron der dritten Schicht, noch Verbindungen zu allen Eingabe Neuronen (in [Abb. 10] nicht eingezeichnet). Hier wird die Konklusion der Regel berechnet

$$o_1 = \overline{w} \cdot (\alpha_1 \xi_1 + \alpha_2 \xi_2 + r_1)$$

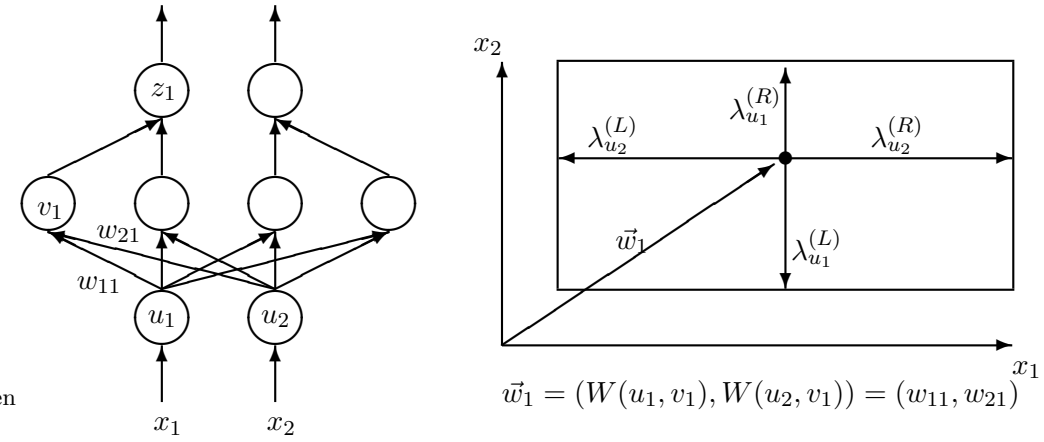
Die Parameter α_i und r sind auch vom Lernverfahren zu bestimmen.

Aus einem funktionierenden ANFIS kann man Regeln des Sugeno-Typs extrahieren. Ihre allgemeine Form wäre aus unserem Beispiel mit zwei Eingaben:

IF ξ_1 is A_1 and ξ_2 is A_2, THEN $f_1 = \alpha_{11}\xi_1 + \alpha_{12}\xi_2 + r_1$.
IF ξ_1 is A_1 and ξ_2 is A_2, THEN $f_1 = \alpha_{21}\xi_1 + \alpha_{22}\xi_2 + r_2$.

Wenn die Zugehörigkeitsfunktionen bekannt sind dann kann man die Parameter (α's) der 4. Schicht durch Lösung eines Gleichungssystems errechnen. Diese sehr effiziente Bestimmung beruht dann nicht auf einem Lernverfahren.
Möchte man auch die Parameter für die Zugehörigkeitsfunktionen aus der ersten Schicht erlernen, wendet man ein zweistufiges Gradientenabstiegsverfahren an, bei dem man im ersten Schritt die Parameter der 4. Schicht für fest gewählte Zugehörigkeitsfunktionen wieder durch das Gleichungssystem bestimmt und dann im zweiten Schritt versucht für die Eingabemusster den Ausgabefehler zu minimieren.

3.1.4 Beispiel: Fuzzy RuleNet



[Abb. 12] Bsp. mit 2 Dimensionen

Fuzzy RuleNet basiert auf der uns schon bekannten Struktur der RBF Netze. Die radialsymetrischen Aktivierungsfunktionen in der verborgenen Schicht der RBFNs legen eine Interpretation als Zugehörigkeitsfunktionen nahe. Bei diesem Netztyp ist es möglich aus einem Trainierten Fussy RuleNet die erlernten Fuzzy Regeln wieder zu extrahieren, indem man die mehrdimensionalen Radialen Basisfunktionen wieder auf einzelne Dimensionen projiziert und diesen dann die linguistischen Terme zugeordnet.
In [Abb. 11] ist ein zweidimensionales Beispiel zu sehen. Der Eingabevektor $\vec{x} = (x_1, x_2)$ wird an alle Neurone der verdeckten Schicht weiter gereicht. Diesen Neuronen sind jeweils 2 Abstandsbegrenzungsvektoren $\vec{\lambda}^{(L)}$ und $\vec{\lambda}^{(R)}$, der gleichen Dimension wie die Eingabevectoren, zugeordnet. $\vec{w}$ ist wie auch bei den RBF Netzen der Vektor gleicher Dimension des Eingabevektors, der die räumlichen Eigenschaften des verdeckten Neuron beschreibt. Die Netzeingabefunktion dieser Neuronen berechnet sich durch den Abstand von $\vec{u}$ und $\vec{w_1}$, allerdings verwendet man hier nicht die Euklidische Norm sondern die ∞-Vektorneorm (Maixmum der Abstände aller Dimensionen). Also wird der Abstand von der Dimension bestimmt die $\vec{w}$ am fernsten ist.

$$\text{net}_{v_1} = ||\vec{u} - \vec{w}_1||_\infty = \max(|u_1 - w_{11}|, |u_2 - w_{21}|)$$

Dies ist auch die Ausgabe von v allerdings mit evtl. modifiziertem Vorzeichen und zwar wird es negativ wenn eine Dimension von $\vec{u}$ außerhalb dem im rechten Teil von [Abb. 11] gekennzeichneten Gebiet liegt. Es wird für jedes Neuron der verdeckten Schicht von seinem Gewichtsvektor $\vec{w}$ und den beiden Begrenzungsvektoren λ bestimmt.
Die Neurone der Ausgabeschicht kann man nach dem winner-take-all Prinzip feuern lassen dann ergaben sich diese Regeln:

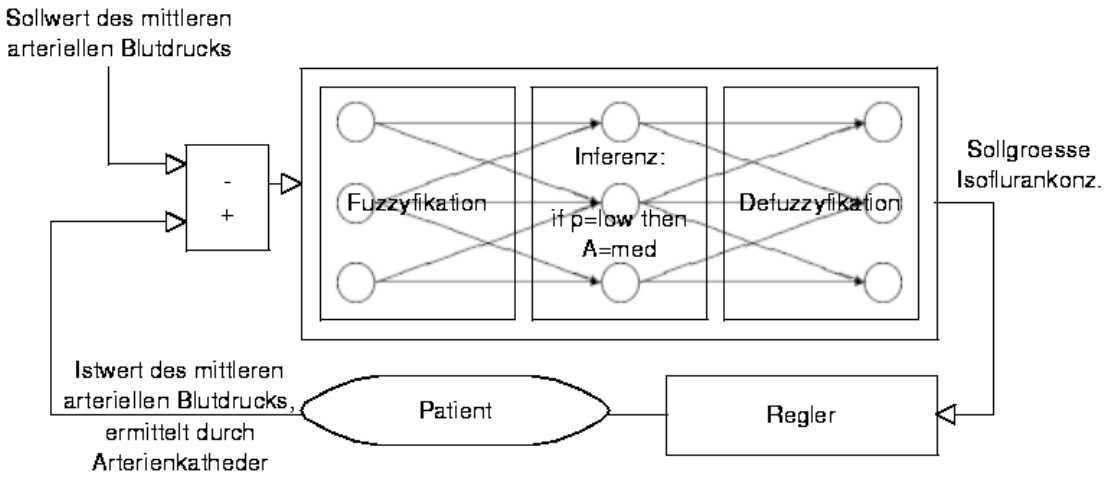
if $u_1 \in [w_{11} - \lambda_{u_1}^{(L)}, w_{11} + \lambda_{u_1}^{(R)}]$ **and** $u_2 \in [w_{21} - \lambda_{u_2}^{(L)}, w_{21} + \lambda_{u_2}^{(R)}]$ **then** $\vec{x}$ gehört zu Klasse Z

Die Regel besagen also wenn ein Eingabeverktor in eine von einem Neuron in der Zwischenschicht dargestellt Region fällt, dann aktiviere ein spezielles Neuron. Diese Funktionalität enspricht einem Clustering Verfahren, welches man zur Datenanalyse verwenden kann. Der Lernalgorithmus muss die Partitionierung des Eingaberaumes in die verschiedenen Cluster vornehmen. Im Schlimmstenfall kann sich ein Cluster pro Eingabevektor ergeben. Das Verfahren ist allerdings etwas komplizierter und daher soll hier nicht weiter darauf eingegangen werden. Siehe dazu [Nauck96].
Möchte man Fuzzy RuleNet zum Funktionen zu approximieren einsetzen, so werden die Aktivierungsmuster der Ausgabeeinheiten durch Summation zusammen gefasst, sodass man beispielsweise

einen Stellwert bekommt. In diesem Falle liesse sich das Netz als eine Sugeno-Regler interpretieren, denn die repräsentierten Fuzzy Regeln haben eine reellwertige Konklusion.

3.2 Anwendungsbeispiel

Neben den vielen weiteren Modellen für Neuro-Fuzzy-Hybridsysteme, von denen hier nur 2 grob beschrieben wurden gibt es aber auch einige Beispiele aus der Anwendung.
Ein recht interessantes stammt aus der Biomedizintechnik. Es wurde ein System zur Narkosetiefenreglung geschaffen. Ein medizinische Grösse die ein Arzt zu Rate zieht wenn er die Tiefe eines Narkoseschalfes überprüfen möchte ist der arterielle Blutdruck. Steigt dieser an so kann der Patient bald aufwachen. Ein zu starkes abfallen des Blutdrucks ist aber auch nicht wünschenswert. Es gibt also für einen Patienten einen geeigneten Sollwert des Blutdrucks, bei dem er nicht gleich aufwachen droht. Das Programm bildet die Differenz zwischen den aktuellen Blutdruck und diesem Sollwert und leitet nach dem diese Abweichung durch ein Neuro-Fuzzy-System prozessiert wurde eine Änderung in der verabreichten Isoflurankonzentration ein. Isofluran ist ein gasförmiges Anästetikum, was über ein Maske zugegeben wird.
Das System gleicht vom Aufbau einem Fuzzy-Regler ist aber in neuronaler Architektur gehalten, entspricht also dem Modell (B).



[Abb. 13] Narkosereglung

3.3 Resume

Vielleicht ist es am Ende doch möglich eine deskriptive Definition von Neuronalen-Fuzzy-Hybrid Sytemen zu geben:

Bei einem Neuro-Fuzzy-Hybridsystem handelt es sich um ein *lernfähiges*, *interpretierbares* Modell mit dem Expertensysteme oder Controller realisiert werden können, wo bei die Qualitäten der einzelnen Systeme weitgehend vereint werden.

Wenn wir uns die Tabelle 1 auf Seite 14 in Erinnerung rufen dann können wir feststellen das einige der Erwünschten Sysergieeffekte durch Hybridmodelle erreicht werden können. Je nachdem

ob man mit vorgegebenen Regeln oder ohne arbeiten möchte läßt sich bei ANFIS ein anderes Lernverfahren auswählen. Bei gelernten Parametern bleibt dieses gewonnene Wissen aber weiterhin interpretierbar, in dem man die Regeln extrahiert, selbiges gilt auch für Fuzzy RuleNet. Das Black-Box-Verhalten wurde also beseitigt. Alleine das Faktum das für Hybridmodelle Lernverfahren existieren und somit ein Anpassen und Nachbessern der Regel möglich ist schon ein Fortschritt zu den starren Fuzzy-Systemen.

Alles in allem handelt es sich bei Neuro-Fuzzy-Hybridsystemen um einen gewinnbringenden Ansatz, der die auch schon in der Anwendung unter Beweis gestellt hat. Welche Einsatzbereiche sich in der Zukunft noch auf tun werden kann man mit Interesse entgegensehen.

Literaturverzeichnis

[Aristotels OrganonIV] Lehre vom Beweis oder Zweite Analytik (Organon IV)
Aristoteles

[Demant93] Fuzzy-Theorie oder Die Faszination des Vagen,
Bernd Demant vieweg

[Jang92] Neuro-Fuzzy Modelning: Architectures, Analyses and Applications
Jyh-Shing Roger Jang

[Kosko92] Neural networks and fuzzy systems: a dynamical systems approach to machine intelligence
Kosko, Bart (1992)
Prentice Hall

[Kosko93] Fuzzy Thinking - The new science of fuzzy logic
Kosko, Bart
Hyperion

[Nauck96] Neuronale Netze und Fuzzy-Systeme
Grundlagen des Konnektionismus, Neronaler Fuzzy-Systeme und der Kopplung mit wissensbasierten Methoden
Detlef Nauck, Frank Klawonn, Rudolf Kruse wieweg

[Scharl99] Arno Scharl, Neurofuzzy-Hybridsysteme
Theoretische Grundlagen,Vergleich mit statistischen Problemlösungsklassen und Anwendug im Rahmen leistungsdiagnostischer Fragenstellungen
WUV-Universitßtverlag Wien

[Seraphin94] Neuronale Netze und Fuzzy-Logik
Marco Seraphin
Fransis'

[YaTeo96] Fuzzy Logic Impelementation and Application
Neuro-fuzzy Systems: Hybrid Configuration
h.-N.L. Teodorescu, T. Yamakawa

[Zell94] Simulation neuronaler Netze, A. Zell,
Oldenbourg

Ich bedanke mich bei meiner Betreuerin Elena Sapozhnikova für das Korrigieren der Arbeit. Es war vor allem aufgrund meines eher "unscharfen" Verständnis der deutschen Orthographie sicher ein schwieriges Unterfang.