1 Einleitung

Bei Neuro-Fuzzy-Systemen handelt es sich um die Kombination eines Neuronalen Netzes mit einem Fuzzy-System. Da der Umfang dieser Arbeit beschränkt ist, setzen die Autoren bestimmte Vorkenntnisse aus den Bereichen Neuronale Netze und Fuzzy-Logik voraus. Dazu zählen der grundlegende Aufbau eines mehrschichtigen Perzeptrons und der Ablauf des Backpropagation-Algorithmus aus dem Bereich der Neuronalen Netze. Die Kenntnis der Begriffe T-Norm und T-Conorm, Zugehörigkeitsfunktion, linguistische Variable sowie Mamdani- und Sugeno-Controller bildet weitere Voraussetzung. Darin eingeschlossen sind Kenntnisse über Fuzzy-Mengen und -Regeln sowie Fuzzy-Logik im Allgemeinen.
Basierend auf diesem Grundwissen betrachten wir bestimmte Eigenschaften von Fuzzy-Systemen und Neuronalen Netzen.
Im Allgemeinen orientiert sich der Entwurf von Fuzzy-Controllern an Expertenwissen, das der Fuzzy-Controller nachahmt. Dies setzt voraus, dass für das zu regelnde System die Reaktionen und das Wissen des Experten bekannt sind. Ohne dieses Wissen lassen sich weder Fuzzy-Mengen noch Fuzzy-Regeln entwerfen. Ebenso besteht das Problem der Vollständigkeit. Da ein Fuzzy-Controller nicht zur Laufzeit lernen kann, muss die Lösung des Problems beim Entwurf vollständig und richtig über alle Eingabedaten sein. Auch kann sich der Fuzzy-Controller nicht an sich ändernde Parameter z. B. durch mechanischen Verschleiß anpassen.
Im Gegensatz dazu stehen Neuronale Netze, deren Hauptmerkmal ihre Lernfähigkeit darstellt. Als Nachteil steht dem gegenüber, dass evtl. vorhandenes Wissen über das Problem nicht in einem Neuronalen Netz implementiert werden kann. Gleichzeitig besteht darin der Vorteil, dass eine Problemlösung durch rein daten-getriebenes Lernen erfolgt. Der Lernerfolg eines Neuronalen Netzes stellt einen Unsicherheitsfaktor dar, da nicht vorher bestimmt werden kann, ob und wann sich ein Trainingserfolg einstellt. Unter ungünstigen Voraussetzung kann eine Überanpassung an die Trainings- oder Validierungsdaten eintreten, was die Lösung für reale Daten negativ beeinflussen kann. Da sich Neuronale Netze wie eine *Blackbox* nach außen darstellen, lässt sich ihr innerer Zustand nicht oder nur sehr eingeschränkt interpretieren.
Die Motivation zur Kombination Neuronaler Netze und Fuzzy-Systeme zu Neuro-Fuzzy-Systemen liegt in der Idee des *best of both worlds*. Das Ziel besteht hierbei darin, mit einer Stärke des einen Ansatzes in einem bestimmten Bereich eine Schwäche des anderen in diesem Bereich zu kompensieren, um so zu einem leistungsfähigeren Gesamtsystem zu kommen.

2 Grundlagen

Bei der Betrachtung von Neuro-Fuzzy-Systemen liegt das Hauptunterscheidungsmerkmal im Grad der Kopplung von Neuronalem Netz und Fuzzy-System. Kooperative Neuro-Fuzzy-Systemezeichnen sich durch eine lose Kopplung aus, wobei hybride Neuro-Fuzzy-Systemeeine enge Kopplung aufweisen.

2.1 Kooperative Neuro-Fuzzy-Systeme

Bei kooperativen Neuro-Fuzzy-Systemen bleibt die Eigenständigkeit des beteiligten Neuronalen Netzes und Fuzzy-Systems erhalten und es besteht eine klar definierte Trennung. Die einfachste Kombination eines Neuronalen Netzes und eines Fuzzy-Systems besteht in dem Ansatz der Aufbereitung der Eingabedaten des Fuzzy-Controllers durch das Neuronale Netz. Ebenso lassen sich die Ausgaben eines Fuzzy-Controllers durch ein Neuronales Netz aufbereiten. Hierbei beeinflussen sich die beiden Systeme gegenseitig nicht sondern nur die Daten.

Wenn sich die Fuzzy-Mengen der linguistischen Variablen eines Fuzzy-Controllers zur Regelung eines Systems nicht bestimmen lassen, aber gleichzeitig Datenpaare aus Eingabe und gewünschter Ausgabe vorliegen, lassen sich diese Fuzzy-Mengen mit einem Neuronalen Netz erzeugen. Dies erfolgt offline durch Bestimmung geeigneter Parameter der Zugehörigkeitsfunktionen oder Approximation der Zugehörigkeitsfunktionen durch das Neuronale Netz. Die Abbildung 1 zeigt den schematischen Aufbau einer solchen Lösung. Das Training und die Erzeugung der Fuzzy-Mengen erfolgt offline, so dass das Neuronale Netz zur Laufzeit nicht benötigt wird.

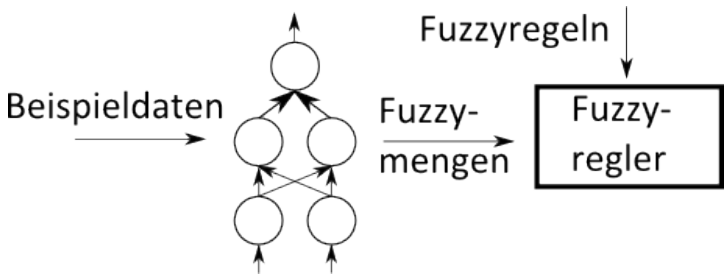

Abbildung 1: Neuronales Netz zum Offline-Lernen der Fuzzy-Mengen eines Fuzzy-Controllers

Mit einem ähnlichen Ansatz lassen sich auch Fuzzy-Regeln erzeugen, wie Abbildung 2 zeigt. [BKKN03] nennt hierzu Clustering-Verfahren oder selbstorganisierende Karten als Neuronale Methode zur Bestimmung der Fuzzy-Regeln. Auch hier arbeitet das Neuronale Netz offline.

Die Anpassung der Fuzzy-Mengen eines Fuzzy-Controllers durch ein Neuronales Netz kann auch online erfolgen, wie in Abbildung 3 gezeigt. Dazu benötigt das Neuronale Netz die Ausgaben des Fuzzy-Controllers als Eingabe zur Fehlerbestimmung.

Mit einer solchen Fehlerbestimmung lässt sich auch ein System wie in Abblidung 4 gezeigt entwerfen, das die Gewichte der Fuzzy-Regeln des Fuzzy-Controllers online verändert.

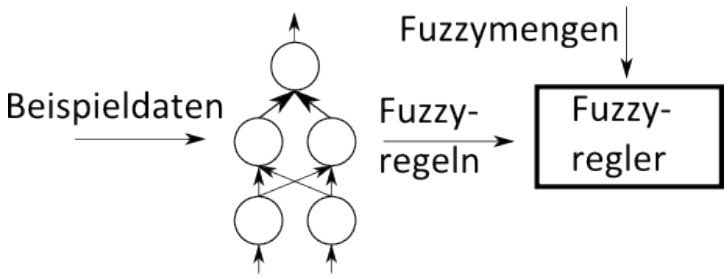

Abbildung 2: Neuronales Netz zum Offline-Lernen der Fuzzy-Regeln eines Fuzzy-Controllers

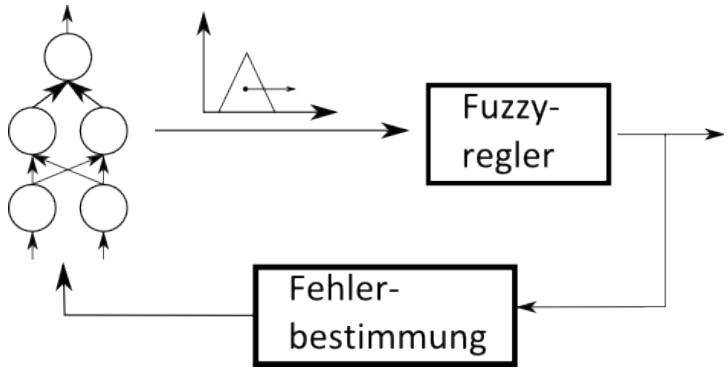

Abbildung 3: Neuronales Netz zur Online-Anpassung der Fuzzy-Mengen eines Fuzzy-Controllers

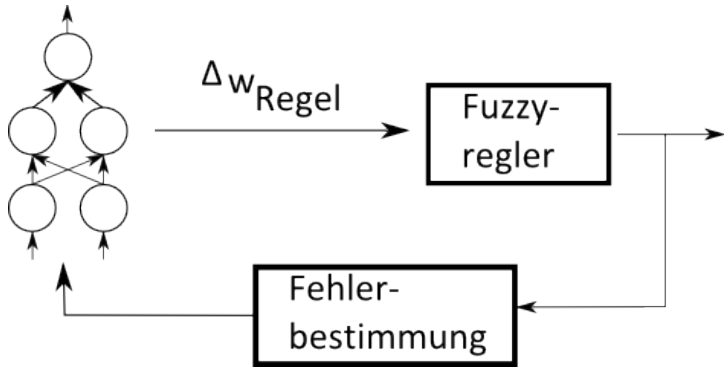

Abbildung 4: Neuronales Netz zur Online-Anpassung der Fuzzy-Regeln eines Fuzzy-Controllers

2.2 Hybride Neuro-Fuzzy-Systeme

Bei hybriden Neuro-Fuzzy-Systemen verschmelzen Neuronales Netz und Fuzzy-Controller zu einem System bei dem keine klare Trennung zwischen Neuronalem und Fuzzy-Teil

mehr möglich ist. Abbildung 5 zeigt das Schema eines solchen hybriden Neuro-Fuzzy-Controllers nach [BKKN03].

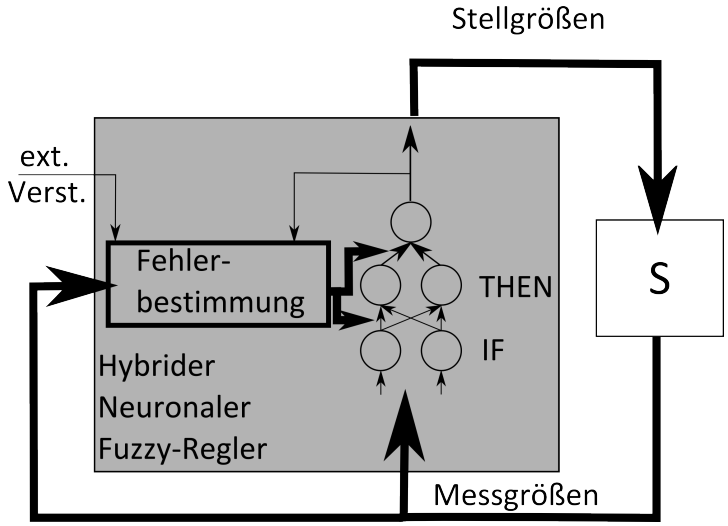

Abbildung 5: Schema eines hybriden Neuro-Fuzzy-Controllers

3 Das generische Fuzzy-Perzeptron

Als ein Beispiel hybrider Neuro-Fuzzy-Systemebetrachten wir in diesem Abschnitt das generische Fuzzy-Perzeptronnach [BKKN03] genauer. Abbildung 6 zeigt ein generisches Fuzzy-Perzeptron zur Lösung des XOR-Problems. Dabei handelt es sich um ein dreischichtiges vorwärtsgetriebenes Neuronales Netz bestehend aus einer Eingabeschicht, einer versteckten Schicht und einer Ausgabeschicht. Die Gewichte der Verbindungen zwischen den Neuronen werden durch den Zugehörigkeitsgrad der Ausgabe der vorherigen Schicht zu den beiden abgebildeten Fuzzy-Mengen s für small und b für big repräsentiert. Die Ausgabefunktion der Neuronen der versteckten Schicht bildet eine T-Norm, die Ausgabefunktion der Ausgabeschicht bildet die entsprechende T-Conorm, hier min und max.

Wertetabelle Die Tabelle 1 zeigt die Ausgaben der jeweiligen Schichten für die möglichen Eingaben des XOR-Problems. An den Ausgabewerten der Ausgabeschicht lässt sich die korrekte Lösung des XOR-Problems durch das generische Fuzzy-Perzeptronerkennen.

3.1 Beschreibung

Dieser Abschnitt beschreibt die einzelnen Komponenten des generischen Fuzzy-Perzeptrons genauer.

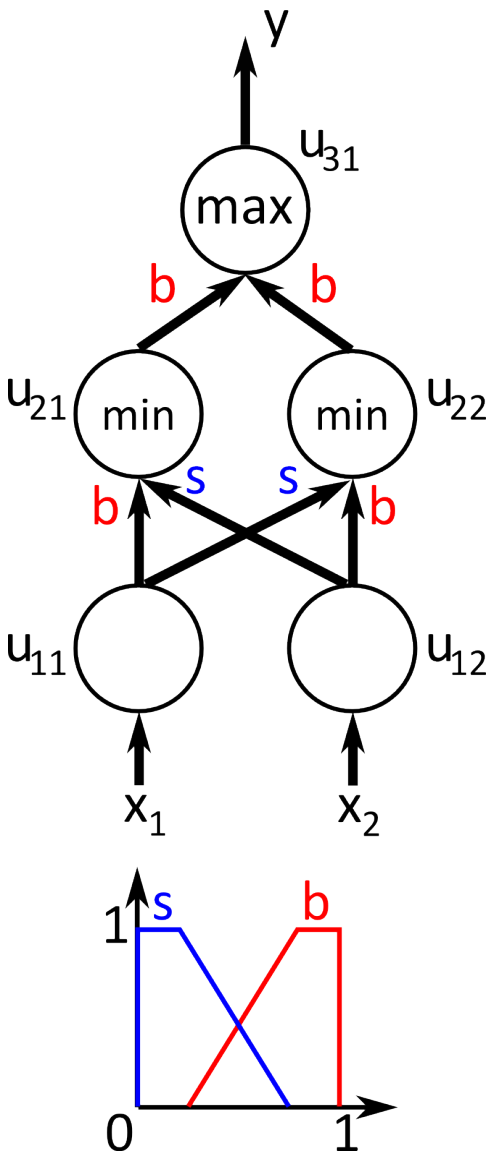

Abbildung 6: Generisches Fuzzy-Perzeptron zur Lösung des XOR-Problems

Schichten Die drei Schichten des generischen Fuzzy-Perzeptrons lassen sich jeweils als Menge von Neuronen beschreiben.

- Eingabeschicht $U_1 = \{x_1, \ldots, x_n\}$
- Versteckte Schicht $U_2 = \{R_1, \ldots, R_r\}$
- Ausgabeschicht $U_3 = \{y_1, \ldots, y_m\}$

Jedes Neuron verfügt dabei über drei Funktionen:

- Die Netzeingabefunktion net_u

x_1	x_2	$out_{u_{11}}$	$out_{u_{12}}$	$out_{u_{21}}$	$out_{u_{22}}$	$out_{u_{31}}$ = y
0	0	0	0	min(0,1)= 0	min(0,1)= 0	max(0,0)= 0
0	1	0	1	min(0,0)= 0	min(1,1)= 1	max(0,1)= 1
1	0	1	0	min(1,1)= 1	min(0,0)= 0	max(1,0)= 1
1	1	1	1	min(1,0)= 0	min(1,0)= 0	max(0,0)= 0

Tabelle 1: Ausgaben der Neuronen für das XOR-Problem

- Die Aktivierungsfunktion act_u
- Die Ausgabefunktion out_u

Die Verarbeitung je Neuron u erfolgt dabei in der Reihenfolge $net_u \rightarrow act_u \rightarrow out_u$.

Gewichte Das Gewicht der Verbindung zwischen zwei Neuronen unterschiedlicher Schichten ist definiert durch die Fuzzy-Menge μ für die Verbindungen zwischen Neuronen der Eingabeschicht und Neuronen der versteckten Schicht. Für die Verbindungen zwischen Neuronen der versteckten Schicht und der Ausgabeschicht erhalten ihr Gewicht über die Fuzzy-Menge ν. Das Gewicht w der Verbindung zwischen dem Neuron u und dem Neuron v der nachgelagerten Schicht lässt sich also wie folgt berechnen.

$$w_{vu} = \begin{cases} \mu_j^{(i)} & \text{für } u = x_i, v = R_j \\ \nu_j^{(k)} & \text{für } u = R_j, v = y_k \\ \text{undefiniert} & \text{sonst} \end{cases}$$

Mit $1 \leq i \leq n$, $1 \leq j \leq r$, $1 \leq k \leq m$
$\mu_j^{(i)} \in F(\mathbb{R})$, $\nu_j^{(k)} \in F(\mathbb{R})$
und $F(\mathbb{R}) :=$ die Menge aller Fuzzy-Mengen über $\mathbb{R}$

Eingabeschicht Für Neuronen der Eingabeschicht $u \in U_1$ lassen sich die drei Funktionen net_u, akt_u und out_u wie folgt beschreiben.

Die Netzeingabefunktion net_u stellt eine Abbildung von $\mathbb{R}$ auf $\mathbb{R}$ für die externe Eingabe ext_u dar.
$f_{net}^{(u)} : net_u = ext_u$

Die Aktivierungsfunktion act_u stellt eine Abbildung der Netzeingabe net_u von $\mathbb{R}$ auf $\mathbb{R}$ dar.
$f_{act}^{(u)} : act_u = net_u$

Die Ausgabefunktion out_u bildet die Aktivierung act_u von $\mathbb{R}$ auf $\mathbb{R}$ ab.
$f_{out}^{(u)} : out_u = act_u$

Versteckte Schicht Für Neuronen der versteckten Schicht $u \in U_2$ lassen sich die drei Funktionen net_u, akt_u und out_u wie folgt beschreiben.

Die Netzeingabefunktion net_u stellt eine Abbildung von $(\mathbb{R} \times F(\mathbb{R}))$ auf $[0,1] \in \mathbb{R}$. Sie verwendet dabei das Gewicht $w_{uu'}$ zwischen u und $u' \in U_1$ unter Bildung der T-Norm. Dieses Beispiel verwendet hierfür die Minimum-Norm min.

Die Aktivierungsfunktion act_u stellt eine Abbildung von $\mathbb{R}$ auf $\mathbb{R}$ für die Netzeingabe net_u dar.
$f_{act}^{(u)} : act_u = net_u$

Die Ausgabefunktion out_u bildet die Aktivierung act_u von $\mathbb{R}$ auf $\mathbb{R}$ ab.
$f_{out}^{(u)} : out_u = act_u$

Ausgabeschicht Für Neuronen der $u \in U_3$ lassen sich die drei Funktionen net_u, akt_u und out_u wie folgt beschreiben.

Die Netzeingabefunktion net_u verwendet die T-Conorm über die Ausgaben aller Neuronen der versteckten Schichten $u' \in U_2$ mit dem Gewicht $w_{uu'}$ zur Berechnung des Netzeingabewertes. Dieses Beispiel verwendet die Maximum-Conorm max.

Die Aktivierungsfunktion act_u stellt eine Abbildung von $F(\mathbb{R})$ auf $F(\mathbb{R})$ für die Netzeingabe net_u dar.
$f_{act}^{(u)} : act_u = net_u$

Die Ausgabefunktion out_u Durch eine geeignete Abbildung der Aktivierung act_u von $F(\mathbb{R})$ auf $\mathbb{R}$ erfolgt hier die Defuzzifizierung der Aktivierung act_u zu einem scharfen Wert.

3.2 Fuzzy-Fehlerbestimmung

Für ein Lernverfahren bildet die Bestimmung des Fehlers für eine bestimmte Eingabe bei gewünschter Ausgabe ein zentrales Element. Für ein Neuron u der Ausgabeschicht lässt sich der Fehler E für ein Muster p nach [BKKN03] wie folgt berechnen.

$$E_u^{(p)} = 1 - \exp\left(\beta\left(\frac{t_u^{(p)} - o_u^{(p)}}{range_u}\right)^2\right)$$

Unter Verwendung von

- $u \in U_3$ bezeichnet das Neuron der Ausgabeschicht.
- Der Zielwert t bezeichnet die gewünschte Ausgabe für das Muster p.
- Der Ausgabewert o repräsentiert die aktuelle Ausgabe des Neurons.

- $range_u$ bezeichnet die Differenz zwischen dem maximalem und dem minimalem Ausgabewert des Neurons u.
- β bildet einen multiplikativen Toleranzfaktor.

Aus dieser Formel ist leicht zu erkennen, dass es sich hierbei um ein Delta-Lernverfahren handelt. Stimmen Ausgabe und Zielwert überein, wird der Exponent im zweiten Term 0. Durch $e^0 = 1$ ergibt sich wie erwartet ein Fehler $E = 1 - 1 = 0$

3.3 Fuzzy-Backpropagation

Der Backpropagation-Algorithmus für dieses Perzeptron lässt sich nach [BKKN03] wie folgt formulieren.

1. Wähle ein beliebiges Muster p einer gegebenen festen Lernaufgabe und propagiere den Eingabevektor i^p
2. Bestimme
$$\delta_u^{(p)} = \begin{cases} sgn\left(t_u^p - out_u^p\right) * E_u^{(p)} & \text{für } u \in U_3 \\ \sum_{v \in U_3} act_u^{(p)} * \delta_v^{(p)} & \text{für } u \in U_2 \end{cases}$$
3. Bestimme
$$\Delta_p w_{vu} = f\left(\delta_u^{(p)}, act_u^{(p)}, net_u^{(p)}\right)$$

Wiederhole diese Schritte bis der Gesamtfehler

$$E = \sum_p \sum_{u \in U_3} E_u^{(p)}$$

hinreichend klein ist oder ein anderes Abbruchkriterium erfüllt wird.
Im zweiten Schritt lässt sich erkennen, wie der Fehler der Ausgabeneuronen $v \in U_3$ auf die Neuronen der versteckten Schicht $u \in U_2$ wirken.
Die Funktion f, die in Schritt drei Verwendung findet ist nicht genauer spezifiziert. Hier könnte beispielsweise eine Anpassung der Parameter der Zugehörigkeitsfunktionen der Fuzzy-Mengen stattfinden. Bei den Fuzzy-Mengen in Abbildung 6 könnte z. B. die Steigung der Geraden variiert werden.