Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Bachelorarbeit

Feridun Yildirim

Adaptiver Neuro-Fuzzy-Ansatz zur Diagnose

Fakultät Technik und Informatik
Department Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Feridun Yildirim

Adaptiver Neuro-Fuzzy-Ansatz zur Diagnose

Bachelorarbeit eingereicht im Rahmen Bachelorprüfung

im Studiengang Technische Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr.-Ing. Andreas Meisel
Zweitgutachter: Prof. Dr. Wolfgang Fohl

Abgegeben am 6. April 2017

Feridun Yildirim

Thema der Arbeit

Adaptiver Neuro-Fuzzy-Ansatz zur Diagnose

Stichworte

Adaptiver Neuro-Fuzzy-Inferenzsystem, MATLAB, WEKA, MLP, RBF-Netz, Diagnose

Kurzzusammenfassung

In der realen Computerumgebung ist es schwierig, eine Entscheidung zu treffen, die durch Unvollständigkeit und Ungenauigkeit charakterisiert ist. Mehrere Algorithmen und Technologien (Fuzzy-Logik, neuronale Netze, genetische Algorithmen etc.) wurden entwickelt, um eine akkurate Diagnose sicher zu stellen. In dieser Bachelorarbeit wird ein adaptiver Neuro-Fuzzy-Ansatz als Klassifikator für eine Diagnose benutzt. Viele Aspekte des Adaptiven Neuro-Fuzzy-Inferenzsystems (ANFIS) wie Architektur, Variationen der Architektur, Lernalgorithmen, Implementationen und medizinische Anwendungen werden vorgestellt. Für eine Realisierung an einem Fallbeispiel wurden mehrere Modelle mit unterschiedlichen Konfigurationen entwickelt und evaluiert. Die Leistung der Modelle sind mit der Wurzel der mittleren quadratischen Fehler (RMSE) ermittelt, um das beste Modell zu erhalten. Die Analyse der Diagnoseresultate und die Vergleiche mit MLP/RBFN demonstrieren eine vielversprechende Leistung für eine Modellierung von Diagnosesystemen.

Feridun Yildirim

Title of the paper

Adaptive neuro-fuzzy approach for diagnosis

Keywords

Adaptive neuro-fuzzy inference system, MATLAB, WEKA, MLP, RBFN, diagnosis

Abstract

It is difficult to derive a decision in a real-world computing environment, which is characterized by incompleteness, inaccuracy and imprecision. Several algorithms and technologies (soft computing techniques - fuzzy logic, neural networks, genetic algorithm etc.) have been constructed to ensure accurate diagnosis. In this thesis, an adaptive neuro-fuzzy approach is used as a classifier for diagnosis. Many aspects of the proposed adaptive neuro-fuzzy inference system (ANFIS) are introduced such as architecture, variations, learning algorithms, implementations and medical diagnostic applications. For case example, several models with different configurations are developed and evaluated. The performance of these models was measured by root-mean-square error (RMSE) to obtain the best fit model. Analysis of the diagnosis results and comparisons with MLP/RBFN demonstrates a promising performance for modeling of diagnostic systems.

Inhaltsverzeichnis

Abkürzungsverzeichnis

ABC	Artificial Bee Colony
ANFIS	Adaptives Neuro-Fuzzy-Inferenzsystem **(**Adaptive neuro-fuzzy inference system/ Adaptive network-based fuzzy inference system)
ARD	Automatic Relevance Determination
AWPSO	Adaptiv-Weighted-PSO
BChr	Bland Chromatin
CANFIS	Coactive neuro-fuzzy inference system
CART	Classification and Regression Trees
CAT	Cat Swarm Optimization
DEACS	Differential Evolution Ant Colony Search
EA	Evolutionärer Algorithmus
EZ	Epithelial Zellengröße
FA	Firefly-Algorithmus
FCM	Fuzzy C-Means
FFRLS	Forgetting-Factor-RLS
FIS	Fuzzy-Inferenzsystem
FNB	Feinnadelbiopsie
GA	Genetische Algorithmen
GD	Gradient descent
GZf	Gleichmäßigkeit der Zellenformen
GZg	Gleichmäßigkeit der Zellengröße
KNN	Künstliche neuronale Netze/Netzwerke
LM	Levenberg-Marquardt

LSE	Least Square Estimation
M	Mitose
MAdh	Marginale Adhäsion
MF	Zugehörigkeitsfunktion (Membership function)
MIMO	Multiple Input Multiple Output
MISO	Multiple Input Single Output
MLP	Mehrlagiges Perzeptron
MRT	Magnetresonanztomografie
NNcl	Normal Nucleolus
PPB	Probabilistic Patch-Based
PSO	Partikelschwarmoptimierung
QP	Quickprop
QPSO	Quamtum-behaved PSO
RBF-Netze	Radiale-Basisfunktionen-Netze
RLS	Recursive least square
RMSE	Wurzel der mittleren quadratischen Abweichung (Root-mean-square error)
Rprop	Resilient backpropagation
TSK	Takagi-Sugeno-Kang
UCI	University of California, Irvine
ZP	Zythoplasmaarme Zellen

1 Einleitung

Die Subjektivität eines Spezialisten ist in vielen Situationen ein Problem für eine akkurate Diagnose. Dabei beruft der Spezialist bei der Beurteilung auf sein Erfahrungswissen und kann in einem routinierten Ablauf oder schwierigen Fällen eine falsche bzw. vorschnelle Schlussfolgerung ziehen. Die Zeit ist ebenfalls ein ernst zu nehmendes Problem in Situationen, die eine zeitnahe Diagnose erfordern. So können computergestützte Diagnosen ein effektives Hilfsmittel für einen Spezialisten sein, um Diagnosen akkurat und schnell zu treffen. Einer dieser Mittel sind künstliche neuronale Netze, die in vielen Bereichen der Wissenschaft und Technologie ihre Anwendung finden. Die Einsatzgebiete von neuronalen Netzen sind weit gefächert, wie Bildverarbeitung, Klassifikation, Klangsynthese, Zeitreihenanalyse etc. und ist damit ein gutes Mittel für komplexe Anwendungen.

1.1 Zielsetzung

Das Ziel ist es, ein unterstützendes Diagnosemittel vorzustellen, die auch auf das Wissen des Spezialisten zurückgreift. Die vorliegende Arbeit beschränkt sich dabei auf die Diagnose mit einem Neuro-Fuzzy-System - im Speziellen auf das Adaptive Neuro-Fuzzy-Inferenzsystem (ANFIS). Hierbei soll das vorgestellte System und dessen Entwicklung bis zum heutigen Zeitpunkt beleuchtet werden. In dieser Arbeit werden dafür die unterschiedlichen Variationen, Lernalgorithmen, Implementationen und die medizinischen Diagnoseanwendungen des adaptiven Systems betrachtet. Am Beispiel der Brustkrebsdiagnose soll ein konkretes Neuro-Fuzzy-System realisiert und verschiedene Modelle untersucht und bewertet werden.

1.2 Struktur der Arbeit

Die Struktur der Arbeit gliedert sich, einschließlich der Einleitung, in sechs Kapitel.

- Das zweite Kapitel widmet sich der Thematik über die Grundlagen von künstlichen neuronalen Netzen und den Themen, die für das Verständnis der Arbeit notwendig sind.

- Im Fokus des dritten Kapitels steht das Adaptive Neuro-Fuzzy-Inferenzsystem (ANFIS), die in dieser Arbeit als Arbeitsgrundlage dient. Hierbei werden die Architektur des Systems vorgestellt und eine Übersicht über die Variationen der Architektur, der Lerntechniken und Implementationen des ANFIS verschafft.

- Darauf aufbauend wird im vierten Kapitel das System für die Diagnose vorgestellt. Zunächst wird ein Überblick über die Einsatzgebiete des Systems in der medizinischen Diagnose verschafft und im Anschluss auf die Nutzung des Ansatzes mit der MATLAB Software eingegangen.

- Im fünften Kapitel findet eine Realisierung einer Brustkrebsdiagnose mit unterschiedlichen MLP-, RBF- und ANFIS-Modellen statt, die im Anschluss untersucht und bewertet werden.

- Im letzten Kapitel werden die Inhalte der Arbeit kurz zusammengefasst. Abschließend erfolgen ein Fazit und ein Ausblick des vorgestellten Systems.

2 Grundlagen

Im folgenden Kapitel werden die Grundlagen, die für die vorliegende Arbeit zum Verständnis des Themas erforderlich sind, erläutert. Im ersten Unterpunkt 2.1 wird die Thematik über künstliche neuronale Netze (KNN) vorgestellt sowie ein grober Überblick über KNN verschafft und die Einteilung in deren Untergruppen kurz angeschnitten. Im Unterpunkt 2.2 werden Fuzzy-Systeme mit dem Fokus auf die Fuzzy-Logik und im Speziellen das Fuzzy-Regelsystem von Takagi-Sugeno-Kang behandelt. Die Kombination aus Fuzzy-Systemen und neuronalen Netzen wird im Unterkapitel 2.3 thematisiert. Dabei werden die Vorteile eines solchen Neuro-Fuzzy-Systems in den Vordergrund gestellt und die Abgrenzung zwischen zwei unterschiedlichen Modellen vorgenommen. Im letzten Unterpunkt 2.4 werden Partitionierungsansätze zur Bestimmung der Regeln eines Fuzzy-Systems erklärt.

2.1 Künstliche neuronale Netze

Künstliche neuronale Netze bilden die Struktur und Funktionsweise der Nervensysteme von Tieren und Menschen ab. Hierbei ist der Fokus nicht primär die Nachbildung von biologischen neuronalen Netzen, sondern eher eine Abstraktion der Informationsverarbeitung. Die KNNs bestehen aus einer großen Anzahl an parallel laufenden Einheiten. Diese Einheiten sind Neuronen, die über gerichtete Verbindungen Signale senden.

McCulloch und Pitt [1] veröffentlichten ein Modell, dass die Charakteristiken eines biologischen Neurons zur Übertragung und zum Empfangen von Informationen modelliert. Dieses Modell dient seither als Referenz für viele KNN-Modelle. Aus mathematischer Sicht ist ein

Neuron ein Prozessor mit einer bestimmten Anzahl an Ein- und Ausgaben, die zwei Zustände besitzt: Aktiv (d. h., das Neuron feuert) und passiv (d. h., das Neuron feuert nicht). Der mathematische Ausdruck eines Neurons, die in der heutigen Literatur sehr häufig zu sehen ist, kann aus der Gleichung (2.1) entnommen werden.

$$u_{(k)} = \sum_{j=1}^{n} w_{kj} x_j \quad \wedge \quad y_{(k)} = \varphi\left(u_{(k)} - \theta_{(k)}\right) \tag{2.1}$$

Hierbei ist $u_{(k)}$ der Ausgang der Summenfunktion des Modells, x_j ist das Eingabesignal der Synapse j und w_{kj} ist die Gewichtung von der Synapse j zum Neuron k. Der Ausgang des Neurons wird durch $y_{(k)}$ repräsentiert, die von der Aktivierungsfunktion φ und dem Schwellenwert $\Theta_{(k)}$ abhängig ist.

Die Eingabewerte eines Neurons werden mit einem Gewicht[1] multipliziert und gibt darauf eine Ausgabe. Wenn die Summe über das Produkt aller Eingaben mit den entsprechenden Gewichten größer als der Schwellenwert $\Theta_{(k)}$ ist, wird die Ausgabe auf aktiv gesetzt, sonst ist die Ausgabe passiv. Der Schwellenwert kann auch durch einen weiteren Eingang x_0, dem sogenannten Bias, dargestellt werden. In der Abbildung 2.1 ist ein Neuron mit seinen Elementen zu sehen.

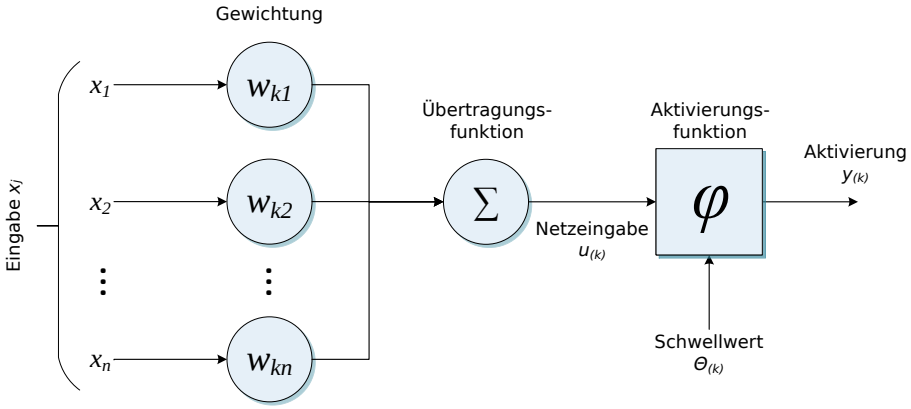

Abbildung 2.1: Darstellung eines Neurons (ohne Bias)

Durch die Verbindungen der Neuronen mit anderen Neuronen bildet sich die Netzarchitektur

[1] Ein Gewicht kann positiv oder negativ sein, und entsprechend werden die Eingaben verstärkt bzw. gehemmt.

zusammen. Die zusammengesetzten Neuronen werden in ein Schichtenmodell unterteilt. So besteht ein KNN grundsätzlich[2] aus drei Schichten, die die Neuronen in Eingabe-, Ausgabe- und in versteckte Neuronen einteilt. Es wird damit festgelegt, welche Neuronen die Eingabedaten erhalten und welche die Ausgabe der KNN ausgeben. Die restlichen Neuronen haben keinen Kontakt nach außen und werden als versteckte Neuronen (Hidden-Layer) bezeichnet. In der ersten Schicht werden die ankommenden Daten zur nächsten Schicht übertragen. Diese Schicht kann mehr als ein Neuron besitzen und ist nicht an Regeln gebunden, die festlegt, wie viele Neuronen die Schicht haben muss. Vielmehr hängt es davon ab, wie viele Einträge benutzt werden sollen.

Die zweite Schicht ist die versteckte Schicht. Diese Schicht beinhaltet Neuronen, die entweder Daten oder Signale von der vorherigen Schicht (Eingangs- oder versteckte Schicht) empfangen. In den Schichten werden die Daten bzw. Signale durch Funktionen verarbeitet. Die versteckten Neuronen können in Vielzahl (eins oder mehr) vorhanden sein. Es hängt u. a. von der Komplexität bzw. Zweckmäßigkeit des vorliegenden Falles ab. Die bearbeiteten Resultate dieser Schicht werden zur Ausgangsschicht weitergeleitet. Die Ausgangsschicht bestimmt die Validität der Daten, die durch die Begrenzung der Aktivierungsfunktionen analysiert sind. Die Einteilung in Schichten eines KNN ist in der Abbildung 2.2 aufgeführt.

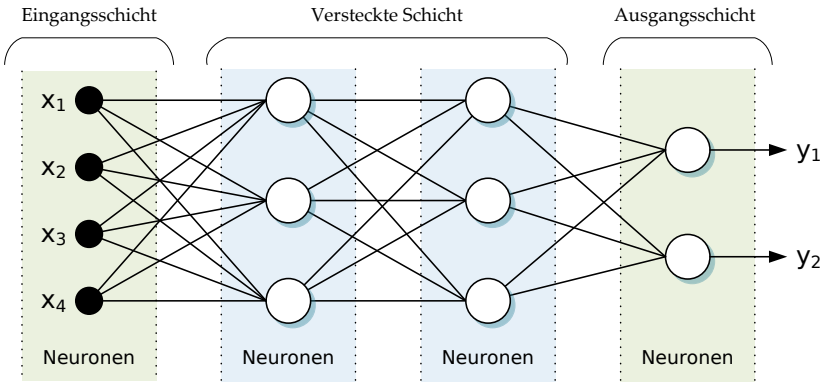

Abbildung 2.2: Schichten eines künstlichen neuronalen Netzes mit mehreren Eingängen und mehreren Ausgängen (MIMO)

Je nachdem wie die Neuronen miteinander verbunden sind, wird eine KNN-Architektur in zwei Typen von Netzen unterteilt: Vorwärts gerichtete Netze (Feed-forward-Netze) und rückgekoppelte neuronale Netze (Rekurrente-Netze) [2]. Die Feed-forward-Netzwerke besitzen

[2] Ausgenommen sind die einstufigen Netze ohne versteckte Neuronen.

in seiner Architektur keine Rückführung zu den vorherigen Schichten. Die Daten bzw. Signale verlaufen nur in eine Richtung und haben dadurch keinen Einfluss auf die Schichten davor. Die Netzarchitektur kann aus einem einstufigen Netz oder mehrstufigen Netz bestehen. Ein einstufiges Netz setzt sich aus den Ein- und Ausgabeneuronen, wogegen das mehrstufige Netz zusätzlich aus den versteckten Neuronen zusammensetzt. Zu den Feed-forward-Netzen gehört u. a. das einlagige Perzeptron, mehrlagige Perzeptron (MLP) und die Radiale-Basisfunktionen-Netze (RBF-Netze). Der andere Typ von Netzen ist das Rekurrente-Netz. Dieses Netz hat ein ähnliches Design wie die Feed-forward-Netze. Jedoch ist die Architektur mit einer zusätzlichen Rückkopplung versehen. Dies hat zufolge, dass die Daten bzw. Signale vorwärts propagieren können und der Feed-back kann für die Eingänge der vorherigen Neuronen dienen. Dieser Typ von KNN kann für dynamische Anwendung benutzt werden. Zu den Rekurrenten-Netzen gehören u. a. Elman-Netze, Jordan-Netze und Hopfield-Netze. Eine Übersicht der Einteilung in Feed-forward- und Rekurrente-Netze ist anhand der Abbildung 2.3 dargestellt.

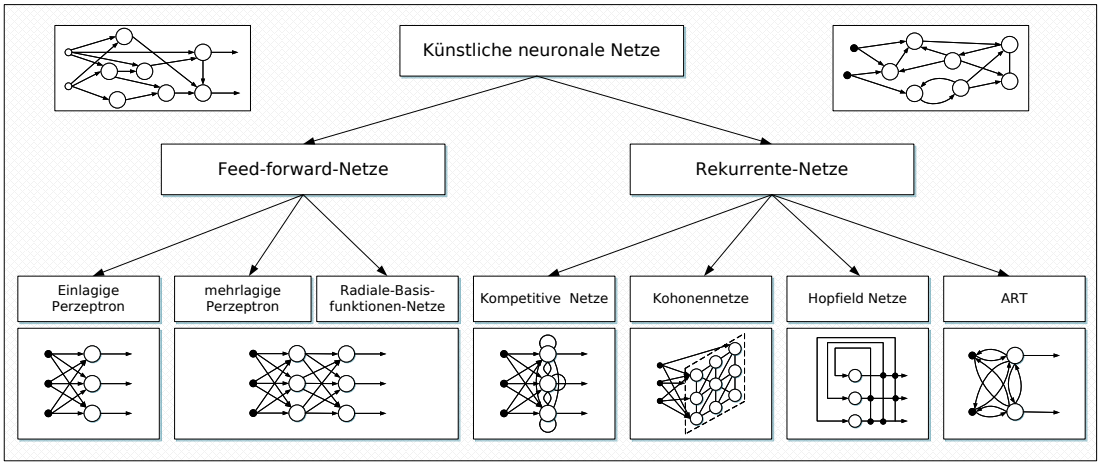

Abbildung 2.3: Eine Taxonomie von Feed-forward und Rekurrenten-Netzwerken [2]

Die bekannteste Form eines KNN ist das mehrlagige (bzw. mehrschichtige) Perzeptron. Wenn man die Ausgabeschicht als Eingabe für weitere Neuronenschichten benutzt und diese mehrmals reiht, erhält man einen MLP. Das Lernen erfolgt zumeist mit der Fehlerrückführung (Backpropagation). Dabei werden die Gewichte der Verbindung verändert, sodass das MLP das erstrebte Ziel nach einem überwachten Training erreicht.

Die Radiale-Basisfunktionen-Netze sind vorwärts gerichtete Netze mit einer strengen Schichtstruktur. Die Anzahl der Schichten ist stets in drei Stufen eingeteilt. Demnach existiert in einem RBF-Netz nur eine versteckte Schicht. In der versteckten Schicht findet die

Informationsverarbeitung des Netzes statt. Des Weiteren unterscheidet sich das RBF-Netz zum MLP durch eine andere Netzeingabefunktion $u_{(k)}$ und Aktivierungsfunktion φ. Hier kommt die radiale Basisfunktion zur Nutzung. Die Funktion beeinflusst primär die Ausgabe eines Netzes, indem es jedem Neuron eine Art „Einzugsgebiet“ zuordnet [3].

2.2 Fuzzy-Systeme

Der Mensch verwendet vorwiegend unscharfe bzw. unpräzise Konzepte, um seine Umgebung zu beschreiben und sind durch einfache Modelle schwer zu bestimmen. So sind auch Beschreibungen der Umgebung durch Attribute kontextabhängig und benötigen eine genauere Betrachtung. Deshalb ist der Hauptgedanke an dieser Stelle der Umgang mit Mengen, deren Elemente nur graduell zu einer Menge gehören. Zu der Problematik veröffentlicht Lotfi A. Zadeh [4] seine Arbeit über die Fuzzy-Mengen. Die Strategie des Fuzzy-Systems ist es, anstatt scharfe bzw. zweiwertige Unterscheidungen zu machen, einen gewissen Grad einer Zugehörigkeit zu treffen. Mit diesem Ansatz lassen sich Entscheidungen treffen, in denen widersprüchliche oder auch unvollständige Information vorhanden sind. Es ist wichtig zu unterscheiden, dass die Graduierung nicht mit dem Konzept der Wahrscheinlichkeit gleichzusetzen ist [3]. Um den Unterschied klar zu machen, veranschaulicht Bezdek [5] mit einem Beispiel in seiner Publikation über Fuzzy-Modelle. In seinem Beispiel soll eine Person die Wahl zwischen zwei Flaschen (A und B) treffen. Dabei ist L die Menge der ungiftigen Flüssigkeiten. Die Flasche A soll mit einer Wahrscheinlichkeit zu 0,91 zu der Menge L gehören und die Flasche B einen Zugehörigkeitsgrad von 0,91 besitzen. So kann die Wahrscheinlichkeit der Flasche A daher beruhen, dass aus 10 Flaschen ungefähr 9 Flaschen mit Wasser gefüllt sind und nur eine Flasche mit einer giftigen Flüssigkeit gefüllt ist. Andererseits bedeutet eine Zugehörigkeit von 0,91, dass die Flasche zu einem Prozentanteil von 91 % trinkbar ist – z. B. bedingt durch das Ablaufen des Haltbarkeitsdatums. Somit ist die Wahl zu der Flasche B eher zu empfehlen.

2.2.1 Fuzzy-Mengen und Fuzzy-Logik

Die Fuzzy-Mengen, auch als unscharfe Mengen bezeichnet, bilden die Grundlage für die Fuzzy-Logik. Bei den Fuzzy-Mengen muss die Angabe getroffen werden, inwiefern die Elemente zu einem bestimmten Grad zu der Fuzzy-Menge angehören. So ist die Definition 2.1 einer Fuzzy-Menge wie folgt [3].

Definition 2.1:

Eine Fuzzy-Menge oder Fuzzy-Teilmenge μ der Grundmenge X ist eine Abbildung $\mu: X \rightarrow [0, 1]$, die jedem Element $x \in X$ seinen Zugehörigkeitsgrad $\mu(x)$ zu μ zuordnet. Die Menge aller Fuzzy-Mengen von X wird mit $F(X)$ bezeichnet.

In einer Fuzzy-Menge μ mit endlichen, diskreten Grundmengen $X = \{x1, ..., xn\}$, kann nur durch Angabe der Zugehörigkeitsgrade $\mu(x)$ spezifiziert werden.

$$\mu \hat{=} \left\{\left(x_1, \mu\left(x_1\right)\right), \ldots, \left(x_n, \mu\left(x_n\right)\right)\right\} \tag{2.2}$$

Die Fuzzy-Mengen werden in der Regel durch linguistische Ausdrücke umschrieben, um einen gewissen sprachlichen Zusammenhang zu erzielen. So werden die Ausdrücke wie beispielsweise „langsam“, „schnell“ als linguistische Terme bezeichnet und der Ausdruck „Geschwindigkeit“ ist die linguistische Variable der Fuzzy-Menge. Diese linguistischen Ausdrücke werden letztlich benutzt, um einen unscharfen Wert/Intervall zu beschreiben.

Jedes Element x aus der Menge X wird durch eine Funktion beschrieben. Diese Funktionen können in unterschiedlicher Form dargestellt werden. So können Fuzzy-Mengen durch wenige Parameter beschrieben werden. Bespiele für eine parametrische Fuzzy-Menge sind die Dreiecksfunktionen, Trapezfunktionen und die Glockenkurve. In der Abbildung 2.4 sind die drei unterschiedlichen Funktionen mit seinen jeweiligen Parametern aufgelistet.

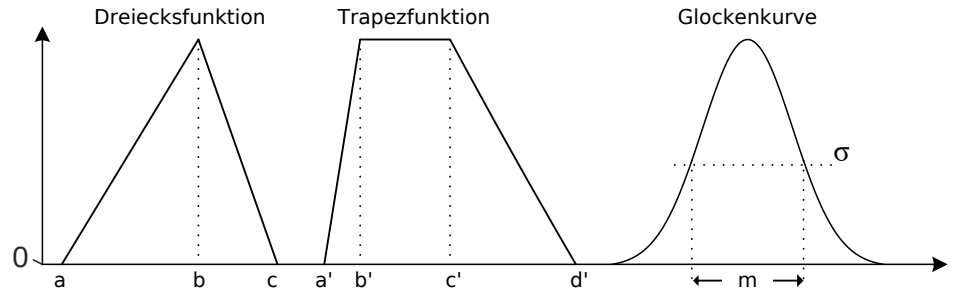

Abbildung 2.4: Die grafische Darstellung der Dreiecksfunktion, Trapezfunktion und der Glockenkurve [3]

Fuzzy-Logik im engeren Sinne[3] wird für mengentheoretische Operationen benötigt. Die logischen Verknüpfungen wie Konjunktion (UND), Disjunktion (ODER) und Negation (NICHT) bilden die Grundlage für die Operationen einer Fuzzy-Menge wie Durchschnitt, Vereinigung und Komplement. Um die logischen Operationen zu modellieren, werden Funktionsklassen der T-Norm und T-Conorm benutzt. Die Operationen von Fuzzy-Mengen sind aus der Abbildung 2.5 zu entnehmen.

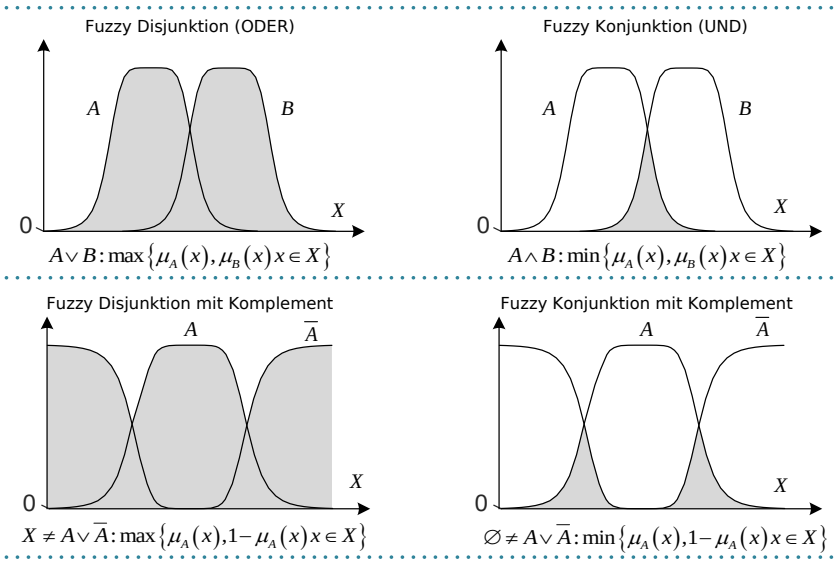

Abbildung 2.5: Logische Operationen auf Fuzzy-Mengen

2.2.2 Takagi-Sugeno-Kang Fuzzy-Regelsystem

Ein Fuzzy-Regler soll physikalische Eingangssignale eines Prozesses durch linguistische Begriffe über Zugehörigkeitsfunktionen und einer Wenn-Dann-Beziehung bewerten. Dieser Ansatz hat einen großen Erfolg der Fuzzy-Systeme in der industriellen und kommerziellen Anwendung nach sich gezogen [3].

[3] Im weiteren Sinne bezeichnet die Fuzzy-Logik für alle Anwendungen und Theorien bei denen Fuzzy-Menge vorkommt. Fuzzy-Logik im engeren Sinne befasst sich aus der Sicht der mehrwertigen Logik mit den logischen Kalkülen und deren Deduktionsmechanismen [3].

Der Takagi-Sugeno-Kang-Ansatz (TSK) [6] ist auf der Fuzzy-Logik basierender Regler, dessen Verhalten durch Regeln beschrieben werden. Eine solche Regel setzt sich aus einer Prämisse und einer Konklusion zusammen (2.3).

$$Regel: \underbrace{WENN \left\{ \begin{matrix} Prämisse \\ (Vorbedingung) \end{matrix} \right\} DANN \left\{ \begin{matrix} Konklusion \\ (Schlussfolg.) \end{matrix} \right\}}_{Implikation\ (Fuzzy\text{-}Regel)} \tag{2.3}$$

Dabei ist die Konklusion einer einzigen Regel von den Eingangsgrößen abhängig. Die Konklusion im TSK ist keine unscharfe Menge, sondern besteht aus einer linearen Funktion[4] der Form

$$f_i = \boldsymbol{a}_i^T \cdot \boldsymbol{x} + r_i \tag{2.4}$$

mit dem Vektor $\boldsymbol{x}$ aus der Menge der Eingabegrößen und den Parametern $\boldsymbol{a}_i$ entsprechend der Anzahl der Eingaben. Demnach hat jede Regel seine eigenen Parameter, und die Anzahl der Parameter in der Konklusion entspricht der Anzahl der Eingänge+1. Die Zugehörigkeitsfunktionen in der Prämisse sind durch eine UND-T-Norm verbunden.

Ein Beispiel mit zwei Eingängen (x_1, x_2) und jeweils zwei Funktionen (A_1 und A_2 für x_1; B_1 und B_2 für x_2) für jeden Eingang besteht aus maximal[5] vier Regeln ($2^N = 2^2 = 4$), die wie folgt aussieht.

$$\begin{aligned}
R_1 &: x_1\ ist\ A_1\ und\ x_2\ ist\ B_1\ dann\ f_1 = p_1x_1 + q_1x_2 + r_1 \\
R_2 &: x_1\ ist\ A_1\ und\ x_2\ ist\ B_2\ dann\ f_3 = p_2x_1 + q_2x_2 + r_2 \\
R_3 &: x_1\ ist\ A_2\ und\ x_2\ ist\ B_2\ dann\ f_3 = p_3x_1 + q_3x_2 + r_3 \\
R_4 &: x_1\ ist\ A_2\ und\ x_2\ ist\ B_1\ dann\ f_4 = p_4x_1 + q_4x_2 + r_4
\end{aligned}$$

[4] Ausgenommen ist ein spezieller Fall von TSK bei denen die Fuzzy-Mengen durch Konstanten ersetzt werden [3]. Wenn die Funktion dadurch eine Konstante ist, spricht man von TSK 0. Ordnung. Dies entspricht funktional, unter kleinen Einschränkungen, dem RBF-Netz [7].

[5] Je nach Partitionierung und Typ des TSK kann es unterschiedlich sein.

2.3 Neuro-Fuzzy-Systeme

In den vorherigen Abschnitten wurden neuronale Netze und Fuzzy-Systeme kurz vorgestellt. Beide Systeme haben ihre Stärken und ihre Schwächen in unterschiedlichen Bereichen der Modellierung von Systemen. Demnach ist es wünschenswert, die Stärken der einen Technik durch die Andere zu kompensieren. So kombiniert ein Neuro-Fuzzy-System beide Eigenschaften von Fuzzy-Systemen und neuronalen Netzen. In der Tabelle 2.1 ist ein Vergleich beider Systeme aufgeführt.

Tabelle 2.1: Vergleich von neuronalen Netzen und Fuzzy-Systemen [8]

Neuronale Netze	Fuzzy-Systeme
kein mathematisches Modell notwendig	kein mathematisches Modell notwendig
kein Regelwissen notwendig	A-priori (Regel-)Wissen nutzbar
mehrere Lernalgorithmen stehen zur Verfügung	nicht lernfähig
Black-Box-Verhalten	einfache Interpretation und Implementierung

Damit versucht der Neuro-Fuzzy-Ansatz, die Transparenz mit dem Regelwissen von Fuzzy-Systemen mit der Lernfähigkeit von neuronalen Netzen zusammenzubringen. Das daraus resultierende System soll die linguistischen Regeln und die Zugehörigkeitsfunktionen erlernen und entsprechend optimieren.

In der Literatur [3; 8] wird zwischen zwei Typen von Neuro-Fuzzy-Systemen, wie es in der Abbildung 2.6 zu sehen ist, unterschieden: Kooperative Neuro-Fuzzy-Systeme und hybride Neuro-Fuzzy-Systeme. In einem kooperativen Neuro-Fuzzy-System ist das neuronale Netz sowie das Fuzzy-System unabhängig voneinander. Beide bestehen als eigenes System und sind nicht als Ganzes zu betrachten. Die Verbindung von neuronalen Netzen und Fuzzy-Systemen besteht in der Bestimmung und Optimierung der Parameter des Fuzzy-Systems durch das neuronale Netz. Das daraus resultierende Fuzzy-System benötigt nach der Optimierung, die neuronale Komponente nicht. Ein kooperatives System kann auf verschiedene Weisen realisiert werden.

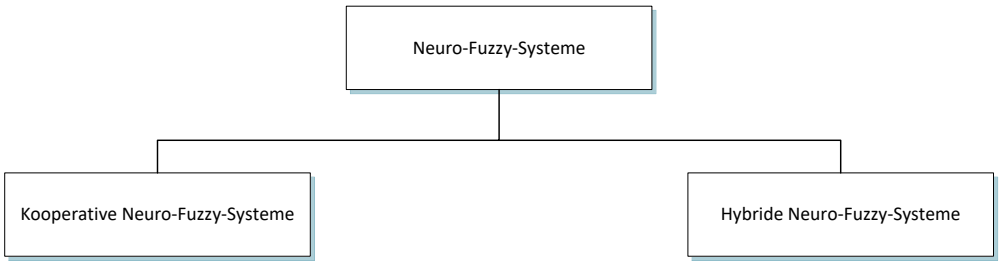

Abbildung 2.6: Typen von Neuro-Fuzzy-Systemen

(a) Das neuronale Netz erlernt mit den Trainingsdaten die Fuzzy-Mengen. Dies geschieht in der Regel durch Anpassen der Zugehörigkeitsfunktionen mit einem neuronalen Netz, die zur Modellierung von linguistischen Termen genutzt werden. Die Fuzzy-Menge wird dann in Folge Off-line bestimmt. Mit den Fuzzy-Mengen vom neuronalen Netz und den vorgegebenen Regeln setzt sich das Fuzzy-System zusammen.

(b) Ein weiteres Modell eines kooperativen Systems benutzt das neuronale Netz zur Ermittlung der linguistischen Regeln. Die Regeln werden grundsätzlich durch ein Clustering-Verfahren definiert bzw. erlernt. Wie auch in der Aufzählung (a) wird zuerst das neuronale Netz verarbeitet und die Fuzzy-Regeln Off-line bestimmt.

(c) Die Parameter der Zugehörigkeitsfunktionen werden On-line durch das neuronale Netz ermittelt. Für dieses Modell müssen daher die Zugehörigkeitsfunktionen und die entsprechenden Fuzzy-Regeln im Voraus bestimmt werden. Zudem muss ein Fehlermaß definiert werden, um den Lernvorgang eines neuronalen Netzes verbessern zu können.

(d) Im letzten Modell bestimmt das neuronale Netz die Gewichte für die Fuzzy-Regeln. Diese kann sowohl Off-line oder auch On-line erfolgen. Das Gewicht einer Regel skaliert damit die Regelausgabe.

In einem hybriden Neuro-Fuzzy-System werden beide Systeme (neuronale Netze und Fuzzy-Systeme) zu einem System vereint. D. h., das hybride System besteht, nicht wie in einem kooperativen System, aus einzelnen Komponenten, sondern ist als Ganzes zu sehen. Somit ist die Trennung in Teilsysteme nicht möglich. In diesem Kontext wird das hybride System als eine spezielle Form des neuronalen Netzes interpretiert. Der große Vorteil eines hybriden Systems besteht darin, dass es nicht mehr, wie beim kooperativen Netz, untereinander kommunizieren muss. Ein hybrides System kann sowohl Off-line als auch On-line lernen. Hierbei interpretiert das neuronale Netz die Regelbasis des Fuzzy-Systems. Die Fuzzy-Mengen können als Gewichte betrachtet werden. Die Ein- und Ausgabevariablen als auch die

Regeln werden als Neuronen modelliert. Für den Lernvorgang können die Neuronen eingefügt oder entfernt werden. Das ANFIS-Modell, das in die Kategorie hybrides Neuro-Fuzzy-Systeme fällt, wird in dieser Arbeit für Diagnosezwecke genutzt.

2.4 Clustering – Partitionierung

Um die Fuzzy-Regeln eines Fuzzy-Systems zu modellieren, benötigt es in der Regel das Wissen eines Experten. Demzufolge ist die Partitionierung des Eingaberaumes ein schwieriges und zeitintensives Unterfangen. Es gibt verschiedene Partitionierungsansätze, um die Regelbasis zu definieren. In der Abbildung 2.7 ist die generelle Einteilung von Partitionierung aufgelistet. Für diese Arbeit werden drei unterschiedliche Ansätze betrachtet, die in den folgenden Unterkapiteln erläutert werden.

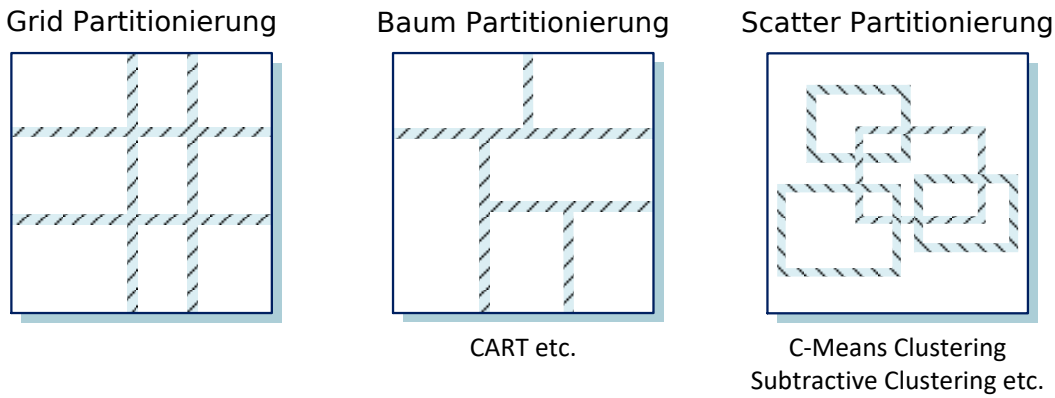

Abbildung 2.7: Unterschiedliche Methoden zur Partitionierung des Eingaberaums

2.4.1 Grid-Partitionierung

Bei der Grid-Partitionierung werden alle möglichen Kombinationen der Zugehörigkeitsfunktionen von allen Eingängen durchgelaufen. Dieser Ansatz ist hauptsächlich für Modelle mit kleinem Eingaberaum gedacht, die mit wenigen Zugehörigkeitsfunktionen (pro Eingang) auskommen. Beide Aspekte, die Anzahl der Eingaben und die Anzahl der Zugehörigkeitsfunktionen sind von Bedeutung für die Dimensionierung des Fuzzy-Systems.

Die maximale Anzahl von unterschiedlichen Regeln m und n Eingängen mit jeweils q Zugehörigkeitsfunktionen ist aus der Formel (2.5) zu entnehmen.

$$Maximale\ Anzahl\ von\ Regeln:\ \ m = q^n \tag{2.5}$$

Eine große Anzahl an Eingaben, wie beispielsweise zehn Eingaben mit zwei Zugehörigkeitsfunktionen, führt zu 1024 Regeln. Man spricht an dieser Stelle auch von „Fluch der Dimensionalität“. Für ein Modell mit großem Eingaberaum muss folglich ein anderes Verfahren gewählt oder der Eingaberaum entsprechend dezimiert werden. Die folgenden Unterpunkte bieten in dieser Hinsicht eine Alternative zur Grid-Partitionierung an.

2.4.2 Subtractive-Clustering

Das Subtractive-Clustering ist eine weitere Methode, um Regeln aus den Eingabe- und Ausgabedaten zu generieren. Die Methode hilft bei der Suche von Clusterzentren mittels Ein- und Ausgabepaare. Damit lassen sich die Regeln ermitteln, da jedes Cluster[6], ein Hinweis für eine Regel ist. Mit dem Wissen wird die Anzahl der linguistischen Variablen ermittelt. Zusätzlich liefert die Methode die Initialwerte der Prämissenparameter. Dies kann zu einer schnelleren Konvergenz beim Training führen, da die Werte durch das Verfahren schon gut optimiert sind [9].

Gegeben sei eine Menge von n Datenpunkten $\{x1, x2, \ldots, xn\}$ in einem M dimensionalen Raum. Jeder Datenpunkt ist ein mögliches Clusterzentrum. Die Nachbarn-Dichte D_i in einem Datenpunkt x_i in Abhängigkeit der Abstände zu allen benachbarten Punkten x_j ist durch die Gleichung (2.6) definiert.

$$D_i = \sum_{j=1}^{n} e^{-\frac{\|x_i - x_j\|^2}{(r_a/2)^2}} \tag{2.6}$$

Dabei definiert der Radius r_a den Umkreis der Nachbarn. Das bedeutet, je höher die Anzahl der benachbarten Punkte x_j im festgelegten Radius r_a und je kleiner die Distanzen[7] zwischen x_i und x_j ist, desto höher wird die Dichte D_i. Nachdem alle Werte D_i für alle Punkte x_i ermittelt sind, wird der Datenpunkt x_{cz} mit der höchsten Dichte D_{cz} als Clusterzentrum gewählt. Damit

[6] gefundenen Gruppen von „ähnlichen“ Objekten

[7] die Euklid’sche Distanz $\|.\|$

ist der erste Datenpunkt x_{c1} mit seiner Dichte D_{c1} initialisiert. Für die verbleibenden Punkte x_k im Radius r_b werden die Dichte D_k mit der Formel (2.7) neu berechnet. Wobei sich der Radius r_b aus dem Produkt von r_a und einem sogenannten „squash" Faktor η zusammensetzt.

$$D_k = D_k - D_{cz} \cdot e^{-\frac{\|x_k - x_{cz}\|^2}{(r_b/2)^2}} \tag{2.7}$$

Durch dieses Update wird verhindert, dass die Zentren zu dicht beieinanderliegen. Da die Dichte eines Punktes x_k reduziert wird, je näher es am ermittelten Clusterzentrum liegt. Der Prozess mit den Gleichungen (2.6) und (2.7) wird so lange wiederholt, bis das vorher festgelegte Abbruchkriterium erfüllt ist. Das Abbruchkriterium kann beispielsweise die Bedingung $D_{cz}^1 < s \cdot D_{cz}^j$ sein, worin D_{cz}^j die aktuelle Dichte, D_{cz}^1 die anfänglich gefundene Dichte und s der vorher definierte Schwellwert ist.

2.4.3 Fuzzy C-Means Clustering

Fuzzy C-Means (FCM) ist eine Methode, die es erlaubt, ein Objekt mithilfe von Zugehörigkeitsgraden auf mehrere Cluster zuzuordnen. Die Methode basiert darauf, die Funktion (2.8) zu minimieren.

$$J(U,V) = \sum_{i=1}^{n}\sum_{j=1}^{c}\left(\mu_{ij}\right)^m \left(d_{ij}\right)^2 \tag{2.8}$$

Dabei sind $(d_{ij})^2$ die Euklid'sche Distanzen zwischen den Punkten x_i und den Clusterzentren aus der Matrix $V = v_1, v_2, \ldots, v_c$. Hierbei gibt c die Anzahl der Cluster und n die Anzahl der Datenpunkte an. Die Matrix U beinhaltet die Zugehörigkeitsgrade μ_{ij}. Der unscharfe Index m gibt an, wie scharf ein Objekt den Clustern zugeordnet ist. Die Funktion J wird durch die Minimierung der Werte v_j und μ_{ij} mit der Formel aus (2.9) und (2.10) bestimmt.

$$v_j = \frac{\sum_{i=1}^{n}\left(\mu_{ij}\right)^m x_i}{\sum_{i=1}^{n}\left(\mu_{ij}\right)^m} \tag{2.9}$$

$$\mu_{ij} = \frac{1}{\sum_{k=1}^{c} \left(\frac{d_{ij}}{d_{jk}} \right)^{\frac{2}{m-1}}} \tag{2.10}$$

Die Anordnung der Objekte zu einem Cluster wird so vorgenommen, dass der Abstand $(d_{ij})^2$ minimal ist. Die Formel (2.9) und (2.10) werden solange iteriert, bis die Abbruchbedingung $\left\| U^r - U^{r-1} \right\| < s$ erfüllt ist. Dabei ist r der aktuelle Iterationsschritt und s der vordefinierte Schwellwert.

3 Adaptives Neuro-Fuzzy-Inferenzsystem

Das folgende Kapitel befasst sich mit einer Klasse aus der Kategorie Neuro-Fuzzy-Netze. Dabei werden die Architektur und der Lernprozess eines adaptiven Netzes beleuchtet. Die zugrunde liegende Netzwerkstruktur ist eine Obermenge von neuronalen Netzen mit überwachter Lernfähigkeit.

Jyh-Shing Roger Jang [7] veröffentlichte eine Architektur und einen Lernalgorithmus, welcher neuronale Netze mit Fuzzy-Logik verbindet. Es ist ein Modell, welches die Stärken dieser Methoden in sich vereint. Das ANFIS bedient sich der neuronalen Lernregel zur Identifizierung und Anpassung der Parameter bzw. der Struktur eines Fuzzy-Inferenzsystems (FIS). Das ANFIS-Modell beseitigt durch die Kombination die Grundproblematik in Fuzzy-Systemen: Die Frage nach der Bestimmung der Parameter einer Zugehörigkeitsfunktion (MF) und der Entwurf einer Fuzzy-Regelbasis. Des Weiteren bietet das Modell folgende Vorteile:

- schnelles und akkurates Lernen,
- einfache Implementierung des Modells,
- keine bedingte Notwendigkeit von Expertenwissen,
- linguistische Transparenz und Interpretierbarkeit mittels Fuzzy-Logik,
- keine Black-Box,
- ermöglicht das Einbringen von linguistischem/numerischem Wissen.

Durch diese Besonderheiten ermöglicht das ANFIS-Modell eine große Breite an Anwendungen [10].

Das Modell basiert auf einer Fuzzy-Regelbasis wie sie in Takagi-Sugeno-Kang-Reglern (TSK) in der Form

$$Regel_i\text{: } Wenn \left(x_1 \; ist \; A_1^{(i)}\right) \; \ldots \; und \; \ldots \; \left(x_n \; ist \; A_n^{(i)}\right) \; dann \; f_i = \boldsymbol{a}_i^T \cdot \boldsymbol{x} + r_i \tag{3.1}$$

verwendet wird. Der Vektor $\boldsymbol{x} \in \mathbb{R}^n$ sind die Eingabewerte der ANFIS-Struktur und $(\boldsymbol{a}_i, r_i)$ die Koeffizienten der Konklusion des TSK-Reglers.

3.1 Architektur von ANFIS

Ein adaptives Netzwerk ist ein mehrschichtiges Feed-forward-Netzwerk, dessen Knotenpunkte durch Kanten verbunden sind. Jeder Knoten erfüllt seine spezielle Aufgabe, die bei entsprechenden Eingabesignalen ein einziges Ausgabesignal liefert. Die Kanten der Struktur sind nicht gewichtet, d. h., die Konfiguration des adaptiven Netzwerks erfolgt durch die parametrisierten Funktionen der jeweiligen Knoten. Durch die Modifikation der Parameter der einzelnen Knoten ändert sich die Funktion und damit auch das gesamte Verhalten des adaptiven Netzes.

In der Abbildung 3.1 ist die komplette Systemarchitektur von ANFIS mit zwei Eingabevariablen und einer Ausgabe aufgeführt.

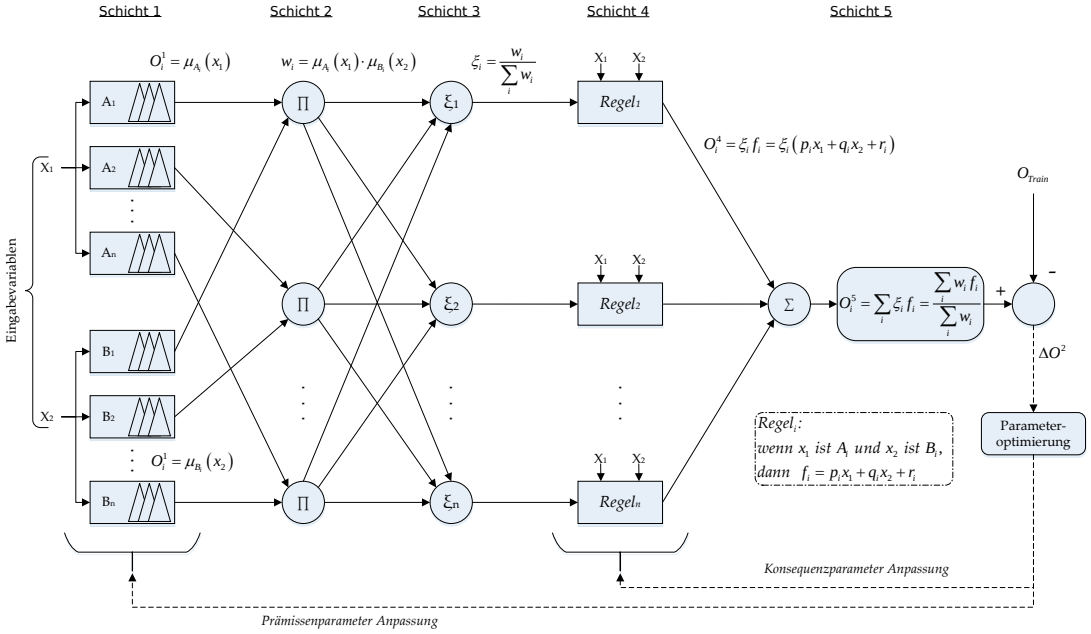

Abbildung 3.1: Architektur von ANFIS mit zwei Eingabewerten (x_1, x_2) und einer Ausgabe. Die gestrichelten Linien zeigen eine Anpassung der Parameter entsprechend der Fehlerrate zwischen der Ausgabe und der Trainingsdaten.

Die rechteckigen Knoten sind adaptive Knoten mit Parameter, die beim Lernen bzw. Trainieren angepasst werden. Runde Knoten sind feste Knoten ohne Parameter.

Die Knotenfunktion[8] erhält die Notation O_i^k . Der Index *i* bezeichnet die Position in der k-ten Schicht einer ANFIS-Architektur. Damit lässt sich die Funktion wie folgt beschreiben.

$$O_i^k : O_i^k \left(O_1^k, \ldots O_{\#(i-1)}^{k-1}, a, b, c, \ldots \right) \tag{3.2}$$

Der Knotenausgang hängt stark von den Eingangswerten der vorherigen Knoten und deren Parameter *a*, *b*, *c* usw. ab. Der Index *#(i-1)* gibt die Anzahl der Knoten der jeweiligen Schicht an.

[8] Im Folgenden wird die Notation für die Knotenfunktion auch als Knotenausgang benutzt.

3.1.1 Schichten von ANFIS

Die gesamte Systemarchitektur besteht aus fünf, nicht homogen aufgebauten Schichten: Fuzzy-Schicht, Regelschicht/T-Norm-Schicht, normierte Schicht, Defuzzifizierungsschicht und der 5. Schicht als Ausgabeschicht (siehe Abbildung 3.1).

Die erste Schicht von ANFIS ist ein adaptiver Knoten. Jede Eingabegröße der Schicht hat seine eigene Zugehörigkeitsfunktion,

$$O_i^1 = \mu_{A_i}(x_1) \tag{3.3}$$

$$O_i^1 = \mu_{B_i}(x_2) \tag{3.4}$$

die üblicherweise durch eine gaußsche Glockenkurve dargestellt wird. Die Indizes A_i und B_i können als linguistische Variablen definiert werden. Mögliche Ausprägungen der Variablen sind etwa linguistische Terme wie beispielsweise „klein“, „mittel“, „groß“, „sehr groß“ und bieten damit die Möglichkeit, Expertenwissen einzubringen, wodurch die Entwicklung der Regelbasis stark vereinfacht wird. Es ist wichtig, die richtige Wahl der Form und die Anzahl einer Zugehörigkeitsfunktion zu bestimmen, da es einen wesentlichen Einfluss auf das Verhalten des adaptiven Netzes hat. Letzten Endes bestimmt der Modellierungserfolg über die Wahl einer konkreten Funktion und ist dadurch eine Design-Frage. In der Literatur u. a. wird daher geraten, von einer Form auszugehen, die man für passend erachtet und dann zu variieren, bis das System akzeptabel ist [11].

In der Gleichung (3.5) und (3.6) ist eine generalisierte gaußsche Glockenkurve aufgeführt, die in MATLAB die Bezeichnung „gbellmf“ trägt (siehe Abbildung 3.2).

$$\mu_{A_i}(x_1) = \frac{1}{1+\left[\left(\frac{x_1 - c_i}{a_i}\right)^2\right]^{b_i}} \tag{3.5}$$

$$\mu_{B_i}(x_2) = \frac{1}{1+\left[\left(\frac{x_2 - d_i}{e_i}\right)^2\right]^{f_i}} \tag{3.6}$$

Die Parameter der Funktionen [a_i, b_i, c_i] und [d_i, e_i, f_i] der ersten Schicht werden als Prämissenparameter bezeichnet. Die Veränderung der Parameter führt zu unterschiedlich geformten Gaußfunktionen und somit zu einem andersgearteten Systemmodell. MATLAB bietet unterschiedliche Typen von Zugehörigkeitsfunktionen an, die in Abbildung 3.2 dargestellt sind.

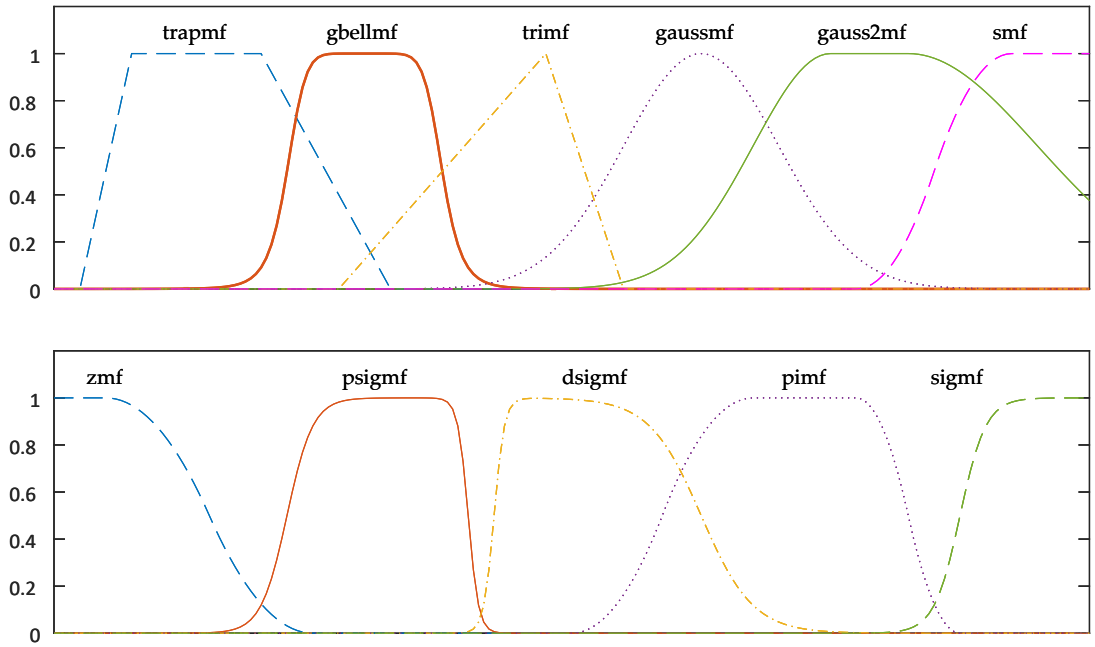

Abbildung 3.2: Verfügbare Zugehörigkeitsfunktionen (MF) in der MATLAB Fuzzy-Toolbox [12]

Die zweite Schicht besteht aus einem festen Knoten, die durch eine beliebige T-Norm (UND Operatoren) gebildet werden kann. Diese Schicht ist eine simple Operation zwischen Eingabewerten der Ergebnisse von den Zugehörigkeitsfunktionen. In der Abbildung 3.3 werden zwei mögliche T-Norm Operatoren für eine ANFIS-Architektur aufgezeigt.

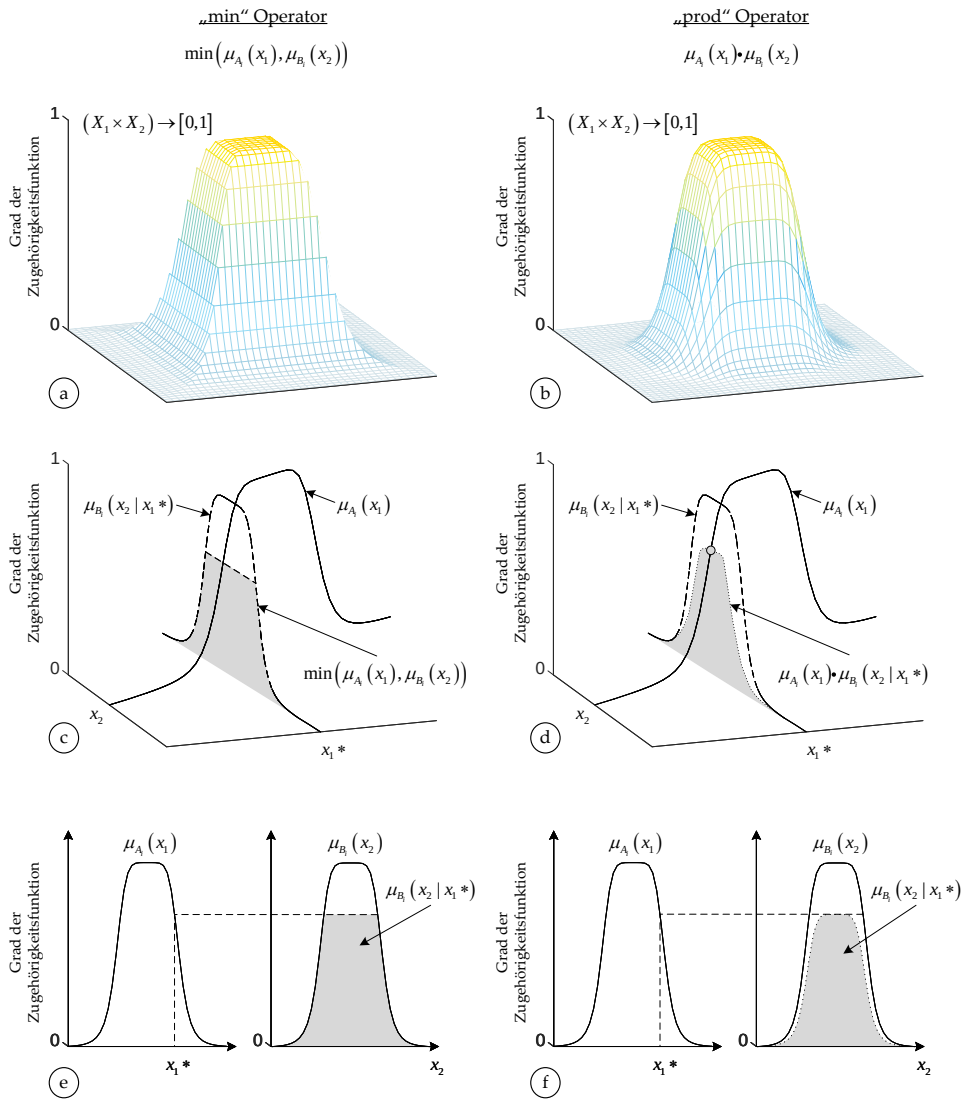

Abbildung 3.3: Drei unterschiedliche Darstellungen der Fuzzy-Implikationen für zwei weitverbreitete Operatoren.

Die Abbildung 3.3 zeigt drei mögliche Darstellungen einer Fuzzy-Implikation für den Minimum-Operator (min) und für das algebraische Produkt (prod) von Fuzzy-Mengen. Abschnitt 3.3 (a) und (b) bilden die gesamte Implikation durch die Operatoren *min* und *prod* über den kompletten Definitionsbereich X und Y. Abschnitt 3.3 (c) und (d) zeigen eine eindimensionale Schnittfläche der Implikation für $x = x^*$, d. h., es wird nicht der komplette Definitionsbereich $R: (X \times Y)$ aufgeführt, sondern nur ein Ausschnitt des momentanen

Wertes x^*. Die beiden letzten Abschnitte (e) und (f) veranschaulichen dieselbe eindimensionale Implikation, wie die zwei vorherigen Darstellungen, durch zwei unabhängige Graphen für x_1 und x_2. Die schraffierten Flächen bilden die resultierenden Zugehörigkeitsfunktionen, hervorgerufen durch die Implikation.

In der Abbildung 3.1 ist der Knoten der 2. Schicht durch ein Produkt repräsentiert. Infolgedessen werden die Eingangsgrößen miteinander multipliziert. Der Ausgang der Knoten ist das Produkt aller Eingangsgrößen und jede Ausgangsgröße steht für eine Feuerungsstärke einer Regel. Die Feuerungsstärke wird durch die Gleichung (3.7) ausgedrückt.

$$O_i^2 = w_i = \mu_{Ai}(x_1) \bullet \mu_{Bi}(x_2) \qquad \text{mit } i = 1, 2, ..., 10. \tag{3.7}$$

In der dritten Schicht werden die Feuerungsstärken der vorherigen Knoten normiert. Der i-te Knoten berechnet den Anteil der i-ten Regel an der Feuerungsstärke. Diese Schicht wird als normierte Schicht bezeichnet und die Ausgänge der Schicht werden auch als normierte Feuerungsstärke genannt. Genauso wie in der zweiten Schicht, ist jeder Knoten dieser Schicht ein fester Knoten. Die normierte Funktion für die dritte Schicht ist durch die Gleichung (3.8) beschrieben.

$$O_i^3 = \xi_i = \frac{w_i}{\sum_i w_i} \qquad \text{mit } i = 1, 2, ..., 10. \tag{3.8}$$

Die vierte Schicht ist ein adaptiver Knoten, der die normierte Feuerungsstärke ξ_i mit der Konklusion der zugehörigen, parametrisierten Regel f_i multipliziert. Diese Regel stammt vom Takagi-Sugeno-Modell ab. Die Parameter [p_i, q_i, r_i] dieser Schicht werden als Konklusionen- bzw. Konsequenzparameter bezeichnet. Die Takagi-Sugeno-Regelbasis ist in den Punkten (3.9) bis (3.11) aufgeführt.

$$\begin{aligned} &\textit{Regel 1 : Wenn } x_1 \textit{ ist } A_1 \textit{ und } x_2 \textit{ ist } B_{1,} \\ &\textit{dann } f_1 = p_1 x_1 + q_1 x_2 + r_1 \end{aligned} \tag{3.9}$$

$$\begin{aligned} &\textit{Regel 2 : Wenn } x_1 \textit{ ist } A_2 \textit{ und } x_2 \textit{ ist } B_{2,} \\ &\textit{dann } f_2 = p_2 x_1 + q_2 x_2 + r_2 \end{aligned} \tag{3.10}$$

...

$$\begin{aligned} &Regel\ 10: Wenn\ x_1\ ist\ A_{10}\ und\ x_2\ ist\ B_{10,} \\ &dann\ f_{10} = p_{10}x_1 + q_{10}x_2 + r_{10} \end{aligned} \tag{3.11}$$

Die vierte Schicht ist durch die Gleichung

$$O_i^4 = \xi_i f_i = \xi_i \left(p_i x_1 + q_i x_2 + r_i \right) \tag{3.12}$$

beschrieben.

Die fünfte und letzte Schicht der ANFIS-Struktur summiert alle eingegangenen Größen. Die Schicht besteht aus einem festen Knoten. Der Ausgang des Knoten *y* ist aus der Gleichung (3.13) zu entnehmen.

$$y = O_i^5 = \sum_i \xi_i f_i = \frac{\sum_i w_i f_i}{\sum_i w_i} \tag{3.13}$$

3.1.2 Variation der ANFIS-Architektur

Eine Erweiterung der ANFIS-Architektur bildet das CANFIS [13]. Im Vergleich zum ANFIS kann die Architektur mehrere Ausgaben mit nicht-linearen Fuzzy-Regeln besitzen. Ein weiterer Ansatz für ein MIMO-System ist MANFIS [14]. Hierbei werden so viele ANFIS-Netze nebeneinander aufgestellt, wie Eingänge erforderlich sind. Der Nachteil des Ansatzes ist es, dass die modifizierbaren ANFIS-Parameter nicht untereinander geteilt werden und die Regeln unabhängig voneinander sind. Beim CANFIS hingegen werden die Prämissenparameter untereinander geteilt. Eine andere Architektur mit mehreren Ausgängen sind von den Autoren Tomar, Qureshi et al. [15] vorgestellt. Das MIMO-ANFIS unterscheidet sich von der ANFIS-Architektur in der vierten Schicht mit zusätzlichem Konklusionsparameter für die Ausgänge.

Eine Rekurrente ANFIS-Architektur in Kombination mit einem lokalen Suchalgorithmus sind von den Autoren Tamura, Tanno et al. [16] publiziert. Dabei besteht das Rekurrente ANFIS-Netz aus zwei Rückkopplungen: Ausgangs- und Fehlerrückkopplung. Die Ausgangsrückkopplung wird als zusätzlicher Eingang ins ANFIS-System zurückgeführt und die Fehlerrückkopplung nach dem Fuzzifizierungsprozess in die vierte Schicht umgeleitet.

3.2 Lernalgorithmus von ANFIS

In ANFIS ist der Lernalgorithmus vorwiegend[9] zur Anpassung der Parameter der adaptiven Knoten ausgelegt. Der Algorithmus bestimmt die Parameter durch die Minimierung der Fehlerrate zwischen dem ANFIS-Ausgang und der erwarteten Ausgabe.

Nach der Veröffentlichung von ANFIS durch Jang [7] wurden Ansätze zur Optimierung der Parameter vorgestellt. In diesem Zusammenhang haben Jang und Mizutani [19] eine Anwendung mit dem Levenberg-Marquardt-Algorithmus vorgelegt. Hierbei handelt es sich um ein Verfahren zum Lösen nicht-linearer Ausgleichs-Probleme mithilfe der Methode der kleinsten Quadrate (KQ-Methode/LSE). In einer anderen wissenschaftlichen Publikation haben Frattale Mascioli, Varazi et al. [20] vorgeschlagen das ANFIS-Modell und MIN-MAX-Algorithmus zusammenzuführen, um das optimale Neuro-Fuzzy-Netzwerk zu bestimmen. Loganathan und Girija [21] benutzen das Runge-Kutta-Verfahren zum Trainieren der Parameter.

Die meisten Implementierungsansätze haben die Partikelschwarmoptimierung (PSO) in Kombination mit anderen Algorithmen zum Trainieren angewandt. In diesem Kontext existieren einige Verfahren, die das PSO mit LSE kombinieren, um die Prämissenparameter durch PSO und die Konsequenzparameter durch das LSE-Verfahren zu optimieren [22–24]. Die PSO in Kombination mit GD wurde durch Shoorehdeli et al. [25] vorgestellt. Diese Verfahren legen den Fokus auf Parameteroptimierung und trainieren nicht die Fuzzy-Regelbasis. Lernverfahren, die ausschließlich den PSO-Algorithmus benutzen, um sowohl Prämissen- als auch Konsequenzparameter zu trainieren, wurden von Rini et al. [17] und Turki et al. [26] eingesetzt.

Variationen des PSO sind vielseitig in der Literatur vorzufinden. Shoorehdeli et al. [27] präsentieren ein Verfahren, das PSO mit einer rekursiven KQ-Methode (RLS) vereint. Eine weitere Form des PSO-RLS bieten Shooredeli et al. [28] in ihrer Arbeit mit dem Algorithmus Adaptiv-Weighted-PSO (AWPSO) in Kombination mit Forgetting-Factor-RLS (FFRLS). Ebenfalls von Shoorehdeli et al. [29] stammt die Implementation mit einer erweiterten Variation des LSE. Hierbei benutzen die Autoren einen erweiterten Kalman-Filter (EKF) mit dem AWPSO-

[9] Die Optimierung der TSK-Fuzzy-Regelbasis ausgenommen; wie bspw. durch die Begrenzung der Werte der Feuerungsstärke [17] oder Minimierung der redundanten Regeln [18].

Verfahren. Weitere Variationen wie das QPSO werden von Bagheri et al. [30] verwendet, um die Prämissenparameter anzupassen.

Einen ebenfalls großen Anteil der Lernalgorithmen in ANFIS bildet das GA-Verfahren, das in die Kategorie Evolutionären Algorithmen (EF) fällt. Eine Kombination mit LSE veröffentlichten Soto et al. [31] in ihrer Arbeit über die Vorhersage von Mackey-Glass-Zeitreihenanalyse. Bagheri et al. [32] benutzen die Singulärwertzerlegung (SWZ), um die Konklusion zu bestimmen. Cárdenas et al. [33], Lutfy et al. [34] und Haznedar et al. [35] verwenden GA zum Trainieren der Parameter der einzelnen Zugehörigkeitsfunktionen als auch der linearen Koeffizienten der TSK-Regelbasis.

Weitere Lernalgorithmen zum Trainieren von ANFIS-Strukturen wurden erforscht und veröffentlicht. Dazu gehören Resilient-Backpropagation (Rprop), Quickprop (QP) und Levenberg-Marquardt (LM) [36]. Andere Algorithmen, die auf metaheuristische Verfahren basieren, sind Firefly-Algorithmus (FA) [37], Differential Evolution mit dem Ameisenalgorithmus (DEACS) [18], Cat Swarm Optimization (CSO) [38] und Artificial Bee Colony (ABC) [39].

Eine Übersicht der Algorithmen zur Anpassung der ANFIS-Parameter ist in der Tabelle 3.1 zu finden.

Tabelle 3.1: Lernansätze zum Trainieren der ANFIS-Parameter [40] (Schicht 1 und 4)

ALG.	Forschungsarbeit	Prämissenparameter	Konsequenzparameter
PSO	Catalao et al. [22]	PSO	LSE
	Jiang, Kwong et al. [24]	PSO	LSE
	Pousinho et al. [23]	PSO	LSE
	Shoorehdeli et al. [25]	PSO	GD
	Rini et al. [17]	PSO	PSO
	Turki et al. [26]	PSO	PSO
	Shoorehdeli et al. [27]	PSO	RLS
	Shoorehdeli et al. [28]	AWPSO	FFRLS
	Shoorehdeli et al. [29]	AWPSO	EKF
	Bagheri et al. [30]	QPSO	LSE
GA	Soto et al. [31]	GA	LSE
	Bagheri et al. [32]	GA	SWZ
	Cárdenas et al. [33]	GA	GA

	Lutfy et al. [34]	GA	GA
	Haznedar; Kalinli [35]	GA	GA
Sonstige	Karaboga et al. [39]	ABC	ABC
	Orouskhani et al. [38]	CAT	GD
	Wang et al. [18]	DEACS	DEACS
	Kamarian et al. [37]	FA	FA
	Frattale Mascioli et al. [20]	MIN-MAX	GD
	Mu-Song Chen [19; 36]	Rprop/QP/LM	RLS

Jang [7] nennt in seiner Publikation vier Methoden, um die Parameter eines ANFIS-Netzes zu aktualisieren:

1. Nur Gradientenverfahren: Alle Parameter werden durch das Gradientenverfahren aktualisiert.
2. Gradientenverfahren und LSE (ein Durchlauf): Das LSE-Verfahren wird nur zu Beginn einmal eingesetzt, um die Anfangswerte von den Konsequenzparametern zu bestimmen. Im Anschluss übernimmt das Gradientenverfahren die Aktualisierung aller Parameter.
3. Gradientenverfahren und LSE: Diese Methode ist der hybride Lernalgorithmus. Die Vorwärtsrechnung übernimmt das LSE und bei der Rückwärtsrechnung wird das Gradientenverfahren angewandt.
4. Sequenzielles LSE: Benutzt einen erweiterten Kalman-Filter für das Aktualisieren der Parameter.

Der Lernalgorithmus, der für die Implementierung von ANFIS benutzt wird, ist ein hybrider Lernalgorithmus von Jang [7]. Der größte Nutzen eines solchen hybriden Ansatzes ist die schnellere Konvergenzgeschwindigkeit. Da hierbei die Dimension des Suchraums beim Rückpropagieren wesentlich reduziert wird [41].

Der hybride Algorithmus setzt sich aus der Vorwärtsrechnung und der Rückwärtsrechnung zusammen. In der Vorwärtsrechnung werden die Parameter der Konklusionen [p_i, q_i, r_i], d. h. die Koeffizienten der linearen Ausgabefunktionen f_i aus der Takagi-Sugeno-Regelbasis, durch die Methode der kleinsten Quadrate reguliert (siehe Abbildung 3.4).

Bei der Rückwärtsrechnung propagiert die Fehlerrate rückwärts und die Parameter der Prämisse [a_i, b_i, c_i, d_i, e_i, f_i] werden durch das Gradientenverfahren angepasst.

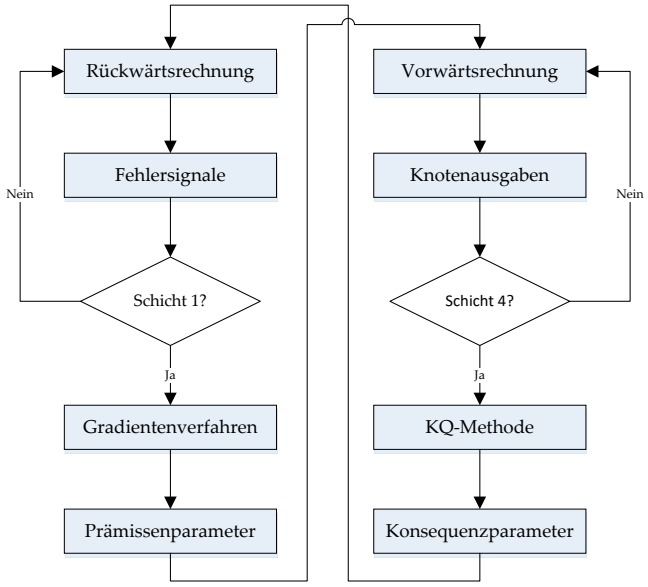

Abbildung 3.4: Flussdiagramm des hybriden Lernalgorithmus zur Bestimmung der ANFIS-Parameter

3.2.1 Off-line-Lernen

Die Methode des Off-line-Lernens beinhaltet beide Algorithmen (Gradientenverfahren und Methode der kleinsten Quadrate), um die Parameter der ANFIS-Struktur zu bestimmen. Dadurch wird das Lernen verbessert[10], statt nur das Gradientenverfahren anzuwenden [7]. Um die Basis der Methode erklären zu können, wird zuerst die Beziehung zwischen der Eingabe und der Ausgabe betrachtet. Das adaptive Netz besitzt nur eine Ausgabe, wobei I die Menge der Eingabewerte bildet und S die Menge der Parameter.

$$\boldsymbol{Ausgabe} = F\left(I, S\right) \tag{3.14}$$

Es wird angenommen, dass eine Funktion H existiert, die die Komposition $H \circ F$ mit einigen Elementen von S linear abbildet. Dann ist es möglich, die KQ-Methode zur Identifizierung der

[10] Nachteil des GD-Verfahren sind die langsame Konvergenzgeschwindigkeit und das Hängenbleiben im lokalen Minimum [7; 41].

Parameter anzuwenden. D. h. wenn gezeigt werden kann, dass die Menge der Parameter S in zwei Mengen zerlegen werden[11],

$$S = S_1 \oplus S_2 \tag{3.15}$$

mit den Elementen,

$$S = Menge\ der\ gesamten\ Parameter$$
$$S_1 = Menge\ der\ Prämissenparameter$$
$$S_2 = Menge\ der\ Konsequenzparameter$$

sodass die Komposition $H \circ F$ mit den Elementen S_2 linear ist. Jetzt ist es möglich, die Funktion H zu der Gleichung (3.14) anzuwenden. Dadurch resultiert die Gleichung (3.16).

$$H(\boldsymbol{Ausgabe}) = H \circ F\left(I, S\right) \tag{3.16}$$

Es sei angenommen, dass die Trainingsdaten P gegeben sind, welche mit der Gleichung (3.16) verbunden werden. Daraus wird eine allgemeine Gleichung (3.17) erhalten.

$$A\,\boldsymbol{X} = B \tag{3.17}$$

In der allgemeinen Gleichung bildet $\boldsymbol{X}$ den Vektor mit den Elementen der unbekannten Parameter in S_2. Wenn $|S_2| = M$ ist, dann hat A die Dimension $P \times M$, X die Dimension $M \times 1$ und B die Dimension $P \times 1$. Da gewöhnlich die Anzahl der Trainingsdaten P größer ist als die linearen Parameter M, besteht die Problematik der Überbestimmung und damit gibt es keine akkurate Lösung für die Gleichung (3.17). Anstatt eine genaue Lösung zu finden, ist es möglich, eine Pseudo-Lösung anzubieten. Hierbei hilft die Methode der kleinsten Quadrate bei der Lösungsfindung. Die folgende Gleichung (3.18) liefert die Lösung des Problems.

$$X^* = (A^T A)^{-1} A^T B \tag{3.18}$$

[11] Das Zeichen $\oplus$ entspricht der direkten Summe; Das Zeichen $\circ$ ist die Verkettung von Funktionen.

In dieser Gleichung ist der Term $(A^TA)^{-1}A^T$ die Pseudo-Inverse von A. Die Lösung für die Gleichung ist wegen seiner Inverse relativ rechenintensiv. Um das Problem zu lösen, wird ein sequenzieller Ansatz (LSE) zur Berechnung benutzt.

$$\begin{aligned} X_{i+1} &= X_i + S_{i+1}a_{i+1}\left(\boldsymbol{b}_{i+1}^T - \boldsymbol{a}_{i+1}^T X_i\right) \\ S_{i+1} &= S_i - \frac{S_i a_{i+1} + \boldsymbol{a}_{i+1}^T S_i}{1 + \boldsymbol{a}_{i+1}^T S_i a_{i+1}} \end{aligned} \tag{3.19}$$

Dabei ist S_i die Kovarianzmatrix und der Index i hat die Wertemenge zwischen 0 und $P\text{-}1$. Die Startkondition der Formel sind $X_0 = 0$ und $S_0 = \gamma I$, wobei γ eine große positive Zahl ist und I ist die Einheitsmatrix der Dimension $M \times M$. An dieser Stelle können das Gradientenverfahren und die Methode der kleinsten Quadrate kombiniert werden, um die Parameter mit dem gewonnenen hybriden Lernverfahren anzupassen. Ein Trainingszyklus (eine Epoche) des hybriden Lernalgorithmus besteht aus der Vorwärtsrechnung und der Rückwärtsrechnung (Tabelle 3.2).

Tabelle 3.2: Rechnungsdurchlauf des hybriden Lernalgorithmus [41]

	Vorwärtsrechnung	**Rückwärtsrechnung**
Prämissenparameter	Fest	Gradientenverfahren
Konsequenzparameter	KQ-Methode	Fest
Signale	Knotenausgaben	Fehlersignale

Bei der Vorwärtsrechnung werden die Eingangsdaten ins System initiiert und jeder Ausgangsknoten des Systems berechnet. Dies geschieht, bis die Matrizen A und B aus der Matrixgleichung (3.17) ausgerechnet und die Konsequenzparameter S_2 mit der KQ-Methode (3.19) identifiziert sind. Die Vorwärtsrechnung dauert solange, bis das Fehlermaß

$$E_p = \sum_{m=1}^{\#(L)} \left(T_{m,p} - O_{m,p}^L\right)^2 \tag{3.20}$$

erreicht wird. Hierbei ist $\#(L)$ die Anzahl der Schichten im ANFIS. Die Ausgabe wird mit den Trainingsdaten verglichen, um das Fehlermaß zu berechnen. Das Element $T_{m,p}$ beschreibt die Trainingsdaten der m-ten Komponente von dem p-ten Ausgangsvektor und das Element $O_{m,p}$

ist die momentane Ausgabe der m-ten Komponente von dem p-ten Vektor. Durch das Summieren der Fehlermaße für jedes Trainingsdaten p wird das Gesamtfehlermaß E aus der Gleichung (3.21) erhalten.

$$E = \sum_{p=1}^{P} E_p \tag{3.21}$$

Bei der Rückwärtsrechnung propagiert die Fehlerrate

$$\frac{\partial E_p}{\partial O_{i,p}^L} = -2\left(T_{i,p} - O_{i,p}^L\right) \tag{3.22}$$

rückwärts. Jeder interne i-te Knoten der k-ten Schicht hat seine eigene Fehlerrate, die durch das Anwenden der Kettenregel wie folgt lautet

$$\frac{\partial E_p}{\partial O_{i,p}^k} = \sum_{m=1}^{\#(k+1)} \frac{\partial E_p \partial O_{m,p}^{k+1}}{\partial O_{m,p}^{k+1} \partial O_{i,p}^k} \ . \tag{3.23}$$

Mit den Formeln aus (3.22) und (3.23) kann jede Fehlerrate für alle Knoten berechnet werden.

Die Prämissenparameter werden durch das Gradientenverfahren angepasst. Um dies zu beschreiben, wird das Symbol α für die Parameter des adaptiven Netzes eingeführt. Der Ausgang ist abhängig von dem Parameter α. Im Folgenden ist S die Menge der Knoten, die von dem Parameter α abhängig sind.

$$\frac{\partial E_p}{\partial \alpha} = \sum_{O^* \in S} \frac{\partial E_p \partial O^*}{\partial O^* \partial \alpha} \tag{3.24}$$

Die Formel für die Parameter α kann durch die Gleichung (3.25) des Gradientenverfahrens aktualisiert werden.

$$\triangle \alpha = -\eta \frac{\partial E}{\partial \alpha} \tag{3.25}$$

In der obigen Formel gibt η die Lerngeschwindigkeit an. Die Geschwindigkeit kann durch die Schrittgröße k, wie aus der Formel (3.26) zu entnehmen, manipuliert werden.

$$\eta = \frac{k}{\sqrt{\sum_{\alpha}\left(\frac{\partial E}{\partial \alpha}\right)^2}} \tag{3.26}$$

Die Konvergenzgeschwindigkeit hängt ebenfalls von der Länge jeder Gradienten-Transition im Parameterraum ab [7].

3.2.2 On-line-Lernen

On-line-Lernen ist für Systeme geeignet, die anfänglich nicht alle Eingabedaten zur Verfügung haben. D. h., die Daten sind zu Beginn des Systems nicht vorhanden und werden zur Ausführung ergänzt. Um diese zu gewährleisten, muss der Algorithmus aus der Formel (3.19) entsprechend angepasst werden [7]. Dieser Algorithmus berücksichtigt veraltete Parameter mit der Einführung des Vergessensfaktors λ.

$$\begin{aligned} X_{i+1} &= X_i + S_{i+1} a_{i+1}\left(\boldsymbol{b}_{i+1}^T - \boldsymbol{a}_{i+1}^T X_i\right) \\ S_{i+1} &= \frac{1}{\lambda}\left[S_i - \frac{S_i a_{i+1} + \boldsymbol{a}_{i+1}^T S_i}{\lambda + \boldsymbol{a}_{i+1}^T S_i a_{i+1}}\right] \end{aligned} \tag{3.27}$$

Der Lambdawert λ liegt zwischen 0 und 1. Ein kleiner Lambdawert bedeutet ein schnelles Veralten der Daten. Jedoch kann dies eine Instabilität des Systems hervorbringen und sollte deshalb vermieden werden [7].

3.3 Implementation von ANFIS

Das ANFIS-Modell wurde in unterschiedliche Programmiersprachen umgesetzt. Der Autor Jang selbst hat eine Implementierung als C-Code mit vier Simulationsbeispielen veröffentlicht [42]. Der C-Code und die Beispiele basieren auf seiner Originalpublikation über ANFIS [7]. Das Training der ANFIS-Parameter läuft über den hybriden Lernansatz. Die Zugehörigkeitsfunktion ist fest implementiert und entspricht der Gleichung (3.5).

Eine Implementierung für die R-Bibliothek haben die Autoren Fresno und Fernandez veröffentlicht [43]. Die Bibliothek ermöglicht die freie Wahl von Zugehörigkeitsfunktionen, sowie Bestimmung der Anzahl von Funktionen für jeden Eingang. Die Regeln der ANFIS-Struktur können ebenfalls in jeglicher Kombination definiert werden. Zum Lernen der ANFIS-Parameter kann der Anwender zwischen hybriden On-line- oder Off-line-Verfahren entscheiden. Die Implementierung bietet zusätzlich die Möglichkeit, die Ergebnisse grafisch zu veranschaulichen. Die Besonderheit der Bibliothek liegt darin, dass es mehrere Ausgaben ermöglicht.

Eine Umsetzung in der Programmiersprache Python bietet der Autor Meggs an [44]. Als Grundlage für die Implementierung in Python dient die R-Bibliothek (Version 1.01). In seiner Version 0.3.0 ist die Implementierung noch in der Beta-Phase und ist bedingt nutzbar. So kann der Nutzer zwischen drei unterschiedlichen Zugehörigkeitsfunktionen wählen und die Eingänge auch mit beliebigen Funktionen belegen. Eine grafische Darstellung ist ebenfalls möglich.

Weitere Implementierungen sind in den Programmiersprachen C# und Java umgesetzt [45; 46]. Die Umsetzung für C# ermöglicht drei Zugehörigkeitsfunktionen und fünf verschiedene Lerntechniken, wie Backpropagation, QPROP etc. Dabei ist der hybride Lernalgorithmus nicht inbegriffen. Die Java-Implementierung ist in seiner Funktion schlicht gehalten und ebenfalls begrenzt nutzbar. Als Zugehörigkeitsfunktion ist die Sigmoidfunktion implementiert.

Die Anwendung MATLAB bietet in seiner Fuzzy-Logic-Toolbox eine Schnittstelle für ANFIS. Der Anwender kann entweder über die Kommandozeile oder die GUI-Schnittstelle ein System modellieren. Als Optimierungsmethoden für die ANFIS-Parameter dienen der hybride Lernalgorithmus oder die Backpropagation. Zusätzlich stehen dem Nutzer, eine Vielzahl an Zugehörigkeitsfunktionen, die in der Abbildung 3.2 aufgeführt sind, zur Verfügung. Die Anzahl von Funktionen für jeden Eingang können je nach Wunsch definiert werden. Wie bei vielen MATLAB-Anwendungen ist ANFIS[12] als eigenständige Anwendung kompilierbar und auch als Bibliothek für andere Programmiersprachen wie C/C++, Java, Python etc. anwendbar.

In der vorliegenden Arbeit soll MATLAB R2016b als Arbeitsumgebung fungieren. Die zur Verwendung kommende Fuzzy-Logic-Toolbox wird im Kapitel 4 eingeführt.

[12] ab der MATLAB-Version R2016b

4 ANFIS zur Diagnose

In diesem Teil der Arbeit wird die Diagnose durch neuronale Netze näher betrachtet. Der Schwerpunkt liegt dabei auf der Nutzung des Adaptiven Neuro-Fuzzy-Inferenzsystem. Zu Beginn wird dafür eine Einführung in die Einsatzgebiete von Diagnosen mit ANFIS gegeben. Es ist wichtig zu erwähnen, dass die Diagnose im Zuge des medizinischen Rahmens durchgeführt wird. Trotz dessen ist die Anwendung auf andere Diagnose-Begrifflichkeiten übertragbar. Im zweiten Abschnitt wird die Fuzzy-Logic-Toolbox, die in dieser Arbeit verwendet wird, thematisiert. Die Wahl des Eingaberaumes für eine ANFIS-Struktur wird im Anschluss erklärt.

4.1 Medizinische Diagnose in der Literatur

In der Literatur sind viele Diagnoseansätze mithilfe von KNN für diverse Zwecke publiziert worden. So besteht eine Vielzahl an medizinischen Diagnoseanwendungen für ANFIS, die ausgesuchte Merkmale einer Krankheit klassifizieren. Ziel ist es, eine Zuordnung für zwei oder wenige Klassen zu treffen. Die Merkmale einer Diagnose können auf unterschiedliche Mittel gewonnen werden. Hierbei kann das Wissen auf Beobachtungen bzw. Symptome (körperliche Untersuchung), Proben oder medizinische Aufnahmen (apparative Untersuchungen) des Patienten basieren.

Sridevi und Nirmala [47] veröffentlichten eine Methode, um eine Herzfehlbildung (Truncus arteriosus communis) mit ANFIS als Klassifikator zu diagnostizieren. Die Merkmale werden aus 2-D-Bildern mit einem PPB-Filter extrahiert und im Anschluss die Parameter mit der Fisher’sche Diskriminanzfunktion selektiert. Eine weitere Methode zur Bestimmung, ob eine

Herzkrankheit vorliegt oder nicht, haben die Autoren Abushariah, Alqudah et al. [48] und die Autorin Amma [49] veröffentlicht. Die Merkmale basieren auf einer aufbereiteten Datenbasis der University of California mit 13 Attributen. Diese werden durch die Methode der Singulärwertzerlegung (SWZ) verringert.

Einen großen Bereich der Anwendung findet in Krebsdiagnosen. Obayya und Areed [50] publizierten einen Ansatz zur Diagnose von Leberkrebs. Die Daten stammen von Bildern aus Computertomografien und sind mit der morphologischen Bildverarbeitung und dem Schwellenwertverfahren aufbereitet. Der Tumor ist durch den FCM-Algorithmus extrahiert und die Merkmale mit der diskreten Wavelet-Transformation ermittelt. Eine Arbeit zur Ermittlung von Hautkrebs findet sich in der Arbeit von Odeh [51]. Dabei werden alle Eigenschaften von Aufnahmen extrahiert und der Parameterraum durch den Algorithmus G-flip verkleinert. Weiterhin sind Diagnosen für Brustkrebs publiziert [52; 53]. Hierbei benutzen die Autoren Bilder von Mammografien. Aus den vorbearbeiteten Daten werden die Eigenschaften (Entropie, Energie, Homogenität) der Bilder selektiert. Ein Diagnosesystem für Darmkrebs wurde von den Autoren Lim, Maguib et al. [54] vorgestellt. Die Bilder sind von einer virtuellen Koloskopie entnommen und durch die Grauwertematrix die Eigenschaften ermittelt. Für die Selektion der Eigenschaften haben die Autoren einen Kohonennetz mit Genetischem-Algorithmus (GA-SOM) genutzt und mit ANFIS, die Klassifikation durchgeführt. Loganathan und Girija [55] behandeln in ihrer Arbeit den Lymphdrüsenkrebs und die Leukämie. Eine Kombination aus ANFIS und Runge-Kutta-Verfahren dient als Klassifikator. Publikationen zu Diagnosen von Gehirntumoren auf Basis von MRT-Bildern sind ebenfalls in der Literatur vorzufinden [56; 57]. Andererseits wird in einer anderen Veröffentlichung ein ANFIS-System vorgestellt, die ausschließlich auf Symptome des Patienten aufbaut [58].

Andere Diagnosegebiet wie Alzheimer werden von den Autoren Al-Naami, Mallouh et al. [59] vorgestellt. Die Datenbasis setzt sich aus MRT-Bildern von Alzheimerpatienten und von gesunden Patienten. Die Merkmale werden aus dem vorbearbeiteten Bildern mit der diskreten Wavelet-Transformation extrahiert.

Weiterhin besteht eine Vielzahl von ANFIS-Systemen, die nicht auf bildgebende Diagnostik basiert. So benutzen die Autoren Ansari, Gupta et al. [60] ANFIS für die Diagnose von Asthma. Übeyli und Karahoca [61; 62] präsentieren in ihren Publikationen ein Verfahren, um Diabetes zu diagnostizieren. Eine Variation mit Genetischem-Algorithmus und ANFIS zur Bestimmung von Hepatitis ist in der Arbeit von Adeli, Bigdeli et al. [63] aufgeführt. Andere Diagnosegebiete sind Malaria [64], Influenza [65] oder auch Sprach- [66] und Schlafstörung [67].

4.2 MATLAB Fuzzy-Logic-Toolbox

Die Fuzzy-Logic-Toolbox bietet eine Vielzahl an Möglichkeiten, um ein FIS zu gestalten und in unterschiedliche Systeme auf Basis von Fuzzy-Logik einzubinden. Zu den Hauptmerkmalen gehören die

- Erstellung von Mamdani und Sugeno-Fuzzy-Regler,
- Erzeugung von Zugehörigkeitsfunktionen und
- die Adaptierung der Funktionen durch neuro-adaptive und Fuzzy-Clustering-Lerntechniken.

In MATLAB ist eine Modellierung von Systemen durch Eingabe von Kommandozeilen oder mit grafischer Oberfläche möglich. Hierbei wird für die Arbeit die MATLAB-Version R2016b (9.1.0.441655) 64-Bit mit dem Betriebssystem Microsoft Windows 10 Professionell genutzt. Die Fuzzy-Logic-Toolbox hat die Version 2.2.24. In folgenden Unterpunkten erfolgt eine Einführung in die Struktur des Tools.

4.2.1 FIS-Editor

Der FIS-Editor besteht aus fünf zusammenhängenden GUI-Schnittstellen, welche in der Abbildung 4.1 dargestellt sind. Als Beispiel wird eine Demo[13] aus MATLAB zur Beschreibung der einzelnen Funktionen des Editors benutzt.

[13] *„fuzzyLogicDesigner tipper“ – angepasst auf TSK-System.*

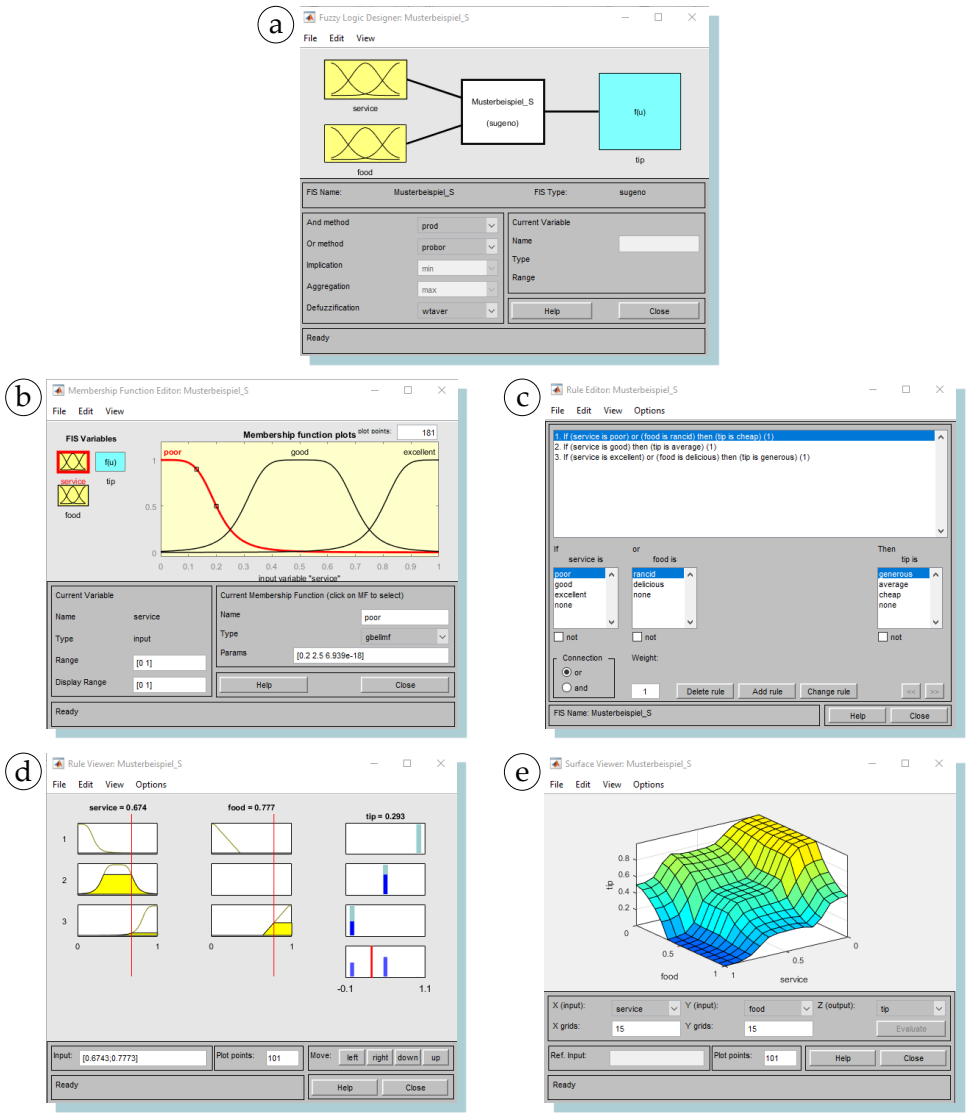

Abbildung 4.1: Grafische Oberfläche des Fuzzy-Logic Designers zum Erstellen/Bearbeiten/Betrachten eines FIS

(a) Definieren des FIS – Bei der Erstellung/Aufruf eines FIS durch die Kommandozeile *fuzzyLogicDesigner* erfolgt die Sicht auf die Grundstruktur des FIS. Im oberen Bereich befindet sich die FIS-Architektur. In der Abbildung 4.1 (a) sind zwei Eingänge mit entsprechenden Zugehörigkeitsfunktionen (links), der Takagi-Sugeno-Regelbasis (mitte) und dem Ausgang (rechts) abgebildet. Im unteren Bereich wird festgelegt, welche Operatoren und Defuzzifizierungsmethode benutzt werden sollen.

(b) Zugehörigkeitsfunktion – Jeder Eingang kann durch eine oder mehrere Zugehörigkeitsfunktionen dargestellt werden. Diese Oberfläche zeigt drei „gbellmf“ (siehe

Abbildung 3.2), die mit linguistischen Termen versehen sind. Ferner sind die Parameter im unteren Bereich aufgelistet. Die ANFIS-Parameter werden im Zuge des Lernvorgangs trainiert.

(c) Regel-Editor – Die Regelbasis kann auf Mamdani oder Takagi-Sugeno-Kang-Logik[14] basieren. Die Beschriftungen der Regeln entsprechen den in Abschnitt (b) definierten linguistischen Termen. Im unteren Bereich können Regeln hinzugefügt, bearbeitet oder nicht erwünschte Regeln entfernt werden. Die Regeln der ANFIS-Struktur werden durch die Partitionierungsalgorithmen generiert.

(d) Regelanzeige – Die Regeln können interaktiv analysiert werden. Die horizontale Linie gibt den Eingabewert an. Die Aktivierung der Zugehörigkeitsfunktionen wird schraffiert angezeigt. Durch Änderung dieser Werte lässt sich in der dritten Spalte das Ergebnis ablesen, um so ein gewisses Verständnis aufzubauen.

(e) Oberflächenbetrachter – Im Gegensatz zu der Regelanzeige kann in diesem Bereich ein gesamtes Bild des Systems angezeigt werden. Hier ist eine dreidimensionale Anzeige der Struktur definiert, da zwei Eingabewerte vorhanden sind. Bei mehr als zwei Eingaben kann die Auswahl zwischen zwei erwünschten Eingaben gewählt werden.

4.2.2 Struktur vom FIS

Im folgenden Listing 4.1 ist eine detaillierte FIS-Struktur mit dem Typ Takagi-Sugeno-Kang aufgelistet. In den ersten zwölf Zeilen ist die Grundstruktur eines FIS definiert. Dabei wird der Name, der Typ, die Anzahl der Ein- und Ausgabe sowie die Inferenzmethoden festgelegt.

```
1  [System]
2          Name = 'Musterbeispiel_S'
3          Type = 'sugeno'
4       Version = 2.0
5     NumInputs = 2
6    NumOutputs = 1
7      NumRules = 3
8     AndMethod = 'prod'
9      OrMethod = 'probor'
10    ImpMethod = 'prod'
11    AggMethod = 'sum'
12 DefuzzMethod = 'wtaver'
13
14 [Input1]
15    Name = 'service'
16   Range = [0 1]
17 NumMFs = 3
18    MF1 = 'poor':'gbellmf',[0.2 2.5 6.939e-18]
```

[14] ANFIS benutzt ausschließlich TSK-Regler [7].

```
19    MF2 = 'good':'gbellmf',[0.2 2.5 0.5]
20    MF3 = 'excellent':'gbellmf',[0.2 2.5 1]
21
22 [Input2]
23   Name = 'food'
24  Range = [0 1]
25 NumMFs = 2
26    MF1 = 'rancid':'trapmf',[-0.36 -0.04 0.04 0.36]
27    MF2 = 'delicious':'trapmf',[0.64 0.96 1.04 1.36]
28
29 [Output1]
30   Name = 'tip'
31  Range = [0 1]
32 NumMFs = 3
33    MF1 = 'generous':'constant',[0]
34    MF2 = 'average':'constant',[0.5]
35    MF3 = 'cheap':'linear',[0 0 1]
36
37 [Rules]
38 1 1, 3 (1) : 2
39 2 0, 2 (1) : 1
40 3 2, 1 (1) : 2
```

Listing 4.1: Komplette Struktur des FIS – Systemaufbau, zwei Eingänge, ein Ausgang und den Regeln

In den Zeilen 14 – 35 werden genauere Angaben über die Eingänge und Ausgang der FIS-Struktur aufgeführt. In diesem Beispiel sind zwei Eingänge und ein Ausgang vorzufinden, die in MATLAB als *input:[1x2 struct]* und *output:[1x1 struct]* definiert sind. Der Aufbau umschließt den Namen, die Grenzen und die Zugehörigkeitsfunktionen mit den entsprechenden linguistischen Termen. Im letzten Abschnitt werden Regeln festgesetzt. Die Nummerierung vor dem Komma gibt den Index[15] der Zugehörigkeitsfunktion von den Eingängen an und die Nummerierung nach dem Komma definiert die Ausgangsfunktion. In den Klammern wird das Gewicht der Regel festgelegt, die hauptsächlich eins ist. Die letzte Ziffer gibt den Operator[16] zwischen den Zugehörigkeitsfunktionen an. Damit lässt sich die erste Regel

$$Regel_1: \textit{Wenn 'service' ist 'poor' oder 'food' ist 'rancid' dann 'tip' ist 'cheap' (1)} \quad (4.1)$$

ableiten.

4.2.3 ANFIS-Editor

In der Abbildung 4.2 ist die grafische Oberfläche des ANFIS-Editors zu sehen, die im Wesentlichen die einzelnen Arbeitsschritte zeigt.

[15] Index startet bei 1; 0 entspricht keine Zugehörigkeitsfunktion

[16] 1 für Und-Verknüpfung, 2 für Oder-Verknüpfung

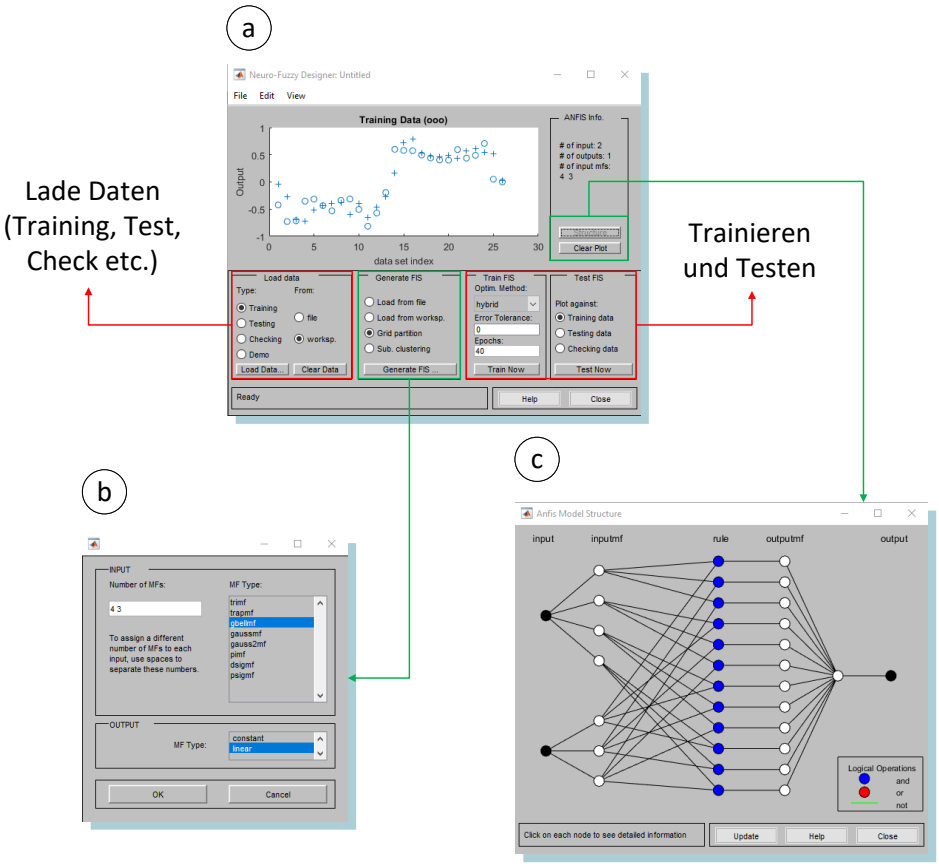

Abbildung 4.2: Grafische Oberfläche des ANFIS-Editors

(a) ANFIS-Editor – Der ANFIS-Editor ist in vier Schritte aufgeteilt: Dem Laden der Daten, die Generierung der ANFIS-Struktur, dem Training und dem Testen der Ergebnisse. Zum Laden der Daten benötigt ANFIS eine Matrix, dessen erste Spalte jeweils die Eingänge und die letzte Spalte den Ausgang repräsentiert. Jede Reihe ist eine Dateneinheit. Der obere Bereich der Oberfläche dient der Visualisierung der Daten.

(b) FIS-Generator/Clustering – Eine FIS-Struktur kann auf unterschiedliche Form generiert werden. Der ANFIS-Editor bietet zwei Clustering-Methoden[17] an: Grid-Partitionierung und Subtractive-Clustering. Hier werden der Typ und die Anzahl der Zugehörigkeitsfunktionen festgelegt.

(c) ANFIS-Strukturbetrachter – Die Grafik zeigt eine gesamte Übersicht der ANFIS-Struktur mit zwei Eingängen und jeweils drei bzw. vier Zugehörigkeitsfunktionen. Aus der Grid-Partitionierung der FIS-Struktur entstehen zwölf Regeln.

[17] Über die Kommandozeile (*genfis3*) ist das Clustering-Verfahren Fuzzy C-Means (FCM) möglich.

4.3 Auswahl der Eingabedaten

Die Auswahl der Eingabedaten spielt eine wichtige Rolle bei der Modellierung eines ANFIS-Systems. Bei einem Problem in der Realität ist es nicht unüblich, dass die mögliche Datenmenge in einer großen Vielzahl vorhanden ist. Eine derart exzessive Anzahl an Eingabedaten führt zwar zur besseren Darstellung des Systems, allerdings erhöht sich die Komplexität zum Aufbau der Modellierung und damit auch die Rechenzeit. Dementsprechend ist es notwendig, eine akkurate Methode zur Bestimmung der Eingabedaten zu definieren. Insbesondere um eine angemessene Darstellung mit den begrenzten Eingabedaten zu erreichen. Durch die Begrenzung ist das System auch entsprechend praktikabel in der Implementierung, die das Modell einfacher und zuverlässiger gestalten lässt.

Die optimale Methode, um entsprechende Resultate zu erzielen, wäre alle möglichen Kombinationen der Eingabedaten zu vergleichen und nach den Ergebnissen zu urteilen. Folglich ist dieser Ansatz suboptimal, da durch eine Eingabe von N Daten, die mögliche Kombination um 2^{N} steigt. Nur bei geringer Anzahl an Parametern kommt diese Methode infrage. In der Literatur werden Ansätze zur Bestimmung der Eingabe vorgeschlagen. Verfahren wie ARD, CART und δ-test präsentieren eine Selektion der Eingabewerte durch Bestimmung der Abhängigkeit der Daten [68]. Jang veröffentlicht eine Publikation, die sich mit der Problematik auseinandersetzt [69; 70]. In dieser Arbeit präsentiert der Autor einige praktische Überlegungen:

- Entferne nicht relevante Eingaben. Dieses Wissen kann nur durch einen Experten in dem Gebiet des modellierten Systems durchgeführt werden.
- Entferne Eingabedaten, die von anderen Daten abgeleitet werden können.
- Erstelle das Modell knapp und transparent.
- Verringere die Zeit zur Konstruktion des Modells. Die Einschränkung der Anzahl der Parameter führt zu einer Reduktion der Zeit zur Modellierung der Konstruktion.
- Die ausgewählten Parameter sollten einen Einfluss auf den Ausgang des Systems haben. Eine starke Verbindung zwischen den Parametern der Eingabe und dem Ziel des Systems bzw. Ausgangsvariablen muss bestehen.
- Die ausgewählten Parameter und deren korrespondierenden Daten müssen rein von Störungen sein.

In der Publikation von Jang werden zwei Ansätze aufgeführt, die die Modelle gemäß dem kleinsten Fehler auswählt [70].

Die erste Methode bestimmt die Eingabedaten durch den sequenziellen Suchalgorithmus. Für diesen Algorithmus existieren zwei Varianten:

- sequenzielle Rückwärtssuche (SFS) und
- sequenzielle Vorwärtssuche (SBS).

Bei der Rückwärtssuche werden nach und nach die Eingabewerte aus der Prämisse der Regeln entfernt und nach der Fehlerberechnung die Performance des Modells evaluiert. Wenn der Fehler des Modells sich verringert, wird die nächste Variable aus den Eingabeparametern entfernt. Sofern der Fehler sich erhöht, wird die vorher entfernte Variable behalten und eine andere Variable aus der Datenmenge rausgenommen. Dieser Prozess wird solange durchgeführt, bis der Fehler durch die Eliminierung der Eingabevariablen sich nicht mehr verringert.

Zur Bestimmung des Fehlers wird die Wurzel der mittleren quadratischen Abweichung (RMSE) benutzt.

$$RMSE = \sqrt{\frac{\sum_{p=1}^{n}\left(Y_p - y_p\right)^2}{n}} \tag{4.2}$$

Dabei ist Y der erwartete Wert und *y* der vorausgesagte Wert. An dieser Stelle wird angenommen, dass bei der ersten Epoche des Trainings, der kleinste RMSE Wert auch bei weiteren Epochen klein bleibt [69]. Diese Annahme trifft nicht immer zu, ist aber heuristisch vertretbar. Hier sei noch einmal hervorgehoben, dass das Trainieren von einer großen Anzahl an unterschiedlichen ANFIS-Modellen mit nur einer Epoche weniger Rechenzeit in Anspruch nimmt, als wenn man ein ANFIS-Modell mit mehreren Epochen trainiert [69]. Ferner gilt RMSE als ein guter Ansatz zur Beurteilung der Prognosegüte, wie Chai und Draxler in ihrer Arbeit nachweisen [71].

Als Beispiel für die sequenzielle Rückwärtssuche soll ein Modell mit vier Eingängen dienen. So hat das Modell gemäß der TSK-Regelbasis die Form (4.3).

$$
\begin{aligned}
&Regel_i: \\
&Wenn\ \left(x_1\ ist\ A_1^{(i)}\right)\ und\ \left(x_2\ ist\ A_2^{(i)}\right)\ldots\ und\ \ldots\ \left(x_4\ ist\ A_4^{(i)}\right) \\
&dann\ \ f_i = a_i^T \cdot x + r_i
\end{aligned}
\tag{4.3}
$$

Durch die Eliminierung einer Variable aus der Regel kann überprüft werden, ob die Variable einen Einfluss auf das Modell hat. An dieser Stelle ist die Variable x_3 ausgewählt, die aus dem Modell temporär ausgeschlossen wird. Demnach entsteht die Regelbasis der Form (4.4).

$$
\begin{aligned}
&Regel_i: \\
&Wenn\ \left(x_1\ ist\ A_1^{(i)}\right)\ und\ \left(x_2\ ist\ A_2^{(i)}\right) und \left(x_4\ ist\ A_4^{(i)}\right) \\
&dann\ \ f_i = a_i^T \cdot x + r_i
\end{aligned}
\tag{4.4}
$$

Falls dieses Modell einen minimalen RMSE-Wert hat, d. h. ein geringerer Verlust der Leistung gegenüber dem erwarteten Wert erzielt, kann die Variable x_3 entfernt werden. In der Abbildung 4.3 ist der mögliche Ablauf eines Modells mit vier Eingabedaten zu sehen.

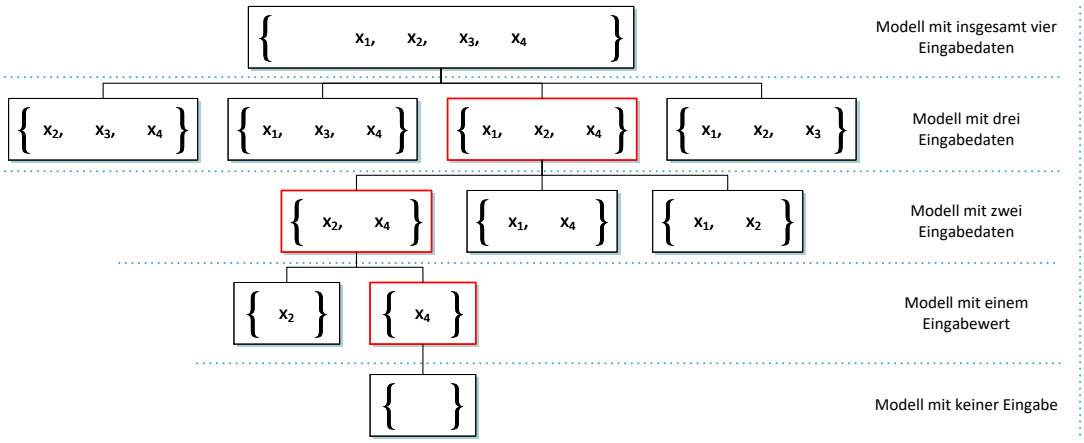

Abbildung 4.3: Auswahl der Eingabedaten mit der sequenziellen Rückwärtssuche

Bei der sequenziellen Vorwärtssuche wird jeder Eingang einzeln betrachtet und die Auswahl nach dem kleinsten RMSE-Wert getroffen. Ist die Auswahl für eine Variable bestimmt, können mit dieser Variable weitere Modellvariationen getestet werden. Dieser Vorgang

wiederholt sich bis die optimalen Variablen, gemäß des RMSE-Kriteriums, definiert sind. Ein möglicher Ablauf ist in der Abbildung 4.4 dargestellt.

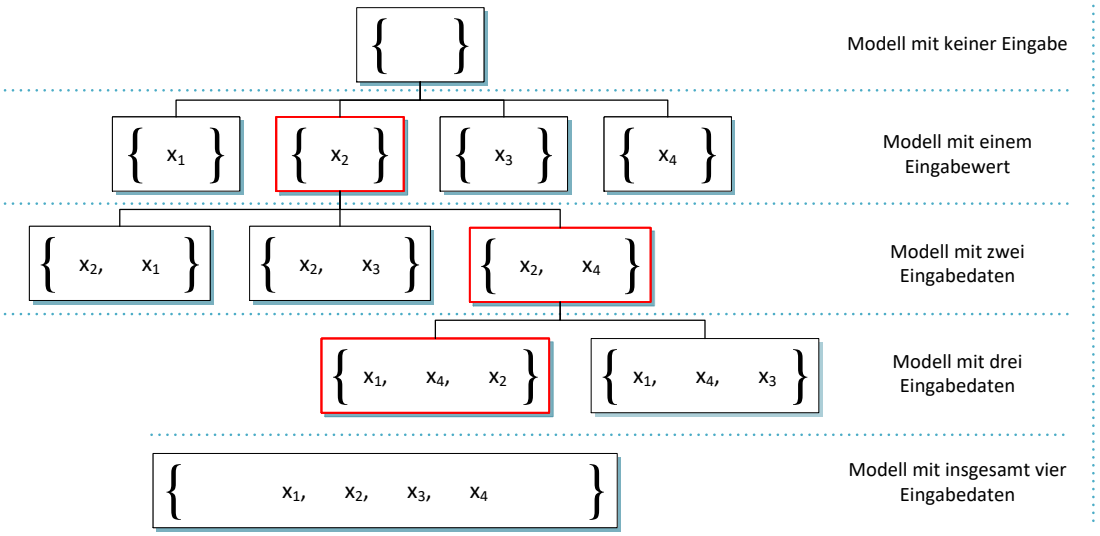

Abbildung 4.4: Auswahl der Eingabedaten mit der sequenziellen Vorwärtssuche

MATLAB bietet in diesem Kontext die Methode „*seqsrch*“ an, um die Variablen mithilfe der Vorwärtsrechnung zu finden. Die Ausgabe der Methode zum obigen Beispiel, kann aus dem Listing 4.2 entnommen werden.

```
Selecting input 1 ...
ANFIS model 1: x1 --> trn=4.6400, chk=4.7255
ANFIS model 2: x2 --> trn=4.2577, chk=4.0863
ANFIS model 3: x3 --> trn=4.5399, chk=4.1713
ANFIS model 4: x4 --> trn=4.3106, chk=4.4316
Currently selected inputs: x2

Selecting input 2 ...
ANFIS model 5: x2 x1 --> trn=3.8741, chk=4.6763
ANFIS model 6: x2 x3 --> trn=4.0271, chk=4.6345
ANFIS model 7: x2 x4 --> trn=2.7657, chk=2.9953
Currently selected inputs: x2 x4

Selecting input 3 ...
ANFIS model 8: x2 x4 x1 --> trn=2.4951, chk=4.0435
ANFIS model 9: x2 x4 x3 --> trn=2.3603, chk=2.9152
Currently selected inputs: x2 x4 x3
```

Listing 4.2: Ausgabe der MATLAB-Funktion „*seqsrch*“ für vier Eingabedaten

An dieser Stelle soll erwähnt werden, dass die „*seqsrch*“-Methode zwei Datenbasen benutzt. Die erste Datenbasis dient dem Training des ANFIS-Modells und die zweite Datenbasis zur Überprüfung. Das aus dem Training gewonnene System wird in zweiter Instanz mit den Trainingsdaten überprüft und der RMSE-Wert (trn) ermittelt. Im Anschluss wird das System mit der zweiten Datenbasis validiert und der RMSE-Wert (chk) berechnet. Um eine gute Generalisierung zu erreichen, ist es wichtig, dass die Trainingsdaten um ein mehrfaches größer sind, als die Testdaten.

Neben der sequenziellen Suche veröffentlicht der Autor Jang einen zweiten Ansatz zur Selektierung der Eingabedaten [70]. Bei diesem Algorithmus handelt es sich um die erschöpfende Suche. Diese Methode unterscheidet sich von der vorherigen Methode maßgebend darin, dass es zu jeder Kombination ein ANFIS-Modell erstellt. An dieser Stelle ist zu betonen, dass diese Methode sehr aufwendig ist und deshalb eher für kleine Anzahl an Eingabeselektion erfolgen sollte.

Es sei angenommen, dass in einem System sechs Eingabevariablen vorhanden sind und daraus nur drei Variablen selektiert werden. Damit entstehen mit der Kombination[18] $C\,(6,3)$

$$C(n,r) = C_r^n = \binom{n}{r} = \frac{n!}{r!(n-r)!} \tag{4.5}$$

zwanzig Testmodelle, die durch RMSE bewertet werden. Die Methode „*exhsrch*“ liefert entsprechend die Ausgabe aus Listing 4.3.

```
Train 20 ANFIS models, each with 3 inputs selected from 6 candidates...

ANFIS model 1: x1 x2 x3 --> trn=0.3752, chk=0.3041
ANFIS model 2: x1 x2 x4 --> trn=0.3916, chk=0.2465
ANFIS model 3: x1 x2 x5 --> trn=0.3896, chk=0.2823
ANFIS model 4: x1 x2 x6 --> trn=0.3136, chk=0.2908
ANFIS model 5: x1 x3 x4 --> trn=0.3696, chk=0.2972
ANFIS model 6: x1 x3 x5 --> trn=0.3918, chk=0.3289
ANFIS model 7: x1 x3 x6 --> trn=0.3354, chk=0.3127
ANFIS model 8: x1 x4 x5 --> trn=0.4045, chk=0.3589
ANFIS model 9: x1 x4 x6 --> trn=0.3412, chk=0.3919
ANFIS model 10: x1 x5 x6 --> trn=0.3255, chk=0.3983
ANFIS model 11: x2 x3 x4 --> trn=0.4084, chk=0.3070
ANFIS model 12: x2 x3 x5 --> trn=0.4056, chk=0.3048
ANFIS model 13: x2 x3 x6 --> trn=0.3558, chk=0.2250
ANFIS model 14: x2 x4 x5 --> trn=0.4208, chk=0.3183
```

[18] Ohne Wiederholung mit ungeordneter Stichprobe

```
17 ANFIS model 15: x2 x4 x6 --> trn=0.3668, chk=0.1777
18 ANFIS model 16: x2 x5 x6 --> trn=0.3640, chk=0.2670
19 ANFIS model 17: x3 x4 x5 --> trn=0.4117, chk=0.3462
20 ANFIS model 18: x3 x4 x6 --> trn=0.3601, chk=0.2866
21 ANFIS model 19: x3 x5 x6 --> trn=0.3628, chk=0.3047
22 ANFIS model 20: x4 x5 x6 --> trn=0.4128, chk=0.3030
```

Listing 4.3: Ausgabe der MATLAB-Funktion „*exhsrch*“ für drei Eingaben aus sechs Eingabedaten

Aus dem Listing 4.3 kann entnommen werden, dass das ANFIS-Modell Nr. 4 mit seinen Variablen $\{x1, x2, x6\}$ den besten RMSE-Wert liefert. Diese Modellierung wurde mit einer Epoche trainiert. Soll eine genauere Vorhersage getroffen werden, können weitere Epochen in das Training einfließen. Als negativen Effekt ist mit einer weitaus längeren Bearbeitungsdauer zu erwarten.

Ein weiterer Ansatz zur Ermittlung von den relevanten Eingabedaten bietet die Anwendung WEKA [72]. WEKA eröffnet dem Anwender eine Vielzahl an Techniken aus maschinellen Lernen und Data-Mining. Zu denen gehören Klassifikatoren wie beispielsweise KNN, Bayes-Klassifikatoren, Support-Vector-Maschinen oder auch Clusteranalysen wie EM-Clustering, k-Means-Clustering, Cobweb. Zudem kann eine Visualisierung der Analysen ausgegeben werden, die dem Anwender eine bessere Sicht auf die Daten verschaffen kann. Die Software ist in der Sprache Java geschrieben und von der Universität Waikato unter der GNU General Public License frei zur Verfügung gestellt. Durch die modulare Architektur von WEKA und dem Paket-Manager lassen sich weitere Verfahren in die Software einbinden. Damit ist stets ein passendes Verfahren vorhanden, um die Umsetzung zu realisieren.

Die aufgeführten Methoden werden im Kapitel 5 für die Auswahl der Eingabeparameter an einer Brustkrebsdiagnose angewandt und die Ergebnisse verglichen.

5 Realisierung und Test

In diesem Kapitel wird ein konkretes ANFIS-System an einem Fallbeispiel in der Brustkrebsdiagnose realisiert. Hierbei wird auf unterschiedliche neuronale Ansätze im Vergleich zum ANFIS aufgezeigt und bewertet.

Zunächst findet eine Analyse der Brustkrebsdaten statt. Im Anschluss darauf wird der Eingaberaum selektiert und mit unterschiedlichen Modellen umgesetzt. Dabei werden folgende neuronale Netze verwendet.

- mehrlagiges Perzeptron,
- Radiale-Basisfunktionen-Netze und
- Adaptives Neuro-Fuzzy-Inferenzsystem.

5.1 Fallbeispiel – Brustkrebs

Krebs ist mit 8,8 Millionen einer der größten Ursachen von Todesfällen weltweit [73]. Dazu zählt der Brustkrebs zu den häufigsten Krebserkrankungen bei Frauen. Im Jahre 2013 erkrankten allein schon in Deutschland 71 640 Frauen und 682 Männer an Brustkrebs. Der Krankheitsverlauf eines Krebspatienten erstreckt sich über mehrere Jahre. So ist die Zahl der Frauen wesentlich höher, die in Behandlung oder Nachsorge befinden [73; 74]. In den letzten Jahrzehnten ist die Brustkrebsinzidenz in Deutschland angestiegen. Allein schon in 2013 lag die Mortalität der Frauen bei 17 853 Todesfällen durch Brustkrebs. Der Anstieg der Inzidenz ist auf unterschiedliche Ursachen, wie die Zunahme von kinderlosen Frauen bzw. Zunahme des Alters bei der ersten Geburt oder auch die vermehrte Einnahme von Arzneimittel

(Hormonersatztherapie, Schwangerschaftsverhütung) zurückzuführen. Ebenso sind die Veränderungen von Ernährungs- und Bewegungsgewohnheiten ein Faktor, der zu einem Anstieg an Erkrankungen beigetragen hat. Zusätzlich ist noch die Erhöhung der Entdeckungsrate durch organisierte oder nicht-organisierte Früherkennungsmaßnahmen zu nennen [74].

Bei einem Verdacht einer Krebserkrankung ist es wichtig, eine akkurate Diagnose zu fällen. Dabei kann es bei einem Geschwulst, um einen gutartigen oder bösartigen Brusttumor handeln. Eine richtige Diagnose zwischen den beiden Fällen zu treffen, kann in manchen Situationen ein schwieriges Unterfangen sein. Dementsprechend soll der Ansatz mit ANFIS bei der Diagnosefindung eine unterstützende Rolle spielen. Im Folgenden wird die Anwendung des Ansatzes auf Grundlage einer frei zugänglichen Datenbasis betrachtet. Hierzu wird ein Vergleich zum MLP- und RBF-Netzen durchgeführt.

5.2 Datenanalyse

Für das Fallbeispiel wird die Datenbasis von University of Wisconsin Hospitals (Madison) benutzt. Die Daten stammen von Proben, die mithilfe einer Feinnadelbiopsie[19] (FNB) entnommen wurden. Die Diagnoseergebnisse wurden der University of California Irvine gespendet und sind auf deren Archiv für maschinelles Lernen (UCI machine learning repository) [75] öffentlich verfügbar. Die Datenbasis setzt sich aus 699 Proben mit jeweils elf Attributen zusammen. Die Eingabeparameter sind auf einen Wertebereich von [1;10] normalisiert. Eine Übersicht der Attribute ist in der Tabelle 5.1 zu sehen.

Tabelle 5.1: UCI-Datenbasis für Brustkrebs (Wisconsin) aus 699 Proben [75]

Nr.	Attribute	Attributwerte	Mittelwert	Varianz
1	Probecode-Nummer	ID Nummer	-	-
2	Gruppendichte (Gdi)	1 - 10	4,4422	7,9567
3	Gleichmäßigkeit der Zellengröße (GZg)	1 - 10	3,1508	9,3951
4	Gleichmäßigkeit der Zellenformen (GZf)	1 - 10	3,2152	8,9316
5	marginale Adhäsion (MAdh)	1 - 10	2,8302	8,2057
6	Epithelial-Zellengröße (EZ)	1 - 10	3,2343	4,9421
7	zythoplasmaarme Zellen (ZP)	1 - 10	3,5447	13,277
8	Bland-Chromatin (BChr)	1 - 10	3,4451	6,0010
9	Normal-Nucleolus (NNcl)	1 - 10	2,8697	9,3188

[19] Verfahren zur Gewinnung von Zellen aus einem Tumor oder Organ.

10	Mitose (M)	1 - 10	1,6032	3,0022
11	Klasse	2 für gutartig 4 für bösartig	2,6999	0,9112

In der Datenbasis existieren 241 bösartige Fälle (34,48 %) und 458 gutartige Fälle (65,52 %). Die bösartigen Datensätze sind durch eine Biopsie der Brustgewebe ermittelt und die Gutartigkeit der Fälle sind entweder mithilfe einer Biopsie oder periodische Untersuchungen bestätigt. Aus den 699 Proben sind 16 Datensätze nicht vollständig angegeben. Für diese Arbeit werden diese Datensätze aus der Datenbasis entfernt. Davon sind 14 gutartige Datensätze und zwei bösartige Datensätze. Damit wird eine Datenbasis von 683 Proben mit einem Bestandteil aus 239 bösartigen Fällen (34,99 %) und 444 gutartigen Fällen (65,01 %) erhalten.

Eine kurze Beschreibung der Begrifflichkeiten aus der Tabelle 5.1 ist folgend aufgeführt.

- **Gruppendichte:** Krebszellen sind oft als mehrschichtig gruppiert, wogegen Gutartige einschichtig sind.
- **Gleichmäßigkeit der Zellengröße:** Bewertet die Konsistenz in Bezug auf die Zellengröße der Probe.
- **Gleichmäßigkeit der Zellenformen:** Bewertet die Qualität der Zellenform und identifiziert die marginale Varianz.
- **marginale Adhäsion:** Normale Zellen tendieren dazu, zusammenzukleben. Kranke Zellen verlieren diese Eigenschaft. Der Verlust der Adhäsion ist ein Indiz für Krebszellen.
- **Epithelial-Zellengröße:** Bezieht sich auf die Zellen-Gleichmäßigkeit, um zu ermitteln, ob sich die Zelle signifikant vergrößert hat.
- **zythoplasmaarme Zellen:** Berechnet die Proportion von der Anzahl der Zellkerne, die nicht von einem Zytoplasma umschlossen sind und denen die mit Zytoplasma umschlossen sind.
- **Bland-Chromatin:** Beschreibt die einheitliche Struktur der Zellkerne (fein bis grob).
- **Normal-Nucleolus:** Kernkörperchen (Nucleolus) sind kleine Strukturen in einem Zellkern. In normalen Zellen sind die Kernkörperchen sehr klein (sogar nicht sichtbar). Bei einem Krebs ist die Kernzelle hervorgehoben oder auch mehrfach vorhanden.
- **Mitose:** Beschreibt die Stufe der Aktivität einer Zellproduktion.

Die Datenbasis wird in zwei Datenbereiche aufgeteilt: Einmal die Trainingsdaten und als zweite Instanz die Daten zur Überprüfung des trainierten Systems. Diese Aufteilung ist notwendig, um zu überprüfen, ob festgelegte Nutzungsziele erfüllt sind und somit die Anforderungen an das System gerechtfertigt sind. Bezüglich der Größe der Aufteilung wird in der Literatur empfohlen, dass die Trainingsdaten den wesentlichen Anteil ausmachen bzw. die Eigenschaften des Modells gut erfassen. Das Minimum der Trainingsdaten sollte nicht kleiner als die Anzahl der modifizierbaren ANFIS-Parameter sein [76]. Damit die Trainingsdaten alle Eigenschaften des Modells einschließen, ist es wichtig, die Eigenschaften der Daten zu kennen. Falls die Eigenschaften nicht bekannt sein sollten, kann eine Analyse der Fehler von den Prüfdaten Aufschluss geben, ob diese mit den Trainingsdaten gut übereinstimmt. Ist der Fehler der Prüfdaten viel zu groß, ist es nötig, die Trainingsdaten größer zu wählen oder die Zugehörigkeitsfunktionen dem System entsprechend anzupassen. Dabei können der Typ der Zugehörigkeitsfunktion und die Anzahl der Funktionen verändert werden. Ein kompletter Ausschluss der Prüfdaten kann stattfinden, falls die Trainingsdaten alle Eigenschaften genügend umfassen.

Für dieses Beispiel sind die Daten in 400 Trainings- und 283 Prüfdaten eingeteilt. Die Trainingsdaten haben 228 gutartige und 172 bösartige Fälle. Die Prüfdaten hingegen bestehen aus 216 gutartigen und 67 bösartigen Fällen. Die Ausgabewerte sind auf einen Wertebereich [0,1] angepasst.

5.3 Wahl der Eingabeparameter

Wie auch in Kapitel 4.3 betont, ist die Wahl der Eingabeparameter ein wichtiger Prozess, um eine gute Modellierung eines Systems zu gewährleisten. Im Folgenden werden unterschiedliche Modellierungen vorgenommen. Das erste Attribut aus der Tabelle 5.1 wird entfernt, da es keine Relevanz für die Aussagekraft der Diagnose darstellt. Für die anderen Eingabeparameter werden die Methoden der MATLAB-Funktionen „*exhsrch*“ und „*seqsrch*“ (jeweils mit einer Epoche Training) benutzt, um die relevantesten Eingaben aus den neun möglichen Parametern zu bestimmen. Des Weiteren wird ein Ranking der Parameter mit der Anwendung WEKA erstellt. Damit ist eine weitere Instanz zur Einstufung der Parameter im vorgegebenen Modell gegeben.

Als Erstes kommt der Ansatz mit der sequenziellen Suche für maximal vier Parametern über neun Eingangsdaten zur Verwendung. Bei der Auswertung wird ein guter RMSE-Wert mit einer Kombination aus drei Parametern erhalten. Die Ausgabe der Ergebnisse ist aus dem

Listing 4.1 zu entnehmen. Demnach bietet das Modell mit der Nummer 18 den besten RMSE-Wert.

```
...
21 ANFIS model 16: in2 in8 --> trn=0.2236, chk=0.1749
22 ANFIS model 17: in2 in9 --> trn=0.2427, chk=0.1771
23 Currently selected inputs: in2 in6
24
25 Selecting input 3 ...
26 ANFIS model 18: in2 in6 in1 --> trn=0.1651, chk=0.1269
27 ANFIS model 19: in2 in6 in3 --> trn=0.1859, chk=0.1300
28 ANFIS model 20: in2 in6 in4 --> trn=0.1878, chk=0.2445
29 ANFIS model 21: in2 in6 in5 --> trn=0.1917, chk=0.1439
30 ANFIS model 22: in2 in6 in7 --> trn=0.1898, chk=0.1323
31 ANFIS model 23: in2 in6 in8 --> trn=0.1778, chk=0.1215
32 ANFIS model 24: in2 in6 in9 --> trn=0.1910, chk=0.1406
33 Currently selected inputs: in1 in2 in6
34
35 Selecting input 4 ...
36 ANFIS model 25: in1 in2 in6 in3 --> trn=0.1406, chk=0.4101
37 ANFIS model 26: in1 in2 in6 in4 --> trn=0.1315, chk=0.2904
...
```

Listing 5.1: Ausgabe der *„seqsrch“*-Methode von vier Parametern aus neun Eingangsdaten.

Mit der sequenziellen Suche ergeben sich daraus die Eingabeparameter:

- Gruppendichte (Gdi),
- Gleichmäßigkeit der Zellengröße (GZg) und
- zythoplasmaarme Zellen (ZZ).

Eine Kombination aus vier Parametern verschlechtert den RMSE-Wert als auch die Testresultate und wird deshalb nicht berücksichtigt. Dies lässt sich aus der Abbildung 5.1[20] entnehmen. Ebenso schneidet eine Kombination aus zwei Parametern im Durchschnitt schlechter ab.

[20] Die Eingänge in den Abbildungen, wie in der Abbildung 5.1 vorzufinden, werden aus Platzgründen und zur besseren Übersicht die Bezeichnung „*inX*“ erhalten, wobei X der Index des jeweiligen Eingangs ist.

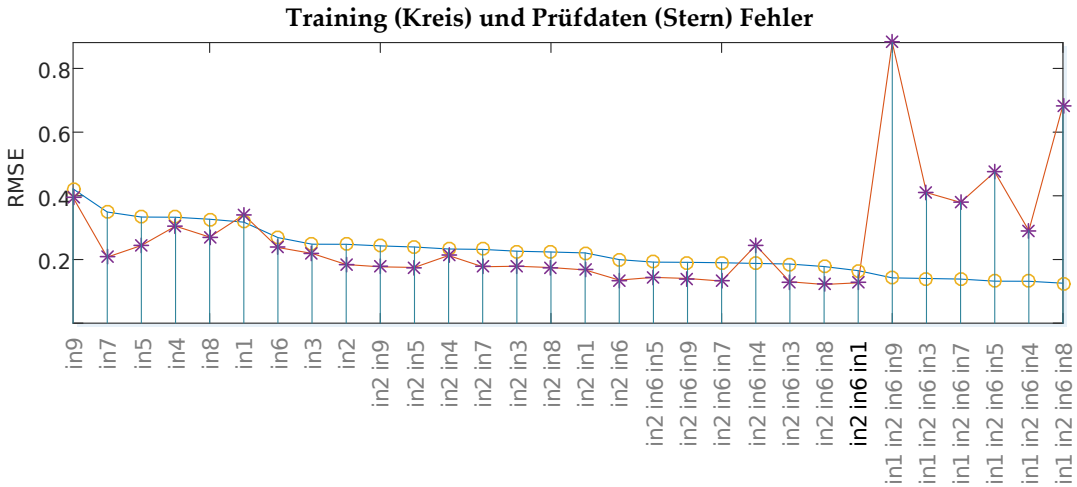

Abbildung 5.1: Grafische Darstellung der RMSE-Ausgabe mit der sequenziellen Methode für 1 – 4 Eingänge

Die Methode der erschöpfenden Suche für drei Parameter liefert in diesem Fall auch das gleiche Endergebnis, wie in der Abbildung 5.2 zu erkennen ist. Eine Modellierung mit vier Parametern liefert einen höheren RMSE als mit drei Parametern.

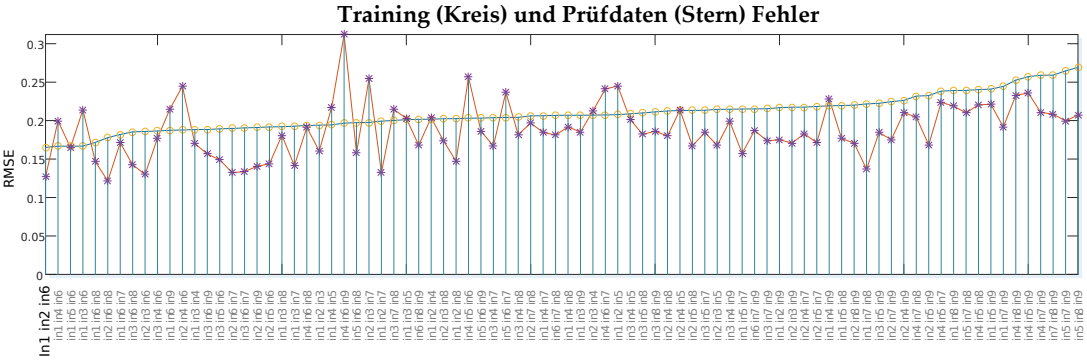

Abbildung 5.2: Grafische Darstellung der RMSE-Ausgabe mit der erschöpfenden Suche für drei Eingänge

An dieser Stelle sei noch mal betont, dass alle Berechnungen mit der „*seqsrch*“- und „*exhsrch*“-Methode nur mit einer Epoche trainiert sind und damit kein genaues Indiz für ein optimales Modell darstellen. Dies hat sich jedoch in der Regel als guter Wegweiser herausgestellt. Mit diesem Wissen ausgehend, werden die Eingabeparameter Gdi, GZg und ZZ für unsere Modellierung verwendet.

Als dritte Instanz wird die Anwendung WEKA (Version 3.8.1) benutzt, um weitere Eingabemöglichkeiten zu ermitteln. Als Erstes müssen die Daten entsprechend vorbereitet werden, um das Ranking zu ermöglichen. Dafür werden die fehlenden Parameter mit der Filtereingabe aus Listing 5.2 bereinigt und die Merkmale nominal skaliert.

```
> Breast_cancer-
weka.filters.unsupervised.attribute.ReplaceMissingWithUserConstant-
Afirst-last-R600-Fyyyy-MM-dd\'T\'HH:mm:ss-
weka.filters.unsupervised.instance.RemoveWithValues-S599.0-C6-Lfirst-
last-V
```

Listing 5.2: Filterung der Brustkrebs-Datensätze mit WEKA

Um die Auswahl der Parameter zu treffen, wurde das Algorithmus „*GainRatioAttributeEval*“ mit der Option „*Ranker -T -1.7976931348623157E308 -N -1*“ verwendet. Dabei bewertet der Algorithmus den Wert der Attribute hinsichtlich des Gewinn-Verhältnisses zum Ausgangsparameter. Auch bei einer Option „*-N -3*“ führt der Algorithmus für drei Eingabeparameter zum gleichen Ergebnis. Eine Auflistung des Rankings ist aus dem Listing 5.3 zu entnehmen.

```
...
=== Attribute Selection on all input data ===

Search Method:
    Attribute ranking.

Attribute Evaluator (supervised, Class (nominal): 10 Klasse):
    Gain Ratio feature evaluator

Ranked attributes:
 0.303    6 Zythoplasmaarme_Zellen
 0.3      2 Gleichmaessigkeit_der_Zellengroesse
 0.272    3 Gleichmoessigkeit_der_Zellenformen
 0.237    8 Normal_Nucleolus
 0.233    5 Epithelial_Zellengroesse
 0.21     4 Marginale Adhaesion
 0.201    7 Bland Chromatin
 0.188    9 Mitose
 0.152    1 Gruppendichte

...
```

Listing 5.3: Ausgabe des WEKA-Rankings mit der Option „*GainRatioAttributeEval*“

Die Attribute, welche weniger als 25 % bewertet wurden, sind aus der Eingabeliste herausgenommen. Damit kommen folgende Parameter zustande.

- Gleichmäßigkeit der Zellengröße (GZg),
- Gleichmäßigkeit der Zellenformen (GZf) und
- zythoplasmaarme Zellen (ZZ).

Zusätzlich soll ein Diagnosesystem mit fünf Eingaben modelliert werden. Da die sequenzielle- und erschöpfende Suche, wie in Kapitel 4.3 besprochen, ab einer Anzahl von vier Parametern und seinen Zugehörigkeitsfunktionen sehr schnell dimensioniert, wird die Anwendung WEKA benutzt. Hierfür wird aus dem Listing 5.3 die ersten fünf Parameter entnommen und noch einen weiteren Algorithmus *„InfoGainAttributeEval“* mit der Option *„Ranker -T -1.7976931348623157E308 -N -1“* verwendet. Eine Übersicht über alle Modelle und die entsprechenden Parameter ist in der Tabelle 5.2 aufgelistet.

Tabelle 5.2: Zusammenfassung der Parameterselektion mit jeweils unterschiedlichen Algorithmen

	Eingabeparameter									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	
Modell	**Gdi**	**GZg**	**GZf**	**MAdh**	**EZ**	**ZP**	**BChr**	**NNcl**	**M**	
Seqsrch/Exhsrch	X	X	-	-	-	X	-	-	-	3 Param.
WEKA-Gain 1	-	X	X	-	-	X	-	-	-	
WEKA-Gain 2	-	X	X	-	X	X	-	X	-	5 Param.
WEKA-InfoGain	-	X	X	-	X	X	X	-	-	

5.4 Mehrlagiges Perzeptron

Für das erste MLP-Netz werden drei Eingänge für die erste Schicht, eine Schicht für den Ausgang und eine Schicht mit zehn versteckten Neuronen benutzt. Damit besteht das Netz aus drei Schichten. Aus dem Listing 5.4 können die Angaben zum Netz entnommen werden.

```
>
Algorithms

   Data Division: dividerand
   Training: scaled conjugate gradient
   Performance: cross-entropy
```

```
   Calculations: MEX
...
```

Listing 5.4: Algorithmen für das MLP-Netz

Die Ergebnisse der sequenziellen/erschöpfenden Suche werden benutzt, um die Eingänge mit Daten zu beliefern. Damit schließt das erste MLP-Netz sein Training nach 22 Iterationen ab. Mit diesem Netz ist ein RMSE-Wert von 0,1472 erreicht.

Für das zweite MLP-Netz werden die Daten aus dem Modell „*WEKA-Gain 1*“ benutzt. Die Struktur bleibt dieselbe wie beim ersten Netz. Nach 15 Iterationen wird das Training des Netzes beendet. Dieses Netz zeigt eine leichte Verbesserung bezüglich des RMSE mit einem Wert von 0,1422.

Das dritte Netz für die fünf Parameter GZg, GZf, EZ, ZP und NNcl besteht ebenfalls aus drei Schichten und schließt sein Training nach 36 Iterationen ab. Die Performanz des Netzes mit drei Eingängen fällt etwas geringer aus, aber das Netz liefert einen besseren RMSE-Wert von 0,1318.

Für die vierte Variante werden die Eingänge aus „*Weka-InfoGain*“ mit den Parametern GZg, GZf, EZ, ZP und BChr verwendet. Das Netz setzt sich, wie bei den Vorgängern auch, aus drei Schichten zusammen. Das Training des MLP mit fünf Eingängen dauert 22 Iterationen. Der RMSE-Wert mit diesem Modell ist angestiegen auf den Wert 0,1351.

Die Treffsicherheit[21] der Diagnosen variieren zwischen 94,7 % (mit 15 Fehldiagnosen) und 96,5 % (mit zehn Fehldiagnosen). Eine Übersicht von allen Ergebnissen ist aus der Tabelle 5.3 zu entnehmen.

Tabelle 5.3: Ergebnisse der Brustkrebsdiagnosen für unterschiedliche Modelle mit einem MLP-Netz

MLP-Modell	Seqsrch/ Exhsrch	WEKA-Gain 1	WEKA-Gain 2	WEKA-InfoGain
Iterationen	22	15	36	14
RMSE	0,1472	0,1422	0,1318	0,1351
nicht erkannter Krebs	4	2	2	0
fälschlicherweise erkannter Krebs	2	2	2	2

[21] Treffsicherheit (oder auch Treffgenauigkeit) ist der Quotient von allen richtigen Diagnosen (richtig positiv + richtig negativ) und von allen Diagnosefällen (richtig positiv + richtig negativ + falsch positiv + falsch negativ + unklare Fälle).

unklare Diagnose	5	9	6	13
Diagnosefehler insgesamt	11	13	10	15

5.5 Radiale-Basisfunktionen-Netz

Aus den Eingabe- und Ausgabedaten wird ein RBF-Netz erzeugt, dass das Diagnose-Modell bestmöglich repräsentiert. Das Netz setzt sich aus dem Eingangsvektor, der Radialbasis-Schicht und der Ausgangsschicht zusammen. Dafür soll das RBF-Netz schrittweise generiert werden. D. h. dem Netz wird bei jedem Schritt ein neues Neuron hinzugefügt, bis die Summe der quadratischen Fehler unter dem gewünschten Fehlerwert liegt oder die maximale Anzahl an Neuronen erreicht ist. Der Nachteil gegenüber einem normalen neuronalen Netz wie MLP ist, dass es im Vergleich viele Neuronen erzeugt.

Für das erste Modell mit den Parametern Gdi, GZg und ZP wird ein Netz mit 170 Neuronen erzeugt. Das Netz liefert einen RMSE-Wert von 0,1465. Das zweite Modell mit „*WEKA-Gain 1*“ setzt sich aus 200 Neuronen mit einem RMSE-Wert von 0,1769. Das RBF-Netz mit den Parametern aus dem Modell „*WEKA-Gain 2*“ hat einen RMSE von 0,1809 und besteht aus 246 Neuronen. Das letzte Modell hat den höchsten RMSE von 0,1898 mit einer Anzahl an Neuronen von 265.

Die fehlerhaften Diagnosen liegen zwischen neun und 18. Demnach ist eine prozentuale Treffsicherheit zwischen 93,64 % und 96,11 % gegeben. Wobei das vierte Modell das schlechteste Ergebnis und das erste Modell das beste Ergebnis liefert. In der Tabelle 5.4 sind alle Ergebnisse für das RBF-Netz aufgelistet.

Tabelle 5.4: Ergebnisse der Brustkrebsdiagnosen für unterschiedliche Modelle mit einem RBF-Netz

RBF-Modell	**Seqsrch/ Exhsrch**	**WEKA-Gain 1**	**WEKA-Gain 2**	**WEKA-InfoGain**
Anzahl der Neuronen	230	200	246	265
RMSE	0,1465	0,1769	0,1809	0,1898
nicht erkannter Krebs	0	0	0	0
fälschlicherweise erkannter Krebs	4	6	8	8
unklare Diagnose	5	6	7	10
Diagnosefehler insgesamt	9	12	15	18

5.6 ANFIS

In diesem Kapitel werden unterschiedliche ANFIS-Modelle betrachtet. Wie in Kapitel 5.4 und 5.5 werden die Parameter aus der Tabelle 5.2, die im Zuge der Auswahlverfahren definiert sind, als Eingänge gewählt. Zusätzlich werden die Partitionierungsansätze aus dem Kapitel 2.4 für jedes Modell benutzt. Dabei wird die optimale Anzahl von Epochen für das jeweilige System bestimmt und mit dieser trainiert.

Das erste ANFIS-Netz benutzt die Eingänge Gdi, GZg und ZP, die mit dem Verfahren der sequenziellen- und erschöpfenden Suche ermittelt wurden. Die FIS-Struktur wird mit der Grid-Partitionierung erstellt. Jeder der drei Eingänge hat jeweils zwei Zugehörigkeitsfunktionen der Form

$$\mu_{ji}\left(x_j\right)=\frac{1}{1+\left[\left(\frac{x_j-c_i}{a_i}\right)^2\right]^{b_i}}. \tag{5.1}$$

Dies entspricht in MATLAB der speziellen Glockenkurve „gbellmf“. Damit entsteht ein FIS mit der Struktur *input:[1x2 struct]*, *output:[1x1 struct]* und *rule: [1x8 struct]*. Das System setzt sich aus insgesamt, wie im Listing 5.5 aufgeführt, aus fünfzig Parametern, die durch den hybriden Lernalgorithmus trainiert werden. Nach einem Training von 112 Epochen resultiert ein RMSE-Wert von 0,2014.

```
 1
 2 ANFIS info:
 3     Number of nodes: 34
 4     Number of linear parameters: 32
 5     Number of nonlinear parameters: 18
 6     Total number of parameters: 50
 7     Number of training data pairs: 400
 8     Number of checking data pairs: 283
 9     Number of fuzzy rules: 8
10
...
388 RMSE_ANFIS =
389
390     0.2014
```

Listing 5.5: Information über das ANFIS-Modell „Seqsrch/Exhsrch“ mit drei Eingangsparametern und jeweils zwei Zugehörigkeitsfunktionen (Grid-Partitionierung)

Erhöht sich die Anzahl der Zugehörigkeitsfunktionen für jeden Eingang auf drei, kommen gemäß der Partitionierung 27 Regeln zustande und die Parameteranzahl steigt auf 135. Dieses Netz erreicht einen RMSE von 1,5278 mit 58 Epochen. Weitere Epochen verschlechtern das Ergebnis des Modells.

Im dritten Modell wird das Subtractive-Clustering verwendet, um das FIS zu generieren. Wie zu erwarten, bringt das Netz eine geringe Anzahl an Fuzzy-Regeln hervor. Für das Clustering werden ein „squash"-Faktor von η = 1,25 und ein Radius von r = 0,7 konfiguriert. Die Zugehörigkeitsfunktionen haben die Form einer Glockenkurve aus der Gleichung (5.2), welche in MATLAB der „gaussmf"-Funktion entspricht.

$$\mu_{ji}(x_j) = e^{-\left(x_j - b_i\right)^2 / 2a_i^2} \tag{5.2}$$

Durch die minimale Parameteranzahl der Zugehörigkeitsfunktion und der wenigen Regeln müssen nur zwanzig Parameter für das ANFIS-Modell angepasst werden. Das Training erreicht in 51 Epochen den minimalen RMSE von 0,1156.

Im nächsten Modell kommt das Fuzzy C-Means-Clustering zum Einsatz. Mit den Parametern aus dem „*Seqsrch/Exhsrch*"-Modell wird das Netz in siebzig Epochen trainiert und ein RMSE-Wert von 0,1127 erhalten. Die Optionen für das Clustering setzen sich aus dem Exponenten m = 2 und der maximalen Iterationsschritte r = 100 zusammen[22]. Durch das Clustering erhält jeder Eingang drei Zugehörigkeitsfunktionen. Die Funktionen entsprechen der Gleichung (5.2).

Im Folgenden sind die Eingangsparameter GZg, GZf und ZP aus dem Modell „*WEKA Gain 1*" zu betrachten. Als erstes Netz soll ein ANFIS dienen, dass auf einer Grid-Partitionierung basiert. Alle Eingänge werden mit zwei Zugehörigkeitsfunktionen versehen, die der Gleichung (5.1) entsprechen. Das Netz wird in 58 Epochen trainiert und liefert einen RMSE von 0,1293. Ein Netz mit drei Zugehörigkeitsfunktionen liefert bei 36 Epochen Training einen schlechten RMSE von 0,7873. Eine weitere Erhöhung der Zugehörigkeitsfunktionen ist ebenfalls nicht hilfreich, sondern verschlechtert den Fehlerwert.

[22] Diese Optionen werden für alle folgenden ANFIS-Netze verwendet.

Die mit dem Subtractive-Clustering erzeugte FIS-Struktur besteht aus zwei Clustern. Das Clustering wird wie beim Vorgänger mit dem „squash“-Faktor von η = 1,25 und dem Radius von r = 1,0 erzeugt. Daraus erhält jeder Eingang zwei „gaussmf“-Zugehörigkeitsfunktionen. Mit der Struktur trainiert das ANFIS in sechs Epochen und erhält einen RMSE von 0,1338.

Das Fuzzy C-Means-Clustering erzeugt eine FIS-Struktur mit zwei Regeln. Dabei entstehen zwei Zugehörigkeitsfunktionen pro Eingang. Das Training liefert bei 102 Epochen den besten RMSE-Wert von 0,1265.

Die Parameter GZg, GZf, EZ, ZP und NNcl aus dem Modell „*WEKA-Gain 2*“ soll als Nächstes behandelt werden. Bei einer Grid-Partitionierung des Eingangsraums ist das Netz mit je zwei „gbellmf“-Zugehörigkeitsfunktionen trainiert. Den optimalsten RMSE von 1,3077 ist bei einem Training von 117 Epochen erreicht. Eine höhere Zugehörigkeitsfunktion führt zur Überschreitung der modifizierbaren ANFIS-Parameter zum Verhältnis der vorhandenen Trainingsdaten. Bei drei Funktionen kommt damit über 1500 modifizierbare Parameter zustande, welche die Trainingsdaten deutlich überschreitet. Deshalb wird in diesem Kontext auf die Modellierung mit mehr als zwei Parametern verzichtet.

Mit dem Subtractive-Clustering wird ein Netz mit zwei Regeln erzeugt. Jeder Eingang hat zwei „gaussmf“-Funktionen. Der „squash“-Faktor liegt bei η = 1,25 und der Radius bei r = 1,0. Dieses Modell hat nach 79 Epochen einen RMSE-Wert von 0,1197.

Das Fuzzy C-Means-Clustering liefert bei acht Clustern das beste Ergebnis. Damit entsteht ein FIS mit acht Regeln. Nach siebzig Epochen ist ein RMSE-Wert von 0,1155 erreicht. Eine grafische Darstellung der RMSE-Kurve für siebzig Epochen ist in der Abbildung 5.3 zu sehen.

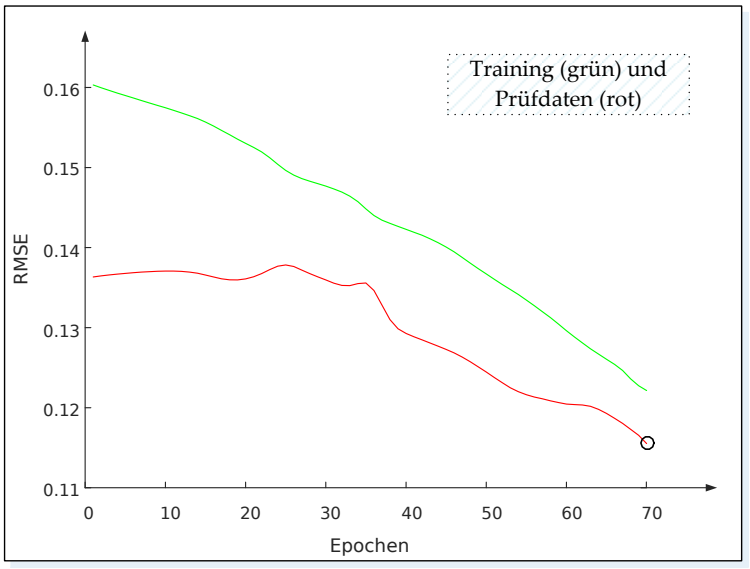

Abbildung 5.3: RMSE-Kurve der Trainings- und Prüfdaten für das „WEKA Gain 2"-Modell mit der Fuzzy C-Means-Clustering

Als Letztes sollen die fünf Eingänge GZg, GZf, EZ, ZP und BChr aus dem „*WEKA-InfoGain*"-Modell betrachtet werden. Das erste ANFIS-Netz liefert einen hohen RMSE-Wert von 0,7383 bei einem Training von 117 Epochen. Das Netz ist mit der Grid-Partitionierung konfiguriert und hat jeweils zwei „gbellmf"-Funktionen pro Eingang. Damit werden 32 Regeln mit insgesamt 222 anpassbaren ANFIS-Parametern erzeugt. Wie auch schon bei der Modellierung von „*WEKA-Gain 2*" erwähnt, können die Zugehörigkeitsfunktionen beschränkt erhöht werden, da die ANFIS-Parameter größer ausfallen als die Trainingsdaten.

Im nächsten Modell soll das Substractive-Clustering genutzt werden, um die Regeln und die Struktur des ANFIS-Netzes zu bestimmen. Das Clustering wird mit einem „squash"-Faktor von η = 1,25 und einem Radius von r = 1,0 eingestellt. Mit der Einstellung ist ein guter RMSE-Wert von 0,1170 erzielt. Das Training der ANFIS-Parameter endet nach 84 Epochen.

Mit dem Fuzzy C-Means-Clustering wird ein ähnliches ANFIS-Netz wie beim Subtractive-Clustering erzeugt. Dieser erreicht ebenfalls nach 95 Epochen einen RMSE-Wert von 0,1170. Die Fehldiagnose erhöht sich um einen Fehler gegenüber dem Subtractive-Clustering.

Eine Übersicht der ANFIS-Modelle ist in der Tabelle 5.5 vorzufinden. Wie schon erwähnt, werden die Modelle, mit mehr als 400 modifizierbaren Parametern nicht betrachtet.

Tabelle 5.5: Ergebnisse der Brustkrebsdiagnosen für unterschiedliche Modelle mit einem ANFIS-Netz

ANFIS-Modell	Seqsrch/Exhsrch				WEKA-Gain 1				WEKA-Gain 2				WEKA-InfoGain			
Anzahl von Eingängen	3								5							
Anzahl an Trainingsdaten	400															
Anzahl an Prüfdaten	283															
Zugehörigkeitsfunktion	gbellmf		gaussmf		gbellmf		gaussmf		gbellmf		gaussmf		gbellmf		gaussmf	
Partitionierungstyp	Grid		Sub	C-M	Grid		Sub	C-M	Grid		Sub	C-M	Grid		Sub	C-M
Anzahl der Zugehörigkeitsfunktionen	2+2+2	3+3+3	2+2+2	3+3+3	2+2+2	3+3+3	2+2+2	2+2+2	10	15	10	40	10	15	10	10
Epochen	112	58	51	70	54	36	6	102	117	-	79	70	119	-	84	95
Anzahl der Neuronen	34	78	22	30	34	78	22	22	92	-	32	104	92	-	32	32
Anzahl der linearen Parameter	32	108	8	12	32	108	8	8	192	-	12	48	192	-	12	12
Anzahl der nichtlinearen Parameter	18	27	12	18	18	27	12	12	30	-	20	80	30	-	20	20
gesamte Anzahl der Parameter	50	135	20	30	50	135	20	20	222	1503	32	128	222	1503	32	32
Anzahl der Fuzzy-Regeln	8	27	2	3	8	27	2	2	32	-	2	8	32	-	2	2
RMSE	0,2014	1,5278	0,1156	0,1127	0,1293	0,7873	0,1338	0,1265	1,3077	-	0,1197	0,1155	0,6824	-	0,1170	0,1170
nicht erkannter Krebs	0	9	0	0	0	10	0	0	13	-	1	1	8	-	0	0
fälschlicherweise erkannter Krebs	3	3	1	2	1	5	1	2	2	-	2	1	4	-	2	2
unklare Diagnose	10	15	9	5	11	22	13	10	33	-	5	5	17	-	5	5
Diagnosefehler insgesamt	13	27	10	7	12	37	14	12	48	-	8	7	29	-	7	7

5.7 Auswertung/Fazit

In den vorherigen Kapiteln wurden Brustkrebsdiagnosen mithilfe von neuronalen Netzen getroffen. Die Tabelle 5.6 zeigt die besten Resultate mit vier verschiedenen Modellen als Ausgangsbasis. Durch die Selektion der Eingabedaten von neun Parametern auf drei und fünf Parametern hat dazu beigetragen, dass die Klassifizierung gute Ergebnisse für die Bestimmung von gutartigen oder bösartigen Tumoren geliefert hat.

Tabelle 5.6: Zusammenfassung der besten Diagnoseergebnisse gemäß der Treffsicherheit

	MLP	RBF-Netz	ANFIS				
Anzahl der Parameter	5	3		5			
Modell	WEKA-Gain 2	Seqsrch/Exhsrch		WEKA-Gain 2		WEKA-InfoGain	
Typ	-	-	C-M	Sub	C-M	Sub	C-M
RMSE	0,1318	0,1465	0,1127	0,1197	0,1155	0,1170	0,1170
nicht erkannter Krebs	2	0	0	1	1	0	0
fälschlicherweise erkannter Krebs	2	4	2	2	1	2	2
unklare Diagnose	6	5	5	5	5	5	5
Diagnosefehler insgesamt	10	9	7	8	7	7	7
Klassifizierungsrate (Treffsicherh.)	0,9646	0,9682	0,9752	0,9717	0,9752	0,9752	0,9752

Die Modelle für das MLP-Netz schneiden im Vergleich zu den anderen Netzen im Testdurchgang am schlechtesten ab. Aus den Ergebnissen ist zu erkennen, dass die nicht erkannten Krebsfälle beim MLP-Netz im Durchschnitt sehr hoch ausfallen. D. h., Patienten werden als gesund diagnostiziert, obwohl die tatsächlich krank sind. Deshalb müssen derartige Diagnosen hoch angerechnet werden. Das RBF-Netz hingegen liefert eine

Sensitivität[23] von 100 %. Dagegen ist die Ausfallrate[24] höher angesiedelt als bei MLP- und ANFIS-Netzen. Im besten Fall sind es vier Fälle, wo ein gesunder Patient zu Unrecht als krank diagnostiziert ist. Bei dem Großteil der Diagnosen handelt es sich um unklare Diagnosen. Das sind Diagnosen, welche nicht eindeutig zu einer der Kategorien klassifiziert werden können.

Das MLP-Netz hat mit sechs unklaren Diagnosen die höchste Fehlerrate. Alle anderen Modelle liegen bei fünf nicht klassifizierbaren Diagnosen. Obwohl das RBF-Modell einen höheren RMSE-Wert als das MLP-Modell hervorbringt, liefert das RBF-Modell eine minimal höhere Treffgenauigkeit. Ausschlaggebend für die Klassifizierungsrate sind hierbei die nicht eindeutig identifizierbaren Diagnosen.

Bei den ANFIS-Modellen wird eine höhere Treffgenauigkeit erreicht. Dabei liegen die fehlerhaften Diagnosen der Patienten zwischen sieben und acht Fehlern. Die Modelle „WEKA-Gain 2“ mit Subtractive-Clustering und Fuzzy C-Means-Clustering haben jeweils einen nicht erkannten Krebsfall als Ergebnis. Das ist eine Sensitivität von 98,5 % (ohne unklare Diagnosen) mit jeweils 65 kranken Patienten (richtig positiv). Bei den drei Eingabedaten aus dem Modell „WEKA-Gain 1“ ist das Resultat nicht optimal und entfällt als Systemmodell für alle Netze. Weiterhin ist bei diesem Fallbeispiel die Erweiterung der Zugehörigkeitsfunktionen keine Konklusion für einen Anstieg an Genauigkeit. Eine hohe Anzahl an Regeln für die ANFIS-Modelle hat einen negativen Effekt auf das Diagnoseresultat. Dies trifft vorwiegend für die Netze mit der Grid-Partitionierung zu. Ein Netz mit einer kleineren Regelbasis modelliert die Krebsdiagnose angemessener.

Das „Seqrch/Exhsrch“-Modell in Verbindung mit dem Fuzzy C-Means Clustering für das ANFIS-Netz – ebenfalls mit drei Eingabedaten – liefert den besten RMSE-Wert. Zum Vergleich mit dem Modell „WEKA-InfoGain“ mit fünf Eingangsdaten ist die Klassifizierungsrate bei Subtractive-Clustering und Fuzzy C-Means-Clustering identisch. Die unterschiedlichen Partitionierungen führen bei diesen Modellen zu einer ähnlichen Netzstruktur, die als Folge einen gleichen RMSE-Wert haben.

---

[23] Anteil der korrekt als positiv klassifizierten Fälle (richtig positiv) an der Gesamtheit der tatsächlich positiven Fälle (richtig positiv + falsch negativ)

[24] Anteil der fälschlich als positiv klassifizierten Fälle (falsch positiv) an der Gesamtheit der in Wirklichkeit negativen Fälle (richtig negativ + falsch positiv)

In der Abbildung 5.4 ist die Häufigkeit der vier möglichen Kombinationen von Testergebnissen und Gesundheitszustand von Patienten für das ANFIS-Modell „WEKA-InfoGain“ mit Subtractive-Clustering zu sehen.

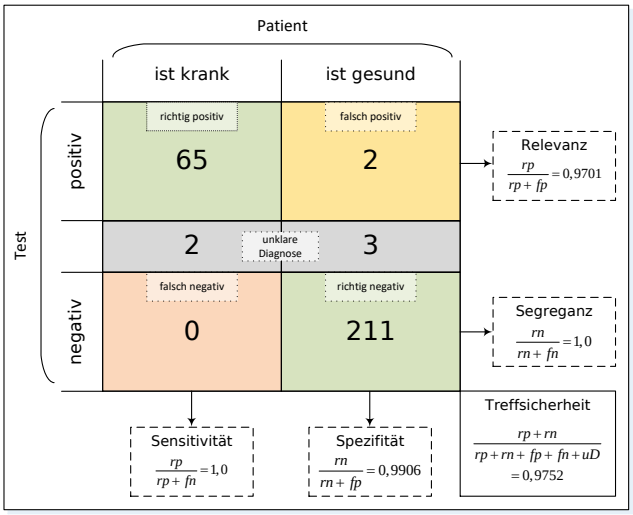

Abbildung 5.4: Konfusionsmatrix für das ANFIS-Modell „WEKA-InfoGain“ mit Subtractive-Clustering

Bei diesem Modell sind keine Krebsfälle fälschlich als negativ klassifiziert. Damit liegt die Sensitivität bei 100 %. Ausgenommen sind die zwei nicht klassifizierbaren Diagnosen, welche in die Beurteilung nicht einfließt. Die Spezifität[25] des Modells ist mit nur zwei falschen Diagnosen bei 99,06 %. Bei einer Ausfallrate von 0,0094 wird ein gutes Modell für ein Diagnosesystem erhalten. Der tatsächliche Anteil an Kranken, die auch als krank diagnostiziert wurden, ist mit einer Rate von 0,9701 festgelegt. Dies entspricht einer positiven Vorhersage von 97,01 %.

Die ANFIS-Modelle aus der Tabelle 5.6 mit Subtractive- und Fuzzy C-Means-Clustering mit jeweils drei und fünf Parametern sind gute Ansätze für ein Diagnosesystem mit den gegebenen Daten. Um die Praxistauglichkeit des Systems auch für zukünftige Diagnosen zu bestätigen und auch die Genauigkeit zu verbessern, sollte eine größer angelegte Datenbasis unter ständiger Beobachtung eines Experten getestet werden.

[25] Anteil der korrekt als negativ klassifizierten Fälle (richtig negativ) an der Gesamtheit der in Wirklichkeit negativen Fälle (richtig negativ + falsch positiv)

6 Schlussbetrachtung

In diesem Kapitel werden die Inhalte der Arbeit zusammengefasst und ein Ausblick der Diagnoseanwendung mit ANFIS gegeben.

6.1 Zusammenfassung

Das Ziel dieser Arbeit war es einen Überblick über die Anwendung von ANFIS in der Diagnose zu erstellen. Für diesen Zweck wurden die Architektur und die unterschiedlichen Varianten des adaptiven Systems näher erläutert. Ebenfalls wurden die verfügbaren Implementierungen und Anwendungsprogramme, welche die ANFIS-Architektur umsetzt, erläutert. Darüber hinaus wurde eine umfangreiche Übersicht von Lernalgorithmen für die Optimierung der ANFIS-Parameter und die Anwendungsgebiete für medizinische Diagnostik aufgeführt. Des Weiteren wurden die Auswahlkriterien des Eingaberaumes eines Modells näher erläutert und die Methoden für die Bestimmung der Eingaben präsentiert. Die dafür genutzte Programme MATLAB und WEKA wurden im Zuge der Arbeit eingeführt und darauf ein Fallbeispiel in der Brustkrebsdiagnose realisiert. Zum Vergleich der Ergebnisse aus den ANFIS-Modellen wurden die neuronalen Netzwerke MLP und RBF mit gleichen Eingangskonfigurationen untersucht und bewertet.

6.2 Fazit und Ausblick

Zusammenendfassend lässt sich sagen, dass ANFIS ein erfolgversprechendes Mittel ist, um Diagnosen mit einer guten Trefferrate zu bestimmen. Diese Erkenntnis wird auch durch

andere Studien untermauert. Ebenso wurde an einem konkreten Fallbeispiel in der Brustkrebsdiagnose gezeigt, dass das ANFIS eine hohe Klassifizierungsrate erzielt und auch besser abschneidet als MLP und RBF-Netz. Jedoch ist bei dem Fallbeispiel mit einer anderen Datenbasis oder mit einem anderen Eingaberaum eine bessere Klassifikation für ein ANFIS-Modell möglich und ist auch nicht auszuschließen. Das adaptive System bietet damit für den Nutzer eine gute unterstützende Rolle in der Diagnose. Allerdings ist in einem riskanten Fall bzw. System eine Kontrolle durch einen Spezialisten zu raten. Ein solches System kann nicht das Wissen und die Fähigkeiten eines Spezialisten ersetzen. Vor allem in der Medizin trifft dies zu.

Ein ANFIS-System ist in der Implementierung leicht umzusetzen und damit für viele Bereiche, wie mobile Anwendung oder als Webdienst vorstellbar. Offene Quelldaten stehen dafür im Internet frei zur Verfügung. Damit kann ein globales Netzwerk für Diagnosesysteme aufgestellt und in Kooperation mit Spezialisten, ein praxistaugliches Diagnosesystem aufgebaut werden. In der medizinischen Diagnose sind nicht nur Anwendungen beim Arzt, sondern auch beim Patienten zu Hause (z. B. durch Pflegebetreuung) denkbar. So kann beispielsweise mithilfe eines ANFIS-Diagnosesystems unter ständiger Beobachtung der essenzielle Zustand des Bedürftigen kontrolliert werden. Damit sind durch die Architektur von ANFIS und dessen Transparenz weitere Möglichkeiten für zukünftige Projekte geboten.

Literaturverzeichnis

[1] McCulloch, W. S., Pitts, W. (1943): A logical calculus of the ideas immanent in nervous activity. The Bulletin of Mathematical Biophysics, 5(4):115–133.

[2] Jain, A. K., Mao, J., Mohiuddin, K. M. (1996): Artificial neural networks. A tutorial. Computer, 29(3):31–44.

[3] Kruse, R. (2015): Computational Intelligence. Eine methodische Einführung in künstliche neuronale Netze, evolutionäre Algorithmen, Fuzzy-Systeme und Bayes-Netze. 2. Auflage. Springer Vieweg, Wiesbaden.

[4] Zadeh, L. A. (1965): Fuzzy sets. Information and Control, 8(3):338–353.

[5] Bezdek, J. (1993): Fuzzy models—What are they, and why? [Editorial]. IEEE Transactions on Fuzzy Systems, 1(1):1–6.

[6] Takagi, T., Sugeno, M. (1985): Fuzzy identification of systems and its applications to modeling and control. IEEE Transactions on Systems, Man, and Cybernetics, SMC-15(1):116–132.

[7] Jang, J.-S. R. (1993): ANFIS: adaptive-network-based fuzzy inference system. IEEE Transactions on Systems, Man, and Cybernetics, 23(3):665–685.

[8] Kruse, R. (2008): Fuzzy neural network. Scholarpedia, 3(11):6043.

[9] Chiu, S. L. (1996): Selecting input variables for fuzzy models. Journal of Intelligent & Fuzzy Systems, (4):243–256.

[10] Walia, N., Kumar, S., Singh, H. (2015): A Survey on Applications of Adaptive Neuro Fuzzy Inference System. International Journal of Hybrid Information Technology, 8(11):343–350.

[11] Ross, T. J. (2010): Fuzzy Logic with Engineering Applications. John Wiley & Sons, Ltd, Chichester, UK.

[12] Ohne Verfasser - MathWorks: Documentation - Membership Function Gallery. https://www.mathworks.com/help/fuzzy/examples/membership-function-gallery.html. Abgerufen am 05.12.2016.

[13] Mizutani, E., Jang, J.-S. (1995): Coactive neural fuzzy modeling. In: , *Proceedings / 1995 IEEE International Conference on Neural Networks, the University of Western Australia, Perth, Western Australia, 27 November - 1 December 1995.* IEEE Service Center, Piscataway, NJ.

[14] Jang, J.-S. R., Sun, C.-T., Mizutani, E. (1997): Neuro-fuzzy and soft computing. A computational approach to learning and machine intelligence. Prentice Hall, Upper Saddle River, NJ.

[15] Tomar, R. S., Qureshi, M. F., and Shrivastava, S. K. (2014): Development of mimo anfis control system for seismic response reduction using multi-objective genetic algorithm. International Journal of Information Research and Review, 1(10):106–112.

[16] Tamura, H., Tanno, K., Tanaka, H., Vairappan, C., Tang, Z. (2008): Recurrent type ANFIS using local search technique for time series prediction. In: , *IEEE Asia Pacific Conference on Circuits and Systems, 2008. APCCAS 2008 ; Nov. 30, 2008 - Dec. 3, 2008, Macao, China.* IEEE, Piscataway, NJ.

[17] Rini, D. P., Shamsuddin, S. M., Yuhaniz, S. S. (2013): Balanced the Trade-offs Problem of ANFIS using Particle Swarm Optimization. TELKOMNIKA (Telecommunication Computing Electronics and Control), 11(3):611.

[18] Wang, R., Zhang, J., Zhang, Y., Wang, X. (2012): Assessment of human operator functional state using a novel differential evolution optimization based adaptive fuzzy model. Biomedical Signal Processing and Control, 7(5):490–498.

[19] Jang, J.-S. R., Mizutani, E. (1996): Levenberg-Marquardt method for ANFIS learning. In: Smith, MH (Hrsg), *New frontiers in fuzzy logic and soft computing. 1996 biennial conference of the North-American Fuzzy Information Processing Society - NAFIPS, Berkeley, California, USA, June 19 - 22, 1996.* Inst. of Electrical and Electronics Engineers, New York, NY.

[20] Frattale Mascioli, F. M., Varazi, G. M., Martinelli, G. (1997): Constructive algorithm for neuro-fuzzy networks. In: , *Proceedings of the Sixth IEEE International Conference on Fuzzy Systems. Barcelona, Spain, July 1-5, 1997.* IEEE Service Center, Piscataway, NJ.

[21] Dr. Loganathan, C., Girija, K. V. (2013): Hybrid Learning For Adaptive Neuro Fuzzy Inference System. Research Inventy: International Journal Of Engineering And Science, (11):6–13.

[22] Catalao, J. P. S., Pousinho, H. M. I., Mendes, V. M. F. (2010): Hybrid Wavelet-PSO-ANFIS Approach for Short-Term Wind Power Forecasting in Portugal. IEEE Transactions on Sustainable Energy.

[23] Pousinho, H., Mendes, V., Catalão, J. (2012): Short-term electricity prices forecasting in a competitive market by a hybrid PSO–ANFIS approach. International Journal of Electrical Power & Energy Systems, 39(1):29–35.

[24] Jiang, H. M., Kwong, C. K., Ip, W. H., Wong, T. C. (2012): Modeling customer satisfaction for new product development using a PSO-based ANFIS approach. Applied Soft Computing, 12(2):726–734.

[25] Aliyari Shoorehdeli, M., Teshnehlab, M., Sedigh, A. K. (2009): Identification using ANFIS with intelligent hybrid stable learning algorithm approaches. Neural Computing and Applications, 18(2):157–174.

[26] Turki, M., Bouzaida, S., Sakly, A., M'Sahli, F. (2012): Adaptive control of nonlinear system using neuro-fuzzy learning by PSO algorithm. In: , *16th IEEE Mediterranean Electrotechnical Conference (MELECON), 2012. 25 - 28 March 2012, Yasmine Hammamet, Tunisia.* IEEE, Piscataway, NJ.

[27] Aliyari Shoorehdeli, M., Teshnehlab, M., Sedigh, A. K. (2007): Novel Hybrid Learning Algorithms for Tuning ANFIS Parameters Using Adaptive Weighted PSO. Fuzzy Systems Conference, 2007. FUZZ-IEEE 2007. IEEE International:1–6.

[28] Shoorehdeli, M. A., Teshnehlab, M., Sedigh, A. K., Khanesar, M. A. (2009): Identification using ANFIS with intelligent hybrid stable learning algorithm approaches and stability analysis of training methods. Applied Soft Computing, 9(2):833–850.

[29] Shoorehdeli, M. A., Teshnehlab, M., Sedigh, A. K. (2009): Training ANFIS as an identifier with intelligent hybrid stable learning algorithm based on particle swarm optimization and extended Kalman filter. Fuzzy Sets and Systems, 160(7):922–948.

[30] Bagheri, A., Mohammadi Peyhani, H., Akbari, M. (2014): Financial forecasting using ANFIS networks with Quantum-behaved Particle Swarm Optimization. Expert Systems with Applications, 41(14):6235–6250.

[31] Soto, J., Melin, P., Castillo, O. (2014): Optimization of interval type-2 fuzzy integrators in ensembles of ANFIS models for prediction of the Mackey-Glass time series. In: , *IEEE Conference on Norbert Wiener in the 21st Century (21CW)*.

[32] Bagheri, A., Nariman-Zadeh, N., Jamali, A., Dayjoori, K. (2009): Design of ANFIS Networks Using Hybrid Genetic and SVD Method for the Prediction of Coastal Wave Impacts. In: Mehnen, J, Köppen, M, Saad, A, Tiwari, A (Hrsg), *Applications of Soft Computing. From Theory to Praxis.* Springer-Verlag, s.l.

[33] Cardenas, J. J., Garcia, A., Romeral, J. L., Kampouropoulos, K. (2011): Evolutive ANFIS training for energy load profile forecast for an IEMS in an automated factory. In: , *Factory Automation (ETFA 2011)*.

[34] Lutfy, O. F., Noor, S. B. M., Marhaban, M. H. (2011): A simplified adaptive neuro-fuzzy inference system (ANFIS) controller trained by genetic algorithm to control nonlinear multi-input multi-output systems. Scientific Research and Essays, 6(31).

[35] Haznedar, B., Kalinli, A. (2016): Training ANFIS Using Genetic Algorithm for Dynamic Systems Identification. International Journal of Intelligent Systems and Applications in Engineering, (4):44–47.

[36] Chen, M.-S. (12-15 Oct. 1999): A comparative study of learning methods in tuning parameters of fuzzy membership functions. In: , *IEEE SMC'99 Conference Proceedings. 1999 IEEE International Conference on Systems, Man, and Cybernetics*.

[37] Kamarian, S., Yas, M. H., Pourasghar, A., Daghagh, M. (2013): Application of firefly algorithm and ANFIS for optimisation of functionally graded beams. Journal of Experimental & Theoretical Artificial Intelligence, 26(2):197–209.

[38] Orouskhani, M., Mansouri, M., Orouskhani, Y., Teshnehlab, M. (2013): A hybrid method of modified cat swarm optimization and gradient descent algorithm for training ANFIS. International Journal of Computational Intelligence and Applications, 12(02):1350007.

[39] Karaboga, D., Kaya, E. (2013): Training ANFIS using artificial bee colony algorithm. In: , *IEEE International Symposium on Innovations in Intelligent Systems and Applications (INISTA)*.

[40] Salleh, M. N. M., Hussain, K. (2016): A Review of Training Methods of ANFIS for Applications in Business and Economics. International Journal of u- and e- Service, Science and Technology, 9(7):165–172.

[41] Walia, N., Singh, H., Sharma, A. (2015): ANFIS: Adaptive Neuro-Fuzzy Inference System-A Survey. International Journal of Computer Applications, 123(13):32–38.

[42] Jang, J.-S. R. (1995): C code and simulation example for ANFIS. http://www.cs.cmu.edu/Groups/AI/areas/fuzzy/systems/anfis/0.html. Abgerufen am 29.01.2017.

[43] Fresno, C., Fernandez, E. A. (2012): Implementation of ANFIS for R library. http://www.bdmg.com.ar/software/anfis/. Abgerufen am 29.01.2017.

[44] Meggs, T. (2015): Python implementation for adaptive neuro-fuzzy inference system. https://pypi.python.org/pypi/anfis. Abgerufen am 29.01.2017.

[45] Gurov Yury (2016): .NET ANFIS. C# implementation for adaptive neuro-fuzzy inference system. https://github.com/kenoma/adaptive-neuro-fuzzy. Abgerufen am 29.01.2017.

[46] Karlo Knezevic (2013): Java implementation of adaptive neuro-fuzzy inference system. https://github.com/KarloKnezevic/ANFIS. Abgerufen am 29.01.2017.

[47] Sridevi, S., Nirmala, S. (2016): ANFIS based decision support system for prenatal detection of Truncus Arteriosus congenital heart defect. Applied Soft Computing, 46:577–587.

[48] Abushariah, M. A. M., Alqudah, A. A. M., Adwan, O. Y., Yousef, R. M. M. (2014): Automatic Heart Disease Diagnosis System Based on Artificial Neural Network (ANN) and Adaptive Neuro-Fuzzy Inference Systems (ANFIS) Approaches. Journal of Software Engineering and Applications, 07(12):1055–1064.

[49] Amma N.G, B. (2013): An intelligent approach based on Principal Component Analysis and Adaptive Neuro Fuzzy Inference System for predicting the risk of cardiovascular diseases. In: , *Fifth International Conference on Advanced Computing (ICoAC), 2013. 18 - 20 Dec. 2013, Chennai, India.* IEEE, Piscataway, NJ.

[50] Obayya, M. I., Areed, N. F., Abdulhadi, A. O. (2016): Liver Cancer Identification using Adaptive Neuro-Fuzzy Inference System. International Journal of Computer Applications, 140(8):1–7.

[51] Odeh, S. M. (2011): Using an Adaptive Neuro-Fuzzy Inference System (ANFIS) Algorithm for Automatic Diagnosis of Skin Cancer. Journal of Communication and Computer, (8):751–755.

[52] Al-daoud, E. (2010): Cancer Diagnosis Using Modified Fuzzy Network:73–78.

[53] Fernandes, F. C., Brasil, L. M., Lamas, J. M., Guadagnin, R. (2010): Breast cancer image assessment using an adaptative network-based fuzzy inference system. Pattern Recognition and Image Analysis, 20(2):192–200.

[54] Gan Lim, L. A., Maguib, R. N., Dadios, E. P., Avila, J. M. C. (2012): Implementation of GA-KSOM and ANFIS in the classification of colonic histopathological images. In: , *TENCON 2012 IEEE Region 10 Conference.* IEEE.

[55] Loganathan, C., V. Girija, K. (2013): Cancer Classification using Adaptive Neuro Fuzzy Inference System with Runge Kutta Learning. International Journal of Computer Applications, 79(4):46–50.

[56] Shanthakumar, P., Ganeshkumar, P. (2015): Performance analysis of classifier for brain tumor detection and diagnosis. Computers & Electrical Engineering, 45:302–311.

[57] Thirumurugan, P., Shanthakumar, P. (2016): Brain tumor detection and diagnosis using ANFIS classifier. International Journal of Imaging Systems and Technology, 26(2):157–162.

[58] Bhardwaj, S., Singhal, N., Gupta, N. (2014): Adaptive neurofuzzy system for brain tumor. In: , *2014 Innovative Applications of Computational Intelligence on Power, Energy and Controls with their impact on Humanity (CIPECH).* IEEE.

[59] Al-Naami, B., Abu Mallouh, M., Kheshman, A. A. (2014): Automated intelligent diagnostic of alzheimer disease based on neuro-fuzzy system and discrete wavelet transform. Biomedical Engineering: Applications, Basis and Communications, 26(03):1450035.

[60] Ansari, A. Q., Gupta, N. K., Ekata (2012): Automatic Diagnosis of Asthma Using Neurofuzzy System. In: , *2012 Fourth International Conference on Computational Intelligence and Communication Networks.* IEEE.

[61] Übeyli, E. D. (2010): Automatic diagnosis of diabetes using adaptive neuro-fuzzy inference systems. Expert Systems, 27(4):259–266.

[62] Karahoca, A., Karahoca, D., Kara, A. (2010): Diagnosis of diabetes by using adaptive neuro fuzzy inference systems. In: , *2009 Fifth International Conference on Soft Computing, Computing with Words and Perceptions in System Analysis, Decision and Control (ICSCCW 2009). Famagusta, Cyprus, 2 - 4 September 2009.* IEEE, Piscataway, NJ.

[63] Adeli, M., Bigdeli, N., Afshar, K. (2013): New hybrid hepatitis diagnosis system based on Genetic algorithm and adaptive network fuzzy inference system. In: , *2013 21st Iranian Conference on Electrical Engineering (ICEE). 14 - 16 May 2013, Mashhad, Iran.* IEEE, Piscataway, NJ.

[64] Appiah, R., Kobina Panford, J., Riverson, K. (2015): Implementation of Adaptive Neuro Fuzzy Inference System for Malaria Diagnosis (Case Study Kwesimintsim Polyclinic). International Journal of Computer Applications, 115(7):33–37.

[65] Lin, C.-L., Hsieh, S.-T. (2015): Work-In-Progress: An intelligent diagnosis influenza system based on adaptive neuro-fuzzy inference system. In: Cecati, C, Guo, S, Shu, L (Hrsg), *Proceedings of the 2015 1st International Conference on Industrial Networks and Intelligent Systems. INISCom 2015 : Tokyo, Japan, March 2-4, 2015.* IEEE, Piscataway, NJ.

[66] Fazeli, S., Naghibolhosseini, M., Bahrami, F. (2008): An Adaptive Neuro-Fuzzy Inference System for Diagnosis of Aphasia. In: , *The 2nd International Conference on Bioinformatics and Biomedical Engineering (iCBBE 2008). May 16 - 18, 2008, Shanghai, China.* IEEE Operations Center, Piscataway, NJ.

[67] Garg, V. K., Bansal, R. K. (2015): Soft computing technique based on ANFIS for the early detection of sleep disorders. In: , *International Conference on Advances in Computer Engineering and Applications (ICACEA), 2015. 19 - 20 March 2015, Ghaziabad, India ; conference proceeding.* IEEE, Piscataway, NJ.

[68] Al-Hmouz, A., Shen, J., Al-Hmouz, R., Yan, J. (2012): Modeling and Simulation of an Adaptive Neuro-Fuzzy Inference System (ANFIS) for Mobile Learning. IEEE Transactions on Learning Technologies, 5(3):226–237.

[69] Jang, J.-S. R. (1996): Input selection for ANFIS learning. In: , *IEEE 5th International Fuzzy Systems*.

[70] Jang, J.-S. R. (1996): Neuro-fuzzy modeling for dynamic system identification. In: , *Soft Computing in Intelligent Systems and Information Processing. 1996 Asian Fuzzy Systems Symposium*.

[71] Chai, T., Draxler, R. R. (2014): Root mean square error (RMSE) or mean absolute error (MAE)? – Arguments against avoiding RMSE in the literature. Geoscientific Model Development, 7(3):1247–1250.

[72] Witten, I. H., Frank, E., Hall, M. A., Pal, C. J. (2017): Data mining. Practical machine learning tools and techniques. Morgan Kaufmann/Elsevier, Amsterdam, Boston, Heidelberg.

[73] Ohne Verfasser - World Health Organization (WHO) (2017): Cancer - Fact sheet. http://www.who.int/mediacentre/factsheets/fs297/en. Abgerufen am 20.02.2017.

[74] Barnes, B., Bertz, J., Buttmann-Schweiger, N., Fiebig, J., Jordan, S. et al. (2016): Bericht zum Krebsgeschehen in Deutschland 2016, Berlin.

[75] Dr. Wolberg, W. H., Mangasarian, O. L.: Original Wisconsin Breast Cancer Database. https://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin. Abgerufen am 28.09.2016.

[76] Jang, J.-S. R.: Frequently Asked Questions - ANFIS in the Fuzzy Logic Toolbox. http://www.cs.nthu.edu.tw/~jang/anfisfaq.htm#4. Abgerufen am 04.02.2017.

Versicherung über Selbstständigkeit

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne fremde Hilfe selbstständig verfasst und nur die angegebenen Hilfsmittel benutzt habe.

Hamburg, den _____ _____