Erik Schaffernicht

Lernbeiträge im Rahmen einer kognitiven Architektur für die intelligente Prozessführung

Lernbeiträge im Rahmen einer kognitiven Architektur für die intelligente Prozessführung

Erik Schaffernicht

Universitätsverlag Ilmenau
2012

Impressum

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Angaben sind im Internet über http://dnb.d-nb.de abrufbar.

Diese Arbeit hat der Fakultät für Informatik und Automatisierung der Technischen Universität Ilmenau als Dissertation vorgelegen.

Tag der Einreichung:	20. Dezember 2011
1. Gutachter:	Univ.-Prof. Dr.-Ing. Horst-Michael Groß (Technische Universität Ilmenau)
2. Gutachter:	Prof. Dr. Martin Riedmiller (Albert-Ludwigs-Universität Freiburg)
3. Gutachter:	Prof. Dr. rer. nat. habil. Thomas Villmann (Hochschule Mittweida)
Tag der Verteidigung:	13. Juni 2012

Technische Universität Ilmenau/Universitätsbibliothek
Universitätsverlag Ilmenau
Postfach 10 05 65
98684 Ilmenau
www.tu-ilmenau.de/universitaetsverlag

Herstellung und Auslieferung
Verlagshaus Monsenstein und Vannerdat OHG
Am Hawerkamp 31
48155 Münster
www.mv-verlag.de

ISBN 978-3-86360-043-3 (Druckausgabe)
URN urn:nbn:de:gbv:ilm1-2012000276

Titelfoto: photocase.com

Kurzbeschreibung

In dieser Arbeit werden wichtige Aspekte einer kognitiven Architektur für das Erlernen von Regelungsaufgaben beleuchtet. Dabei geht es primär um die Merkmalsextraktion, das Reinforcement Learning und das Lernmanagement im Rahmen des Wahrnehmungs-Handlungs-Zyklus.

Für die Merkmalsextraktion werden dabei mit Hilfe informationstheoretischer Größen, wie der Transinformation, neue hybride Merkmalsextraktionsverfahren vorgestellt. Neuartig ist dabei der Ansatz, Merkmale zu suchen, die explizit mit den gemachten Fehlern eines lernenden Systems verknüpft sind. Es wird gezeigt, dass diese residuumsbasierten Ansätze klassischen Methoden überlegen sind. Es wird ebenfalls untersucht, welche Schätzverfahren für die Bestimmung der Transinformation im Sinne der Merkmalsextraktion geeignet sind.

Als Entscheidungsinstanz der Gesamtarchitektur werden aktuelle Reinforcement Learning Verfahren auf ihre Eignung für komplexe Anwendungen hin untersucht. Dabei wird auch auf Probleme des Lernmanagements, wie das Explorations-Exploitations-Dilemma, das Stabilitäts-Plastizitäts-Dilemma und das Rewarddekompositionsproblem eingegangen. Neue Beiträge werden dabei in Form des Diffusionsbaumbasiertes Reinforcement Learning und des SMILE-Algorithmus geliefert. Ebenso wird eine Architekturerweiterung zum Organisieren der Lernprozesse vorgeschlagen, welche im Kern um eine Prozesskarte angeordnet ist.

Der experimentelle Nachweis, dass das vorgestellte System die Lösung für reale Probleme erlernen kann, wird am herausfordernden Szenario der intelligenten Feuerungsführung erbracht. Dabei wird das Gesamtsystem zur Regelung eines mit Steinkohle gefeuerten Kraftwerks eingesetzt, wobei Ergebnisse erzielt werden, die bisher existierende Systeme und auch menschliche Experten übertreffen.

Abstract

In this thesis, important aspects of a cognitive architecture for learning control tasks are discussed. Highlighted are the topics of feature extraction, reinforcement learning and learning management in the context of the perception-action-cycle.

The contributions in the field of feature extraction utilize information-theoretic measures such as mutual information to formulate new hybrid feature extraction algorithms. Finding features that are explicitly linked with the errors made by a learning system are the focus. It is shown this approach based on residuals is superior to classical methods. Another topic of interest is the estimation of mutual information in the context of feature extraction.

State of the art reinforcement learning methods are investigated for their suitability for challenging applications. This work addresses issues of learning management, such as the exploration-exploitation dilemma, the plasticity-stability dilemma and the reward decomposition problem. New contributions are made in the form of the diffusion tree-based reinforcement learning algorithm and the SMILE approach. Likewise, an architectural extension is proposed to organize the learning process. It uses a process map as the core piece to achieve this organization.

Experimental evidence that the proposed system can learn the solution to real problems are demonstrated in the challenging scenario of intelligent combustion control. The system is used to learn a control strategy in a coal-fired power plant. The achieved results surpass existing systems and human experts.

Inhaltsverzeichnis

1. Einleitung

Als im Februar 1996 Garry Kasparov ein Schachspiel verlor, ging diese Nachricht um die Welt. Es war nicht irgendein Schachspiel. Zum ersten Mal hatte ein Computer, Deep Blue, gegen einen amtierenden Schachweltmeister gewonnen. Ein Jahr später gewann Deep Blue, gar ein ganzes Match. Ein Computer, der bei etwas so Komplexen, wie Schach, den Mensch schlagen konnte.

War damit der Informatik das Nachbilden und das Übertreffen menschlicher Intelligenz gelungen?

Über die korrekte Antwort kann sicherlich gestritten werden, und das wird es auch, je nach Disziplin mit sehr unterschiedlichen Erklärungen. Für diese Arbeit sollten aus dieser Diskussion die folgenden Argumente in Betracht gezogen werden.

Der Computer war in der Lage, das Problem der optimalen Züge auf dem Schachbrett besser zu lösen als Kasparov. Für alles andere hatte Deep Blue menschliche Helfer. Das Bewegen der Figuren wurde von einem Menschen durchgeführt, die Züge von Kasparov wurden von einem Menschen wahrgenommen und in eine für Deep Blue verständliche Form übersetzt. Dinge die Kasparow allein erledigt hatte, Deep Blue aber überfordert hätten.

Von diesen Aspekten, dem Wahrnehmen, dem Planen und dem Handeln, hat Deep Blue Kasparov im Planungsaspekt geschlagen. Das ist sicher ein wichtiger Schritt, aber für ein wirklich intelligentes System kann man nicht einzelne Teile losgelöst voneinander betrachten.

Natürlich ging die Entwicklung weiter. Mittlerweile fahren autonome Autos erfolgreich durch Wüsten und Städte, Roboter helfen beim Arbeiten und Einkauf, die heimischen vier Wände werden zu Smarthomes und komplexe Prozesse in der Industrie werden automatisch geregelt - die Technik um uns herum wird klüger, intelligenter. Sie übertrifft dabei zum Teil den Menschen, wenn auch bisher nur in engen Grenzen. In den meisten Fällen ist das Ziel, dem Menschen zu helfen und das Leben einfacher, bequemer und sicherer zu machen, oder vielleicht auch überhaupt möglich zu machen, ohne dabei unnötigen Aufwand zu verursachen.

Um diese Systeme alltagstauglich nutzen zu können, müssen sie nicht nur einen bestimmten Aspekt lösen, wie Deep Blue es tat, sondern ein Gesamtsystem realisieren, welches vom Wahrnehmen über das Entscheiden zum Handeln alle wichtigen Aspekte selbst löst.

Ein solches Gesamtsystem ist Thema dieser Arbeit.

Entstanden ist diese Dissertation im Rahmen des SOFCOM-Projektes im Fachgebiet für Neuroinformatik und Kognitive Robotik der Technischen Universität Ilmenau. Das Akronym SOFCOM steht dabei für *Selbst-Optimierende Feuerungsführung zur CO2-Emissions-Minderung in Großindustriellen Kohlekraftwerken*. In diesem Projekt geht es um die Optimierung von Verbrennungsprozessen mit Hilfe eines intelligenten Systems.

Es wird diskutiert, inwieweit sich ein solches System als *kognitive Architektur* interpretieren lässt und die benannten Aspekte des Wahrnehmens, Planens und Handelns sich darauf abbilden lassen. In diesem Rahmen werden Beiträge zum Lernen auf den Feldern der automatischen Merkmalsextraktion, dem Reinforcement Learning und der Adaptivität des Gesamtsystems vorgestellt. Die Funktionalität dieser Gesamtarchitektur wird dabei an einem komplexen, herausfordernden Beispiel, der Regelung einer Kohleverbrennung in einem Kraftwerk demonstriert.

Die Arbeit wird dabei auf die folgenden Schwerpunkte eingehen:

- **Kognitive Architektur**

  Die grundlegende Funktionalität zur Realisierung eines intelligenten Systems zur Problemlösung wird dabei durch eine kognitive Architektur bereitgestellt, in welche das notwenige Wissen durch Expertenvorgaben oder Lernprozesse eingekoppelt wird. Im Rahmen dieser Arbeit wird der subsymbolische, datengetriebene Wissenserwerb im Rahmen des Wahrnehmungs-Handlungs-Zyklus für Automatisierungsaufgaben betrachtet - dabei wird nicht aus einer biologisch orientierten Herangehensweise gehandelt, sondern die ingenieurtechnische Perspektive steht im Mittelpunkt. Dieser einschränkende, spezielle Blickwinkel auf die Problematik steht dabei nicht im Widerspruch zum Ziel eines Gesamtsystems, sondern stellt eine mögliche Herangehensweise dar.

  Die zu beantwortenden Kernfragen für eine solche Architektur sind dabei: Welche Informationen sind wichtig? Wie kann ein optimales Verhalten effizient erlernt werden? Wie organisiert man Lernprozesse und Wissen geschickt, um lebenslang lernfähig zu bleiben?

  Im Rahmen der Arbeit werden in einzelnen Teilbereich auch immer Probleme und Einschränkungen benannt, die durch die gewählte Architektur und die Methoden nicht zu beherrschen sind. Auf Basis dieser Erkenntnisse wird diskutiert werden, welche Konsequenzen für eine zukünftige, weiterentwickelte Architektur zu ziehen sind.

- **Automatische Merkmalsextraktion**

  Der erste bedeutsame Block in der Verarbeitung durch ein intelligentes System ist die Wahrnehmung der Umwelt. Die Menge an verfügbaren Daten ist für reale Probleme oftmals wesentlich größer als die Menge an Informationen, die in den Daten enthalten ist. Daher ist es von essentieller Bedeutung, dafür zu sorgen, dass die Entscheidungsinstanzen innerhalb der kognitiven Architektur nur informa-

tive Daten bekommen. Dies wird mittels der Merkmalsextraktion[1] umgesetzt, wobei im Rahmen dieser Arbeit vorrangig auf informationstheoretische Konzepte zurückgegriffen wird. Die Transinformation und ihrer Bestimmung aus den Daten ist dabei von zentraler Bedeutung. Die Kombination der Transinformation mit Filter- und Wrapper-Verfahren führt zu einer effizienten Beurteilung von Eingangskanälen. Mit Einschränkungen lassen sich die Verfahren auch zur Auswahl von relevanten Aktionen nutzen und schließen somit den Zyklus durch die Ausführung einer Aktion. Der Sinn und Nutzen der Merkmalsextraktion wird dabei für verschiedene Anwendungen beispielhaft gezeigt.

- **Reinforcement Learning**

  Bei der eigentlichen Planung und Entscheidungsfindung, manchmal auch als Aktionsauswahl bezeichnet, steht das Reinforcement Learning im Mittelpunkt. Dabei werden verschiedene aktuelle Verfahren untersucht, verglichen und zum Teil erweitert, um speziell mit dynamischen und hochdimensionalen Problemen, die nur unvollständig und verrauscht beobachtbar sind, umgehen zu können. Diese werden auch anderen Alternativen, wie z.B. der klassischen Regelungstechnik (MPC) oder probabilistischen Ansätzen (BPC), gegenübergestellt.

- **Lernmanagement**

  Da sich die zu regelnden Prozesse mit der Zeit in ihrer Charakteristik verändern können, ist es notwendig, Mechanismen zu realisieren, die ein Adaptieren an die neue Situation erlauben. Dazu müssen bekannte Probleme, wie das Stabilitäts-Plastizitäts-Dilemma oder das Exploration-Exploitation-Dilemma, behandelt werden. Hierzu werden Beobachtungen und Erkenntnisse präsentiert, die eine sinnvolle Organisation von Lernprozessen und Wissensrepräsentation er-

[1] Der Begriff wird hier im Sinne der Signifikanzanalyse als Überbegriff für die automatische Auswahl und Transformation von relevanten Eingangsvariablen verwendet.

leichtern sollen. Diese Fragestellung steht außerhalb des eigentlichen Wahrnehmungs-Handlungs-Zyklus und beeinflusst das System auf einer anderen Zeitskala.

- **Intelligente Feuerungsführung**

  Die Funktionalität des Gesamtkonzepts soll dabei an einem komplexen, herausfordernden Szenario, der intelligenten Führung großtechnischer Feuerungsprozesse, gezeigt werden. Die Anforderungen in einem solchen Anwendungsfeld sind vielzählig und werden im Folgenden genauer vorgestellt. Diese Arbeit stellt dabei die Lösung dieses Ingenieurtechnischen Problem nicht in den Mittelpunkt, sondern nutzt es als herausfordernden Demonstrator.

1.1. Anspruch der Arbeit

Schwerpunkt dieser Arbeit ist eine Architektur, welche in der Lage ist, herausfordernde regelungstechnische Probleme zu lösen. Dazu lernt das System basierend auf Beobachtungen die Lösung selbstständig und passt diese an Änderungen im Prozess an.

Im Bereich der Merkmalsextraktion, welcher auch das Kernstück der Arbeit darstellt, werden neue Algorithmen vorgestellt und untersucht, die Vorteile gegenüber existierenden Ansätzen bieten. Die Untersuchungen im Bereich des Reinforcement Learnings hingegen zielen darauf ab, aktuelle Verfahren aus diesem Feld miteinander unter verschiedenen Gesichtspunkten zu vergleichen und daraus eine Entscheidung über deren Nutzbarkeit unter den gegebenen Umständen abzuleiten. Die Verfahren aus der Merkmalsextraktion und die Reinforcement Learning Ansätze werden dann daraufhin untersucht, inwieweit sich Wissen wiederverwenden lässt oder ob es sinnvoller ist, bei Änderungen komplett neu zu lernen. Zusätzlich wird ein neuer Algorithmus vorgestellt, der für kontinuierliche Aktionsräume eine sinnvolle Erkundungsstrategie

liefert. Auch auf das Problem der Rewarddekomposition wird eingegangen.

Schließlich wird im Sinne eines erweiterten Ausblicks aufgezeigt, wie aus Sicht des Autors eine Weiterentwicklung der Architektur aussehen könnte und welche Aspekte dabei im Mittelpunkt stehen sollten.

Im Anwendungsszenario der industriellen Feuerungsführung wird nicht nur die Funktionsweise des Gesamtsystems demonstriert, sondern damit auch eine fortschrittliches System zur Wirkungsgradsteigerung und Emissionsminderung bei der Kohleverbrennung vorgestellt, welches auch im Kontext der aktuellen Klimaschutzdebatte ein wichtiger Beitrag ist.

1.2. Szenario

Als Demonstrator für das in dieser Arbeit vorgestellte System dient die Regelung eines industriellen Steinkohleofens im Kraftwerk Tiefstack in Hamburg. Das Kraftwerk dient der Strom- und Fernwärmeerzeugung.

Das entwickelte System wird zur Regelung der Verbrennung eingesetzt. Dabei wird gemahlene Kohle in den Ofen geblasen und entzündet. Die stattfindende exotherme Reaktion der Umwandlung von Kohlenstoff und Sauerstoff in Kohlendioxid setzt dabei die Energie frei, die die Turbine des Kraftwerks antreibt. Die kontinuierliche Zufuhr der Kohle erfolgt typischerweise aus einem Silo über eine Kohlemühle. Die Menge der zugeführten Kohle wird dabei durch den momentanen Energiebedarf bestimmt und ist in diesem Szenario gegeben. Damit verbleibt die Luft als Aktionsgröße um die Verbrennung zu beeinflussen. Das beinhaltet nicht nur die Gesamtmenge der Luft, welche in direktem Zusammenhang mit dem Wirkungsgrad, der Korrosion des Ofens und der Kohlenmonoxidbildung steht, sondern auch die Verteilung der Luft im Ofen. Informell könnte man sagen, dass die Luft dort sein muss,

wo unverbrannter Kohlenstaub im Kessel ist. Dazu existieren Klappen an verschiedenen Stellen des Kessels, mit denen die Luft in den Ofen gebracht wird.

Diese Klappen befinden sich typischerweise in einer Standardeinstellung, die im Mittel für eine theoretisch günstige Verteilung der Luft sorgen sollte und werden im Normalbetrieb nicht verändert. Das liegt nicht daran, dass nicht bekannt wäre, dass eine sinnvolle Luftverteilung vorteilhaft für die Verbrennung wäre, sondern vielmehr darin begründet, dass für eine Regelung dieser Klappen kein ausreichendes Expertenwissen vorhanden ist, und es sich schwierig gestaltet, Führungsgrößen abzuleiten.

Die Verbrennung in einem 30 Meter hohen Ofen ist ein vergleichsweise chaotischer Prozess. Physikalische Modelle stoßen bei dem Versuch diesen zu beschreiben an ihre Grenzen. Aufgrund der herrschenden Temperatur und der Verschmutzung sind die notwendigen Messgrößen nur schwer oder gar nicht ermittelbar. Daher besteht meist nur das Bestreben, die Verbrennung so zu betreiben, dass die Wärme und Energie erzeugt werden, die Grenzwerte nicht verletzt werden und eine direkte Gefährdung von Mensch und Umwelt ausgeschlossen ist. Dieses Ziel wird mit den Standardeinstellungen erreicht. Die Suche nach einer optimalen Regelung bleibt somit eine große Herausforderung.

An dieser Stelle setzt die in dieser Arbeit vorgestellte Architektur an. Basierend auf Beobachtungen soll gelernt werden, wie der Prozess besser geregelt werden kann.

Die folgenden Eigenschaften charakterisieren den Prozess näher:

- Die Beobachtungen (z.B. Flammenbilder) sind durch den Menschen aufgrund fehlenden Expertenwissens schwer zu bewerten.
- Die Beobachtungen sind mit einer großen Unsicherheit belegt. Sensorrauschen und fehlerhafte Messungen durch Verschmutzungen sind eher die Regel als eine Ausnahme.

- Es stehen riesige Datenmenge von vielen Messstellen in einem hochdimensionalen Raum zur Verfügung. Jedoch ist häufig unklar, inwieweit die entsprechenden Messungen hilfreiche Informationen für die Lösung des Problems enthalten.
- Etliche wichtige Prozessgrößen können nicht direkt oder nur punktförmig gemessen werden. Dies liegt an den Kosten für die Sensorik, an der heißen und schmutzigen Einsatzumgebung, die herkömmliche Lösungen für einen längerfristigen Einsatz scheitern lassen oder daran dass die notwendigen Messeinrichtungen den Prozess selbst negativ beeinflussen würden. Damit ergibt sich eine Menge von versteckten Prozessgrößen.
- Die Ziele einer Optimierung der Feuerung sind teilweise konträr zueinander. Es handelt sich eigentlich um Multikriterien-Optimierungsproblem.

Warum wird dieses Szenario betrachtet?

- Es ist ein reales Problem. Natürlich lassen sich Algorithmen und Architekturen auch auf Spielbeispielen und Simulationen testen und bewerten. Allerdings vereinfachen solche Modelle auch immer gewisse Teile des Problems. In der Realität gibt es solche Vereinfachungen nicht und somit verkompliziert sich die Gesamtaufgabe zusehends. Ziel für das hier vorgestellte System ist der Einsatz für reale Anwendungen.
- Es ist eine Herausforderung. Neben den oben auflisteten Eigenschaften des Problems ist auch anzumerken, dass es, aufgrund der Schwierigkeit und Komplexität, kaum Lösungen für dieses Problem gibt.
- Eine erfolgreiche Lösung für dieses Problem hat ökologischen und ökonomischen Nutzen. Eine Erhöhung des Wirkungsgrads und Verringerung der Schadstoffe dient dem Umweltschutz. Gerade vor dem Hintergrund der Klimaschutzziele und des Atomausstiegs sind die Ergebnisse von hohem gesellschaftlichem Interesse.

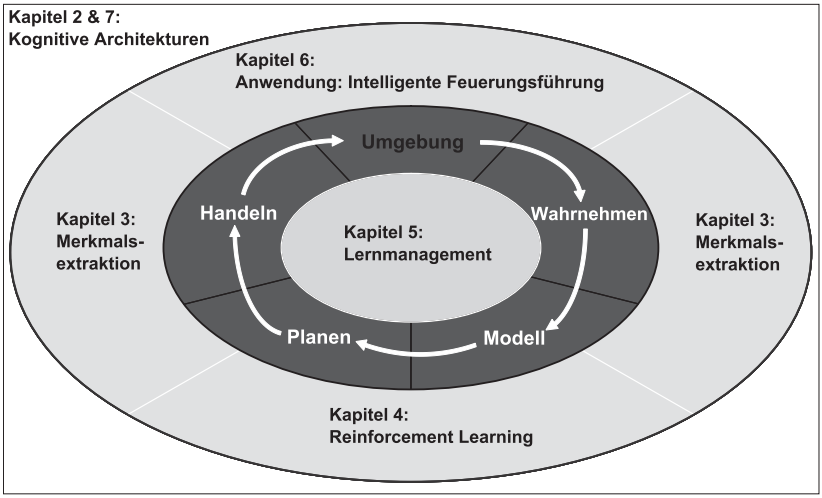


Abbildung 1.1.: Gesamtübersicht für diese Arbeit. Der hier dargestellte Wahrnehmungs-Handlungs-Zyklus wird durch eine kognitive Architektur realisiert, deren einzelne Schwerpunkte in den nachfolgenden Kapiteln wie in der Grafik gezeigt, diskutiert werden.

1.3. Gliederung und Leseleitfäden

Für den Leser ergeben sich mehrere sinnvolle Wege, sich diese Arbeit ganz oder in Teilen zu erschließen. Die Strukturierung der Arbeit ist in Abbildung 1.1 zu sehen.

Der geradlinige Weg führt von der als Klammer dienenden Diskussion kognitiver Architekturen in Kapitel 2, über die Methoden der Merkmalsextraktion in Kapitel 3 als erste Stufe in einem Wahrnehmungs-Handlungs-Zyklus, hin zu Reinforcement Learning Methoden als Entscheidungsfinder eines kognitiven Systems im Kapitel 4. Die Organisation des Lernens im Gesamtsystem ist Thema des 5. Kapitels. Abschließend wird die Anwendung der vorgestellten Konzepte im Rahmen der Regelung eines Kohlekraftwerks diskutiert. Mit all den gewonnenen

Erkenntnissen werden dann in Kapitel 7 Erweiterungen und Weiterentwicklungen für die Gesamtarchitektur als auch einzelne Teilsysteme skizziert, deren Umsetzung jedoch über die vorliegende Arbeit hinausgehen würde.

Die Kapitel 2, 3 und 4 lassen sich jeweils auch einzeln weitestgehend ohne die andere Abschnitte erschließen. Die Konzepte, die dort vorgestellt werden, sind so abstrakt dargestellt, dass sie auch ohne den Kontext der kognitiven Gesamtarchitektur oder des Anwendungsszenarios genutzt werden können. Im speziellen betrifft dies die Merkmalsextraktionsverfahren in Kapitel 3. Diese können alternativ vollkommen losgelöst vom Rest der Arbeit betrachtet werden. Die Diskussion des Lernmanagements hingegen ergibt nur mit den Kapiteln 3 und 4 zusammen Sinn, da dort regelmäßige Bezüge hergestellt werden. Auch die Erweiterungen der Architektur in Kapitel 7 erschließen sich nicht allein, da hier die Konsequenzen aus allen vorangegangenen Kapiteln diskutiert werden.

Schließlich ergibt sich für den praktisch veranlagten Leser die Möglichkeit, die Anwendung in den Mittelpunkt zu stellen. Dazu kann nach dieser Einleitung zu Kapitel 6 gesprungen werden. Von der Beschreibung der Problematik im Kraftwerk und dem entwickelten Lösungsansatz kann dann an den entsprechenden Stellen in die vorhergehenden Kapitel zurückgeblättert werden, um die Details der Lösung zu ergründen.

2. Kognitive Architekturen

Viele Arbeiten im Bereich der künstlichen Intelligenz und des maschinellen Lernens beschäftigen sich mit wichtigen Teilproblemen, wie es bereits in der Einführung motiviert wurde. Jedoch ergeben viele Einzelteile noch keine Gesamtlösung. Will man ein Problem, wie die Regelung eines komplexen Prozesses, lösen, müssen die Teilkonzepte im Rahmen eines Systems zusammenarbeiten. Die Architektur der Gesamtlösung muss demnach das harmonische Miteinander aller Komponenten zur Lösung der gestellten Aufgabe realisieren. Denn einige Schwierigkeiten ergeben sich erst durch das Zusammenspiel der Komponenten miteinander und würden nicht betrachtet werden, wenn man die Teilprobleme alle losgelöst betrachtet.

Da es für ein nutzbares intelligentes Gesamtsystem ein solches Zusammenspiel jedoch unerlässlich ist, wird in diesem Kapitel diskutiert, welche Komponenten, Eigenschaften und Funktionen eine kognitive Architektur im Kontext bestimmter Aufgaben haben muss. Dazu wird der Frage nachgegangen, was eine kognitive Architektur ist, in welche Klassen sie typischerweise eingeteilt werden und welche Umsetzungen in der Literatur existieren. Im Zusammenhang mit der Zielstellung dieser Arbeit, wird dann untersucht, welche Architekturen speziell im Feld der Automatisierung von Interesse sind.

Folgt man der Veröffentlichung von [Langley et al., 2009], welche eine gute Übersicht über aktuelle Fragen und Probleme im Bereich der Kognitiven Architekturen gibt, kann man eine solche Architektur wie folgt definieren:

Definition 2.1

KOGNITIVE ARCHITEKTUR

Eine kognitive Architektur stellt die grundlegende Funktionalität für die Realisierung eines intelligenten Systems bereit.

Andere Definitionen (z.B. [MATARIC und MICHAUD, 2008] oder [ARKIN, 1998]) sprechen davon, dass eine solche Architektur eines intelligenten Systems eine Ordnung von Komponenten und Interaktionen zwischen diesen formuliert. Dabei beschränkt eine solche Ordnung die Möglichkeiten, wie ein solches System die Problemlösung angehen kann.

Die Architektur bildet einen Rahmen mit elementaren Fähigkeiten intelligenter Agenten und Mechanismen zur Repräsentation und Verarbeitung von Wissen. Die zugrundeliegende Infrastruktur dieses intelligenten Systems besteht dabei aus jenen Elementen, die in unterschiedlichen Anwendungsszenarien und über die Zeit hinweg gleichbleiben. [LANGLEY et al., 2009] zählt dazu folgende Funktionalitäten und Elemente:

- Kurz- und Langzeitgedächtnis zur Speicherung von Wissen
- Repräsentationsform von Wissen innerhalb der Gedächtnisstrukturen
- Funktionen, die über diesen Strukturen definiert sind (z.B. Lernalgorithmen oder Anwendung von Wissen)

Das eigentliche Wissen, also der Inhalt des Gedächtnisses, wird nicht durch die Architektur definiert, sondern muss applikationsspezifisch erworben werden. Dadurch ergibt sich ein flexibles Konzept, mit welchem kognitive Architekturen auf eine breite Zahl von Anwendungen hinzielen. In der Literatur werden sie mitunter als Gegenstück zu den Expertensystemen bezeichnet, welche bei ihrem Design immer auch das konkrete Wissen mit einbeziehen.

Man sollte beachten, dass viele intelligente Systeme, die nicht explizit eine kognitive Architektur beschreiben, oftmals als eine solche interpretiert werden können.

Ein oft genutztes Unterscheidungsmerkmal kognitiver Architekturen ist dabei die Repräsentationsform des Wissens [Langley et al., 2009]. Man differenziert zwischen der symbolischen und der subsymbolischen Wissensrepräsentation. Symbolisches Wissen ist typischerweise eng mit den klassischen Methoden der künstlichen Intelligenz verknüpft - es werden Symbole, Ausprägungen und Operationen über den Symbolen definiert, die beispielsweise in sogenannten Ontologien [Uschold und Grüninger, 1996] repräsentiert und mittels logischer Programmiersprachen manipuliert werden können. Dies Darstellungsform entspricht der "natürlichen" Form von Wissen, die auch vom Menschen genutzt wird. Sie wird oft in kognitiven Architekturen verwendet, die eine dem Menschen analoge Wissensverarbeitung simulieren und implizieren oftmals einen Top-Down Ansatz.

Subsymbolische, oder auch konnektionistische, Ansätze hingegen setzten auf eine verteilte Repräsentation und arbeiten auf sich aus den Daten ergebenden Mustern. Mit vergleichsweise einfachen Verarbeitungsregeln setzen diese subsymbolischen Repräsentationen oftmals biologisch inspirierte Ideen um, die der Neuronen und Synapsenstruktur im Gehirn angenähert ist. Die datengetriebene Wissenakquisition impliziert einen Bottom-Up Ansatz.

Natürlich ist es oftmals nicht möglich und auch nicht erwünscht, symbolisches und subsymbolisches Wissen strikt zu trennen. Damit ergeben sich hybride Wissensrepräsentationen als Mischformen.

Die Art der Entscheidungsfindung wird ebenfalls als Unterscheidungsmerkmal genutzt. Dabei wird zwischen reaktiven [Kortenkamp und Simmons, 2008] und deliberativen [Mataric und Michaud, 2008] Ansätzen unterschieden. Reaktiv bedeutet eine einfache Sensor-Aktor Kopplung nach dem aus der

Biologie bekannten Reiz-Reaktion-Modell, was typischerweise sehr schnelle Aktionen des Systems zulässt. Deliberativ hingegen beinhaltet das Einschätzen der Situation und die Entwicklung oder Anwendung eines Plans zur Problemlösung. Es ist eng mit dem sogenannten Sense-Plan-Act Paradigma verbunden, welches das Problem funktionsorientiert angeht. Praktisch ist auch hier eine klare Trennung oftmals nicht möglich und man erhält hybride Mischformen, bei denen beispielsweise die deliberative Ebene dafür zuständig ist, verschiedene Verhaltensmuster zu aktivieren, nach denen auf der reaktiven Ebene gehandelt wird. In der Robotik findet sich daneben noch ein Konzept, welches auf Verhaltensmustern (engl. *behaviour*) basiert [Mataric und Michaud, 2008]. Im Gegensatz zu den bisherigen Ansätzen wird dabei auf eine verteilte Entscheidungsfindung realisiert. Parallel existierende Verhaltensmuster, welche einzelne Teilprobleme lösen und meist durch Expertenwissen zu definieren sind, werden durch Interaktionen untereinander zu einem Gesamtsystem verwoben.

Funktionen kognitiver Architekturen

Neben der Repräsentationsform von Wissen in einer Architektur ist natürlich auch die Nutzung dieses Wissens von zentraler Bedeutung. Dabei steht der Wahrnehmungs-Handlungs-Zyklus im Mittelpunkt. Im ersten Schritt wird mittels der Sensorik die Umwelt wahrgenommen. Basierend auf diesen Beobachtungen und dem internen Wissen (z.B. in Form eines Modells) wird ein Plan formuliert, der zu einer Aktion führt. Diese Aktionen beeinflusst wiederum die Umwelt des intelligenten Systems. Diese Abfolge wiederholt sich zyklisch, wobei das System über die Wahl der richtigen Aktionen seine Ziele erfüllt. In Abbildung 2.1 ist ein gegenüber Kapitel 1 erweiterter Wahrnehmungs-Handlungs-Zyklus dargestellt, der versucht einen möglichst umfassenden Überblick über die Aufgaben und Funktionen einer kognitiven Architektur zu geben.

Der innere Ring der Darstellung entspricht dabei dem Wahrnehmungs-

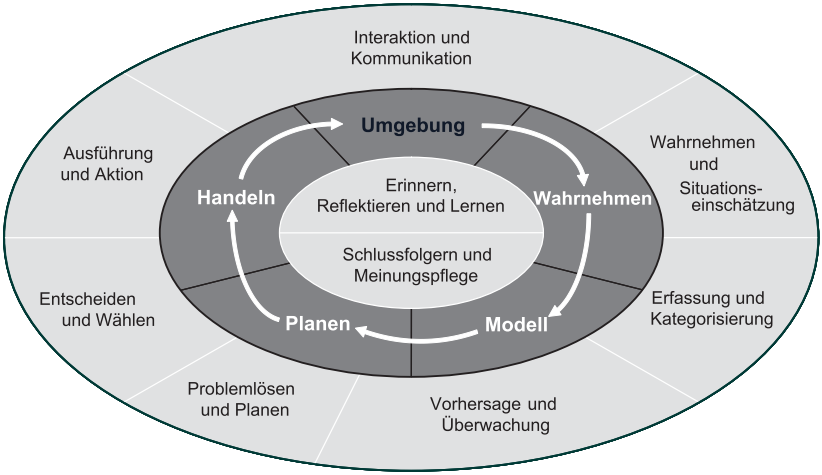


Abbildung 2.1.: Erweiterter Wahrnehmungs-Handlungs-Zyklus im Rahmen von kognitiven Architekturen. Der mittlere Ring stellt dabei den grundlegenden Wahrnehmungs-Handlungs-Zyklus dar. Die Umwelt wird mittels wie auch immer gearteter Sensorik wahrgenommen. Diese Beobachtungen werden dann zum Planen genutzt, wozu ein Modell zum Einsatz kommen kann. Basierend auf dem Plan wird eine Handlung ausgeführt, die die Umgebung beeinflusst. Dies wird wieder beobachtet und der Zyklus beginnt von neuem. Der äußere Ring hingegen beschreibt detaillierter die Aufgaben, die sich für eine kognitive Architektur direkt aus diesem Zyklus ergeben. Der Kern der Darstellung beschreibt Aufgaben, die nur indirekt auf den Wahrnehmungs-Handlungs-Zyklus abbildbar sind, sondern es wird das interne Wissensmanagement der Architektur beschreiben.

Handlungs-Zyklus, der äußere Ring und der Kern der Darstellung hingegen sind die Fähigkeiten und Aufgaben die [Langley et al., 2009] einer kognitiven Architektur zuweist. Im äußeren Ring sind die Fähigkeiten aufgelistet, die direkt auf den Wahrnehmungs-Handlungs-Zyklus abbildbar sind.

Dies sind:

- **Wahrnehmen und Situationseinschätzung**
  Ein Agent muss seine Umwelt mittels seiner Sensorik wahrnehmen. Das können einfache Punktmessungen, wie sie von einem Druck- oder Sonarsensor stammen, sein. Auch komplexere Messungen wie Kamerabilder sind möglich. Dabei müssen die Unzuverlässigkeit und Ungenauigkeit der Sensoren sowie möglicherweise begrenzte Ressourcen zur Verarbeitung beachtet werden. Diese Aspekte führen dabei in den Bereich der selektiven Aufmerksamkeit.

  Ebenfalls zu dieser Kategorie zählen Zustandsschätzer, die aktuelle Beobachtungen durch ihren zeitlichen Kontext anreichern. Durch diesen Schritt können nicht nur Rauschen und fehlerhafte Messungen korrigiert werden, sondern auch zeitliche Zusammenhänge erfasst werden, die mehr als einen Beobachtungszyklus benötigen.

  Die Fusion mehrerer Sensoren und das Erweitern der Wahrnehmung über einzelne Objekte hinaus auf Objektrelationen, soll die Gesamtsituation des Agenten einschätzen. Dies ist für die Nutzung eines Modells von entscheidender Bedeutung. Diese komplexe Gesamteinschätzung kann nur im Zusammenspiel mit der Erfassung und Kategorisierung des Wahrgenommenen geschehen.

- **Erfassung und Kategorisierung**
  Zwischen den wahrgenommenen Eindrücken und dem Wissen des Agenten muss eine Verknüpfung hergestellt werden. Das kann geschehen, indem die sensorischen Eindrücke nach typischen Mustern durchforstet oder/und in Klassen eingeteilt, also kategorisiert, wer-

den. Dazu muss die Architektur diese Muster und Klassen speichern können und eine Relation zwischen den Mustern und Klassen definieren z.B. über ein Konzept der Ähnlichkeit zwischen Mustern.

- **Vorhersage und Überwachung**
  Mittels eines Modells können Vorhersagen über Auswirkungen von bestimmten Aktionen gemacht werden, die über den beschränkten Horizont eines einzelnen Durchlaufs des Zyklus hinausgehen. Damit wird einerseits ein Planen ermöglicht und andererseits die Überwachung eines Plans möglich. Wenn die Umwelt sich anders verhält als erwartet, ist dies ein sicheres Zeichen dafür, dass entweder der Plan geändert werden muss, man spricht auch vom Planzusammenbruch, oder das Modell schlecht ist. Beides sollte einen Adaptionsprozess anstoßen.

- **Problemlösen und Planen**
  Wenn ein Modell zur Verfügung steht, dass die Auswirkungen der eigenen Aktionen abschätzen kann, wird Planung möglich. Ein solcher Plan wird simuliert oder ausgeführt und bewertet, wie erfolgreich er ist. Gegebenenfalls kann der Plan auch angepasst werden. Daher muss eine Architektur Komponenten besitzen, welche in der Lage sind, einen Plan zu repräsentieren und zu speichern, z.B. als Folge von Aktionen. Während Planung jenes beschreibt, was intern im Agenten vorgeht, beschreibt Problemlösungsfähigkeit zusätzlich solche Aspekte, die durch Interaktion mit der Umwelt zu einem Ziel führen, beispielsweise durch Versuch und Irrtum.

- **Entscheiden und Wählen**
  Während Planung und Problemlösung eher abstrakte Entscheidungen auf höherer Ebene darstellen, gibt es meist auch die direkte Kopplung von wahrgenommenen Mustern und Handlungen auf niedriger Ebene. Diese direkte Sensor-Aktor-Kopplung bildet die Grundlage für die meisten kognitiven Architekturen. Die höheren Ebenen zur Planung schränken dazu beispielsweise die Möglichkeiten der Ak-

tionen ein oder geben Verhaltensmuster vor. Auch müssen Widersprüche oder Konflikte, die aus Plänen höhere Ebenen herrühren, aufgelöst werden, um eine Aktion durchführen zu können.

In fast allen Fällen ist es wünschenswert, dass der Agent in der Lage ist, seine Entscheidungen aufgrund der gemachten Erfahrungen zu verbessern.

- **Ausführung und Aktion**
  Um die getroffenen Entscheidungen zur Manipulation der Umwelt durchführen zu können, muss die Architektur in der Lage sein, diese als Aktionen (Bewegungsprimitive oder komplexere Aktionsfolgen) zu repräsentieren und über die Aktuatorik ausführen.

- **Interaktion und Kommunikation**
  Bestandteil der Umwelt, die manipuliert wird, können andere Agenten oder Menschen sein, von denen Hilfe angefordert oder gar Wissen transferiert werden kann. Dazu ist es notwendig, dass die Architektur ihr Wissen transformieren und kommunizieren kann. Man kann dies auch als eine komplexe Aufgabe interpretieren, die Wahrnehmung (Was hat mein gegenüber verstanden? Was möchte er?), Planung (Wie erkläre ich es ihm?) und Handlung (Meine Botschaft) erfordert, wenn man den Gegenüber als Teil der Umwelt ansieht.

Im Kern der Darstellung in Abbildung 2.1 findet man die Eigenschaften, die sich nicht explizit auf einzelne Bereiche im Wahrnehmungs-Handlungs-Zyklus abbilden lassen.

- **Erinnern, Lernen und Reflektieren**
  Die Fähigkeit zu lernen ist an vielen Stellen innerhalb der kognitiven Architektur umsetzbar. So können Klassen für die Kategorisierung gelernt werden, oder ein Modell zur Repräsentation der Umwelt, wie auch optimale Aktionen für bestimmte Situationen. Daher fallen alle Fragen, die sich mit dem „Welche Teile der Architektur lernen? Wann lernt welcher Teil?, Wie beeinträchtigt dies die Handlungen der Ar-

chitektur?“ beschäftigen, in diese Kategorie. Ebenfalls von Interesse sind Fragestellungen, die das Speichern, Abrufen und Abstrahieren von Erfahrungen angeht, also die Organisation von Wissen innerhalb der Architektur.

Sehr selten findet man auch Konzepte, in denen reflektiert wird. Es geht dabei um das Finden von Erklärungen und Rechtfertigungen, warum bestimmte Handlungen ausgeführt wurden oder warum bestimmte andere kognitive Fähigkeiten, wie z.B. Planung an einer bestimmten Stelle, durchgeführt werden.

- **Schlussfolgern und Meinungspflege**[1]
  Eng verwandt mit der Planung und dem Problemlösen ist das Schlussfolgern. Während die beiden erstgenannten Fähigkeiten direkt zum Erreichen von Zielen eingesetzt werden, geht es beim Schlussfolgern um das Ableiten von neuem Wissen aus vorhandenem Wissen. Man spricht dabei oft von Inferenz. Neues Wissen kann sich induktiv (vom Speziellen zum Allgemeinen) oder deduktiv (vom Allgemeinen zum Speziellen) ergeben.

  Meinungspflege bezieht sich auf die interne Konsistenz des erlernten Wissens. Gerade in veränderlichen Umgebungen kann gelerntes Wissen veralten und damit an Nutzen verlieren, da es Widersprüche zwischen internem Weltbild und der Umwelt gibt. An dieser Stelle muss sichergestellt werden, dass das Wissen des Agenten erneuert wird.

  Beim Schlussfolgern und bei der Meinungspflege handelt es sich wohl um eine der größten Herausforderungen im Kontext lernender Systeme und kognitiver Architekturen.

[1] In der englischsprachigen Literatur wird dies als *belief maintance* bezeichnet.

2.1. Architekturen in der Automatisierung

Es existiert im Bereich der Kognitionswissenschaften eine Vielzahl von verschiedenen Architekturkonzepten. In [Langley et al., 2009] werden mehr als 15 kognitive Architekturen vorgestellt, die sich in ihrer Art, Wissen zu repräsentieren und zu verwenden, unterscheiden.

Dies beginnt bei bekannten Vertretern wie die ACT-R Architektur (Abkürzung für: *Adaptive control of thought-rational*)[Anderson et al., 2004] bis hin zu modernen Ansätzen wie die Architektur CLARION (Abkürzung für: *Connectionist Learning with Adoptive Rule Induction ON-line*) [Sun et al., 2001].

Der Fokus bei diesen Architekturen liegt darauf, das menschliche Denken im Gehirn zu modellieren. Die verwendeten Module unterscheiden sich deutlich, wie auch die Repräsentation von Wissen vielfältig realisiert wird, z.B. durch Chunks und Produktionsregeln [Anderson et al., 2004] als symbolische Repräsentationen oder Aktivierungswahrscheinlichkeiten und neuronale Netze als subsymbolische Vertreter [Sun et al., 2001]. Das Lernen erfolgt durch Ansätze wie Reinforcement Learning oder das Erstellen neuer Produktionsregeln basierend auf Methoden der Prädikatenlogik. Die möglichen Kombinationen sind endlos und jede einzelne Architektur bräuchte etliche Seiten, um hier die Grundkonzepte darzulegen.

Schaut man stattdessen in den riesigen Bereich der Automatisierungs- und Regelungstechnik, z.B. in das einen aktuellen Überblick bietende „Springer Handbook of Automation“ [Nof, 2009], stellt man fest, dass der Begriff der Architektur fast ausschließlich im Sinne der Softwarearchitektur, dem softwaretechnischen Rahmen für die Implementierung einer Automatisierungslösung, verwendet wird. Auch den Wahrnehmungs-Handlungs-Zyklus findet man kaum als solchen.

Löst man sich jedoch von den Begriffen, stellt man fest, dass hier über die gleichen Dinge geredet wird. Jeder geschlossene Regelkreis

entspricht dem Wahrnehmungs-Handlungs-Zyklus. Ein einfacher PID-Regler realisiert eine Sensor-Aktor-Kopplung, die basierend auf der aktuellen Regelabweichung (Wahrnehmung) eine Stellgröße berechnet und auf die Regelstrecke angewendet wird (Handlung). Wissen über das Problem ist dabei in den Konstanten des Reglers, die zur Berechnung der Stellgröße verwendet werden, gespeichert. Ein offener Regelkreis, also eine Regelstrecke ohne Rückkopplung, kann als einmaliger Durchlauf des Wahrnehmungs-Handlungs-Zyklus betrachtet werden. Basierend auf einer initialen Beobachtung werden ein Plan und die zugehörigen Aktionen ausgeführt. Ein Beispiel dafür sind medizinische, automatische Operationsroboter, bei denen basierend auf einer Aufnahme eines entsprechend fixierten Patienten ein Eingriff und die dazu notwendige Roboterbewegung geplant und durchgeführt werden [Troccaz, 2009].

Auch der Begriff einer hybriden Regelung entspricht einer Kopplung von reaktiven Komponenten auf einer problemnahen Ebene mit einer deliberativen (meist überwachenden) Komponente auf symbolischer Ebene - also der Definition einer hybriden Architektur. Decision Support Systeme werden mit einer Problembeschreibung konfrontiert und bestimmen auf Basis von Modellwissen einen Lösungsvorschlag, der über eine Benutzerschnittstelle dem Menschen präsentiert wird. Dies sind alles Aspekte, die auch in der Beschreibung der kognitiven Architekturen Platz fanden.

Im Feld der Robotik und damit der Steuerarchitekturen für Roboter fügen sich die beiden Welten von Automatisierung und Kognitionswissenschaften am ehesten zusammen. Dort findet man das klassische Sense-Plan-Act Paradigma [Arkin, 1998], rein reaktive Systeme, die Wissen ausschließlich subsymbolisch repräsentieren wie die Subsumption-Architecture [Brooks, 1986] und auch hybride Ansätze, wie die 3T-Architektur [Bonasso et al., 1997]. In der 3T Architektur setzt die unterste Ebene ein reaktives Verhalten um, in dem direkte sensomotrische Verhaltensweisen realisiert werden. Die oberste Ebene ist ein

deliberativer, abstrakter Planer, der die Ziele des Roboters verwaltet und ihr Erreichen plant. Die mittlere Schicht dazwischen dient als Vermittler zwischen dem abstrakten Plan und dem reaktiven Verhalten. Dazu wird der Plan zerlegt und durch Verhaltensfolgen modelliert, die dann in der unteren Schicht zur Anwendung gebracht werden.

Sehr weit in Richtung der klassischen kognitiven Architekturen geht dabei die Verwendung des PolyScheme Modells in der Mensch-Roboter-Interaktion [Trafton et al., 2005], welche eine gewisse Verwandtschaft zur oben erwähnten ACT-R Familie aufweist, allerdings im Gegensatz zur Definition von kognitiven Architekturen ebenfalls gewisse Anforderungen an das Wissen selbst stellt.

Eine konkrete Architektur, die zur Regelung komplexer Prozess im Bereich der Automatisierungstechnik zum Einsatz kommt, konnte jedoch nicht gefunden werden.

2.2. Verwendete Systemarchitektur

Bei dem in dieser Arbeit vorgestellten System handelt es sich um eine hybride Architektur, die jedoch sehr stark in Richtung der subsymbolischen Wissensverarbeitung ausgelenkt ist. Dies ergibt sich aus der Tatsache, dass für komplexe Regelungsaufgaben oftmals nur unzureichendes, unscharfes oder gar falsches Symbolwissen vorhanden ist. Daher wird als Basis von der Prämisse ausgegangen, dass Wissen durch Beobachtung des Prozesses erlernt werden muss. Symbolisches Wissen wird erst auf der Ebene des Lernmanagements einbezogen. Bei der Frage nach einem reaktiven oder deliberativen Verhalten wird hier auf verschiedene Verfahren des Reinforcement Learnings eingegangen, die sich als Hybridverfahren einstufen lassen.

Die Komponenten der Architektur lehnen sich dabei sehr nah an den am Anfang des Kapitels diskutierten Wahrnehmungs-Handlungs-Zyklus an

und entsprechen damit einer funktionsorientierten Architektur. Was die aufgezählten Fähigkeiten und Funktionen angeht, kann im Rahmen dieser Arbeit nur eine Untermenge sinnvoll betrachtet werden.

- **Wahrnehmung, Erfassung, Kategorisierung und Situationseinschätzung**

  Diese Aspekte werden vor allem unter dem Gesichtspunkt der Vielzahl verschiedener Sensoren betrachtet, die alle ein riesiges Datenvolumen produzieren. Allerdings sind nicht alle Daten informativ für die Zielstellung des Systems. Vielmehr können sich unnütze Daten negativ auswirken, indem sie Rechenkapazität belegen und Störungen einbringen. Daher muss eine Kategorisierung verschiedener Kanäle vorgenommen werden, ob diese für bestimmte Aufgaben relevant sind oder nicht. Die Methoden dazu werden in Kapitel 3 vorgestellt. Der Frage, was beachtet werden muss, wenn sich der Informationsgehalt im Laufe der Zeit ändert (z.B. durch Verschmutzung von Sensoren oder andere Prozessdynamiken) wird in Kapitel 5 nachgegangen.

  Dies wird den umfangreichsten Beitrag dieser Arbeit darstellen, da hier neue Ansätze und Algorithmen vorgestellt werden. Dies kann auch mit folgendem Zitat aus [Langley et al., 2009] im Abschnitt *Open issues in cognitive architectures* (Seite 15) motiviert werden:

  „Most architectures emphasize the generation of solutions to problems or the execution of actions, but categorization and understanding are also crucial aspects of cognition, and we need increased attention to these abilities."

- **Vorhersage und Überwachung**

  Diese Funktionen werden im Rahmen der Dissertation nicht explizit betrachtet, finden sich jedoch implizit wieder. So wird beispielsweise eine steigende Abweichung zwischen Vorhersagen des Modells und den Beobachtungen genutzt, um neue Modelle zu lernen (Kapitel 5).

Eine Überwachung kann dadurch realisiert werden, dass Sensorkanäle deren Informationsgehalt schwindet, überprüft werden. Entweder rührt dieser Informationsverlust vom Verschleiß des Sensors her oder durch Änderungen im Prozess selbst.

- **Problemlösen, Planen, Entscheiden und Wählen**

  Im Rahmen der hier eingesetzten Architektur wurde der Fokus auf moderne Reinforcement Learning Verfahren gelegt. Dabei werden sowohl Verfahren betrachtet, die ein explizites Modell des Prozesses verwenden, als auch ein modellfreies Verfahren. Gemein ist allen Reinforcement Learning Verfahren, dass sie eine implizite Planung realisieren. Implizit bedeutet in diesem Zusammenhang, dass sie nicht eine fertige Abfolge von Aktionen festlegen, sondern in der akkumulierter Belohnung (z.B. in Form einer Action-Value-Function, vgl. Kapitel 4) diese Aktionsfolge kodiert ist. Im Kapitel 6 werden im Kontext der Anwendung Vergleiche mit anderen Ansätzen zur Planung und Entscheidung - namentlich der Modellprädiktiven Regelung und einer wahrscheinlichkeitsbasierten Modellierung auf Basis von Faktorgraphen - vorgenommen.

- **Ausführung und Aktion**

  Die Aktuatorik zur Beeinflussung der Umwelt wird als inverses Problem zur Sensorik aufgefasst. Daher wird auch hier die Frage gestellt, welche der Aktionsmöglichkeiten, die dem System zur Verfügung stehen, auch zielführend zur Lösung der bestehenden Aufgabe beitragen. Dies wird daher ebenfalls in Kapitel 3 angesprochen.

- **Erinnern und Lernen**

  Lernverfahren, und damit auch die Problematik des Erinnerns und Vergessens, sind für alle Teile der Architektur von Bedeutung, wenn man es mit dynamisch veränderlichen Umgebungen zu tun hat, die den Erwerb neuen Wissens und die Formulierung neuer Strategien

erfordern. Die sich daraus ergebenden Abhängigkeiten und Lernmechanismen werden im Kapitel 5 vorgestellt.

- **Reflektieren, Schlussfolgern, Meinungspflege, Kommunikation und Interaktion**

  Diese Aspekte werden im Rahmen dieser Arbeit nicht weiter vertieft, was der subsymbolischen Herangehensweise geschuldet ist. Diese Funktionen erfordern eine symbolische Repräsentation des Wissens. Eine Kopplung zwischen der Symbolik und ihrer subsymbolischen Repräsentation ist explizit nicht Bestandteil dieser Arbeit, daher werden diese Aspekt nur im Ausblick in Kapitel 7 angesprochen.

Die nächsten drei Kapitel beschreiben die benannten Aspekte ausführlich, während danach in Kapitel 6 die funktionierende Gesamtarchitektur am Beispielszenario der intelligenten Feuerungsführung vorgestellt wird. Danach wird in Form einer kritischen Wertung darauf eingegangen, welche Aspekte in zukünftigen Arbeiten in den Mittelpunkt rücken sollten.

3. Merkmalsextraktion

Der erste Schritt beim Durchlaufen des Wahrnehmungs-Handlungs-Zyklus besteht, wie der Name bereits sagt, im Wahrnehmen der Umwelt mittels der verfügbaren Sensorik. In der Fülle der gemessenen Daten finden sich Informationen, welche für die aktuelle Aufgabe von Relevanz sind und solche, die weniger hilfreich sind. Damit die Vorhersage-, Planungs- und Problemlösungsinstanzen nicht in der Datenflut ertrinken, besteht die Notwendigkeit die Daten vorher zu bewerten.

In komplexeren Systemen könnte dazu ein Kategorisierungssystem zum Einsatz kommen, welches versucht, den Sensorinformationen semantische Klassen zuzuordnen. Bei der in dieser Arbeit betrachteten rein datengetriebenen Arbeitsweise jedoch, reduziert sich das Problem auf die Frage, ob bestimmte wahrgenommene Daten für eines oder mehrere der zu lösenden Teilprobleme von Wichtigkeit sind. Im Bereich des Maschinellen Lernens wird diese Fragestellung als Merkmalsextraktionsproblem oder auch Signifikanzanalyse bezeichnet.

Der weitere Aufbau dieses Kapitels ist dabei wie folgt. Zunächst sollen die unterschiedlichen Klassen von Signifikanzanalysetechniken vorgestellt werden, wobei klar wird, dass ein geeignetes Kriterium zur Messung von Relevanz von Merkmalen notwendig ist. Die Transinformation ist ein solches und wird, da sie von zentraler Bedeutung im weiteren Verlauf des Kapitels ist, ausführlich theoretisch vorgestellt. Danach folgen Untersuchungen, wie die Transinformation praktisch bestimmt werden kann. Unter Verwendung dieses Kriteriums werden dann neue Algorithmen vorgestellt, die eine schnelle Merkmalsauswahl erlauben.

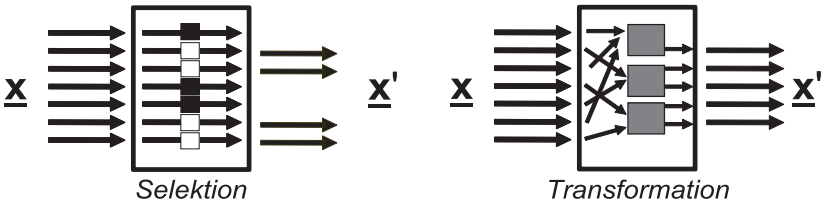


Abbildung 3.1.: Einteilung der Merkmalsextraktionsverfahren. Links die Merkmalsselektionsverfahren, welche eine binäre Entscheidung über die Weiterverwendung der Eingangsvariablen treffen und rechts die Transformationsverfahren, welche basierend auf einem funktionalen Zusammenhang neue Kanäle aus den Eingangsvariablen berechnen.

Die beschriebenen Methoden werden dann auf das eng verwandte Feld der Aktionsraumauswahl übertragen. Schließen wird dieses Kapitel mit einer Übersicht über Anwendungsszenarien, in denen die hier entwickelten Methoden erfolgreich eingesetzt werden konnten.

Merkmalsselektion und Merkmalstransformation

Die Verfahren zur Merkmalsextraktion werden in zwei Gruppen unterteilt. Einerseits handelt es sich dabei um Merkmalsselektionsverfahren, welche eine binäre Entscheidung treffen, ob eine bestimmte Eingangsvariable von Nutzen ist oder nicht. Andererseits gibt es die Merkmalstransformationsverfahren, welche versuchen die Eingangskanäle anteilig so zu vermischen, dass die Information in wenigen neuen Kanälen gebündelt werden kann. Schematisch werden diese Ansätze in Abbildung 3.1 dargestellt.

Beide Paradigmen haben ihre Daseinsberechtigung. [Torkkola, 2002] argumentiert, dass die Selektionstechniken zwar die leichtere Entscheidung zu treffen haben, nämlich nur ob ein Kanal relevant ist oder nicht, als die Transformationsansätze, welche konkret den Anteil bestimmen müssen, mit dem ein Kanal Relevanz zeigt. Trotzdem haben die Trans-

formationsverfahren durch die kontinuierlichen Anteile den Charme, dass hier klassische Optimierungsverfahren wie beispielsweise Gradientenverfahren einfach angewandt werden können. Die diskrete Selektion hingegen ist auch in der Optimierungstheorie schwieriger zu handhaben, da der Raum, in dem optimiert wird, Unstetigkeiten und undefinierte Bereiche aufweist. Daher postuliert Torkkola, dass unter der Bedingung der Existenz eines geeigneten Optimierungskriteriums, die Merkmalstransformation das einfachere Problem ist.

Ein weiterer Aspekt bei der Unterscheidung zwischen Selektion und Transformation ist die *intrinsische Dimension* der Daten. Diese, meist unbekannte, Größe gibt an, wie viele Eingangskanäle zur Lösung eines Problems minimal benötigt werden. Ist diese Zahl sehr niedrig, ist es oft einfacher, die wichtigen Kanäle zu selektieren, während die Transformation bei einer hohen intrinsischen Dimension die Information aus vielen Kanälen effektiv komprimiert.

Von einem praktischen Standpunkt aus gesehen, ist diese Diskussion allerdings unerheblich, da oftmals beide Ansätze miteinander kombiniert werden. Daher werden in dieser Arbeit auch beide Gruppen betrachtet, wobei mit der Selektion begonnen werden soll.

3.1. Merkmalsselektionstechniken

Ziel der Selektion ist es, eine minimale hinreichende Merkmalsmenge zu finden. Dazu wird eine möglichst kleine Teilmenge der Eingangsvariablen gesucht, die möglichst dieselbe Aussagekraft haben soll, wie die Menge aller Eingangsvariablen. Dazu werden irrelevante Variablen ausgeschlossen und relevante Variablen genutzt. Irrelevante Kanäle sind dabei solche, die nicht für das zu lösende Approximations- oder Klassifikationsproblem nützlich sind. Die relevanten Variablen gibt es in starker und schwacher Ausprägung. Stark bedeutet in diesem Zusam-

menhang, dass die Nutzung eines solchen Kanals immer bei der Erfüllung der Aufgabe hilft. Schwach relevante Kanäle hingegen führen nur unter bestimmten Umständen zu einer Verbesserung des Ergebnisses - so zum Beispiel bei redundanten Kanälen oder abhängigen Kanälen, wie beim XOR-Problem. Mehr zu dieser Einteilung und den Problemen mit schwach relevanten Kanälen findet man in [Guyon und Elisseeff, 2003].

Formal kann die Selektion als Suche im diskreten Raum der Merkmale angesehen werden. Folgt man [Langley, 1994], gibt es vier entscheidende Eigenschaften einer Merkmalsselektionstechnik:

1. Startpunkt der Suche im Suchraum (z.B. leere Merkmalsstartmenge oder vollständige Merkmalsstartmenge)
2. Suchstrategie (z.B. Hinzufügen eines neuen Merkmals oder zufälliges Raten einer Merkmalsmenge)
3. Evaluierungskriterium für einen Punkt im Suchraum (z.B. Transinformation der Kanäle zum Ziel)
4. Haltekriterium für das Ende der Suche (z.B. festgelegte Merkmalszahl oder Approximationsgüte eines neuronalen Netzes)

Die Kriterien eins und zwei sind dabei algorithmenspezifisch, während der vierte Punkt entweder durch den Algorithmus definiert ist oder sich aus der Anwendung ergibt. Von fundamentaler Bedeutung ist jedoch der dritte Punkt, da das Evaluierungskriterium zwei Wege aufzeigt, nämlich die sogenannten Filteransätze und die Wrapperverfahren (deutsch *einhüllende Ansätze*) [Kohavi und John, 1997]. Diese sollen nun näher betrachtet werden.

Definition 3.1

Filterverfahren

Die Bewertung der Eingangsvariablen erfolgt unabhängig vom verwendeten lernenden System auf Basis eines definierten Relevanzkri-

teriums. Die Bildung der Merkmalsteilmenge erfolgt mit Hilfe der ermittelten Rangfolge der Eingangskanäle.

Ursprünglich entstammen die Filterverfahren aus der Statistik, dem Data Mining und der Informationstheorie. Ein typischer Vertreter ist dabei die Verwendung des Korrelationskoeffizienten als Relevanzkriterium. Dabei wird im einfachsten Fall zwischen jeder Eingangsvariablen X_i und der Zielgröße Y die Korrelation bestimmt. Diese Korrelationskoeffizienten können dann betragsmäßig sortiert und eine Auswahl der relevantesten Kanäle getroffen werden. Andere Relevanzkriterien sind ebenfalls denkbar. Im Abschnitt 3.2 werden Größen aus der Informationstheorie Verwendung finden.

Definition 3.2

WRAPPERVERFAHREN

Ein beliebiger Funktionsapproximator (Black Box) wird mit unterschiedlichen Merkmalsteilmengen trainiert. Die Fehlerrate des resultierenden Approximators wird benutzt, um die Nützlichkeit der aktuell ausgewählten Merkmalsmenge zu bewerten.

Wrapperverfahren schlagen nach einer definierten Suchstrategie Kombinationen von Variablen vor und trainieren damit einen Approximator. Dessen Ergebnis und resultierender Fehler wird genutzt, um neue Variablenkombinationen zu bestimmen. Eine vollständige Suche ist oft nicht möglich, da das Problem NP-schwer ist. Deshalb sind hier effiziente, approximierende Suchstrategien notwendig. Ein sehr einfaches Beispiel ist dabei die sequentielle Vorwärtssuche (Sequential Foward Selection) [REUNANEN, 2006], die in Abschnitt 3.4 vorgestellt wird.

In jüngerer Zeit [GUYON und ELISSEEFF, 2003] wurde eine weitere, dritte Kategorie eingeführt, die Embeddedverfahren (deutsch *eingebettete Ansätze*). Es handelt sich dabei um Ansätze, die zuvor zur Klasse

der Wrapperverfahren gezählt wurden und viele Eigenschaften mit diesen teilen.

Definition 3.3

EMBEDDEDVERFAHREN

Ein spezieller Approximator wird mit allen vorhandenen Merkmalen trainiert. Aus der Struktur des resultierenden Approximators wird auf die Nützlichkeit der einzelnen Merkmale geschlossen.

Eingebettete Verfahren sind immer an eine spezielle Architektur eines Klassifikators oder Approximators gekoppelt, da sie die Auswahl der Merkmale auf Basis spezifischer Eigenschaften der Lernverfahren treffen. Sie entstammen daher ausnahmslos dem Bereich des Maschinellen Lernens. Beispiele dazu umfassen den Optimal Brain Damage Ansatz für mehrschichtige Vorwärtsnetze [LE CUN et al., 1990], Random Forest auf Basis von Klassifikations- und Regressionsbäumen [BREIMAN, 2001], Automatic Relevance Determination im Zusammenhang mit Bayes Neural Networks [NEAL, 1996] und den Recursive Feature Elimination Ansatz für Support Vector Machines [GUYON et al., 2002].

Diese dritte Gruppe von Verfahren wird im weiteren Verlauf dieser Arbeit nicht näher betrachtet. Für die weiteren Aussagen, die in diesem Abschnitt getroffen werden, können sie vereinfachend als Teil der Wrapperverfahren angesehen werden.

Vor- und Nachteile der Ansätze

Betrachtet man die Gruppe der Filteransätze, so lässt sich feststellen, dass sie unabhängig vom verwendeten Lernalgorithmus sind. Die Auswahl erfolgt nur über die statistische *Relevanz*. Dies ist sowohl ein Vorteil als auch ein Nachteil. Im Allgemeinen sind Filteransätze schneller

als Wrapperansätze, da die zeitaufwendigen Operationen nicht die Bestimmung der Relevanzkriterien sind[1]. Vielmehr erfordert das Training von Funktionsapproximatoren und deren Bewertung eine Vielzahl von Operationen. Dieses aufwendige Training ist bei Wrappern mindestens einmal, meist jedoch sehr viel häufiger notwendig. Daher sind Filteransätze auch bei einer großen Anzahl von Eingangsvariablen nutzbar.

Zwar langsamer in der Berechnung, bieten die einhüllenden Verfahren jedoch den Vorteil, dass sie nicht ausschließlich die statistische Relevanz betrachten, sondern die *Nützlichkeit* für den konkreten Approximationsalgorithmus. Nützlichkeit beschreibt dabei den konkreten Gewinn bei der Minimierung des Approximations- oder Klassifikationsfehlers, und ist damit die praktisch entscheidendere Größe.

Nützlichkeit und Relevanz sind dabei nicht immer gleich. Es können zwei Fälle unterschieden werden:

1. Die Relevanz eines Kanals ist größer als seine Nützlichkeit.
   Dies ist dann der Fall, wenn der Bias des Klassifikators verhindert, dass alle Informationen des Eingangskanals auch genutzt werden können. Man stelle sich einen linearen Klassifikator (z.B. Single Layer Perceptron) vor, für den eine Eingangsgröße nicht nützlich ist, falls sie nur einen nichtlinearen Zusammenhang enthält. Je nach gewähltem Relevanzkriterium wird dieser aber durch die statistischen Maße erkannt und als relevant eingestuft.

2. Die Relevanz ist kleiner als die Nützlichkeit.
   Wenn ein Kanal durch hohes Rauschen und redundante Informationen nur eine niedrige Relevanz durch ein Filterverfahren zugewiesen bekommt, kann dieser sich trotzdem als nützlich erweisen, in dem er z.B. die numerische Stabilität erhöht oder die Generalisierungsfähigkeit verbessert. Dieses Verhalten wird auch in [Kohavi und John, 1997] beschrieben und näher untersucht.

[1] Es lassen sich auch Gegenbeispiele mit sehr komplexen Relevanzkriterien finden, für die diese Aussage nicht wahr ist.

Der Wunsch ist es daher, die Nützlichkeit der Eingangskanäle zu kennen. Jedoch scheitert dies meist an einem zu großen Berechnungsaufwand. Ein Weg, der in dieser Arbeit beschritten werden soll, propagiert die Kombination beider Ansätze, um mit vertretbarem Aufwand die Nützlichkeit von Kanälen zu bestimmen.

Dazu ist es notwendig, beide Seiten der Medaille näher zu beleuchten. Die nächsten beiden Abschnitte werden ein umfassendes Relevanzkriterium, die aus der Informationstheorie stammende Transinformation, definieren und aufzeigen, wie sie berechnet werden kann. Danach wird dieses Konzept zur Formulierung effektiver Suchstrategie angewendet.

3.2. Grundlagen aus der Informationstheorie

In diesem Abschnitt soll der Begriff der *Information* mit Hilfe der Konzepte aus der Informationstheorie mathematisch definiert werden. Typischerweise wird nicht die *Information* selbst ausformuliert, sondern, um der notwendigen Breite gerecht zu werden, die zwei wichtigen Begriffe Entropie und Transinformation. Beide zusammengenommen entspricht am ehesten dem intuitiven Verständnis von *Information*. Der Ursprung dieser Konzepte sind dabei die Arbeiten von Shannon [Shannon, 1948]. Die nachfolgenden Definitionen basieren auf [Cover und Thomas, 2006].

Entropie ist ein Maß für die Unsicherheit über eine diskrete Zufallsvariable. Weniger formal kann man sie auch als Maß für die Überraschung sehen, die erwartet wird, wenn man die Ausprägung der Variable beobachtet.

Definition 3.4

Entropie

Sei X eine diskrete Zufallsvariable mit der Wahrscheinlichkeitsfunktion $p(x) = Prob(X = x)$ wobei x aus der Menge der möglichen

Ausprägungen für die Zufallsvariable stammt. Dann ist die Entropie $H(X)$ dieser Zufallsvariable definiert als

$$H(X) = -\sum_x p(x) \log p(x).$$

Die Art der Basis des verwendeten Logarithmus ist funktional unerheblich, jedoch wird im weiteren Verlauf der Arbeit immer vom Logarithmus zur Basis 2 ausgegangen. Dies erlaubt die Verwendung von Bit als Maßeinheit für die Information. Die Entropie ist immer ein nichtnegativer Wert. Die Entropie ist genau dann 0, wenn keine Unsicherheit über die Zufallsvariable besteht. Falls es genau eine Ausprägung der Zufallsvariable gibt, die mit Wahrscheinlichkeit $p(x_1) = 1$ auftritt enthält diese Variable keine Information. Die Entropie ist maximal, wenn alle möglichen Ausprägungen gleich wahrscheinlich sind. Das heißt, die Messung einer Ausprägung ist am informativsten, falls alle Ausprägungen mit gleicher Wahrscheinlichkeit auftreten oder, anders formuliert, die Unsicherheit über die Variable am höchsten ist. Die Entropie entspricht dann $H(X) = \log |X|$, wobei $|X|$ die Anzahl der Ausprägungen von X angibt.

Die Entropiedefinition nach Shannon ist ein Spezialfall der Rényi-Entropie [Renyi, 1961].

Definition 3.5

Rényi-Entropie

Die Rényi-Entropie der Ordnung α ist dabei definiert als

$$H_\alpha(X) = \frac{1}{1-\alpha} \sum_x \log p(x)^\alpha,$$

wobei $\alpha > 0$ gelten muss.

Für den Spezialfall von $\alpha = 1$ kann mittels Grenzwertbetrachtung gezeigt werden, dass dies der Definition nach Shannon entspricht [Renyi, 1961]. Im Rahmen dieser Arbeit wird ebenfalls die Ordnung $\alpha = 2$ von Interesse sein, welche auch als Korrelationsentropie bezeichnet wird.

Das Konzept der Entropie kann auf zwei Zufallsvariablen X und Y erweitert werden.

Definition 3.6

Verbundentropie

Die Verbundentropie $H(X, Y)$ gibt die Unsicherheit über X und Y an und ist als

$$H(X, Y) = -\sum_x \sum_y p(x, y) \log p(x, y)$$

definiert.

Analog zur bedingten Wahrscheinlichkeit in der Stochastik lässt sich die bedingte Entropie definieren.

Definition 3.7

Bedingte Entropie

Die bedingte Entropie $H(X|Y)$ gibt die verbleibende Unsicherheit über X an falls die Ausprägung der Zufallsvariablen Y bekannt ist

$$H(X|Y) = -\sum_x \sum_y p(x, y) \log p(x|y).$$

Dabei gilt, dass die Kenntnis einer zusätzlichen Variable die Unsicherheit niemals erhöhen kann. Falls Y keine Informationen über X enthält, verringert sich die Unsicherheit nicht. Daher gilt

$$H(X|Y) \leq H(X).$$

Die eben benannte Verringerung der Unsicherheit über die Variable X durch Kenntnis der Variable Y ist dabei die Information, die Y über X enthält.

Definition 3.8

TRANSINFORMATION

Damit ergibt sich eine erste Definition der Transinformation (engl. Mutual Information) $I(X;Y)$ wie folgt

$$\begin{aligned} I(X;Y) &= H(X) - H(X|Y) \\ &= H(X) + H(Y) - H(X,Y) \end{aligned}$$

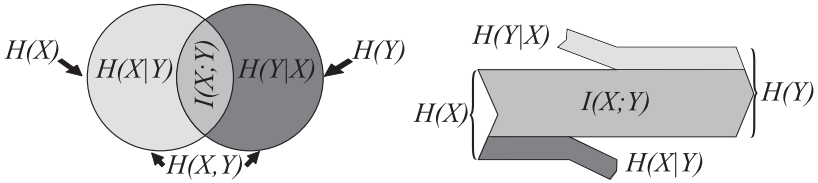
Daraus lassen sich folgende Eigenschaften ableiten:

- Die Transinformation ist nicht negativ. $I(X;Y) \geq 0$.
- Die Transinformation ist maximal, wenn X vollständig durch Kenntnis von Y erklärt wird. Sie entspricht dann der Entropie von X.
- Die Transinformation ist symmetrisch. Wenn Y Informationen über X enthält, so gilt umgekehrt auch, dass X Information über Y enthält. Daraus folgt

$$I(X;Y) = H(Y) - H(Y|X).$$

Grafisch werden diese Zusammenhänge in Abbildung 3.2 als Venn-Diagramm und als Kanaldarstellung verdeutlicht.

Durch Einsetzen der Definitionen 3.4 und 3.7 in die Gleichung für die Transinformation ergibt sich unter Anwendung der Logarithmengesetze folgende Form:



(a) (b)

Abbildung 3.2.: **(a)** Zusammenhänge von Entropie und Transinformation als Venndiagramm. Die Entropie über die Zufallsvariable X ist als kreisförmige Menge links dargestellt, analog dazu die Entropie der Zufallsvariable Y auf der rechten Seite. Die Vereinigung beider Mengen bildet die Verbundentropie über X und Y, während der Teil, den beide Variablen gemein haben, durch den Schnitt der Mengen darstellt ist. **(b)** Derselbe Zusammenhang als Kanaldarstellung wie sie Nachrichtentechnik verbreitetet ist. Ein Teil der von der Merkmal X ausgesendeten Information findet sich auch im Ziel Y wieder, dies ist die Transinformation. Allerdings gibt es auch Teile von X, die nichts über Y aussagen ($H(X|Y)$) und es gibt Teile des Ziels Y, die nicht durch Merkmal X erklärt werden können ($H(Y|X)$).

$$
\begin{aligned}
I(X;Y) &= H(X) - H(X|Y) \\
&= -\sum_x p(x) \log p(x) + \sum_x \sum_y p(x,y) \log p(x|y) \\
&= -\sum_x \sum_y p(x,y) \log p(x) + \sum_x \sum_y p(x,y) \log p(x|y) \\
&= \sum_x \sum_y p(x,y) \log \frac{p(x|y)}{p(x)} \\
&= \sum_x \sum_y p(x,y) \log \frac{p(x,y)}{p(x)p(y)}
\end{aligned}
$$

.

Dieses Ergebnis entspricht der Kullback-Leibler Divergenz (KLD) [Kullback, 1959] zwischen der Verbundverteilung $p(x, y)$ und dem Produkt ihrer Marginale $p(x)p(y)$. Die Kullback-Leibler Divergenz wird oft als Distanzmaß zwischen Verteilungen betrachtet, auch wenn es sich nicht um ein echtes Distanzmaß handelt, da sie nicht die Eigenschaften der Symmetrie und der Dreiecksungleichung erfüllt.

Allerdings lässt sich daraus folgende zusätzliche Eigenschaft der Transinformation ableiten:

- Die Transinformation ist genau dann null wenn X und Y unabhängig voneinander sind. X und Y sind statistisch unabhängig, wenn gilt $p(x, y) = p(x)p(y)$. In diesem Fall wird der Teilterm, von dem der Logarithmus zu berechnen ist, genau 1 und der Logarithmus von 1 ist immer 0.

Ein weiterer Vorteil der Sichtweise als Kullback-Leibler Divergenz ist die einfach Übertragbarkeit auf kontinuierliche Zufallsvariablen

Definition 3.9

Transinformation für kontinuierliche Variablen

$$I(X;Y) = \int_x \int_y p(x,y) \log \frac{p(x,y)}{p(x)p(y)} dydx.$$

Alle zuvor genannten Eigenschaften der Transinformation behalten hier ihre Gültigkeit - was beispielsweise für den Entropiebegriff nicht der Fall ist. Bei Erweiterung der Entropie auf kontinuierliche Variablen, was als differentielle Entropie bezeichnet wird, ist die Eigenschaft der Nichtnegativität nicht mehr gewährleistet. Daher ist im kontinuierlichen Fall die KLD-Formulierung von entscheidender Bedeutung.

Das Konzept der Transinformation lässt sich auch auf mehrere Variablen erweitern.

Definition 3.10

VERBUNDTRANSINFORMATION

Bei der Verbundtransinformation wird gemessen, wie viel Information eine Menge von Variablen $X_1, X_2, ..., X_n$ über eine andere Variable Y enthalten

$$I(X_1, ..., X_n; Y) = \int_x \int_y p(x_1, ..., x_n, y) \log \frac{p(x_1, ..., x_n, y)}{p(x_1, ..., x_n)p(y)} dydx.$$

Merkmalsselektion aus Sicht der Informationstheorie

In Abschnitt 3.1 wurde bereits informal die minimale hinreichende Merkmalsmenge eingeführt. Mit den in diesem Abschnitt vorgestellten Konzepten kann dies nun auch formal definiert werden.

Definition 3.11

MINIMALE HINREICHENDE MERKMALSMENGE

Die Merkmalsselektion sucht nach einer Menge S, welche dieselben Informationen über das Ziel Y enthält, wie die Menge aller verfügbaren Informationen X. Diese wird als hinreichende Merkmalsmenge bezeichnet. Die minimale hinreichende Merkmalsmenge S^* enthält eine Anzahl von Merkmalen die kleiner gleich jeder anderen hinreichenden Merkmalsmenge ist.

$$I(X; Y) = I(S^*; Y) \text{ mit } |S^*| \rightarrow \min$$

Zusammengefasst lässt sich feststellen, dass mit dem Konzept der Transinformation gemessen werden kann, wie viel Information eine (oder mehrere) Variable(n) über eine andere enthält. Dabei ist das

Konzept der Information nicht beschränkt auf lineare Zusammenhänge, wie beispielsweise der Korrelationskoeffizient oder die Fisher-Diskriminante, sondern erfasst jegliche Zusammenhänge in den Verteilungen. Dies ist im Sinne der Merkmalsextraktion eine herausragende Eigenschaft.

Doch so erfreulich die theoretischen Eigenschaften der Transinformation sind, gibt es beim praktischen Einsatz ein Problem. Um die Transinformation berechnen zu können, werden die Wahrscheinlichkeitsverteilungen $p(x), p(y)$ und $p(x, y)$ benötigt. Diese sind jedoch nur in den seltensten Fällen bekannt. Sie müssen daher aus den verfügbaren Daten geschätzt werden. Welche Methoden und Ansätze dazu existieren, und welche Probleme bei der Schätzung auftreten können, wird im nächsten Abschnitt näher erörtert.

3.3. Schätzung der Transinformation

Die Berechnung der Transinformation kann für praktische Probleme meist nur approximativ erfolgen, da die wahren Verteilungen der Daten nicht bekannt sind. In diesem Abschnitt sollen verschiede Verfahren zur Schätzung der Transinformation vorgestellt, systematisiert und verglichen werden. Dabei wird besonderes Augenmerk auf die Tauglichkeit zur Merkmalsselektion gelegt. Es werden in diesem Abschnitt auch Ergebnisse aus der Bachelorarbeit von Robert Kaltenhäuser [Kaltenhäuser, 2010] und der Praktikumsarbeit von Saurabh Verma verwendet. Diese wurden direkt vom Autor der vorliegenden Arbeit betreut und die Ergebnisse wurden in einer gemeinsamen Publikation veröffentlicht [Schaffernicht et al., 2010].

Aus der Literatur heraus können drei verschiedene Gruppen von Methoden abgeleitet werden. Diese sind in Abbildung 3.3 dargestellt. Es handelt sich dabei um die Gruppe der Verfahren, welche direkt die

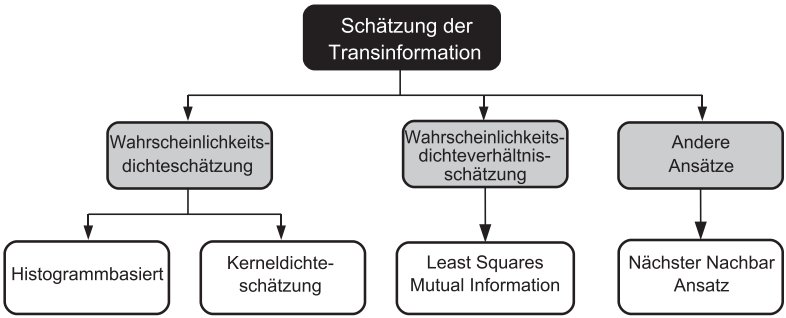


Abbildung 3.3.: Systematisierung der Verfahren zur Schätzung der Transinformation. In jeder der drei Gruppen lassen sich auch weitere Verfahren einordnen, die auch in diesem Abschnitt zumindest erwähnt werden. Als konkrete Verfahren sind nur jene benannt, die im Rahmen dieser Arbeit auch näher untersucht wurden.

Wahrscheinlichkeitsverteilungen schätzen, um solche Ansätze, die das Wahrscheinlichkeitsverteilungsverhältnis schätzen und andere Methoden, die auf der entropiebasierten Formulierung der Transinformation aufsetzen.

3.3.1. Verfahren zur Schätzung der Transinformation mittels direkter Wahrscheinlichkeitsdichteschätzung

Die Grundidee der Verfahren dieser Gruppe besteht darin, dass die notwendigen Verteilungen $p(x), p(y)$ und $p(x, y)$ direkt aus den Daten geschätzt werden. Die Umsetzung dieses intuitiven Ansatzes wird typischerweise entweder mittels Histogrammen oder einer Kerneldichteschätzung durchgeführt.

Histogrammbasierte Methoden

Histogramme sind die einfachste Form zur Schätzung der Wahrscheinlichkeitsverteilung, welche hierbei durch diskrete Fächer approximiert wird. Jede Achse unterteilt man in eine Anzahl i von nichtüberlappenden Fächern der Breite w_i und bestimmt die Anzahl n_i der Beobachtungen, die in dieses Fach fallen. Um daraus die Wahrscheinlichkeitsdichte $p(x)$ zu bestimmen, wird diese Anzahl durch die Breite der Fächer und die Gesamtzahl der Beobachtungen N geteilt.

Definition 3.12

HISTOGRAMMBASIERTE WAHRSCHEINLICHKEIT

Die Wahrscheinlichkeit für eine Ausprägung $p(x)$ die innerhalb des Faches i auftritt, ist konstant über die gesamte Breite des Faches und ergibt sich als

$$p_i = \frac{n_i}{N w_i}.$$

Dabei gilt $\int p(x)dx = 1$.

Die Verbundwahrscheinlichkeit $p(x, y)$ lässt sich ebenfalls auf diese Art und Weise berechnen. Dazu werden die Fächer in der zweidimensionalen XY-Ebene definiert und obige Formel angewendet. Damit ergibt sich $p_{ij} = \frac{n_{ij}}{N w_i w_j}$. Die Randverteilungen $p(x)$ und $p(y)$ lassen sich daraus durch einfache Marginalisierung bestimmen.

Definition 3.13

HISTOGRAMMBASIERTE TRANSINFORMATION

Die Transinformationsberechnung ergibt sich als

$$I(X;Y) = \sum_i \sum_j P_{ij} \log \left(\frac{P_{ij}}{P_i P_j} \right).$$

Dabei ist $P_i = p_i \cdot w_i$ (P_j analog) und $P_{ij} = p_{ij} \cdot w_i \cdot w_j$.

Die Transinformation wird hierbei nicht mehr über die einzelnen Datenpunkte bestimmt, sondern über die diskrete Verteilung in den Fächern des Histogramms.

Verbleibt die Frage nach der Wahl der Breite der Fächer w_i und damit auch nach der Anzahl der Fächer. Werden die Fächer zu breit gewählt, können die Eigenschaften der zugrundeliegenden Verteilung nicht genau genug approximiert werden, die Schätzung wäre dann übergeneralisiert und man spricht von einem hohen Bias-Fehler. Im gegenteiligen Fall, der Wahl zu kleiner Fachbreiten, würden viele leere oder nur spärlich besetzte Fächer auftreten und geringe Änderungen in der Datenbasis könnte die Approximation der Verteilung deutlich ändern. Dies wird als Overfitting bzw. Varianzfehler bezeichnet. Die korrekte Wahl der Breite ist demnach entscheidend, allerdings auch nicht trivial. Zur Behandlung dieses Bias-Varianz-Dilemmas[2] [Bishop, 2006] gibt es in der Literatur verschiedene Ansätze. Nachfolgend werde einige wichtige Verfahren vorgestellt im Kontext der Histogramme vorgestellt.

Histogramme mit einheitlicher Fachgröße Zunächst werden Fälle betrachtet in denen es einheitliche Fachgrößen gibt. Eine umfassende Übersicht über Regeln zur Wahl der Fachbreite findet sich in [Scott, 1992]. Zu den bekanntesten Ansätzen zählen Sturges Regel [Sturges, 1926], welche die erste publizierte Abschätzung war. Die Regel bestimmt dabei die Anzahl der zu verwendenden Fächer k aus der sich die Breite dann ableiten lässt:

[2] Das Problem des Bias-Varianz Dilemmas tritt nicht nur im Zusammenhang mit der Wahl der Fachbreite auf, sondern bei vielen Verfahren des Maschinellen Lernens, bei denen die Komplexität des lernenden Systems manipuliert wird. Ein zu einfaches System führt zu einem Bias-Fehler, diese Einschränkung ist systemseitig. Ein zu komplexes System variiert zu stark, da nicht genug Datenmaterial als Lernbeispiele zur Verfügung stehen, um alle wichtigen Kombinationen abzudecken. Diese Einschränkung ist dateninduziert. Wenn im weiteren Verlauf der Arbeit vom Bias gesprochen wird, sind immer die Einschränkungen des Systems gemeint.

$$k = \lceil 1 + \log_2(N) \rceil$$

Diese Regel findet weit verbreitete Anwendung auch in vielen Statistiksoftwarepaketen, allerdings gibt es Einschränkungen zu beachten [Scott, 2009]. Einerseits geht die Herleitung der Formel von normalverteilten Daten aus und andererseits funktioniert sie nur bei kleinen Datenmengen $N < 100$ zufriedenstellend. Für das erste Problem existieren Erweiterung wie beispielsweise Doanes Regel [Doane, 1976], die Zusatzterme für die Nichtgaußhaftigkeit der Verteilung einführen. Für das zweite Problem wird zumeist auf moderne Regeln verwiesen, etwa die Freedman-Diaconis Regel [Freedman und Diaconis, 1981], die Terrel-Scott Regel [Terrell und Scott, 1985] und die Regel nach Scott [Scott, 1979]. Für die letztgenannte Regel gibt es Untersuchungen, die zeigen, dass diese den Integrated Mean Square Error zwischen Approximation und wahrer Verteilung minimiert [Scott, 1992].

Definition 3.14

Regel nach Scott

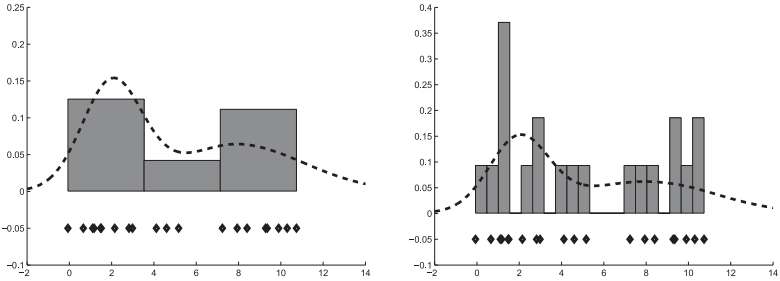
Die optimale Fachbreite w berechnet sich nach

$$w \approx 3.49\sigma N^{-1/3}.$$

N gibt dabei die Anzahl der verfügbaren Datenpunkte an und σ deren Standardabweichung.

Für die Hintergründe und eine Herleitung wird hier auf die Literatur verwiesen [Scott, 1979]. Ein eindimensionales Beispiel zur Schätzung mit Histogrammen ist in Abbildung 3.4 gezeigt.

Ensemble von Histogrammen mit einheitlicher Fachgröße Sogenannte Ensemble Methoden basieren auf der einfachen Annahme,



(a) (b)

Abbildung 3.4.: Beispiel für die Verwendung von Histogrammen zur Approximation der Wahrscheinlichkeitsdichte. Aus der originalen, bimodalen Verteilung (schwarz gestrichelt dargestellt) wurden Beispiele (schwarze Rhomben) gezogen. Auf Basis dieser Beispiele wird dann die Verteilung approximiert. **(a)** Hier wurde mittels der Regel von Scott die Breite der Fächer bestimmt, woraus drei Fächer resultieren. Das Resultat erhält die Bimodalität der originalen Verteilung. **(b)** Histogramm mit unnötig vielen Fächern. Die Charakteristik der Verteilung lässt sich kaum aus dem Histogramm ablesen.

dass durch Kombination von mehreren Ergebnissen unter bestimmten Bedingungen ein besseres Gesamtergebnis erreicht werden kann. Dabei können systematische Einschränkungen (Bias) der Einzelergebnisse überwunden und die Generalisierungsfähigkeit erhöht werden [Dietterich, 2000].

Überträgt man dieses Konzept auf die Bestimmung der Transinformation mittels Histogrammen ergibt sich die Hoffnung, dass Fehler, welche durch die falsche Wahl der Fachbreite entstehen, verringert werden können. Dazu wird die Transinformation mehrmals mit unterschiedlicher Fachbreite berechnet und daraus ein Mittelwert bestimmt. Es wird dabei auf die Regel von Scott (Definition 3.14) und einen Parameter λ zurückgegriffen, um die Größe des Ensembles n zu bestimmen. Dazu sei k_{Scott} die Zahl der Fächer, die für die Daten mittels der Re-

gel von Scott bestimmt wurden. Alle ganzzahligen Werte im Intervall $[\lceil k_{Scott}/\lambda \rceil, \lfloor k_{Scott} \cdot \lambda \rfloor]$ entsprechen einer Bestimmung der Transinformation mit der jeweiligen Anzahl an Fächern.

Definition 3.15

TRANSINFORMATION MIT EINEM ENSEMBLE VON HISTOGRAMMEN

Die Transinformation $I(X;Y)$ ergibt sich als Mittelwert der unterschiedlichen Transinformationsberechnungen $I_i(X;Y)$ mit unterschiedlichen Fachbreiten nach Definition 3.13

$$I(X;Y) = \frac{1}{n} \sum_{i=1}^{n} I_i(X;Y)$$

Die Anzahl der Histogramme n ist dabei abhängig von der berechneten Zahl nach Scott k_{Scott}, welche datenabhängig ist, sowie dem Parameter λ. In Untersuchungen hat sich gezeigt, das $1 < \lambda \leq 2$ ausreichend ist [KALTENHÄUSER, 2010]. Größere Werte bewirken kaum Änderungen am Ergebnis, erhöhen aber deutlich den Rechenaufwand.

Histogramme mit unterschiedlicher Fachgröße Eine andere Herangehensweise erlaubt unterschiedlich große Fachgrößen abhängig von der lokalen Datenverteilung. Dabei werden an Stellen mit wenigen Datenpunkten breite Fächer, also eine gröbere Auflösung, angestrebt, und umgekehrt in Bereichen mit vielen Datenpunkten werden die Fächer schmaler und damit die Auflösung der Approximation genauer.

Der bekannteste Ansatz aus dieser Gruppe stellt der in [FRASER und SWINNEY, 1986] vorgestellte Algorithmus dar. Daran orientieren sich alle weiteren Entwicklungen, wie beispielsweise [DARBELLAY und VAJDA, 1999] oder [CELLUCCI et al., 2005].

Die Grundidee dieser Algorithmen besteht darin, nicht alle Fächer gleich breit zu gestalten, wie es bisher der Fall war, sondern die Fächer

sollen alle annähernd dieselbe Wahrscheinlichkeit haben bzw. innerhalb der Fächer sollen die Daten möglichst gleichverteilt sein.

Dabei werden in [Fraser und Swinney, 1986] die Achsen rekursiv in zwei Hälften mit der gleichen Anzahl an Datenpunkten unterteilt, solange bis sich nur noch gleichverteilte Daten innerhalb eines jeden Faches befinden. Dieses Kriterium der Gleichverteilung wird dabei typischerweise mit Hilfe eines χ^2-Tests überprüft. Motiviert wird dieses Abbruchkriterium dadurch, dass die Fachrepräsentation selbst auch einer Gleichverteilung über der Fachbreite entspricht.

In der originalen Veröffentlichung werden dabei immer alle Fächer gleichzeitig geteilt, im Endergebnis erhält man also 2^i Fächer auf jeder Achse. [Darbellay und Vajda, 1999] entschärft dieses Vorgehen, in dem die weitere Unterteilung nicht von allen Fächern einer Achse abhängig gemacht wird, sondern vom Inhalt eines Faches selbst. Trotzdem bleibt es hier bei einem rekursiven Vorgehen.

Eine nicht rekursive Erweiterung stellt [Cellucci et al., 2005] vor. Hier wird die Partitionierung im Voraus berechnet, wobei als Kriterium die gleiche Anzahl an Datenpunkten pro Fach zugrunde gelegt wird.

Definition 3.16

Anzahl von Fächern nach Celluci

Die Anzahl der verwendeten Fächer k ergibt sich nach

$$k = \left\lfloor \sqrt{\frac{N}{5}} \right\rfloor .$$

N gibt dabei die Anzahl der verfügbaren Datenpunkte an.

Die Idee ist dabei, dass in jedem Fach mindestens fünf Datenpunkte liegen sollen - die Zahl fünf leitet sich dabei aus dem Cochrane-Kriterium [Cochran, 1954] her. Die Quadratwurzel ist damit zu erklären, dass

diese fünf Beispiele pro Fach im Verbundraum gelten sollen und daher in den Randverteilungen entsprechend die quadratische Menge aufweisen müssen. Die Aufteilung der Fächer wird dann auf den Randverteilungen so durchgeführt, dass in jedem Fach N/k Datenpunkte liegen. Sollten in jedem Fach exakt dieselbe Anzahl von Datenpunkten liegen, N also ein Vielfaches von k sein, kann die Transinformation wie folgt berechnet werden

$$I(X;Y) = \sum_i \sum_j P_{ij} \log(25 P_{ij}).$$

Ist dies nicht der Fall, kommt zur Berechnung wieder Definition 3.13 zur Anwendung, in welcher auch P_{ij} definiert wird.

Fazit Praktisch leicht umzusetzen, stellen Histogramme eine einfache Option zur Schätzung der Verteilungen dar. Jedoch verbleibt hier immer das Problem, dass es an den Übergängen von einem Fach zum anderen Unstetigkeiten gibt. Gerade in den Fällen, in denen viele Datenpunkte nahe den Fachgrenzen liegen, verändert beispielsweise eine geringfügige Verschiebung des Mittelpunkts aller Fächer die Wahrscheinlichkeitsschätzung deutlich. Eine andere Möglichkeit zur robusten, kontinuierlichen Schätzung der Verteilungsdichte wird als nächstes vorgestellt.

Kerneldichteschätzungsbasierte Methoden

Ein anderer Ansatz zur Bestimmung der Wahrscheinlichkeitsdichten ist die Schätzung mittels Kernelmethoden. Dazu werden Kernelfunktionen an die Positionen der Datenpunkte gelegt. Diese werden dann überlagert und normiert, um die Wahrscheinlichkeitsverteilung zu schätzen. Man kann sich diese Schätzung als Potentialfeld vorstellen, welches durch die Datenpunkte aufgespannt wird.

Während beim Histogramm einfach das Fach hochgezählt wird, in dem sich der Datenpunkt befindet, berücksichtigt dies nicht die Lage der Punkte innerhalb des Fachs. Man könnte die Kernelidee auch so interpretieren, dass nun jeder Datenpunkt sein eigenes Fach definiert und an allen Stellen innerhalb eines gewissen Umkreises um den Datenpunkt hochgezählt wird. Die Schätzung der Verteilung wäre dann also eine Summe von Rechtecken (Fächern) in die jeder Punkt der Datenverteilung mit genau einem Rechteck eingeht. In [Silverman, 1986] wird dies auch als *Naive Estimator* bezeichnet.

Definition 3.17

Kerneldichteschätzung

Allgemein ergibt sich die Wahrscheinlichkeitsdichte $p(x)$ als

$$p(x) = \frac{1}{Nh} \sum_{n \in N} K\left(\frac{x - x_n}{h}\right).$$

N gibt dabei die Anzahl der verfügbaren Datenpunkte an, K ist die gewählte Kernelfunktion und h der entsprechende Bandweiteparameter. x_n sind hier bei die n Positionen an denen sich die Kernelmittelpunkte befinden, in diesem Zusammenhang also die gegebenen Datenpunkte.

Für diesen einfachen Fall des Naive Esitmators würde man als Kernelfunktion ein entsprechendes Rechteck wählen

$$K_{Rechteck}(x) = \begin{cases} \frac{1}{2} & \text{falls } |x| < 1 \\ 0 & \text{sonst} \end{cases}$$

Dieser Kernel wird auch als uniformer Kernel bezeichnet. Es gibt dabei ein Vielzahl anderer Kernel, so beispielsweise den Dreieckskernel, den Cosinuskern oder den Epanechnikovkern. Für alle Kernelfunktionen müssen dabei zwei Eigenschaften erfüllt sein.

1. Die Kernelfunktion muss immer nichtnegativ sein.
   $K(x) \geq 0, \forall x \in [-\infty, \infty]$
2. Das Integral der Fläche der Kernelfunktion muss eins ergeben.
   $\int_{-\infty}^{\infty} K(x)dx = 1$

Praktisch gern eingesetzt wird der Gaußkernel. Er ist definiert als

$$K_{Gauss}(x) = \frac{1}{\sqrt{2\pi}} \exp^{-\frac{1}{2}x^2}.$$

Definition 3.18

KERNELDICHTESCHÄTZUNG MIT GAUSSKERN

Verwendet man nun diesen Gaußkern in der Definition der Kerneldichteschätzung (3.17), so erhält man

$$p(x) = \frac{1}{N} \sum_{n \in N} \frac{1}{\sqrt{2\pi}h} \exp\left(-\frac{(x - x_n)^2}{2h^2}\right).$$

N gibt dabei die Anzahl der verfügbaren Datenpunkte an, h der entsprechende Bandweiteparameter und x_n die Position des n-ten Datenpunktes.

Dies lässt sich wie folgt auf die zweidimensionale Verbundverteilung $p(x, y)$ übertragen:

$$p(x, y) = \frac{1}{N} \sum_{n \in N} \frac{1}{2\pi h^2} exp\left(-\frac{(x - x_n)^2 + (y - y_n)^2}{2h^2}\right).$$

Der Parameter h gibt dabei die Breite des Kernels an. Es handelt sich hierbei um das Äquivalent zur Fachbreite bei den Histogrammverfahren. Auch hierzu existieren Regeln die eine sinnvolle Wahl ermöglichen. Für den Gaußkern ist dies beispielsweise die Regel aus [SILVERMAN, 1986].

Definition 3.19

GAUSSKERNELBANDBREITE NACH SILVERMAN

Die optimale Bandbreite für einen Gaußkern h berechnet sich nach

$$h = \sigma \left(\frac{4}{d+2} \right)^{\frac{1}{d+4}} N^{-\frac{1}{d+4}}$$

N gibt dabei die Anzahl der verfügbaren Datenpunkte an, σ ist deren Standardabweichung und d die Dimensionalität der Daten.

Auch diese Regel beruht, wie die Regel von Scott (3.14), auf dem Ansatz, den Integrated Mean Square Error zwischen Approximation und wahrer Verteilung zu minimieren. Eine Übersicht zu anderen Möglichkeiten zur datengetriebenen Bandbreiteauswahl findet sich in [TURLACH, 1993]. Es existieren auch Verfahren mit variablen Bandbreiteparametern, allerdings werden diese aufgrund des damit verbundenen Rechenaufwands normalerweise nicht bei der Kerneldichteschätzung eingesetzt [MOON et al., 1995].

Zu beachten ist hierbei, dass diese Kernbandbreite nur einmal für die Verbundverteilung, also mit $d = 2$, bestimmt wird und dann so auch für die Randverteilungen $p(x)$ und $p(y)$ verwendet wird. Dies steht im Widerspruch zur Fachbreite bei den Histogrammen, in der jede Dimension einzeln eine optimale Breite erhalten kann.

Nun könnte die Transinformation mittels der Kerneldichteschätzung berechnet werden. [MOON et al., 1995] zeigt dabei, dass mittels der Kerneldichteschätzung unter Verwendung eines Gaußkerns bessere Ergebnisse erzielt werden, als das adaptive Histogramm-Verfahren aus [FRASER und SWINNEY, 1986].

Allerdings erfordert dieses Vorgehen eine numerische Integration der Integrale zur Berechnung der Transinformation (Definition 3.9), welche aufwendig ist. Praktisch macht man sich allerdings zunutze, dass

die Transinformation auf dem Mittelwert über der Verteilung basiert. Dieser Mittelwert wird dabei über die gegebenen Datenpunkte approximiert. Dadurch müssen nicht die vollständigen Verteilungen berechnet werden, sondern nur an den gegebenen Datenpunkten. Dieses Vorgehen findet sich in [Steuer et al., 2002] und in abgewandelter Form auch in [Principe et al., 2000]. Die Approximation ergibt sich als

$$\hat{I}(X;Y) = \frac{1}{N} \sum log_2 \frac{p(x_n, y_n)}{p(x_n)p(y_n)}.$$

Wie zu erkennen ist, wird hierbei wiederum nur über die Kernel an den gegebenen Datenpunkten summiert, zur Berechnung wird die Formel entsprechend Definition 3.18 eingesetzt.

Ein Beispiel für einen Kerneldichteschätzung und die Problematik der Bandbreiteschätzung ist in Abbildung 3.5 gezeigt.

3.3.2. Verfahren zur Schätzung der Transinformation mittels Wahrscheinlichkeitsverhältnisschätzung

Durch die Verrechnung (Produkt- und Quotientenbildung) der drei geschätzten Einzelwahrscheinlichkeiten $p(x), p(y)$ und $p(x, y)$ wird der Fehler der Approximation unter Umständen verstärkt. Daher wurde in [Suzuki et al., 2008a] und [Suzuki et al., 2008b] vorgeschlagen, das Wahrscheinlichkeitsverteilungsverhältnis $\frac{p(x,y)}{p(x)p(y)}$ direkt zu schätzen.

Grundidee ist dabei, dass das Wahrscheinlichkeitsverhältnis

$$\omega(x, y) = \frac{p(x, y)}{p(x)\, p(y)}$$

als Linearkombination von Basisfunktionen $\varphi(x, y)$ auszudrücken. Diese Basisfunktionen können dabei frei gewählt werden, es können also auch

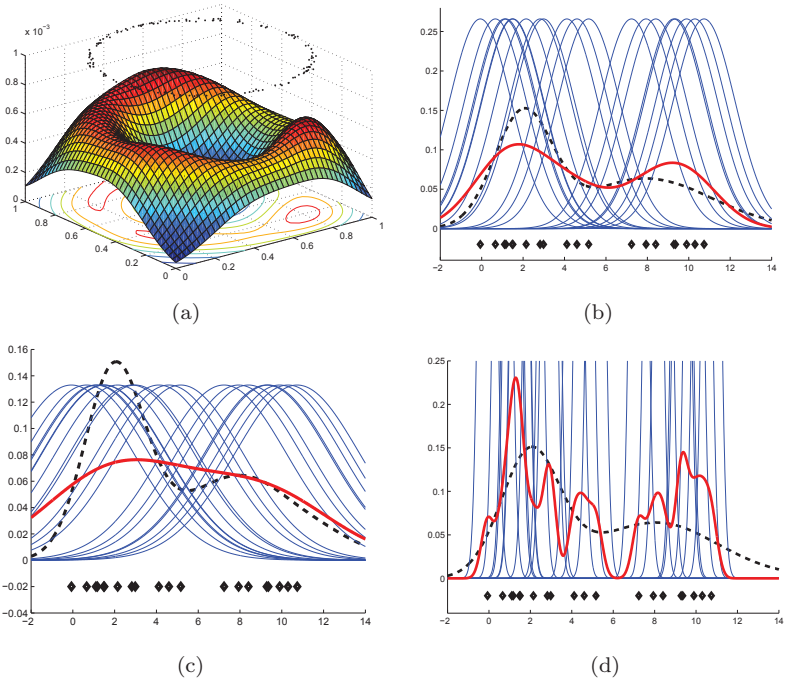


Abbildung 3.5.: **(a)** Beispiel für eine Kerneldichteschätzung in 2D bei einer Kreisförmigen Verteilung der Datenpunkte (Punktewolke im oberen Teil). Das dargestellte Gebirge ist dabei die Überlagerung der Gaußkerne, die an jedem dieser Datenpunkte liegen. **(b)-(d)** Zeigt eine Kerneldichteschätzung im eindimensionalen Fall. Es wird dieselbe Verteilung wie in Abbildung 3.4 verwendet. Über jedem gezogenen Datenpunkt werden die Gaußkerne (blaue Kurven) platziert. Die Mittelwertkurve dieser Kerne (rote Kurve) ist dann das Ergebnis der Schätzung. Für (b) ist $h = 1,5$, was nahe der Silverman-Regel liegt. Die Verteilung kann mittels der 20 Punkte einigermaßen gut approximiert werden. In (c) wurde $h = 3$ gewählt, und es zeigt sich, dass die Glättung zu groß ist, als dass die Charakteristik der Verteilung erhalten bliebe. Bei (d) ist mit $h = 0.3$ die Generalisierung hingegen nur unzureichend gegeben, es existieren zu viele Extrempunkte im Funktionsverlauf der Schätzung.

wieder Kernelfunktionen zum Einsatz kommen. Jedoch sind die Kerneleigenschaften hier keine notwendigen Eigenschaften, die diese Basisfunktionen erfüllen müssen.

Das approximierte Wahrscheinlichkeitsverhältnis $\hat{\omega}(x, y)$ wird somit als

$$\hat{\omega}_\alpha(x, y) := \underline{\alpha}^T \varphi(x, y)$$

dargestellt.

Als Basisfunktionen werden wieder Gaußkerne (siehe Definition 3.18) verwendet. Ihre Positionierung im Raum erfolgt jedoch vergleichsweise aufwendig durch ein Kreuzvalidierungsverfahren. Basierend auf den Datenpunkten wird dann der Vektor $\underline{\alpha}$ ermittelt, der die linearen Anteile der Basisfunktionen am Dichteverhältnis darstellt. Die beiden vorgeschlagenen Möglichkeiten dies zu tun, basieren auf der Optimierung entweder der Maximum Likelihood oder des quadratischen Fehlers. Der erste Ansatz sucht nach der wahrscheinlichsten Kombination der Basisfunktionen, die mittels eines Expectation-Maximization Algorithmus bestimmt wird [Suzuki et al., 2008a]. Der zweite Ansatz minimiert den quadratischen Fehlers zwischen Approximation und wahrem Quotienten [Suzuki et al., 2008b]. In dieser Arbeit wird dem zweiten Vorschlag gefolgt, da diese Formulierung dem Integrated Mean Square Error der Dichteverhältnisse entspricht, und somit eine Analogie zu den Kriterien der Regel von Scott(3.14) und auch der Bandbreite nach Silverman(3.19) darstellt.

Das Finden der Linearkombinationen $\underline{\alpha}$ wird durch Minimierung der folgenden Kostenfunktion J_0 realisiert.

$$J_0(\alpha) = \frac{1}{2} \int_x \int_y (\hat{\omega}_\alpha(x, y) - \omega(x, y))^2 p(x)\, p(y)\, dx\, dy.$$

Diese Gleichung beschreibt den Abstand der Schätzung des Wahrscheinlichkeitsverhältnisses vom wahren Verhältnis als gewichteter,

quadratischer Fehler. Da für die Berechnung von J_0 jedoch das reale Verteilungsverhältnis bekannt sein müsste, welches bestimmt werden soll, wird stattdessen folgende Approximation der Kostenfunktion verwendet:

$$\hat{J}(\alpha) = \sum_{(x,y)\in Z} \frac{\hat{\omega}_\alpha(x,y)^2}{2N^2} - \sum_{(x,y)\in Z} \frac{\hat{\omega}_\alpha(x,y)}{N}$$

Folgt man dabei der nicht-trivialen Herleitung in [Suzuki et al., 2008b], welche hier nicht wiedergegeben werden soll, geschieht dies durch

$$\alpha = \left(\frac{1}{N^2} \sum_{i,j=1}^{N} \left(\varphi(x_i, y_j) \varphi(x_i, y_j)^T \right) + \lambda I_b \right)^{-1} \frac{1}{N} \sum_{i=1}^{N} \varphi(x_i, y_i).$$

Dabei entspricht b der Anzahl der Basisfunktion, I_b ist die b-dimensionale Einheitsmatrix und λ ein Regularisierungsparameter.

Wie bereits beschrieben werden die Basisfunktionen per Kreuzvalidierung ermittelt. Dieses Verfahren ermöglicht es weiterhin, zusätzliche Parameter zu schätzen, namentlich die Regularisierung λ oder den Bandbreiteparameter h für die Basisfunktionen. Für die Wahl der Anzahl der zu verwendenden Basisfunktionen wird in [Suzuki et al., 2008b] 200 empfohlen, oder entsprechend weniger, für den Fall, dass weniger als 200 Datenpunkte zur Verfügung stehen.

Die Kreuzvalidierung erfolgt, indem die Kostenfunktion J_0 für r disjunkte Teilmengen der Daten berechnet wird. Das Mittel daraus ist ein Maß für die Güte der gewählten Parameterkonstellation von Basisfunktionen und Regularisierung. Dies wird für alle Kandidatenfunktionen wiederholt. Je niedriger der Wert der Kostenfunktion, desto besser ist die Güte der Approximation.

Durch die notwendige Kreuzvalidierung handelt sich bei diesem Verfahren auch um den aufwendigsten, der hier vorgestellten Vertreter zur Schätzung der Transinformation.

3.3.3. Andere Schätzmethoden

Es gibt weitere Ansätze zur Schätzung der Transinformation, welchen gemein ist, dass sie nicht auf der Kullback-Leibler-Divergenz Formulierung beruhen, sondern auf der originalen Formulierung über die Entropie. Beispiele sind dabei die Edgeworth-basierte Schätzung [Van Hulle, 2005] oder das Nächste-Nachbar-Verfahren [Kraskov et al., 2004]. Bei letzterem Verfahren handelt es sich um den aktuellen de facto Standard zur Schätzung von Transinformation und daher soll dieses etwas näher betrachtet werden.

Die Grundidee des Nächsten-Nachbar-Verfahrens besteht darin für jeden Punkt die Anzahl von Nachbarn in jeder Dimensionen zu zählen, und mittels dieser Information auf die Entropie und dadurch auf die Transinformation zu schließen. Grafisch ist diese Idee in Abbildung 3.6 angedeutet.

Dabei basiert die Formulierung des Nächsten-Nachbar-Ansatzes auf dem Kozachenko-Leonenko Schätzer für Entropie [Kozachenko und Leonenko, 1987].

Definition 3.20

Nächster-Nachbar-Schätzer für Entropie

Die Schätzung der Entropie erfolgt dabei nach folgender Formel

$$H(X) = -\frac{1}{N}\sum_{i=1}^{N}\psi(n_x(i)) - \frac{1}{k} + \psi(N) + log\, c_{d_x} + \frac{d_x}{N}\sum_{i=1}^{N} log\, \epsilon(i).$$

Die Summe wird dabei über alle N Datenpunkte gebildet. Dabei ist k die Anzahl der verwendeten Nächsten-Nachbarn, also ein frei-

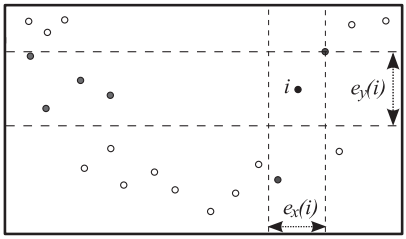


Abbildung 3.6.: Grundidee des Nächsten-Nachbar Verfahrens. Für Punkt i wird der nächste Nachbar bestimmt. Dieser definiert nun für jede Dimension einen Schlauch $e_{x/y}(i)$ für den bestimmt wird, wie viele andere Punkte sich innerhalb dieses Schlauches befinden. Damit ergibt sich $n_x(i) = 1$ und $n_y(i) = 4$. Dies kann dann in Gleichung 3.21 eingesetzt werden und wird für jeden Datenpunkt wiederholt, was der Summe in der Gleichung entspricht. Darstellung in Anlehnung an [Kraskov et al., 2004].

er Parameter. Dieser Parameter spannt damit den Schlauch auf, in dem benachbarte Punkte n_x gezählt werden. d_x ist die Dimensionalität der Zufallsgröße X und c_x das Volumen der Einheitskugel im d_x-dimensionalen Raum. ψ ist dabei die Digammafunktion mit $\psi(x) = \Gamma(x)^{-1} d\Gamma(x)/dx$. Die Berechnung erfolgt rekursiv nach $\psi(x+1) = \psi(x) + 1/x$ bei $\psi(1) = \gamma$, wobei γ die Euler-Mascheroni Konstante ist. Weiterhin bezeichnet $\epsilon(i)$ die Maximumsdistanz von Punkt i zu seinem k-ten Nachbarn.

Die Herleitung dieser Formel ist sehr umfangreich und kann in [Kozachenko und Leonenko, 1987] und [Kraskov et al., 2004] nachgelesen werden. Eine intuitive Interpretation dieses mathematischen Zusammenhangs ist dabei leider nicht möglich.

Die Idee aus [Kraskov et al., 2004] besteht nun darin, diese Entropieschätzung auf die Verbundentropie $H(X, Y)$ zu erweitern und dies dann zur Berechnung der Transinformation nach Definition 3.8 ($I(X;Y) = H(X) + H(Y) - H(X,Y)$) zu verwenden. Dabei wurde darauf Wert

gelegt, dass die Approximationsfehler der drei Teilterme sich möglichst aufheben und so eine genauere Gesamtschätzung ermöglichen.

Definition 3.21

NÄCHSTER-NACHBAR-SCHÄTZER FÜR TRANSINFORMATION

Die Schätzung der Transinformation nach [KRASKOV et al., 2004] ergibt sich als

$$I(X;Y) = \psi(k) - \frac{1}{k} - \frac{1}{N}\sum_{i=1}^{N}[\psi(n_x(i)) + \psi(n_y(i))] + \psi(N).$$

Neu sind hierbei die Größen n_x und n_y. Diese Zählen die Anzahl von Punkten, die innerhalb eines Schlauches um den aktuellen Datenpunkte herum liegen. Die Breite des Schlauches wird dabei durch die Nächsten-Nachbarn in dieser Dimension definiert. Zur Verdeutlichung sei noch einmal auf Abbildung 3.6 verwiesen.

3.3.4. Verbundtransinformation

Bisher wurde nur auf die Frage eingegangen, inwieweit sich die Transinformation zwischen einem Eingangskanal und den Zielwerten schätzen lässt. Allerdings ist es oft notwendig, gerade bei Berücksichtigung von Redundanzen, die Frage zu stellen, welche Information mehrere Eingangskanäle über das Ziel haben. Dazu wurde bereits die Verbundtransinformation definiert (siehe Definition 3.10).

Bei der Übertragung der vorgestellten Schätzverfahren auf diese höherdimensionale Problematik gibt es ein Hindernis, für welches der Begriff *Fluch der Dimensionalität* von Bellmann geprägt wurde [BELLMAN, 1957]. Es beschreibt die Problematik, dass das Hinzufügen einer Dimension in einem mathematischen Raum dazu führt, dass das Volumen dieses Raumes exponentiell wächst. Für die Schätzung

von Wahrscheinlichkeiten bedeutet dies, dass exponentiell mehr Datenpunkte einer Verteilung benötigt werden. Wenn für ein Histogramm im Mittel fünf Datenpunkte in jedem Fach liegen sollen und pro Dimension zehn Fächer existieren, wären für den eindimensionalen Fall 50 Datenpunkte ausreichend. Für den vierdimensionalen Fall benötigt man bereits 50000 Datenpunkte und verallgemeinert $10^d \cdot 5$ Punkte um dieselbe Abdeckung zu erreichen.

Praktisch stehen nur selten hinreichend viele Datenpunkte zur Verfügung und es kommt damit zu spärlichen Verteilungen der Datenpunkte, die eine korrekte Schätzung der zugrundeliegenden Wahrscheinlichkeitsverteilung nicht nur erschweren sondern oft ganz unmöglich machen. Diese Problem betrifft sowohl die histogrammbasierten Verfahren, die Kerneldichteschätzung wie auch die Wahrscheinlichkeitsverhältnisschätzung. Die entropiebasierten Schätzer aus Abschnitt 3.3.3 sind nach den Aussagen in [Van Hulle, 2005] und [Kraskov et al., 2004] diesbezüglich etwas resistenter, haben aber grundsätzlich mit demselben Problem zu kämpfen.

Es existieren jedoch auch Approximationsverfahren, die auf Basis niedrig dimensionaler Transinformationsschätzung auf die Verbundtransinformation schließen. Ein solches Verfahren im Kontext der Merkmalsselektion wurde in [Battiti, 1994] vorgestellt. Bei diesem *Mutual Information for Feature Selection* (MIFS) Verfahren wird auf die paarweise Transinformation zwischen den Eingangsvariablen untereinander zurückgegriffen. Auch zu diesem Verfahren existieren Erweiterungen, deren Bestreben es ist, die Approximation zu verbessern, so zum Beispiel [Kwak und Choi, 1999] oder [Estevez et al., 2009]. Allerdings wird in dieser Arbeit der originale Ansatz von Battiti betrachtet.

Der Algorithmus berechnet dazu für jedes Merkmal einen sogenannten MIFS-Wert. Dieser entspricht der Transinformation zwischen einer Eingangsvariable und dem Ziel abzüglich der Summe über alle paarweisen Transinformationswerte zwischen dem Kandidatenmerkmal X und

allen bereits gewählten Eingangskanälen S. In dieser Summe stecken damit die paarweisen Redundanzen der Merkmale untereinander.

Definition 3.22

MUTUAL INFORMATION FOR FEATURE SELECTION

Der MIFS-Wert nach [BATTITI, 1994] ergibt sich als

$$MIFS(X) = I(X;Y) - \beta \sum_{S \in Subset} I(X;S).$$

S bezeichnet dabei eine Eingangsvariable, die bereits gewählt wurde und sich demzufolge in der Auswahlmenge befindet. β ist ein freier Parameter und gibt den Einfluss der bereits gewählten Auswahlmenge an. Er gewichtet den Einfluss der redundanten Informationen.

Die Merkmalsselektion läuft dann nach dem einfachen Rankingprinzip mit einer Vorwärtssuchstrategie ab. Es wird für jeden Eingangskanal der MIFS-Wert berechnet und das Merkmal mit dem höchsten Wert wird der Auswahlmenge hinzugefügt. Danach beginnt eine neue Runde zur Berechnung des MIFS-Wertes, da sich der zweite Teil des Terms mit dem neugewählten Merkmal geändert hat. Wird der Parameter $\beta = 0$ gesetzt erhält man die klassische Merkmalsauswahl bei der nacheinander jeweils das Merkmal mit der maximalen Transinformation zum Ziel gewählt wird. Typischerweise wird $0.1 \leq \beta \leq 0.3$ gewählt. Eine Darstellung als Pseudocode erfolgt in Algorithmus 1.

3.3.5. Experimentelle Untersuchungen

Ziel dieses Abschnittes ist es, die verschiedenen Verfahren, die in den vorangegangenen Abschnitten vorgestellt wurden, zu untersuchen, um Aussagen über ihre Tauglichkeit im Rahmen der Merkmalsauswahl zu treffen. Dazu werden zwei Aspekte betrachtet: Erstens die Approximationsgüte der Transinformation, wobei hier die Experimente aus

Algorithmus 1 MI FOR FEATURE SELECTION(X, Y, β)

Eingabe: Datensatz von Beobachtungen X, die entsprechenden Labels Y, Redundanzwichtungsfaktor β
Ausgabe: Merkmalsteilmenge S

$S \leftarrow \emptyset$ {Initiale Merkmalsmenge sei leer.}
repeat
  for $\forall x_i \in X \setminus S$ **do**
    $m(x_i) = I(x_i; Y) - \beta \sum_{s \in S} I(x_i; s)$
  end for
  $S \leftarrow S \cup \arg\max_{x_i}(m)$ {Aufnahme des besten Merkmals in die Auswahlmenge}
until $\max(m) \leq 0$ oder $|S|$ hat festgelegte Anzahl erreicht

[KHAN et al., 2007] nachvollzogen und um neue Verfahren erweitert wurden. Der zweite Aspekt beschäftigt sich mit der Nützlichkeit für den Merkmalsselektionsprozess.

Approximationsgüte

In [KHAN et al., 2007] wurden verschiedene Verfahren zur Transinformationsschätzung miteinander experimentell verglichen. Besonderes Augenmerk legten die Autoren dabei auf die Eignung für den Fall das nur wenige, verrauschte Daten zur Schätzung zur Verfügung stehen. Dazu wurden drei Funktionen (linear, quadratisch und trigonometrisch-periodisch) definiert, für welche die wahre Transinformation analytisch berechnet werden kann. Wie diese wahre Transinformation bestimmt werden kann, ist ausführlich im Anhang von [KHAN et al., 2007] beschrieben. Die Grundidee leitet sich daraus ab, dass für einen einfachen gegebenen funktionalen Zusammenhang, die wahren Entropien $H(Y)$ und $H(Y|X)$ analytisch (im linearen Fall) oder durch numerische Integration (im quadratischen und periodischen Fall) bestimmt werden

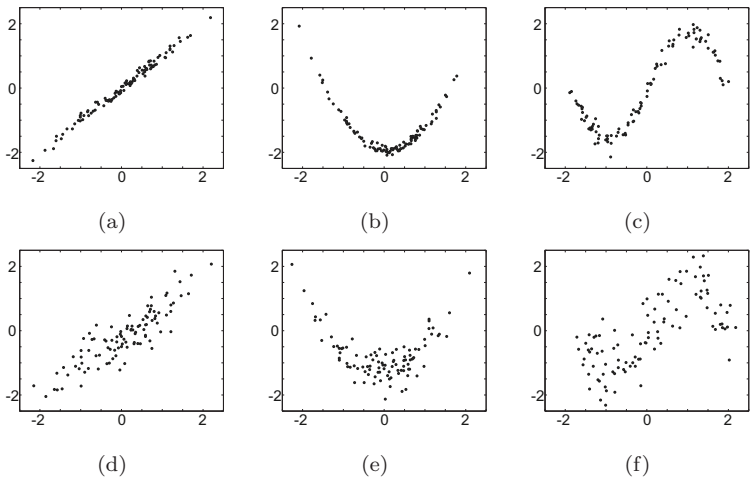


Abbildung 3.7.: Funktionen mit denen die Schätzung der Transinformation getestet wurde. Erste Spalte (a) und (d) linear, zweite Spalte (b) und (e) quadratischer Zusammenhang und dritte Spalte (c) und (f) trigonometrisch-periodisch. Obere Zeile (a)-(c) 10% Rauschen. Untere Zeile (d)-(f) 50% Rauschen.

kann.

Von diesen wurden dann verrauschte Beispiele gezogen, welche den Verfahren als Eingaben dienten. Die Zusammenhänge sind in Abbildung 3.7 gezeigt.

Die untersuchten Verfahren waren dabei die Kerneldichteschätzung, der Histogrammansatz von Cellucci (beide siehe Abschnitt 3.3.1), der Nächster-Nachbar Ansatz (siehe Abschnitt 3.3.3), sowie die Edgeworth Erweiterung von [Van Hulle, 2005] und andere Spielarten der adaptiven Histogramme. Das Ergebnis dieser Untersuchungen zeigte zwei überlegene Verfahren, namentlich den Nächsten-Nachbar Ansatz bei wenig verrauschten Daten und die Kerneldichteschätzung bei stärker verrauschten Daten.

Abhängigkeit	linear		quadratisch		periodisch	
$\sigma_{\epsilon}/\sigma_S$	0.1	0.5	0.1	0.5	0.1	0.5
Histogramm	1.49	0.82	0.90	0.67	0.96	0.57
Cellucci	1.07	0.53	0.55	0.31	0.58	0.38
Ensemble	1.40	**0.81**	0.90	0.67	0.92	**0.54**
KDE	1.49	0.85	0.97	0.73	1.05	0.58
LSMI	2.40	0.85	1.50	0.73	1.29	0.450
KNN	**2.25**	0.77	**1.84**	**0.77**	**1.72**	0.64
Wahre MI	**2.31**	**0.80**	**1.98**	**0.79**	**1.70**	**0.53**

Tabelle 3.1.: Transinformation bei N=100 Datenpunkten. Fett dargestellt ist das für jede Spalte am nächsten zur wahren Transinformation liegende Ergebnis.

Diese Untersuchungen wurden im Rahmen der Bachelorarbeit von Robert Kaltenhäuser [Kaltenhäuser, 2010] nachvollzogen und um das Ensemble von Histogrammen (siehe Abschnitt 3.3.1) und das Least Squares Mutual Information Verfahren (siehe Abschnitt 3.3.2) erweitert. Ein Ausschnitt aus den Ergebnissen ist in den Tabellen 3.1 und 3.2 dargestellt.

Im Wesentlichen wurden dabei die Ergebnisse von [Khan et al., 2007] bestätigt. Bei den Testdaten der drei Funktionen mit wenig Rauschen ($\sigma_{Rauschen}/\sigma_{Signal} = 0.1$) kam der Nächste-Nachbar-Ansatz zu den besten Ergebnissen. Bei starkem Rauschen ($\sigma_{Rauschen}/\sigma_{Signal} = 0.5$) konnte bei wenigen Datenpunkten das Ensembleverfahren seine Stärken ausspielen, während die Kerneldichteschätzung bei vielen Datenpunkten und viel Rauschen überzeugte. Aber auch der Nächste-Nachbar Ansatz kam zu sehr guten Ergebnissen.

Eine abschließende Empfehlung zu geben, welches das zu bevorzugende Verfahren ist, gestaltet sich schwierig. Zwei der besten Verfahren, die Kerneldichteschätzung und der Nächste-Nachbar-Ansatz, stellen näm-

Abhängigkeit	linear		quadratisch		periodisch	
σ_ϵ/σ_S	0.1	0.5	0.1	0.5	0.1	0.5
Histogramm	2.20	0.85	1.75	0.83	1.59	0.57
Cellucci	2.19	0.85	1.73	0.80	1.68	0.60
Ensemble	2.16	0.85	1.70	0.83	1.55	0.57
KDE	2.07	0.82	1.50	**0.79**	1.45	**0.53**
LSMI	3.83	0.83	2.08	0.71	2.11	0.46
KNN	**2.32**	**0.80**	**1.99**	0.79	**1.71**	0.53
Wahre MI	**2.31**	**0.80**	**1.98**	**0.79**	**1.70**	**0.53**

Tabelle 3.2.: Transinformation bei N=10000 Datenpunkten. Fett dargestellt ist das für jede Spalte am nächsten zur wahren Transinformation liegende Ergebnis.

lich zwei Extrema im Sinne des Bias-Varianz-Dilemmas dar. Während der Nächste-Nachbar Ansatz so gut wie keinen Bias aufweist, zeigt sich bei Versuchen mit viel Rauschen, dass hier die Tendenz zur Überanpassung gegeben ist. Umgekehrt neigen Kerneldichteschätzer zu einem hohen Bias [Rajagopalan et al., 1997], was sich in Fehlern bei geringem Rauschen niederschlägt. Jedoch zeigt dieser Schätzer eine gute Generalisierung, wenn es um Daten mit viel Rauschen geht.

Auch darf nicht außer Acht gelassen werden, dass beide Verfahren je einen Parameter besitzen, der es ermöglicht diese Extrema aufzuweichen. So führt beim Nächsten-Nachbar Ansatz die Verwendung von mehr Nachbarn zu einer besseren Generalisierung, während die Wahl einer sehr schmalen Kernelbandbreite h bei der Kerneldichteschätzung den Bias verringert. Jedoch zeigt sich, dass dies sich immer auch zu Ungunsten der Approximationsgüte niederschlagen kann.

Ergebnisse im Rahmen der Merkmalsselektion

Jedoch ist für eine erfolgreiche Merkmalsselektion der korrekte Wert der Transinformation nur zweitrangig. Wichtiger ist bei den Auswahlverfahren, dass die approximierten Transinformationswerte im korrekten Verhältnis zueinander stehen. Die Arbeitshypothese für die durchgeführten Untersuchungen war dabei, dass sich im Verhältnis der geschätzten Werte eventuelle systematische Fehler aufheben und somit auch Verfahren, welche nicht die genausten Approximationen der Transinformation erreichen, nützlich für die Merkmalsselektion sein können. Sollte diese Hypothese falsch sein, müsste sich ein qualitativ ähnliches Bild wie in den vorangegangenen Experimenten ergeben. Das heißt, es müssten klare Vorteile für die Kerneldichteschätzung und das Nächste-Nachbar-Verfahren erkennbar sein.

Diese Hypothese wurde wie folgt getestet. Für mehrere Datensätze aus dem UCI Machine Learning Repository [Asuncion und Newman, 2007] wurden mit den vorgestellten Verfahren die Transinformation geschätzt, wobei die MIFS Approximation (3.22) zum Einsatz kam. Beim Nächsten-Nachbar-Ansatz (3.21) wurde zusätzlich die originäre Verbundtransinformation bestimmt, da die Literatur hier Vorteile für dieses Verfahren sieht. Basierend auf diesem MIFS Ranking bzw. der Verbundtransinformation (mittels einer Vorwärtsstrategie, wie in [Kwak und Choi, 2002] beschrieben) wurden dann die m besten Merkmale ausgewählt. Zusätzlich zu den beschriebenen Verfahren wurde eine zufällige Auswahl von Merkmalen aufgenommen und bewertet, wobei diese über zehn Versuche gemittelt wurden.

Mit Hilfe eines einfachen Nächsten-Nachbar-Klassifikators und Kreuzvalidierung wurde dann die Klassifikationsgüte in Form der Balanced Error Rate[3] bestimmt. Diese dient dabei als Maß für die Güte der Merk-

[3]Diese ergibt sich als $BER = \frac{1}{2}\left(\frac{FN}{FN+TP} + \frac{FP}{FP+TN}\right)$. Dabei ist FN die Anzahl

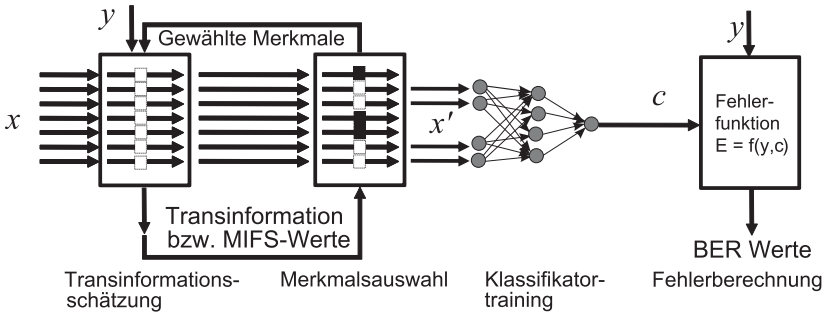


Abbildung 3.8.: Schematische Darstellung des Ablauf der Untersuchung. Im ersten Block werden die Transinformationswerte bzw. die MIFS-Werte (welche die bereits gewählten Merkmale mit beachtet, daher die Rückkopplung im Bild) bestimmt - während danach die eigentliche Auswahl stattfindet. Mit den gewählten Merkmalen kann dann ein Klassifikator trainiert werden. Dessen finaler Fehler, der über Kreuzvalidierung bestimmt wird, dient zur Bewertung der Güte der selektierten Merkmale.

malsselektion. Schematisch ist der Ablauf in Abbildung 3.8 zu sehen. Zahlenmäßig sind die Ergebnisse in Tabelle 3.3 dargestellt.

Das Auftreten von gleichen Fehlerraten ist ein Zeichen dafür, dass dann von den unterschiedlichen Methoden dieselben Merkmale während des Selektionsprozesses ausgewählt wurden. Dies bedeutet allerdings nicht, dass diese Merkmale auch in derselben Reihenfolge hinzugefügt wurden. Was man in der Tabelle erkennen kann, ist, dass jedes Verfahren mindestens einmal das beste Ergebnis erzielt. Allerdings ist auch jede Methode auf anderen Datensätzen mitunter deutlich schlechter als andere Ansätze, aber immer besser als die zufällige Merkmalsselektion mit Ausnahme des Cellucci-Histogramm Ansatzes beim *Hearts* Datensatz, welches eine schlechtere Lösung anbot, als die zufällig gewählte.

falsch negativ klassifizierter Beispiele, FP die Zahl falsch positiver Beispiele und analog dazu sind TP und TN die korrekten Ergebnisse für die Positiv- und Negativklasse.

Methode	IS	GC	BC	PK	HR
Histogramm	0.099	0.379	**0.046**	0.16	0.368
Ensemble	0.119	0.379	**0.046**	0.16	0.375
Celluci	0.101	**0.36**	0.0639	0.092	0.455
KDE	0.119	0.369	**0.0463**	0.158	0.375
LSMI	**0.082**	0.369	0.0548	0.136	0.362
KNN	0.113	0.396	0.0632	**0.068**	0.407
KNNJ	0.143	0.387	0.0775	0.163	**0.351**
Zufall	0.227	0.436	0.0854	0.162	0.415

Tabelle 3.3.: Ergebnisse der Experimente. Dargestellt ist die Balanced Error Rate für die Auswahl von $m = 8$ Merkmalen und einem 1-Nächster-Nachbar Klassifikator. Fett hervorgehoben sind die jeweils besten Ergebnisse pro Datensatz in jeder Spalte. Die verwendeten Abkürzungen für die Verfahren stehen dabei für: KDE - Kerneldichteschätzung, LSMI - Least Squares Mutual Information / Wahrscheinlichkeitsverhältnisschätzung, KNN - Nächster-Nachbar Schätzer, KNNJ - Nächster-Nachbar Schätzer für Verbundtransinformation. Die Abkürzungen in der Kopfzeile benennen den verwendeten Datensatz: IS - Ionosphere, GC - german Credit, BC - breast cancer, PK - Parkinsion, HR - Hearts.

Tabelle 3.4 zeigt den Sachverhalt als Rangliste der Verfahren. Formuliert man basierend auf dieser Tabelle eine Funktion, welche den Rang eines Verfahrens für die unterschiedlichen Datensätze aufsummiert, so lässt sich damit eine vorsichtige Abschätzung der Brauchbarkeit der Verfahren erreichen. Dieses Ergebnis ist in Tabelle 3.5 dargestellt. Dabei fällt auf, dass beispielsweise sowohl das Ensemble von Histogrammen, als auch die adaptiven Histogrammfächer nach Celluci schlechter abschneiden, als die einfache Form mit der Fachbreitenwahl nach Scott.

Am schlechtesten abgeschnitten hat der Nächste-Nachbar Ansatz zur direkten Verbundtransinformationsberechnung - ein Ansatz von dem nach diesen Ergebnissen eher abzuraten ist. Auch die Variante des Nächsten-Nachbar Ansatzes, welcher mittels MIFS die Merkmalsselek-

	IS	GC	BC	PK	HT
1	LMSI	Celluci	Hist/Ens KDE	KNN	KNNJ
2	Hist	KDE / LMSI	-	Celluci	LSMI
3	Celluci	-	-	LSMI	Hist
4	KNN	Hist/Ens	LSMI	KDE	KDE/Ens
5	Ens/KDE	-	KNN	Hist/Ens	-
6	-	KNNJ	Celluci	-	KNN
7	KNNJ	K-NN	KNNJ	KNNJ	Celluci

Tabelle 3.4.: Rangliste der Schätzverfahren basierend auf Tabelle 3.3. Die benutzten Abkürzungen entsprechen ebenfalls denen aus der vorhergehenden Tabelle (Tab. 3.3).

tion durchführt, fällt trotz exzellenter Approximationsgüten hinter die anderen Verfahren zurück. Andererseits zeigt sich im Least Squares Schätzer ein gutes Verfahren zur Merkmalsselektion, auch wenn die Ergebnisse bei der Approximationsgüte durchwachsen waren (siehe Tabelle 3.1 und 3.2). Sowohl die einfache Histogramm-Schätzung als auch die Kerneldichteschätzung konnten bei der Merkmalsselektion überzeugen.

Um einen Einfluss des Bias des verwendeten Nächsten-Nachbar Klassifikators auszuschließen, wurden die Untersuchungen mit einem mächtigeren Klassifikator, einem mehrschichtigen neuronalen Netz wiederholt. Hierbei zeigten sich sehr ähnliche Ergebnisse. Die Eingangs aufgestellte Hypothese, dass der absolute Approximationsfehler bei der Schätzung der Transinformation zweitrangig ist, muss als zutreffend gewertet werden, da sich doch ein gänzlich anderes Bild als bei der Approximationsgüte ergibt.

	Verfahren	Punkte
1	Least Squares Mutual Information	12
2	Histogramm	15
3	Kerneldichteschätzung	16
4	Ensemble von Histogrammen	19
	Celluci	19
6	Nächster Nachbar Klassifikator	23
7	Verbundtransinformation mit k-NN	28

Tabelle 3.5.: Summe über die erzielten Ränge der Schätzverfahren, welche in Tabelle 3.4 erzielt wurden. Eine geringere Punktzahl ist dabei besser.

3.3.6. Schlussfolgerungen

Welches Schätzverfahren zur Bestimmung der Transinformation sollte im Rahmen der Merkmalsselektion verwendet werden?

Wie die Ausführungen gezeigt haben, gibt es nicht ein überlegenes Verfahren, sondern die optimale Wahl ist problemabhängig. Dieses empirische Ergebnis könnte man unter Umständen als Ausprägung des No-Free-Lunch-Theorems [Wolpert, 1996] interpretieren, d.h. dass gemittelt über die Menge aller möglichen Datenverteilungen, die Verfahren ohne Verwendung von Apriori-Informationen alle gleich gut abschneiden.

Sofern also die Möglichkeit gegeben ist, kann mittels einer Kreuzvalidierung das beste Verfahren gewählt werden. Allerdings rechtfertigt der zu erwartende Gewinn in den meisten Fällen wohl nicht den notwendigen Aufwand für diese Auswahl.

Die Empfehlung, die aus den Untersuchungen abgeleitet wird, ist es, den Kerneldichteschätzer zu verwenden. Dies motiviert sich durch sehr gute Ergebnisse sowohl beim Approximieren der wahren Transinformation, als auch der Merkmalsselektion. Weiterhin handelt es sich um ein, im

Vergleich zur LSMI, einfaches Verfahren, so dass hier das Argument von Occam's Razor zu Gunsten des Kerneldichteschätzers angebracht werden könnte.

Nach dieser Wahl der Kerneldichteschätzung als geeignetes Instrument zur Schätzung der Transinformation soll nun im weiteren Verlauf diskutiert werden, an welcher Stelle diese Größe sinnvoll zur Merkmalsselektion eingesetzt werden kann.

3.4. Transinformation und Wrapper-Verfahren

Bisher wurde die Transinformation als einfaches Relevanzkriterium verwendet, um damit ein Merkmalsranking durchzuführen. Dabei wurden bereits einfach Möglichkeiten angesprochen die Verbundtransinformation zu berücksichtigen [Battiti, 1994] [Kwak und Choi, 1999]. Es existieren jedoch etliche weitere Ansätze, die die Transinformation oder verwandte Spielarten im Rahmen eines Filterfahrens zur Merkmalsselektion zu nutzen. Eine Übersicht dazu findet man in [Torkkola, 2006]. Jedoch haben alle hier betrachteten Ansätze den Nachteil, dass sie ausschließlich die *Relevanz* eines Merkmals in Betracht ziehen. Um die *Nützlichkeit*, wie in Abschnitt 3.1 diskutiert, zu bestimmen, sind Filteransätze ungeeignet. Zu diesem Zweck müssen Wrapper Verfahren verwendet werden.

Eine umfassende Übersicht zu Verfahren die mittels einer definierten Suchstrategie nach geeigneten Merkmalsteilmengen suchen, wird in [Reunanen, 2006] gegeben. Man unterscheidet dabei zwischen deterministischen und stochastischen Suchstrategien. In letztere Gruppe zählen häufig Heuristiken zur globalen Suche auf diskreten Räumen, wie man sie auch aus der mathematischen Optimierung kennt. Dazu zählen evolutionären Algorithmen [Vafaie und Jong, 1992] [Yang und Honavar, 1998], Simulated An-

nealing [Debuse und Rayward-Smith, 1997] und andere. Da der Rechenaufwand bei solchen global optimierenden Verfahren ungleich höher ist, werden in der Praxis oft deterministische Suchstrategien verwendet.

Sequentielle Suche

Die bekanntesten Vertreter hierbei sind die sequentielle Vorwärts- (SFS) sowie die sequentielle Rückwärtssuche (SBS) [Reunanen, 2006]. Bei der Vorwärtssuche wird dabei mit einer leeren Teilmenge gestartet, und es werden alle Merkmale einzeln als Eingabe für einen Klassifikator verwendet. Das Merkmal, welches zum Klassifikator mit dem geringsten Fehler führt, wird dauerhaft in die Teilmenge der ausgewählten Merkmale aufgenommen. Dann wiederholt sich das Vorgehen mit allen verbleibenden Merkmalen. Diese werden einzeln den bereits gewählten Merkmalen hinzugefügt und in die Auswahlmenge aufgenommen, falls damit der geringste Fehler erzielt wurde. Dies wird solange wiederholt, bis entweder der Klassifikationsfehler des Netzes nicht mehr geringer wird oder eine vorgegebene Anzahl von Merkmalen ausgewählt wurde. Analog dazu funktioniert die Rückwärtssuche. Hierbei wird mit einer vollständigen Merkmalsmenge begonnen und diese schrittweise um jeweils ein Merkmal reduziert bis ein Minimum des Klassifikationsfehlers erreicht wurde. Ein Schritt der sequentiellen Vorwärtssuche (SFS) ist als Pseudocode in Algorithmus 2 gegeben.

Erweiterungen, wie die Einbeziehung von mehreren Merkmalen pro Suchschritt oder die Kombination von Vorwärts- und Rückwärtsschritten (sogenannte *Floating Search* Ansätze), machen die Verfahren flexibler, da sie den Suchraum vergrößern. Jedoch geht dies immer auf Kosten der Rechenzeit, da diese Flexibilität durch zusätzliche Trainingsvorgänge erkauft wird.

Algorithmus 2 SFS(X, Y, S, C, E_S)

Eingabe: Datensatz von Beobachtungen X, die entsprechenden Labels Y, die Menge bereits gewählter Merkmale S und die Menge alle Kandidaten C (für die klassische Vorwärtssuche gilt, dass C alle Merkmale enthält, die nicht in S sind) und der Approximationsfehler E_S, der mit der Auswahlmenge S erzielt wurde

Ausgabe: Merkmal c_{best} welches der Auswahlmenge S hinzugefügt wird, sowie der erzielte minimale Approximationsfehler E_{best}

for $\forall c_i \in C$ **do**
  $E_i = \text{TrainClassifier}(X, Y, S \cup c_i)$
end for
if $\exists E_i \in E; E_i + \varepsilon < E_S$ **then**
  $c_{best} = \arg\min_{c_i}(E)$
  $E_{best} = \min(E)$
else
  $c_{best} = \emptyset$
end if

In den beiden einfachen Algorithmen ist es notwendig (und zeitaufwendig), mehrmals einen Klassifikator zu trainieren, um den Klassifikationsfehler, also die Nützlichkeit, bewerten zu können. Im ersten Durchlauf wird für jedes Merkmal ein Klassifikator trainiert, also n-mal. Im zweiten Durchlauf wird für jedes nichtgewählte Merkmal zusammen mit dem gewählten Merkmal ein Klassifikator trainiert, also $(n-1)$-mal. Diese Folge kann bis zur Auswahl des letzten Merkmals fortgesetzt werden, wo nur noch einmal ein Netz zu trainieren wäre. Natürlich endet der Algorithmus typischerweise früher, nach Auswahl von n_{sub} Merkmalen. Die Anzahl der Trainingsvorgänge TV ergibt sich als

$$TV = \sum_{i=0}^{n_{sub}} (n-i)\,, n >= n_{sub}.$$

Um eine explizite Formulierung des Sachverhalts zu erhalten, bietet sich die Schreibweise als arithmetische Reihe an

$$TV = n(n_{sub}) - \frac{n_{sub}^2 - n_{sub}}{2}, n >= n_{sub}$$

Man sieht, dass die Anzahl dieser Trainingsvorgänge in einem quadratischen Zusammenhang zur Gesamtzahl der Merkmale und der Zahl zu wählender Merkmale steht. Daher besteht das Bestreben, diese Anzahl zu verringern, ohne dass dabei der Suchraum wesentlich eingeschränkt wird.

Zu diesem Zweck wird versucht, Techniken der Filterverfahren mit denen der Wrapperverfahren zu kombinieren. Dies wird dann in Teilen der Literatur als Hybridverfahren bezeichnet. Dazu wird beispielsweise mittels der Transinformation eine Vorauswahl relevanter Merkmale getroffen, welche dann mittels eines Wrappersuchverfahrens auf ihre Nützlichkeit hin untersucht werden [Van Dijck und Van Hulle, 2006], es werden Boosting-inspirierte Techniken zur Merkmalsselektion hier eingeordnet [Das, 2001], oder es werden Merkmale basierend auf ihrer Relevanz bestimmt durch Markov Blanket Filter ausgewählt, während per Wrapperverfahren die Qualität der unterschiedlich großen Teilmengen des Filterschrittes bewertet wird [Xing et al., 2001].

Der Grundgedanke bei allen Verfahren ist es, die Vorteile von Filtern und Wrappern zu kombinieren. Dazu werden Filtertechniken eingesetzt um mit einer cleveren Suchstrategie, möglichst die Menge der Trainingsvorgänge zu reduzieren, aber es wird das lernende System mit einbezogen um die gewünschten Aussagen über die *Nützlichkeit* zu erhalten.

Basierend auf dieser Prämisse der Hybridverfahren werden im folgenden Algorithmen entwickelt, welche versuchen mittels informationstheoretischer Maße die Suche zu steuern und die Menge der zu bewertenden Merkmalsteilmengen verringern, ohne auf die Aussagen über die Nützlichkeit von Merkmalen zu verzichten.

Dieses Vorgehen kann auch mit dem No-Free-Lunch Theorem [Wolpert, 1996] für Optimierung [Wolpert und Macready, 1997] motiviert werden. Stark vereinfacht sagt dieses Theorem, dass alle vorwissenfreien Suchverfahren gemittelt über die Menge aller möglichen Kostenfunktionen gleich gut sind. Daher ist es notwendig Vorwissen einzubringen. Im Merkmalsselektionsszenario entspricht der Wrapper dabei dem Suchverfahren, welche die finale Bewertung der Nützlichkeit vornehmen kann, während die Filterkomponente versucht, Struktur aus den Daten als Vorwissen einzubringen.

3.5. Auswahl mit Chow-Liu Bäumen

Die grundlegende Idee des hier entwickelten Verfahrens besteht darin einen Wrapper eine Vorwärtsselektion durchführen zu lassen. Anstatt jedoch alle Merkmale für einen Selektionsschritt in Betracht zu ziehen, werden nur vorausgewählte Merkmale betrachtet. Der sogenannte Chow-Liu Baum über den Daten wird dazu verwendet diese Vorauswahl sinnvoll zu treffen und dirigiert somit die Suche.

Das Hauptproblem beim Verwenden von Wrapperverfahren sind die häufigen Trainingsvorgänge. Ziel in dieser Arbeit ist es, die Anzahl der Trainingsvorgänge zu reduzieren, ohne dabei auf gute Kandidaten zu verzichten.

Als erstes werden die Chow-Liu Bäume eingeführt. Danach wird gezeigt, inwieweit sich dies für eine Vorwärtsauswahl eignet und die theoretischen Vorteile dieser Strukturierung werden diskutiert. Dann wird erläutert, warum eine Übertragung auf die Rückwärtssuche schwierig ist, bevor die Aussagen dieses Abschnittes mit Experimenten belegt werden.

Wenn im Folgenden von (Verbund-)Verteilungen die Rede ist, sind dabei im Kontext der Merkmalsselektion immer die Verteilungen der Da-

tenpunkte gemeint, wobei jedes Merkmal eine Dimension des Gesamtmerkmalraums aufspannt.

3.5.1. Chow-Liu Bäume

Die Chow-Liu Bäume (*Chow-Liu tree* - CLT) wurden ursprünglich als generative Klassifikatoren entwickelt. Für jede Klasse eines Klassifikationsproblems wurde die Verteilung der Beispiele approximiert. Für die komplette Verbundverteilung wird eine geeignete Approximation dieser Verteilung gesucht. Dies steht in engem Zusammenhang mit dem bereits diskutierten Fluch der Dimensionalität, wonach hochdimensionale Verteilungen aufgrund spärlicher Daten nur unzureichend dargestellt werden. Genau diese Approximation liefert der CLT. In der Anwendungsphase wird dann die Wahrscheinlichkeit des zu klassifizierenden Beispiels für alle Bäume bestimmt, und die Klasse des Baumes mit der maximalen Wahrscheinlichkeit entspricht der Klassifikationsantwort. Man kann sich dies vereinfacht analog zu einem Hidden Markov Modell zur Klassifikation vorstellen - jedoch ohne zeitliche Zusammenhänge.

Chow-Liu Bäume wurden entwickelt, um Verbundverteilungen effektiv durch einen Abhängigkeitsbaum erster Ordnung repräsentieren und approximieren zu können. [Chow und Liu, 1968] entwickelten dazu ein Verfahren, welches eine Verbundverteilung als Produkt von zweidimensionalen bedingten Wahrscheinlichkeiten ausdrückt. Wird dieser Zusammenhang als grafisches Modell interpretiert, erhält man die namensgebende Baumstruktur. Es wurde dabei gezeigt, dass ein Chow-Liu Baum dabei auch die optimale Baumstruktur darstellt, also den Approximationsfehler zur wahren Verbundverteilung im Sinne eines Maximum Likelihood Schätzers minimiert.

Definition 3.23

Chow-Liu Baum

Um eine k-dimensionale Verteilung X zu approximieren, wird ein

Baum mit $k-1$ Verbindungen bedingter Wahrscheinlichkeiten konstruiert. Maximiert dieser Baum dabei die Summe der logarithmischen Wahrscheinlichkeiten für jedes gegebene Beispiel, so heißt dieser Baum Chow-Liu Baum.

$$T_{ChowLiu} = \arg\max_{T} \sum_{i=1}^{N} \log T(x^i)$$

Dabei ist $T(x^i)$ die durch den Baum T approximierte Wahrscheinlichkeit des Beispiels x^i mit $1 \leq i \leq N$.

Wichtig ist dabei, dass zwar die allgemeine Struktur des Baumes festgelegt wird, also die Zusammenhänge zwischen den Variablen, allerdings kann die Wurzel des Baumes frei gewählt werden - jeder Knoten, ein Merkmal im Sinne der Merkmalsselektion, ist ein potentieller Wurzelknoten. Die Auswahl eines bestimmten Knotens hat keinen Einfluss auf die Approximationsgüte des CLT. Ein Beispiel für einen solchen Chow-Liu Baum ist in Abbildung 3.9 gezeigt.

Für den hier eingeführten Algorithmus wird nur die Struktur des Baumes, also welche Merkmale an welchen anderen Merkmalen hängen, von Bedeutung sein, nicht aber die Verteilungen oder die Wahrscheinlichkeiten, die sich für konkrete Beispiele ergeben.

Ermittlung des Chow-Liu Baumes

Der Algorithmus zur Erstellung eines solchen Chow-Liu Baumes folgt dabei drei Schritten, die anschließend erläutert werden:

1. Berechnung einer Transinformationsmatrix. Diese enthält alle paarweisen Transinformationen zwischen allen Merkmalen.
2. Berechnung des maximalen Spannbaums über dieser Transinformationsmatrix.

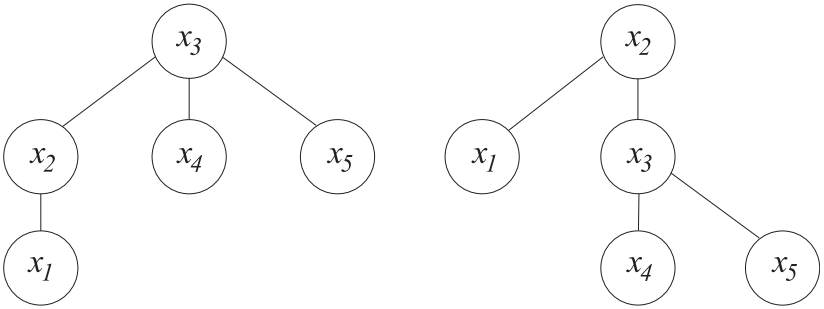


Abbildung 3.9.: Beispiel für zwei mögliche Baumdarstellungen der fünfdimensionalen Verteilung $P(x) = P(x_1, x_2 \dots x_5)$. Links wird die Verbundverteilung als $P(x) = P(x_3)P(x_4|x_3)P(x_5|x_3)P(x_2|x_3)P(x_1|x_2)$ dargestellt, rechts dagegen als $P(x) = P(x_2)P(x_1|x_2)P(x_3|x_2)P(x_4|x_3)P(x_5|x_3)$. Beide Bäume unterscheiden sich nur durch den unterschiedlichen Wurzelknoten, ihre Approximation der Verbundverteilung ist äquivalent. Nach den aktuellen Arbeiten im Bereich graphischer Modelle [Bishop, 2006], müssten diese Graphen gerichtet (Pfeile von der Wurzel weg) dargestellt werden, da es sich um bedingte Verteilungen handelt. Praktisch wird jedoch meist die Verbundverteilung (ungerichtet) gespeichert und die Konditionierung erfolgt beim Berechnen der Wahrscheinlichkeiten für konkrete Beispiele.

3. Berechnung der bedingten Wahrscheinlichkeitsverteilungen für jede Kante des Spannbaumes.

Zur Aufstellung der Transinformationsmatrix werden alle paarweisen Werte der Transinformation zwischen allen Merkmalen berechnet. Die Hauptdiagonale (Transinformation einer Variablen zu sich selbst - also ihre Entropie) wird weggelassen. Aufgrund der Symmetrie der Transinformation ist es ausreichend, entweder die obere oder die untere Dreiecksmatrix zu bestimmen. Daraus folgt, dass bei einer k-dimensionalen Verteilung $\frac{k^2-k}{2}$ Transinformationsberechnungen durchzuführen sind.

Dazu können alle in Abschnitt 3.3 vorgestellten Verfahren eingesetzt werden. Im Rahmen dieser Arbeit wurde die Kerneldichteschätzung verwendet.

Für den zweiten Schritt wird diese Transinformationsmatrix als Adjazenzmatrix eines ungerichteten Graphen interpretiert. Dabei entspricht jedes Merkmal einem Knoten V in diesem vollvermaschten Graphen G, während die Kanten E zwischen den Knoten entsprechend der Transinformation zwischen beiden Merkmalen gewichtet werden.

Definition 3.24

MAXIMALER SPANNBAUM MST

Ein Spannbaum ist ein Teilgraph von G, der alle Knoten V enthält und dessen Kanten einen Baum (zusammenhängend, aber keine Kreise) bilden. Ein Spannbaum ist maximal, falls die Summe über alle Gewichte der Kanten E dabei größer oder gleich der Summe jedes anderen Spannbaums über demselben zusammenhängenden, ungerichteten Graphen G ist.

Eine Möglichkeit zur Berechnung des maximalen Spannbaums ist dabei eine modifizierte Version des Algorithmus von Kruskal [KRUSKAL, 1956]. Dieser Algorithmus tut nichts anderes, als immer wieder unter den nicht gewählten Kanten jene mit dem höchsten Gewicht auszuwählen, die keinen Kreis mit den schon gewählten Kanten bildet. Wenn keine Kante mehr diese Bedingung erfüllt, terminiert der Algorithmus und die Struktur der gewählten Kanten ist dann der maximale Spannbaum[4].

[4]Sollte die Berechnung des maximalen Spannbaums von zeitkritischer Bedeutung sein, kann auch der Algorithmus von Prim [PRIM, 1957] verwendet werden. Dieser ist effizienter als Kruskals Ansatz, allerdings nur bei Nutzung von Fibonacci-Heaps als Datenstruktur. Im Rahmen der hier anvisierten Nutzung zur Merkmalsselektion ist der Algorithmus von Kruskal ausreichend, da die Berechnung des Spannbaums nur einen kleinen Bruchteil der Gesamtrechenzeit ausmacht.

Der erhaltene Spannbaum ist der gesuchte Chow-Liu Baum. Um damit eine Approximation der Verbundverteilung durchzuführen, ist es zusätzlich notwendig, die einzelnen bedingten Wahrscheinlichkeiten, die eine Kante in dem Baum bilden, zu bestimmen und zu speichern. Für die Merkmalsselektionsproblematik ist jedoch die Struktur entscheidend, und die eigentlichen Wahrscheinlichkeiten können vernachlässigt werden. Der dritte Schritt bei der Erstellung eines Chow-Liu Baumes kann daher in diesem Kontext, trotz der einfachen Realisierung, übergangen werden. In der Pseudocodedarstellung von Algorithmus 3 sind alle Schritte angegeben.

Algorithmus 3 CHOW-LIU BAUM(X)

Eingabe: Datensatz von Beobachtungen X mit Dimensionalität k aus Domäne K
Algorithmus MST, welcher den maximalen Spannbaum über einer Adjazenzmatrix bestimmt
Ausgabe: Chow-Liu Baum T

Berechne alle Randverteilungen P_u, P_{uv} mit $u, v \in K$ {z.B. mit Kerneldichteschätzung}
Berechne alle paarweisen Transinformationsgrößen I_{uv} mit $u, v \in K$
$E_T = \mathrm{MST}(\{I_{uv}\})$
$T_{uv} \leftarrow P_{uv}$ für $uv \in E_T$

Für den formalen Nachweis, warum dies zu einer optimalen Approximation führt, sei hier auf die Ausführungen in [CHOW und LIU, 1968] verwiesen. Intuitiv kann man sich aber überlegen, dass die maximale Spannbaumsuche die Gesamtmenge an Transinformation zwischen den Variablen maximiert, d.h. der Informationsverlust, der durch das Weglassen von Kanten zwangsläufig entsteht, wird minimiert.

3.5.2. Vorwärtsauswahl mit Chow-Liu Bäumen

In diesem Abschnitt soll nun erläutert werden, wie die eben eingeführte Struktur des Chow-Liu Baumes in der Merkmalsselektion genutzt werden kann. Dazu wird der CLT in den Rahmen einer Vorwärtsauswahl eingepasst.

Zuerst muss geklärt werden, über welchen Daten der Baum erstellt wird. Zusätzlich zu den Eingangsvariablen wird der Zielwert, also die Klasseninformation oder der zu approximierende Funktionswert, als eine weitere Eingangsgröße interpretiert. Damit schätzt man die Verbundverteilung über $P(X_1, X_2, \cdots, X_k, Y)$. Als Festlegung wird dann der Knoten, der die Variable Y repräsentiert, als Wurzelknoten diesen Baumes betrachtet. Von dieser Wurzel beginnend wird nun die Vorwärtssuche gestartet. Dabei kommt der Standardalgorithmus zur sequentiellen Vorwärtssuche (siehe Abschnitt 3.4) zum Einsatz - mit der entscheidenden Änderung, dass nicht mehr alle nichtgewählten Variablen in jedem Schritt als Kandidaten zur Verfügung stehen, sondern diese Kandidatenmenge über die berechnete Baumstruktur ausgewählt wird.

Konkret bedeutet dies, dass im ersten Schritt nur jene Variablen als Addition zur Merkmalsmenge in Betracht kommen, die direkt an der Wurzel des Baumes hängen. Diese werden einzeln mit dem gewählten Lernalgorithmus ausprobiert und das Merkmal, welches den geringsten Fehler erzeugt, wird dauerhaft ausgewählt. Danach wird die Menge der Kandidatenvariablen für den nächsten Schritt aktualisiert. Dazu wird das ausgewählte Merkmal aus dieser Menge entfernt und alle Kinder dieses Merkmals im Chow-Liu Baum werden der Kandidatenmenge hinzugefügt. Des Weiteren werden alle Merkmale, deren Hinzunahme keine Auswirkung auf den Fehler haben, ebenfalls aus der Kandidatenmenge gelöscht und deren Kinder hinzugefügt. Dieses Schema wird solange wiederholt, bis alle Knoten, und damit Merkmale, durchlaufen wurden und entweder als unwichtig oder relevant eingestuft wurden.

Algorithmus 4 MERKMALSAUSWAHL MIT CHOW-LIU BÄUMEN(X, Y)

Eingabe: Datensatz von Beobachtungen X und die entsprechenden Labels Y
Ausgabe: Merkmalsteilmenge S

$Z \leftarrow X \cup Y$
$T \leftarrow$ CHOW-LIU BAUM(Z)
$N \leftarrow$ NODE(Y) {Beginne mit Wurzelknoten Y als Startpunkt der Suche}
$S \leftarrow \emptyset$ {Initiale Merkmalsmenge sei leer.}
repeat
  $C \leftarrow children(N)$ {Alle Kinder der Suchknotenmenge sind Kandidaten}
  $c \leftarrow \mathrm{SFS}(S, C, X, Y)$
  $S \leftarrow S \cup c_{best}$ {Aufnahme des besten Merkmals in die Auswahlmenge}
  $N \leftarrow N \cup c_{best} \cup c_{unimportant}$ {Aufnahme der besten und unwichtigen Merkmale in die Suchknotenmenge}
until $c_{best} = \emptyset$ AND $c_{unimportant} = \emptyset$

Eine grafische Darstellung der Selektion ist in Abbildung 3.10 an einem Beispiel zu sehen. Zu Beginn sind alle Merkmale und die Labelinformationen gegeben. In Schritt 1 wird daraus der Chow-Liu Baum konstruiert. Die eigentliche Merkmalsselektion beginnt in Schritt 2 mit der Wurzel als Suchknotenmenge (gestrichelt). Alle Kinder der Wurzel sind damit Kandidaten für die Wrapper-Vorwärtssuche (gepunktet). Merkmal x_2 sei das beste Merkmal gewesen, und wird damit in Schritt 3 in die Merkmalsmenge und die Suchknotenmenge aufgenommen, damit wird auch die Kandidatenmenge aktualisiert. Bei der nächsten Wrappersuche (Schritt 4) über x_1, x_3, x_4 und x_5 wird x_1 als besten Merkmal identifiziert und x_3 sowie x_5 als unwichtig eingestuft (durchgezogen). Die Kinder aller drei Knoten werden der Kandidatenmenge hinzugefügt. Im letzten Suchschritt über x_4 und x_6 wird x_6 ausgewählt und x_4 als unwichtig erkannt. Damit sind alle Knoten abgearbeitet und die Merkmalsselektion ist abgeschlossen.

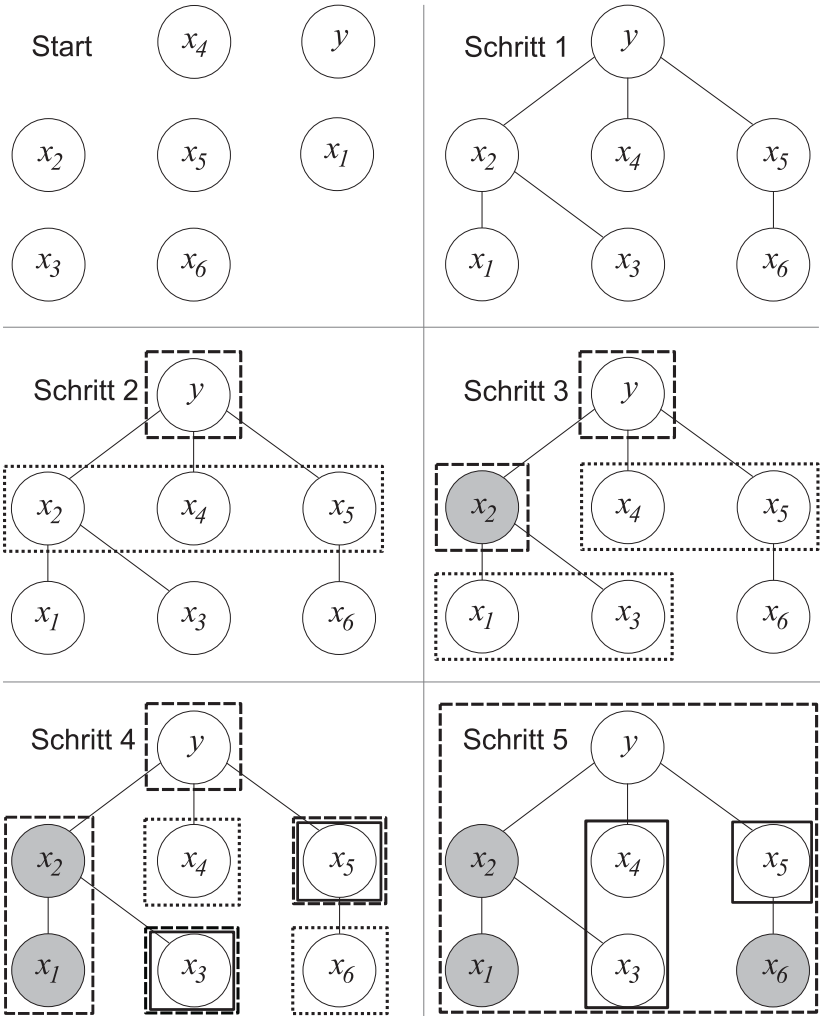


Abbildung 3.10.: Ablauf der Merkmalsselektion mittels eines Chow-Liu Baumes. Die Erläuterung des Ablaufs findet sich im Text auf der nebenstehenden Seite.

3.5.3. Diskussion

Dieser Abschnitt wird die Frage klären, welche Vorteile sich durch die Nutzung der Chow-Liu Bäume ergibt. Dabei hängt die Struktur des sich ergebenden Baumes von den Zusammenhängen in den Daten ab.

Für die erste Überlegung wird angenommen, dass alle Merkmale $x_1, x_2, \cdots, x_k$ statistisch unabhängig voneinander seien. Das bedeutet, dass sich in den Eingangsvariablen keine Redundanzen befinden. Eine Teilmenge dieser Merkmale $x_p, ..., x_q$ enthalte Informationen über die Labelinformation y. Für die Transinformation bedeutet dies, dass alle Werte zwischen den Merkmalen untereinander und zum Ziel nahe null liegen, außer für die informationstragenden Variablen $x_p, ..., x_q$ und dem Ziel y. Um die in Definition 3.23 benannte Gleichung zu maximieren, ist es notwendig, dass alle Verbindungen zwischen $x_p, ..., x_q$ sowie y Teil des Spannbaumes werden. Als Folge hängen alle relevanten Variablen an der Wurzel des Baumes. Im Rahmen der Vorwärtssuche würden diese sukzessive ausgewählt werden. Alle anderen, irrelevanten Merkmale hängen jeweils an einem zufälligen Knoten - die Transinformationswerte, die über den Daten geschätzt werden sind auch bei Unabhängigkeit nie exakt null.

Der Vorteil gegenüber der klassischen Vorwärtsauswahl ist dabei, dass zu Beginn nicht alle Merkmale probiert werden müssen, sondern nur jene mit hoher Relevanz getestet werden. Trotzdem werden die irrelevanten Merkmale auf ihre Nützlichkeit hin untersucht, jedoch erst im späteren Teil der Abarbeitung. Für eine Verbildlichung sein auf Schritt 2 in Abbildung 3.10 verwiesen, wenn man annimmt, dass die Merkmale x_2, x_4 und x_5 relevant sind und die anderen irrelevant. Somit werden zu Beginn nur die relevanten Merkmale einbezogen, erst in den weiteren Schritten 3-5, werden sukzessive die irrelevanten Kandidaten einbezogen.

Nehmen wir zwei vollkommen redundante Merkmale x_r und x_s mit auf,

können wir dies wie folgt formulieren:

$$I(X_r; Y) \approx I(X_s; Y) \approx I(X_r, X_s; Y).$$

Die Information, die jedes der beiden Merkmale zum Ziel enthält, ist dieselbe die beide Merkmale zusammen zum Ziel enthalten. Umgekehrt ausgedrückt, ist die Transinformation zwischen den beiden Variablen mindestens so groß, wie die der Variablen zur Labelinformation.

$$I(X_r; X_s) \geq I(X_r; Y) \approx I(X_s; Y)$$

Die Gleichheit ist dabei auch nur gegeben, falls Y sich vollständig durch X_r erklären lässt. Für die Konstruktion des Chow-Liu Baumes bedeutet dies, dass die Verbindung zwischen den beiden Variablen X_r und X_s Bestandteil des Baumes sein muss, um die Summe über die Gewichte zu maximieren. Daraus folgt dann auch, dass nur noch eine der beiden Variablen an die Wurzel gehängt werden kann, da sich sonst ein Kreis ergeben würde. Diese Argumentation lässt sich einfach auch auf mehrere redundante Variable übertragen.

Diese Eigenschaft ist aus Sicht der Systemidentifikation ein Vorteil, da aus der Perspektive der Wurzel alle Merkmale, die untereinander redundant sind, sich in einem Zweig des Baumes befinden - wobei das informativste Feature dieses Zweiges mit der Wurzel verbunden ist.

Für zwei Merkmale, die zwar teilweise redundant sind $I(X_t; X_u) > I(X_t; Y)$, aber trotzdem neue Informationen enthalten $I(X_t; Y) < I(X_t, X_u; Y)$, ist es notwendig einen solchen Zweig mit redundanten Merkmale ebenfalls in die Vorwärtssuche mit einzubeziehen.

Degenerierte Bäume

Es gibt zwei Extrema von Bäumen, die entstehen können und im Sinne der Merkmalsselektion mit Chow-Liu Bäumen als degeneriert aufgefasst werden. Diese sind grafisch in Abbildung 3.11 dargestellt.

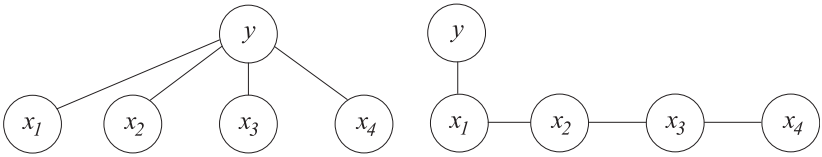


Abbildung 3.11.: Degenerierte Chow-Liu Bäume. Links: Alle Knoten hängen an der Wurzel. Es gibt durch den Chow-Liu Baum keinen Vorteil verglichen mit der Standardvorwärtsauswahl. Rechts: Alle Knoten hängen in einem Zweig des Baumes. Dies führt zu einer minimalen Menge von Trainingsvorgängen.

1. Alle Knoten der Variablen hängen direkt am Wurzelknoten Y. Dabei handelt es sich um die schlechtmöglichste Struktur des Baumes, denn die Ausführung des Algorithmus entspricht nun der sequentiellen Vorwärtssuche. Eine Einschränkung der Kandidatenmenge wird nicht vorgenommen. In dieser Form gilt der in Abschnitt 3.4 benannte quadratische Zusammenhang zwischen Eingangsvariablen und der Anzahl der Trainingsvorgänge.

2. Alle Knoten bilden einen Pfad bis zum einzigen Blatt des Baumes. Diese Struktur führt dazu, dass jedes Merkmal einzeln geprüft wird und entweder zum Merkmalssatz hinzugefügt wird, oder endgültig abgelehnt wird, bevor das nächste Merkmal betrachtet wird. Daraus ergibt sich ein direkter linearer Zusammenhang zur Anzahl der Eingangsvariablen.

Die Struktur von realen Datensätzen liegt zwischen diesen beiden Extrema. Für verschiedene Datensätze der UCI Machine Learning Repository [Asuncion und Newman, 2007] wurde die durchschnittliche Zahl von Kindern der Nichtblattknoten c des Chow-Liu Baumes bestimmt. Falls $c = n$ würde dies Extremum 1 bedeuten, falls $c = 1$ würde dies dem zweiten degenerierten Baum entsprechen. Für die Datensätze ergaben sich dabei Zahlen von $1.61 \leq c \leq 2.63$. Der Versuch hier einen konkreten logarithmischen Zusammenhang hineinzuinterpretieren, er-

wies sich als schwierig. Dies liegt daran, dass die Struktur des Baumes ausschließlich von den in den Daten gegebenen Zusammenhängen abgeleitet wird und nicht von der Anzahl der Dimensionen und damit der Eingangsvariablen. Daher ist eine rigorose Analyse der Laufzeit leider nicht möglich. Nichtsdestotrotz zeigt sich, dass die Anzahl der Trainingsvorgänge im Mittel in der Größenordnung $O(n \log n)$ bewegen, wie man es bei einer Baumstruktur erwarten würde.

3.5.4. Rückwärtssauswahl mit Chow-Liu Bäumen

Die sich ergebende Frage ist, ob dieses Verfahren auch auf das Problem der Rückwärtssuche übertragbar ist. Um dieses zu erreichen, wäre es notwendig die Baumstruktur, welche die Suche dirigiert, so zu verändern, dass nicht mehr die informativsten Verbindungen die Kandidaten bestimmen, sondern die uninformativsten. Im Idealfall müssten also alle über das Ziel unaussagekräftigen Merkmale direkt an der Wurzel hängen.

Um dieses zu erreichen, kann man den Algorithmus zur Bestimmung des Chow-Liu Baumes so modifizieren, dass statt dem maximalen Spannbaum nach dem minimalen Spannbaum gesucht wird. Der entstehende Baum (welcher dann kein CLT mehr ist), strukturiert die Variablen dann so, dass nur noch minimale Information in der Gesamtstruktur enthalten ist. Theoretisch erlaubt dieses Konstrukt der *Uninformation*, schnell die uninformativen Merkmale im Rahmen einer Rückwärtsauswahl aus der Gesamtmenge zu eliminieren. Dazu können alle Argumente analog zur Vorwärtsvariante angebracht werden.

Leider erwies sich in den praktischen Untersuchungen, dass die Rückwärtssuche nicht praktikabel ist. Das liegt daran, dass der minimale Spannbaum auf allen Testdatensätzen ein degenerierter Baum ist. In jedem Datensatz fand sich eine Variable, welche zu fast allen anderen Knoten minimale Information enthält. Zur Bildung des minimalen

Spannbaums werden damit alle anderen Knoten inklusive der Wurzel an diese Variable gehängt.

Als Folge hat die Wurzel genau ein Kind - den uninformativen Knoten, der im ersten Schritt auch eliminiert wird. Allerdings gibt es danach keine weitere Struktur, die das Verfahren ausnutzen kann. Dadurch, dass alle anderen Knoten an diesem einem Zentrum hängen, kann der Algorithmus nur nach dem klassischen, und damit teuren, Rückwärtssuchverfahren vorgehen. Somit erweist sich die Rückwärtsvariante der Merkmalsauswahl mit Chow-Liu Bäumen als unsinnig. Dabei ist zu betonen, dass es sich nicht um ein Problem im Algorithmus handelt, sondern in der Struktur der Daten. Ein einzelner Knoten der keine Informationen über andere Knoten enthält führt zur degenerierten Baumstruktur.

Da in allen durch geführten Experimenten wurde ein solcher Knoten gefunden wurde, muss die Idee der Rückwärtssuche verworfen werden.

3.5.5. Experimente

Um das vorgestellte Verfahren zu untersuchen, wurden auf mehreren Datensätzen aus dem UCI Machine Learning Repository [Asuncion und Newman, 2007] die Merkmalsselektion und anschließend eine Klassifikation durchgeführt. Verglichen werden dabei eine Klassifikation ohne jegliche Merkmalsauswahl, die sequentielle Vorwärtssuche (Algorithmus 2) als reines Wrapperverfahren, MIFS (Algorithmus 1) als Vertreter der Filterverfahren sowie der eben vorgestellte Ansatz der Merkmalsauswahl mit Chow-Liu Bäumen (Algorithmus 4).

Als Klassifikatoren kamen dabei sowohl ein 3-Nächster-Nachbar Klassifikator (Tabelle 3.6) als auch ein Multi-Layer Perceptron (Tabelle 3.7) mit zwei verborgenen Schichten mit 20 bzw. 10 Neuronen zum Einsatz. Für die eigentliche Merkmalsauswahl wurde eine dreifache Kreuzvalidierung benutzt um eine Überanpassung während der Merkmalsselekti-

on zu vermeiden [Reunanen, 2003], während für die eigentliche Klassifikationsbewertung eine zehnfache Kreuzvalidierung verwendet wurde.

Von Interesse ist dabei allerdings nicht nur das Klassifikationsergebnis, sondern auch die Anzahl der verwendeten Trainingsvorgänge, da ein Ziel in der Verringerung dieser liegt, ohne schlechtere Klassifikationsergebnisse zu erzielen.

Betrachtet man die in den Tabellen dargestellten Ergebnisse, fällt zuerst auf, dass das Multi-Layer Perceptron schlechtere Ergebnisse liefert, als der Nächste-Nachbar-Klassifikator. Dies ist darauf zurückzuführen, dass keine explizite Modellselektion und Optimierung der Parameter durchgeführt wurde. Jedoch ist der damit induzierte Bias für den Vergleich der Merkmalsselektionstechniken derselbe, was die Ergebnisse innerhalb einer Tabelle vergleichbar macht - und damit auch die Problematik der Nützlichkeit einbringt.

Eine weitere Beobachtung, die sich aus beiden Tabellen ergibt, ist, dass die Zahl der durchgeführten Trainingsvorgänge beim CLT-Verfahren deutlich unter der der einfachen Vorwärtsauswahl liegt. Die theoretische Verringerung dieser Trainingszyklen lässt sich also auch praktisch beobachten. Die Klassifikationsergebnisse liegen dabei beim MLP gleichauf mit denen der umfangreicheren Vorwärtssuche und sind in einem Fall signifikant besser, während beim Nächsten-Nachbar Klassifikator auch schlechtere Ergebnisse zustande kommen. Ähnlich wenig eindeutig ergibt sich das Bild im Vergleich zum MIFS-Filterverfahren.

Die Gesamtzahl der ausgewählten Merkmale divergiert signifikant ohne eine klare Aussage treffen zu können, dass die CLT basierte Auswahl immer mehr oder weniger Merkmale als die Vorwärtssuche ergeben würde. Daher kann zusammenfassend nur festgestellt werden, dass die Verwendung des Chow-Liu Baumes die Zahl der benötigten Trainingszyklen vermindert und die Qualität der Auswahl in derselben Größenordnung wie die Vergleichsverfahren liegt.

Datensatz	Ionosphere			Spambase			GermanCredit			Breast Cancer		
Merkmale	34			57			24			30		
Beispiele	351			4601			1000			569		
	BER	F	TV	BER	F	TV	BER	F	TV	BER	F	TV
All	23.78	34	-	10.84	57	-	36.33	24	-	3.55	30	-
MIFS	11.80	5	-	8.65	19	-	33.90	6	-	4.36	5	-
SFS	12.04	5	189	8.44	12	663	31.61	7	164	4.21	6	189
CLT-FS	12.19	6	39	15.97	6	76	34.89	5	28	4.42	4	35

Tabelle 3.6.: Ergebnisse für verschiedene Merkmalsselektionstechniken. Als Klassifikator wurde ein 3-Nächster-Nachbarklassifikator mit 10-facher Kreuzvalidierung verwendet. Für jedes Verfahren sind die Balanced Error Rate (BER), die Anzahl der gewählten Merkmale (F) und die Anzahl der Trainingsvorgänge (TV) gezeigt. All beschreibt die Verwendung aller Merkmale, MIFS ist die Mutual Information for Feature Selection aus Algorithmus 1, SFS die sequentielle Vorwärtssuche aus Algorithmus 2 und CLT-FS ist das Chow-Liu Baum basierte Verfahren aus Algorithmus 4.

Datensatz	Ionosphere			Spambase			GermanCredit			Breast Cancer		
Merkmale	34			57			24			30		
Beispiele	351			4601			1000			569		
	BER	F	TV	BER	F	TV	BER	F	TV	BER	F	TV
All	20.08	34	-	13.81	57	-	41.70	24	-	13.78	30	-
MIFS	24.54	5	-	16.29	19	-	37.47	6	-	12.48	5	-
SFS	18.47	3	130	17.39	8	477	39.06	4	110	13.44	4	140
CLT-FS	18.12	6	38	17.26	9	97	38.52	3	24	9.37	8	37

Tabelle 3.7.: Ergebnisse für verschiedene Merkmalsselektionstechniken. Als Klassifikator wurde ein Multi-Layer Perceptron mit zwei Hiddenschichten bei 10-facher Kreuzvalidierung verwendet. Dabei wurden die Ergebnisse über drei Gesamtdurchläufe gemittelt. Für jedes Verfahren sind die Balanced Error Rate (BER), die Anzahl der gewählten Merkmale (F) und die Anzahl der Trainingsvorgänge (TV) gezeigt. All beschreibt die Verwendung aller Merkmale, MIFS ist die Mutual Information for Feature Selection aus Algorithmus 1, SFS die sequentielle Vorwärtssuche aus Algorithmus 2 und CLT-FS ist das Chow-Liu Baum basierte Verfahren aus Algorithmus 4.

Die Verwendung der Transinformation zur Merkmalsselektion erfolgt hierbei nur mittelbar - nämlich zur Konstruktion der Baumstruktur. Bei der eigentlichen Selektion spielen die berechneten Werte auch keine Rolle mehr, nur in der Struktur sind diese indirekt abgebildet. Im nächsten Abschnitt soll daher der Frage nachgegangen werden, inwieweit die Transinformation direkt in den Selektionsprozess integriert werden kann.

3.6. Auswahl mit Residual Mutual Information

In diesem Abschnitt sollen Verfahren vorgestellt werden, die die Kombination von Filter- und Wrapper-Merkmalsselektionstechniken realisieren, in dem Aussagen über den Informationsgehalt im Residuum eines Klassifikators getroffen werden. Die Idee ist dabei, dass im Residuum Informationen stecken, die verwendet werden können, um weitere Merkmale zu wählen. Verbal gesprochen stecken darin alle vom Funktionsapproximator gemachten Fehler. Gesucht werden nun Merkmale die in Zusammenhang mit diesen Fehlern stehen, damit diese genutzt werden können, um den gemachten Fehler zu verringern.

Dabei wird der Begriff des Residuums analog zur Numerischen Mathematik verwendet.

Definition 3.25

RESIDUUM

Als Residuum wird die Abweichung vom gewünschten, realen Ergebnis bezeichnet, welche entsteht, wenn ein Funktionsapproximator verwendet wird. Sei $f(x) = t$ die organale Funktion und $\hat{f}(x) = y$ die ermittelte Approximation beispielsweise realisiert mittels eines neuronalen Netzes. Dann ergibt sich das Residuum r als

$$r = f(x) - \hat{f}(x) = t - y.$$

Man beachte, dass im Gegensatz zum Approximationsfehler, das Vorzeichen eine Rolle spielt und daher auch keine mittleren Residuen oder ähnliches gebildet werden. Im Sinne dieser Definition werden Klassifikationsprobleme als Spezialfall des Approximationsproblems interpretiert.

Es werden drei unterschiedliche Algorithmen vorgestellt, die die Merkmalsauswahl mittels des Residuums durchführen. Diese werden dann ausführlich diskutiert und experimentell untersucht.

3.6.1. Algorithmen zur Residual Mutual Information

Die ersten beiden Algorithmen wurden gemeinsam mit Christoph Möller in seiner Diplomarbeit [Möller, 2009] entwickelt und später veröffentlicht [Schaffernicht et al., 2009a].

Der Ausgangspunkt für die beiden Verfahren ist jeweils derselbe. Zuerst werden alle Transinformationswerte zwischen den Eingangsvariablen $X_1, X_2, ..., X_n$ und der Zielgröße Y berechnet. Das Merkmal mit der größten Transinformation wird verwendet, um damit den Funktionsapproximator zu trainieren. Dieser wird ausgewertet und das Residuum bestimmt. Nun wird eine neue Rangliste von Transinformationswerten erstellt, allerdings nicht mehr von den Variablen zum Ziel, sondern zwischen Merkmalen und dem Residuum. Der beste Eingangskanal wird wieder hinzugefügt, und die Prozedur wiederholt sich.

1. Beginne mit einer leeren Merkmalsteilmenge und setze für den ersten Schritt das Residuum gleich den Zielwerten[5].
2. Berechne die Transinformation zwischen jedem nichtgewähltem Merkmal und dem Residuum.

[5] Genauer gesagt entspricht dies dem Residuum zwischen dem Ziel und einem Approximator mit der Ausgabe von null.

Algorithmus 5 $S = \text{RMI.1}\ (X, Y)$

Eingabe: Datensatz von Beobachtungen X und die entsprechenden Labels Y

Ausgabe: Menge von gewählten Merkmalen S und den letzten Klassifikator

$S \leftarrow \emptyset$ {Starte mit leerer Merkmalsmenge}
$R \leftarrow Y$ {Residuen entsprechen den Zielwerten}
while Abbruchkriterium nicht erfüllt **do**
  $X_{max} = \arg\max_{X_i} [I(X_i; R)]$
  $S \leftarrow S \cup X_{max}$
  $X \leftarrow X \setminus X_{max}$
  Classifier $\leftarrow$ TrainClassifier(S, Y)
  Prediction $\leftarrow$ ApplyClassifier (Classifier, S)
  $R \leftarrow Y -$ Prediction
end while

3. Bestimme jenes Merkmal mit dem maximalen Transinformationswert.
4. Füge dieses Merkmal der Menge ausgewählter Merkmale hinzu.
5. Trainiere einen neuen Approximator.
6. Berechne das neue Residuum zwischen der aktuellen Approximation und dem Zielwert und gehe zu Schritt 2 - falls nicht das Abbruchkriterium erfüllt ist.

Eine formalere Beschreibung in Pseudocode ist in Algorithmus 5 gegeben.

Als Abbruchkriterien kommen dabei eine bestimme Anzahl von gewählten Merkmalen, der verbleibende Fehler des Klassifikators oder auch das Unterschreiten einer Schranke bei der maximalen, berechneten Transinformation in Betracht.

Von entscheidender Bedeutung bei diesem Algorithmus ist die Tatsache, dass der Klassifikator in jedem Schritt wieder verworfen wird und

mit den neuen Merkmalen eine komplett neue Instanz trainiert wird. Dies erscheint im ersten Moment etwas unintuitiv, da auf diese Weise der Klassifikator, der zum Erzeugen des Residuums benutzt wurde, verworfen wird. Welche Argumente dafür sprechen, wird in Abschnitt 3.6.2 näher erläutert.

Jedoch führt diese Überlegung zur zweiten Variante des Algorithmus, welche sich dadurch unterscheidet, dass anstatt den Klassifikator immer zu verwerfen, einfach ein neuer Klassifikator angehängt wird. Dieser erhält als Eingabe das Klassifikationsergebnis der vorhergehenden Stufe der Kaskade sowie das residuumbasiert neu gewählte Merkmal und kann darauf basierend seine Entscheidung fällen. Dargestellt ist dieser Ansatz in Abbildung 3.12. Vom Vorgehen sind dabei Parallelen zu Cascade-Correlation Netzen [Fahlman und Lebiere, 1990] oder der Neuronalen Hauptkomponentenanalyse nach Sanger [Sanger, 1989] zu erkennen. Es wird mit jedem Merkmal eine neue Stufe in der Verarbeitungsstruktur hinzugefügt.

Der Pseudocode ist in Algorithmus 6 dargestellt.

3.6.2. Diskussion

Offensichtlicher Vorteil der Verfahrensweise mit den RMI Algorithmen ist die Reduktion der Trainingsvorgänge. Für jedes ausgewählte Merkmal wird nur noch ein einziges Mal ein Trainingsvorgang durchgeführt. Die Laufzeitkomplexität der Trainingsvorgänge ist linear und liegt damit in $O(n)$.

Doch es bleibt die Frage zu klären, warum das Residuum und warum speziell die Information, die in den Variablen über das Residuum steckt, nützlich für die Merkmalsselektion ist.

Erstens steckt im Residuum all jenes, was durch die bisher ausgewählten Merkmale in Kombination mit dem gewählten Klassifikator noch

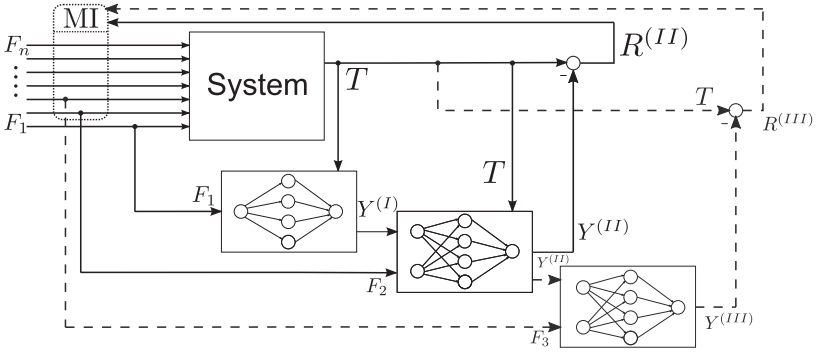


Abbildung 3.12.: Schematische Darstellung des RMI2 Algorithmus, die von links oben nach rechts unten zu lesen. Es wird die Auswahl des dritten Merkmals gezeigt. Im ersten Schritt wurde das Merkmal F_1 gewählt und mit nur diesem Merkmal ein neuronales Netz mit den Trainingswerten T gelernt. Im zweiten Durchlauf des Algorithmus wurde Merkmal F_2 gewählt und ein Netz trainiert, welches als Eingaben das Merkmal F_2 und die Ausgabe des letzten Netzes erhält. Die Ausgabe $Y^{(II)}$ wird benutzt, um das Residuum $R^{(II)}$ zu bestimmen. Zwischen diesem und allen nicht gewählten Merkmalen $F_3 \dots F_n$ wird dann die Transinformation (MI) bestimmt und damit das nächste Merkmal mit dem höchsten Wert hinzugefügt. Gestrichelt ist der Fortgang des Algorithmus angedeutet.

nicht erklärt werden kann. Diese intuitive Idee lässt sich auch formal sehr leicht zeigen. Unter der Definition, dass das Residuum R alles vom Ziel Y umfasst, was von den gewählten Merkmalen S nicht erklärt werden kann, ergibt sich

$$H(Y) - I(S;Y) + I_{BiasVerlust} = H(R).$$

Da $H(Y)$ konstant ist und für ein redundantes Merkmal x_i gilt $I(S;Y) = I(S \cup x_i;Y)$ folgt auch, dass $H(R)$ sich bei Hinzunahme eines redundanten Merkmals nicht ändern kann. Eine komplett redundante Variable trägt daher auch keine Informationen über das Residuum in

Algorithmus 6 $S = \text{RMI.2}\ (X, Y)$

Eingabe: Datensatz von Beobachtungen X und die entsprechenden Labels Y

Ausgabe: Menge von gewählten Merkmalen S und die Klassifikatorkaskade

$S \leftarrow \emptyset$ {Starte mit leerer Merkmalsmenge}
$R_0 \leftarrow Y$ {Menge aller Merkmale aus X}
$j \leftarrow 1$
while Abbruchkriterium nicht erfüllt **do**
  $X_{max} = \arg\max_{X_i} [I(X_i; R_{j-1})]$
  $S \leftarrow S \cup X_{max}$
  $X \leftarrow X \setminus X_{max}$
  $\text{Classifier}_j \leftarrow \text{TrainClassifier}(\text{Predicition}_{j-1}, X_{max}, Y)$
  $\text{Prediction}_j \leftarrow \text{ApplyClassifier}\ (\text{Classifier}_j, S)$
  $R_j \leftarrow Y - \text{Prediction}_j$
  $j \leftarrow j + 1$
end while

sich.

Daraus folgt ebenfalls, dass eine informative Variable x_j, die nicht redundant ist (also gilt $I(S;Y) < I(S \cup x_j; Y)$), dass $H(R|S) > H(R|S \cup x_j)$ ist. Das heißt, dass diese Verringerung des Residuums auch durch eine Berechnung von $I(X;R)$ als Auswahlkriterium erfolgen kann.

Anders interpretiert bedeutet dies, dass falls eine Eingangsvariable Informationen über das Residuum enthält, dann stecken in dieser Variable offensichtlich Informationen, die eingesetzt werden können, um dieses Residuum zu verringern.

Ein weiterer Vorteil, der sich in diesem Verfahren ergibt, ist, dass sich der Bias des verwendeten Approximators im Residuum widerspiegelt. Die bisherigen Überlegungen haben stillschweigend vorausgesetzt, dass der verwendete Approximator keinen Bias besitzt ($I_{BiasVerlust} = 0$), was allerdings vor dem Hintergrund des Bias-Varianz Dilemmas eher

unwahrscheinlich ist.

Daher muss davon ausgegangen werden, dass die Information, die in einer Merkmalsteilmenge S steckt, nur teilweise vom Approximator umgesetzt werden kann. Das bedeutet, dass ein Teil der Information verloren geht $I(S;Y) = I_{nutzbar} + I_{BiasVerlust}$. Allerdings spiegelt sich dieser Verlust, der durch Einschränkungen des verwendeten Klassifikators zustande kommt, auch im Residuum wieder. Das Residuum enthält also nicht nur die fehlenden Informationen in den gewählten Merkmalen, sondern es beinhaltet alles, was der eingesetzte Klassifikator unter Verwendung der Merkmale nicht erklären kann. Dies führt dazu, dass ein redundanter Kanal unter Umständen gewählt wird, falls durch die Redundanz der Biasfehler reduziert wird.

Man kann zusammenfassend sagen, dass die Verwendung der Residual Mutual Information den Vorteil hat, dass die Redundanzproblematik in diesem Verfahren implizit gelöst wird.

Allerdings wurde im Rahmen von Experimenten eine entscheidende, systematische Limitierung der kaskadierten Variante des Algorithmus (RMI.2) festgestellt. Auf den ersten Blick erscheint das Vorgehen sehr intuitiv mit jedem neuen Merkmal einfach die Entscheidung des vorhergehenden Klassifikators zu verbessern. Auf den zweiten Blick wird jedoch offensichtlich, dass in jeder Stufe des Klassifikators nur eine zweidimensionale Entscheidungsfläche zur Verfügung steht. Durch die Kaskade entsteht somit eine Reihe von ineinander geschachtelten Klassifikatoren. Damit sind rechentechnischen Anforderungen natürlich geringer als in einem monolithischen n-dimensionalen Gesamtentscheidungsraum, allerdings wird damit auch die Menge der Lösungen auf einen Unterraum beschränkt.

Wesentlich drastischere Auswirkung hat diese Einschränkung bei der Verwendung von Klassifikatoren, die ausschließlich die Klassenentscheidung ausgeben, wie der einfache Nächste-Nachbar Klassifikator. Damit erhält die nächste Entscheidungsstufe der Kaskade den Wert 0 oder 1

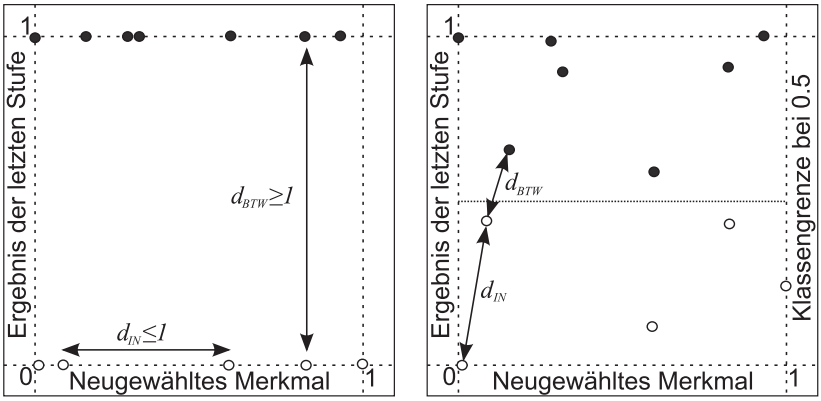


Abbildung 3.13.: Probleme mit der kaskadierten Variante der Residual Mutual Information. (a) Diskreter Klassifikator (z.B. Nächster Nachbar). Der Abstand zweier Beispiele aus derselben Klasse d_{IN} ist immer kleiner gleich dem Abstand zu jedem Beispiel aus der anderen Klasse d_{BTW}. Daher kann unter Verwendung dieses einen neuen Merkmals keine Änderung der Klassenzugehörigkeit herbeigeführt werden. (b) Kontinuierlicher Klassifikator mit expliziter Klassifikationsschwelle (z.B. neuronales Netz mit einer anderen Ausgabefunktion als der Stufenfunktion). Hier entsteht diese Problem nicht, da der Ausgang der letzten Kaskadenstufe nicht nur Extremwerte annimmt und der Abstand von Beispielen unterschiedlicher Klassen d_{BTW} kleiner sein kann als der nächste Innerklassennachbar d_{IN}.

zusammen mit dem neu ausgewählten Merkmal, welches auch auf das Intervall $[0, 1]$ skaliert ist. Damit dominiert die Entscheidung der vorhergehenden Stufe immer das neue Merkmal, da die Distanz zu einem Beispiel der anderen Klasse immer größer gleich 1 ist, während die Distanz zu allen Nachbarn der eigenen Klasse immer kleiner gleich 1 ist. Daher kann die in der ersten Stufe der Kaskade getroffene Entscheidung nie mehr korrigiert werden. Dargestellt ist dieses Problem in Abbildung 3.13.

Man könnte dieses Problem umgehen, indem man hier eine variable

Skalierung des neuen Merkmals zulässt. Allerdings bedeutet dies, dass neue Hyperparameter während des Lernens geschätzt werden müssen. Ebenfalls denkbar wäre die Verwendung von speziellen Distanzmaßen, die dieses Problem umgehen. Die einfache intuitive Lösung ist damit allerdings immer nicht mehr gegeben. Daher wird empfohlen, dieses Verfahren nicht mit solchen diskreten Klassifikator zu kombinieren.

Im Falle eines kontinuierlichen Funktionsapproximators oder eines Klassifikators, der eine Klassenentscheidung basierend auf einem kontinuierlichen Wert (z.B. Abstand zu einer Trenngerade) trifft und diesen Wert der nächsten Kaskadenstufe zur Verfügung stellt, stellt dieses Verhalten jedoch kein Problem dar.

3.6.3. Gewichtete Residual Mutual Information

Bisher wurde das Residuum auf der Ebene der Merkmale betrachtet - allerdings kann man sich auch eine andere Anwendung vorstellen, die in diesem Abschnitt diskutiert werden soll. Diese Idee wurde in [Schaffernicht und Gross, 2011] veröffentlicht.

Die gedankliche Grundidee ist dabei ähnlich dem AdaBoost-Algorithmus [Freund und Schapire, 1995]: Beispiele, die bisher falsch klassifiziert werden, dominieren die Selektion neuer Merkmale, im Gegensatz zu Beispielen, die korrekt klassifiziert werden.

Dazu existiert in der Literatur der Begriff der gewichteten Transinformation [Guiasu, 1977], dort wird diese als

$$wI(X;Y) = \int_x \int_y w(x,y)p(x,y)\log\frac{p(x,y)}{p(x)p(y)}dydx$$

definiert. Für die hier vorgestellte Umsetzung entspricht das Gewicht, dem betragsmäßigen Residuum, also dem Fehler, der für das jeweilige Beispiel $p_{x,y}$ gemacht wird.

Definition 3.26

RESIDUUMSGEWICHTETE TRANSINFORMATION

Damit ergibt sich für die mittels des Residuums gewichtete Transinformation folgende Berechnungsvorschrift:

$$rI(X;Y) = \int_x \int_y |r(x,y)|\, p(x,y) \log \frac{p(x,y)}{p(x)p(y)} dydx.$$

Für die binäre Klassifikation ergibt sich dazu eine einfache Umsetzung, welche einen Spezialfall darstellt. Alle Beispiele, denen die korrekte Klasse zu geordnet wurde, finden keine Verwendung, um die Transinformation für den nächsten Schritt zu berechnen. Korrekt klassifiziert bedeutet nichts anderes als ein Residuum von null und daher ein entsprechendes Gewicht, während alle Fehlklassifikationen im gleichen Verhältnis einen Fehler machen und daher auch dasselbe Residuum und damit dasselbe Gewicht erhalten.

Etwas diffiziler gestaltet sich das Problem im Rahmen einer Approximationsaufgabe. Hier muss jedes Beispiel mit einem kontinuierlichen Wert gewichtet werden, welcher durch das Residuum geliefert wird. Eine Normierung dieser Gewichte ist nicht zwingend erforderlich, da das korrekte Verhältnis bei der Berechnung der gewichteten Transinformation ausreichend ist.

Eine Pseudocodedarstellung zur Merkmalsselektion mittels der gewichteten Transinformation ist unter Algorithmus 7 zu finden.

Auch hier lassen sich die zwei Hauptargumente zur Verwendung des Residuums wieder einbringen. Erstens werden Redundanzen durch dieses Verfahren implizit berücksichtigt. Alle Beispiele, zu denen in den bereits gewählten Kanälen Informationen vorliegen, werden ein geringes Residuum aufweisen und damit kaum in die Berechnung des nächs-

Algorithmus 7 $S = \text{wRMI}\ (X, Y)$

Eingabe: Datensatz von Beobachtungen X und die entsprechenden Labels Y

Ausgabe: Menge von gewählten Merkmalen S und der finale Klassifikator

$S \leftarrow \emptyset$
$R \leftarrow 1$
$r(x, y) = 1; \forall (x, y)$
while Abbruchkriterium nicht erfüllt **do**
  $X_{max} = \arg\max_{X_i} \left[\int_x \int_y |r(x_i, y)|\, p(x_i, y) \log \frac{p(x_i,y)}{p(x_i)p(y)} dy dx_i \right]$
  $S \leftarrow S \cup X_{max}$
  $X \leftarrow X \setminus X_{max}$
  Classifier $\leftarrow$ TrainClassifier(S, Y)
  $r(x, y) =$ApplyClassifier(Classifier$, x) - y$
end while

ten Merkmals einbezogen. Die redundanten Kanäle können demzufolge auch keinen hohen Wert für die gewichtete Transinformation erreichen.

Zweitens wird auch der Bias des verwendeten Approximators berücksichtigt, da ein nützlicher Kanal Informationen über die Beispiele enthält, die aufgrund des Biasfehlers noch nicht korrekt gelernt wurden.

3.6.4. Experimente

Um die bisher gewonnenen Erkenntnisse über die Verfahren zu bestätigen und zu vertiefen, wurden auch hier Experimente durchgeführt. Um die Konsistenz der Ergebnisse zu gewährleisten, folgen diese Untersuchungen dem bereits in Abschnitt 3.5.5 vorgestellten Schema. Auch die Ergebnistabellen dieses Abschnitts werden hier fortgeschrieben.

Auffällig ist hierbei die Anzahl der durchlaufenen Trainingszyklen für den jeweiligen Klassifikator. Diese ist immer nur einen Durchlauf höher,

als die Anzahl der ausgewählten Merkmale und damit deutlich geringer als bei der einfachen Vorwärtsauswahl oder der Auswahl über Chow-Liu Bäume. Das heißt, dem Ziel, diese Zahl so niedrig wie möglich zu halten, ist man hier näher gekommen. Leidet darunter die Qualität der Auswahl?

Betrachtet man Tabelle 3.8, so fällt auf, dass der Nächste-Nachbar Klassifikator in Kombination mit der RMI1 Methode durchweg die schlechtesten Ergebnisse erzielt (RMI2 wurde aus den weiter oben diskutierten Gründe nicht mit aufgenommen). Bei der Verwendung eines mehrschichtigen Vorwärtsnetzes tritt diese Dissonanz jedoch nicht zutage (siehe Tabelle 3.9). Dieses Verhalten liegt darin begründet, dass der Nächste-Nachbar Ansatz, im Gegensatz zum globalen Funktionsapproximator eines MLPs, auf lokalen Nachbarschaften basiert.

Bei der Verwendung von lokalen Nachbarschaften verändert sich in jedem Schritt, in dem eine neue Dimension hinzugenommen wird, diese Nachbarschaft, was unter Umständen auch wieder zu einer Verschlechterung des Ergebnisses führen kann. Zwei Beispiele gleicher Klasse, die auf einer zweidimensionalen Ebene direkt nebeneinander lagen, sind unter Umständen im 3D-Raum weit voneinander entfernt, da sie unterschiedliche Höhen haben. Die Auswahl dieser Höhendimension erfolgte allerdings nur unter dem Gesichtspunkt der Beispiele, welche falsch klassifiziert wurden - nicht danach, dass diese neue Dimension durch neue Nachbarschaftsverhältnisse eventuell mehr Fehler produzieren könnte. Daher muss man nach diesen Experimenten von der Verwendung des RMI1 und auch des RMI2 Verfahrens mit lokalen Klassifikatoren abraten.

Für einen globalen Approximator hingegen reduziert sich das Problem darauf den Unterraum ohne die neue Variable wiederzufinden, um eine Verschlechterung zu vermeiden. Im den Experimenten mit dem MLP als Klassifikator zeigt sich, dass das Verfahren konkurrenzfähig ist. Der RMI.2 Ansatz erreicht zwar teilweise bessere Ergebnisse als die Refe-

Datensatz	Ionosphere			Spambase			GermanCredit			Breast Cancer		
Merkmale	34			57			24			30		
Beispiele	351			4601			1000			569		
	BER	F	TV	BER	F	TV	BER	F	TV	BER	F	TV
All	23.78	34	-	10.84	57	-	36.33	24	-	3.55	30	-
MIFS	11.80	5	-	8.65	19	-	33.90	6	-	4.36	5	-
SFS	12.04	5	189	8.44	12	663	31.61	7	164	4.21	6	189
CLT-FS	12.19	6	39	15.97	6	76	34.89	5	28	4.42	4	35
RMI.1	13.82	5	6	23.62	3	4	35.45	5	6	4.49	3	4
wRMI	11.57	5	6	10.73	10	11	33.31	8	9	4.48	6	7

Tabelle 3.8.: Ergebnisse für verschiedene Merkmalsselektionstechniken. Als Klassifikator wurde ein 3-Nächster-Nachbarklassifikator mit 10-facher Kreuzvalidierung verwendet. Für jedes Verfahren sind die Balanced Error Rate (BER), die Anzahl der gewählten Merkmale (F) und die Anzahl der Trainingsvorgänge (TV) gezeigt. Die Zeile All beschreibt die Verwendung aller Merkmale und MIFS ist die Mutual Information for Feature Selection aus Algorithmus 1. Diese beiden Ansätze benötigen zur Merkmalsauswahl keine Trainingsvorgänge. SFS ist die sequentielle Vorwärtssuche aus Algorithmus 2 und CLT-FS ist das Chow-Liu Baum basierte Verfahren aus Algorithmus 4. RMI.1 und wRMI wurden in diesem Abschnitt in den Algorithmen 5 und 7 vorgestellt.

Datensatz	Ionosphere			Spambase			GermanCredit			Breast Cancer		
Merkmale	34			57			24			30		
Beispiele	351			4601			1000			569		
	BER	F	TV	BER	F	TV	BER	F	TV	BER	F	TV
All	20.08	34	-	13.81	57	-	41.70	24	-	13.78	30	-
MIFS	24.54	5	-	16.29	19	-	37.47	6	-	12.48	5	-
SFS	18.47	3	130	17.39	8	477	39.06	4	110	13.44	4	140
CLT-FS	18.12	6	38	17.26	9	97	38.52	3	24	9.37	8	37
RMI.1	17.08	5	6	13.93	54	55	39.73	15	16	8.58	5	6
RMI.2	18.52	4	5	17.15	12	13	39.68	15	16	9.21	4	5
wRMI	16.97	5	6	16.41	9	10	39.52	6	7	8.03	3	4

Tabelle 3.9.: Ergebnisse für verschiedene Merkmalsselektionstechniken. Als Klassifikator wurde ein Multi-Layer Perceptron mit zwei Hiddenschichten bei 10-facher Kreuzvalidierung verwendet. Dabei wurden die Ergebnisse über drei Gesamtdurchläufe gemittelt. Für jedes Verfahren sind die Balanced Error Rate (BER), die Anzahl der gewählten Merkmale (F) und die Anzahl der Trainingsvorgänge (TV) gezeigt. Die Zeile All beschreibt die Verwendung aller Merkmale, MIFS ist die Mutual Information for Feature Selection aus Algorithmus 1, SFS die sequentielle Vorwärtssuche aus Algorithmus 2 und CLT-FS ist das Chow-Liu Baum basierte Verfahren aus Algorithmus 4. RMI.1 und .2 und wRMI wurden in diesem Abschnitt in den Algorithmen 5-7 vorgestellt.

renzverfahren, bleibt aber immer hinter den anderen beiden residuumsbasierten Verfahren zurück.

Die Verwendung der gewichteten Transinformation erzielt durchweg gute bis sehr gute Ergebnisse, auch die Problematik der lokalen Klassifikatoren tritt hier nicht zu Tage.

Daher ergibt sich als Empfehlung aus diesem Abschnitt, das Verfahren mit der gewichteten Transinformation einzusetzen - es verwendet nur sehr wenige Trainingsvorgänge und erreicht Ergebnisse, die auf Augenhöhe mit den anderen Verfahren liegen oder besser sind.

3.7. Merkmalstransformation

Bisher wurden Verfahren dargestellt, die mittels der Transinformation und verwandter Konzepte eine Auswahl von Merkmalen trifft. Allerdings kann es, wie schon zu Beginn des Kapitels bemerkt, sinnvoll sein, Merkmale zu transformieren. Dies ist insbesondere dann der Fall, wenn in den Eingangskanälen davon ausgegangen werden kann, dass es zwischen den Kanälen nachbarschaftliche Beziehungen gibt und die informationstragenden Elemente nicht in wenigen Variablen akkumuliert sind, sondern sich über viele Kanäle verteilen.

Dies ist beispielsweise bei Bildern der Fall, wenn jede Pixelposition als Eingangsvariable aufgefasst wird. Die Pixel stehen in Beziehung zueinander und erst eine gewisse Menge an Pixeln ermöglicht es, den Bildinhalt zu erschließen. Das Auswählen einzelner Pixel als relevante Kanäle ist oftmals wenig sinnvoll.

Trotzdem soll, gerade bei Bildern, die Zahl der Merkmale deutlich verringert werden. Dazu werden die Bilder „verlustbehaftet komprimiert“, in dem alle für die Aufgabe irrelevanten Teile weggelassen werden. Das klassische Beispiel für eine solche Merkmalstransformation ist dabei

die Hauptkomponentenanalyse (Principal Component Analysis - PCA, auch Karhunen-Love Transformation)[Pearson, 1901], ein Standardverfahren aus der multivariaten Statistik.

Bei diesem Verfahren werden die Raumrichtungen, in denen die größten Varianzen der Daten auftreten, gesucht und mit deren Hilfe ein neues, orthogonales Basissystem aufgespannt. Jede zusätzliche Raumrichtung trägt weniger zum Gesamtvarianzgehalt der Daten bei, und daher werden zum Zwecke der Dimensionsreduktion jene Achsen mit geringen Varianzen weggelassen. Praktisch kann dies über die Eigenwertzerlegung der Datenkovarianzmatrix erfolgen oder mit neuronalen Approximationstechniken [Sanger, 1989]. Die eigentliche Merkmalstransformation erfolgt dann durch die lineare Projektion der Daten in das neue Basissystem. Als Folge dieses Vorgehens sind die Daten dort dekorreliert.

Um auch nichtlineare Zusammenhänge entflechten zu können, gibt es auch nichtlineare Erweiterungen z.B. basierend auf autoassoziativen, mehrschichtigen neuronalen Netzwerken [Kramer, 1991] oder auf der Transformation im Kernelraum [Schölkopf et al., 1998].

Diese Transformation, linear als auch nichtlinear, basiert auf der Grundannahme, dass die Varianz in den Daten auch der relevanten Information entspricht. Diese Annahme ist problematisch, falls Rauschen die Ursache für die hohe Varianz ist und zumindest suboptimal für jene Fälle, in denen zusätzliche Informationen zur Verfügung stehen, denn bei der klassischen Hauptkomponentenanalyse handelt es sich um ein unüberwachtes Verfahren.

Natürlich gibt es auch andere Transformationsverfahren, die andere Kriterien anstelle der Varianz optimieren. Zu den bekanntesten gehören die Unabhängige Komponentenanalyse (Independent Component Analysis - ICA) [Hyvärinen et al., 2001], welche versucht, statisch unabhängige Datenrichtungen zu finden, oder die Nichtnegative Matrixfaktorisierung (NMF) [Lee und Seung, 2000], welche nur positive Komponenten zulässt, da es in vielen Anwendungen keine gute Begrün-

dung für negative Komponenten (z.B. Negativbilder) gibt. Auch diese Verfahren sind dabei unüberwacht.

Möchte man allerdings ein Klassifikationsproblem lösen, ermöglicht das Vorhandensein von Zielwerten, die Transformation auf das für die Klassifikation Wesentliche auszurichten. Die bekannteste Version ist dabei die Lineare Diskriminanzanalyse (LDA), welche auf dem Fisher-Kriterium basiert [Fisher, 1936]. Dabei wird jene Transformation gesucht, die die beste lineare Trennbarkeit der Klassen ermöglicht [Fukunaga, 1990]. Dies führt in vielen Fällen zu besseren Klassifikationsergebnissen als unüberwachte Verfahren. Dennoch können auch Szenarien konstruiert werden, in denen die PCA günstigere Ergebnisse liefert [Martinez und Kak, 2001].

Im weiteren Verlauf soll nun ein ähnliches, ebenfalls überwacht lernendes Verfahren näher beleuchtet werden. Dieses nutzt jedoch anstelle des Fisher-Kriteriums die quadratische Transinformation und wurde in [Torkkola, 2003] vorgestellt. Hierbei wird vorher festgelegt, wie hochdimensional der Unterraum sein soll, in den die Daten transformiert werden. Dieser Unterraum wird im Gesamtdatenraum dann mittels eines Gradientenverfahrens solange gedreht, bis die Quadratische Transinformation ein lokales Maximum erreicht.

Der generelle Ablauf dieses Ansatzes, der Transinformationsmaximierung (TIM), wird in Abbildung 3.14 gezeigt.

3.7.1. Quadratische Transinformation

Die Formulierung der Quadratischen Transinformation basiert dabei auf der Korrelationsentropie (siehe Rényi-Entropie Def. 3.5)

$$H^2(X) = -\log \sum_x p(x)^2.$$

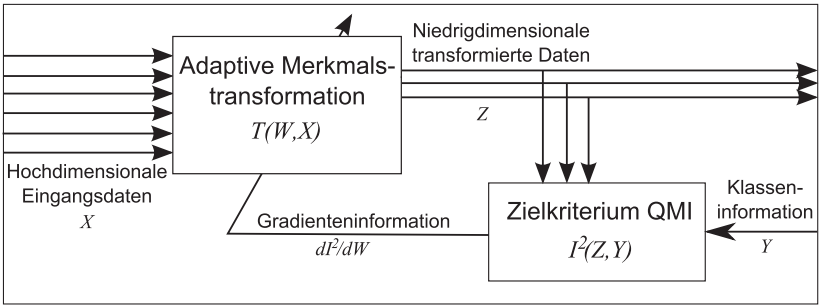


Abbildung 3.14.: Genereller Ablauf der Transinformationsmaximierung. Die gegebenen hochdimensionalen Eingangsdaten X werden, mittels einer Transformation T und deren Parameter W in einen niedrigdimensionaleren Raum überführt. Mit den transformierten Datenpunkten Z kann nun ein Klassifikator/Approximator trainiert werden. Um die in den transformierten Daten enthaltene Information zu den Zielwerten Y zu maximieren, muss die Transformation schrittweise angepasst werden. Dazu wird der momentane Informationsgehalt I^2 bestimmt, und dieser nach den Parametern der Transformation abgeleitet. Nach Anpassung der Parameter W kann erneut die Transformation der Ausgangsdaten durchgeführt werden, welche nun einen höheren Informationsgehalt besitzt. Sobald die Optimierung konvergiert und der Parametersatz sich nicht mehr ändert, kann in die Anwendungsphase übergegangen werden, in der nur noch die Transformation der Daten stattfindet, bevor diese an das nachfolgende Modul weitergereicht werden.

Die Kombination dieser Formulierung mit der Kerneldichteschätzeung (siehe Abschnitt 3.3.1 und Def. 3.17) ermöglicht es die Schätzung als reine Summe paarweiser Interaktionen zu formulieren. Dies war ein fundamentales Ergebnis aus [Principe et al., 2000], welches erst die von Principe propagierte Form des informationstheoretischen Lernens ermöglichte. Basierend auf diesem Ansatz wurde in [Torkkola, 2003] eine quadratische Form der Transinformation abgeleitet.

Definition 3.27

QUADRATISCHE TRANSINFORMATION

Die Quadratische Transinformation nach Torkkola ist definiert als

$$I^2(X;Y) = \int_x \int_y (p(x,y) - p(x)p(y))^2 dydx.$$

Ohne hier auf die Details eingehen zu wollen, ist die Idee dabei wie bei der Kullback-Leibler Formulierung (Def. 3.9) den „Abstand“ zwischen der Verbundverteilung und dem Produkt der Marginale zu bestimmen, denn dieses Divergenzmaß wird als Grad der (Un-)Abhängigkeit der beiden Variablen betrachtet (siehe auch der Diskussion zu diesem Thema in Abschnitt 3.2). Die quadratische Form ähnelt dabei rein formal der euklidischen Distanz zwischen $p(x,y)$ und $p(x)p(y)$, nur dass es sich um Verteilungen und nicht um Punkte im Raum handelt. Es handelt sich dabei auch nicht um ein Distanzmaß sondern nur um ein Divergenzmaß und bei der Herleitung werden zum Teil Konstanten vernachlässigt. Für Details sei auf [TORKKOLA, 2003] und die Referenzen dort in Abschnitt 4.1 verwiesen.

Die Formel 3.27 wird nicht direkt berechnet, sondern unter Anwendung der binomischen Formel ausmultipliziert

$$\begin{aligned} I^2(X;Y) = & \underbrace{\int_x \int_y p(x,y)^2 dydx}_{V_{IN}} + \underbrace{\int_x \int_y (p(x)p(y))^2 dydx}_{V_{ALL}} \\ & - \underbrace{2\int_x \int_y p(x,y)p(x)p(y)dydx}_{V_{BTW}} \\ = & V_{IN} + V_{ALL} - 2V_{BTW} \end{aligned}$$

Dieser Schritt erlaubt es, die markierten Teilterme einzeln zu berechnen und ermöglicht später eine grafische Interpretation des Ansatzes, was dann auch die Bedeutung der Bezeichner klarmacht. Eine notwendige Einschränkung, die an dieser Stelle gemacht wird, ist es, dass die Eingangsvariablen zwar kontinuierliche Wertebereiche haben dürfen, für die Zielwerte wurden jedoch noch diskrete Verteilungen, also Klasseninformationen zugelassen. Somit sind alle Integrale über y als Summe aufzufassen

$$I^2(X;Y) = \int_x \sum_y p(x,y)^2 dx + \int_x \sum_y (p(x)p(y))^2 dx - \int_x \sum_y p(x,y)p(x)p(y)dx.$$

Damit muss der komplizierte Teil der Dichteschätzung nur im eindimensionalen Fall durchgeführt werden. Im Falle von kontinuierlichen Zielwerten, beispielsweise bei Approximationsaufgaben, kann im einfachsten Fall eine Diskretisierung mit Histogrammen geschehen (siehe Abschnitt 3.3).

3.7.2. Transinformationsmaximierung

Die Idee dieses Ansatzes besteht darin, Raumrichtungen zu suchen, in denen sich die maximale Information (im Sinne der informationstheoretischen Definition) über das Ziel befindet. Als Maß für diese Menge an Information dient die eben eingeführte Quadratische Transinformation. Zur Maximierung dieser kommt nun ein iteratives Gradientenverfahren zum Einsatz.

Die Transformation selbst kann dabei eine klassisch lineare Transformation (wie bei z.B. PCA oder LDA) sein. Allerdings lassen sich in diesem

Framework auch sehr einfach nichtlineare Transformationen einbringen. Torkkola selbst nutzte hierbei neuronale Netze mit radialen Basisfunktionen [Torkkola, 2003] oder einfachen Multi-Layer Perceptrons [Torkkola, 2001]. Im Rahmen der Diplomarbeit von Ronny Niegowski [Niegowski, 2007] wurden in diesem Zusammenhang auch mit partiell rekurrenten Elman-Netzen experimentiert.

Für alle Möglichkeiten der Transformation T gilt, dass sie einen Satz von Parametern W beinhalten, welche die Transformation steuern. Dies sind z.B. die Matrixeinträge bei der linearen Transformation oder die Gewichte eines neuronalen Netzes. Diese werden nun schrittweise angepasst, so dass sie in Richtung der relevanten Merkmale zeigen. Dazu ergibt sich folgende Aktualisierungsregel

$$W(t+1) = W(t) + \eta \frac{\partial I^2}{\partial W},$$

wobei η die Lernrate ist. Die Information zwischen Eingangskanälen und Zielen I_2 wird nach den Parametern der Transformation W abgeleitet. Dazu ist es notwendig, die Transformation durchzuführen, also die Samples x_i in den neuen (Unter-)raum abzubilden. In diesem Raum werden die transformierten Beispiele mit z_i bezeichnet. Für diese Beispiele z_i kann die Transinformation $I_2(Z;Y)$ berechnet werden. Die Notwendigkeit der Durchführung dieses Zwischenschritts lässt es zu, die Aktualisierungsgleichung umzuschreiben:

$$W(t+1) = W(t) + \eta \frac{\partial I^2}{\partial W} = W(t) + \eta \sum_{z_i} \frac{\partial I^2}{\partial z_i} \frac{\partial z_i}{\partial W}.$$

Durch diese Aufsplittung wird erreicht, dass die Berechnung der Gradienteninformation aus den Datenbeispielen $\frac{\partial I_2}{\partial z_i}$ unabhängig von der Anpassung der Parameter der verwendeten Transformation $\frac{\partial z_i}{\partial W}$ ist, d.h. der zweite Teil ist transformationsspezifisch.

Für die Gradientenberechnung kann man nach obigen Überlegungen auch folgendes Schreiben

$$\frac{\partial I^2}{\partial z_i} = \frac{\partial V_{IN}}{\partial z_i} + \frac{\partial V_{ALL}}{\partial z_i} - 2\frac{\partial V_{BTW}}{\partial z_i}.$$

Um diesen Ausdruck berechnen zu können, wird die Verteilung $p(y)$ gebraucht, welche als diskret angenommen wird und damit unproblematisch ist, sowie die Verteilungen $p(y, z)$ und $p(z)$. Für deren Berechnung kommt wieder der Ansatz der Kerneldichteschätzung, wie in Abschnitt 3.3.1 erläutert, zum Einsatz. Für die Details sei an dieser Stelle auf Anhang A.1 verwiesen.

Das Interessante an dieser Darstellung ist, dass sie eine Interpretation als Potentialfeld erlaubt, z.B. wie physikalische Teilchen, die sich gegenseitig anziehen und abstoßen. Dabei steht V_{ALL} für alle Interaktionen, die zwischen allen Teilchen wirken, V_{IN} für Interaktionen, die zwischen Teilchen derselben Klasse wirken und V_{BTW} beschreibt die Interaktionen, die zwischen Teilchen unterschiedlicher Klassen wirken. Bildet man die partiellen Ableitungen dieser Potentiale erhält man *Informationskräfte*, die anzeigen, in welche Richtung sich die Teilchen bewegen müssten um das Potential zu maximieren. Siehe dazu auch Abbildung 3.15.

Für die Datenpunkte sagt dies aus, wo sie sich hinbewegen müssten, um die Quadratische Transinformation zur Klasseninformation zu maximieren. Dieses Wissen wird genutzt, um die Transformationsparameter zu aktualisieren. Für den Fall einer linearen Transformation

$$z_i = W^T x_i$$

ergibt sich die Ableitung dieser Informationskräfte nach den Parametern der Matrix W als

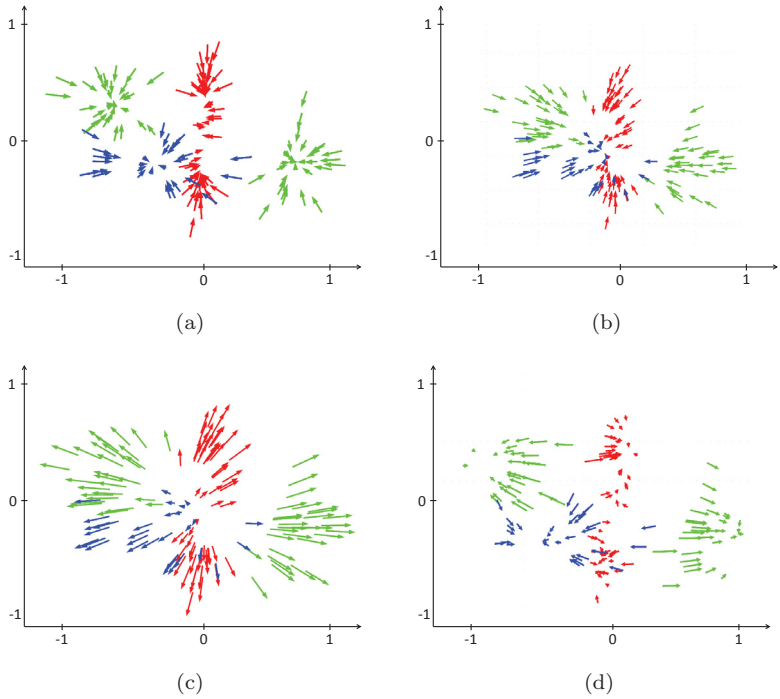


Abbildung 3.15.: Die Bilder zeigen die einzelnen wirkenden Teilkräfte und die resultierende Gesamtkraft. **(a)** $\frac{\partial V_{IN}}{\partial z_i}$ Alle Beispiel derselben Klasse ziehen sich untereinander an. **(b)** $\frac{\partial V_{ALL}}{\partial z_i}$ Alle Beispiele ziehen sich an. **(c)** $\frac{\partial V_{BTW}}{\partial z_i}$ Beispiele verschiedener Klassen stoßen sich ab und **(d)** als Summe der Teilkräfte $\frac{\partial I^2}{\partial z_i} = \frac{\partial V_{IN}}{\partial z_i} + \frac{\partial V_{ALL}}{\partial z_i} - 2\frac{\partial V_{BTW}}{\partial z_i}$. Quelle: [Niegowski, 2007].

$$\frac{\partial z_i}{\partial W} = x_i^T.$$

W ist dabei eine $|X| \times d$ (Anzahl der Eingangsvariablen und gewählte Unterraumdimensionalität).

Die Initialisierung dieser Matrix zu Beginn des Algorithmus kann dabei auf unterschiedliche Arten erfolgen. Die einfachste Form wäre eine

Algorithmus 8 Transinformationsmaximierungsschritt

Eingabe: Datensatz von Beobachtungen X und die entsprechenden Labels Y, sowie die momentane Transformationsmatrix W_t

Ausgabe: Neue Transformationsmatrix W_{t+1}

$Z = g(W, X) = W^T X$ // Durchführen der Transformation g auf den Originaldaten

$\frac{\partial I^2}{\partial z_i} = \frac{\partial V_{IN}}{\partial z_i} + \frac{\partial V_{ALL}}{\partial z_i} - 2\frac{\partial V_{BTW}}{\partial z_i}$ //Bestimmung der Informationskräfte

$\frac{\partial z_i}{\partial W} = x_i^T$ // Gradientenupdate für die lineare Transformation

$W_{t+1}^{'} = W_t + \alpha \frac{\partial I}{\partial W} = w_t + \alpha \sum_{i=1}^{N} \frac{\partial I}{\partial z_i} \frac{\partial z_i}{\partial W}$ //Aktualisierungsschritt

$W_{t+1} = \textsc{GramSchmidt}(W_{t+1}^{'})$ //Orthonormalisierung für $W^T W = \mathrm{I}$

zufällige Initialisierung. Allerdings führt dies schnell zu lokalen Maxima, die unerwünscht sind (siehe Abbildung 3.16). Dies kann vermieden werden, indem die Unterraumsuche mit dem Ergebnis eines anderen Verfahrens, wie der PCA oder LDA, initialisiert wird.

Damit kann nun der Ablauf der Transinformationsmaximierung als Algorithmus 8 formuliert werden, eine grafische Interpretation ist in Abbildung 3.14 gezeigt.

Wichtig ist, dass neben den bisher besprochenen Schritten nach dem Aktualisieren der Transformationsmatrix ein Orthonormalisierungsschritt eingefügt wird. Dazu wird ein Standard Gram-Schmidt Algorithmus verwendet. Dies führt dazu, dass der Projektionsunterraumdimensionen senkrecht aufeinander stehen und dass für die Matrix W gilt $W^T W = \mathrm{I}$.

Für die Laufzeit des Algorithmus von entscheidender Bedeutung sind dabei die Anzahl der zur Verfügung stehenden Trainingsbeispiele und die Berechnung der Informationskräfte daraus. Daher wird von Torkkola vorgeschlagen, bei größeren Datenmengen für jeden Durchlauf nur eine zufällig gezogene Teilmenge zu betrachten. Dabei muss sicherge-

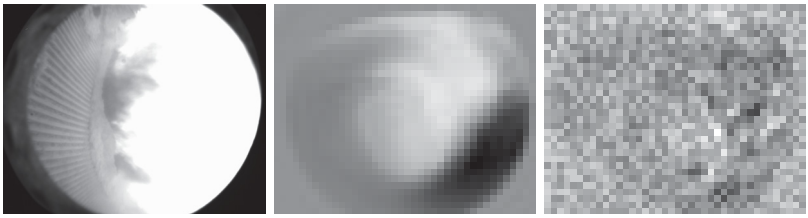


Abbildung 3.16.: Bildbeispiel aus dem Szenario der intelligenten Feuerungsführung (Kapitel 6. **(Links)** Beispielhafte Aufnahme aus einem Verbrennungsofen. **(Mitte)** Lokales Maximum welches eine sinnvolle Transformationsmaske darstellt. Der Zusatzschritt zur Glättung mittels eines 2D-Filters wurde während des Lernens durchgeführt. **(Rechts)** Unnützes lokales Maximum an dem der Algorithmus terminierte. Dabei kam die Filterung nicht zum Einsatz. Die Darstellung ist dabei analog einer Eigenraumdarstellung bei der Hauptkomponentenanalyse. Allerdings wird hier nicht die Varianz im Bild gezeigt, sondern der Informationsgehalt. Dargestellt ist die erste Dimension (analog der ersten Hauptkomponente) des neuen Unterraums. Beide Ergebnisse wurden mit zufälliger Startinitialisierung in einem Eingaberaum der Größe 40x32 Pixel erzeugt. Weiße Pixel kodieren positive Werte, schwarze analog negative Werte, während graue Pixel nahe null sind. Das Vorzeichen sagt dabei nichts über die Wichtigkeit, daher sind sowohl schwarze als auch weiße Gebiete von Interesse, während die grauen Werte unwichtig sind.

stellt werden, dass die Klassen auch anteilig in der gezogenen Untermenge korrekt repräsentiert sind. Dieses Vorgehen erhöht die Anzahl der Maximierungsschritt bis zur Konvergenz etwas, verringert aber die Laufzeit deutlich.

Anwendung auf Bilddaten

Bei der Anwendung der Transinformationsmaximierung (TIM) auf Bilddaten entsteht ein Problem. Bilder, bei denen jeder Pixel als einzelner Eingangskanal aufgefasst wird, bilden einen riesigen Eingaberaum.

Dies sprengt schnell den Rahmen der verfügbaren Rechenzeit zur Bestimmung der Transformation und erhöht drastisch die Chance in ungünstigen lokalen Maxima während des Lernens zu terminieren.

Allerdings widerspricht die Annahme unabhängiger Eingangskanäle der Nachbarschaftsbeziehung von Pixeln im Bild. Benachbarte Pixel zeigen häufig auch ähnliche Informationen. Um diesen Zusammenhang während des Lernprozesses nutzen zu können, wurde untersucht, inwieweit das Einbringen einer zusätzlichen Information dabei hilft, lokale Minima zu vermeiden.

Das Einbringen dieser Nachbarschaft erfolgt über die Verwendung eines Gaußfilters, der genutzt wird um die Transformationsmatrix W nach jedem Aktualisierungsschritt, aber vor der Orthonormalisierung, zu glätten. Im Falle eines Bildes entsprechen die Spalten der Transformationsmatrix Bildmasken (analog sind die Eigenvektoren als Äquivalent bei einer Hauptkomponentenanalyse zu sehen) und somit wird in diesem Fall auch ein 2D-Gaussfilter eingesetzt. Für Daten die einen anderen Zusammenhang vermuten lassen, kann hier natürlich variiert werden, beispielsweise beim Powerspektrum einer diskreten Fouriertransformation, wo man einen Zusammenhang benachbarter Frequenzen erwarten kann. In diesem Fall wäre ein eindimensionaler Gaußfilter zu wählen.

Ein Beispiel aus dem Szenario der intelligenten Feuerungsführung (Kapitel 6) ist in Abbildung 3.16 gezeigt. Die mittlere Darstellung ist ein Ergebnis, welches unter Verwendung des Gaußfilters erzielt wurde, während das rechte Ergebnis ohne diesen zusätzlichen Glättungsschritt auskommen musste.

3.7.3. Untersuchungen

Zuerst wurde auf künstlich erzeugten Daten untersucht, inwieweit die Dimensionalität des Zielraums Einfluss auf das Ergebnis haben. Jeweils

untersetzt wurde dies mit Untersuchungen an Bildmaterial aus dem Feuerungsführungsszenario.

Grundsätzlich schwierig sind hier Aussagen zu bringen, welchen quantitativen Vorteil das jeweilige Verfahren bringt, da dieser nicht direkt bestimmbar ist. Man könnte zwar direkt statistische Werte (Varianzen, Transinformation, etc.) über den transformierten Daten ausrechnen - allerdings ist dies nicht gerechtfertigt, da die Verfahren alle unterschiedliche Optimierungskriterien benutzen und daher nicht fair zu vergleichen sind. Es bleibt nur der Weg über das Training eines Klassifikators/Approximators und der Bestimmung des resultierenden Fehlers. Die Schwierigkeit hierin ist wieder, dass unklar bleibt, in welchem Ausmaß ein lernendes System eine suboptimale Transformation kompensieren kann. Daher wird hier auf die qualitativen Ergebnisse wertgelegt. Ein Vielzahl weiterer Experimente und Auswertungen zum Vergleich PCA, LDA und TIM finden sich in [Niegowski, 2007], dabei auch viele quantitative Angaben, die jedoch unter Berücksichtigung des Ebengesagten kritisch betrachtet werden müssen.

Dies ist beispielsweise die Dimensionalität der transformierten Daten. Hier zeigen die Untersuchungen, dass das Verfahren nach weniger Iterationen terminierte, je höherdimensional der Raum war. Dieses Verhalten lässt sich darauf zurückführen, dass mit zunehmendem Volumen des Raumes auch die Menge an lokalen Optima drastisch steigt, der Gradientenabstieg dort hängenbleibt und man in jedem Lauf zu unterschiedlichen Ergebnissen kommt. Bei sehr wenigen Dimensionen erreicht man hingegen stabil gleiche Ergebnisse.

Dabei handelt es sich wieder um eine Ausprägung des Problems der hohen Dimensionalität, denn die Berechnung der Quadratischen Transinformation findet im Unterraum nach der Transformation statt. Daher ist die Zieldimensionalität d eine entscheidende Größe. Je mehr Datenpunkte zur Verfügung stehen, desto eher werden auch in höherdimensionalen Räumen stabile Ergebnisse gefunden. Als Beispiel lag die

Schwelle bei einem vierdimensionalen Zielraum Dimensionen unter Verwendung von 1400 Bildern der Größe 32x40. Bei verdoppelter Anzahl von Bildern lag die Schwelle bis zu der stabile Resultate erzielt wurden bei fünf Dimensionen. Im Falle der Bilddaten lässt sich dies um eine weitere Dimension erhöhen, wenn der oben angesprochene, zusätzliche Schritt der Gaußfilterung eingebracht wird.

Daraus lässt sich der Schluss ableiten, dass eine niedrige Zieldimension bevorzugt werden sollte. Diese Aussage lässt sich durch eine weitere Beobachtung untersetzen. Es wurde untersucht, inwieweit sich die resultierenden Klassifikations- bzw. Approximationsprobleme in einem durch Transinformationsmaximierung erzeugten Unterraum besser lösen lassen, als beispielsweise durch PCA Unterräume. In Tabelle 3.10 zeigt sich, dass die Transinformationsmaximierung der PCA überlegen ist, wenn die Größe der Unterraumdimension d_y sehr klein ist. Mit zunehmender Dimensionalität gleichen sich die Approximationsfehler im PCA und TIM Unterraum an. Das heißt, der praktische Vorteil, den das komplexere Verfahren der Transinformationsmaximierung bietet, lässt sich nur bei sehr geringer Dimensionalität des neuen Unterraums erreichen.

Ein weiterer Effekt, der hierbei eine Rolle spielt, ist die Anzahl der diskreten Klassen, die der Zielwert Y vorgibt. Je größer diese Zahl ist, desto langsamer konvergiert das gesamte Verfahren. Die gezogene Schlussfolgerung ist, dass in hinreichend niedrigdimensionalen Räumen, die lineare Transformation mittels Transinformationsmaximierung der LDA und PCA überlegen ist, wenn auch auf Kosten einer höheren Rechenzeit.

Das Verfahren kann leicht auf nichtlineare Transformationen übertragen werden, indem die Ableitung $\frac{\partial z_i}{\partial W}$ z.B. mittels des Backpropagation-Algorithmus in ein neuronales Netz propagiert wird [Torkkola, 2001]. Dies kann zu besseren Ergebnissen führen. Allerdings müsste man die Vergleiche ebenso mit nichtlinearen Varianten der PCA und verwand-

	Fehler für CO		Fehler für O2		Fehler für NOx	
d_y	PCA	TIM	PCA	TIM	PCA	TIM
1	3.11	3.07	0.90	0.24	28.88	25.99
2	3.33	2.43	0.25	0.29	35.50	25.00
3	4.07	2.66	0.22	0.28	27.65	30.26

Tabelle 3.10.: Beispiel aus dem Feuerungsführungsszenario. Nach der Durchführung einer Hauptkomponentenanalyse (PCA) bzw. Transinformationsmaximierung (TIM) und einer Dimensionsreduktion auf d_y wird versucht, verschiedene Größen (Kohlenmonoxid, Restsauerstoff und Kohlendioxid) mittels eines Multi-Layer Perceptrons zu schätzen. Der resultierende mittlere quadratische Approximationsfehler für die Vorhersage der Größen ist in der Tabelle angegeben.

ter Verfahren durchführen um belastbare Aussagen zu erhalten, was im Rahmen dieser Arbeit jedoch nicht getan wurde

Fazit aus Sicht der Anwendung war jedoch, dass die lineare Transinformationsmaximierung im Fall einer möglichst großen Kompression der Daten auf sehr wenige Dimensionen der Hauptkomponentenanalysis und Linearen Diskriminanzanalyse vorzuziehen ist, da im Mittel die Approximationsergebnisse besser und damit die Nützlichkeit höher ist.

3.8. Merkmalsextraktion für Aktionsräume

Bisher bewegten sich die Ausführungen am Beginn des Wahrnehmungs-Handlungs-Zyklus. Im Sinne einer kognitiven Architektur stehen die auszuführenden Aktionen des Agenten am anderen Ende. Methodisch liegen sie allerdings sehr nah bei der Merkmalsextraktion, und daher soll an dieser Stelle auf das Problem der Aktionsraumauswahl näher eingegangen werden. Es geht dabei darum, einen gegebenen Aktionsraum, also die Menge aller Aktionsmöglichkeiten aufgespannt über allen

beeinflussbaren Stellgrößen, auf relevante und wesentliche Aktionen zu reduzieren. Die Intention dahinter ist dabei dieselbe, wie bei der Merkmalsextraktion - den Raum der Möglichkeiten einzuschränken, um den Suchraum für Lernverfahren zu verkleinern und somit schneller gute Lösungen des Problems zu finden.

Auf den ersten Blick scheint es sich dabei um die gleiche Aufgabenstellung wie bei der Merkmalsextraktion zu handeln, und somit sind auch die in diesem Kapitel vorgestellten Methoden hier genauso anwendbar. Die Unterschiede sind dabei praktischer Natur. Während es bei der Merkmalsselektion problemlos möglich ist, offline auf einem Datensatz die Relevanz der Eingangsvariablen zu unterschiedlichen Zielen zu bestimmen, erfordert dies auf der Aktionsseite auch immer ein Durchführen von Aktionen. Dies kann sich insofern als schwierig erweisen, dass meist eine Aktion mehrere Zielgrößen beeinflusst und dies nicht in jedem Fall für jedes Ziel unabhängig bewertet werden kann, wie bei der Merkmalsextraktion.

Falls dies im Anwendungsszenario durchführbar ist, können dazu Experimente durchgeführt werden, um die notwendigen Daten zu gewinnen. Wie solche Experimente anzulegen sind, um möglichst aussagekräftige Daten zu erhalten, sei hier auf das Feld optimalen Versuchsplanung verwiesen, so zum Beispiel [Kleppmann, 2006] oder [Montgomery, 2004].

Wenn genügend Daten zur Verfügung stehen, können die bisher besprochenen Verfahren oder andere Selektionsverfahren verwendet werden, um entweder Aktionen auszuwählen oder sie zu transformieren.

Die Selektion ist dabei verhältnismäßig einfach zu handhaben: Eine Stellgröße, die keinen messbaren Einfluss auf den Prozess und damit die Zielgrößen hat, ist irrelevant und kann damit aus dem Gesamtaktionsraum entfernt werden. Gleiches gilt für eine Aktion, welche zu einer zweiten Aktion exakt dasselbe Verhalten zeigt, also redundant ist.

Was aber bedeutet eine Transformation des Aktionsraums? Man kann Verfahren wie beispielsweise die Hauptkomponentenanalyse anwenden, allerdings muss hier dann von Anwendungsfall zu Anwendungsfall kritisch hinterfragt werden, was in diesem Zusammenhang die Hauptkomponenten bedeuten. Problem an den unüberwachten Verfahren ist, dass hierbei nicht beachtet wird, ob Stellgrößen einen Einfluss auf das Ziel haben, sondern nur die Varianz und Frequenz der Benutzung einer Stellgröße eine Rolle spielt.

Will man die Zielgröße mit einbeziehen, entsteht etwas, dass als parallele Makroaktion bezeichnet werden soll. Man denke hierbei an das Beispiel eines bremsenden Zuges. Jeder Wagon besitzt eine eigene Bremse, stellt also eine eigene Dimension im Aktionsraum dar. Soll der Zug anhalten, dann bremsen alle Wagons, fährt er an, sollten alle Bremsen gelöst sein. Zumeist macht es wenig Sinn, dass nur einzelne Wagen bremsen und andere nicht. Der Zugführer wird daher in den meisten Fällen alle Bremsen parallel betätigen, und nicht jede einzeln. Diese Aktion, Bremsen, ist dann eine Abstraktion des realen Stellraums. Die Umsetzung, dass durch die Aktion Bremsen alle vorhandenen Bremssysteme aktiviert werden, entspricht der parallelen Makroaktion und damit einer Transformation im Aktionsraum.

Formaler beschrieben wird die Komplexaktion A in eine Kombination aus Basisaktionen $b_1, ..., b_n$ übersetzt. Im einfachen Fall einer linearen Transformation könnte man schreiben $A = w_1 b_1 + w_2 b_2 + ... + w_n b_n$. Dabei sei $\sum_{i=1}^{n} w_i = 1$. Natürlich ist es auch möglich, dass der Raum der Komplexaktion mehrdimensional ist, dann ergibt sich in Matrixschreibweise folgende Form: $A = W^T B$. In den Parametern W steckt der Zusammenhang zwischen den Basisaktionen, z.B. Bremsen bedeutet, dass wenn Bremse A gedrückt wird auch Bremse B im gleichen Verhältnis betätigt werden muss. Um aus der gewählten Komplexaktion auf die Basisaktionen zu schließen muss also $B = W^{-1} A$ gelöst werden.

Allerdings ergibt sich hier bereits das erste Problem. In dieser Form gibt es mehrere Möglichkeiten das Gesamtziel zu erreichen, da die Gleichung unterbestimmt ist. Dies liegt daran, dass die abstrakte Aktion weniger Stellmöglichkeiten hat, als der komplette Stellraum. Daher gibt es einige Nebenbedingungen zu beachten, die bei der Bestimmung von W einzuhalten sind. Dies können beispielsweise Nichtnegativitätbedingungen sein (keiner der Wagons außer der Lok besitzt einen Antrieb, kann also „negativ bremsen“) oder Bedingungen, die sich aus Vorwissen ergeben (Um ungleichmäßige Abnutzung zu vermeiden, sollten alle Wagons mit ähnlicher Stärke bremsen).

Um dieses Problem sinnvoll zu lösen, ist es daher notwendig Expertenwissen zur Formulierung dieser Nebenbedingungen einzubringen.

3.9. Einordnung und verwandte Arbeiten

Zu allen in diesem Abschnitt vorgestellten Methoden und Untersuchungen wurden eigene wissenschaftliche Ergebnisse publiziert. Dies umfasst die Untersuchungen zur Schätzung der Transinformation zur Merkmalsselektion [Schaffernicht et al., 2010] (Abschnitt 3.3), die Merkmalsauswahl mit Chow-Liu Bäumen [Schaffernicht et al., 2007] (Abschnitt 3.5, die Verfahren zur Verwendung des Residuums [Schaffernicht et al., 2009a] [Schaffernicht und Gross, 2011] (Abschnitt 3.6) und auch die Merkmalstransformation für Bilddaten [Schaffernicht et al., 2009c] (Abschnitt 3.7).

Nachdem die Methoden inhaltlich vorgestellt und mit Experimenten untersetzt wurden, verbleibt die Frage, wie sich diese Neuerungen in das Gesamtgefüge der Forschung in diesem Feld einordnen. Dabei erhebt dieser Abschnitt nicht den Anspruch auf Vollständigkeit, da das Feld der automatischen Merkmalsselektion in ständiger Bewegung ist und immer neue Spielarten veröffentlicht werden. Zum Einstieg in das Feld

werden [Guyon und Elisseeff, 2003], [Kohavi und John, 1997] und [Koller und Sahami, 1996] empfohlen. Für eine grundsätzliche und aktuelle Übersicht zur Merkmalsextraktion sei auf [Guyon et al., 2006] verwiesen. Dort wird neben den Grundlagen auf aktuelle Weiterentwicklungen und Benchmarks auf verschiedenen künstlichen Datensätzen eingegangen.

Der erste Themenkomplex zur eigentlichen Schätzung der Transinformation findet sich kaum im Feld der Merkmalsextraktion. Diese Problematik wird zumeist in der Statistik und der Informationstheorie (z.B. *IEEE Transactions on Information Theory*) abgehandelt, jedoch auch oft in Zusammenhang mit dem Neurocomputing (viele Veröffentlichungen finden sich in der *Neural Computation*). Eine gute Übersicht zur Problematik ist in [Khan et al., 2007] gegeben. Verschiedene Verfahren werden in [Paninski, 2003], [Kraskov et al., 2004] oder [Bonachela et al., 2008] vorgestellt, wobei jeweils Wert auf die Behandlung der statistischen Fehler und des Bias gelegt wird. Der Standard in der Machine Learning Community ist dabei die von Kraskov vorgestellte und auch in Abschnitt 3.3 diskutierte Nächste-Nachbar Methode.

Diese Arbeit liefert in diesem Bereich keinen eigenen Beitrag, sondern überträgt die Problematik explizit auf das Problem der Merkmalsextraktion, was bisher nicht in dieser Form getan wurde. Es gibt zwar Veröffentlichungen wie [Fleuret, 2004] oder [Chow und Huang, 2005], welche explizit eine Transinformationsbestimmung für die Merkmalsselektion vornehmen, allerdings nicht wirklich den Vergleich mit anderen Methoden angehen. Zudem sind vorgestellten Methoden speziell auf das Auswahlverfahren zugeschnitten und lassen sich daher nur schwer auf andere Ansätze übertragen.

Im Bereich der Hybridansätze zur Kombination von Filter- und Wrapper Ansätzen gibt es eine Vielzahl von Arbeiten, die darauf abzielen durch eine clevere Kombination beider Paradigmen eine schnel-

le und nützliche Auswahl zu treffen. In [Estevez et al., 2009] wird mit der *Normalized mutual information feature selection* eine Weiterentwicklung des MIFS-Ansatzes vorgestellt, welcher dann mit einem genetischen Algorithmus zum GAMIFS Hybridverfahren kombiniert wird. Innerhalb der genetischen Suche wird der Mutationsoperator dabei durch die Transinformation kontrolliert. Ebenfalls eng verwandt ist der *Markov Blanket Enhanced Genetic Algorithm* Ansatz von [Zhu et al., 2007], bei welchem die genetischen Operatoren durch Approximation von Markov Blankets [Pearl, 1988] mittels Transinformation gesteuert werden. Ebenfalls eine Kombination von Transinformation und evolutionären Algorithmen wird in [Van Dijck und Van Hulle, 2006] vorgestellt, wobei hier der simpelste Fall angenommen wird, indem mittels der Transinformation eine Vorauswahl der Merkmalen stattfindet und der Suchalgorithmus den verringerten Suchraum erforscht.

Ein Verfahren, dass Verwandtschaft zur hier vorgestellten Auswahl mit Chow-Liu Bäumen hat, wird in [Sebban und Nock, 2002] vorgestellt. Dort wird ein minimaler Spannbaum basierend auf der geometrischen Nachbarschaft der Daten und der quadratischen Entropie berechnet. Die resultierende Struktur trifft dann Aussagen darüber, ob die Hinzunahme eines Merkmals zu einer Erhöhung der Klassendiskriminanz (ähnlich wie bei Klassifikationsbäumen [Breiman, 2001]) führt. Dies wird dann im Rahmen einer Vorwärtssuche auf Basis der Baumstruktur realisiert.

Andere Hybridansätze, die nicht zwingend auf informationstheoretischen Methoden basieren, aber denselben Gedanken verfolgen, werden beispielsweise in [Somol et al., 2006] vorgestellt. Dort wird die Bhattacharyya Distanz [Bhattacharyya, 1943] als Filterkriterium verwendet, um die Vorauswahl im Rahmen einer fließenden Vorwärtssuche (Floating Search [Reunanen, 2006]) durchzuführen. Interessant an diesem Ansatz ist die Existenz eines Hybridisierungsfaktors, der es

erlaubt, den Einfluss von Filter und Wrapperkomponente zu steuern. [Souza et al., 2005] kombiniert ein Wrapperbasisverfahren (dabei können unterschiedliche Methoden angewendet werden, solange sie gewissen Kriterien genügen) mit einer simplen stochastischen Filterkomponente und kann darauf basierend den Vorteil von Hybridansätzen demonstrieren.

In [Leung und Hung, 2010] wird argumentiert, dass selbst die Kombination von Filter und Wrapperansätzen nicht ausreichend ist, um generelle Aussagen zu treffen, denn implizit ist die Merkmalsauswahl klar am benutzten Approximator ausgerichtet. Daher schlagen die Autoren vor, mehrere Filter mit multiplen Wrappern zu kombinieren. Dabei können verschiedene Verfahren eingebracht werden, so auch jene, die in dieser Arbeit vorgestellt wurden.

Was die Verwendung des Residuums zur Merkmalsauswahl angeht, konnten in der Literatur keine verwandten Ansätze gefunden werden, die diese Idee ebenfalls verfolgen. Daher kann dieser Ansatz als neuartig eingestuft werden.

Was den Bereich der Merkmalstransformation angeht, so gibt es hier weit weniger Entwicklungen. In der praktischen Anwendung hat die klassische Hauptkomponentenanalyse [Pearson, 1901] nach wie vor einen sehr hohen Stellenwert. So nutzt beispielsweise der Sieger das Feature Selection Contest [Neal und Zhang, 2006], welcher auch im Rahmen von [Guyon et al., 2006] beschrieben wurde, als ersten Schritt eine Hauptkomponentenanalyse. Auf den transformierten Merkmalen wurden dann mittels *Bayes Neural Networks* oder *Dirichlet Diffusion Trees* die Merkmale bestimmt. Ebenfalls sehr populär sind natürlich die Unabhängige Komponentenanalyse (ICA) [Hyvärinen et al., 2001], die nichtnegative Matrixfaktorisierung [Lee und Seung, 2000] als auch die lineare Diskriminanzanalyse [Fukunaga, 1990], sowie deren Spielarten. Die vielen Veröffentlichungen in den letzten Jahren bis 2010 modifizieren diese Basisansätze zumeist durch eine kleine Änderung des

Optimierungskriteriums oder durch neue Algorithmen, die das jeweilige Optimierungsziel effizienter oder auf anderen Wegen erreichen.

Die Verwendung des *Informationtheoretic Learning (ITL) Frameworks* von [Principe et al., 2000] ist eher die Ausnahme. Dieses ist in der Lage, gezielte Transformation auf Zielgrößen vorzunehmen [Torkkola, 2003] (wie auch in diesem Kapitel vorgestellt), oder andere Verfahren, wie zum Beispiel die ICA, zu emulieren. Auch informationserhaltende Transformation in niedrigere Dimensionen sind möglich [Vera et al., 2010]. Inwieweit sich das sinnvoll als Merkmalstransformation nutzen lässt, hängt dabei von der Anwendung und dem genutzten Optimierungskriterium ab. Der Beitrag dieser Arbeit betrifft hierbei klar den Umgang mit Nachbarschaftsbeziehungen in den Daten, wie es bei Bilddaten der Fall ist.

3.10. Praktische Anwendungen

Um zu zeigen, dass die in diesem Kapitel dargestellten Ansätze auch praktisch von Nutzen sind, sollen hier kurz ein paar Anwendungsszenarien skizziert werden, in denen hier vorgestellte Verfahren zur Merkmalsselektion zur Anwendung kamen. Diese entstammen dem Bereich der Mensch-Maschine-Kommunikation und der intelligenten Regelung und sind detaillierter im Anhang der Arbeit beschrieben.

3.10.1. Schätzung von Nutzerinteresse aus Bewegungstrajektorien

In diesem Anwendungsszenario, wie es in der Diplomarbeit von Antje Ober[6] [Ober, 2007] und einer resultierenden Veröffentlichung

[6] Autor ist kein direkter Betreuer dieser Arbeit, sondern wurde nur beim Problem der Merkmalsselektion hinzugezogen.

[Müller et al., 2008] vorgestellt wurde, geht es um eine mobile Roboterplattform zur Mensch-Maschine-Interaktion. Eine der wichtigsten Entscheidungen, die ein solcher Roboter zu fällen hat, ist es, ob und wann er einen Interaktionsvorgang mit einer Person starten soll. Es ist nicht zweckdienlich alle Leute anzusprechen, sondern nur jene, die einer Interaktion nicht abgeneigt sind. Es wurde daher untersucht, inwieweit aus Trajektoriendaten dieses Interaktionsinteresse geschätzt werden kann. Laser-, Sonar- und Kameradaten werden in einem Personentracker geeignet fusioniert und zu Trajektorien verknüpft. In deren Verarbeitung ergibt sich ein breites Spektrum an möglichen Repräsentationsformen, was Referenzsystem, Koordinatensysteme, Samplingstrategien und Hauptkomponentenanalyse angeht. Dieser potentiell sehr große Merkmalsraum wurde nun mit einem Merkmalsranking basierend auf dem MIFS Algorithmus (siehe Definition 3.22) und der direkten Schätzung der Verbundtransinformation untersucht um festzulegen, welche Form der Vorverarbeitung für das Ziel der Nutzerinteressenschätzung die beste ist.

Mittels der Merkmalsextraktion wurden eine geeignete Koordinationtransformation und Merkmalstransformation durchgeführt, die das beste Klassifikationsergebnis zur Nutzeraufmerksamkeit ermöglichte. Hauptsächlich half dabei, dass viele redundante Kanäle in den unterschiedlichen Darstellungsformen reduziert werden konnten. Mehr Details finden sich dazu in Anhang B.1.

3.10.2. Schätzung von Emotionen aus Gesichtsbildern

Ein weiteres Szenario, in welchem ein einfaches Merkmalsranking zur Anwendung kam, um geeignete Vorverarbeitungsschritte zu bestimmen, kommt ebenfalls aus der Mensch-Maschine-Interaktion. Im Rahmen der Arbeiten von Christian Martin[7], wurden dazu in Bildern Ge-

[7] Bisher nicht veröffentlicht - Der Autor dieser Arbeit wurde zur Merkmalsauswahl hinzugezogen.

sichter gesucht und mittels eines Active-Appearance Modells (AAM) [Cootes et al., 1998] verfolgt. Dieses AAM besteht aus zwei Teilen, einem Formmodell und einem Appearancemodell. Das Formmodell ist dabei ein Graph, der einzelne markante Punkte des Gesichts in Relation zueinander bringt, während das Appearancemodell das Aussehen in Form von Grauwerten modelliert. Diese Modellinformationen werden typischerweise einer Hauptkomponentenanalyse unterzogen, und das Gesicht mit Pose und Mimik als Projektionsparameter beschrieben. Durch Variation des Formmodells (Anzahl und Anordnung der Knoten) und der verwendeten Projektion (Anzahl verwendeter Hauptkomponenten bzw. alternative Unterraumtransformationen wie unabhängige Komponentenanalyse oder nichtnegative Matrixfaktorisierung) ergibt sich auch hier ein sehr hochdimensionaler Merkmalsraum.

Einerseits wurde versucht mittels 10 Form- und 20 Appearanceparametern die Gesichter einer von sechs Basisklassen zuzuordnen. Dieses Problem lies sich mit einem Multi-Layer Perceptron mit zwei Hiddenschichten lösen. Mittels der Merkmalsselektion basierend auf MIFS, konnte die releavanten Parameter auf 8 eingegrenz werden und die Problemlösung war mit einem einfacheren Netz mit nur einer Hiddenschicht möglich.

Zum zweiten wurde eine Modellselektion durchgeführt, um einen niedrigdimensionalen Raum zu finden, in dem eine Kohonenkarte (SOFM) trainiert wurde. Ziel war es zu untersuchen, ob sich auf einer solchen SOFM die in der Literatur benannten psychologischen Emotionsmodelle wieder finden lassen. Es wurden verschiedene Repräsentationsformen in PCA und ICA Komponenten der Form- und Appearancemodelle untersucht. Dabei fanden sich Repräsentationsformen mit rund 6-8 Parametern, in denen sich die Gesichter auf der SOFM ähnlich gruppierten, das die Basisemotionen topologisch trennbar wurden.

3.10.3. Audiobasierte Nutzermodellierung

In der Diplomarbeit von Tobias Prüger[8] [Prüger, 2008] wurde untersucht, inwieweit sich mittels Methoden des maschinellen Lernens Nutzereigenschaften auf Basis von Sprachsignalen schätzen lassen. Geschätzt werden sollte dabei an Hand der Stimme wer der Nutzer ist, der emotionale Zustand des Nutzers (sechs Basisemotionen) und sein Stresszustand (physische und psychische Anspannung).

Dazu werden aus dem Sprachsignal eine Vielzahl von Merkmalen extrahiert, darunter die Grundfrequenz, MFCC (Mel Frequency Ceptral Coefficents), Formanten und weitere. Diese Vielzahl an möglichen Merkmalen sollte auf die wichtigen Merkmale reduziert werden. Ursprünglich sollte die Merkmalsauswahl mit einer einfachen Vorwärtssuche (siehe Abschnitt 3.4) durchgeführt werden. Eine vorsichtige Abschätzung zeigte jedoch, dass der verfügbare Zeitrahmen deutlich gesprengt werden würde. Daher wurde die Auswahl mit Chow-Liu Bäumen 3.5 als schnellerer Ansatz verwendet.

Aus den ursprünglich 300 Kanälen wurden 13 als nützlich für die Emotionserkennung angesehen, für die Sprecheridentifikation waren etwas mehr 50 Kanäle ausgewählt worden. Erst durch diese deutliche Reduktion war eine sinnvolle Klassifikation überhaupt möglich. Einen tieferen Einblick in dieses Thema gewährt der Anhang B.2.

3.10.4. Prädiktion des Schnittregisterfehlers einer Druckmaschine

Beim Schneiden der bedruckten Papierbahnen einer Buchdruckmaschine muss das Schnittmesser, welches am Ende die einzelnen Seiten zurechtschneidet, kontinuierlich angepasst werden. Wird dies nicht getan,

[8] Autor ist kein direkter Betreuer dieser Arbeit, sondern wurde nur beim Problem der Merkmalsselektion hinzugezogen.

kommt es mit der Zeit zu Verschiebungen auf der Papierbahn und die Seiten dürfen nicht an beliebiger Stelle zertrennt werden. Dieser sogenannte Schnittregisterfehler soll möglichst gering gehalten werden. Im Rahmen der Diplomarbeit von Christoph Möller [Möller, 2009] wurde untersucht, inwieweit ein nichtlineares neuronales Modell in der Lage ist diesen Fehler aufgrund von Sensoren entlang der Druckmaschine vorhersagen. Es sollte mittels einer Signifikanzanalyse untersucht werden, welche Sensoren notwendig sind und welche Sensoren der Testmaschine aufgrund von irrelevanten oder redundanten Informationen weggelassen werden können. Dazu kam wieder die einfache Merkmalsbewertung und auch die Residual Mutual Information (Abschnitt 3.6) zum Einsatz. Schlussendlich konnten rund zwei Drittel der original verfügbaren Sensoren vernachlässigt werden. Ausführlichere Informationen finden sich in Anhang B.3.

3.10.5. Feuerungsführung in einem Kohlekraftwerk

Eine Vielzahl von Sensoren und Kameras erzeugen in einem Kraftwerk Daten. Um mit diesen sinnvoll eine solche Anlage regeln zu können, muss die Anzahl der betrachteten Dimensionen auch hier drastisch reduziert werden. Dazu kommt für die Bild- und die Spektraldaten die MMI aus Abschnitt 3.7 zum Einsatz, deren Ergebnisse nachfolgend zusammen mit allen anderen Kanälen einer MIFS unterworfen wird. Auf dem solchermaßen reduzierten Datensatz wird dann ein Regler gelernt. Dieser entscheidet dann die Stelleingriffe. Der Stellraum wurde dabei auch mit den in Abschnitt 3.8 vorgestellten Methoden untersucht. Detaillierter wird dieses Szenario und die darin erzielten Ergebnisse in Kapitel 6 erörtert, da es die kognitive Gesamtarchitektur dieser Arbeit exemplarisch umsetzt.

	Nutzerinteresse	Emotionen aus Bilden	Emotionen aus Sprache	Schnittregisterfehler	Feuerungsführung
Transinformation	X	X	X	X	X
Verbundtransinformation/MIFS	X	X		X	X
Auswahl mit Chow-Liu Bäumen			X		X
Residual Mutual Information				X	
Transinformationsmaximierung					X
Aktionsraumauswahl					X

Tabelle 3.11.: Übersicht über die Anwendung der vorgestellten Verfahren in unterschiedlichen Szenarien.

Zusammenfassung

Mit den hier aufgezeigten Anwendungen wird deutlich, dass die Problematik der Merkmalsselektion in vielen Feldern von Bedeutung ist und genutzt werden kann. In Tabelle 3.11 werden die Szenarien und die verwendeten Ansätze noch einmal tabellarisch zusammengefasst.

3.11. Fazit

In diesem Kapitel wurde diskutiert, wie im Rahmen der Gesamtarchitektur wichtige Informationen von unwichtigen getrennt werden können. Dazu können Informationskanäle entweder im Rahmen eines Selektionsprozess ausgewählt oder durch eine sinnvolle Transformation komprimiert werden. Als zentrales Bewertungskriterium kam die Trans-

information zum Einsatz, welche, wie gezeigt wurde, auf unterschiedlichen Wegen aus den Daten geschätzt werden kann. Mit Hilfe dieser Größe wurden dann neue Verfahren zur schnellen Merkmalsselektion eingeführt, wobei Chow-Liu Bäume oder Informationen im Residuum zum Einsatz kamen. Ebenfalls findet die Transinformation Anwendung bei der Transinformationsmaximierung, welche speziell für Bilddaten untersucht und erweitert wurde. Die Methodiken wurden dann auf das analoge Problem der Aktionsraumselektion übertragen. Eine Sammlung von Anwendungen zeigt den vielfältigen Nutzen der Ansätze im praktischen Einsatz.

Nachdem der Informationsfluss auf wesentliche Teile reduziert wurde, können mit Hilfe der informativen Daten die eigentlichen Planungs-, Entscheidungs- und Problemlösungsinstanzen ihre Arbeit aufnehmen. Wie dies im Rahmen der hier vorgestellten Architektur geschieht, wird im nächsten Kapitel diskutiert.

4. Reinforcement Learning

Im Zentrum eines intelligenten Systems steht immer eine Instanz, der das Fällen von Entscheidungen obliegt. Diese Entscheidung kann dabei beispielsweise reaktiv basierend auf den gemachten Beobachtungen abgeleitet werden oder teil eines Plans sein. Es existiert eine Vielzahl von Paradigmen, die geeignet sind, solche Entscheidungen zu treffen. Für den hier betrachteten Ansatz einer datengetriebenen, lernenden Architektur engt sich das Spektrum der Möglichkeit zwar bereits ein, aber dennoch ist es nicht möglich, alle Varianten umfassend zu betrachten. Daher wird sich der weitere Verlauf dieses Kapitels auf eine Variante der Entscheidungsfindung beschränkten: das Reinforcement Learning. Dazu wird ein Abriss des Grundprinzips und aktueller Entwicklungen gegeben, bevor drei Vertreter näher vorgestellt und untersucht werden. Diese werden untereinander verglichen und gewertet, und im Anwendungsszenario anderen Paradigmen gegenüber gestellt.

Grundlagen des Reinforcement Learnings

Das Standardwerk im Bereich des Reinforcement Learning (RL) [Sutton und Barto, 1998] definiert:

Definition 4.1

Reinforcement Learning

Reinforcement Learning beschäftigt sich damit eine Entscheidungsstrategie (*Policy*) zu lernen, welche Aktionen ein Agent in einem be-

stimmten Zustand auszuführen hat, um eine akkumulierte numerische Belohnung, das sogenannte *Reinforcement*, zu maximieren.

Um dieses Ziel zu erreichen, interagiert der Agent (die Planungs- und Entscheidungsinstanz im Sinne der kognitiven Architektur) mit seiner Umgebung. Er nimmt den aktuellen Zustand wahr und wählt aus einer Menge von Aktionen eine aus, die er durchführt. Nach der Aktionsausführung erhält der Agent eine Belohnung oder Bestrafung in Form eines Reinforcement-Signals, welches auch als Reward bezeichnet wird. Das Ziel des Agenten besteht darin, die Summe über alle Rewards zu maximieren. Dazu benötigt der Agent Wissen darüber, welcher Zustand mit seinen Aktionsfolgen die maximale Belohnung verspricht. Da dieses Wissen apriori meist nicht zur Verfügung steht, muss der Agent durch Versuch und Irrtum diese Zusammenhänge selbst erlernen. Dieser Erwerb von neuem Wissen wird als Exploration bezeichnet, während das Durchführen von bekannten Aktionen zur Maximierung der Belohnung als Exploitation bekannt ist.

Typischerweise wird das Reinforcement Learning Problem als Markov-Entscheidungsprozess (Markov Decision Process, MDP) aufgefasst. Dazu muss die Markov-Eigenschaft gewährleistet sein, welche besagt, dass der neue Zustand s_{t+1} nur vom aktuellen Zustand s_t und der darin ausgeführten Aktion a_t abhängt.

Definition 4.2

Bestandteile eines Reinforcement Learning Systems

Die Umgebung in der der Agent operiert, sei definiert durch eine Menge von Zustände S und einer Menge von durchführbaren Aktionen A. Dann ergibt sich seine Handlungsvorschrift, die sogenannte Policy Π, als Abbildung des Zustandes auf eine Aktion $\Pi : S \rightarrow A$. Weiterhin notwendig ist das Rewardsignal R welches in jedem Zustand vergeben wird.

Das formale MDP eines Reinforcement Problems ist als 4-Tupel definiert:

$$MDP = (\mathcal{S}, \mathcal{A}, \mathcal{P}^{a_t}_{s_t, s_{t+1}}, \mathcal{R}_{s_t}) \tag{4.1}$$

- $\mathcal{S}$ ist die Menge aller möglichen Zustände.
- $\mathcal{A}$ ist die Menge aller möglichen Aktionen. Bei bestimmten Problemen kann die Menge verfügbarer Aktionen vom Zustand s_t abhängig sein.
- $\mathcal{P}^{a_t}_{s_t, s_{t+1}}$ ist die Transitionswahrscheinlichkeit, mit der man unter Ausführung von Aktion a_t in Zustand s_t im Zustand s_{t+1} landet.
- $\mathcal{R}_{s_t}$ ist der Reward, den der Agent in Zustand s_t erhält. Hier ist auch denkbar, dass der Reward nicht nur vom Zustand, sondern auch von der gewählten Aktion a_t abhängig ist.

In Abbildung 4.1 werden die benannten Elemente in Relation zueinander gezeigt.

Das Ziel des Systems ist dabei, die Summe aller zukünftigen Rewards zu maximieren

$$R = r_{t+1} + r_{t+2} + r_{t+3} + \dots. \tag{4.2}$$

Dieses Optimierungskriterium sorgt für eine implizite Planung bei Reinforcement Learning Verfahren. Anstatt nur gierig die nächste beste Aktion auszuführen, ermöglicht diese Formulierung, dass eine momentan schlechte Aktion ausgeführt wird, die langfristig jedoch zu einem höheren Gesamtreward führt.

Oftmals unterscheidet man zwischen episodischen und fortlaufenden Problemen. Episodische Probleme haben dabei einen wohldefinierten Endpunkt, z.B. das Erreichen einer bestimmten Zelle in einer Gridwelt oder das Ende eines Spiel. Fortlaufende Probleme arbeiten hingegen

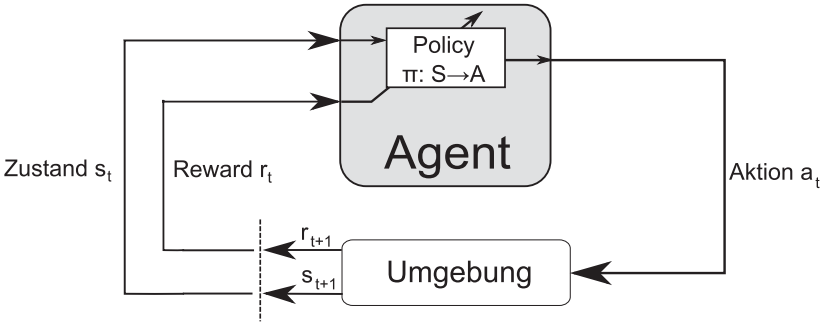


Abbildung 4.1.: Übersicht der Bestandteile eines Reinforcement Learning Systems. Der Agent beobachtet seinen aktuellen Zustand s_t und den erhaltenen Reward r_t. Mit Hilfe der Policy (Strategie) Π wird für den Zustand eine Aktion a_t ausgewählt und ausgeführt, was zu einem neuen Zustand s_{t+1} und Reward r_{t+1} führt. Mittels der Rewardinformation wird während des Lernens die Policy angepasst. Die Abbildung ist an [Sutton und Barto, 1998] angelehnt.

auf unbestimmt lange Zeit und finden sich in vielen Regelungsanwendungen. Das Problem an diesem potentiell unendlichen Zeithorizont ist, dass die in Gleichung 4.2 benannte Summe unendlich groß werden könnte und damit die Optimierung erschwert oder unmöglich gemacht wird. Praktisch umgangen wird dies durch die Einführung eines Diskontierungsfaktors $\gamma \in [0, 1)$, der ferner in der Zukunft liegenden Rewards eine geringere Bedeutung zuweist.

$$R = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \ldots = \sum_{t=0}^{\infty} \gamma^t r_{t+1}$$

Dieses Konzept der Abwertung zukünftiger Einflüsse findet sich sehr häufig in ökonomischen Konzepten wieder, im Bereich des Maschinellen Lernens ist dieses Vorgehen eher ungewöhnlich. Allerdings ist so eine geschlossene Darstellung des Gesamtrewards als geometrische Reihe möglich und damit die Anwendung der verschiedenen Lösungskonzepte.

Reinforcement Learning Verfahren versuchen für das MDP eine rewardmaximierende Policy zu finden. Einteilen lassen sich die Ansätze in zwei großen Klassen - die Policy Iteration Algorithmen und die Policy Search Algorithmen. Die erste Gruppe ist jene, die das „klassische“ Reinforcement Learning umfasst. So beschäftigt sich beispielsweise Sutton und Bartos Reinforcement Learning Standardwerk [Sutton und Barto, 1998] fast ausschließlich mit der Policy Iteration.

Beide Paradigmen lassen sich einfach voneinander unterscheiden. Die Policy Search Ansätze suchen direkt nach einer geeigneten Handlungsvorschrift. Hinter dem Begriff der Policy Search verbergen sich oftmals aus der Mathematik stammende Optimierungsverfahren, welche im Parameterraum der Policy nach der besten Strategie suchen.

Im Gegensatz dazu gehen Policy Iteration Ansätze den Weg über eine Approximation der (Action-)Value-Funktion. Die Value-Funktion V (oder Q-Funktion für Aktions-Zustands-Paare) entspricht dabei dem zu erwartenden zukünftigen Gesamtreward für einen Zustand. Die Policy Iteration besteht aus zwei Teilen, welche alternierend wiederholt werden. Zum einen ist dies der Schritt der *Policy Evaluation* (Strategiebewertung), welche versucht, eine Bewertung einer gegebenen Policy in Form der erwähnten Value-Funktion zu ermitteln. Zum anderen existiert der Schritt des *Policy Improvements* (Strategieverbesserung), welches auf Basis einer gegebenen Bewertungsfunktion die Policy verbessert. Hierbei gibt es in vielen Verfahren keine explizite Repräsentation der Policy in Form einer direkten Abbildung von Zuständen auf Aktionen. Vielmehr wird für jeden Zustand anhand der Value-Funktion auf die Policy geschlossen (z.B. in dem die Aktion ausgeführt wird, die zum Zustand mit dem höchsten Value führt).

Eine Kombination beider Ansätze existiert ebenfalls, es handelt sich dabei um sogenannte Actor-Critic Methoden. Diese kombinieren die Strategiebewertung in Form eines Kritikers mit einem Aktor, einer di-

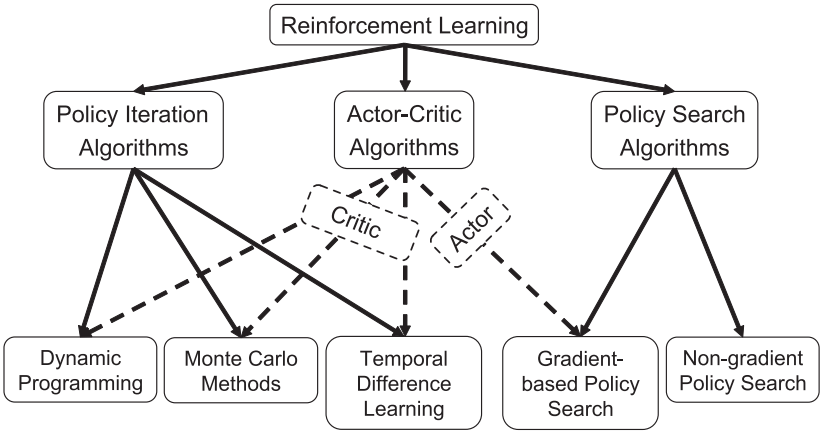


Abbildung 4.2.: Systematisierung der unterschiedlichen Reinforcement Learning Spielarten. Im linken Zweig befinden sich jene Verfahren, die zum Finden der optimalen Policy das Konstrukt einer (Action-)Value Function verwenden, die explizit die zu erwartende zukünftige Belohnung kodiert. Dazu zählen das klassische Dynamic Programming, wie es ursprünglich von Bellmann genutzt wurde, stochastische Monte-Carlo-Methoden und die weitverbeiteten Temporal Difference Methoden. Der rechte Zweig fasst das Problem hingegen als direktes Optimierungsproblem im Raum aller möglichen Policies auf, wobei hier zwischen lokaler Optimierung, die auf Gradientenverfahren basieren, und globaler Optimierung unterschieden wird. Einen Mittelweg dazwischen beschreiten die sogenannten Actor-Critic Methoden, da sie Methoden aus den beiden anderen Zweigen verwenden.

rekten Policyrepräsentation, welche auf Basis des Kritikers lernt.

Grafisch ist diese Einteilung auch in Abbildung 4.2 dargestellt.

Für die Verwendung in der angestrebten kognitiven Architektur wurden drei unterschiedliche Ansätze ausgewählt und untersucht. Sie sollen in den nachfolgenden Abschnitten kurz vorgestellt werden. Aus dem Bereich der Policy Iteration werden das Neural Fitted Q-Iteration (NFQ) Verfahren [Riedmiller, 2005] [Hafner, 2009] und das Reinforcement

Learning mit Gauß'schen Prozessen (RLGP) [Kuss, 2006] betrachtet, aus dem Bereich der Policy Search Algorithmen wird das Cooperative Synapse Neuroevolution (CoSYNE) Verfahren [Gomez et al., 2008] untersucht. Dabei existieren für das NFQ Verfahren auch Erweiterungen [Hafner, 2009], die es in den Bereich der Actor-Critic Verfahren überführen und dort zu einem sehr nahen Verwandten des Action Dependent Heuristic Dynamic Programming (ADHDP) [Si et al., 2004] macht, so das auch diese Gruppe Beachtung findet.

4.1. Neural Fitted Q-Iteration

Neural Fitted Q-Iteration ist ein Value-Iteration Verfahren, welches in [Riedmiller, 2005] vorgestellt wurde. Die grundlegende Idee dabei ist, mittels eines neuronalen Netzes die Zustandsaktionsfunktion (Q-Funktion) bei einer geringen Zahl an Beobachtungen zu approximieren.

Definition 4.3

Q-Funktion

Die Q-Funktion gibt an, wie hoch der erwartete zukünftige Gesamtreward ist, wenn in Zustand s_t die Aktion a_t ausgeführt wird. Der zukünftige Gesamtreward wird dabei als Erwartungswert der diskontierten Summe repräsentiert. Der Diskontierungsfaktor $0 \leq \gamma < 1$ wichtet fern in der Zukunft liegende Belohnungen weniger stark als zeitliche nähere liegende Rewards r.

$$Q^\pi(s,a) = \mathbb{E}\left(\gamma^0 r(s_t,a_t) + \gamma^1 r(s_{t+1},a_{t+1}) + \gamma^2 r(s_{t+2},a_{t+2}) + \ldots\right)$$
$$Q^\pi(s,a) = \mathbb{E}\left(r(s_t,a_t) + \gamma Q^\pi(s_{t+1},a_{t+1})\right)$$

Die beste Aktion kann somit ausgewählt werden, in dem die Aktion gesucht wird, welchen den maximalen Q-Wert hat.

Dabei wird in der Basisvariante davon ausgegangen, dass der Zustandsraum kontinuierlich ist und die Aktionen diskret repräsentiert werden.

Es handelt sich beim NFQ Ansatz um ein sogenanntes modellfreies oder direktes Verfahren, da die Transitionswahrscheinlichkeiten $\mathcal{P}^{a_t}_{s_t,s_{t+1}}$ nicht gelernt werden.

Das Verfahren alterniert dabei zwischen zwei Modi. Einerseits gibt es einen Interaktionsmodus, in welchem der Agent seine Umgebung beobachtet, manipuliert und die Auswirkungen protokolliert. Andererseits gibt es eine Lernphase, in der der Agent mittels der protokollierten Beobachtungen sein Wissen, also die durch ein neuronale Netz approximierte Q-Funktion, aktualisiert.

Als neuronales Netz kommt ein klassisches Multi-Layer Perceptron zum Einsatz, welches mit R-Prop [Riedmiller und Braun, 1993] trainiert wird. Versuche mit dem klassischen Backpropagation-Algorithmus und der Levenberg-Marquardt Variante [Zell, 1994] zeigten, dass die Verwendung des einfachen Backpropagation-Algorithmus aufgrund schlechterer Konvergenzeigenschaften ungünstig ist und nur bei der Verwendung von R-Prop oder dem Levenberg-Marquardt Algorithmus zuverlässig zufriedenstellende Ergebnisse erreicht wurden.

Die Erfahrungen, die während der Interaktionsphase gemacht werden, sind als Datentupel $D = (s, a, s', r)$ gespeichert. Dies entspricht dem aktuellen Zustand s, der ausgeführten Aktion a, dem erreichten Folgezustand s' und dem erzielten Reward r. Dabei wird während dieser Interaktion *on policy* agiert, also in jedem Zustand die bestmögliche bisher bekannte Aktion ausgewählt. Das Wissen, welches die bestmögliche Aktion ist, ist im neuronale Netz gespeichert. Die Aktionsauswahl erfolgt dadurch, dass dem Netz der aktuelle Zustand sowie alle mög-

lichen Aktionen als Eingaben präsentiert werden. Dabei wird für jede Aktion der zu erwartende Reward mit Hilfe des Netzes geschätzt. Jene Aktion, die den maximalen Q-Wert am Ausgang des Netzes erzeugt, wird zur Ausführung ausgewählt. Allerdings können an dieser Stelle, je nach gewählter Explorationsstrategie, auch andere Aktionen bestimmt werden, beispielsweise nach der ϵ-greedy Strategie[1].

Beim Wechsel in die Lernphase muss basierend auf den gespeicherten Datentupeln zuerst die zu lernende Q-Funktion ermittelt werden. Dazu wird mittels des neuronalen Netzes für jedes Datentupel der zu erwartende Gesamtreward t_i bestimmt.

$$t_i = r_i + \gamma \max_a \hat{Q}(s'_i, a)$$

Mit der Information $\mathcal{T} = (t_i, s_i, a_i)$ kann nun das Netz trainiert werden, wobei Zustand und Aktion am Eingang angelegt werden (s, a) und der geschätzte Gesamtreward t am Ausgang ausgegeben werden soll. Als Ergebnis erhält man eine Approximation der Q-Funktion durch das Netz, während die gesammelte Datenbasis ein implizites Modell für die Zustandsübergänge darstellt. Die Schätzung von $\hat{Q}(s'_i, a)$ wird dabei vom neuronalen Netz geliefert und stellt somit die Q-Funktion vor der Aktualisierung dar.

Der gesamte NFQ-Algorithmus ergibt sich nun aus dem zyklischen Wechsel der Interaktions- und der Lernphase.

NFQ für kontinuierliche Aktionsräume

Soll der vorgestellte Apparat auch auf kontinuierliche Aktionsräume ausgeweitet werden, so entfällt die Möglichkeit des Durchprobierens der Aktionen. Man könnte natürlich ein Gitter auf dem Aktionsraum

[1] Für eine Diskussion von Explorationsstrategien sei auf Abschnitt 5.2 verwiesen.

definieren, an dessen Stützstellen Aktionen vom Netz bewertet werden, aber das entspricht einer Diskretisierung des Aktionsraums. Die Alternative hierzu ist, die Information nach der besten Aktion dem Netz selbst zu entnehmen, indem der Q-Wert am Ausgang durch das Netz nach der Aktion abgeleitet wird, formell also die partielle Ableitung von $Q(s, a)$ nach a: $\frac{\partial Q(s,a)}{\partial a}$. Realisiert wird dies durch das mathematische Gerüst des Backpropagation-Algorithmus oder seiner Verwandten. Damit lässt sich nun ein Gradientenaufstieg zur besten Aktionen durchführen. Vorgestellt und ausführlich diskutiert wird diese als Generalized NFQ bezeichnete Erweiterung in [Hafner, 2009].

Das Problem dieses Ansatzes ist, dass der Gradientenaufstieg natürlich nur das lokale Maximum finden kann. Es ist notwendig, mehrere Optimierungsläufe von unterschiedlichen Startpunkten zu initialisieren und das beste Ergebnis zu verwenden. Allerdings ist gerade das Zurückpropagieren ein durchaus zeitkritischer Vorgang, der bei Echtzeitanwendungen problematisch werden kann.

Um dies zu umgehen, wird in [Hafner, 2009] *Neural Fitted Q-Iteration with Continuous Actions* (NFQCA) vorgestellt. Dabei handelt es sich um eine Aktor-Kritik Architektur, in der das bisherige Netz zur Approximation der Q-Funktion bestehen bleibt (und als Kritiknetz bezeichnet wird) während die beste Aktion nicht mehr durch Probieren oder die Gradientensuche bestimmt wird, sondern in einem eigenen Netz, dem sogenannten Aktor- oder Strategienetz, gespeichert ist.

Mit diesem zusätzlichen Aktornetz vereinfacht sich die Suche nach der besten Aktion in der Interaktionsphase zu einer einmaligen Anfrage an das Netz, welches als Eingabe den Zustand s erhält und die Aktion a ausgibt. Man erreicht also eine direkte Sensor-Aktor-Kopplung. Um dieses Netz zu trainieren, wird die Information aus dem Kritiknetz genutzt. Das heißt in der Lernphase wird wie bisher das Kritiknetz mittels der generierten Datentupel bestimmt. Sobald dieser Prozess abgeschlossen ist, beginnt die Phase zum Trainieren des Aktornetzes.

Dazu wird im Kritiknetz die partielle Ableitung nach der Aktion berechnet und zwar nicht nur für einen Zustand, sondern für alle Zustände aus den Trainingsdaten $\mathcal{T}$. Diese Information wird dann genutzt, um das Aktornetz zu adaptieren, welches die Aktionsinformation fest speichert, anstatt sie in jedem Zustand neu zu suchen, wie es bei beim oben beschriebenen Generalized NFQ der Fall ist. Dazu wird die partielle Ableitung nun mit der Ausgabe des Aktors für den betreffenden Zustand multipliziert. Mittels der Kettenregel lässt sich der Zusammenhang für einen Zustand s also wie folgt beschreiben:

$$\frac{\partial Q(s,a)}{\partial w_{Aktor}} = \frac{\partial Q(s,a)}{\partial a} \cdot \frac{\partial a}{\partial w_{Aktor}}.$$

Hafner verwendet zum Training des Aktors auch wieder den RProp-Algorithmus, wobei theoretisch auch jeder andere Trainingsalgorithmus eigesetzt werden könnte.

Grafisch ist ein solches Aktor-Kritik System in Abbildung 4.3 skizziert.

Der resultierende Algorithmus ähnelt dabei sehr stark dem *Action-dependent Heuristic Dynamic Programming* (ADHDP) (manchmal auch als *Neural Dynamic Programming* bezeichnet) [Si et al., 2004], welches exakt dieselbe Struktur und annähernd die gleiche Kostenfunktion zum Training des Aktors verwendet. Weitere Details, gerade zur Verwandtschaft von NFQ und ADHDP, finden sich in der Diplomarbeit von Christian Vollmer [Vollmer, 2009].

In der Dissertation von Hafner [Hafner, 2009] werden verschiedene regelungstechnische Anwendungen präsentiert, die mittels NFQ gelernt und geregelt wurden. Diese unterscheiden sich von herkömmlichen Reinforcement Learning Benchmarks zum Teil deutlich, da die klassischen Benchmarks oftmals sehr allgemein gehaltenen sind und nur selten die Anforderungen realer regelungstechnischer Probleme widerspiegeln. Spezielles Augenmerk wurde dabei auf die Behandlung externer Führungsgrößen gelegt, da diese typischerweise kaum betrachtet werden.

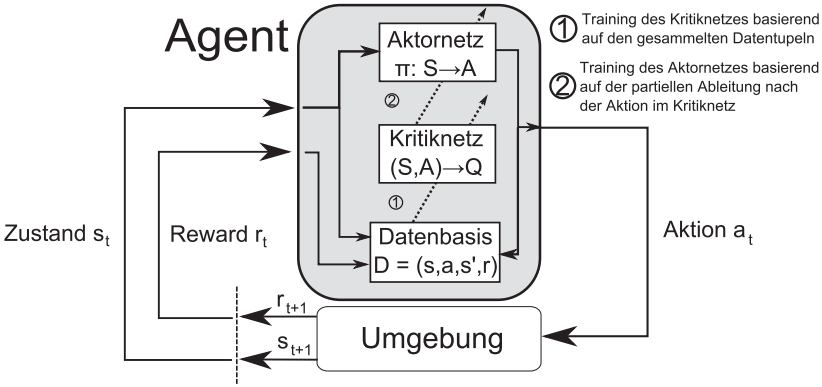


Abbildung 4.3.: Übersicht der Bestandteile eines Reinforcement Learning Systems basierend auf der Aktor-Kritik Formulierung des NFQ Algorithmus. Durchgezogene Linien vermitteln den Informationsfluss während eines Zyklus in der Interaktionsphase. Gestrichelt angedeutet sind die Zusammenhänge während der Lernphase. Für den Fall des klassischen NFQs fällt das Aktornetz im Agenten weg. Stattdessen werden mögliche Aktionen zusammen mit dem aktuellen Zustand dem Kritiknetz präsentiert. Die Aktion mit der höchsten Ausgabe wird dann als Aktion ausgewählt.

Diese beinhalten die Regelung von Gleichstrommotoren für RoboCup-Roboter (siehe auch [Riedmiller et al., 2009]) sowie ein Vielzahl von Simulation für die Regelung von Heizspiralen, Autopiloten oder aktiven Schwingungsdämpfern. Dabei werden auch die Auswirkungen von Rauschen und nichtdirekt beobachtbare Änderungen im Problem betrachtet.

In [Riedmiller et al., 2007] wird die Verwendung von NFQ für ein autonomes Auto beschrieben, aber auch zur Regelung von Müllverbrennungsanlagen fand das Verfahren Verwendung [Steege et al., 2010]. Auch in vielen anderen Publikationen, die mit klassischen Benchmarks, wie dem Stabbalance oder dem MountainCar Problem, arbeiten, findet sich mittlerweile recht häufig das NFQ-Verfahren als Referenzverfahren.

Daher sollten Untersuchungen im Bereich des Reinforcement Learnings auch dieses de facto Standardlernverfahren mit berücksichtigen.

4.2. Gauß'sche Prozesse für RL

Im Bereich des maschinellen Lernens haben in den letzten Jahren Gauß'sche Prozesse (GP) an Popularität gewonnen [Rasmussen und Williams, 2005]. Sie dienen dabei nicht nur als einfacher Funktionsapproximator, sondern geben zusätzlich auch eine Konfidenz über die Sicherheit der Schätzung mit an. Dies wird auch im Bereich des Reinforcement Learnings genutzt [Kuss, 2006], [Engel et al., 2003], [Rasmussen und Kuss, 2004], [Deisenroth et al., 2008]. Die Verfahren fallen dabei in die Gruppe des sogenannten Bayesian Reinforcement Learnings, wobei es darum geht Konzepte aus der probabilistischen Modellierung für das Reinforcement Learning zu verwenden.

Im Kern sind Gauß'sche Prozesse Funktionsapproximatoren, die im Rahmen des Reinforcement Learnings eingesetzt werden, um beispielsweise das Prozessmodell oder die Q-Funktion zu approximieren. Die GPs könnten theoretisch durch jeden beliebigen Funktionsapproximator ersetzt werden. Der theoretische Vorteil der Verwendung von GPs gegenüber anderen Approximatoren liegt dabei in der expliziten stochastischen Beschreibung des Approximators und in der Fähigkeit implizit eine Konfidenzaussage über die geschätzten Werte abzuleiten.

Die mathematischen Grundlagen und Hintergründe zu den Gauß'schen Prozessen werden im Anhang A.2 beschrieben. Die Verwendungsmöglichkeiten dieses Approximators als Prozessmodells oder als Repräsentation der Q-Funktion soll kurz diskutiert werden.

Gauß'sche Prozesse als Prozessmodell

Falls ein Modell des zu regelnden Systems zur Verfügung steht, können viele modellbasierte Verfahren problemlos eine geeignete Policy finden. Diese nutzten die im Modell gespeicherten Information um Aktionsfolgen zu simulieren und können somit eine optimale Policy finden. In einer realen Anwendung ist es oftmals nicht möglich direkt über längere Zeit mit dem Prozess zu interagieren, da ein solches Vorgehen mit hohen monetären oder zeitlichen Kosten behaftet ist oder sicherheitskritisch sein könnte und man damit auf Modelle angewiesen ist.

Ein solches Modell aus den Daten zu lernen ist Thema im Bereich der Modellidentifikation. Gauß'sche Prozesse sind dabei eine Möglichkeit dies zu tun. Das heißt, es wird die Transitionsfunktion $\mathcal{P}^{a_t}_{s_t,s_{t+1}}$ (siehe Definition 4.2) mittels einem oder mehreren Gauß'schen Prozessen approximiert. Dabei wird ein Gauß'scher Prozess pro Dimension des Zustandsraums benötigt.

Diese Anwendung Gauß'scher Prozesse erfolgt direkt auf den gemachten Beobachtungen und wird mit den Standardmethoden wie sie in [Rasmussen und Williams, 2005], [Deisenroth, 2009] und [Kuss, 2006] beschrieben werden, realisiert. Dazu wird aus den i Beobachtungen der funktionelle Zusammenhang $s_{t+1} = f(s_t, a_t)$ genutzt, um für unbekannte Zustands-Aktionspaare $(\tilde{s_t}, \tilde{a_t})$ den Folgezustand $\tilde{s}_{t+1}$ zu approximieren. Eingesetzt in das mathematische Gerüst aus Anhang A.2 ergibt sich:

$$E(\tilde{s}_{t+1}|X, Y, [\tilde{s_t}, \tilde{a_t}]) = K([\tilde{s_t}, \tilde{a_t}], X)K(X, X)^{-1}Y^T.$$

Dabei ist X die Matrix in der alle beobachteten Zustands-Aktionspaare $X = \left[(s^1_t, a^1_t)^T, (s^2_t, a^2_t)^T, \ldots, (s^i_t, a^i_t)^T\right]$ stehen und $Y = \left[s^1_{t+1}, s^2_{t+1}, \ldots, s^i_{t+1}\right]$ der Vektor mit den zugehörigen Folgezuständen ist. K ist die verwendete Kovarianzfunktion und der Erwartungswert $E(\tilde{s}_{t+1}|X, Y, [\tilde{s_t}, \tilde{a_t}])$ ist die gesuchte Approximation des Folgezustands.

In der Literatur wurde dieser Ansatz neben den klassischen Szenarien, wie beim MountainCar oder Stabbalanceproblem [Kuss, 2006], beispielsweise zur Modellierung eines Zeppelins genutzt [Ko et al., 2007]. Abseits des Reinforcement Learnings kommen Gauß'sche Prozesse in verwandten Ansätzen der Systemidentifikation zum Einsatz: zum Beispiel zur Modellierung inverser Kinematik bei Roboterarmen [Nguyen-Tuong et al., 2008] oder auch in der Feuerungsführung im Kontext modellprädiktiver Regelungen [Grancharova et al., 2008]. In [Jung und Stone, 2010] wird darauf hingewiesen, dass, aufgrund des Fluch der Dimensionalität, diese Verfahren nur in einem hinreichend niedrigdimensionalen Zustandsraum funktionieren.

Gauß'sche Prozesse als Value-Approximatoren

Die zweite Möglichkeit Gauß'sche Prozesse im Rahmen des Reinforcement Learnings einzusetzen, besteht darin, mittels des GPs die (Aktions-)Wertefunktion (z.B. Q-Funktion) zu approximieren. Dies entspricht dem Zweck des Multi-Layer Perceptrons beim NFQ-Verfahren.

Dazu ist es notwendig, sogenannte Supportpunkte [Kuss, 2006] zu definieren, an denen die Q-Werte bekannt sind[2]. Alle anderen Punkte im kontinuierlichen Zustands-Aktions-Raum werden per Interpolation mit dem GP geschätzt. Diese Supportpunkte im Zustandsraum entsprechen der Matrix X (wie weiter oben), die Q-Werte an den diesen Supportpunkten dem Vektor Y.

$$E(\tilde{Q}_{t+1}|X, Y, [\tilde{s_t}, \tilde{a_t}]) = K([\tilde{s_t}, \tilde{a_t}], X)K(X, X)^{-1}Y^T.$$

Standardmäßig erfolgt die Wahl der Supportpunkte in [Kuss, 2006] möglichst in einer Gitterstruktur über dem Zustands-Aktions-Raum.

[2]Berechnet werden diese mit den klassischen Formeln für das Q-Learning während der Agent mit seiner Umwelt interagiert. Siehe Definition 4.3 und [Sutton und Barto, 1998]

Die Entscheidung welche der Beobachtungen als Supportpunkte verwendet werden, ist dabei von großer Wichtigkeit, da jeder zusätzliche Supportpunkt den Rechenaufwand deutlich erhöht und die Zahl der notwendigen Supportpunkte zur Approximation der Q-Funktion exponentiell mit der Dimensionalität des Zustand-Aktions-Raums wachsen müsste. Man stößt hier bei höherdimensionalen Problemen schnell an die Grenzen der praktisch realisierbaren Berechenbarkeit.

Ebenfalls problematisch ist das Finden der optimalen Policy unter einer gegeben Value Funktion (*policy improvement*). Wie auch beim NFQ für kontinuierliche Aktionsräume ist bei der Verwendung von Gauß'schen Prozessen das Finden der besten Aktion ein nichtkonvexes Optimierungsproblem, welches beispielsweise mit einem Gradientenverfahren gelöst wird. Es besteht daher auch hier die Gefahr in einem lokalen Optimum hängen zu bleiben.

Neben der reinen Schätzung der (Action-)Value-Funktion kann die Konfidenzaussage, also die Varianz über $\tilde{Q}_{t+1}$, genutzt werden. Die Berechnungsvorschrift findet sich in Anhang A.2 (Gleichung A.21). So lassen sich intuitiv Explorationsstrategien formulieren, welche darauf abzielen, die Unsicherheit über die Schätzung der Q-Funktion zur verringern. Siehe dazu z.B. [Jung und Stone, 2010].

Die Verwendung von Gauß'schen Prozessen zur Value-Approximation wird in der Dissertation von Kuss jedoch sehr kritisch gesehen:

„In general it must be questioned whether a Gaussian process [...]is well suited for representing the value function [...]" - [Kuss, 2006], Seite 155

Als Gründe werden angeführt, dass die Value Funktion oftmals instationär ist, während der Gauß'sche Prozess nur stationäre Funktionen approximieren kann, und die Menge der benötigten Datenpunkte sehr groß ist, um eine sinnvolle Approximation zu erhalten. Gerade in hochdimensionalen Zustands-Aktions-Räumen wird dies zu einem schwer beherrschbaren Problem. Dieses zweite Problem zeigte sich auch in den

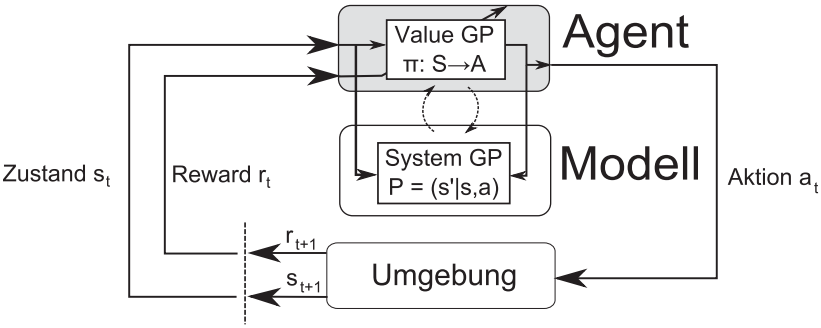


Abbildung 4.4.: Übersicht der Bestandteile eines Reinforcement Learning Systems mit denen beiden Einsatzmöglichkeiten Gauß'scher Prozesse. Einerseits kann damit die Value Funktion approximiert werden, als Bestandteil des RL Agenten selbst. Andererseits ergibt sich die Option Gauß'sche Prozesse als Systemmodell zu verwenden. Und schließlich können beide Varianten kombiniert werden.

durchgeführten Experimenten, welche in Abschnitt 4.4 vorgestellt und diskutiert werden.

In [Deisenroth, 2009] wird das kombinierte Framework beider GP Anwendungen als *Gaussian Process Dynamic Programming* (GPDP) vorgestellt und um eine explizite Onlinevariante (Active Learning GPDP) erweitert. Allerdings bleibt auch hier das Problem großer Zustandsräume ungelöst.

Zusammenfassend sind beide Optionen zur Verwendung von Gauß'schen Prozessen in Abbildung 4.4 dargestellt.

4.3. Cooperative Synapse Neuroevolution

Cooperative Synapse Neuroevolution (CoSYNE) wurde in [Gomez et al., 2006] und [Gomez et al., 2008] als Verfahren vor-

gestellt, das speziell bei komplexen Regelungsaufgaben Stärken aufweist.

Es ist in direkter Linie verwandt zu Neuroevolution of Augmenting Topologies (NEAT) [Stanley und Miikkulainen, 2002] und Symbiotic Adaptive Neuro-Evolution (SANE) [Moriarty und Miikkulainen, 1996] und stellt in diesem Stammbaum die modernste Form neuroevolutionären Reinforcement Learnings dar.

Die Grundidee ist hierbei die Policy durch ein rekurrentes neuronales Netz zu approximieren. Diesse Netz dient, anders als beim NFQ-Verfahren oder den Gauß'schen Prozessen, nicht zur Approximation einer Q-Funktion, sondern es handelt sich um ein Aktornetz, also den Regler selbst. Es findet eine direkte Abbildung des Zustands s_t auf die auszuführende Aktion a_t statt.

Die Verwendung eines rekurrenten Netzes soll hier ein praktisches Problem umgehen. Oftmals ist es in der Praxis so, dass der wahrgenommene Zustand nicht die Markov-Eigenschaft erfüllt. Man hat es also nicht mit einem MDP, wie in Definition 4.2 beschrieben, zu tun, sondern mit einem Partially Observable MDP (POMDP). Für diese Problemklasse ist die Konvergenz der meisten Reinforcement Learning Verfahren nicht gesichert und die Mehrdeutigkeiten können das erzielte Ergebnis beeinträchtigen.

Daher wird versucht, dieses Problem unter Hinzunahme zeitlicher Kontextinformationen zu umgehen. Dies kann explizit durch einen Zustandsschätzer geschehen, der die aktuellen Beobachtungen mit Hilfe von älteren Informationen in einen Zustand umwandelt. Dies wäre im Wahrnehmungs-Handlungs-Zyklus Bestandteil der Situationseinschätzung. Einen zweiten Weg stellen rekurrenten Netze dar, welche implizit den zeitlichen Kontext durch Rückkopplungen beachten und zur Entscheidungsfindung nutzen. Dieser Weg wird beim CoSYNE Ansatz beschritten, als Regler kommen vollständig rekurrente Neuronale Netze

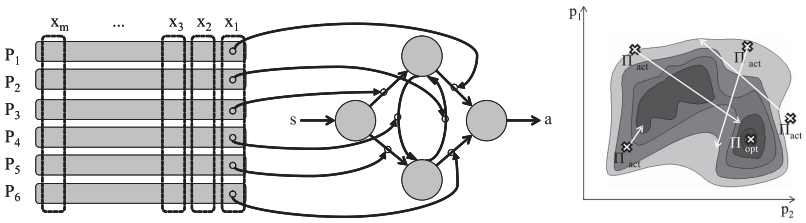


Abbildung 4.5.: **(Links)**Rekurrentes neuronales Netz, welches als Aktornetz fungiert, und die Kodierung als Individuum im Rahmen der evolutionären Optimierung. Eine Spalte entspricht einem kompletten Netz, eine Zeile der Subpopulation aller verfügbaren Werte für ein spezielles Gewicht des Netzes. Abbildung nach [Gomez et al., 2008]. **(Rechts)** Abstrakter Suchraum aufgespannt über die beiden Parametern p (z.B. Gewichte im neuronalen Netz). Die Höhenlinien repräsentieren die Güte der Qualität. Die momentanen Policies Π_{act} werden evolutionären Operationen unterzogen, die zur Bewegung im Raum führen und mit der Zeit zur optimalen Policy Π_{opt} konvergieren. Abbildung angelehnt an [Hellwig, 2009].

zum Einsatz. Die Struktur des Netzes und seine Kodierung muss vor dem Lernprozess ausgewählt werden. Dargestellt ist dies im linken Teil von Abbildung 4.5.

Der Lernprozess unterscheidet sich von den bisher besprochenen Verfahren und läuft wie folgt ab:

1. Erzeugen einer initialen Menge (Population) von Netzen
2. Bewerten der aktuell vorhandenen Netze
3. Erzeugen einer neuen Generation von verbesserten Netzen durch Anwendung evolutionärer Operatoren auf die aktuell vorhandenen Netze
4. Falls das Abbruchkriterium nicht erfüllt ist, weiter mit Schritt 2.

In Schritt 1 werden entweder zufällige Strategien oder mit Vorwissen kodierte Netze verwendet um eine Anzahl von Handlungsstrategien zu

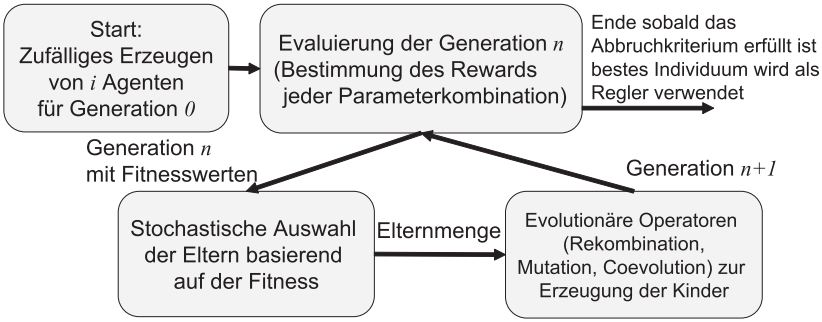


Abbildung 4.6.: Prinzipieller Ablauf der evolutionären Suche des CoSYNE-Algorithmus.

erhalten. In diesen Teilnetzen sollten möglichst verschiedene Teilstrategien enthalten sein, um einen möglichst großen Suchraum abzudecken.

Der problematische Teil eines solchen Neuroevolutionsverfahrens ist, dass immer eine ganze Population von rekurrenten Netzen, bewertet werden muss, um die Güte der Policy zu bestimmen (Schritt 2). Dies ist in realen Anwendungen typischerweise nicht möglich, da der Zeitaufwand immens ist. Daher muss die Bewertung, die Bestimmung der Fitness, auf anderem Weg erfolgen. Typischerweise kommen dazu Modelle zum Einsatz.

Eine Beschreibung der beim CoSYNE zur Optimierung vewendeten evolutionären Operatoren wird in Anhang A.3 gegeben.

Als Abbruchkriterium sind verschiedene Optionen realisierbar. Das reicht von einer festen Anzahl von Iterationsschritten, über einen Mindestwert bei der Bewertung, den die beste Policy übertreffen muss, bis hin zur Konvergenz des Lernverfahrens.

Eine visuelle Interpretation der Suche im Parameterraum ist im linken Teil von Abbildung 4.5 zu sehen, der Ablauf als solches ist in Abbildung 4.6.

Bei solchen Verfahren, die explizit ein Modell verwenden, besteht jedoch immer die Gefahr, dass der Regler das Modell erlernt und nicht das reale Problem. Normalerweise gibt es aber eine deutliche Diskrepanz zwischen Modell und realem Problem, was einen überangepassten Regler (Stichwort Overfitting) für den realen Einsatz untauglich macht. Um dieses Problem zu mildern, wird in dieser Arbeit auf eine Idee aus dem Bereich des Ensemble Learnings [Dietterich, 2000] zurückgegriffen. Dazu werden mehrere Modelle verwendet, um die Fitnessfunktion zu berechnen, statt auf ein Modell beschränkt zu bleiben. Die Bewertung eines Reglers erfolgt dann als Mittelwert über die Einzelbewertungen auf den Modellen.

Um die notwendige Diversität der Modelle zu erreichen, kann hier auf die üblichen Methoden zurückgegriffen werden. Beispielsweise sind das die Verwendung unterschiedlicher Modelltypen (einfache Multi-Layer Perceptrons, probabilistische Faktorgraphenbeschreibungen, vgl. Kapitel 6, oder auch die oben beschriebenen Gauß'schen Prozesse), Präsentation unterschiedlicher Muster während der Lernphase (z.B. durch Bagging) oder unterschiedlichen Initialisierungen bei der Modellidentifikation.

Nachteil an diesem Vorgehen ist natürlich der drastisch erhöhte Rechenaufwand, der durch die notwendige Erstellung zusätzlicher Modelle entsteht und die notwendigen mehrfachen Bewertungsläufe der Individuen auf den verschiedenen Modellen.

Zusammenfassend ist zu sagen, dass CoSYNE als Policy Search Verfahren einen Weg benötigt, Strategien/Policies zu bewerten. Dies kann entweder am Problem selbst geschehen, wenn sich dies schnell und kostengünstig realisieren lässt, oder muss an Hand eines oder mehrerer Modelle erfolgen. Vorteilhaft bei diesem Verfahren ist die Verwendung von rekurrenten Netzen, welche eine implizite Behandlung unbekannter, zeitlicher Zusammenhänge erlauben.

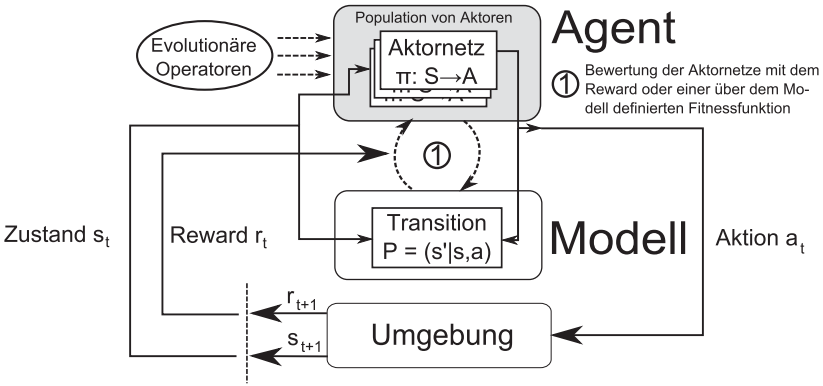


Abbildung 4.7.: Übersicht der Bestandteile eines Reinforcement Learning Systems basierend auf dem CoSYNE Algorithmus. Der Agent besteht in diesem Fall nur aus dem Aktornetz. Dieses wird wird mittels evolutionärer Suche aus einer Population von Policies bestimmt. Die Bewertung dieser Policies erfolgt dabei entweder am realer Prozess mit dem beobachteten Reward oder unter Verwendung eines (oder mehrere Modelle) und einer darüber definierten Rewardfunktion. Gute Individuen werden weiter entwickelt, schlechte Individuen werden aussortiert. Dieser Lernzyklus kann bei Verwendung eines Modells offline erfolgen.

4.4. Vergleichende Untersuchungen

Um die Eigenschaften der verschiedenen Ansätze miteinander vergleichen zu können, wurden die Verfahren Tests unterzogen. Dazu kamen zwei Testumgebungen zum Einsatz. Einerseits handelt es sich dabei um das wohlbekannte MountainCar Problem, welches auch schon von [Sutton und Barto, 1998] als Benchmark verwendet wurde. Andererseits wurde ein spezieller Simulator eingesetzt, der die Probleme und Eigenheiten, die im Kontext der Feuerungsführung (vgl. Kapitel 6) auftreten, berücksichtigt. Die Untersuchungen wurden dabei teilweise von Christian Barth in dessen Diplomarbeit [Barth, 2008] durchgeführt.

4.4.1. Mountain Car

Beim sogenannten Mountain Car Problem handelt es sich um ein klassisches Problem aus der Literatur des Reinforcement Learnings, siehe [Moore und Atkeson, 1995] und [Sutton und Barto, 1998]. Dabei soll ein Fahrzeug in einer zweidimensionalen Welt aus einem Tal heraus einen Hügel erklimmen. Der Anstieg ist allerdings so steil, dass die Beschleunigung des Fahrzeugs nicht ausreichend ist, um den Anstieg direkt zu überwinden. Daher scheitern klassische Ansätze, die die Regelabweichung gierig behandeln, an dieser Aufgabe. Stattdessen ist es notwendig, mit dem Fahrzeug auf der gegenüberliegenden Talseite Schwung zu holen und somit durch Aufschaukeln eine ausreichende Beschleunigung zu erreichen.

Für die mathematischen Details der Simulation und genaue Definitionen für den verwendeten Zustands-Aktionsraum und die Rewardfunktion, sei auf Anhang C verwiesen.

Es handelt sich dabei um ein episodisches Problem, der Versuch endet normalerweise, sobald der Agent sein Ziel erreicht hat. Das Ziel ist hierbei das Erreichen einer festgelegten Position an der der Agent stehen bleibt, also eine Geschwindigkeit von null hat. Zu beachten ist dabei, dass hier ein verzögerter Reward verwendet wird. Das bedeutet, dass der Agent nur eine Belohnung erhält, wenn er sein Ziel erreicht hat bzw. ihm sehr nahe gekommen ist. An anderen Orten und mit unpassenden Geschwindigkeiten erhält der Agent einen negativen Reward.

Dieses Szenario wurde in dieser Form untersucht, um einerseits bei einem einfachen, überschaubaren und bekannten Benchmark die Verfahren auf ihre Anfälligkeit gegenüber Rauschen zu vergleichen. Andererseits wurde die Problematik des verzögerten Rewards, welcher nur sehr nah an der eigentlichen Zielposition vergeben wurde, mit Hinblick auf die intelligente Feuerungsführung gewählt. Dort gibt es zwar dauerhaft einen Reward, aussagekräftig ist dieser allerdings auch nur in der Um-

gebung des Ziels. Zusätzlich verstärkt dieser Art der Rewardvergabe das Rauschproblem, da durch Rauschen hervorgerufene Abweichungen sich damit eher im Reward bemerkbar macht.

Experimente

In [Barth, 2008] wurden NFQ, GP und auch der Aktor-Kritik-Ansatz des Action-Dependent Heuristic Dynamic Programming [Si et al., 2004] untersucht. Jedoch zeigte sich dort, dass der Aktor-Kritik-Ansatz nicht zuverlässig eine brauchbare Lösung erzielen. Die Varianz in den Ergebnissen zwischen einzelnen Versuchen war sehr hoch, in einigen Fällen wurde keine sinnvolle Policy gelernt. Dadurch, dass selbst für das einfache MoutainCar-Problem die Suche nach einer stabilen Lösung so schwierig war, wurde dieses Verfahren verworfen.

Stattdessen wurde in dieser Arbeit das CoSYNE-Verfahren aufgegriffen und dem NFQ-Verfahren sowie dem Reinforcement Learning mit Gauß'schen Prozessen gegenübergestellt.

Die Untersuchungen zum Rauschen beinhalteten ein Verrauschen des Systemzustands als auch des vergebenen Rewards. Die Varianz des Rauschens war dabei auf 10% des jeweiligen Größe festgesetzt. Verglichen wurde dies mit einem geringerem Rauschen (Varianz von 3%) und ohne Rauschen (Varianz von 0%).

Die Ausgangsdaten für alle drei Verfahren waren dabei 1000 Zustands-Aktionsfolgen, die zum Lernen verwendet werden konnten. Der Versuchsaufbau für die drei Verfahren war dabei:

- NFQ: Es wurde ein Multi-Layer Perceptron mit einer Hiddenschicht mit fünf Neuronen als Approximator der Q-Funktion verwendet.
- Gauß'sche Prozesse: Aus den Trainingsdaten wurde ein Prozessmodell GP gelernt und damit dann ein Value GP trainiert. Dabei wurde

die Stärke des Rauschens jeweils auch für den Hyperparameter σ (siehe Anhang A.2) auf die wahre Größe gesetzt.

- CoSYNE: Es wurde als Aktor ein vollständig rekurrentes Netz mit 3 Hiddenneuronen verwendet (entspricht damit annähernd der Zahl freier Parameter beim NFQ-Verfahren).

Die Bewertung wurde über dabei über fünf Versuche gemittelt und ist in Abbildung 4.8 gezeigt. Jeder Versuch bestand dabei aus 100 Aktionen die der Agent nach Abschluss des Lernens durchgeführt hat.

Verhalten bei Rauschen

Die Ergebnisse zeigen für das NFQ-Verfahren und den CoSYNE-Ansatz einen klaren Zusammenhang zwischen der Stärke des Rauschens und des mittleren Rewards, der durch die Agenten erreicht wird. Die Unterschiede in der Qualität der Ergebnisse beider Verfahren ist dabei nicht signifikant. Allerdings ist der Berechnungsaufwand für das NFQ-Verfahren deutlich geringer, als für das Neuroevolutionsverfahren. Das Rauschen führt bei beiden Algorithmen dazu, dass der Wagen um das Ziel herum nicht wirklich stillgehalten wird, sondern immer in leichter Bewegung bleibt und damit auch geringere Rewards erhält.

Im Gegensatz dazu profitiert der Algorithmus mit den Gauß'schen Prozessen deutlich von einem leichten Rauschen. Interessanterweise generalisiert das Verfahren erst beim Vorhandensein von Rauschen sehr gut, ohne Rauschen liegt die erzeugt Policy hinter den anderen beiden Ansätzen. Bei vorhandenem Rauschens war das Verfahren in der Lage den Wagen genau an der Zielposition zu halten und somit einen hohen Reward zu akkumulieren. Allerdings muss auch darauf hingewiesen werden, das die Gauß'schen Prozesse hier Zusatzinformationen in Form der Stärke des Rauschens hatten. Lässt man dieses Apriori-Wissen weg und schätzt die Stärke des Rauschens als Hyperparameter, erhöht sich die

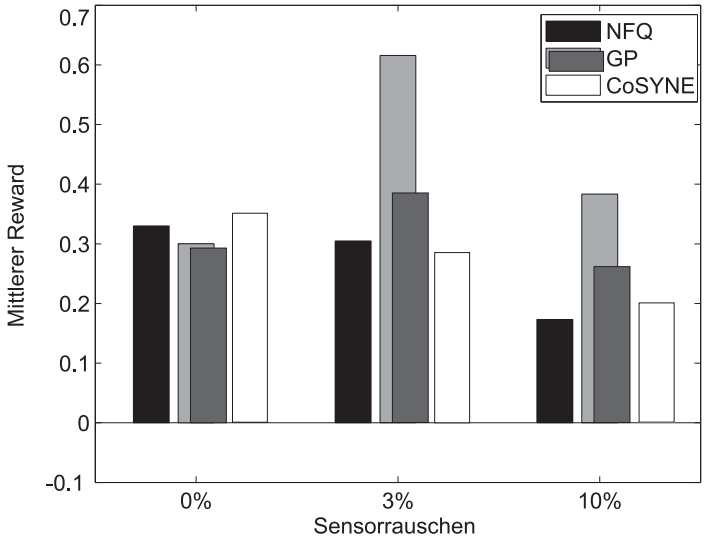


Abbildung 4.8.: Der mittlere Reward des jeweiligen Reinforcement Learning Agenten gemittelt über 5 Versuche. Maximaler Reward ist dabei 1, was bedeuten würden, dass der Agent bereits im Ziel gestartet wäre und sich dort die ganze Zeit befand. Das Minimum ist -0.1 was bedeuten würde, dass der Agent während der Episode nie in der Nähe des Ziels war und somit auch keinen höheren Reward erhalten hat. Je länger der Agent benötigt sein Ziel zu erreichen, desto geringer fällt der mittlere Reward aus. Beim GP Verfahren sind zwei Ergebnisse gezeigt. Einmal unter Verwendung der Information über das Rauschen (hinten, hellgrau) und einmal mit Schätzung dieses Wertes (vorn, dunkelgrau).

Rechenzeit deutlich und die Ergebnisse verschlechtern sich, liegen aber noch über den beiden anderen Verfahren.

Qualitativ sehr ähnliche Ergebnisse wurden in einem modifizierten Versuch erzielt. Dabei war die Start- und auch Zielposition ein und dieselbe Stelle des Hanges. Der Agent musste nur lernen, das Fahrzeug gegen die Schwerkraft zu halten. Alle oben aufgezählten Zusammenhänge zwischen Rauschen und Reward ließen sich auch hier wiederfinden.

4.4.2. Kraftwerkssimulator

Im Rahmen der Untersuchung im Hinblick auf das Kernanwendungsfeld, der intelligenten Feuerungsführung, wurde auf einen Simulator zurückgegriffen, welcher qualitativ die Herausforderungen in einem Kohlekraftwerk beschreibt. Dabei wird ein Verbrennungsofen mit einem nicht beobachtbaren Kohlezustrom simuliert. Aus diesem und der Stellgröße Luftzufuhr ergeben sich dann Kenngrößen, wie Ofentemperatur und Schadstoffausstoß.

Der simulierte Ofen besteht aus übereinanderliegenden Ebenen. Jede Ebene wird aus einer Kohlemühle gespeist und beinhaltet zwei Brenner (links und rechts). Wie die Kohle zwischen links und rechts verteilt ist, ist unbekannt. Allerdings muss die Luft, welches die relevante Stellgröße ist, für optimale Ergebnisse im gleichen Verhältnis verteilt werden. Ablesen lässt sich die Güte indirekt aus den Temperaturen, die an den Brennern herrschen, jedoch ist dieser indirekte Schluss nicht immer eindeutig. Daher handelt es sich hier um einen Problem, welches nur teilweise beobachtbar ist. Die Rewardfunktion setzt sich aus drei Elementen (Restsauerstoff, Kohlenmonoxid und Stickoxide) zusammen, die minimiert werden sollen, sich aber zum Teil konträr zueinander verhalten.

Eine detaillierte Beschreibung des Simulators inklusive des mathematischen Modells findet sich in Anhang C.

Dieses Szenario ist dabei nichtepisodisch, dass heißt es gibt keinen definierten Endzustand, bei dessen Erreichen der Versuch beendet wird, sondern der Verbrennungsprozess muss kontinuierlich geregelt werden.

Die wesentlichen Herausforderungen in diesem Szenario sind die unvollständige Beobachtbarkeit wichtiger Prozessgrößen, stark nichtlineare Zusammenhänge zwischen Aktionen und den daraus resultierenden Zustandsübergängen und damit einer komplizierten Q-Funktion, sowie der

Einfluss verschiedener Störgrößen (Messrauschen, systematische Störungen und langsame zeitliche Änderungen, die Verschmutzungen simulieren). Alle diese Schwierigkeiten wurden bewußt in dieser Form im Simulator integriert, um möglichst gut die Probleme bei der Feuerungsführung nachbilden zu können und setzen diese Szenario damit von den klassischen Benchmarks ab.

Experimente

Für die Verfahren wurden folgende Randbedingungen gewählt:

- NFQ: Es wurde ein Multi-Layer Perceptron mit zwei Hiddenschichten mit je fünf Neuronen als Approximator der Q-Funktion verwendet.
- Gauß'sche Prozesse: Es wurde nur ein Gauß'scher Prozess für die Q-Werte gelernt, es kam kein Prozessmodell zum Einsatz. Dieser Value GP wurde durch bis zu 10000 Supportpunkte im Zustandsraum approximiert. Die Hyperparameter für das Rauschen wurden vorgegeben.
- CoSYNE: Es wurde als Aktor ein vollständig rekurrentes Netz mit 6 Hiddenneuronen verwendet (entspricht damit annähernd der Zahl freier Parameter beim NFQ-Verfahren).

Auch hier lag das Hauptaugenmerk auf den unterschiedlichen Störungen, die den Prozess verkomplizieren. Dazu wurde das Sensorrauschen in drei Stufen betrachtet (0%, 3% und 10% Rauschstärke).

Zusätzlich wurden weitere Störungen(Verschmutzungen, systematische Störungen, etc.), wie sie in Anhang C.2 beschrieben werden, hinzugenommen um das Problem zu erschweren.

Die einzelnen Untersuchungen sollen hier nicht detailliert wiedergegeben werden (man findet diese in [Funkquist et al., 2009] und teilweise in [Barth, 2008] sowie [Hellwig, 2009]), sondern vielmehr werden die Ergebnisse und Schlussfolgerungen zusammengefassend dargestellt:

- Neural Fitted Q-Iteration

  Das NFQ-Verfahren erreichte durchweg gute Ergebnisse, die auch bei Experimenten mit allen Störungen signifikant besser sind, als wenn keine Regelung eingesetzt würde. Allerdings tendierte der Agent in einigen Experimenten mit vielen Störungen zu einer Übergeneralisierung, d.h. die ausgewählten Aktionen blieben für benachbarte Zustände gleich, auch wenn unterschiedliche Aktionen zu besseren Ergebnisse geführt hätten. Auch bei Verwendung eines größeren neuronalen Netzes ließ sich dieser Effekt beobachten, und steht vermutlich in Zusammenhang mit den Mehrdeutigkeiten des Problems.

  Auffällig ist, dass das Verfahren bereits mit vergleichsweise wenigen Beobachtungen zu seinen guten Ergebnissen kommt. Es ist im Vergleich mit den beiden anderen Verfahren klar das schnellste und dateneffizienteste Verfahren.

- Die Gauß'schen Prozesse scheiterten in diesem Szenario. Der durch die gelernten Policies akkumulierte Reward, lag nicht nur deutlich unter dem der beiden anderen Verfahren, sondern war in vielen Fällen schlechter als das Ausführen einer festen Aktion (keine Regelung).

  Der Grund liegt in der hohen Dimensionalität des Eingaberaums. Um eine nutzbare Approximation der Q-Funktion zu erhalten, war eine relative feine Abdeckung mit Supportpunkten notwendig. Dies führt zwangsläufig zu sehr großen Matrizen, welche in jedem Schritt invertiert und multipliziert werden müssen und somit schnell an praktische Grenzen der Hardware stoßen.

  Daher wurde die Anzahl der Supportpunkte begrenzt. Jedoch war es mit dieser begrenzten Anzahl von Supportpunkten nicht möglich die Q-Funktion sinnvoll zu approximieren. Auch eine Optimierung der Hyperparameter, also beispielsweise die Anpassung des geschätzten Rauschens in den Beobachtungen, brachte keine Verbesserung. Die Schätzung lief an den meisten Stellen des spärlich besetzten Zu-

standsaktionsraums auf den Mittelwert hinaus, was bei der Regelung der neutralen Aktion entspricht.

Es zeigte sich, dass die in Abschnitt 4.2 geäußerten Bedenken, was komplexere Szenarien angeht, gerechtfertigt sind.

- Der neuroevolutionäre CoSYNE-Ansatz erzielte im Sinne des erreichten Rewards die besten Ergebnisse. Auch unter dem Einfluss aller Störungen konnte eine gute Policy gefunden werden, die auch mit den Mehrdeutigkeiten des Problems umgehen konnte. Die Verminderung des Rewards in verrauschten Eperimenten war etwas geringer als beim NFQ-Verfahren. Dabei erwiesen sich die Ergebnisse als konsistent, was die Wahl verschiedener Lernparameter (z.B. Mutations- und Rekombinationswahrscheinlichkeit) angeht.

  Bei Versuchen, die nicht den Simulator selbst als Bewertung für die Policies verwendeten, sondern ein Prozessmodell (ebenfalls ein rekurrenten neuronales Netzes, welches per Evolutionsstrategie trainiert wurde) benutzten, ergaben sich sehr ähnliche, geringfügig schlechter Ergebnisse.

  Der nötige Rechenaufwand liegt zwischen den beiden anderen Ansätzen. Die Evolutionszyklen sind schneller als die Berechnung der Wahrscheinlichkeiten für die Gauß'schen Prozesse, können aber nicht mit dem Training des einzelnen neuronalen Netzes des NFQ mithalten. Zu dem wurde hier, wie auch beim Lernen mit den Gauß'schen Prozessen, keine Zeit berücksichtigt, die für das Training von Modellen notwendig ist.

Als Fazit aus diesen Untersuchungen ist mitzunehmen, dass die Gauß'schen Prozesse sich nicht problemlos auf komplexe Aufgaben übertragen lassen und daher im Rahmen dieser Arbeit nicht weiterverfolgt wurden. Sowohl das NFQ-Verfahren, als auch der CoSYNE-Ansatz erzielten zufriedenstellende Ergebnisse. Auch wenn die Ergebnisse des NFQ im Sinne des akkumulierten Rewards etwas schlechter ausfallen,

wird dies durch schnelles Lernen mit wenigen Daten kompensiert. Falls Rechenzeit unproblematisch ist, kann auch das CoSYNE-Verfahren verwendet werden.

4.5. Vergleiche in der Literatur

Hier soll kurz auf vergleichende Untersuchungen aus der Literatur und deren Ergebnisse eingegangen werden, soweit diese die betrachteten Verfahren oder nahe Verwandte betreffen.

In [Deisenroth, 2009] findet sich ein Vergleich zwischen NFQ (Abschnitt 4.1) und GPDP (Abschnitt 4.2). Anhand eines Pendelaufschwingproblems werden hier Qualität der Lösung und Rechenaufwand verglichen. Dabei bleiben die Ergebnisse der GP-Variante knapp hinter denen des NFQ-Verfahrens zurück. Das gilt sowohl für den akkumulierten Reward, als auch die notwendige Rechenzeit, wobei beachtet werden muss, dass hierbei für das GPDP bereits Optimierungen für den GP zur Approximation der Q-Funktion verwendet wurden.

In [Gomez et al., 2008] wird anhand eines Stabbalanceproblems CoSYNE (Abschnitt 4.3) gegen verschiedene Verfahren verglichen. Dazu zählen viele Evolutionsansätze, wie auch klassische Reinforcement Learning Methoden darunter *Q-Learning with MLP* (QMLP), welches dem NFQ vom Verfahren nahe kommt, ohne Wert auf eine effiziente Datenverarbeitung zu legen. Dabei erreicht das QMPLP Verfahren unter den verglichenen Value Function Methoden die besten Ergebnisse. Diese liegen auf gleichem Niveau mit dem CoSYNE Ansatz. Es wird auch ein Vergleich der Rechenzeit durchgeführt, allerdings sind die Aussagen zu QMLP nicht auf das NFQ-Verfahren übertragbar, da QMLP wesentlich ineffizienter ist als der NFQ-Ansatz. Das Szenario wurde dann auf ein Problem mit zwei Pendeln erweitert. Das CoSYNE Verfahren erzielt hier mit großem Vorsprung die besten Ergebnisse. Allerdings bleibt un-

klar, wie stark dieses spezielle Szenario auf die Stärken von CoSYNE anspielt und warum die anderen Verfahren so deutlich zurückfallen.

In [Taylor et al., 2006] und [Whiteson et al., 2009] wird bemängelt, dass es nur wenige Arbeiten gibt, die die grundlegend unterschiedlichen Ansätze des Temporal Difference (TD) Learnings (siehe Abbildung 4.2) und der Neuroevolutionsverfahren rigoros vergleichen. In den Publikationen werden SARSA als Vertreter des TD-Learnings und NEAT, ein Vorläufer und enger Verwandter von CoSYNE, verglichen. Dazu kommen das Mountain Car Szenario und das Keepaway Szenario aus dem RoboCup zum Einsatz. Die wesentliche Erkenntnis, die die Autoren aus ihren Ergebnissen ableiten, ist, dass im Falle eines vollständig beobachtbaren MDPs, die TD-Learning Ansätze schneller und zuverlässiger gute Ergebnisse erzielen. Im Falle von nur teilweise beobachtbaren POMDPs jedoch, kehrt sich dieses Verhältnis um. Die Neuroevolutionsverfahren verhalten sich hierbei signifikant robuster. Allerdings verlieren auch diese ihren Vorteil, falls auch die beobachteten Rewards nicht eindeutig sind.

Diese Ergebnisse aus der Literatur stehen in keinem Widerspruch zu den hier experimentell gewonnen Ergebnissen, sondern bestätigen diese und vervollständigen das Gesamtbild.

4.6. Fazit

Als prinzipielle Aussage aus diesem Abschnitt ist mitzunehmen, dass Reinforcement Learning Ansätze eine formidable Möglichkeit darstellen, ein Regelungsproblem in seinem Kern zu lernen und zu lösen. Welche konkreten Ansätze für spezielle Probleme die besten Ergebnisse liefern, kann auf der anderen Seite nicht apriori festgestellt werden.

Für den Anwendungskontext der in dieser Arbeit primär behandelt wird, erzielte das CoSYNE-Verfahren vielversprechenden Er-

gebnisse, wobei auch die Familie der NFQ-Ansätze sehr gut Ergebnisse lieferte. Klare Defizite zeigten sich bei auf Gauß'schen Prozessen basierten Verfahren für höherdimensionale Problemfälle, hier schlägt der von Bellman thematisierte Fluch der hohen Dimensionalität [Bellman, 1957] am deutlichsten zu. Zwar existieren in der Literatur (z.B. [Snelson und Ghahramani, 2006] oder [Jung und Stone, 2010]) auch Ansätze dieses Problem im Kontext der Gauß'schen Prozesse zu lindern, jedoch erfordert dies eine intensive Auseinandersetzung mit den Details der Gauß'schen Prozesse, was nicht Thema dieser Arbeit sein soll. Als Fazit verbleibt, dass die Gauß'schen Prozesse zwar großes Potential im Umgang mit verrauschten Daten besitzen, allerdings schwierig in der Handhabung sind. Auch die vergleichenden Untersuchungen aus der Literatur bestätigen den hier gewonnen Eindruck über die Stärken und Schwächen der einzelnen Verfahren.

Trotzdem soll hier nicht der Eindruck erweckt werden, dass Reinforcement Learning das einzig adäquate Mittel sei, um die Entscheidungsfindung im Rahmen der kognitiven Architektur durchzuführen. Es gibt eine Unzahl an weiteren Alternativen aus anderen Feldern. Ein paar wenige davon werden in Kapitel 6 vorgestellt und im Kontext der realen Anwendungen in einem Kohlekraftwerk mit dem CoSYNE-Algorithmus verglichen.

5. Lernmanagement

Betrachtet man die in den bisherigen Kapiteln vorgestellten Komponenten der Gesamtarchitektur, so sind bereits alle Bausteine zum Durchlaufen eines Wahrnehmungs-Handlungs-Zyklus vorhanden. Jedoch kann das System nur mit einer statischen Umgebung arbeiten. Sobald sich die Randbedingungen ändern, nutzt das bisher erworbene Wissen der Merkmalsextraktion oder des Reinforcement Learning Agenten weniger oder ist im schlimmsten Fall vollkommen unbrauchbar. Da die Annahme einer statischen Umgebung für viele Realweltanwendungen illusorisch ist, muss demzufolge eine Möglichkeit gefunden werden, beständig und flexibel auf Änderungen reagieren zu können und neues Wissen zu lernen.

Dazu wird auf die Aspekte des Stabilitäts-Plastizitäts-Dilemmas eingegangen, welches die Problematik zwischen Lernen und Vergessen thematisiert. Ebenfalls von Bedeutung ist die Frage nach einem Kompromiss zwischen dem Ausnutzen vorhandenen Wissens und dem Erwerb neuen Wissens, welches als Explorations-Exploitations-Dilemma bekannt ist. Diese beiden Aspekte werden in Bezug auf die in Kapitel 3 und 4 vorgestellten Teilsysteme diskutiert. Schlussendlich wird diskutiert, wie das Lernen im Falle von mehreren Agenten durch Rewarddekomposition beschleunigt werden kann.

5.1. Stabilitäts-Plastizitäts-Dilemma

In Szenarien in denen sich die Randbedingungen ändern, ist es notwendig, sich durch kontinuierliche oder zumindest regelmäßige Lernzyklen an diese Veränderungen anzupassen. Dabei ergeben sich zwei extreme Möglichkeiten, die sich aus dem Stabilitäts-Plastizitäts-Dilemma ableiten.

Definition 5.1

STABILITÄTS-PLASTIZITÄTS-DILEMMA

Als Stabilität wird die Fähigkeit der Verwendung von altem Wissen bezeichnet. Plastizität steht für die Fähigkeit eines Systems neue Zusammenhänge zu Erlernen. Aus der Problematik eines Gedächtnisses mit beschränkter Größe bzw. der Schwierigkeit in riesigen Wissensbasen effizient die richtige Antwort zu finden, ergibt sich das Stabilitäts-Plastizitäts-Dilemma. Wann kann altes Wissen verworfen, „vergessen“ werden um Platz für neues Wissen zu machen? Wie kann verhindert werden, dass der Erwerb neuen Wissens, das Verwerfen nützlichen alten Wissens erfordert?

Einerseits wäre es denkbar, das kognitive System komplett neu zu trainieren und alles bisher Gelernte zu ignorieren[1]. Ein solches Vorgehen ist nicht nur ineffizient, sondern auch im Vorbild der Natur nicht wiederzufinden. Eventuell ist ein komplettes Neutraining eines komplexen kognitiven Systems auch langsamer, als die Änderungen der Umgebung stattfinden.

Andererseits, keine Änderungen zuzulassen, löste das Problem auch nicht. Der Versuch, jede neue Beobachtung dem Gesamtwissen hinzuzufügen, erweist sich ebenfalls als schwierig. Nicht nur physikalische

[1] Wobei sich allerdings durchaus Abhängigkeiten durch eine teilweise gemeinsam genutzte Datenbasis ergeben können.

Limitierungen des Systems (Speicher, Rechenkapazität), sondern auch Komplexität des Gesamtprozesses beschränken, was effektiv erlernbar ist.

Ein weiterer wichtiger Aspekt ist, ob der Arbeitspunkt des Systems ein beobachtbarer Zustand ist oder dieser von versteckten Variablen abhängt, und wie vielfältig dieser Arbeitspunkt ist. Gibt es nur sehr wenige unterschiedliche Zustände und lassen sich diese auch noch einfach erkennen, dann spricht nichts dagegen, eine Art Datenbank zu nutzen, in der für den momentanen Arbeitspunkt der korrekte Regler nachgeschlagen wird.

Jedoch ist es für viele Anwendungen so, dass die Zahl der Randbedingungen und Zusammenhänge unüberschaubar groß und sehr komplex sind, als das sich für jede Änderung eine eigene Lösung vorhalten ließe. Auch das Problem, den korrekten Zustand zu erkennen, kann sich für verschiedene Probleme schwierig gestalten. Dann ist ein einfaches Wiederverwenden bekannter Lösungen ebenfalls problembehaftet. So bleibt in vielen Fällen nur die Lösung des Neulernens und Anpassens.

Daher stellt sich die zentrale Frage: Wie kann das bisherige Vorwissen beim Adaptieren das Systems an die neue Situation genutzt werden? Diskutiert werden soll dies an zwei Aspekten, die bisher in dieser Arbeit besprochen wurden. Dabei geht es um die Merkmalsextraktionsverfahren aus Kapitel 3 und das Reinforcement Learning aus Kapitel 4.

Natürlich können nur in Ausnahmefällen einzelne Teilaspekte einer kognitiven Architektur unabhängig von anderen nachtrainiert werden. So ist es beispielsweise nicht möglich, die Merkmalsextraktion zu ändern, ohne dass die Planungs- und Entscheidungsinstanz dahinter angepasst wird. Auch eine Anpassung möglicher Aktionen macht nur Sinn, wenn die Entscheidungsebene mit diesen neuen Möglichkeiten konfrontiert wird. Umgekehrt ist es allerdings sehr wohl möglich, die Planungsinstanz neu zu lernen, ohne dass die Merkmalsextraktion angepasst werden muss.

Im Rahmen der Architektur muss klar sein, welche Elemente von welchen anderen Elementen abhängen. Ebenso muss sichergestellt werden, dass, wenn eine Komponente einen Lernprozess initialisiert, alle abhängigen anderen Teile geeignet darauf reagieren, beispielsweise durch eine eigene Neuadaption.

5.1.1. Lebenslanges Lernen für Merkmalsextraktionsverfahren

Im Rahmen der Problematik aus Kapitel 3 ergibt sich die Frage, ob alle gewählten Merkmale immer noch relevant bzw. nützlich für das Problem sin. Oder gibt es vielleicht alte oder neue Kanäle, die momentan wichtiger sind? Ein einfaches Szenario dazu wäre der Ausfall eines wichtigen Sensors. Die damit assoziierten Variablen würden ihre Relevanz verlieren und sollten damit nicht weiter in einen Lernprozess einbezogen werden. Im Gegenzug sollte ein zweiter Sensor, der bisher nicht betrachtet wurde, da er nur redundante Daten lieferte, jetzt natürlich als Informationsquelle genutzt werden.

Merkmalsselektion

Mögliche Strategien müssen nach der Klasse der Merkmalsextraktionsverfahren unterschieden werden. Für Filterverfahren ergibt sich hier eigentlich nur die Möglichkeit der Neuberechnung des Relevanzwertes. Eine Nutzung vorhandenen Wissens kann erfolgen, indem nicht nur die aktuellen Werte betrachtet werden, sondern bisherige Relevanzwerte mit Berücksichtigung finden. Realisiert werden kann dies beispielsweise durch eine zeitliche Tiefpassfilterung.

Für Wrapper, und auch die ausführlich diskutierten Hybridverfahren mit Filter- und Wrapperanteilen, ergibt sich die Option, die bisher gewählten Merkmale als Startmenge zu verwenden und ausgehend

von diesen eine lokale Suche zu realisieren. Eine einfache Realisierung einer solchen lokalen Suche stellt die sogenannte Ersetzungssuche [Reunanen, 2006] dar. Dazu werden ausgehend von einer nicht leeren Startmenge (hier also die bisher verwendeten Merkmale) Merkmale einzeln ausgetauscht. Die bereits in Abschnitt 3.4 vorgestellte Floating Search Strategie realisiert dies durch abwechselndes Ausführen von Vorwärts- und Rückwärtssuchschritten.

Die Verfahren, welche auf dem Residuum als Auswahlkriterium basieren (siehe Abschnitt 3.6), können ebenfalls mit der vorherausgewählten Merkmalsmenge neugestartet werden. Dieses Vorgehen realisiert allerdings wiederum nur eine Vorwärtsauswahl, zum Entfernen nun irrelevanter Kanäle ist eine Form der Rückwärtssuche notwendig. Hierzu können sinnvollerweise Embedded-Verfahren, wie Optimal Brain Damage [Le Cun et al., 1990] bei neuronalen Netzen, eingesetzt werden. Embedded Verfahren realisieren eine Rückwärtssuche, die explizit die Nützlichkeit in Betracht zieht und aufgrund des Startens auf einer für gut befundenen Merkmalsmenge effizient realisierbar ist. Auf dieser so reduzierten Auswahlmenge, genauer gesagt über dem Residuum der für das Embedded Verfahren verwendeten neuronalen Netzes können dann direkt die Methoden angewendet werden.

Für das Chow-Liu Baum Verfahren (siehe Abschnitt 3.5) ergibt sich leider keine einfache Vorgehensweise, wie Wissen aus vorhergehenden Schritten übernommen werden kann. Die Struktur des Chow-Liu Baumes ändert sich unter Umständen deutlich. Daher ist es nicht möglich, zufällig verteilte Knoten (die bereits gewählten Merkmale) sinnvoll für eine effektive Suche zu nutzen. Insofern eignet sich dieses Verfahren nicht für ein adaptives Gesamtsystem, es sei denn, ein komplettes Neutraining ist durchführbar.

Merkmalstransformation

Detaillierte Untersuchungen in Hinblick auf die Adaptivität wurden für die in Abschnitt 3.7 vorgestellte Transinformationsbasierte Merkmalstransformation durchgeführt. Diese Untersuchungen wurden in [Schaffernicht et al., 2009c] publiziert. Ziel der Untersuchungen war es, zu evaluieren, wie stark die extrahierten Merkmale über der Zeit veränderlich sind.

Für die Transformationsmatrix W (siehe Abschnitt 3.7), welche den höchsten Informationsgehalt erzielt, gibt es unendlich viele korrekte Lösung selbst für den Fall, dass es ein eindeutiges Minimum existiert. Die Matrix kann mit einem beliebigen Skalar ungleich null multipliziert werden, ohne dass sich der Informationsgehalt ändert. Der Orthonormalisierungsschritt im Algorithmus 8 reduziert die Menge der gültigen Lösungen durch die Projektion auf den Hypereinheitskreis auf zwei. Dabei handelt es sich um W^* und $-W^*$, welche sich nur durch das Vorzeichen unterscheiden. Ein solches Verhalten ist nicht unbedingt erwünscht, wenn genau zwei gegensätzliche Matrizen die Lösung darstellen und zwischen zwei Optimierungsläufen diese unterschiedlichen Ergebnisse erzielt werden, da nachfolgende Instanzen im Wahrnehmungs-Handlungs-Zyklus sich darauf einstellen müssen.

Im Falle eines stationären Prozesses kann dieses Problem auf einfache Weise umgangen werden. Dazu kann mittels eines geeigneten Ähnlichkeitsmaßes das Ergebnis des letzten Optimierungslaufes w_{alt} mit dem neuen Ergebnis w_{neu} und $-w_{neu}$ verglichen werden und einfach das ähnlichere Ergebnis akzeptiert werden. Für instationäre Prozesse gestaltet sich das Definieren von sinnvollen Ähnlichkeitsmaßen und Schwellwerten jedoch meistens schwierig.

Der vielleicht offensichtlichste Ansatzpunkt ist die Initialisierung des Optimierungsprozesses. Anstelle eines zufälligen Startpunktes oder der Hauptkomponenten einer PCA ist es natürlich möglich, das vorherge-

hende Ergebnis der Optimierung als Ausgangspunkt zu nutzen. Wenn die Änderung des Prozesses langsam genug ist, sollte sich auch in den Ergebnissen der Transinformationsmaximierung eine langsame Verschiebung der relevanten Areale ergeben. Insbesondere für den Fall vieler lokaler Minima, was in der Praxis recht häufig der Fall ist, sorgt eine solche Startbedingung für das Finden eines nahegelegenen, neuen lokalen Optimums.

Für die Umsetzung der Adaptivität ergeben sich mehrere Möglichkeiten auf verschiedenen Zeitskalen. Eine Option ist es, den aktuellen Filter nach wenigen Messungen zu aktualisieren. Die dazu notwendigen Techniken werden in [Torkkola, 2003](Anhang A) beschrieben. Dabei wird nicht die gesamte gesammelte Datenmenge verwendet, sondern nur eine kleine Untermenge für einzelne Aktualisierungsschritte genutzt. Im Extremfall bedeutet dies die Verwendung von zwei Datenpunkten. Torkkola zieht diese zufällig aus allen Daten, im Sinne einer Online-Anwendung wären dies die letzten Beobachtungen. Für diese wird dann einfach ein Adaptionsschritt (Algorithmus 8) ausgeführt.

Jedoch führt dieser Ansatz für Anwendungen mit sehr stark verrauschten Daten zu dem Problem, dass der Filter versucht, sich an das Rauschen anzupassen, anstatt an die zugrundeliegende Prozessänderung. In diesem Fall scheint daher ein Mittelweg sinnvoll zu sein, bei welchem erst eine gewisse Menge an Daten gesammelt wird, um dann eine Aktualisierung mit diesen durchzuführen (*Batch Update*). Dabei muss auch darauf geachtet werden, dass die ausgewählte Trainingsmenge auch repräsentativ für die Datenverteilung ist, da sonst im Rahmen der Optimierung Lösungen bevorzugt werden, die eine schlechte Generalisierung aufweisen.

Im Rahmen der Feuerungsführungsanwendung in einem Kohlekraftwerk interessieren hauptsächlich die langsamen Änderungen im Prozess durch die Änderung der Kohlesorte und der Verschmutzung im Ofen. Natürlich gibt es auch hier Änderungen auf schnelleren Zeitskalen, diese

sind allerdings durch das starke Rauschen kaum zu detektieren.

Für die hier gezeigten Experimente wurde eine tägliche Aktualisierung durchgeführt, es kamen ähnliche Daten wie auch schon für die Experimente in Abschnitt 3.7.3 zum Einsatz. Dazu standen jeweils die fünf letzten Tage als Trainingsdaten zur Verfügung. Für acht aufeinanderfolgende Tage wurden diese Daten genutzt, um eine Hauptkomponentenanalyse (PCA), eine lineare Diskriminanzanalyse (LDA) und eine Transinformationsmaximierung (TIM) zu berechnen. Für die TIM wurden dabei drei unterschiedliche Initialisierungen verwendet. Dies waren erstens die Eigenflames einer PCA, die über dem gesamten Zeitraum berechnet wurde, und in einer realen Anwendung nicht zur Verfügung stehen würden. Zum Zweiten wurde das PCA-Ergebnis auf den aktuell verfügbaren Daten als Startpunkt verwendet. Im dritten Fall wurde das letzte Ergebnis der Transinformationsmaximierung verwendet, als Ausgangspunkt der Optimierung verwendet.

Ein Teil der Ergebnisse sind in Abbildung 5.1 gezeigt. Die Ergebnisse der PCA (obere Reihe) sind über die acht Tage am stabilsten, da die Varianz in den Daten sehr ähnlich ist. Nur zwischen Tag zwei und drei ist das angesprochene Problem des verdrehten Vorzeichens aufgetreten. Demgegenüber zeigt die LDA für jeden Tag sehr unterschiedliche Ergebnisse für jeden Tag. Die Ergebnisse der Transinformationsmaximierung, welche die PCA als Initialisierung verwenden, schwanken ebenfalls für jeden Tag. Verwendet man jedoch das vorhergehende Ergebnis als Startpunkt, ergeben sich nur geringfügige Änderungen. Die erzielten QMI-Werte (siehe Definition 3.27) für die Lösungen liegen dabei zahlenmäßig sehr nah beieinander, was dafür spricht, dass das Problem mehrere ähnliche lokale Minima aufweist. Durch die Verwendung des vorhergehenden Ergebnisses kann allerdings ein sehr ähnliches Minimum gefunden werden.

Wenn über den zu regelnden Prozess Vorwissen vorhanden ist, welches vermuten lässt, dass der Prozess sich nicht sprunghaft grundlegend

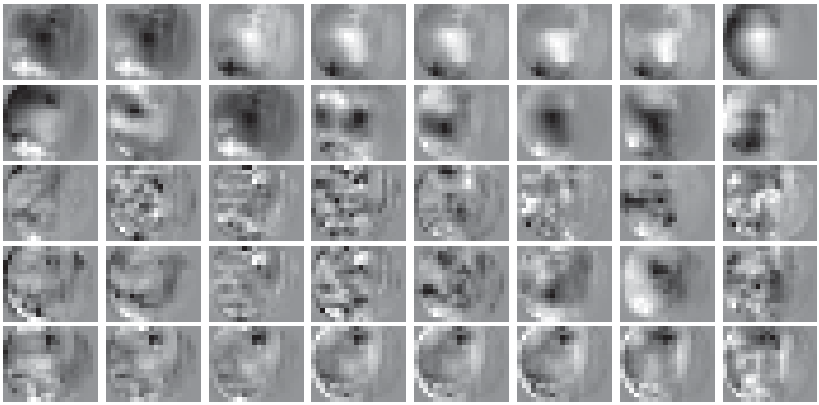


Abbildung 5.1.: Jede Zeile zeigt für jeweils ein Verfahren die erste Dimension des neuen Unterraums, jede Spalte entspricht einem Tag an dem ein Neutraining durchgeführt wurde. Für die LDA und TIM wurden die Stickoxide als Zielgrößen verwendet. Erste Zeile: konventionelle PCA. Zweite Zeile: Lineare Diskriminanzanalyse. Dritte Zeile: TIM, welche mit einer dem tagesaktuellen PCA-Ergebnis aus erste Zeile initialisiert wurden. Vierte Zeile: TIM, welche mit einer PCA über dem Gesamtzeitraum initialisiert wurde. Diese Starttransformation wurde dabei über alle Tage des Experiments berechnet. Fünfte Zeile: TIM, welche mit dem vorhergehenden Ergebnis initialisiert wurde. Von Interesse sind dabei die Änderungen von links nach rechts, bei denen möglichst wenig sprunghafte Änderungen gewünscht sind.

ändert, dann scheint eine solche Initialisierung sinnvoll. Wenn solche sprunghaften Änderungen jedoch regelmäßig auftreten, führt die Initialisierung mit dem vorhergehenden Ergebnis möglicherweise zu deutlich schlechteren Ergebnissen, da die guten Lösungen für die neuen Daten möglicherweise nicht mehr in der Umgebung des alten Ergebnisses liegen.

Unter Berücksichtigung dieser Erkenntnisse wird im Rahmen der hier verwendeten Architektur das letzte Ergebnis als Ausgangspunkt der

neuen Suche verwendet, da neben den oben besprochenen Eigenschaft anzumerken ist, dass das Verfahren wesentlich schneller konvergiert und somit potentiell öfter ein Nachtraining stattfinden kann.

5.1.2. Lebenslanges Lernen für Reinforcement Learning Strategien

In diesem Abschnitt soll diskutiert werden, ob und falls ja, wie, Wissen im Rahmen des Reinforcement Learnings wiederverwendet kann. Dazu werden die in Kapitel 4 vorgestellten Verfahren Neural Fitted Q-Iteration (NFQ) (Abschnitt 4.1) und Cooperative Synapse Neuroevolution (CoSYNE) (Abschnitt 4.3) bezüglich ihres Verhaltens bei Änderungen des zu optimierenden Problems hin untersucht. Es wird hierbei auf Ergebnisse aus den Diplomarbeiten [Barth, 2008] für das NFQ Verfahren und [Hellwig, 2009] für das CoSYNE Verfahren zurückgegriffen.

Es wurde der bereits in Kapitel 4 verwendete und in Anhang C erläuterte Simulator des Kraftwerks und das MountainCar Szenarios verwendet.

Beide Ansätze sammeln Beobachtungen für den Lernprozess. NFQ tut dies in Form von Tupeln, die direkt zum Training der neuronalen Approximation der Q-Funktion verwendet werden. Der CoSYNE Ansatz benutzt die Daten, um sein(e) Modell(e) zu adaptieren, welche benutzt werden, um die Regler zu bewerten.

Im Sinne des Stabilitäts-Plastizitäts-Dilemmas wäre das stabile Extrem, das Netz oder den Regler nicht zu verändern. Dies führt, je nach Änderung des Prozesses, zu einer deutlichen Verschlechterung, und das Ergebnis kann schlechter sein, als wenn auf jegliche Regelung verzichtet wird. Das plastische Extrem hingegen bedeutet ein komplettes Neutraining des Agenten, wodurch kein Wissen übernommen wird.

Die grundsätzliche Frage hierbei ist, ob sich das Verwenden vorheriger Ergebnisse und damit eine Wissensbewahrung positiv auf Lernergebnisse und -geschwindigkeit auswirkt.

Wiederverwendung von Wissen beim NFQ-Ansatz

Bei diesem Versuch wird wieder die Umgebung des MountainCar-Simulators verwendet. Es wurden zuerst 1000 Zustandsübergänge zufällig durchgeführt und basierend auf den gesammelten Datentupeln eine Policy gelernt. Danach wurde die Masse des Fahrzeugs geändert, welche entscheidend für das Verhalten des Fahrzeugs ist. Dabei wurde einmal die Masse verdoppelt und einmal halbiert. Mit diesen geänderten Randbedingungen wurden weitere 1000 Simulationsschritte durchgeführt, davon die Hälfte gemäß der bisher gelernten Policy und die andere Hälfte zufällig, also *off-policy*. Für die Untersuchungen hier wurde die zweite Adaptationsrunde nach der Masseänderung einfach mit den in der ersten Runde bestimmten Parametern gestartet. Eine exemplarische Untersuchung ist in Abbildung 5.2 gezeigt.

Das NFQ-Verfahren kommt bereits nach wenigen Episoden zu seiner initialen Policy. Nach einem Massewechsel benötigt das System aber länger. In den durchgeführten Experimenten dauerte es zwischen anderthalb und zweimal so viele Episoden, um sich auf die neuen Gegebenheiten einzustellen im Vergleich zu einem komplett neuen NFQ-Agenten, der ausschließlich das neue Problem mit halbierte Masse lernen sollte. Dieser erreicht den maximalen Reward in einem ähnlichen Zeitrahmen, wie für die initiale Policy gebraucht wurde.

Diese Beobachtung konnte auch im Kraftwerkssimulator gemacht werden. Die dort relevante Änderung ist die Kohlesorte, welche aufgrund unterschiedlicher chemischer Zusammensetzungen Änderung im Brennwert und dem Schadstoffausstoß nach sich zieht. Auch hier dauerte das

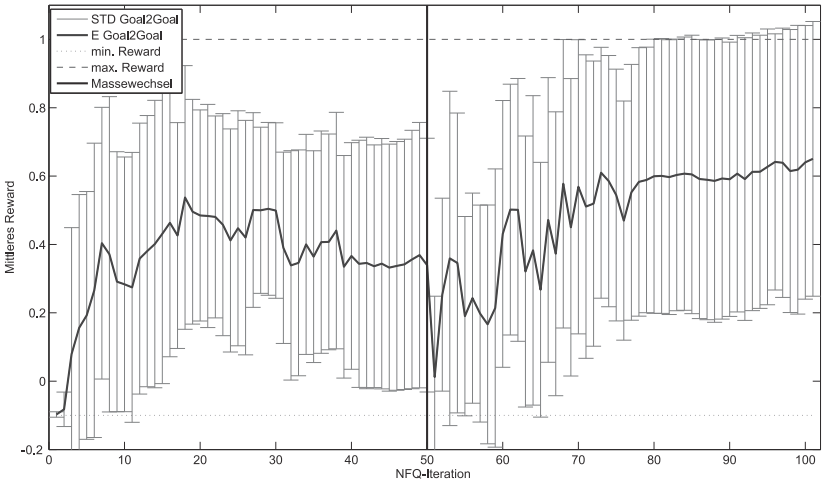


Abbildung 5.2.: Mittlerer Reward des NFQ Ansatz im MountainCar Szenario. Nach 50 Episoden wurde die Masse des Wagens halbiert. Der höhere durchschnittliche Reward der im Bereich von 51-100 Iterationen erreicht wird, kommt durch das leichtere Fahrzeug zustande, da in dem Fall nur ein sehr geringes Aufschwingen notwendig ist und das Ziel schneller erreicht werden kann. Im ersten Teil wird das (niedrigere) Endniveau nach rund 10 Episoden erreicht. Nach der Änderung der Masse ist deutlich der Einbruch der Belohnung zu erkennen. Nach rund 20 Schritten wird das neue erreichbare Belohnungsniveau erreicht.

Adaptieren eines vorhandenen Reglers länger als das komplette Neuerlernen.

Offensichtlich ist es bei diesen Experimenten der Fall, dass die alte Regelstrategie aufgrund der Änderungen kaum auf das neue Problem übertragen werden kann. Vielmehr erscheint es so, dass ein gewisser Aufwand betrieben werden muss, das alte Wissen zu verlernen. Der erzielte Gesamtreward ist in beiden Fällen nicht signifikant unterschiedlich, auch wenn auf dem Kraftwerkssimulator ein leichter Trend zu besseren Ergebnissen beim Wiederverwenden sichtbar war. Vermutlich

hängt dies mit einer besseren Optimierung im lokalen Bereich zusammen. Der Trainingszeitaufwand war jedoch, wie oben bereits erwähnt, höher.

Zusammenfassend kann gesagt werden, dass man sich durch das Weiterlernen des bestehenden Reglers nichts zerstört. Jedoch steht bringt es auch keine offensichtlichen Vorteile, gegenüber einem Neutraining, allerdings wird im Mittel mehr Zeit benötigt. Diese Ergebnisse entsprechen somit dem Ergebnis aus dem vorangegangenen Abschnitt zur Merkmalsextraktion, nur mit dem Unterschied, dass der Aufwand sich umgekehrt hat. Daher kann bei NFQ-Agenten grundsätzlich ein Neutraining durchgeführt werden.

Wiederverwendung von Wissen beim CoSYNE-Verfahren

Beginnt man dabei mit einer neuen zufälligen Startpopulation wird kein Wissen übernommen. Die einfachste Idee wäre an dieser Stelle, statt der zufälligen Population die Population des letzten Optimierungslaufes zu nutzen und von diesem zu starten. Dies entspricht der Strategie, die bei der Transinformationsmaximierung umgesetzt wurde.

Leider führt dies hier nur bedingt zum Erfolg. Der Lernprozess wird gestoppt, wenn das Ergebnis des besten Reglers sich über mehrere Schritte nicht mehr verbessert. Implizit führt das “Überleben des Stärksten” Prinzip zu einer zunehmenden Homogenisierung der Population. Die genetische Vielfalt verringert sich, da nur die Spezialisten überleben. Wenn bestimmte Aspekte nicht mehr in einer Population vorhanden sind, kann diese nur per Mutation wieder eingebracht werden. Damit helfen aber die Schritte der Rekombination und Koevolution nicht mehr.

Daher wurde untersucht, inwieweit eine Vermischung von Individuen aus dem letzten Lernprozess und zufälligen Individuen oder Individuen

aus länger zurückliegenden Populationen sich auswirken. Das Einbringen zufälliger oder älterer Individuen erhöht die genetische Vielfalt, was den Suchraum für die Optimierung vergrößert. Um zu verhindern, dass die Ergebnisse der letzten Population nach wenigen Schritten aussterben oder den zufälligen neuen Individuen diese Schicksal widerfährt, wurden 50 Prozent der letzten Population übernommen und die anderen 50 Prozent durch zufällige Individuen ersetzt. Bei Versuchen, die eine einfache Optimierung einer Funktion zum Ziel hatten, als auch beim Massewechsel im MountainCar Szenario, führte diese Kombination, verglichen mit einer rein zufälligen Population und einer vollständigen Population aus dem vorhergehenden Lernzyklus, am schnellsten zu den gewünschten Ergebnissen. Dieser Vorsprung betrug dabei bis zu 50 Prozent der benötigten Evolutionszyklen.

Jedoch zeigte sich, dass mit zunehmender Komplexität des Problems, beispielsweise im Kraftwerkssimulator, dieser Geschwindigkeitsvorteil dahin schmolz. Bei solchen herausfordernden Szenarien war am Ende kein signifikanter Unterschied in der erreichten Leistung oder der Lerngeschwindigkeit zwischen den unterschiedlichen Initialisierungsstrategien erkennbar.

Damit ergibt sich für das CoSYNE-Verfahren der Ansatz, dass die Startpopulation gemischt werden sollte und sowohl zufällige neue Individuen, für die genetische Vielfalt, als auch vorhergehende Ergebnisse einfließen sollten. Für den Fall, dass bekannt ist, dass die Prozessänderungen nicht zu groß sind, kann die Mutationsrate für die Individuen, die übernommen wurden erhöht werden um die Suche in der lokalen Nachbarschaft der alten Lösung zu verbessern.

5.1.3. Fazit

Die Ergebnisse der Untersuchungen in diesem Abschnitt waren in gewisser Weise ernüchternd in dem Sinne, als dass es oftmals keinen wesent-

lichen Unterschied macht, ob Vorwissen eingebracht wird oder nicht. Anscheinend sind in dem untersuchten Szenario des Kraftwerks die Änderungen so gravierend, dass das Vorwissen keinen hilfreichen Beitrag leistet. In einfacheren Szenarien hingegen konnten positive Aspekte beobachtet werden.

Auch wurde hier nicht untersucht, inwieweit ältere Lösungen, die vor dem letzten Ergebnis erzielt wurden, gewinnbringend in den Lernprozess eingebracht werden können. Dazu ist es notwendig, die bisherigen Ergebnisse in Relation zueinander zu setzen, das aktuelle Problem zu identifizieren und zu entscheiden welche Informationen genutzt werden sollten. Wie dies beispielsweise mit Hilfe einer Prozesskarte funktionieren könnte, wird im Sinne der Erweiterungen in Kapitel 7 erörtert.

5.2. Exploration-Exploitation-Dilemma

Eine große Herausforderung für jedes System, welches sich an ändernde Randbedingungen anpassen muss, ist die Frage, wie sehr und wann das System vom gelernten optimalen Verhalten abweichen darf und muss. Denn wenn sich die Umwelt verändert, ist der bisherige Aktionsplan nicht mehr zwangsweise der beste. Um eine bessere Aktionsfolge zu finden, ist es jedoch notwendig, andere Aktionen auszuprobieren, was in sich ein riskanter Vorgang ist. Formal lässt sich dies als Explorations-Exploitations-Dilemma (EED) beschreiben, was oft im Zusammenhang mit Reinforcement Learning Verfahren diskutiert wird.

Definition 5.2

EXPLORATIONS-EXPLOITATIONS-DILEMMA

Exploration bezeichnet die Suche nach neuem Wissen, d.h. es gibt keine oder kaum Informationen über die langfristigen Auswirkungen der Aktion, während Exploitation die Nutzung von vorhandenem Wissen beschreibt, d.h. der langfristige Reward bei Ausführung dieser Aktion

ist sicher gewinnbringend. Das Dilemma entsteht nun dadurch, dass ohne Exploration keine Verbesserung entstehen kann. Allerdings kann jeder Schritt der zur Exploration genutzt wird, deutlich schlechtere Ergebnisse erzielen, als wenn vorhandenes Wissen ausgenutzt worden wäre. Daher ist ein Kompromiss zwischen der Suche nach neuem, besseren Wissen und dem Nutzen vorhandenen Wissens notwendig.

Praktisch am weitesten verbreitete Ansätze sind heuristischer Natur, welche in [Thrun, 1992] systematisiert sind. Die bekanntesten Strategien sind dabei die ϵ-greedy Auswahl und die Boltzmann-Auswahl. Bei der ϵ-greedy Strategie wird einfach an jedem Entscheidungspunkt mit Wahrscheinlichkeit ϵ eine zufällige Aktion ausgewählt, während mit Wahrscheinlichkeit $1 - \epsilon$ die beste bekannte Aktion durchgeführt wird. Die Boltzmann-Auswahl kann als Erweiterung betrachtet werden, bei der ϵ nicht fest ist, sondern die zu Beginn sehr große Wahrscheinlichkeit ϵ wird über den Fortgang das Lernprozesses verringert. Diese Verringerung erfolgt dabei nach dem Temperaturabkühlungsschema, wodurch die Aktionsauswahl einer Boltzmann-Verteilung [Sutton und Barto, 1998] folgt. Erweiterungen dieser Heuristiken beziehen zusätzlich Information über die Zustände mit ein, beispielsweise die letztmalige Ausführung bestimmter Aktionen und Gesamthäufigkeit der Ausführung. In [Wiering und Schmidhuber, 1998] formuliert man gar aus diesen beiden Faktoren eine Rewardfunktion für ein neues Reinforcement Learning Problem zur Lösung des Dilemmas.

Für einfache akademische Szenarien existieren dazu Untersuchungen und Beweise so zum Beispiel in [Berry und Fristedt, 1985], [Narendra und Thathachar, 1989] und [Strehl und Littman, 2005]. Jedoch sind die behandelten Probleme alle diskreter Natur. Die in den Veröffentlichungen aus den Erkenntnissen abgeleiteten Algorithmen haben sehr harte Einschränkungen und werden daher kaum eingesetzt. Es existieren

viele Untersuchungen aus dem Bereich des Bayes'schen Reinforcement Learnings [Poupart et al., 2006], dem Lernen mit Gauß'schen Prozessen [Krause und Guestrin, 2007] der Informationstheorie [Iwata et al., 2004], und Erweiterung von ϵ-greedy und Softmax [Tokic und Palm, 2011] die versuchen mit unterschiedlichen Kriterien dem Explorations-Exploitations-Dilemma Herr zu werden.

Für die Anwendung in kontinuierlichen Aktionsräumen, also nicht auf einer endlichen Anzahl von möglichen Aktionen, sondern mit unendlich vielen Optionen ergeben sich zusätzliche Schwierigkeiten. Das $\epsilon-$greedy Äquivalent ist die Gauß-Exploration. Hierbei wird die beste Aktion um ein normalverteiltes Rauschen modifiziert, die Standardabweichung der Gaußverteilung σ steuert dabei analog zum ϵ das Maß an Exploration. Jedoch kann dieses Verfahren zu Oszillation und im schlimmsten Fall zur Divergenz führen [Peters und Schaal, 2008], so dass das Verfahren nie zu einer optimalen Policy findet.

Sampling-basierte Methoden bieten eine intuitive Möglichkeit, die Verteilung über kontinuierlichen Aktionsräumen darzustellen. Dabei repräsentieren durchgeführte Aktionen Datenpunkte im Aktionsraum und formen ähnlich zur Kerneldichteschätzung (siehe Abschnitt 3.3.1) eine Verteilung über die zu wählende Aktion. Einfache Sampling-Schemata werden in [Kearns et al., 2002], [Atkeson, 2007] und [Ross et al., 2008] vorgestellt. Man kann sich den Ablauf vereinfacht so vorstellen, dass immer, falls eine ausgeführte Aktion einen besseren langfristigen Reward erreicht, diese Aktion als Sample gespeichert wird und somit die Wahrscheinlichkeitsverteilung in Richtung der besseren Aktion verschiebt. Schlechtere Aktionen werden nicht aufgenommen und beeinflussen die Verteilung nicht.

Im Rahmen dieser Arbeit wurde versucht, diese Sampling Methoden dahingehend zu erweitern, dass durch eine geeignete Struktur das Explorations-Exploitations-Dilemma behandelt werden kann.

5.2.1. Diffusionsbaum-basiertes Reinforcement Learning

Die Grundidee dieser unter anderem im Rahmen von [Vollmer, 2009] und [Vollmer et al., 2010] entwickelten Variante des Reinforcement Learnings basiert auf der Idee der Sampling-basierten Ansätze, versucht jedoch explizit durch eine geeignete Struktur in der Repräsentation eine Lösung des Explorations-Exploitations-Dilemmas herbeizuführen.

Dabei wird für jeden Zustand die Explorationsgeschichte in einem lokalen Baum gespeichert. Zur Aktionsauswahl wird dieser Baum traversiert, wobei das Folgen bestehender Teile des Baums der Exploitation entspricht und analog das Bilden eines neuen Astes der Exploration. Die verwendete Struktur ist dabei von den sogenannten Dirichlet-Diffusionsbäumen abgeleitet, die daher als erstes kurz charakterisiert werden sollen. Danach wird diskutiert, wie dieser Baum verwendet wird, um die Exploration zu steuern.

Dirichlet-Diffusionsbäume

Dirichlet-Diffusionsbäume wurden von Neal zur Dichteschätzung und als Clusterverfahren vorgestellt [Neal, 2003]. Später wurden die Bäume auch erfolgreich zur Merkmalsselektion eingesetzt [Neal und Zhang, 2006]. Im Folgenden soll dabei nicht auf alle Details der Dirichlet-Diffusionsbäume eingegangen werden, sondern nur auf ihre Konstruktion, da das hier vorgestellte Diffusionsbaum-basiertes Verfahren diesen Konstruktionsprozess ausnutzt.

Ein solcher Baum entsteht dabei durch das sequentielle Ziehen von Beispielen/Partikeln. Die folgende Erläuterung wird inhaltlich von Abbildung 5.3 begleitet. Im ersten Schritt wird ein Beispiel an zufälliger Stelle im Raum (z.B. der Aktionsraum) initialisiert. Für eine Anzahl von Zeitschritten diffundiert das Partikel nun nach einem Brown'schen

Bewegungsmuster[2] umher. Der über die Zeit zurückgelegte Pfad wird gespeichert und bildet die erste Komponente des Baumes (Abbildung 5.3 Links). Der Endpunkt ist die ausgewählte Aktion. Wird nun ein zweites Beispiel gezogen, wird dies an derselben Stelle initialisiert, wie das erste Beispiel. Nach der Initialisierung folgt es dem Pfad des ersten Beispiels[3]. Zu einem zufällig bestimmten Zeitpunkt T_d divergiert das neue Beispiel nun vom Pfad seines Vorgängers und legt die restliche Zeit seinen Weg mittels der Brown'schen Bewegung zurück. Damit ergibt sich ein Zweig im Baum, der den neuen Pfad repräsentiert (Abbildung 5.3 Mitte). Der Zeitpunkt dieser Divergenz steuert den Ausgleich zwischen Exploration und Exploitation. Ein drittes Beispiel folgt zu Beginn wieder dem gemeinschaftlichen Pfad von dem es nach einer zufällig gewählten Zeit abweicht. Interessant wird es, falls das dritte Partikel vorher an einen Verzweigung im Baum kommt, dann muss entsprechend einer zu definierenden Wahrscheinlichkeitsverteilung entschieden werden, welchem Pfad das Partikel folgt. Es folgt dann weiter dem gewählten Ast, von welchem es später explorativ abweicht (Abbildung 5.3 Rechts). Welche Kriterien für den Divergenzzeitpunkt und dem Folgen welches Astes in Betracht kommen, wird im nächsten Schritt diskutiert.

Um Diffusionsbäum beispielsweise zum Clustern einzusetzen ist es darüber hinaus notwendig die korrekte Baumstruktur aus gegebenen Daten zu lernen. Die kann mittels des Metropolis-Hastings-Algorithmus realisiert werden [Neal, 2003]. Zur Behandlung des EED sind die hier

[2]Die Position wird zufällig um das Ergebnis eines Ziehens aus einer Normalverteilung mit dem Mittelwert null und einer gegebenen Varianz verändert. Die Zeitentwicklung kann daher auch als Gauß'scher Prozess betrachtet werden.

[3]Die Grundidee ist, dass man einfach den Pfad geht, den schon andere gegangen sind. Mathematisch gesehen spricht man auch von der Pólya Verteilung. Dabei wird das aus der Stochastik bekannte Urnenexperiment so modifiziert, dass nach dem Ziehen einer Kugel n weitere Kugeln der gleichen Farbe zurück in die Urne gelegt werden. Das bedeutet, wenn man eine weiße Kugel gezogen hat, wird diese und weitere weiße Kugeln in die Urne zurückgelegt und die Wahrscheinlichkeit wieder Weiß zu ziehen, steigt. Vorgestellt wurde sie in [Pólya, 1930] und ist in Standardwerken zu Wahrscheinlichkeitsverteilungen zu finden.

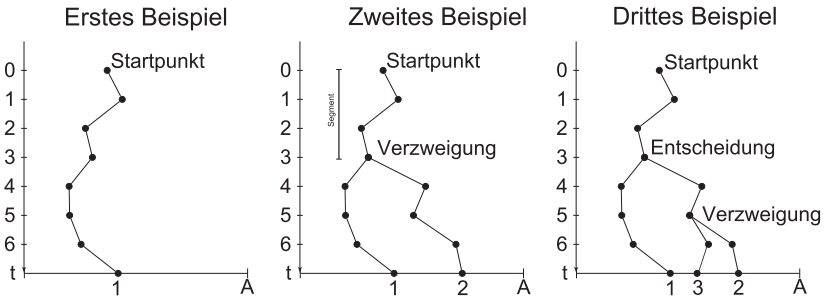


Abbildung 5.3.: Entstehung eines Dirichlet-Diffusionsbaums. **(Links)** Das erste Partikel bewegt sich für mehrere Zeitschritte (entlang der Y-Achse) nach einer Brown'schen Bewegung. Seine Endposition ist das Ergebnis der Ziehung, beispielsweise die ausgewählte Aktion im Aktionsraum A. **(Mitte)** Das zweite Beispiel folgt dem ersten Pfad bis zu einem Divergenzzeitpunkt t, ab welchem dem es abweicht und einen neuen Teilpfad generiert. **(Rechts)** Beim Ziehen eines dritten Beispiels folgt dieses vorhergehenden Pfaden bis es selbst wieder verzweigt. Sollte es an eine Verzweigung des Weges gelangen, muss es sich für einen Richtung entscheiden.

dargestellten Schritte jedoch ausreichend.

Neben der reinen Struktur des Baumes, welche aus der Wurzel, den Pfaden, den Verzeigungspunkten und den Blättern, also den Endpunkten, besteht, werden zusätzliche Informationen benötigt. Dazu wird Erstens ein Zähler eingeführt, der angibt, wie oft ein bestimmter Pfad bereits benutzt wurde. Zweitens wird für jedes Segment der maximal erreichte Q-Wert (siehe Definition 4.3) angegeben. Ein Segment ist dabei ein Baumabschnitt zwischen zwei charakteristischen Punkten. Bei diesen charakteristischen Punkten des Baumes handelt es sich um den Startpunkt, alle Verzweigungspunkte und alle Endpunkte.

Algorithmus

Der Algorithmus baut für jeden Zustand einen solchen Baum auf. Dieser dient dazu eine intelligente Samplingstrategie zu implementieren. Die Entscheidungsfindung entspricht dann einem Diffusionsprozess in diesem Baum. Bei den ersten Aktionen in einem Zustand soll im Sinne der Exploration früh vom Pfad des bisherigen Baums abgewichen werden, um andere Punkte im Aktionsraum zu erreichen und auszuprobieren. Später soll den guten Pfaden möglichst lange gefolgt werden und nur noch lokal um diese Aktionen herum exploriert werden. Wichtig anzumerken ist, dass wenn von der Zeit t gesprochen wird, keine Aktionen des Agenten gemeint sind, sondern eine interne 'Mikrozeit' die nur den Diffusionsprozess zur Aktionsauswahl betrifft.

In Abbildung 5.3, die einen möglichen Baum zeigt, ist die Abszisse mit A bezeichnet und stellt den kontinuierlichen Aktionsraum dar. Die Aktionsauswahl erfolgt einfach in dem ein Wert auf dieser Achse ausgewählt (*sampling*) und dann vom Agenten ausgeführt wird. Nach der Auswahl und Durchführung der Aktion wird der Q-Wert bestimmt (siehe Abschnitt 4.1) und im Baum an diesem Pfad gespeichert. Visualisiert mit einem einfachen Beispiel wird dies in Abbildung 5.4.

Besucht der Agent zum ersten Mal einen Zustand, existiert noch kein Baum[4] und wird ein Partikel zufällig im Aktionsraum eingefügt und folgt einer Brown'schen Bewegung. Die Aktionsauswahl ist also zufällig. Existiert bereits ein Baum, wird als erstes der Divergenzzeitpunkt T_d berechnet. T_d ergibt sich als Funktion in Abhängigkeit der Anzahl der Besuche in diesem Zustand. Je öfter der Zustand bereits besucht wurde, desto später das Sampling vom Pfad abweicht. Dahinter steht die Idee, dass je später die Diffusion stattfindet, desto weniger weicht die ausgewählte Aktion von bisherigen Aktionen ab. Die Anzahl gewählter Aktionen ist als Zähler z in den Segmenten des Baumes kodiert.

[4] Es wäre allerdings möglich hier einen Baum durch einen Experten vorzugeben und so Vorwissen einzubringen.

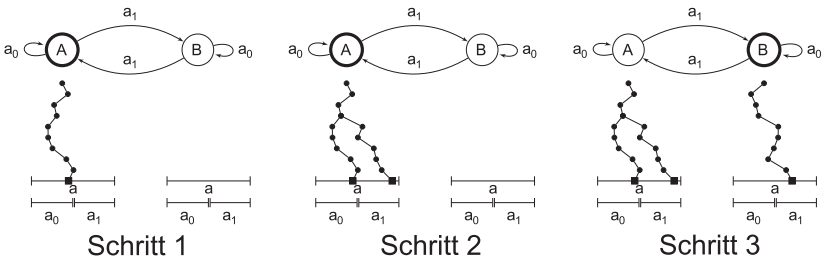


Abbildung 5.4.: Diese Abbildung zeigt beispielhaft die ersten drei Schritte beim Explorieren mit Diffusionsbaum-basiertem Reinforcement Learning. Es existieren zwei Zustände (A,B) und ein kontinuierlicher Aktionsraum. Bestimmte Aktionen a_0 führen zu einem Wechsel des Zustands, andere Aktionen a_1 führen zum Verbleiben im aktuellen Zustand. Für dieses Beispiel werden keine Aussagen über den Reward gemacht, der über diesem Aktionsraum definiert ist. In Schritt 1 befindet sich der Agent in Zustand A und es existiert noch kein Baum, daher wird zufällig ein Pfad gezogen und die Aktion an dessen Ende ausgeführt. Dies führt zu einem Verbleiben in Zustand A und zu Schritt 2. Für den vorhandenen Baum wird nun ein Divergenzzeitpunkt bestimmt. Bis zu diesem folgt die Aktionsauswahl dem alten Pfad, danach wird ein neuer Pfad erzeugt. Je später diese Divergenz stattfindet, desto weniger weit weicht die ausgewählte Aktion im Mittel ab. Die gefundene Aktion führt den Agenten in Zustand B, für den in Schritt 3 begonnen wird, einen eigenen Baum zu erzeugen.

Definition 5.3

DIVERGENZWAHRSCHEINLICHKEIT

Die Wahrscheinlichkeit zum Zeitpunkt t zu divergieren ist

$$p(t)dt = \frac{\gamma/(T_{max} - t)dt}{z}.$$

z ist dabei der Zähler wie oft der aktuelle Ast des Baums bereits beschritten wurde, T_{max} der Endzeitpunkt des Diffusionsprozesses und γ ein freier Parameter.

Mittels γ kann das allgemeine Verhalten des Explorationsprozesses gesteuert werden, große Werte fördern ein sehr exploratives Verhalten, während sehr kleine Werte schneller zu einer Exploitation führen. Dies entspricht funktionell dem Abkühlungsparameter bei der Boltzmannauswahl. Der Term $1/(T_{max} - t)$ sorgt für die strenge Monotonie, da mit dem Ende des Diffusionsprozesses $t \to T_{max}$ geht. Praktisch bedeutet dies, dass die Wahrscheinlichkeit zu divergieren steigt, je länger der Partikel dem Baum folgt. Mathematische Techniken um aus einer solchen Verteilung effizient Beispiele zu ziehen, werden in [Neal, 2003] vorgestellt.

Wichtig ist, dass die Wahrscheinlichkeit p, zu einem Zeitpunkt t zu divergieren, eine streng monoton steigende Funktion ist. Hintergründe zu dieser Bedingung und alternative Funktionen werden in [Neal, 2003] und [Vollmer, 2009] diskutiert.

Bis zum Zeitpunkt T_d folgt das Sample damit dem schon gegebenen Baum, danach geht es seinen eigenen Weg in Form einer Brown'schen Bewegung. Solange es dem Baum folgt, ist das Verhalten an Verzweigungen wichtig. Anstatt wie in den ursprünglichen Arbeiten der Pólya-Verteilung zu folgen, kommen hier die beobachteten Q-Werte ins Spiel. Im einfachsten Fall wird der Weg gewählt, in dessen Segment der bisher höchste Q-Wert beobachtet wurde.

Diese Auswahl führt unter Umständen zu einer sehr fokussierten Exploration um den bisher beobachteten maximalen Q-Wert. Um dies zu umgehen, gibt es zwei Möglichkeiten. Entweder man führt auch an dieser Stelle eine probabilistische Auswahl, beispielweise ϵ-greedy basiert, ein, oder man erhöht den weiter oben angesprochenen γ-Faktor. Ersterer Ansatz bringt mit sich, dass es neue Parameter gibt, allerdings kann so das Explorationsverhalten modularisiert werden. Der zweite Weg hingegen erhöht einfach die Wahrscheinlichkeit, dass der Pfad divergiert bevor man an eine Abzweigung kommt.

Dieses gesamte Vorgehen führt dazu, dass am Anfang häufig früh vom

Pfad abgewichen wird und damit eine Exploration des Aktionsraumes stattfindet. Mit zunehmender Beobachtungsdauer wird immer später divergiert und damit nur noch sehr eng um die bisherigen Pfade exploriert. Die Verzweigungsregel führt dazu, dass dieses eingeschränkte Explorieren um jene Zweige herum stattfindet, die einen großen Q-Wert als Belohnung versprechen.

Experimente

Um die prinzipielle Funktionalität des hier vorgestellten Ansatzes zu zeigen, wurden zwei Szenarien untersucht. Einerseits ist dies ein Gridweltszenario der Größe 5x5 und andererseits wurde ein Pendel simuliert, welches in aufrechter Position stabilisiert werden sollte. Dabei wurden andere Szenarien gewählt, als die bisherigen Untersuchungen, da hier die prinzipielle Funktionsweise nachgewiesen wird.

In der Gitterwelt bestand die Aufgabe des Agenten darin, einen Zielpunkt anzufahren. Damit ergeben sich automatisch diskrete Zustände als zweidimensionale Gitterposition. Die diskreten Aktionen links, rechts, oben und unten, wie sie für Bewegungen in Gitterwelten typisch sind, wurden auf einen kontinuierlichen Aktionsraum von null bis eins projiziert. Das heißt, die Aktion links wird im Intervall $[0, 0.25)$ ausgeführt, rechts im Intervall von $[0.25, 0.5)$ und so weiter.

Dies erscheint zunächst unsinnig, hat aber für die Experimente den Effekt, dass der Aktionsraum an den Intervallübergängen zwischen den Aktionen unstetig ist, was für Sampling-basierte Verfahren eine große Herausforderung ist, da bei der Schätzung der Wahrscheinlichkeit mit Partikeln immer eine Form der Interpolation zur Anwendung kommt. Damit lassen sich Verteilungen nahe eines solchen Übergangs nur schwer repräsentieren.

Ein positiver Reward wird für das Erreichen des Ziels vergeben. Von Interesse ist hierbei die Anzahl der Schritte, die der Agent zum Erreichen

des Zielzustandes in einer Episode benötigt. Die Ergebnisse wurden dabei über zehn Versuche gemittelt.

Verglichen wurde der neue Ansatz mit einfachem Random Sampling-basiertem Reinforcement Learning (RSQL) [Atkeson, 2007] und einfachem Q-Lernen [Sutton und Barto, 1998]. Das einfache Q-Lernen ist hier klar im Vorteil, da es nur die vier diskreten Aktionen benutzt und somit als Vergleich für die wesentlich herausfordernden kontinuierlichen Aktionsräume dient. Das RSQL basiert auf dem Ziehen einer zufälligen Aktion, die mit Wahrscheinlichkeit p ausgeführt wird. Mit Wahrscheinlichkeit $1-p$ wird dagegen die bisher beste zufällige Aktion ausgeführt. Die Bewertung der Aktion erfolgt dabei über den Q-Wert. Die Wahrscheinlichkeit p beginnt dabei bei 1 und nimmt während des Lernens kontinuierlich ab. Damit ist dieses Verfahren ein einfaches, intuitives Sampling-basiertes Verfahren.

In Abbildung 5.5 sind die Resultate abgetragen. Erwartungsgemäß erreicht das nur auf diskreten Aktionen operierende Q-Lernen am schnellsten das Ziel. Für die beiden samplingbasierten Verfahren ergibt sich eine langsamere Konvergenz. Der Diffusionsbaumansatz ist jedoch deutlich schneller als der einfache RSRL-Ansatz. Betrachtet man die entstehenden Bäume näher, so fällt auf, dass diese den Bereich der korrekten Aktion deutlich schneller und zielgerichteter ausgewählt werden, als beim einfachen Sampling, welches nicht auf die Historieninformation des Baumes zurückgreifen kann. Stattdessen zieht RSQL vergleichsweise häufig Aktionen, die nicht in die richtige Richtung führen.

Beim zweiten Szenario, dem Balancieren eines umgekehrten Pendels [Doya, 2000], geht es darum, dieses möglichst lange mittels eines Drehmotors in einem aufrechten Zustand zu halten. Der Zustandsraum ist zweidimensional und besteht aus der Position des Pendels als Winkel zwischen 0 und 360 Grad und der Winkelgeschwindigkeit des Pendels zwischen $\pm 10 \frac{rad}{s}$. Für die Experimente wurde der Zustandsraum in 41 Intervalle unterteilt. Der Aktionsraum wird über die Winkelbeschleu-

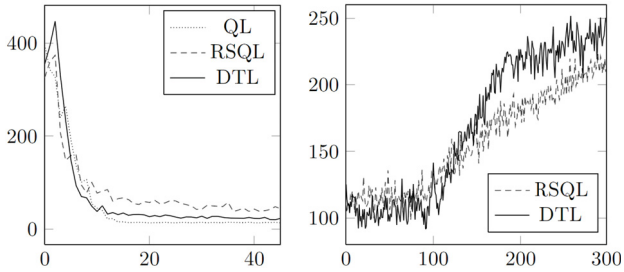


Abbildung 5.5.: **(Links)** Ergebnisse für das Gitterweltszenario. Auf der Abszisse sind die Episoden abgetragen, die das System gelernt hat. Die Ordinate zeigt die durchschnittliche Anzahl der Schritte, die der Agent zum Erreichen des Zielzustandes benötigt. **(Rechts)** Ergebnisse für das Pendel. Es sind die Anzahl der Episoden gegen die Zeit, die das Pendel stabilisiert werden kann, aufgetragen. QL bezeichnet dabei das einfache Q-Learning, RSQL das Random Sampling Q-Learning und DTL das Diffusionsbaumbasierte Reinforcement Learning.

nigung definiert, die zwischen $\pm 10Nm$ liegt und kontinuierlich ist.

In Abbildung 5.5 kann man sehen, dass der Diffusionsbaum-basierte Ansatz schneller das Pendel länger aufrecht halten kann. Auch hier zeigt sich, dass durch den Baum die Region, in denen das Pendel senkrecht gehalten wird und der Reward somit hoch ist, sehr intensiv gesampelt wird, während das RSQL seine Aktionen weniger zielgerichtet auswählt und so langsamer konvergiert.

Für wesentlich ausführlichere Experimente und Diskussion verschiedener Varianten dieser Idee sei [Vollmer, 2009] verwiesen.

Fazit

Im Rahmen dieser Arbeit wurde ein neuer Algorithmus entwickelt, der es ermöglicht, die Explorationsstrategie für kontinuierliche Aktionsräume explizit in einer Baumstruktur zu repräsentieren. Eine Behandlung

des Explorations-Exploitations-Dilemmas wird über diesen Baum gesteuert. Es konnte in Experimenten gezeigt werden, dass dieser Samplingansatz Vorteile gegenüber klassischer Exploration mit Samplingstrategien hat.

Der hier vorgestellte Ansatz hat allerdings die wesentliche Einschränkung, dass er nur für diskrete Zustandsräume funktioniert, da jeder Zustand einen eigenen Baum besitzt, der die Explorationsinformationen speichert. Neben dem Speicherplatzbedarf ergibt sich für praktische Probleme die Frage nach kontinuierlichen Zustandsräumen. Will man dieses Verfahren ohne Diskretisierung auf kontinuierliche Zustandsräume übertragen, müssen Lösungen gefunden werden, um entweder zwischen vorhandenen Bäumen interpolieren zu können oder aber die Baumstruktur muss so erweitert werden, dass auch die Zustandsinformation implizit als Teil der Baums und des Diffusionsprozesses verwendet wird.

Ebenfalls von Interesse für Arbeiten in dieser Richtung ist die Frage nach einem Pruning, also dem Ausdünnen des Baums. Die Plastizität des Verfahrens im Laufe der Zeit immer mehr und die Partikel folgen dann nur noch dem Baum. Um sich also auf neue Situationen einstellen zu können, ist es notwendig den Baum nicht zu groß werden zu lassen, so dass auch wieder explorative Aktionen durchgeführt werden.

Die genannten Problematiken wurde im Rahmen dieser Arbeit jedoch nicht weiterverfolgt, sollen allerdings als Impuls für zukünftige Arbeiten verstanden werden. Für praktische Anwendungen im Kraftwerk (siehe Kapitel 6) erwies sich das hier vorgestellte Verfahren jedoch als noch zu wenig praxistauglich und wurde daher nicht benutzt. Stattdessen wird dort wieder auf die einfachen, zu Beginn dieses Abschnittes vorgestellten Verfahren, wie die ϵ-greedy Strategie zurückgegriffen.

5.3. Rewarddekomposition

Ein weiterer interessanter Aspekt ist, dass sich oftmals komplexe Aufgaben in einfachere Teilaufgaben zerlegen lassen. Ob diese Zerlegung dabei durch Experten vorgenommen wird oder aus den Daten gelernt wird, sei für diese Arbeit unerheblich. Die Idee dahinter ist, dass diese Teilprobleme sich einzeln leichter lösen lassen, anstatt die Summe der Probleme einem Monolithen zu überlassen. Diese Teilaufgaben können dann im Sinne eines kooperativen Multiagentensystems [Jennings, 1994] angegangen werden, wobei jeder Agent mit der Lösung eines solchen Teilproblems zur Gesamtlösung beiträgt. Wenn dabei von Multiagentensystemen gesprochen wird, geht es hier nur um den Teilaspekt der Problemlösung und nicht um multiple Instanzen der gesamten kognitiven Architektur, welche miteinander interagieren.

Ein praktisches Problem entsteht, wenn die Teilagenten ihr Verhalten lernen sollen, als Rückkopplung aber nur eine Gesamtbewertung für das vollständige Problem vorliegt. Weiter unter wird gezeigt, dass diese Gesamtbewertung bei direkter Verwendung das Finden einer guten Lösung unter Umständen unmöglich macht.

Wenn jeder Agent nur den gesamten Reward bekommt, spiegelt sich darin nicht seine wirkliche Leistung wieder. So wird eventuell ein Agent, der ein schlechtes Verhalten aufweist, belohnt, wenn alle anderen Agenten hohe Rewards erzielen. Umgekehrt wird ein Agent mit einer guten Policy bestraft, nur weil alle anderen Agenten eine schlechte Aktion ausgeführt haben.

Daher ist es notwendig, die Gesamtbewertung leistungsgerecht zwischen den Teilproblemlösern aufzuteilen. In der Literatur wird diese Aufgabe als Rewardddekompositionsproblem oder *Structural Credit Assignment* Problem bezeichnet.

Definition 5.4

REWARDDEKOMPOSITION

Ziel der Rewarddekomposition ist es, einen beobachteten globalen Reward R_{Gesamt} so auf die n kooperativen Agenten zu verteilen, dass die lokalen, agentenspezifischen Rewards R_i dem Leistungsanteil des Agenten am Gesamtreward entsprechen.

Die Summe dieser Einzelrewards ergibt den Gesamtreward

$R_{Gesamt} = R_1 + R_2 + ... + R_n$.

Die hier vorgestellten Untersuchungen basieren dabei auf der Diplomarbeit von Markus Eisenbach [EISENBACH, 2009].

5.3.1. Experimentelles Szenario

Wenn man das Feuerungsführungsproblem (siehe Einleitung bzw. Kapitel 6) betrachtet, lässt sich durch Expertenwissen eine einfache Unterteilung ermitteln. Jede Brennerebene, die durch zwei Brenner mit gemeinsamer Kohlezufuhr gekennzeichnet ist, lässt sich als eigener Agent auffassen, der die Luftzufuhr für seine Ebene kontrolliert. Trotzdem lässt sich nur ein gemeinsamer Reward für den gesamten Ofen definieren, da die Abgase und der Wirkungsgrad nur für den Kessel als Ganzes bestimmt werden können.

Um ein besseres Verständnis für die Problematik zu erhalten, wurde das Problem auf ein ähnliches Szenario übertragen, welches jedoch in diskreten Zustandsaktionsräumen definiert ist. Bei diesem Szenario handelt es sich um Agenten in jeweils einer eigenen Gridwelt. Jeder Gridwelt ist das Äquivalent zur Regelung einer Brennerebene und die Zielposition für den Agenten innerhalb der Gridwelt entspricht der gesuchten Luftverteilung. Dargestellt ist diese Idee in Abbildung 5.6.

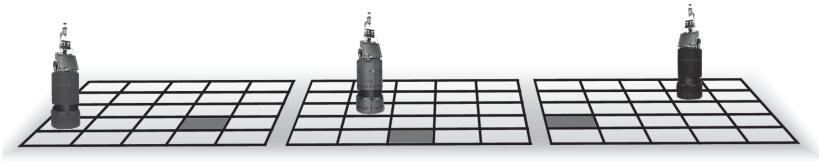


Abbildung 5.6.: Gridweltszenario für die Rewarddekomposition. Drei Agenten agieren in ihrer eigenen Gridwelt und müssen ihr markierte Zielposition erreichen. Als Information über ihre Leistung bekommen sie aber nur die Summe über die Rewards aller Agenten und kennen ihren wahren lokalen Reward nicht.

Im einfachsten Fall müssen die Agenten unabhängig voneinander zu ihrem Ziel finden. Als Belohnungsinformation erhalten sie allerdings nur die Summe über die Rewards der einzelnen Agenten. Diese ergibt sich für den einzelnen Agenten aus $10 - \|Ziel - Position\|_{L1}$. Je näher ein Agent am Ziel ist, desto höher ist der Reward.

Typischerweise ist es jedoch so, dass die Agenten sich gegenseitig beeinflussen. Die eingestellte Luftverteilung auf einer Ebene des Ofens verändert die optimale Luftverteilung in den Ebenen darüber und darunter. Dieses Phänomen wird dadurch modelliert, dass die Position des Agenten auf einer Ebene, das Ziel für einen Agenten auf einer anderen Ebene verändert. Dargestellt und erläutert ist dies in Abbildung 5.7.

5.3.2. Ansätze zur Rewarddekomposition

Das allgemeine Vorgehen zur Lösung des Rewarddekompositionsproblems beinhaltet die Zerlegung des globalen Rewards für jeden Agenten einzeln in einen lokalen Reward. Dieser lokale Reward ist dabei für jeden Agenten die Repräsentation seines Anteils am Gesamtreward. Dieses Umrechnen des globalen Rewards wird auch als *Reward Shaping* bezeichnet.

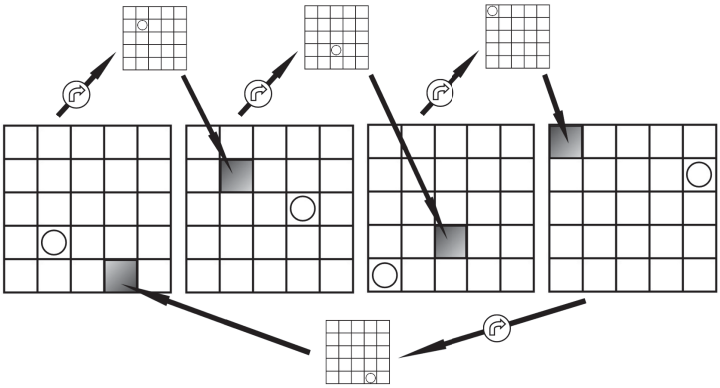


Abbildung 5.7.: Beispiel mit vier Agenten, die voneinander abhängige Zielpositionen besitzen. Dabei definiert sich das aktuelle Ziel eines Agenten, aus der um 90 Grad im Uhrzeigersinn gedrehten Position seines Vorgängers. Die hier dargestellte Variante mit einem Kreis von Abhängigkeiten ist dabei das komplexeste Szenario, das betrachtet wurde. Die triviale Lösung ergibt sich, sobald alle Agenten das mittlere Gitterfeld ansteuern. Falls dieses Feld entfernt wird, ergeben sich eine Zahl anderer optimaler Lösungen. Das Erlernen dieser ist nur mittels eines zerlegten Rewards möglich. Eine einfachere Variante des Problems stellt eine Kette von Abhängigkeiten dar, bei der der erste Agent ein fest vorgegebenes Ziel hat.

Alle Algorithmen folgen dabei einem einheitlichen Ablauf.

1. Alle Agenten führen eine zufällige oder der Policy folgende Aktion aus.
2. Alle Agenten beobachten den gemeinsamen globalen Reward r_{global}.
3. Aus dem globalen Reward berechnet jeder Agent für seinen aktuellen Zustand einen lokalen Reward r_{local}. Ansätze hierzu werden im Folgenden vorgestellt.
4. Der berechnete lokale Reward zusammen mit der Aktion und dem Zustand wird der Trainingsdatenbank hinzugefügt (z.B. beim NFQ

oder CoSYNE-Verfahren, siehe Kapitel 4) oder direkt zum Aktualisieren der Policy verwendet (z.B. einfaches Q-Learning). Danach beginnt wieder Schritt 1.

Definition 5.5

EIGENSCHAFTEN DES LOKALEN REWARDS

Der berechnete lokale Reward sollte zwei Eigenschaften erfüllen.

1. **Rewardskalierbarkeit:** Der errechnete lokale Reward muss dem wahren lokalen Reward entsprechen. Die einzigen zulässigen Änderungen sind dabei eine feste Translation (Addition mit dem gleichen Wert für alle Agenten) und/oder eine feste Skalierung (Multiplikation mit einem Wert).
2. **Rewardreproduzierbarkeit:** Der lokale Reward muss für die gleichen Zustandsaktionspaare für den Agenten einen Markoventscheidungsprozess darstellen.

Die Eigenschaften leiten sich aus den Ergebnissen aus [CHANG et al., 2003] ab. Die erste Eigenschaft sagt nichts anderes, als dass die optimale Lösung auch mit einem skalierten Reward gefunden wird. Die zweite Eigenschaft bedeutet, dass der Einfluss der anderen Agenten auf den lokalen Reward ausgeschlossen werden muss. Dies wird bei der Verwendung des globalen Rewards nicht gewährleistet, und führt somit zu Problemen beim Lernen.

Algorithmen zur Bestimmung des lokalen Rewards

Es wurden fünf Algorithmen zur Rewarddekomposition verglichen. Vier davon entstammen aus der Literatur [PANAIT und LUKE, 2005], [CHANG et al., 2003] und [MARTHI, 2007], während das SMILE Verfahren eine Eigenentwicklung darstellt, die im Rahmen einer Diplomarbeit [EISENBACH, 2009] ausführlich untersucht wurde.

1. **Maximum über die Historie der Rewards**

   Eine der einfachsten Varianten, einen solchen lokalen Reward für jeden Zustand zu ermitteln, ist das Maximum über alle bisher beobachteten globalen Rewards des Zustands als lokalen Reward zu verwenden.

$$r_{local}(s') \leftarrow \max(r_{local}(s'), r_{global})$$

   Die Idee dahinter ist, dass über hinreichend viele Beobachtungen alle anderen Agenten ebenfalls ihren maximalen Reward beobachten. Dadurch, dass immer der Maximalwert übernommen wird, bildet sich damit ein für alle Zustände gleicher Offset, der der Summe der maximalen Rewards aller anderen Agenten entspricht (siehe *Rewardskalierbarkeit*). Dadurch verbleiben als einzige Einflussgrößen für den lokalen Reward eines Zustands die eigenen Aktionen des Agenten.

   Der Nachteil dieses Ansatzes wird klar, sobald die beobachteten Rewards verrauscht sind und damit die Annahme, dass die unterschiedlichen Werte nur durch die eigenen Aktionen induziert sind, hinfällig ist. Ebenfalls problematisch sind Änderungen im vergebenen Reward, wenn der gleiche maximale Reward in einem anderen Zustand vergeben wird. Das Maximum kann nicht vergessen werden und somit ist in der Repräsentation des ermittelten lokalen Rewards die bisherige Lösung genauso gut wie die neue Lösung.

2. **Mittelwert über die Historie der Rewards**

   Basierend auf demselben Grundgedanken kann das Maximum über die beobachteten Rewards durch den Mittelwert über die Beobach-

tungen ersetzt werden.

$$r_{local}(s') \leftarrow \frac{r_{local}(s') \cdot count(s') + r_{global}}{count(s') + 1}$$
$$count(s') \leftarrow count(s') + 1$$

Damit wird der lokale Reward um die Summe der Mittelwerte der anderen Agenten verschoben und die Variation in jedem Zustand ergibt sich durch die eigenen Aktionen.

Der wesentliche Unterschied ist, dass damit auch auf Veränderungen in der Rewardfunktion und Störungen wie Rauschen gehandhabt werden können. Allerdings ist dieser Ansatz langsamer, was das Lernen angeht, da für die Schätzung des Mittelwerts mehrere Beobachtungen notwendig sind, während beim Maximum im besten Fall eine einzige Beobachtung reicht.

3. **Kalman-Filter über die Historie der Rewards**

In [Chang et al., 2003] wird die Idee des zweiten Ansatzes erweitert. Es wird dabei ein Kalmanfilter eingesetzt, um den Mittelwert über die globalen Rewards zu schätzen.

$$\mu(s') \leftarrow \mu(s') + \frac{\sigma(s') \cdot (r_{global} - \mu(s'))}{\sigma(s') + \sigma_{r_{global}}} \tag{5.1}$$
$$\sigma(s') \leftarrow \sigma(s') \cdot \left(1 - \frac{\sigma(s')}{\sigma(s') + \sigma_{r_{global}}}\right) \tag{5.2}$$
$$r_{local}(s') \leftarrow \mu(s') \tag{5.3}$$

$\sigma_{r_{global}}$ ist dabei ein Hyperparameter, der Aussagen über die Unsicherheit beim globalen Reward zulässt. Je kleiner diese Varianz gewählt wird, desto schneller konvergiert das Verfahren, ist dann aber anfälliger gegenüber Rauschen.

In [Chang et al., 2003] wird vorgeschlagen, nicht nur den eigenen Anteil am Gesamtreward zu schätzen, sondern auch den Anteil der anderen Agenten an diesem. Dieser Wert wird mit einem weiteren Kalmanfilter geschätzt und als erstes vom globalen Reward abgezogen. Mit diesem offsetbereinigten Reward, berechnet dann der Kalmanfilter zur lokalen Rewardschätzung das Ergebnis.

Dieser zusätzliche Schritt sorgt für eine schnellere Konvergenz, da durch die wechselseitige Schätzung des eigenen Anteils und des Anteils der anderen Agenten das Problem der Skalierung eliminiert wird. Der Preis dafür ist ein erhöhter Rechenaufwand und die Gefahr von Oszillationen durch eine ungünstige Initialisierung der beiden wechselwirkenden Kalmanfilter.

4. **SMILE** - Kombination der bisherigen Algorithmen

   Das SMILE-Verfahren (*Shaping Rewards with Multi layered average for Independent Local Reward Estimation*) basiert auf der Beobachtung, dass Maximum- und Mittelwertansatz jeweils an Szenarien scheitern, die das jeweils andere Verfahren problemlos lösen kann (siehe dazu die nachfolgenden Experimente). Daher wurde versucht, die Vorteile beider Ansätze zu kombinieren. Ausführlich untersucht wurde das Verfahren in [Eisenbach, 2009].

   Das Maximumsverfahren operiert immer mit höchsten beobachteten Reward, während der Mittelwert über die Rewards normalerweise unter diesem Wert liegt. Die Idee bei SMILE besteht darin, mit einem Wert zu arbeiten, der zwischen diesen beiden Grenzen liegt.

   Dazu wird zuerst der mittlere globale Reward pro Zustand mittels eines Kalmanfilters geschätzt. Danach werden alle Werte betrachtet, die größer als der Mittelwert sind und über dieser 'besseren' Hälfte der Rewards ein neuer Mittelwert berechnet. Diese Reduktion der relevanten Rewards um die Hälfte kann theoretisch weiter wiederholt werden. Jeder dieser berechneten Mittelwerte erfüllt die

Eigenschaften aus Definition 5.5. Verwendet man keine Mittelung der oberen Hälfte, entspricht dies dem einfachen Mittelwertverfahren. Wiederholt man die Mittelung der jeweils oberen Hälfte der beobachteten Rewardwerte hinreichend oft, verhält sich der geschätzte lokale Reward wie beim Maximumsverfahren, da nach einer Anzahl Halbierungen als der bessere Reward nur noch das Maximum verbleibt.

In den Experimenten wurde immer der Mittelwert über den Werten, die größer sind als das Mittel über alle Rewards, verwendet. Zusätzliche Stufen der Mittelung zeigten kein anderes Verhalten, benötigen allerdings zusätzliche Rechenoperationen.

Nachteilig bei diesem Vorgehen ist, dass hier die beobachteten globalen Rewards gespeichert werden müssen, um die zusätzlichen Mittelwerte der 'besseren' Hälfte ermitteln zu können.

5. **Rewardkombination über ein Gleichungssystem**

Dieses Verfahren aus [Marthi, 2007] unterscheidet sich von den anderen Ansätzen dadurch, dass Kommunikation zwischen den Agenten notwendig ist. Dabei wird die Definition 5.5 direkt umgesetzt, in dem der globale Reward in jedem Schritt mit den Zuständen $s'_1, \ldots, s'_n$ der n Agenten als Gleichung der Form

$$r_{global} = r_{local}\left(s'_1\right) + r_{local}\left(s'_2\right) + \ldots + r_{local}\left(s'_n\right)$$

gespeichert wird. Diese Gleichung besitzt einen skalaren Wert r_{global} und n Unbekannte. Es existieren dabei insgesamt $k = |S_1| + \ldots + |S_n|$ Unbekannte, je eine pro Zustand eines Agenten.

Wenn genügend dieser Gleichungen gesammelt wurden, kann das Gleichungssystem (GLS) im Sinne des minimalen quadratischen Fehlers nach den $r_{local}\left(s'_i\right)$ aufgelöst werden. Man erhält als Lösung

des GLS für jeden Agenten eine Tabelle, in der für jeden beobachteten Zustand der geschätzte lokale Reward steht.

Dazu ist es notwendig, dass an einer Stelle im System die einzelnen Zustände der beteiligten Agenten zusammengeführt werden. Dies war für die bisher vorgestellten Verfahren nicht erforderlich. Auch wird der lokale Reward nicht in jedem Schritt sofort berechnet, sondern sobald genügend (neue) Gleichungen aufgestellt wurden. In den Experimenten wurde nach jeder Episode ein solches GLS gelöst. Bei einer großen Zahl von Gleichungen ist die Lösung des GLS aufwändig, was die Rechenzeit angeht. Daher muss hier eine sinnvolle Obergrenze von Gleichungen definiert werden, und es ist eine Strategie notwendig, alte Gleichungen zu ersetzen, d.h. das Vergessen im Sinne des Stabilitäts-Plastizitäts-Dilemmas muss arrangiert werden. In dieser Arbeit wurde ein möglichst gleichhäufiges Auftreten jeder Zustandsvariablen im Gleichungssystem angestrebt.

5.3.3. Experimente

Die vorgestellten Algorithmen wurden auf unterschiedliche Eigenschaften hin untersucht. Im ersten Experiment wurde die prinzipielle Funktionsweise mit drei Agenten in je einer eigenen 5x5 Gitterwelt (siehe Abbildung 5.6) mit festem Ziel und ohne Rauschen untersucht. Die Resultate in Abbildung 5.8 zeigen einerseits, dass die Verwendung des globalen Rewards nicht zur optimalen Policy führt. Andererseits erreichen alle hier vorgestellten Algorithmen die optimale Handlungsvorschrift. Auffällig ist dabei, dass der Ansatz mit dem GLS genauso schnell zum Ziel kommt, wie wenn die korrekten lokalen Rewards bekannt wären, welche zum Vergleich in einem Test ebenfalls zum Lernen verwendet wurden. Hier können die Stärken, die die Kommunikation einbringt, voll ausgeschöpft werden.

Folgende Fragestellungen wurden mit weiteren Untersuchungen unter-

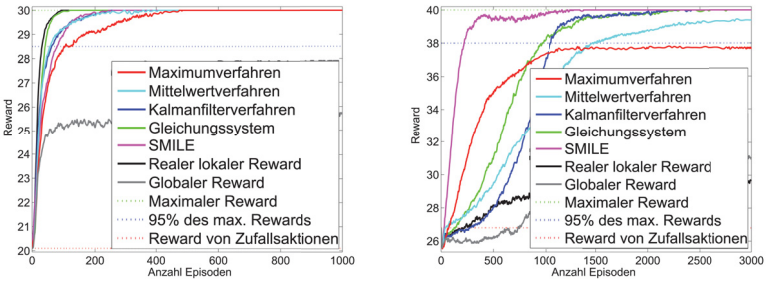


Abbildung 5.8.: Experimentelle Untersuchungen des Rewarddekompositionsproblems. **(Links)** Initiales Szenario mit drei Agenten auf 5x5 Gittern mit festem Ziel. **(Rechts)** Experiment mit 4 Agenten auf dem 5x5 Gitter deren Ziel jeweils von der Position ihres Vorgängers abhängt (siehe Abbildung 5.7).

setzt. Detaillierte Untersuchungen und Diagramme zu allen Ergebnissen finden sich in [Eisenbach, 2009].

- Fragestellung: Skalierung der Algorithmen

  Die Anzahl der Agenten wurde von drei auf bis zu 100 Agenten erhöht. Die Ergebnisse unterschieden sich, von den benötigten Episoden zum Lernen abgesehen, nicht vom Basisszenario. Das Gleichungssystem kann auch hier mit dem unbekannten lokalen Reward konkurrieren.

  Zum Zweiten wurde das Gitter von 5x5 auf die Größe 20x20 erhöht. Hier fällt das Gleichungssystem zurück, da nun wesentlich mehr Gleichungen gesammelt werden müssen, bevor der größere individuelle Zustandsraum für die einzelnen Agenten abgedeckt werden kann. Stattdessen ist hier das Maximumsverfahren in der Lage am schnellsten die beste Lösung zu finden.

- Fragestellung: Störungen in der Rewardfunktion

Dabei wurde einerseits das globale Rewardsignal mit einem normalverteilten Rauschen verschiedener Stärken beaufschlagt. Alle Verfahren außer dem Maximumsansatz konnten das Problem trotzdem noch lösen. Die maximale Störung durch das Rauschen tritt dabei nur sehr selten auf und führt dann zu einer Störung, die der Maximumsansatz nicht mehr ausgleichen kann.

Des Weiteren wurde ein deterministisches Rauschen eingebracht, welches durch einen zusätzlichen, nichtlernenden Agenten mit fester, nichtoptimaler Policy repräsentiert wurde. Hier kamen alle Algorithmen ähnlich schnell zur korrekten Lösung. Das Maximumsverfahren funktioniert hier, da die maximale Störung, anders als bei einer Normalverteilung, eine feste Größe ist, die auch regelmäßig erreicht wird.

- Fragestellung: Änderungen in der Zielposition der Agenten

  Wie bereits in Abbildung 5.7 gezeigt, lag ein Schwerpunkt auf der Frage, wie das System mit Änderungen der Ziele umgehen kann. Diese Frage ist natürlich auch mit den früher in diesem Kapitel diskutierten Themen des Stabilitäts-Plastizitäts-Dilemmas und des Explorations-Exploitations-Dilemmas verknüpft.

  Dazu wurde der Reward betrachtet, der erzielt wurde, wenn die Ziele zufällig über die Zeit wechseln. Dieses Problem konnte vom Maximumsverfahren nicht gelöst werden, da es nicht vergessen kann, was bisher Ziele mit bisherigem Reward waren. Am Ende sind dabei alle Zustände gleich gut, der wahre Reward kann nicht geschätzt werden. Die anderen Ansätze kamen mit dem Problem zurecht, wobei SMILE und der Kalmanfilteransatz sehr gut funktionierten, während der GLS Ansatz nur ein niedrigeres Rewardniveau erreichte. Dies ist darin begründet, dass es eine Weile dauert, bis die aktuellen Zusammenhänge in den Gleichungen hinreichend repräsentiert sind. Dieser Vorgang dauert länger als die Anpassung der Mittelwerte beim Kalmanfilter- oder SMILE-Ansatz.

Für den Fall der veränderlichen Ziele in Abhängigkeit von den anderen Agenten ergibt sich ein anderes Bild. Hier ist es notwendig, dass die Agenten kooperativ zu einem gemeinsamen Ziel finden. Daher ist der wahre lokale Reward allein nicht mehr ausreichend, um eines der Optima zu finden, da es sich bei dieser Aufgabe um ein partiell beobachtbares Problem handelt. Die vorgestellten Verfahren können aufgrund des Rewardstabilitätskriteriums die Problematik abmildern und sind in der Lage eine Lösung zu finden.

Im rechten Teil der Abbildung 5.8 ist der Rewardverlauf gezeigt. Alle Rewarddekompositionsverfahren erreichen ein besseres Ergebnis als unter Verwendung des globalen Rewards oder des realen, nichtbeobachtbaren lokalen Rewards. Jedoch erreicht das Maximumsverfahren nicht das Optimum und auch das Durchschnittsverfahren konvergiert sehr langsam. SMILE, das GLS Verfahren und der Kalmanfilteransatz erreichen die optimale Policy für alle Agenten, wobei SMILE wesentlich schneller zu guten Ergebnissen kommt als die anderen beiden Ansätze.

5.3.4. Fazit

Die Zerlegung in Teilprobleme kann die Lösung komplexer Aufgaben vereinfachen, wenn sichergestellt ist, dass die Teilproblemlösungen auch richtig bewertet werden können. Von den hier untersuchten Algorithmen bieten sich dazu der Gleichungssystemansatz oder bei sich ändernden Zielen und Abhängigkeiten das SMILE-Verfahren besonders an.

Für die reale Anwendung im Szenario der Feuerungsführung verbleiben allerdings offene Probleme. Dies ist einerseits, dass diese Rewarddekompositionsalgorithmen nur online durch Interaktion mit dem Prozess lernen können und viele Interaktionen notwendig sind, um die Zusammenhänge zu lernen, was zeit- und kosten intensiv ist. Zum zweiten verbleibt im realen Prozess die Problematik der Bewertung. Wie in Ka-

pitel 6 noch diskutiert werden wird, ist die Bewertung der Algorithmen eine schwierige und zeitaufwendige Angelegenheit. Im Kontext der Rewarddekomposition fehlen im Kraftwerk, anders als bei dem Gridweltbeispiel, Informationen zu den realen lokalen Rewards. Dies erschwert die Bewertung der Ergebnisse wesentlich, da nicht verifiziert werden kann, ob die gefundene Lösung korrekt ist - und die Bewertung ob die Aufteilung nützlich ist, kann ebenfalls nur am Prozess selbst ermittelt werden.

Daher bleibt zu sagen, dass die hier durchgeführten Untersuchungen klar den Vorteil einer Rewarddekomposition zeigen, für den realen Einsatz in einer kognitiven Architektur ohne Expertenwissen jedoch zurzeit noch nicht geeignet sind. In Kapitel 7 werden hierzu jedoch Überlegungen vorgestellt, welche Erweiterungen notwendig sind, um diesen Teilaspekt sinnvoll in der Gesamtarchitektur zu nutzen.

5.4. Zusammenfassung

In diesem Abschnitt wurde diskutiert, inwieweit es sinnvoll ist, bei einem zyklischen Neutraining von einzelnen Aspekten der Architektur „altes“ Vorwissen einfließen zu lassen. In den Untersuchungen hat sich gezeigt, dass das Einbringen alten Wissens ein zweischneidiges Schwert ist. Solange sicher gestellt ist, dass die Änderungen, die erlernt werden müssen, in der Nähe der alten Lösung liegen, erweist es sich als nützlich, dieses alte Wissen zu verwenden. Sind die Änderung jedoch größer, kann sich das Einbringen des Vorwissens auch negativ auswirken, da unter Umständen ein Verlern- oder Vergessensprozess notwendig ist. In den untersuchten Szenarien hat das Einbringen von Vorwissen sehr ähnliche Ergebnisse erbracht, wie das komplette Erneuern des Wissens.

In diesem Sinne muss abgewogen werden, ob der potentielle Nutzen, Vorwissen einzubringen, größer ist, als der potentielle Schaden, den

dieses Vorgehen anrichten kann. Dafür ist jedoch wieder Vorwissen notwendig, welches rein datengetrieben schwer zu erlangen ist.

Auch die Option, einzelne Komponenten abzuspeichern und bei Bedarf einfach wieder ins Gedächtnis zurückzurufen ohne explizit zu lernen, wurde hier zunächst ausgeklammert, da dafür eine sichere Erkennung und Zuordnung des Systemzustands zum gespeicherten Wissen notwendig ist. Rein datengetrieben ist dies für reale Anwendungen oftmals schwer zu realisieren. Entweder ist ein gutmütiges Problem, bei dem sich die Systemzustände beispielsweise Clustern lassen, notwendig oder aber symbolisches Wissen wird benötigt.

Das Verhältnis von Exploration zum Finden besserer Lösungen und Ausnutzen vorhandenen Wissens zum Erzielen guter Ergebnisse wurde diskutiert. In diesem Zusammenhang wurde ein neuer Algorithmus vorgeschlagen, der im Falle von kontinuierlichen Aktionen eine gezielte Exploration zum Erlangen von neuem Wissen umsetzt. Allerdings muss ganz klar gesagt werden, dass dieser Ansatz noch weiter explorativ entwickelt werden muss, bevor er auch für reale Probleme in Betracht kommt.

Als dritter Schwerpunkt dieses Kapitels wurde das Thema der Rewarddekomposition behandelt. Eine Aufteilung eines Gesamtproblems in einzelne Teilfragestellungen kann das Finden von Lösungen stark vereinfachen und beschleunigen. Allerdings ist dazu notwendig, dass quantifiziert werden kann, welche Teillösung welchen Anteil am Gesamtergebnis hat. Dazu wurden existierende Ansätze aus der Literatur verglichen und in einem neuen Verfahren verschmolzen, um dieses Problem zu lösen.

Die Ergebnisse in diesem Kapitel zeigen an vielen Stellen vielversprechende Ansätze, allerdings die vorgestellten Elemente nicht ohne weiteres in die Gesamtarchitektur zu integrieren. Daher wird in im Kapitel 7 auf Erweiterungen eingegangen, die notwendig sind, um die hier diskutierten Aspekte wirklich behandeln zu können.

6. Intelligente Feuerungsführung

Das Zusammenspiel aller Komponenten, die in den bisherigen Kapiteln vorgestellt wurden, soll nun an einem komplexen und herausfordernden Anwendungsszenario gezeigt werden. Dabei wird etwas näher auf das Anwendungsszenario eingegangen, bevor die konkrete Umsetzung der Teilkomponenten erläutert wird. Vergleichende Untersuchungen und eine Einordnung in den Stand der Technik runden dieses Kapitel ab.

6.1. Anwendungsszenario

Fossile Brennstoffe stellt noch immer eine sehr wichtige Komponente zur Strom- und Wärmeerzeugung in Deutschland dar. Nach Angaben des Bundesministeriums für Wirtschaft und Technologie betrug 2009 der Anteil von Kohle am Energiemix 43,2%, dabei entfallen auf Steinkohle 17,6% und auf Braunkohle 25,6% [Wirtschaftsministerium, 2010]. Auch wenn dieser Anteil rückläufig ist, so wird man auf absehbare Zeit nicht auf Kohle verzichten können.

Im Sinne des Klimaschutzes und den damit verbundenen Klimazielen lohnt es sich, einen genaueren Blick auf die Kohleverbrennung zu werfen. Bei der Verbrennung entstehen an Abgasen primär Kohlendioxid, Schwefeloxide, Stickoxide und Kohlenmonoxid. Letzteres entsteht bei

einer unvollständigen Verbrennung, wenn nicht genug Sauerstoff im Ofen ist um Kohlendioxid zu bilden. Die entstehenden Stickoxide und Schwefeloxide hängen vor allem von der Zusammensetzung der verbrannten Kohle ab und Kohlendioxid ist das unvermeidbare Endprodukt der Verbrennung.

Die Erforschung sogenannter CO_2 freier Kraftwerke[1] steckt noch in den Kinderschuhen und wird frühestens in einigen Jahren oder Jahrzehnten großflächig eingesetzt werden können [Metz et al., 2005].

Allerdings gibt es auch in konventionellen Kraftwerken Möglichkeiten, positiv auf die Verbrennung einzuwirken [Flynn, 2003]. Jedoch werden diese nur unzureichend genutzt, da oftmals nur eine suboptimale Fahrweise des Prozesses mit Hand und PID-Reglern stattfindet und auch an vielen Stellen das notwendige Wissen, wie für ein gegebenes Kraftwerk die optimale Regelungsstrategie aussieht, nicht vorhanden ist. Auch die Verwendung von CFD (Computational Fluid Dynamics) Simulationen hat in der Praxis nur wenig Einfluss. Des Weiteren besteht das Problem, dass viele wichtige Größen des Prozesses nur prozessfern, punktförmig und/oder gar nicht direkt messbar sind.

Hier soll nun gezeigt werden, dass das Problem der Regelung eines industriellen Großkraftwerks mittels eines lernenden Systems, welches eine Implementierung der in dieser Arbeit vorgestellten Architektur ist, angegangen werden kann. Dieses wurde im Rahmen des SOFCOM-Projekts entwickelt, welches in Zusammenarbeit mit der Powitec GmbH, Vattenfall R&D und Vattenfall Heat Hamburg durchgeführt wurde. Alle der hier aufgezeigten praktischen Umsetzungen und Ergebnisse sind in Kooperation mit den Projektpartnern erarbeitet worden.

[1] Bei diesen CCS-Verfahren (Carbon Capture and Storage) wird das Kohlendioxid mit unterschiedlichen, wirkungsgradreduzierenden Ansätzen abgeschieden und muss dann anderweitig, z.B. Untertage, gelagert werden.

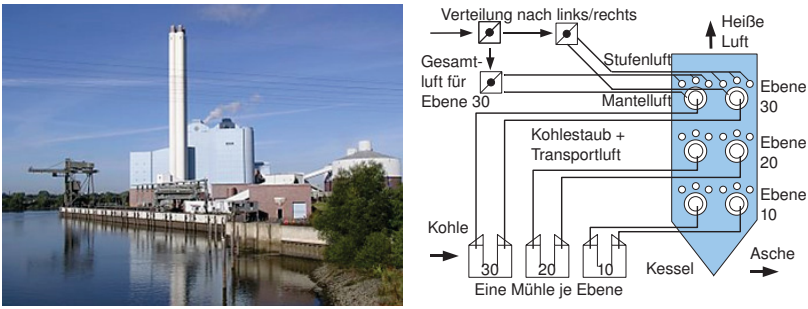


Abbildung 6.1.: Das Kraftwerk Tiefstack in Hamburg. **(Links)** Bild des Kraftwerks. **(Rechts)** schematische Darstellung des Kessels mit den zur Verfügung stehenden Stellgrößen.

Die Anlage

Alle Untersuchungen wurden im Kraftwerk Tiefstack in Hamburg durchgeführt. Dieses Steinkohlekraftwerk aus dem Jahr 1993 dient primär der Grundversorgung mit Fernwärme, wobei es knapp die Hälfte des Bedarfs im Hamburger Fernwärmenetz deckt, und sekundär der Stromerzeugung. In zwei Kesseln mit je sechs Brennern wird Kohle verbrannt, die eine Turbine antreiben. Die maximale Gesamtleistung bei der Wärmeproduktion beträgt 285 Megawatt plus 205 Megawatt Stromerzeugung. Für die Untersuchungen mit dem System basierend auf der kognitiven Architektur wurde einer der beiden Kessel verwendet.

Abbildung 6.1 zeigt das Kraftwerk sowie eine schematische Darstellung des Aufbaus eines Kessels und der Kohlezuführung.

Ziele

Folgende Ziele sollten durch die Regelung mittels des in dieser Arbeit entwickelten lernenden Systems erreicht werden:

1. Verminderung des Schadstoffausstoßes
   Verringerung der Stickoxide um 4-6% und Verringerung des Kohlenmonoxids um 5-10 mg/Nm^3
2. Erhöhung des Wirkungsgrades
   Reduktion des Lambda-Wertes von 1,24 auf unter 1,16. Lambdawerte geben das Verhältnis zwischen der verwendeten Luft und der für eine vollständige (stöchiometrische) Verbrennung notwendigen Luftmenge an und werden als Maß für den Wirkungsgrad genutzt. Zuviel Luft bedeutet, dass die überschüssige Luft unnötigerweise mit erhitzt werden muss, was einer Wirkungsgradreduzierung entspricht. Zu wenig Luft bedeutet erhöhte Korrosionsgefahr des Ofens sowie eine teilweise unvollständige Verbrennung, welche sich im Ausstoß von Kohlenmonoxid widerspiegelt.
3. Erhöhung der Aschequalität
   Verringerung des Anteils von Unverbranntem in der Asche. Liegt dieser Anteil unter eine Schwelle, kann die Asche an die Gipsindustrie verkauft werden, liegt sie darüber muss sie entsorgt werden.
4. Einhaltung sicherheitsrelevanter Grenzwerte
   Weder die Lernprozesse noch die eigentliche Regelung dürfen den Betrieb der Anlage gefährden.
5. Schätzung von Prozessgrößen
   Online-Schätzung von schwer messbaren Größen bzw. dem unter Punkt 3 genannten Unverbranntem in der Asche.

Diese Ziele sind dabei zum Teil konträr zueinander. Eine Verringerung der Gesamtluftmenge erhöht zwar den Wirkungsgrad, gleichzeitig erhöhen sich jedoch die Gefahr der Kohlenmonoxidbildung und die Korrosion der Kesselwand.

Wichtigste Zielgröße ist dabei die Last, also die Auslastung der Turbine. Je nach Nachfrage im lokalen Fernwärmenetz und den Preisen an der Strombörse ergeben sich hier unterschiedliche Anforderungen. Diese

schwanken auf Basis vieler Faktoren z.B. nach Jahreszeit (im Sommer wird weniger Wärme benötigt als im Winter), Wetter (Wind verringert den Strompreis, da Windkraftanlagen dann Strom ins Netz einspeisen können) oder Tageszeit (morgens und in den Abendstunden besteht der höchste Fernwärmebedarf, während er nachts deutlich zurückgeht).

Ein der Schwierigkeiten ergibt sich im Kraftwerk Tiefstack konstruktionsbedingt. Die sechs Brenner pro Kessel sind auf drei Ebenen verteilt. Jede Ebene mit zwei Brennern wird dabei von einer Kohlemühle gespeist. Hinter der Mühle befindet sich ein Y-Rohr, welches die Verteilung auf die zwei Brenner vornimmt, wobei eine 50/50 Verteilung erhofft wird. Aufgrund technischer Randbedingungen ist es nicht möglich, die tatsächlichen Masseströme zu messen. Allerdings zeigen stichprobenartige Untersuchungen und Erfahrungswerte der Anlagenfahrer, dass es hier durchaus zu anderen Verteilungen kommt.

Um diese Ungleichgewichte auszugleichen ist es notwendig die (Sekundär-)Lüfte entsprechend zu regeln. Dieser Zusammenhang ist dem lernenden System nicht bekannt - allerdings sollte sich im gelernten Verhalten des Systems eine entsprechende Luftanpassung zwischen den Brennern einer Ebene widerspiegeln.

Sensorik und Aktuatorik

Jedes Kraftwerk wird durch ein *Distributed Control System* (DCS) geregelt. Dieses hat Zugriff auf Standardsensorik zur Temperatur- und Druckmessung im Kessel sowie Kennzahl zum Dampf, der Turbine und den Mühlenzuflüssen. Es realisiert Stelleingriffe durch die Anlagenfahrer auf der Basis von PID-Reglern. Die wesentlichen Aktuatoren, die hier betrachtet werden, sind dabei die Luftströme. Diese beeinflussen den Verbrennungsprozess wesentlich und werden durch verschiedene Klappen im Ofen manipuliert.

Das hier vorgestellte intelligente System setzt dabei auf dem *Distributed Control System* direkt auf. Alle Stelleingriffe die das kognitive System beschließt, werden als neue Sollgrößen an das DCS weitergereicht. Mittels konventioneller PID-Regler werden diese dann umgesetzt.

Die Realisierung des intelligenten Systems als Erweiterung zum bestehenden System zu betrachten hat zwei Gründe. Einerseits wird so eine einfache Nachrüstbarkeit bestehender Kraftwerke gewährleistet und andererseits dient dies als zweites Sicherheitsnetz. Das heißt, hier können potentiell gefährlich Aktionen einer Instanz des entwickelten Systems hart unterbunden werden.

Als Stellgrößen sind hierbei verschiedene Klappeneinstellungen vorhanden, die die Luftzufuhr im Ofen steuern. Diese werden als Mantel- und Stufenluft bezeichnet, manchmal findet man dafür auch die Begriffe Sekundär- und Tertiärluft. Die Primärluft ist dabei die Transportluft, mit der die Kohle in den Ofen geblasen wird. Dargestellt sind die Stellgrößen exemplarisch in Abbildung 6.1. Damit ergeben sich pro Ebene vier Stellgrößen. Dies sind die Gesamtluftmenge auf der Ebene, die Verteilung zwischen rechtem und linkem Brenner sowie die Verteilung zwischen Stufen- und Mantelluft pro Seite. Damit ergibt sich für den Kessel in Tiefstack ein zwölfdimensionaler, kontinuierlicher Aktionsraum.

Als Besonderheit wurden am Ofen sechs CCD Kameras der Firma Powitec installiert. Diese beobachten direkt jeweils einen Brennermund, jene Zone durch die der Kohlenstaub eingeblasen wird und sich dann entzündet. Das Kamerasystem und der entsprechende Blick in den Ofen sind in Abbildung 6.2 dargestellt. Diese Spezialanfertigungen sind auf den dauerhaften Einsatz in Kraftwerken optimiert. Dazu gehören entsprechende Kühl- und Reinigungssysteme. Neben den Grauwertbildern (siehe auch Abbildung 3.16) liefert die Kamera auch Grauwertspektren die hochfrequent über ausgewählten Bildausschnitten ermittelt werden.

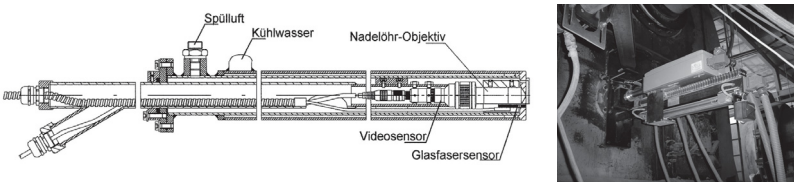


Abbildung 6.2.: Feuerraumlanze der Firma Powitec zur Kameraüberwachung des Kessels. **(Links)** Schematische Darstellung einer Feuerraumlanze. **(Rechts)** Kamerasystem welches an einem Kessel installiert ist. Der größte Teil der Apparatur dient der Kühlung und Reinigung der eigentlichen Kamera.

Randbedingungen

Die durch den Betreiber geforderten Randbedingungen, die einzuhalten waren und, was den Luftanteil angeht, über das DCS erzwungen wurden, sind:

- Der globale Lambdawert für den gesamten Kessel muss immer größer als 1.15 sein.
- Für jeden einzelnen Brenner muss der Lambdawert größer gleich 0.8 sein.
- Das Kohlenmonoxid in der Abluft muss kleiner als 30 mg/Nm^3 sein.

Simulator

Nicht alle notwendigen Untersuchungen und Experimente können am realen Kessel durchgeführt werden, da dies neben sicherheitstechnischen Herausforderungen weder zeitlich noch kostentechnisch beherrschbar wäre. Daher wurde eine Simulatonsumgebung entwickelt, die stark vereinfacht die grundlegende Charakteristik nachbildet. Dieser Simulator basiert auf den Beobachtungen im Kraftwerk und wird im Anhang C.2

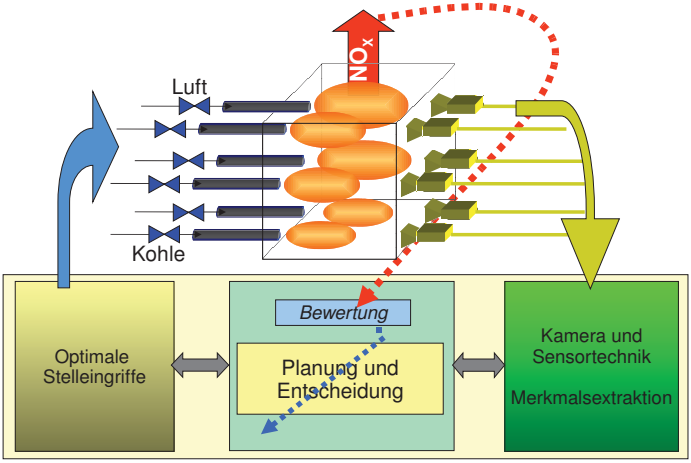


Abbildung 6.3.: Darstellung des Wahrnehmungs-Handlungs-Zyklus im Sinne der Regelung eines Kraftwerks.

beschrieben. Aus dieser Beschreibung kann der geneigte Leser sich ebenfalls ein besseres Bild über die Vorgänge im Kraftwerk machen - auf einige Begriffe und Effekte im Zusammenhang mit Kraftwerken wird dort näher eingegangen.

6.2. Implementierung der Architektur

Als erster Schritt soll der in Kapitel 2 vorgestellte abstrakte Wahrnehmungs-Handlungs-Zyklus auf das Problem der Feuerungsführung übertragen werden. Die geschieht grafisch in Abbildung 6.3. Die drei dargestellten Kernaspekte der Wahrnehmung, Entscheidungsfindung und des Lernmanagements sollen in ihrer konkreten Umsetzung nun näher beleuchtet werden.

6.2.1. Merkmals- und Aktionsauswahl im Kraftwerk

Die Aspekte der Wahrnehmung betreffen hierbei die Kamera und Sensordaten sowie die Aussagen darüber, welche dieser sensorischen Wahrnehmungen Zusammenhänge zu den Zielgrößen zeigen.

Auf der Seite der Aktuatorik kommen, wie in Abschnitt 3.8 beschrieben, die Techniken der Merkmalsauswahl auch auf der Aktionsseite zum Einsatz. Jedoch musste bei den Experimenten und deren Auswertung festgestellt werden, dass keine Aktionsdimensionen ausgeschlossen werden konnten. Alle möglichen Aktionen zeigten deutlichen Einfluss auf den Verbrennungsprozess ohne dabei offensichtlich redundantes Verhalten auszuweisen. Im Rahmen der Aktionsraumselektion konnte daher keine Verringerung erzielt werden.

Auch Untersuchungen zur Aktionsraumtransformation, wie sie ausführlich in der Diplomarbeit von Martin Reinhardt [Reinhardt, 2007] durchgeführt wurden, erbrachten keine nennenswerten Erfolge in Bezug auf das Finden von entkoppelten oder Makroaktionen. Daher wurde im Rahmen des SOFOCM-Projektes der vollständige zwölfdimensionale, kontinuierliche Aktionsraum genutzt.

Für die Merkmalsextraktion wurde das in Abbildung 6.4 dargestellte Schema implementiert. Zuerst werden die Kamerabilder und die Spektren einer Merkmalstransformation unterzogen. Dabei kommt das in Abschnitt 3.7 vorgestellte Verfahren zur Transinformationsmaximierung zum Einsatz. Hierbei werden die hochdimensionalen Bilder und Spektren auf sehr niedrigdimensionale (maximal drei Dimensionen je Zielgröße), informative Kanäle komprimiert. Dabei werden mehrere Zielgrößen verwendet, darunter Stickoxide, Kohlenmonoxid oder der Restsauerstoffgehalt.

Die Berechnung dieser Transformationsmatrizen ist verhältnismäßig aufwendig, was diese Komponente im Sinne eines adaptiven, nachtrainierenden Systems zu einer rechentechnisch teuren Angelegenheit

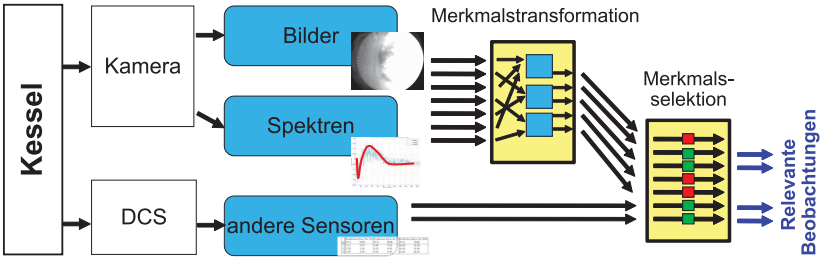


Abbildung 6.4.: Darstellung der im Kraftwerk verwendeten Merkmalsextraktionsverfahren. Die von der Kamera kommenden Bild- und Spektraldaten werden einer Merkmalstransformation unterzogen. Die dabei verwendete Transinformationsmaximierung extrahiert dabei relevante Größen, die informativ in Hinblick auf Zielgrößen, wie die Stickoxide oder den Restsauerstoff sind. Die so extrahierten Merkmale werden danach zusammen mit anderen Messgrößen aus dem Regelsystem des Kraftwerks (DCS) einer einfachen Transinformationsbasierten Merkmalsauswahl unterzogen.

macht. Daher wurde für die nachfolgende Instanz, die Merkmalsselektion, ein vergleichsweise wenig anspruchsvolles Verfahren gewählt. Die transformierten Merkmale aus den Kameradaten werden dazu mit den Daten aus dem DCS kombiniert und einer redundanzberücksichtigenden MIFS Auswahl (siehe Definition 3.22) unterzogen.

6.2.2. Entscheidungsfindung im Kraftwerk

Mit den so ausgewählten Merkmalen wird dann ein Entscheidungssystem gelernt. Neben dem in dieser Arbeit besprochenen Reinforcement Learning Ansatz des CoSyNE (siehe Abschnitt 4.3) wurden zwei andere Verfahren im Kraftwerk zum Einsatz gebracht. Dies ist einerseits ein klassischer Ansatz aus der Regelungstechnik, die Modellprädiktive Regelung, als nichtlineare Variante mit einem Multilayer Perceptron als Modell sowie ein probabilistisches Verfahren basierend auf Faktorgraphen und Bayes'scher Inferenz auf diesen. Diese beiden alternativen

Verfahren sollen hier nur ganz kurz skizziert werden, die angegebenen Quellen können zur Vertiefung genutzt werden. Sie wurden dabei im Rahmen des SOFCOM-Projektes [Funkquist et al., 2011] von den Projektpartnern entwickelt.

Modellprädiktive Regelung

Der Ansatz der modellprädiktiven Regelung (*Model Predictive Control* MPC) [Ogunnaike und Ray, 1994], [Camacho und Bordons Alba, 2004] ist ein weitverbreiteter Ansatz in der Regelung industrieller Prozesse. Dabei wird ein mathematisches Prozessmodell verwendet um die Auswirkungen zukünftiger Aktionen zu simulieren. Basierend auf diesen Simulationen kann dann die beste Aktion ausgewählt werden, die den Prozess in den gewünschten Zustand führt. Welcher Art die verwendeten Modelle dabei sind, ist flexibel. Von klassischen linearen Modellen, über neuronale Netze bis hin zu Gauß'schen Prozessen ist alles möglich.

Für die hier durchgeführten Untersuchungen kamen einerseits ein lineares Modell, genauer gesagt eine Linearisierung um den aktuellen Arbeitspunkt herum, wie auch eine einfache nichtlineare Variante mit einem neuronalen Vorwärtsnetz zum Einsatz.

Hierbei muss jedoch einschränkend gesagt werden, dass für diese nichtlineare MPC Variante nur durch Experten gewählte Merkmale verwendet wurden, eine automatische Selektion der Modellkanäle wurde nicht durchgeführt. Diese Einschränkung wurde gewählt, da dieses System den Stand der Forschung ohne die Erkenntnisse dieser Arbeit und des Projekts darstellt.

Probabilistische Prozessregelung

Eine weitere untersuchte Alternative basiert auf der expliziten Formulierung der Wahrscheinlichkeiten in Form eines graphischen Modells [Bishop, 2006], [Jordan, 1998]. Reale industrielle Prozesse unterliegen oftmals großen Unsicherheiten und sind nur partiell beobachtbar. Mit der expliziten Modellierung der Wahrscheinlichkeiten sollte dieser Tatsache Rechnung getragen werden. In der Praxis besteht der erste Schritt bei diesem Ansatz darin, aus den Beobachtungen Verbundverteilungen aller Zustands-, Aktions- und Zielgrößen zu bestimmen. Natürlich kann dabei keine vollständige Verbundverteilung aller Größen abgeleitet werden, da dies an der hohen Dimensionalität scheitert[2]. Stattdessen wurden mittels Expertenwissen sinnvolle Unterräume zur Berechnung der Wahrscheinlichkeiten ausgewählt.

Mittels beobachteten Zustandsübergängen, ähnlich wie der Datenbasis für das NFQ Verfahren aus Abschnitt 4.1, werden dann Verteilungen geschätzt. Dabei kamen als Repräsentation für die Verteilungen Gauß'sche Mischverteilungen zum Einsatz. Vereinfacht kann man sich vorstellen, dass diese Verteilungen ein Modell formen, das z.B. den Zusammenhang zwischen der Links-Rechts-Luftverteilung und den Stickoxiden darstellt. Basierend darauf kann berechnet werden mit welcher Wahrscheinlichkeit, welche Menge Stickoxide bei einer Aktion zu erwarten ist oder umgekehrt welche Stickoxidemission von welcher Aktion ausgelöst wurde.

Diese Informationen, die in den Verteilungen repräsentiert sind, werden dann über sogenannte Faktorgraphen [Kschischang et al., 2001] verbunden. Mittels Inferenzprozessen basierend auf *message passing* Algorithmen, wie beispielsweise dem *Sum-product* Algorithmus wird dann eine Folge von Stelleingriffen berechnet, die mit der höchsten Wahrscheinlichkeit zum Ziel führen.

[2] Siehe dazu auch die Diskussion in Kapitel 4

Dazu wird ein gewünschtes Endergebnis, z.B. die Emission und der Wirkungsgrad, festgelegt und das System berechnet dann eine Folge von Aktionen, z.B. Luftverteilungen, deren Anwendung mit der höchsten Wahrscheinlichkeit zu diesem Endergebnis führen. Das Modell wird dabei mit aktuellen Sensorbeobachtungen gefüttert und mit neuen Beobachtungen können neue Aktionsfolgen inferiert werden.

Neuroevolutionäre Prozessregelung

Der verwendete Neuroevolutionäre Ansatz Cooperative Synapse Neuroevolution (CoSyNE) wird ausführlich in Abschnitt 4.3 diskutiert. Leider konnte im Rahmen des Projekts nur ein Reinforcement Learning Verfahren im realen Kraftwerk untersucht wurden, auch die Verwendung des NFQ-Algorithmus (siehe Abschnitt 4.1) in einem Kraftwerk[3] wäre sehr interessant gewesen. Die Entscheidung zugunsten des CoSYNE-Algorithmus ist damit zu begründen, das einerseits auch geringfügig bessere Strategien über die Zeit deutliche Auswirkungen auf den Wirkungsgrad und die Emissionen haben. Andererseits steht im Kraftwerk genügend Rechentechnik zur Verfügung, so dass der zusätzliche Rechenaufwand zur Modellbildung und zum Training des Verfahrens nicht übermäßig ins Gewicht fällt.

Visualisiert wird das verwendete System in Abbildung 6.5. Die Bewertung der Population von neuronalen rekurrenten Netzen, wird mittels gelernter Modelle der Zusammenhänge im Kraftwerk durchgeführt. Als Modelle können verschiedene Ansätze genutzt werden, darunter verschiedene neuronale Netze, Gauß'sche Prozesse oder die über die Faktorgraphen repräsentierten graphischen Modelle. Um ein Overfitting auf das Modell zu vermeiden, ist es möglich, die Fitness nicht nur auf einem Modell zu bestimmen, sondern auf mehreren und diese dann zu

[3] Was auch für bestimmte Aspekte in der Müllverbrennung getan wurde [Steege et al., 2010]

kombinieren. Solange die Modelle genug Diversität aufweisen, wird dadurch die Generalisierung verbessert. Jedoch geht dies stark zu Lasten der Rechenzeit, da einerseits zusätzliche Modelle gelernt werden müssen und andererseits die Regler mit den Modellen bewertet werden.

Daher wurde in der Umsetzung in Tiefstack nur eine Sorte von Modellen verwendet. Dabei handelt es sich um rekurrente neuronale Netze, die basierend auf den gemachten Observationen gelernt werden. Diese Netze sind vollverschaltet und verhältnismäßig klein, mit weniger als 20 Hiddenneuronen. Für das Training dieser rekurrenten Modelle werden ebenfalls die im Abschnitt 4.3 besprochenen evolutionären Techniken verwendet.

Mit Hilfe dieser Modelle werden die einzelnen Regler der aktuellen Population bewertet und danach den evolutionären Operatoren unterworfen. Als Größe der Population wurde eine Anzahl von 20 bis 50 Individuen verwendet, wobei eine höhere Anzahl gut für die genetische Vielfalt der Population ist, allerdings auf Kosten der Rechenzeit geht. Die verwendete Reward- bzw. Fitnessfunktion beinhaltet Terme zum Wirkungsgrad, welcher maximiert werden soll, sowie die Emission von Stickoxiden und Kohlenmonoxid, welche minimiert werden sollen. Nach Abschluss der evolutionären Suche wird der aktuell bestbewertete, fitteste Regler dann im Kraftwerk aktiv geschaltet.

6.2.3. Lernmanagment im Kraftwerk

Das Gesamtsystem lief und läuft noch immer rund um die Uhr im Kraftwerk. Aufgrund der sich ergebenden Änderungen im Verbrennungsprozess durch Kohlewechsel, Verschmutzung des Kessels und ähnlicher Probleme, ist es notwendig, dass das Gesamtsystem sich regelmäßig den geänderten Randbedingungen anpasst. Daher werden hier Teile der in Kapitel 5 besprochenen Aspekte des Lernmanagments umgesetzt.

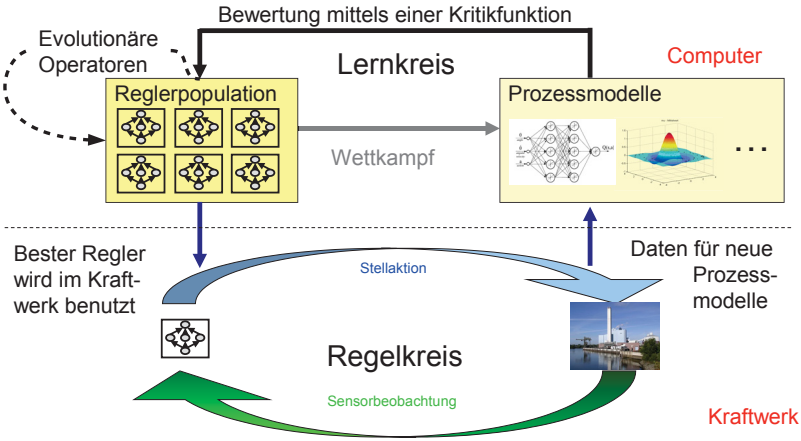


Abbildung 6.5.: Darstellung der im Kraftwerk verwendeten Konfiguration für das CoSYNE Verfahren. Der oben dargestellte Zyklus ist das Training und findet in einem dafür reservierten Rechner statt. Der untere Teil zeigt den Regelkreis im Kraftwerk und die Verbindungen zwischen beiden Systemen.

Es wird dabei kein kontinuierliches Lernen umgesetzt, sondern das System wird in einem Zyklus von 12 Stunden aktualisiert. Während dieser Phase von 12 Stunden werden keine Änderungen am Regler vorgenommen, im Hintergrund finden allerdings rechenaufwendigen Prozesse statt.

Einerseits werden neue Daten gesammelt, indem alle Beobachtungen gespeichert werden und andererseits die einzelnen Teile des Systems nacheinander aktualisiert werden. Dazu werden die gesammelten Daten verwendet. Natürlich kommen dabei nicht alle jemals gemachten Beobachtungen zum Einsatz, sondern nur aus den letzten Tagen und Wochen. Welche Teile aus dem Datenfundus verwendet werden, ist dabei Teil der Merkmalsextraktion.

Die Aktualisierung erfolgt dabei entsprechend des Datenflusses im System. Begonnen wird mit der Merkmalstransformation, welche neu berechnet wird. Dabei werden, wie in Abschnitt 5.1.1 ausgeführt, die letzten Transformationsparameter als Startpunkt verwendet. Danach wird die Auswahl der Merkmale erneuert, bevor die Entscheidungsinstanz neu ermittelt wird.

Basierend auf den neuen Transformationen und ausgewählten Kanälen werden die Daten benutzt, um die Modelle und damit auch die Regler zu aktualisieren. Für die CoSYNE Regler werden komplett neue Populationen in jedem Zyklus verwendet, basierend auf den in Abschnitt 5.1.2 dargestellten Ergebnissen. Am Ende des Zyklus ersetzt der neue Regler, bspw. das beste Netz der CoSYNE-Population, den aktuellen Regler im realen Kraftwerk.

6.3. Untersuchungen

Die Bewertung der Qualität der Merkmalsextraktionsverfahren erweist sich in der praktischen Anwendung als schwierig. Dies hat mehrere Gründe. Einerseits existiert in diesem Zusammenhang nur sehr eingeschränktes Expertenwissen, andererseits kann nur das Gesamtsystem quantitativ bewertet werden. Aussagen, welche Komponenten, welchen Beitrag liefern, sind nur mit extrem aufwändigen Experimenten zu ermitteln. Dies liegt allerdings nicht unbedingt im Sinne des Betreibers, für den die durch das System erzielten Verbesserungen im Vordergrund stehen.

Speziell für die mit der Kamera aufgenommenen Bilder und Spektren gibt nur sehr fundamentales Expertenwissen. Beispielsweise korreliert die Helligkeit der Flamme mit Temperatur. Was jedoch Zusammenhänge zu den Zielgrößen, wie den Stickoxiden oder dem Restsauerstoff angeht, gibt es bislang keine verwertbaren Erkenntnisse.

Bei der Merkmalstransformation der Bilder entsprechen die ermittelten informativen Teile des Bildes der Zone im Ofen, in der der eingeblasene Kohlestaub entzündet wird. Daher erscheint es durchaus sinnvoll, dass hier auch Informationen in Bezug auf die Zielgrößen enthalten sind.

Bei den Spektren ließen sich auch reproduzierbare Filter erzeugen, beispielsweise einen Gaußförmigen Filter im Frequenzbereich der für Stickoxide sein Maximum bei rund 80 Hz hat. Ob es dafür plausible Gründe gibt, konnten die Verfahrenstechniker im Kraftwerk nicht beantworten.

Für die eigentlichen Regler stellte sich für das Modellprädiktive Regelverfahren mit Linearisierung um den Arbeitspunkt bereits auf dem Simulator recht schnell heraus, dass es, ähnlich wie das Reinforcement Learning mit Gauß'schen Prozessen (siehe Abschnitt 4.2) nicht in der Lage ist, das Problem sinnvoll zu behandeln. Daher wurden am Ende im Kraftwerk nur vier Varianten einer ausführlichen Untersuchung unterzogen:

1. Basissystem: Ohne Verbesserungen durch ein spezielles Regelsystem wird hier nur das System des Kraftwerkherstellers eingesetzt. Dazu kommen auch händische Einstellungen der Anlagenfahrer. Diese sind allerdings gerade was die Lufteinstellungen angeht sehr selten. Es handelt sich damit um den Standardkraftwerkbetrieb und ist die Vergleichsgrundlage für die anderen Verfahren.

2. Modellprädiktives Regelsystem (MPC): Dieses basiert auf durch Experten gewählten Eingangskanälen und einem Multilayer-Perceptron als neuronales Netz für einen nichtlinearen modellprädiktiven Ansatz. Das Modell wird regelmäßig nach trainiert.

3. Vorgestelltes System mit automatischer Merkmalsextraktion und dem CoSYNE Neuroevolutionsverfahren als Regler: Die Modelle zur Ermittlung der Fitnessfunktion werden regelmäßig nachtrainiert. Dieses System ist in der Durchführungsphase sehr schnell, benötigt allerdings viel Trainingszeit.

4. Vorgestelltes System mit automatischer Merkmalsextraktion und dem probabilistischen Ansatz über Faktorgraphen als Regler: Die gesammelten Daten werden genutzt, um regelmäßig die Verteilungen zu aktualisieren, auf deren Basis die Stellgrößen inferiert werden. Der Inferenzprozess ist vergleichsweise langsam und begrenzt die Zahl der Stelleingriffe auf einen pro Minute. Allerdings entfällt der Trainingsaufwand für den Regler selbst.

Diese vier Alternativen wurden ausführlichen Untersuchungen im Kraftwerk unterzogen.

Das Durchführen von Experimenten im realen Kraftwerk und vielmehr das sinnvolle Auswerten der Ergebnisse stellt eine große Herausforderung dar. Dies liegt daran, dass für die einzelnen Regler nie gleiche Randbedingungen geschaffen werden können. Durch den Tagesbetrieb ist es unmöglich, die gleichen Lastverhältnisse und Kohlesorten über den notwendigen Zeitraum zu garantieren. Daher besteht nur die Chance, über einen hinreichend großen Zeitraum sicherzustellen, dass alle Regler möglichst ähnliche Randbedingungen beobachtet und zu regeln hatten.

Die Experimente wurden im ganz normalen Betrieb des Kraftwerks durchgeführt. Dabei traten regelmäßig Lastwechsel auf, und es ergaben sich Wechsel in der Kohlesorte. Jedes der Verfahren wurde für eine Zeitscheibe von 10 Stunden aktiviert, danach kam das nächste Verfahren für 10 Stunden an die Reihe. Nach dem Wechsel des Regelansatzes wurden die ersten 30 Minuten aus der Betrachtung ausgeschlossen um Prozessänderungen, die durch den Reglerwechsel entstehen, auszuschließen. Ebenfalls ausgeschlossen wurden je nach Verfahren eventuelle Explorationszeiten, welche maximal weitere anderthalb Stunden ausmachten.

Alle anormalen Betriebszustände, Zeiten in denen Anlagenfahrer das System deaktiviert hatten oder die Lastanforderung weniger als 30 Prozent betrug, wurden für die Bewertung gleichfalls ignoriert. Um eine Vergleichbarkeit der Daten zu garantieren, wurden alle Vergleiche für

einzelne Kohlesorte ausgewertet. Durch die Definition von Lastklassen und Klassen für das Luftbrennstoffverhältnis wurde der Einfluss verschiedener Lastanforderungen minimiert.

In einem ersten, zweiwöchigen Test wurden die Systeme ohne Adapitivität untersucht. Dabei wurde die generelle Anwendbarkeit der Ansätze nachgewiesen und verbliebene Sicherheitsbedenken der Betreiber zerstreut. Die Testphase für das vollständige System dauert dann über mehrere Monate an, unterbrochen von einer Revision des Ofens. Bei einer solchen Revision, welche typischerweise einmal jährlich stattfindet, wird der Kessel komplett gesäubert, was zu drastisch anderen Eigenschaften führt.

Pro Kohlesorte wurde ausgewertet, welche Auswirkungen die untersuchten Systeme auf die Stickoxide, den Wirkungsgrad, welcher im Restsauerstoff repräsentiert ist, und den Kohlenmonoxidausstoß haben. Für eine Kohlesorte sind die Stickoxidemissionen beispielhaft in Abbildung 6.6 dargestellt. Gleiche Auswertungen wurden für 13 Kohlesorten durchgeführt. In gleicher Weise fand dies auch mit den interessanten Größen Kohlenmonoxid und Restsauerstoff statt. Für detaillierte Betrachtungen sei auf [Funkquist et al., 2009] und [Funkquist et al., 2011] verwiesen.

Ohne hier auf die Ergebnisse für einzelne Kohlesorten oder Prozesszustände eingehen zu wollen, wurden die Ergebnisse in Abbildung 6.7 zusammengefasst. Dazu wurden die einzelnen Einsparungen gewichtet nach der beobachteten Zeit gemittelt.

Das erzeugte Kohlenmonoxid liegt in allen Fällen deutlich unter den gesetzlichen Grenzwerten und schwankt auch nur minimal (unter einem Promille). Es stehen im Vergleich zum ungeregelten System gelegentliche Kohlenmonoxidspitzen, die durch die starke Verringerung des Sauerstoffs bei schnellen Wechseln nicht sofort ausgeglichen werden können. Keines der Systeme stellt liegt hier außerhalb der Vorgaben.

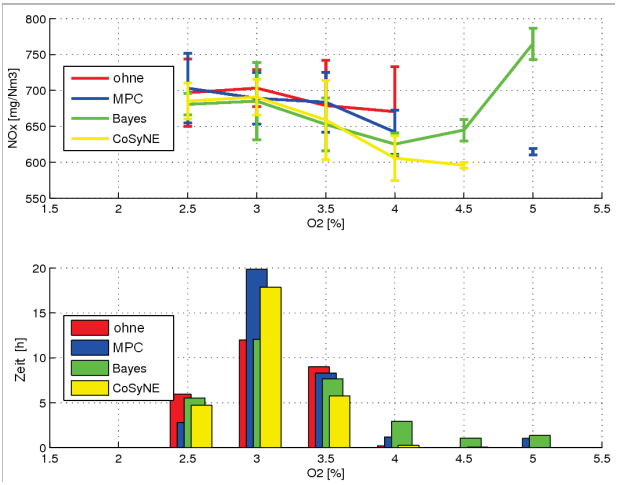


Abbildung 6.6.: Darstellung des Stickoxids bei der Verbrennung eines einzelnen Kohletyps. Im oberen Diagramm ist der Stickoxidgehalt gegenüber den Restsauerstoffklassen aufgetragen. Darunter ist die Zeit aufgetragen, die der Prozess in den einzelnen Zuständen verbrachte. Man kann so beispielsweise erkennen, dass ein Restsauerstoffgehalt von mehr als 4 Prozent nur sehr selten erreicht wurde, und daher die Aussagen mit deutlich größerer Unsicherheit behaftet sind. Man kann erkennen, dass die Stickoxidproduktion für das Reinforcement Learning System (CoSYNE) und das probabilistische System (Bayes) deutlich unter dem ungeregelten Fall liegt. Der modellprädiktive Ansatz (MPC) fällt hier hingegen zurück.

Beim erzeugten Stickoxid stellt sich ein anderes Bild dar. Im Vergleich zum ungeregelten Szenario können alle drei Systeme eine Verringerung erzielen. Der Stickoxidanteil ist dabei extrem vom Kohletyp abhängig. Je nach Sorte schwankt der Ausstoß zwischen 400 mg/Nm^3 und 1100 mg/Nm^3. Im Falle von geringeren Konzentrationen ist der Gewinn durch das System sehr gering (wenige mg/Nm^3), bei hohen Konzentration kann die Reduktion je nach Sauerstoffgehalt auch mehr als 100 mg/Nm^3 betragen, der Einfluss des Sauerstoffs wurde bei dieser

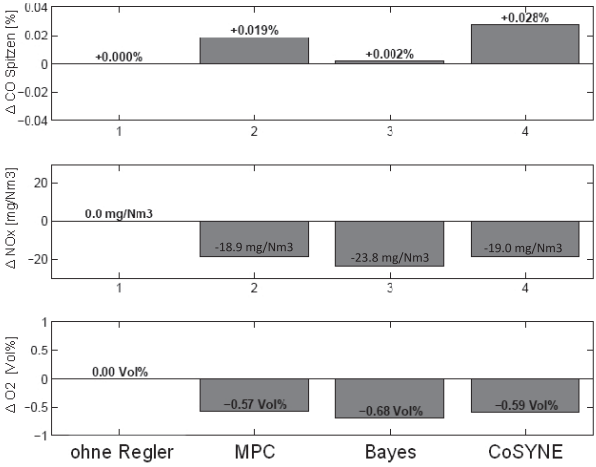


Abbildung 6.7.: Darstellung der Verbesserungen der Systeme im Vergleich zum ungeregelten Fall. Oben ist der Anstieg des Kohlenmonoxids dargestellt. Die Mitte zeigt die Reduktion der Stickoxide, während ganz unten die Reduktion des Restsauerstoffs dargestellt ist. Die Reduktion des Restsauerstoffs entspricht dabei einer Wirkungsgradsteigerung.

Betrachtung bewusst heraus gerechnet. Daher liegt die reale Stickoxidverminderung höher als die hier dargestellten, gewichtet gemittelten 20 mg/Nm^3. Das Reinforcement Learning System liegt dabei gleich auf mit dem MPC Ansatz, während der probabilistische Regler noch einmal signifikant besser ist.

Für die wichtigste Größe ergibt sich ein ähnliches Bild. Die Restsauerstoffreduktion und damit die Wirkungsgradsteigerung gelingen dem probabilistischen Regler am besten. Die Einsparungen des CoSYNE-Systems sind geringer, dicht gefolgt vom MPC Ansatz.

Die erzielten Ergebnisse zeigen, dass das Bayessystem mit dem probabilistischen Verfahren die konsistentesten Ergebnisse erzielt wurden. Daher findet dieses Verfahren mittlerweile Daueranwendung im Kraft-

werk Tiefstack. Betrachtet man die Ergebnisse genauer, wird klar, dass das CoSYNE-Verfahren nicht strikt schlechter ist, sondern deutlicher schwankt.

So zeigten Untersuchungen, dass bei manchen Kohlesorten das neuroevolutionäre System nicht wesentlich besser war, als der ungeregelte Zustand, es bei anderen allerdings das Bayessystem um mehr als das Doppelte übertraf. Die Verbesserungen, die für die einzelnen Kohlesorten erzielt wurden, schwankten zum Teil sehr stark. Worauf diese Schwankungen zurückzuführen sind, konnte nicht abschließend geklärt werden. Jedoch liegt die Vermutung nahe, dass das Problem nicht bei der evolutionären Optimierung des Reglers selbst zu suchen ist, sondern in den Modellen, die zur Bewertung der Regler eingesetzt werden. Die starke rechentechnische Beanspruchung durch dieses Verfahren, machte es nicht möglich, hier weitergehende Ansätze, wie die Mittelung über mehrere Modelle, im Kraftwerk umzusetzen.

Der probabilistische Regler kommt relativ konsistent zu Verbesserungen des Verbrennungsprozesses unabhängig von der Kohlesorte. Auch der modellprädiktive Ansatz weißt diese Konsistenz auf, ist allerdings in den Untersuchungen immer schlechter als das probabilistische System.

Auch hier lässt sich wieder das Bias-Varianz-Dilemma als Interpretation einbringen. Das probabilistische System entspricht dabei einem höheren Bias. Die Varianz der Ergebnisse ist gering und er erreicht nicht immer die besten Ergebnisse. Der Gegenpol dazu ist das neuroevolutionäre Verfahren, welches sehr unterschiedliche Ergebnisse erreicht, und dabei den Bayes'schen Ansatz zum Teil deutlich übertrifft. Ursache dafür sind die große Zahl freier Parameter die sowohl im Regler selbst als auch in den zum Training verwendeten Modellen zu finden sind. Die Ergebnisse zeigen, dass alles Für oder Wider zusammengenommen, der probabilistische Ansatz die besseren Generalisierungseigenschaften aufweist.

6.4. Einordnung

Das hier entwickelte System wurde bereits in verschiedenen Beiträgen vorgestellt: [Rosner et al., 2008], [Schaffernicht et al., 2009b], [Funkquist et al., 2009] und [Funkquist et al., 2011]. Es handelt sich dabei um eine Weiterentwicklung von anderen lernenden Ansätzen zur intelligenten Feuerungsführung, wie sie in [Stephan et al., 2001] und [Stephan et al., 2004] vorgestellt werden.

Das Alleinstellungsmerkmal des hier vorgestellten Systems ist, dass es im Dauereinsatz ein kommerziell genutztes, mittelgroßes Kraftwerk erfolgreich regelt.

Aus der Sicht des maschinellen Lernens konnte gezeigt werden, dass die Ansätze in der Lage sind, ein solch herausforderndes Problem wie die Regelung eines Verbrennungsprozesses zu bewältigen und dabei nicht nur den Wirkungsgrad zu verbessern und damit den Schadstoffausstoß zu verringern, sondern auch das Wissen der menschlichen Experten erweitern kann.

Es gibt in der Literatur nur sehr wenige Arbeiten, die sich mit diesem Szenario und der Anwendung beschäftigen. Und jene die es tun gehen nur sehr selten über Simulationen oder offline Anwendungen hinaus. Beim Lesen der Quellen ist teilweise Vorsicht angebracht, was die Verwendung verschiedener Begrifflichkeiten angeht. Da hier Verfahrenstechniker, Regelungstechniker, Informatiker und andere unterschiedliche Formulierungen nutzen oder dieselben Worte unterschiedliche Bedeutungen in unterschiedlichen Zünften haben[4].

In [Grancharova et al., 2008] wird mittels Gauß'scher Prozesse ein Prozessmodell für die Verbrennung eines Kohleofens gelernt und mittels MPC zum Regeln eines simulierten Kessels genutzt. In

[4] Sehr häufig ist von „intelligenten Systemen“ die Rede, wenn auch nur ein Fuzzy-Regler oder ein neuronales Netz verwendet wird. Das bedeutet nicht, dass dort adaptive oder selbstorganisierende Komponenten Verwendung finden.

[Mühlhaus et al., 1999] wird ein neuronales Prozessmodell für die Prognose von Stickoxiden diskutiert. Dazu werden mittels Expertenwissen Eingabegrößen definiert und mittels statischer Größen im Sinne einer Merkmalsselektion angepasst. Basierend auf dem invertierten Modell[5] wurden dann offline Regeln extrahiert, die die Regeln verbessern sollten. Es wird von vielen Schwierigkeiten berichtet, die sich auf den Arbeitspunkt des Prozesses, nichtbeobachtbare Größen, wie die Kohlequalität, und ähnliches beziehen. Die erzielten Ergebnisse werden nicht quantifiziert.

Auch bieten Firmen, wie ABB[6] oder Rockwell Automation[7], kommerzielle Systeme zur Regelung von Kraftwerken an. Allerdings existieren dazu kaum wissenschaftliche Veröffentlichungen. Aus den diversen Broschüren kann, jedoch ohne die Details zu kennen, entnommen werden, dass die Standardsysteme entweder mit klassischen PID-Reglern, Fuzzy-Reglern oder als modernste Variante mit Modellprädiktiven Reglern arbeiten.

Einzig im Feld der Flammenbildverarbeitung gibt es eine Zahl an Publikationen, die versuchen Informationen aus Kameraaufnahmen von Flammen zu ziehen [Docquier und Candel, 2002]. Dazu kommen typischerweise spezielle System, wie Farbpyrometrie [Lu et al., 2005] [Zipser et al., 2006] oder Infrarotkameras [Marques und Jorge, 2000] [Cignoli et al., 2001] zum Einsatz. Die Flammenformanalyse [Bastiaans et al., 2005] ist nach wie vor nicht in der Lage Zustände des Verbrennungsprozesses zu beschreiben. Daher gibt es ebenfalls Veröffentlichungen die die Verwendung von Eigenflames [Stephan et al., 2001] [Schmid et al., 2006] propagieren.

Wesentlich mehr Publikationen sind in verwandten Gebieten zu finden. Dazu zählen die Müllverbrennung, die Zementherstellung und die

[5] Es ist nicht klar, wie das Modell invertiert wurde oder welche Struktur das Netz aufweist.

[6] http://www.abb.de/

[7] http://www.rockwellautomation.com/solutions/combustioncontrol/

Papier- und Pappherstellung, welche sich alle mit Verbrennungsprozessen in großen Öfen beschäftigen. Die Herausforderungen in diesen Feldern sind sehr ähnlich zu denen in einem Kohlekraftwerk. Auch dort stellen Modellprädiktive Ansätze den Stand der Technik dar, in [Stadler et al., 2011] wird ein aktuelles System für Zementwerke vorgestellt. Die Regelung einzelner Komponenten wird diskutiert, beispielsweise die Modellierung der Mühlen mit neuronalen Netzen [Topalov und Kaynak, 2004] oder Fuzzy-Regler für die Roste [Wardana, 2004]. Eine grundlegende Übersicht für die Müllverbrennung wird in [Görner, 2003] gegeben, viele Untersuchungen mit neuronalen Netzen als Zustandsschätzer und als MPC Modellkomponente findet man in [Müller, 2000].

Aus Sicht der Kraftwerkstechnik ist das in dieser Arbeit vorgestellte System mit seiner automatischen Merkmalsextraktion und adaptiven Regelung eines der fortschrittlichsten Regelungssysteme für Kraftwerke zur Schadstoffminderung und Effizienzsteigerung, welches vergleichsweise einfach in existierende Anlagen integriert werden kann und adaptiv eine saubere Verbrennung in Steinkohlekesseln erzielt.

6.5. Fazit

Ein kognitives, datengetriebenes Regelungssystem, welches zweimal täglich Adaptionszyklen vornimmt, wurde im Hamburger Steinkohlekraftwerk Tiefstack implementiert, untersucht und befindet sich seitdem im Dauereinsatz.

Das Gesamtsystem erzielt durch die Verwendung der adaptiven Ansätze dieselbe Leistung mit wesentlich weniger Kohle und bei einem geringeren Schadstoffausstoß im Vergleich zur konventionellen Regelung. Für einen Kessel in Tiefstack erzielt das System durch Effekte, wie geringeren Restsauerstoff, weniger Gebläseeinsatz und verringertem Sprühwasserbedarf, eine Gesamtersparnis von rund 1800 Tonnen Kohle pro Jahr.

Dies entspricht etwa 4500 Tonnen Kohlendioxid, die weniger freigesetzt werden. Dazu kommen weitere, schwer zu quantifizierende Effekte, wie die Möglichkeit den Kessel länger unter Volllast zu betreiben bevor eine Revision notwendig wird, die eine weitere indirekte Effizienzsteigerung darstellen.

Als solches konnte gezeigt werden, dass die in dieser Arbeit diskutierten Methoden und Strukturen eines kognitiven Systems in der Lage sind, ein komplexes Problem, wie die Steinkohleverbrennung, dauerhaft zu regeln und dadurch bessere Ergebnisse zu erzielen als alle bisherigen im Einsatz befindlichen Systeme zur Feuerungsführung.

7. Erweiterung der Architektur

Im Kapitel 6 konnte gezeigt werden, dass die in dieser Arbeit vorgestellten Methoden in der Lage sind, ein solch komplexes Problem, wie die Regelung eines industriellen Verbrennungsofens, erfolgreich zu lösen. Trotzdem verbleiben Probleme und die Frage, ob es nicht noch besser ginge. Natürlich bietet diese Arbeit, speziell wurde es bereits im Kapitel 5 angesprochen, etliche lose Enden an denen neue Entwicklungen sich anschließen können und müssen.

Dieses Kapitel soll genutzt werden, um zu skizzieren, welche Elemente in einer erweiterten Version einer solchen kognitiven Architektur Eingang finden müssen, um einen wesentlichen Sprung vorwärts zu machen.

Von den drei auf dem Wahrnehmungs-Handlungs-Zyklus basierenden Kerngebieten dieser Arbeit zu Fragen der Wahrnehmung (Kapitel 3), der Entscheidungsfindung (Kapitel 4) und dem Lernmanagement (Kapitel 5), fällt dem letztgenannten vermutlich das größte Potential zu. Doch sollen vorher kurz die beiden anderen Aspekte diskutiert werden.

Im Bereich der Merkmalsextraktion gibt es seit wenigen Jahren verstärkt den Drang zum Finden kausaler Abhängigkeiten, [Aliferis et al., 2010] gibt hier einen Überblick. Alle in dieser Arbeit diskutierten Ansätze basieren auf verschiedenen statistischen Abhängigkeiten, sagen aber nichts über Ursache und Wirkung aus. Wenn es gelingt, Variablen zu identifizieren, die kausal die Ursache für andere Variablen und Zielgrößen sind, ist diese Information bedeutender als die Relevanz und die Nützlichkeit. Wenn in der nächsten Zeit Verfahren entwickelt werden, die diese kausalen Abhängigkeiten erkennen

können, erschließen sich dadurch vollkommen neue Wege, was die Wahrnehmung und Modellbildung in einer kognitiven Architektur angehen, da dadurch Wissen über Ursache und Wirkung von Entscheidungen in den Lernprozessen genutzt werden kann.

Für die datenbasierte Entscheidungsfindung wird es auch in Zukunft weitere interessante Ansätze im Bereich des Reinforcement Learnings, in der probabilistischen Modellierung und der Regelungstechnik geben. Allerdings sollten sich diese relativ einfach in die vorhandene Struktur integrieren lassen und keine wesentlichen Änderungen in der Architektur erfordern. Auch ist das Potential für wirkliche Verbesserungen in diesem Kern eher gering, da fragwürdig ist, wie viel besser beispielsweise ein neues Reinforcement Learning Verfahren sein würde. Die Beschränkung liegt weniger in den Lernverfahren selbst, als vielmehr im intelligenten Management des Lernens.

Im Kapitel 5 wurden dazu zwar wichtige Aspekte beleuchtet, allerdings bleiben aus den Untersuchungen Fragen offen. Die hier vorgeschlagene Erweiterung der Architektur führt dazu einen zweiten Wahrnehmungs-Handlungs-Zyklus ein, dessen Aufgabe die Verbesserung des regelnden Systems ist. Dieser gruppiert sich dabei um die zentrale Idee einer Prozesskarte. Der bisher in dieser Arbeit diskutierte intelligente Regler wird im Weiteren als Regel-Zyklus bezeichnet, die Erweiterung als Management-Zyklus.

Die notwendigen Elemente einer erweiterten Architektur sind in Abbildung 7.1 gezeigt. Die Pfeile in dieser Grafik stellen den Datenfluss in der Architektur dar. Die Rauten symbolisieren eine Kontrolle oder Manipulation eines Blocks durch eine Managementfunktion. Im oberen linken Bereich ist der Wahrnehmungs-Handlungs-Zyklus zu sehen, welcher den Prozess regelt und an vielen Stellen dieser Arbeit ausführlicher beleuchtet wurde. Neu ist die zweite Ebene, welche sich mit der Organisation des Lernens und der Kopplung zum menschlichen Nutzer beschäftigt. Dazu existiert ein zweites Wahrnehmungsmodul, welches durchaus Er-

kenntnisse aus der Wahrnehmung des Prozessregelkreises nutzen kann, aber einen anderen Fokus hat und beispielsweise die Auswirkungen der Regelung beachten muss. Die Informationen können dann genutzt werden, um den Prozesszustand zu identifizieren und zu kategorisieren. Diese Information kann dann auf der zentralen Prozesskarte eingetragen werden, zusammen mit Informationen über die zur Regelung verwendeten Algorithmen. Die mit Wissenspflege markierte Instanz dient dazu, die Karte im Sinne eines lebenslangen Lernens zu pflegen und muss entscheiden, welches Wissen wie gespeichert wird und was vergessen werden kann. Schließlich existiert auch hier eine Handlungsinstanz, deren Aufgabe das Training der Elemente im Regel-Zyklus zu koordinieren. Zusätzlich wird ein Modul benötigt, welches die Kommunikation zwischen dem System und dem Nutzer ermöglicht. Einerseits wird dabei der Prozesszustand mit Hilfe der Karte charakterisiert und dem Nutzer nahe gebracht. Andererseits muss es die Eingaben des Menschen interpretieren und ggf. nutzen, um den Prozess mit diesem Zusatzwissen besser charakterisieren zu können.

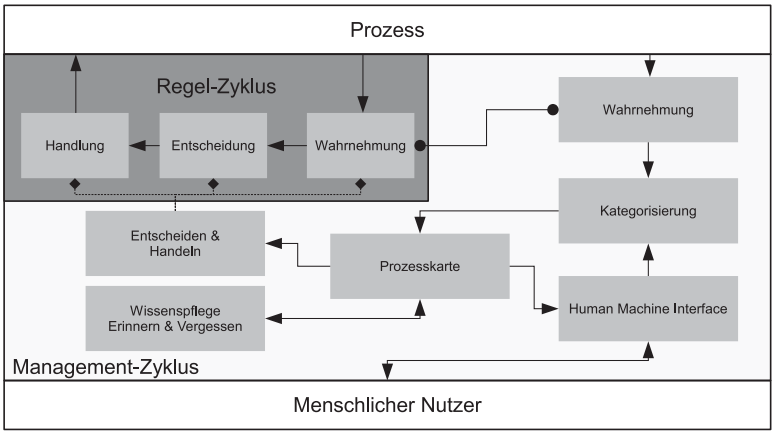


Abbildung 7.1.: Die hier dargestellte erweiterte Architektur wird im Text ausführlich erläutert und diskutiert.

Bezogen auf die in Kapitel 2 benannten Eigenschaften kognitiver Architekturen, sollte dieser zweite Zyklus zum Management des Lernens folgendes leisten:

- **Wahrnehmung, Erfassung, Kategorisierung und Situationseinschätzung**

  Der Prozess muss grob kategorisiert werden. Das heißt, auf einer langsameren Zeitskala als die der eigentlichen Regelung, muss versucht werden, veränderliche Randbedingungen zu erfassen. Dazu werden auch hier Sensorbeobachtungen benutzt, allerdings nicht direkte zur Regelung, sondern zur Identifikation des übergeordneten Prozesszustandes beispielsweise im Sinne des Arbeitspunktes.

  Des Weiteren ist es notwendig, dass das System in der Lage ist, diese Prozesszustände in Relation zueinander zu bringen. Diese können temporaler Natur sein (Welcher Prozesszustand folgt am wahrscheinlichstem dem jetzigen Zustand?) oder auf Ähnlichkeiten basierend (Im welchem Zustand reagiert der Prozess auf Regeleingriffe auf sehr ähnliche Art und Weise?).

  Die Idee besteht darin, eine Art Karte des Prozesses zu entwickeln, die solche Relationen kodiert. Typischerweise wird dies eine topologische, graphbasierte Karte sein, da je nach Art der Relationen eine Metrik schwierig zu finden oder zu lernen sein wird. An den einzelnen Prozesszuständen auf einer solchen Karte könnten dann Informationen über die verwendeten Algorithmen und Regler hängen, die bisher in diesem oder ähnlichen Zuständen die besten Ergebnisse erzielt haben.

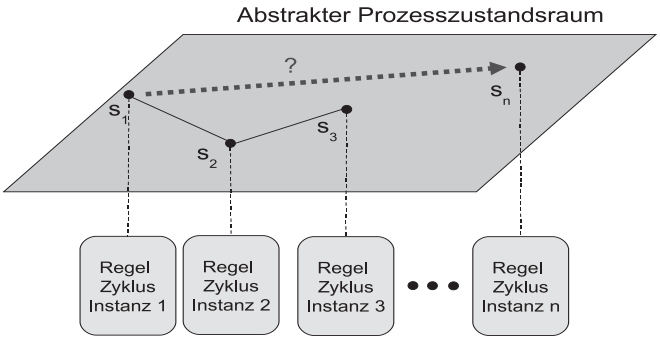


Abbildung 7.2.: Prozesskarte zur Organisation des Wissens und Lernens. Der abstrakte Prozesszustandsraum wird dabei durch ein vorgelagertes System (Wahrnehmung/Kategorisierung) aufgespannt. In diesem werden einzelne Prozesszustände ablegt und mit Information zu den in diesem Zustand verwendeten Regel-Zyklen versehen. Dazu zählen beispielsweise die verwendeten Merkmale und Merkmalstransformationen, die neuronalen Netze für die Entscheidungsfindung oder auch eine Bewertung der Leistung des Systems. Die zu lösenden Fragen sind dabei: Wie kann mit dieser Karte navigiert werden? Bestimmte Zustände sind möglicherweise wünschenswerter als andere. Wie kann eine solche Karte (kontinuierlich) gelernt werden? Wie können Information benachbarter Zustände wiederverwendet werden? Wie kann auf Basis der Informationen der Karte und der Position auf ihr, auf die durchzuführenden Aktionen geschlossen werden?

Eine solche Prozesskarte wird in Abbildung 7.2 veranschaulicht und erläutert.

Zusätzlich ist es im Sinne der Situationseinschätzung notwendig, dass die Leistung des eigentlichen Regelsystems überwacht wird. Es muss eine automatisierte Bewertung der Leistung des momentan verwendeten Regel-Zyklus durch die Management-Instanz erfolgen können.

- **Vorhersage und Überwachung**

  Im Optimalfall erkennt das System, wo auf der Prozesskarte man sich befindet, und es kann auf Basis der Karte Vorhersagen machen, wie der Prozess sich weiterentwickeln könnte. Basierend auf dieser Information muss dann bewertet werden, ob die aktuelle eingesetzt Instanz des Regel-Zyklus zufriedenstellende Ergebnisse erzielt und ob dies auch in der Zukunft der Fall sein wird.

- **Problemlösen, Planen, Entscheiden und Wählen**

  Die wesentlichen Entscheidungen, die zu treffen sind, betreffen den Regel-Zyklus. Falls die Leistung eines Reglers sich verschlechtert, ist zu entscheiden, was getan werden muss. Optionen beinhalten das Neutraining des Regelsystems, des Ersetzen des Regelsystems durch eine andere Instanz, die aufgrund der Karteninformationen als besser geeignet erscheint, um mit dem momentanen Zustand umzugehen, oder das Erlernen eines vollkommen neuen Reglers. Ebenfalls von Bedeutung ist die Frage, was mit dem alten Regler zu tun ist. Soll dieser gelöscht werden oder enthält er wichtige Information, die weitergenutzt werden können. Falls dem so ist, kann der Regler abgespeichert und wiederverwendet werden oder einem Informationspool hinzugefügt werden, welcher in Form von Vorwissen beim Training neuer Regler verwendet werden kann.

- **Ausführung und Aktion**

  Der Management-Zyklus greift nicht selbst auf den zu regelnden Prozess zu, sondern alle Aktionen beeinflussen die Komponenten des Regel-Zyklus. Konkrete Aktionen wären dabei das Ein- und Ausschalten von Komponenten, das Austauschen von Teilen oder das Anstoßen eines Adaptionsvorgangs unter ausgewählten Parametern (Auswahl der Trainingsbeispiele, Auswahl des Algorithmus zum Lernen, der Explorationsstrategie usw.).

- **Erinnern und Lernen**

  Da eine Kategorisierung des Prozesses selten durch Expertenwissen umfassend realisierbar ist, muss die Karte mit ihren Elementen gelernt werden. Dadurch können neue, unbekannte Prozesszustände erfasst werden. Auch ist eine sinnvolle Strukturierung der Karte von Aufgabe zu Aufgabe unterschiedlich zu wählen. Wichtig ist, dass an dieser Stelle auch das Wissen strukturiert werden muss. Somit sind Operationen auf dieser Prozesskarte notwendig, die es erlauben, Orte zusammen zufassen oder auch zu vergessen, wenn sich Informationen als redundant oder unnütz erweist.

  Einen anderen Aspekt, der nicht zwingend mit einer solchen Prozesskarte verknüpft ist, stellt die automatische Problemdekomposition dar. Ziel ist dabei das Gesamtproblem automatisch in kleinere Teilprobleme zu zerlegen. Die Lösungen für die einzelnen Teilprobleme lassen sich einfacher und schneller Finden als für das komplexe Gesamtproblem (siehe Abschnitt 5.3). Mit dem Wissen über die Beziehungen der einzelnen Teile zueinander kann dann aus den einzelnen Teillösungen eine Gesamtlösung formuliert werden. Mögliche Ansätze solche Zerlegungen zu finden, umfassen einerseits die in Kapitel 3 beschriebenen Methoden zur Transinformation, die oben benannten kausalen Abhängigkeiten (z.B. Granger-Kausalität [Granger, 1969]), ICA basierte Ansätze [Hyvärinen et al., 2010] oder evolutionäre Ansätze [Khare et al., 2005]. Dass sich solche zerlegten Probleme auch bei ausschließlichem Vorhandensein von Gesamtbewertungen lernen lassen, wurde bereits in Abschnitt 5.3 dieser Arbeit gezeigt.

- **Kommunikation und Interaktion, Schlussfolgern**

  Eine solche Prozesskarte bietet zudem den Vorteil, dass hier eine sinnvolle Schnittstelle vom gelernten subsymbolischen Wissen zu symbolischen Repräsentationen des Problems gefunden werden kann und

somit auch die Kommunikation und Interaktion mit menschlichen Nutzern erleichtert oder gar erst ermöglicht wird.

So kann ein menschlicher Experte Regionen auf der Karte markieren und mit Zusatzinformationen versehen, ob es sich dabei beispielsweise um normale Betriebszustände handelt oder ob ein Störfall eingetreten ist.

Der wesentlich Sprung jedoch, der mit der vorgestellten erweiterten Architektur zu machen wäre, ist das Loslösen vom rein datengetriebenen Paradigma. Durch das Einbringen von symbolischem Wissen auf einem Top-Down-Pfad und einer zu entwickelnden Schnittstelle zwischen der symbolischen und subsymbolischen Repräsentation lassen sich zwei wesentliche Verbesserungen erzielen.

Einerseits wird es dadurch möglich, menschliches Expertenwissen direkt in das System einzukoppeln und diese Informationen beim Lernen zu nutzen. Die Hindernisse einer rein datengetriebenen Adaptivität, wurden am Ende von Kapitel 5 umrissen.

Andererseits kann mit einer solchen Schnittstelle Wissen aus dem System ausgegeben und analysiert werden. An vielen Stellen stellt die subsymbolische Repräsentation ein Hindernis dar, da (dem Laien) kaum zu erklären ist, warum das System zu dieser oder jener Entscheidung gekommen ist. Wenn diese Information in Symbole verpackt und verständlich gemacht werden kann, erhöht das natürlich auch die Akzeptanz bei den Nutzern. Insbesondere bei sicherheitskritischen Realweltanwendungen ist dies ein wesentlicher Aspekt.

Eine Umsetzung dieser hier vorgeschlagenen erweiterten Architektur würde die Anpassungsfähigkeit des Systems deutlich erhöhen und eine leichte Übertragung auf viele verschiedene Anwendungsgebiete erlauben. Dies bleibt allerdings zukünftigen Forschungsprojekten vorbehalten.

8. Zusammenfassung

In dieser Arbeit wurde eine kognitive Architektur zur Lösung komplexer Probleme aus dem Bereich der Automatisierung vorgestellt. Das Hauptaugenmerk lag dabei auf dem Erlernen einer solchen Lösung aus Daten und den dafür notwendigen Adaptionsvorgängen und dem Lernmanagement innerhalb der Architektur.

Die zwei wesentlichen Fragen, auf die dabei eingegangen wurde, sind:

1. Wie kann gelernt werden, welche Beobachtungskanäle, wie Sensoren, oder welche Aktionsmöglichkeiten, im Sinne von Aktuatoren, wichtig und zur Lösung des Problems nützlich sind?

   Dazu wurden neue hybride Filter-Wrapper Verfahren entwickelt, welche darauf abzielen, mittels Transinformation eine gerichtete Suche nach sinnvollen Merkmalen durchzuführen. Im Vergleich zu existierenden Arbeiten auf dem Gebiet, wird die Transinformation dabei auf neue, innovative Art und Weise verwendet.

   Da die Transinformation immer aus den Daten geschätzt werden muss, bestand der erste Schritt darin, zu untersuchen, welche Schätzverfahren für Transinformation im Kontext der Merkmalsextraktion genutzt werden sollten. Der Neuheitswert ist dabei der Fokus auf die Anwendung im Merkmalsextraktionsbereich.

   Die Untersuchungen zeigten, dass die korrekte Schätzung der Transinformation hierbei zweitrangig ist. Wichtiger ist, dass die Relation der ermittelten Werte zueinander korrekt ist. Dies trifft auf die untersuchten Verfahren zu, da die Schätzfehler der Verfahren zumeist

systematischer Natur sind und sich in der Relation zueinander nicht widerspiegeln. Insofern konnte für die Schätzung der Transinformation kein bestes Verfahren identifiziert werden, jedoch wird aufgrund verschiedener günstiger Eigenschaften die Kerneldichteschätzung als zu bevorzugendes Verfahren eingestuft.

Verwendet wurde die so geschätzte Transinformation in zwei neuen Algorithmen. Einerseits wurden damit Chow-Liu Bäume konstruiert, welche es ermöglichen die Suche nach nützlichen Merkmalen zielgerichteter und damit schneller durchzuführen.

Andererseits wurde die Transinformation zwischen den verfügbaren Kanälen und dem verbleibendem Fehler eines lernenden Systems verwendet. Diese residuumsbasierten Familie von Algorithmen fokussiert dabei auf Informationen, die helfen diesen Fehler zu verringern. Es konnte experimentell gezeigt werden, dass diese neuen Algorithmen klassischen Verfahren in Geschwindigkeit und Güte der Auswahl klar überlegen sind. Abschließend wurden Anwendungsbeispiele vorgestellt, in denen die Merkmalsextraktionsverfahren gewinnbringend eingesetzt wurden.

2. Wie kann gelernt werden, die korrekte Entscheidung für eine gegebene Situation zu fällen?

   Im Rahmen dieser Arbeit wurden für die Entscheidungsfindung aktuelle Reinforcement Learning Verfahren miteinander verglichen. Im Mittelpunkt stand dabei die Tauglichkeit für Herausforderungen, wie sie im Szenario der intelligenten Feuerungsführung zu finden sind. Dabei erwiesen sich Ansätze, die auf Gauß'schen Prozessen basieren, als ungeeignet, während die NFQ- und CoSYNE-Lernverfahren mit den Problemen umgehen konnten. Für die Regelung des Kohlekraftwerks wurde dabei schlussendlich das CoSYNE-Verfahren umgesetzt, da die verwendeten rekurrenten neuronalen Netze implizit leichter mit dem Problem unvollständiger Zustandsinformationen umgehen können.

Ebenfalls diskutiert wurden in diesem Kontext die Probleme des Explorations-Exploitations-Dilemmas und der Rewarddekomposition beim Reinforcement Learning.

Für das EED wurde dabei Wert auf kontinuierliche Aktionsräume gelegt und mit dem Diffusionsbaum-basierten Reinforcement Learning ein Algorithmus vorgeschlagen, der implizit durch einen Diffusionsbaum zwischen Exploration und Ausnutzung des vorhandenen Wissens abwägen kann. Experimentell wurde hier gezeigt, dass dieser Ansatz dem vergleichbaren Sampling-basierten Q-Lernen überlegen ist.

Die Rewarddekomposition wurde in einem anspruchsvollen, kooperativen Szenario betrachtet. Dafür wurden Verfahren aus der Literatur gegen das neuentwickelte SMILE Konzept verglichen. Das vorgestellte SMILE Verfahren konnte die untersuchten Szenarien gut lösen und speziell für den Fall gegenseitiger Beeinflussung durch die einzelnen Teilsysteme, gelang es, die Vergleichsverfahren hinter sich zu lassen.

Diese wissenschaftlichen Beiträge zu einzelnen Teilaspekten des Lernens im Rahmen eines intelligenten Systems wurden am Beispiel der industriellen Feuerungsführung in einem Steinkohlekraftwerk zusammengesetzt und als funktionsfähiges Gesamtsystem zur adaptiven Regelung betrieben. Das entwickelte System konnte für das Kraftwerk Tiefstack in Hamburg eine Verbesserung erreichen, die dem menschlichen Anlagenfahrer und anderen Automatisierungsansätzen weit überlegen ist, den Wirkungsgrad bei der Verbrennung erhöht, die Emissionen reduziert und somit einen wichtigen Beitrag zum Klimaschutz liefert.

Es konnte für dieses herausfordernde Szenario gezeigt werden, dass das Erlernen komplexer Zusammenhänge und die zyklische Anpassung an neue Gegebenheiten mit den in dieser Arbeit vorgestellten Methoden nicht nur möglich, sondern auch lohnenswert ist.

Alle hier diskutierten Konzepte sind dabei nicht als reine Automatisierungslösungen zu betrachten, sondern können ebenfalls in der Robotik, in der Mensch-Maschine-Kommunikation und verwandten Feldern eingesetzt werden. Speziell für die Merkmalsextraktionsverfahren wurde diese Übertragbarkeit auch schon im Rahmen der vorangegangenen Kapitel gezeigt.

Im Sinne der kritischen Reflexion wurde zum Schluss der Arbeit auf sinnvolle Erweiterung im Gesamtkonzept eingegangen. Die Kernpunkte dieser Erweiterungen betreffen dabei eine Struktur zum Steuern des Lernens, die Systematisierung des Erlernten und eine Einbindung symbolischer Informationen um eine Mensch-Maschine-Kommunikation zu erleichtern. Diese erweiterte Architektur, die dort skizziert wird, bietet eine Vielzahl spannender wissenschaftlicher Fragestellung, deren Lösung sicherlich viele zukünftige Arbeiten füllen wird.

A. Algorithmische Details

In diesem Anhang sind algorithmische und mathematische Details zu einzelnen Verfahren zu finden, welche aus Gründen der Übersichtlichkeit und Relevanz nicht in den entsprechenden Kapiteln zu finden sind.

A.1. Transinformationsmaximierung

Die in Abschnitt 3.7 vorgestellte Transinformationsmaximierung ist nicht vollständig hergeleitet worden. Der Vollständigkeit halber wird dies hier nachgeholt. Ausgangspunkt sei folgende Gleichung zur Ableitung der Quadratischen Transinformation I_2 nach z_i:

$$\frac{\partial I_2}{\partial z_i} = \frac{\partial V_{IN}}{\partial z_i} + \frac{\partial V_{ALL}}{\partial z_i} - 2\frac{\partial V_{BTW}}{\partial z_i}. \tag{A.1}$$

Dabei entsprachen die drei Teilterme folgenden Ausdrücken:

$$V_{IN} = \sum_y \int_{\mathbf{z}} p(y, \mathbf{z}) d\mathbf{z} \tag{A.2}$$

$$V_{ALL} = \sum_y \int_{\mathbf{z}} P(y)^2 p(\mathbf{z})^2 d\mathbf{z} \tag{A.3}$$

$$V_{BTW} = \sum_y \int_{\mathbf{z}} p(y, \mathbf{z}) P(c) p(\mathbf{z}) d\mathbf{z} \tag{A.4}$$

Um die unbekannten Wahrscheinlichkeitsdichteverteilungen $\mathrm{p}(\mathbf{z})$ einfach bestimmen zu können, werden diese geschätzt. Dafür greift Torkkola auf die Kerneldichteschätzung (siehe Abschnitt 3.3.1) zurück. Der verwendete Gaußkernel G sei wie folgt definiert:

$$G(\mathbf{z}, \Sigma) = \frac{1}{(2\pi)^{\frac{d}{2}} \, |\Sigma|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}\mathbf{z}^T \Sigma^{-1} \mathbf{z}\right) \tag{A.5}$$

Die Schreibweise des Bandbreitenparameters Σ als Matrix lässt eine unterschiedliche Kernelbreite in jeder Dimension zu, allerdings wird dies vereinfacht, indem nur ein einziger Parameter σ verwendet wird, der für alle Dimension gleichermaßen gilt: $\Sigma = \sigma^2 E$. E steht dabei für die Einheitsmatrix.

Damit ergibt sich:

$$p(\mathbf{z}) = \frac{1}{N} \sum_{i=1}^{N} G(\mathbf{z} - \mathbf{z}_i, \sigma^2 E) \tag{A.6}$$

Es seien die Daten für die weitere Betrachtung in N_y diskrete Klassen eingeteilt und es sei jedem Datenpunkt im transformierten Raum $\mathbf{z}_i$ genau eine Klasse cy_i zugeordnet.

Unter der Hypothese, dass in Klasse y_p genau J_p Datenpunkte der Outputdatenmenge Z fallen, können unter Annahme einer Gleichverteilung die a priori Klassenwahrscheinlichkeiten mit $P(y_p) = \frac{J_p}{N}$ angegeben werden. Die Wahrscheinlichkeitsdichte jeder Klasse y_p wird damit mittels Kerneldichteschätzung als

$$p(\mathbf{z}|y_p) = \frac{1}{J_p} \sum_{j=1}^{J_p} G(\mathbf{z} - \mathbf{z}_{pj}, \sigma^2 E) \tag{A.7}$$

definiert.

Benötigt wird jedoch die Verbundwahrscheinlichkeit $p(y, \mathbf{z}) = p(\mathbf{z}|y)P(y)$, welche jetzt mit

$$p(y, \mathbf{z}) = \frac{1}{N} \sum_{j=1}^{J_p} G(\mathbf{z} - \mathbf{z}_{pj}, \sigma^2 E) \tag{A.8}$$

für alle Klassen $p = 1, ..., N_y$ berechnet werden kann. Da die gesamte Dichte über allen Daten nichts anderes als die Summe über die einzelnen Verbundwahrscheinlichkeiten je Klasse ist, ergibt sich hierfür:

$$\begin{aligned} p(\mathbf{z}) &= \sum_{p=1}^{N_y} p(y_p, \mathbf{z}) \\ &= \frac{1}{N} \sum_{p=1}^{N_y} \sum_{j=1}^{J_p} G(\mathbf{z} - \mathbf{z}_{pj}, \sigma^2 E) \\ &= \frac{1}{N} \sum_{i=1}^{N} G(\mathbf{z} - \mathbf{z}_i, \sigma^2 E). \end{aligned} \tag{A.9}$$

Außerdem ist folgender Zusammenhang bezüglich des Produkts zweier Kernel relevant:

$$\int_Z G(\mathbf{z} - \mathbf{z}_k, \sigma^2 I) G(\mathbf{z} - \mathbf{z}_j, \sigma^2 E) d\mathbf{z} = G(\mathbf{z}_k - \mathbf{z}_j, 2\sigma^2 E)$$

Setzt man dies nun in die Formeln für V_{IN},V_{ALL} und V_{BTW} ein, ergibt sich daraus:

$$\begin{aligned} V_{IN}(y_i, \mathbf{z}_i) &= \sum_{p=1}^{N_y} \int_{\mathbf{z}} p(y_p, \mathbf{z})^2 d\mathbf{z} \\ &= \frac{1}{N^2} \sum_{p=1}^{N_y} \sum_{k=1}^{J_p} \sum_{l=1}^{J_p} G(\mathbf{z}_{pk} - \mathbf{z}_{pl}, 2\sigma^2 E) \qquad \text{(A.10)} \\ V_{ALL}(y_i, \mathbf{z}_i) &= \sum_{p=1}^{N_y} \int_{\mathbf{z}} P(y_p)^2 p(\mathbf{z})^2 d\mathbf{z} \\ &= \frac{1}{N^2} \left(\sum_{p=1}^{N_y} \left(\frac{J_p}{N} \right)^2 \right) \sum_{k=1}^{J_p} \sum_{l=1}^{J_p} G(\mathbf{z}_k - \mathbf{z}_l, 2\sigma^2 E) \qquad \text{(A.11)} \\ V_{BTW}(y_i, \mathbf{z}_i) &= \sum_{p=1}^{N_y} \int_{\mathbf{z}} p(y_p, \mathbf{z}) P(y_p) p(\mathbf{z}) d\mathbf{z} \\ &= \frac{1}{N^2} \sum_{p=1}^{N_y} \frac{J_p}{N} \sum_{j=1}^{J_p} \sum_{k=1}^{N} G(\mathbf{z}_{pj} - \mathbf{z}_k, 2\sigma^2 E). \qquad \text{(A.12)} \end{aligned}$$

Die Summe mit der Zählvariablen p summiert dabei immer über die Klassen auf, während die Zählvariablen k und l die paarweise Interaktion zwischen je zwei Kernels darstellen.

Diese Teilgleichungen werden von Torkkola und Principe [Torkkola, 2003] [Principe et al., 2000] als Informationspotentiale bezeichnet und ähnlich zu physikaschen Potentialen interpretiert (Erläuterung siehe Abschnitt 3.7). Aus diesen Informationspotentialen ergeben sich durch Ableitung der Kernel G nach der Kettenregel A.13 die sogenannten Informationskräfte.

$$\frac{\partial}{\partial \mathbf{z}_i} G(\mathbf{z}_i - \mathbf{z}_j, 2\sigma^2 E) = G(\mathbf{z}_i - \mathbf{z}_j, 2\sigma^2 E) \frac{\mathbf{z}_i - \mathbf{z}_j}{2\sigma^2} \qquad \text{(A.13)}$$

Für die drei einzelnen Informationskräfte sehen die Ableitungen wie folgt aus:

$$\frac{\partial}{\partial \mathbf{z}_{yi}} V_{IN} = \frac{1}{N^2\sigma^2} \sum_{k=1} J_y G(\mathbf{z}_{yk} - \mathbf{z}_{yi}, 2\sigma^2 E)(\mathbf{z}_{yi} - \mathbf{z}_{yk}) \tag{A.14}$$

$$\frac{\partial}{\partial \mathbf{z}_{yi}} V_{ALL} = \frac{1}{N^2\sigma^2} \left(\sum_{p=1}^{N_y} \left(\frac{J_p}{N} \right)^2 \right) \sum_{k=1}^{N} G(\mathbf{z}_k - \mathbf{z}_i, 2\sigma^2 E)(\mathbf{z}_i - \mathbf{z}_k) \tag{A.15}$$

$$\frac{\partial}{\partial \mathbf{z}_{yi}} V_{BTW} = \frac{1}{N^2\sigma^2} \sum_{p=1}^{N_y} \frac{J_p + J_y}{2N} \sum_{j=1}^{J_p} G(\mathbf{z}_{pj} - \mathbf{z}_{yi}, 2\sigma^2 E)(\mathbf{z}_{yi} - \mathbf{z}_{yj}). \tag{A.16}$$

Dabei wurde hier der Übersichtlichkeit halber nach $\mathbf{z}_{yi}$ abgeleitet, statt nach $\mathbf{z}_i$. Die einzige Änderung ist dabei der Wegfall der Summe über die Klassen.

Die letzten drei angegebenen Formeln können mit den vorhandenen Daten ausgerechnet werden und dann für den Term $\frac{I_2}{z_i}$ in Abschnitt 3.7 eingesetzt werden.

A.2. Grundlagen für Gauß'sche Prozesse

Dieser Abschnitt vervollständigt die Ausführungen in Abschnitt 4.2. Die Notation orientiert sich dabei am Standardwerk für Gauß'sche Prozesse [Rasmussen und Williams, 2005].

Sei eine Menge von Basisfunktionen $\Phi_1, ..., \Phi_n$ gegeben, die mit den Gewichten $w_1, ..., w_n$ linear überlagert werden. Man kann hier an ein neuronales Netz mit radialen Basisfunktionen (RBF-Netz)

[Moody und Darken, 1989] denken. Die Basisfunktionen sind Gaußfunktionen, welche distanzbasiert aktiviert werden. Die gewichtete lineare Überlagerung findet in der zweiten Schicht des Netzes statt. Betrachtet man die Gewichte $w_1, ..., w_n$ nun nicht als skalare Werte, sondern als normalverteilte Zufallsvariablen[1] mit Mittelwert und Varianz so erhält man einen Gauß'schen Prozess. Die Basisfunktion kann dabei ein beliebiger Mercer-Kernel (also symmetrisch positiv semidefinit) sein und wird hier auch als Kovarianzfunktion bezeichnet. Im Rahmen dieser Arbeit wird ausschließlich der Gaußkernel verwendet.

Formal nach [Rasmussen und Williams, 2005] definiert sind Gauß'sche Prozesse wie folgt:

Definition A.1

Gauss'scher Prozess

Ein stochastischer Prozess[2] wird als Gauß'scher Prozess bezeichnet, wenn alle Realisierungen über die Zufallswerte des Prozesses normalverteilt sind.

$$f(x) \sim GP(E\{f(x)\}, k(x, x')) \tag{A.17}$$

Ein Gauß'scher Prozesses GP, der eine Funktion f dargestellt, besteht aus zwei Komponenten: der Mittelwertfunktion $E\{f(x)\}$ und der Kovarianzfunktion $k(x, x')$.

Diese Normalverteilung der Zufallswerte des Prozesses ermöglicht in vielen Fällen das Ableiten einer geschlossenen Lösung, was sie für viele Anwendungen attraktiv macht. Man kann sich die Gauß'schen Prozesse

[1] Der Name der Gauß'schen Prozesse rührt aus diesem Fakt her, nicht aus der Verwendung des Gaußkernels im Eingaberaum.

[2] Im Sinne Kolmogorovs sind stochastische Prozesse eine zeitlich geordnete Folge von Zufallswerten. Im zeitdiskreten Fall wird dies oft auch als Zeitreihe bezeichnet.

in diesem Zusammenhang auch als Verteilung über Funktionen statt über einzelne Zufallsvariablen vorstellen.

Für praktische Zwecke wird angenommen, dass der Mittelwert der zu approximierenden Funktion null ist, also $E\{f(x)\} = 0$. Die kann immer dadurch erreicht werden, dass die zu approximierende Funktion durch eine Skalierung mittelwertfrei gemacht wird.

Das Problem der Funktionsapproximation sei wie folgt formal beschrieben. Wenn die Matrix X die Position der gegebenen Stützstellen angibt und der (mittelwertfreie) Vektor Y den zugehörigen Funktionswert angibt, so sind für die Punkte $\tilde{X}$ die zugehörigen Funktionswerte $\tilde{Y}$ gesucht. Eingesetzt in die Definition A.1 ergibt sich

$$\begin{bmatrix} Y \\ \tilde{Y} \end{bmatrix} \sim N\left(0, \begin{bmatrix} K(X,X) & K(X,\tilde{X}) \\ K(\tilde{X},X) & K(\tilde{X},\tilde{X}) \end{bmatrix}\right). \tag{A.18}$$

$K(X,X)$ ist dabei die Matrix, in der alle Datenpunkte zueinander die Kernel- bzw. Kovarianzfunktion $k(x,x') = e^{-\frac{1}{2}|x-x'|^2}$ auswerten. Durch den symmetrischen Kernel ergibt sich eine positiv semidefinite Matrix, welche die Kovarianzen der Datenpunkte untereinander repräsentiert. Unter Verwendung der aus der Stochastik bekannten Gesetzmäßigkeiten[3] können die gesuchten Funktionswerte $\tilde{Y}$ wie folgt berechnet werden

$$E(\tilde{Y}|X,Y,\tilde{X}) = K(\tilde{X},X)K(X,X)^{-1}Y^T. \tag{A.19}$$

Neben dem eigentlichen Schätzwert bieten die Gauß'schen Prozesse den Vorteil, dass zusätzliche eine Konfidenzaussage in Form der Varianz getroffen werden kann:

[3] Für die komplette Herleitung sei auf [Rasmussen und Williams, 2005] Kapitel 2 und 3 verwiesen.

$$var(\tilde{Y}|X,Y,\tilde{X}) = var(\tilde{Y}|X,\tilde{X}) \tag{A.20}$$
$$= K(\tilde{X},\tilde{X}) - K(\tilde{X},X)K(X,X)^{-1}K(X,\tilde{X}) \tag{A.21}$$

Wichtig ist in diesem Zusammenhang allerdings, dass die berechnete Varianz nur auf der Verteilung der bekannten Datenpunkte basiert, aber nicht die Stochastizität der Daten selbst berücksichtigt. Dies bedeutet, dass in Gegenden des Funktionsraums, in dem sich viele Datenpunkte befinden, einer höhere Konfidenz, also eine geringere Varianz, ermittelt wird, als an Orten mit einer geringeren Dichte von Datenpunkten. Die maximale Unsicherheit herrscht an den Orten, in deren Umgebung keine Datenpunkte liegen.

Das Rauschen in den Daten, also die Unsicherheit über den Funktionswert an einer festen Stelle im Raum, wird vielmehr als Eingangsgröße für das Verfahren benötigt. Dieser Hyperparamter muss dabei sinnvoll geschätzt werden. Diese maximale Unsicherheit entspricht nicht einer beliebig großen Varianz, sondern wird apriori über den Term $K(\tilde{X},\tilde{X})$ definiert. Exemplarisch werden diese Aussagen in Abbildung A.1 am Beispiel einer Funktionsapproximation gezeigt.

In den bisherigen Formeln tritt dieses Rauschen bisher nicht auf, es wurde von rauschfreien Daten ausgegangen. Die einzige notwendige Änderung für den Fall, dass Rauschen in den Daten enthalten ist, ergibt sich bei der Kovarianzfunktion zwischen alle bekannten Datenpunkte X. Für ein angenommenes normalverteiltes Rauschen mit Varianz σ^2 ergibt sich

$$cov(Y) = K(X,X) + \sigma^2 I. \tag{A.22}$$

I ist dabei die Einheitsmatrix. Die Varianz in den Daten muss vorhergeschätzt werden oder kann im Rahmen einer Maximum-Likelihood-Schätzung als Hyperparameter optimiert werden.

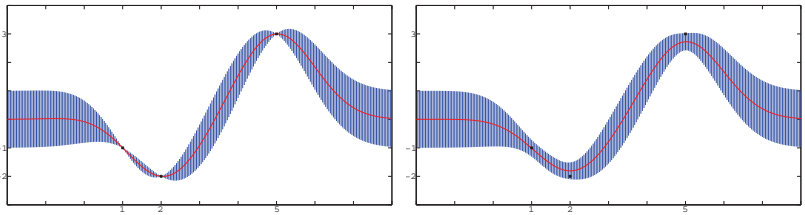


Abbildung A.1.: Approximation eines Funktionsverlaufs durch einen Gauß'schen Prozess. Die Datenpunkte sind die gegebenen Funktionswerte $f(x) = (-1, -2, 3)$ an den Stelle $x = (1, 2, 5)$, mit deren Hilfe die Approximation durchgeführt werden soll. Die durchgezogene Linie stellt den resultierenden Funktionsverlauf dar. Der Schlauch um diese Funktion herum, stellt die einfache Varianz dar. An den drei Punkten, an denen Daten vorhanden sind, geht diese Varianz gegen null, weiter entfernt wird sie maximal. Man sieht, dass eine Extrapolation über die Datenpunkte hinaus verfahrensbedingt immer gegen null gezogen wird. Dies ist immer dann korrekt, wenn die Funktion mittelwertfrei ist. **Links:** Hier werden die Daten als rauschfrei angenommen. Der Funktionswert für $x = 3$ ist $f(x) = -0.91$. **Rechts:** Mit denselben drei Punkten wird unter Annahme eines Rauschen von $\sigma > 0$ dieser Funktionsverlauf geschätzt. Neben einem geringfügig anderen Verlauf ($f(x = 3) = -0.82$) ist deutlich zu erkennen, dass auch an den gegebenen Punkten eine Restunsicherheit verbleibt.

Vom rechentechnischen Aufwand sind die Kernelmatrizen am interessantesten, welche eine zusätzliche Spalte und Zeile für jeden Datenpunkt haben. Die Matrix $K(X, X)$ kann dabei vorberechnet (und invertiert) werden, die Matrizen $K(\tilde{X}, \tilde{X})$, $K(\tilde{X}, X)$ und $K(X, \tilde{X})$ hingegen müssen bei jeder Approximation neu berechnet werden. Dies kann je nach Anwendung sehr oft vorkommen und muss entsprechend beim Systemdesign beachtet werden.

Für einen umfassenderen Überblick zu den Gauß'schen Prozessen sei auf [Rasmussen und Williams, 2005] verwiesen, da hier nicht umfassend auf die mathematischen Hintergründe eingegangen werden kann.

A.3. Evolutionäre Operatoren im CoSYNE

Diese Ausführungen zum evolutionären Training im CoSYNE Algorithmus beziehen sich auf Abschnitt 4.3 dieser Arbeit.

Normalerweise gestaltet sich das Training rekurrenter Netze als sehr schwierig. Man beschränkt sich entweder auf festgelegte rekurrente Verbindungen, sogenannte partiell rekurrente Netze wie beispielsweise das Elman-Netz, oder muss langwierige Trainingsmethoden wie zum Beispiel Backpropagation through Time (BPTT) [Rumelhart et al., 1986] einsetzen. An dieser Stelle bieten die Neuroevolutionsverfahren eine sinnvolle Alternative.

Bei CoSYNE wird die evolutionäre Optimierung ausschließlich zur Parameteroptimierung (also den Gewichten im Netzwerk) verwendet, nicht aber zur Strukturoptimierung (z.B. Anzahl der Neuronen). Daher werden die Gewichte eines rekurrenten Netzes als Individuen kodiert, dargestellt ist dies in Abbildung 4.5.

Um die Gewichte des Netzes so anzupassen, dass es die Abbildung von Zustand auf Aktionen lernt, wird sich verschiedener Mechanismen bedient. Wichtigster Bestandteil ist die Definition einer sogenannten Fitnessfunktion. Diese bewertet die Qualität einer gefundenen Lösung beispielsweise in Form eines Fehlermaßes oder einer Funktion des erzielten Rewards. Nach dem „Überleben der Stärksten“ Prinzip werden gute Lösungen von schlechten Lösungen getrennt. Die so ausgewählten guten Lösungen werden mittels evolutionärer Operatoren manipuliert um noch bessere Lösungen zu finden und bilden eine neue Population. Diese Evolutionsschritte werden wiederholt, bis die beste Lösung gefunden wurde.

Als evolutionäre Operatoren kommen dabei Mutation, Rekombination und Coevolution zum Einsatz. Eine grafische Interpretation diser Operationen ist in Abbildung A.2 gezeigt.

- Mutation ist dabei die zufällige Veränderung eines Gewichtes des Netzwerks. Jedes Gewicht des neuronalen Netzes wird dabei mit einer bestimmten Wahrscheinlichkeit p_{mut} mutiert. Im CoSYNE Framework wird dies realisiert, in dem auf das aktuelle Gewicht eine Standard-Cauchy-verteilte Zufallsvariable addiert wird

$$w_{neu} = w_{alt} + C \tag{A.23}$$

  Die Wahrscheinlichkeitsdichte der Standard-Cauchy-Verteilung ist wie folgt definiert:

$$f(x) = \frac{1}{(1+x^2)\pi} \tag{A.24}$$

  Diese Verteilung ist der Normalverteilung recht ähnlich, allerdings ist die Wahrscheinlichkeit für extreme Ausprägungen wesentlich größer. Das heißt, gegenüber einer Normalverteilung werden größere Gewichtsänderungen bevorzugt und man spricht auch von einer supergaußförmigen Verteilung.

- Bei der Rekombination werden zufällige gewählte Elemente aus zwei Netzwerken miteinander getauscht. Die Auwahl dieser beiden Netzwerke erfolgt stochastisch, wobei die Wahrscheinlichkeit zur Rekombination ausgewählt zu werden proportional zur Fitness ist (Überleben der Stärksten). Nach der Auswahl beider Eltern, werden zufällige Crossoverpunkte bestimmt, die angeben, welche Gewichte zwischen den beiden Eltern ausgetauscht werden. Mehr zu Crossoverpunkten und deren Auswahl findet sich z.B. in [Nissen, 1997].

- Der Begriff der Koevolution ist in der Literatur zu evolutionären Algorithmen nicht eindeutig abgrenzbar. Im Sinne des CoSYNE-Algorithmus wird darunter das Vertauschen eines Gewichtes über mehrere oder alle Individuen der Population verstanden. Angedeutet werden diese Operationen in Abbildung A.2. Die Bestimmung welche Gewichte hier untereinander vertauscht werden wird wieder zufällig bestimmt. In [Gomez et al., 2008] werden verschiedene Verteilungen

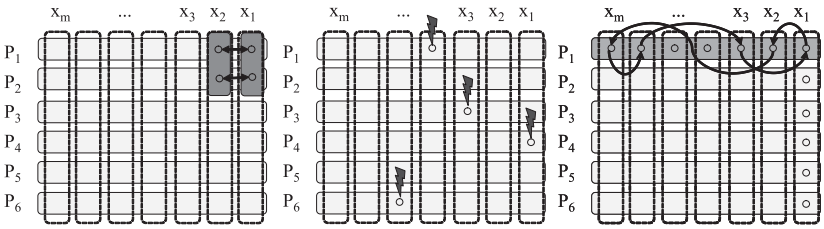


Abbildung A.2.: Übersicht über die drei von CoSYNE verwendeten Evolutionsschritte. Die dargestellten Vektoren enthalten realwertige Netzwerkgewichte: Eine Spalte entspricht einem Individuum; die grafische Interpretation ist in Abbildung 4.5 gezeigt. **Links:** Rekombination von zwei Individuen. **Mitte:** Mutation zufällig ausgewählter Netzwerkgewichte. **Rechts:** Co-Evolutionärer Austausch von Gewichten innerhalb derselben Subpopulation (Zeile). Die Auswahl der zu permutierenden Gewichte erfolgt fitnessgesteuert.

vorgeschlagen, wie dies erfolgen kann. Im einfachsten Fall wird dies über eine feste Wahrscheinlichkeit p_{coev} für ein Individuum realsiert.

Für detaillierte Erläuterung und Spielarten evolutionären Operatoren, genetischer Algorithmen und Evolutionsstrategien sei auf eines der zahlreichen Werke zu diesem Themenkomplex verwiesen, zum Beispiel [Nissen, 1997]. Der Einfluss verschiedener Parameter, wie z.B. die Mutations- und Koevolutionswahrscheinlichkeiten, auf das CoSYNE-Verfahren wurden in der Diplomarbeit [Hellwig, 2009] ausführlich untersucht.

B. Beispiele zur Merkmalsextraktion

Hier sollen die in Abschnitt 3.10 angesprochenen Beispiele etwas vertiefend vorgestellt werden.

B.1. Schätzung von Nutzerinteresse

Im Rahmen der Entwicklung intelligenter Serviceroboter, beispielsweise für den Einsatz als Informationsdienstleister in Baumärkten [Gross et al., 2009] oder anderen öffentlichen Räumen, ist es von entscheidender Bedeutung, wie der Roboter auf sich und sein Angebot aufmerksam machen kann. Weder scheint ein regungsloses Verharren des Roboters angebracht, dann könnte man ein einfaches Infoterminal benutzen, noch sollte er sich auf jede Person stürzen, die er finden kann. Vielmehr ist ein smartes, „natürliches“ Verhalten gewünscht. Um entscheiden zu könne, ob ein Nutzer Interesse an einer Interaktion mit dem Roboter hat, muss er versuchen, basierend auf seinen Beobachtungen, die Intentionen eines potentiellen Interaktionspartners zu schätzen.

Es wurde dazu untersucht, inwieweit die Trajektorie einer Person genutzt werden kann, um diese Entscheidung zu treffen. Für die Datengewinnung wurde der Roboter HOROS [Scheidig et al., 2006] verwendet. Dabei werden über einen Personentracker [Martin et al., 2006],

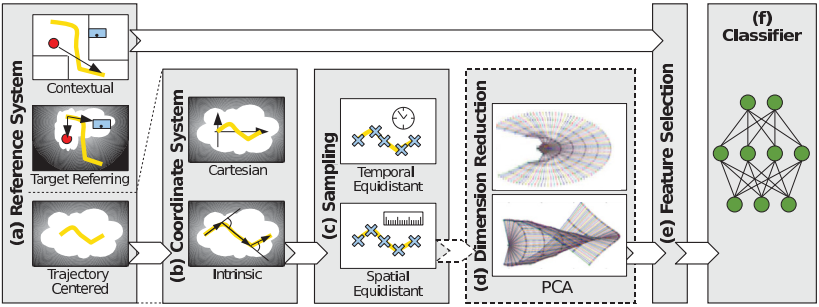


Abbildung B.1.: Übersicht des Systems zur Schätzung des Nutzerinteresses. **(a)** Wahl eines Referenzsystems (abhängig vom Szenario), **(b)** Transformation der Personenposition in ein geeignetes Koordinatensystem, **(c)** Abtasten der Trajektorie, **(d)** eventuelle Dimensionsreduktion mittels Hauptkomponentenanalyse, **(e)** Merkmalsselektion und **(f)** Klassifikator.

welcher auf Sonar-, Laser- und Audiodaten operiert, Trajektorien aufgezeichnet. Während der Datenaufzeichnung offerierte der Roboter Speisepläne, Kinoprogramm und ähnliche Informationen den Passanten im wohlfrequentierten Eingangsbereich eines Universitätsgebäudes. Danach wurde mittels Fragebögen die Meinung der Passanten erfragt, um herauszufinden, warum oder warum nicht sie mit dem Roboter interagierten.

Mit den so gewonnenen Daten, kann ein automatisches Erkennungssystem trainiert werden, welches nur noch Leute anspricht, die einer Interaktion nicht grundsätzlich abgeneigt sind. Die Architektur dieses Erkennungssystems ist in Abbildung B.1 gezeigt.

Dabei sind mehrere Stufen von Interesse. Zu allererst ist von entscheidender Bedeutung, die Wahl eines geeigneten Referenzsystems. Damit ist die kontextuelle Einbindung des Roboters in seine Umgebung gemeint, also ob nur die Trajektorie selbst betrachtet wird oder diese in Relation zum Roboter, zu Türen und anderen interessanten Objekten der Umgebung. Weitere Vorverarbeitungsschritte sind möglich, aller-

dings nicht notwendig. So stellt sich beispielsweise die Frage nach einem geeigneten Koordinatensystem oder nach einem Resampling der Trajektorie in räumlich oder zeitlich äquidistanten Punkten. Ebenfalls ist eine Dimensionsreduktion mittels einer PCA möglich.

Um aus dieser Vielzahl möglicher Repräsentationsformen für die Trajektorie jene Kodierung und Vorverarbeitung auszuwählen, die für die gestellte Aufgabe, also das Erkennen des Nutzerinteresses, von Relevanz sind, wurde eine Merkmalsselektion durchgeführt. Zur Anwendung kamen hierbei die in Kapitel 3 besprochenen Verfahren der (Verbund-)Transinformation. Im Ergebnis wurde dabei festgestellt, dass etliche Kanäle (beispielsweise die X und Y Koordinaten zu unterschiedlichen Zeitpunkten) redundante oder unnütze Informationen enthalten und demzufolge vernachlässigt werden können. Auch nach Anwendung der Hauptkomponentenanalyse konnte etliche der entstandenen neuen Dimensionen eliminiert werden. Hier handelt sich um ein typisches Beispiel dafür, dass die PCA aufgrund des unüberwachten Anpassens, keinen wesentlichen Gewinn erzielt.

Die besten Ergebnisse wurden mit einem zweischichtigen neuronalen Netz und acht ausgewählten Merkmalen (keine PCA Transformation) erzielt und lagen bei 17,5% Fehlerrate. Das ist zwar noch bei weitem nicht die gewünschte Güte, jedoch besser als mit anderen Repräsentationsformen (z.B. PCA und Rohdaten). Und schließlich verbleibt die Frage ob man nur auf der Trajektorie basierend auf das Nutzerinteresse schließen kann.

An den Arbeiten zu diesem Thema waren neben dem Autor dieser Arbeit Antje Ober, Steffen Müller, Sven Hellbach, Andrea Scheidig und Horst-Michael Groß beteiligt.

B.2. Audiobasierte Nutzermodellierung

Sprache als Mittel der zwischenmenschlichen Kommunikation enthält wesentlich mehr als nur die gesprochenen Worte und den sich daraus ergebenden Kontext. Vielmehr kann man anhand des Gehörten auf Geschlecht, Alter und z.B. die Stimmungslage des Gegenübers schließen. Im Rahmen der Diplomarbeit von Tobias Prüger [Prüger, 2008] wurde untersucht, inwieweit ein automatisches System aus Sprachdaten auf Stimmungslage und Stresslevel schließen kann und gegebenenfalls eine Sprecheridentifikation vornehmen kann.

Folgt man [Paeschke, 2003] so lassen sich zum Beispiel die Stimmungen durch Sprechgeschwindigkeit, Stimmlage, Stimmumfang, Lautstärke und Grundfrequenzverhalten auseinanderhalten. Ebenso bei Untersuchungen zum Thema Stress lässt sich der Zustand auf Grundfrequenz, Geschwindigkeit und Signalenergie abbilden. Hier sollten geeignete Merkmale jedoch datenbasiert gelernt werden.

Das mit Mikrophonen aufgenommene Sprachsignal wird danach im ersten Schritt einer adaptiven Rauschunterdrückung unterzogen [Brückmann et al., 2007] und in Sprache bzw. Nicht-Sprache unterteilt. Danach werden aus dem Signal 370 Merkmale extrahiert, darunter die Grundfrequenz, MFCC (Mel Frequency Ceptral Coefficents), Formanten, statistische Momente und andere. Mit diesen Merkmalen wurde dann eine Signifikanzanalyse durchgeführt um das Problem auf die nützlichen Kanäle zu reduzieren. In dem reduzierten Raum wurden dann mittels neuronalen Netzen und Maximum-Likelihood-Klassifikatoren versucht die Stimmungen und der Stresszustand zu schätzen. Der Gesamtablauf ist in Abbildung B.2 zu sehen.

Für die Signifikanzanalyse kam das Verfahren mit Chow-Liu-Bäumen zum Einsatz. Ursprünglich war geplant, dass eine einfache Vorwärtsselektion verwendet wird. Es sollte nämlich auf jeden Fall ein Wrapper zum Einsatz kommen, um die Nützlichkeit der Merkmale zu behandeln.

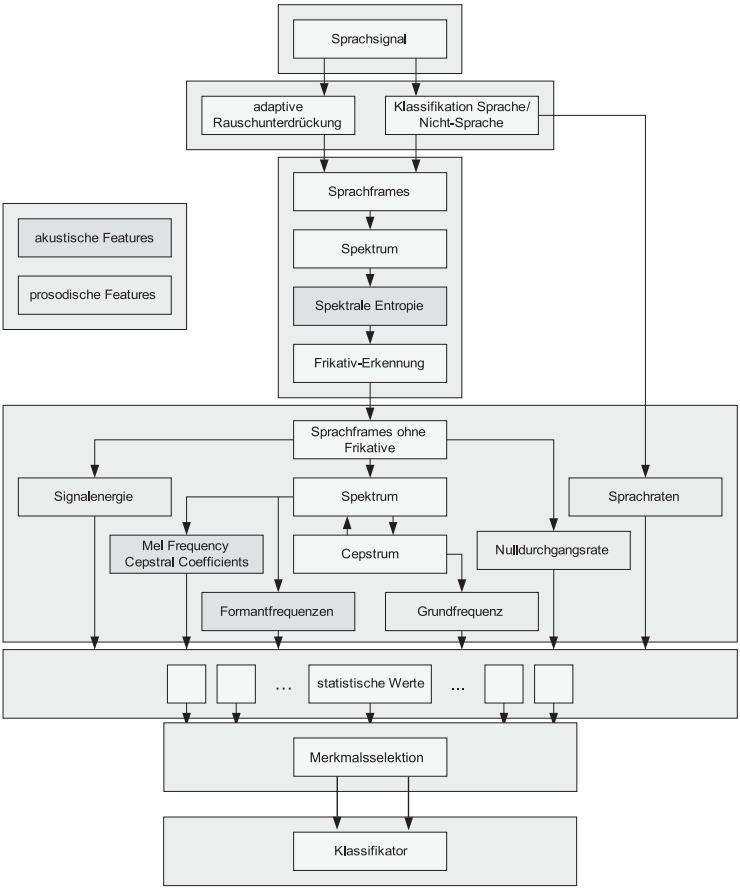


Abbildung B.2.: Allgemeiner Aufbau das Klassifikationssystems für Sprache. Nach der Rauschunterdrückung und der Sprache/Nicht-Sprache Klassifikation werden Frikative (Reibelaut, störende Konsonanten wie z.B. f,s,z) entfernt. Danach werden aus dem Sprachframe, dem Spektrum und dem Cepstrum (informell Spektrum des logarithmierten Spektrums) verschiedene Merkmale extrahiert. Aus diesen werden statistische Momente, Maxima, Minima, zeitliche Änderungen usw. gebildet. Diese Menge an Merkmalen wurden dann mit dem Chow-Liu-Baum Verfahren reduziert und zur Klassifikation verwendet. Das Bild basiert auf [Prüger, 2008].

Der diskutierte quadratische Zusammenhang zur Anzahl der betrachteten Kanäle macht, dies jedoch unmöglich. Statt geschätzter, mehrerer Wochen konnte mit der Chow-Liu Baum Methode die Auswahl in zwei Tagen abgeschlossen werden.

Bei der Aufgabe der Emotionserkennenung wurden über mehrere Versuche/Datensätze gemittelt durchschnittlich 13 Merkmale ausgewählt, wobei beispielsweise Minimum und Median der Grundfrequenz regelmäßig gewählt wurden. Bei Untersuchungen zur Sprecheridentifikation wurden wesentlich mehr Merkmale gewählt (57 Stück) wobei hier hauptsächlich Mittelwerte und Maxima der Formant, MFCCs und Grundfrequenz als nützlich eingestuft wurden.

Mit den so trainierten Klassifikatoren konnte die Emotionserkennung in rund 70-80% der Fälle (Sprecherabhängig, Leave-one-out Kreuzvalidierung) die korrekte Stimmung erkennen. Bei der Stresserkennung waren die Ergebnisse deutlich besser (bis zu 90% korrekte Klassifikation), allerdings die Datenbasis auch wesentlich kleiner. Für die Untersuchungen zur Sprecheridentifikation wurden 7 Sprecher trainiert und in rund 53% auch die korrekte 1-aus-7 Auswahl getroffen.

B.3. Prädiktion des Schnittregisterfehlers

Bei großen industriellen Buchdruckmaschinen wird der Seiteninhalt auf eine Papierbahn gedruckt, welche danach getrocknet, gefaltet und zurechtgeschnitten wird. Der prinzipielle Aufbau einer solchen Maschine ist in Abbildung B.3 dargestellt. Dies geschieht bei sehr großen Geschwindigkeiten, so dass viele Abläufe vollautomatisiert sind. Ein Problem, dass hierbei auftritt, ist der sogenannte Rollenwechsel. Es handelt sich dabei um den Fall, dass eine Papierrolle zu Ende geht und durch eine neue ersetzt werden muss. Dazu werden alte und neue Papierbahn übereinander geklebt um einen kontinuierlichen Druckbetrieb

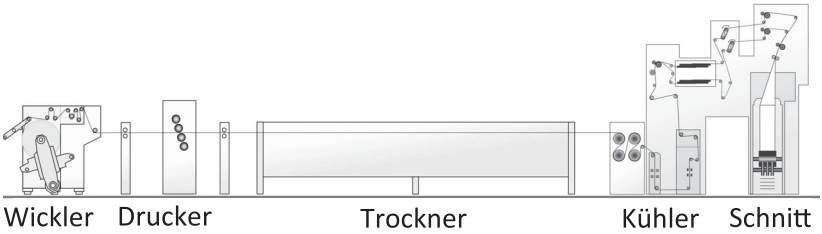


Abbildung B.3.: Allgemeiner Aufbau einer Illustrationsdruckmaschine. Links wird das Papier von den Rollen abgewickelt, bevor es in der Druckeinheit bedruckt wird. Danach folgt Trockner, Kühlung und im letzten Block eine Wiederbefeuchtung, sowie ein Längsschnitt. Danach wird die Bahn über Versatzstangen gefaltet bevor sie in der Falzeinheit vom Messerkopf in Seiten geschnitten wird. Das Bild ist aus [Möller, 2009] entnommen.

zu gewährleisten. Jedoch birgt dieses Vorgehen das Problem, dass dieses Übereinanderkleben den Druckvorgang stört. Ganz speziell geht es hierbei um den Schneidevorgang am Ende des Vorgangs. Die Seiten dürfen nicht an beliebiger Stelle zerschnitten werden, sondern nur an speziellen Stellen (zwischen den Seiteninhalten) gekennzeichnet durch das Schnittmarken. Alles was einen zu großen Schnittregisterfehler aufweist, muss aussortiert werden.

Dieser Fehler muss also nach einem Rollenwechsel schnellstmöglich eliminiert werden, um die unvermeidbare Menge an Makulaturexemplaren zu minimieren. Dazu existieren lineare Bahnlaufmodelle, welche basierend auf physikalischen Modellen versuchen den Fehler vorherzusagen. Zum Vergleich dazu wurde im Rahmen dieses Projekts untersucht, inwieweit eine Signifikanzanalyse und eine Modellierung durch ein neuronales Netz Vorteile bringen. Dabei geht es nicht um die eigentliche Regelung, sondern nur um die Systemidentifikation/-modellierung.

Als Datenmaterial standen 312 Aufzeichnungen (jeweils mit bis zu 6000 einzelnen Datenpunkten) in 29 Kanälen/Sensoren von Rollenwechseln

zur Verfügung. Dabei wurde mit einem zusätzlichen Sensor am Schnittmesser der Schnittregisterfehler bestimmt und stellt damit die Grundwahrheit zur Verfügung. Die Daten wurden dann einer Normalisierung, einer Totzeitbereinigung und einer Tiefpassfilterung unterzogen.

Neben dem Training eines neuronalen Modelles mit allen verfügbaren Eingangskanälen, wurden mit unterschiedlichen Methoden informative Merkmale ausgewählt. Dazu kam der lineare Korrelationskoeffizient (6 ausgewählte Kanäle) zum Einsatz, wie auch die Transinformation (ebenfalls 6 ausgewählte Kanäle) und die Verfahren zur Residual Mutual Information in den Varianten 1 (10 ausgewählte Kanäle) und 2 (12 ausgewählte Kanäle).

Dabei erwies sich die Residual Mutual Information allen anderen Ansätzen als deutlich überlegen, wobei als Bewertungskriterium eine virtuell[1] korrigierte Anzahl von Mängelexemplaren pro Rollenwechsel verwendet wurde. Mit einer Korrektur durch ein neuronales Netz ohne eine Merkmalsselektion unter Verwendung aller 29 Kanäle konnten 69% der Rollenwechsel korrigiert werden, unter Verwendung des Residual Mutual Information Verfahrens lag die Korrekturquote bei immerhin 86% unter Verwendung von nur 10 Merkmalen.

Damit konnte gezeigt werden, dass erstens die Modellierung durch ein neuronales Netz der linearen Modellierung überlegen ist[2] und zweitens die Verwendung der Merkmalsselektion einen wesentlichen Schritt zur Verbesserung der Modellqualität darstellt.

Dieses Szenario wurde im Rahmen der Diplomarbeit [Möller, 2009] untersucht.

[1] Es wurde nicht geregelt, sondern eine optimale Korrektur unter Verwendung des Netzwerkmodells angenommen.

[2] Genaue Zahlen zu nennen ist leider nicht möglich, da die Schätzung über das lineare Bahnlaufmodell extern durch geführt wurde und dabei keine Trennung zwischen Trainings- und Testdaten vorgenommen wurde.

C. Simulationsumgebungen

Es sollen kurz die zentralen Zusammenhänge der in dieser Arbeit verwendeten Simulatoren beschrieben werden.

C.1. Mountain Car

Hierbei handelt es sich um einen klassischen Benchmark aus der Reinforcement Learning Literatur: [Moore und Atkeson, 1995] und [Sutton und Barto, 1998].

Der grundlegende Aufbau des Szenarios ist in Abbildung C.1 gezeigt.

Der Zustandsraum $\mathcal{S}$ ist zweidimensional und besteht aus den kontinuierlichen Werten Position x und Geschwindigkeit v des Fahrzeugs. Der Aktionsraum $\mathcal{A}$ ist eindimensional und beschreibt die auf den Wagen wirkende Kraft. Diese ist so beschränkt, dass es nicht möglich ist, den Anstieg der Umgebung aus dem Stand zu bezwingen. Die Zustandsübergangsfunktion $\mathcal{P}$ ist deterministisch und berechnet sich nach den unten aufgeführten Formeln. Die Rewardfunktion $\mathcal{R}$ ist so gestaltet, dass es nur in direkter Umgebung um den Zielort bei einer Geschwindigkeit nahe Null einen positiven Reward gibt. Modelliert wird dieser durch eine Normalverteilung im Zustandsraum mit einem Mittelwert von $\mu_x = 0.6$ und $\mu_v = 0$ mit den Varianzen $\sigma_x = 0.1$ und $\sigma_v = 0.2$. Alle anderen Geschwindigkeits-Positions-Paare werden mit einem Reward von $R = -0.1$ bestraft.

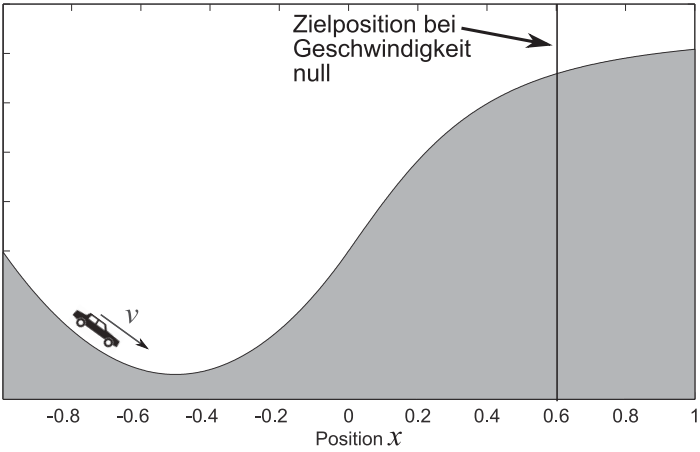


Abbildung C.1.: MountainCar-Umgebung. Der Wagen muss auf die Zielposition gebracht werden und dort anhalten. Die Markierung zeigt dabei die Zielposition an, bei der es eine Belohnung gibt.

Die Umgebung ist durch folgende Gleichung definiert

$$H(x) = \begin{cases} x^2 + x, & \text{wenn } x < 0 \\ \frac{x}{\sqrt{1+5x^2}}, & \text{wenn } x \geq 0 \end{cases} \tag{C.1}$$

dabei ist $H(x)$ die Höhe an der Position x. Die Darstellung dieser Umgebung ist in Abbildung C.1 zu finden.

Begrenzt ist das System wie folgt.

- Position $-1 \leq x \leq 1$
- Geschwindigkeit $-2 \leq \dot{x} \leq 2$
- einwirkende Kraft $-4 \leq F \leq 4$
- Abtastrate von 0.2s

Das Randproblem, also wenn der Wagen über Positionsbeschränkung hinausfahren würde, wurde so gehandhabt, als ob dort eine Wand wäre.

Das heißt, dass die Position beibehalten und die Geschwindigkeit auf null gesetzt wurde.

Ein Simulationsschritt mit $\Delta t = 0.2$ berechnet sich wie folgt:

$$x' = x + \dot{x}\Delta t + \ddot{x}\frac{\Delta t^2}{2} \tag{C.2}$$

$$\dot{x}' = \dot{x} + \ddot{x}\Delta t \tag{C.3}$$

wobei:

$$\ddot{x} = \frac{F}{M\sqrt{1 + (H'(x))^2}} - \frac{gH'(x)}{1 + (H'(x))^2} \tag{C.4}$$

M ist in Gleichung C.4 die Masse des Fahrzeugs mit einem Wert von 1 und g die Fallbeschleunigung von 9.81. $H'(x)$ ist der Anstieg der Umgebung mit $H'(x) = \frac{d}{dx}H(x)$. Bei der Geschwindigkeit wird hier statt v $\dot{x}$ geschrieben und die Beschleunigung entsprechend als $\ddot{x}$, um die physikalischen Zusammenhänge hier einfacher darzustellen.

C.2. Kraftwerksimulator

Für die in Kapitel 6 vorgestellte Anwendung war es notwendig, einen Simulator zu verwenden, der die Besonderheiten der Regelung eines kohlegefeuerten Ofens zumindest qualitativ nachbildet. Entwickelt wurde der verwendete Simulator im Rahmen des SOFCOM Projektes [Rosner et al., 2008], [Funkquist et al., 2009], [Funkquist et al., 2011] von der Powitec GmbH und Vattenfall R&D.

C.2.1. Simulation einer Brennerebene

Die Simulation einer einzelnen Brennerebene, von denen es je nach Größe des Kraftwerks unterschiedlich viele geben kann, ist die kleinste sinnvolle Einheit, in der das Verbrennungsproblem simuliert und geregelt

werden kann. Eine Brennerebene besteht dabei aus zwei Brennern, die von einer einzelnen Kohlemühle gespeist werden. Der Simulator berechnet daraus das Abgasgemisch, welches neben Schadstoffen, die minimiert werden sollen, auch unverbrauchten Sauerstoff (Rest-O_2) enthält.

Die wichtigste Größe für diese Simulation ist der sogenannte Lambda-Faktor λ. Er gibt das Verhältnis von Sauerstoff zu Kohle für einen Brenner an. In der Theorie wäre ein Verhältnis von einem Kohlenstoffatom zu zwei Sauerstoffatomen anzustreben um daraus ein Kohlendixoidmolekül zu bilden. Dies würde $\lambda = 1$ entsprechen. Ist weniger Sauerstoff vorhanden ($\lambda < 1$) führt dies zu unvollständigem Verbrennen und damit zu Kohlenmonoxid. Mehr Sauerstoff ($\lambda > 1$) bedeutet, dass während der Verbrennung der überzählige Sauerstoff mit erhitzt würde, was einer Effizienzminderung gleich kommt. Allerdings schützt überzähliger Sauerstoff den Ofen vor Korrosion, so dass praktisch gesehen, für ein Kohlekraftwerk Werte von rund $\lambda = 1.15$ als untere Schranke normal sind. Diesen Wert nach oben zu begrenzen, liegt im Interesse eines hohen Wirkungsgrades.

$$\lambda_{links} = \frac{v_{Luft} M_{Luft}}{v_{Kohle} M_{Kohle}}$$

v_{Luft} ist ein Wert zwischen 0 und 1 und gibt das Verhältnis der Verteilung zwischen linkem und rechtem Brenner an. M_{Luft} gibt dabei die Gesamtmenge an Luft für beide Brenner an und unterliegt einer systematischen Fluktuation, welche durch den Vorerhitzer, welcher die Tragluft erhitzt, entsteht. Analog dazu finden sich im Nenner des Bruchs dieselben Größen auf die Kohle bezogen. Übertragen auf den zweiten Brenner ergibt sich

$$\lambda_{rechts} = \frac{(1 - v_{Luft}) M_{Luft}}{(1 - v_{Kohle}) M_{Kohle}}.$$

Daraus können nun die relevanten Größen berechnet werden. Dazu gehören die Temperatur T und der Sauerstoffgehalt M_{O2}, sowie davon abgeleitet der Kohlenmonoxid- (M_{CO}) und Stickoxidanteil (M_{NOX}).

$$T_{links} = max(300, \theta(\lambda_{links})) - c + F \tag{C.5}$$

$$M_{O2,links} = max(0, 21 - \frac{21}{\lambda_{links}} \tag{C.6}$$

$$M_{CO,links} = \psi(M_{O2,links}) \tag{C.7}$$

$$M_{NOX,links} = v_{Luft} M_{Luft} \varphi(T_{links} + \frac{1800}{c}) \tag{C.8}$$

Die Größe c ist dabei ein Faktor, der die flüchtigen Bestandteile beschreibt und von der Kohlesorte abhängig ist. F steht für den Grad der Verschmutzung (Fouling) im Ofen. Die Funktionen ψ, φ und θ werden auf Basis von Spline-interpolierten Stützstellen berechnet. Diese Funktionen sind dabei unter Beachtung der physikalischen Zusammenhänge und des realen, beobachtbaren Verhaltens im Kraftwerk gewählt worden. Die Werte der verwendeten Stützpunkte kann dabei aus der Tabelle C.1 abgelesen werden.

Die bisher berechneten Werte dienen als interner Prozesszustand, und können nicht direkt beobachtet werden. Als Beobachtungen werden vom Simulator folgende Größen berechnet:

$$T_{links,gemessen} = T_{links} * \left(1 - \frac{D}{100}\right) + \sigma_T \tag{C.9}$$

$$M_{O2,gemessen} = \frac{1}{2} \left(M_{O2,links} + M_{O2,rechts}\right) \sigma_{O2} \tag{C.10}$$

$$M_{CO,gemessen} = \frac{1}{2} \left(M_{CO,links} + M_{CO,rechts}\right) \sigma_{CO} \tag{C.11}$$

$$M_{NOX,gemessen} = \frac{1}{2} \left(M_{NOX,links} + M_{NOX,rechts}\right) \sigma_{NOX} \tag{C.12}$$

M_{O2} in Prozent	0	1	3	5	7	10
$\psi(M_{O2})$ in mg/m^3	600	200	30	15	8	5

Tabelle C.1.: Stützstellen für die Funktion ψ. Diese Funktion modelliert den Zusammenhang zwischen dem Sauerstoffgehalt im Ofen und dem resultierenden Kohlenmonoxid. Je weniger überschüssiger Sauerstoff vorhanden ist, desto größer ist die Gefahr, dass statt Kohlendioxid Kohlenmonoxid entsteht.

T in °C	200	500	1000	1200	1400
$\varphi(T)$ in mg/m^3	0	0	100	200	500

Tabelle C.2.: Stützstellen für die Funktion φ. Diese Funktion modelliert den Zusammenhang zwischen der Flammentemperatur im Ofen und dem resultierenden Stickoxidausstoß. Je heißer der Ofen ist, desto mehr Stickoxide entstehen bei der Verbrennung.

λ	0	0.3	0.6	0.8	0.95	1.0	1.05	1.2	2.0
$\theta(\lambda)$	100	100	200	600	1350	1400	1340	1130	700

Tabelle C.3.: Stützstellen für die Funktion θ (angegeben in Grad Celcius) . Diese Funktion modelliert den Zusammenhang zwischen dem Kohle-Luft Verhältnis und der Flammentemperatur im Ofen. Die Verbrennung ist am heißesten, wenn das Verhältnis genau 1:1 ist. Bei einem Überschuss von Kohle oder Sauerstoff ist die Temperatur geringer.

D steht hierbei für die Verschmutzung des Sensors zur Temperaturmessung: je größer der Verschmutzungsgrad, desto größer wird der Fehler zur echten Temperatur. Die verschiedenen σ-Terme stellen normalverteiltes Rauschen dar. Die gemessenen Größen für Sauerstoff M_{O2}, Kohlenmonoxid M_{CO} und Stickoxide M_{NOX} sind nicht am einzelnen Brenner, sondern nur für die gesamte Ebene bestimmbar.

Die eigentliche Schwierigkeit ist die Mehrdeutigkeit des Prozesses. Wenn die beobachtete Temperatur niedrig ist, kann dies zwei Gründe haben. Entweder ist zu viel Luft, als Aktionsgröße, am Brenner

oder zu viel Kohle, welche nicht messbar ist. Je nach Ursache sind zwei zueinander gegensätzliche Aktionen notwendig. Die Erhöhung des Zustandsraums um die Stickoxid-, Kohlenmonoxid- und Restsauerstoffinformation löst diese teilweise auf.

Die eben benannten Größen bilden den Zustandsraum $\mathcal{S}$. Der Aktionsraum $\mathcal{A}$ ist eindimensional und beschreibt die Luftverteilung v_{Luft} zwischen dem linken und rechten Brenner. Die Zustandsübergangsfunktion $\mathcal{P}$ ist deterministisch und berechnet sich nach den aufgeführten Formeln. Zu beachten ist hierbei, dass hier nicht versucht wird den eigentlichen Verbrennungsprozess zu modellieren. Dazu wären komplexe Differentialgleichungssysteme als Zustandsübergangsmodell notwendig. Stattdessen wird nur das typische Verhalten simuliert, welches im Kern auf der nicht beobachtbaren Kohleverteilung v_{Kohle} und der Luftverteilung v_{Luft} basiert.

Die Rewardfunktion $\mathcal{R}$ ist so gestaltet, dass möglichst die Schadstoffe (Stickoxid und Kohlenmonoxid) reduziert werden und der Wirkungsgrad maximiert werden soll. Als Kenngröße des Wirkungsgrades dient hierbei der Restsauerstoff, welcher auch minimiert werden muss.

$$R_{NOX} = -\max\left(0, \frac{NOX - 950}{15}\right) \tag{C.13}$$

$$R_{CO} = -\max\left(0, \frac{CO - 45}{20}\right) \tag{C.14}$$

$$R_{O_2} = -O_2 \tag{C.15}$$

$$R_{kombiniert} = R_{NOX} + R_{CO} + R_{O_2} \tag{C.16}$$

Ein beispielhafte Erläuterung der Größen findet sich in Abbildung C.2.

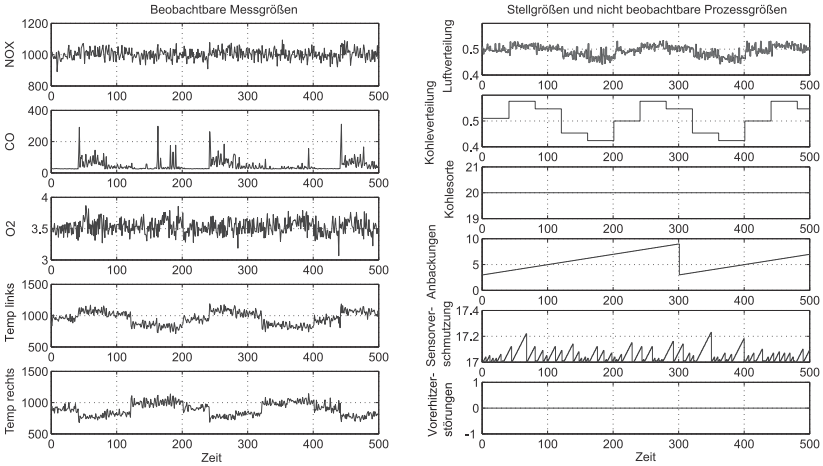


Abbildung C.2.: Darstellung der wichtigsten Größen im Kraftwerksimulator. Links sind die beobachtbaren Größen aufgetragen. Restsauerstoff, Stickoxide und Kohlenmonoxid (O2, NOX und CO) sind die Größen aus denen sich der Reward ableitet. Die Temperatur wird für die linke als auch die rechte Hälfte des Ofens gemessen. Rechts oben ist die Stellgröße Luftverteilung gezeigt, welche im besten Fall der unebkannten Kohleverteilung (darunter) entspricht. Darunter befinden sich verschiedene Störgrößen, welche das Problem erschweren. Dabei handelt es sich um andere Kohlesorten (geänderte Verbrennungseigenschaften), Anbackungen im Ofen (Änderungen im Prozess) und Verschmutzung der Sensoren (Änderung der Wahrnehmung), welche durch Säuberungszyklen ein Sägezahnprofil haben, und eine systematische Störung (Vorerhitzer), welche den Luftstrom verändert. Um das Problem zu verkomplizieren, können weitere Ebenen hinzugefügt werden. Dabei bleiben die Störungen und die Schadstoffe gleich (diese gelten global für den gesamten Ofen), während Temperatur-, Luft- und Kohleverteilungen als neue Größen für die zusätzliche Ebene hinzukommen.

C.2.2. Simulation mehrerer Brennerebenen

Die Dimensionalität des Problems kann beliebig erweitert werden. Jede simulierte zusätzliche Brennerebene erhöht den Zustandsraum um die Dimension zwei und den Aktionsraum um eine Dimension. Die Erweiterung des Zustandsraums sind dabei die Temperaturen auf der neuen Ebene, wieder jeweils links und rechts. Zusätzlich gibt es eine neue, nicht beobachtbare Größe, die Kohleverteilung auf dieser Ebene. Als Stellgröße kommt die Verteilung der Luft auf der neuen Ebene hinzu. Zusätzliche Ebenen erschweren das Gesamtproblem damit deutlich.

Die Berechnung des Restsauerstoffs, des Kohlenmonoxids und der Stickoxide (Gleichung C.10 bis C.12) wird erweitert durch eine einfache Summierung über alle Ebenen. Dies resultiert aus der Tatsache, dass diese Größen erst im Abgas am Schornstein bestimmt werden können. Real auftretende, komplexe Wechselwirkungen zwischen den einzelnen Ebenen werden nicht modelliert.

Literaturverzeichnis

[Aliferis et al., 2010] Aliferis, Constantin F., A. Statnikov, I. Tsamardinos, S. Mani und X. D. Koutsoukos (2010). *Local causal and Markov blanket induction for causal discovery and feature selection for classification*. Journal of Machine Learning Research, S. 171–284.

[Anderson et al., 2004] Anderson, J. R., D. Bothell, M. D. Byrne, S. Douglass, C. Lebiere und Y. Qin (2004). *An integrated theory of the mind.*. Psychol Rev, 111(4):1036–1060.

[Arkin, 1998] Arkin, Ronald C. (1998). *Behavior-Based Robotics*. MIT Press.

[Asuncion und Newman, 2007] Asuncion, A. und D. Newman (2007). *UCI Machine Learning Repository*. http://archive.ics.uci.edu/ml/.

[Atkeson, 2007] Atkeson, Christopher G. (2007). *Randomly Sampling Actions in Dynamic Programming*. In: *Proceedings of the 2007 IEEE Symposium on Approximate Dynamic Programming and Reinforcement Learning (ADPRL), 2007*, S. 185–192.

[Barth, 2008] Barth, Ch. (2008). *Vergleich von Reinforcement Learning Verfahren in kontinuierlichen Zustands-Aktions-Räumen*. Diplomarbeit, Technische Universität Ilmenau, Fachgebiet Neuroinformatik und Kognitive Robotik.

[Bastiaans et al., 2005] Bastiaans, R. J. M., J. Martin, H. Pitsch, A. van Oijen und L. P. H. de Goey (2005). *Flamelet Analysis of Turbulent Combustion*. In: *International Conference on Computational Science*, S. 64–71.

[Battiti, 1994] Battiti, Roberto (1994). *Using mutual information for selecting features in supervised neural net learning*. IEEE Transactions on Neural Networks, 5:537–550.

[Bellman, 1957] Bellman, R.E. (1957). *Dynamic programming*. Rand Corporation research study. Princeton University Press.

[Berry und Fristedt, 1985] Berry, Donald A. und B. Fristedt (1985). *Bandit Problems: Sequential Allocation of Experiments (Monographs on Statistics and Applied Probability)*. Springer.

[Bhattacharyya, 1943] Bhattacharyya, A. (1943). *On a measure of divergence between two statistical populations defined by their probability distributions.*. Bull. Calcutta Math. Soc., 35:99 – 109.

[Bishop, 2006] Bishop, C. M. (2006). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer.

[Bonachela et al., 2008] Bonachela, J. A., H. Hinrichsen und M. A. Munoz (2008). *Entropy estimates of small data sets*. Journal of Physics A: Mathematical and Theoretical, 41(20):1–9.

[Bonasso et al., 1997] Bonasso, R. Peter, D. Kortenkamp und T. Whitney (1997). *Using a robot control architecture to automate space shuttle operations*. In: *Proceedings of the fourteenth national conference on artificial intelligence and ninth conference on Innovative applications of artificial intelligence*, AAAI'97/IAAI'97, S. 949–956. AAAI Press.

[Breiman, 2001] Breiman, Leo (2001). *Random forests*. In: *Machine Learning*, S. 5–32.

[Brooks, 1986] Brooks, R. (1986). *A robust layered control system for a mobile robot*. Robotics and Automation, IEEE Journal of, 2(1):14–23.

[Brückmann et al., 2007] Brückmann, Robert, A. Scheidig und H.-M. Gross (2007). *Adaptive Noise Reduction and Voice Activity Detection for improved Verbal Human-Robot Interaction using Binaural Data*. In: *ICRA*, S. 1782–1787.

[Camacho und Bordons Alba, 2004] Camacho, Eduardo F. und C. Bordons Alba (2004). *Model Predictive Control*. Springer Verlag.

[Cellucci et al., 2005] Cellucci, C. J., A. M. Albano und P. E. Rapp (2005). *Statistical validation of mutual information calculations: Comparison of alternative numerical algorithms*. Physical Review E, 71(6):066208.

[Chang et al., 2003] Chang, Yu-Han, T. Ho und L. P. Kaelbling (2003). *All Learning is Local: Multi-agent learning in global reward games*. In: *NIPS*.

[Chow und Liu, 1968] Chow, C.K. und C. Liu (1968). *Approximating Discrete Probability Distributions with Dependence Trees*. IEEE Transactions on Information Theory, 14:462–467.

[Chow und Huang, 2005] Chow, T. W. und D. Huang (2005). *Estimating Optimal Feature Subsets Using Efficient Estimation of High-Dimensional Mutual Information*. IEEE Transactions on Neural Networks, 16:213–224.

[Cignoli et al., 2001] Cignoli, Francesco, S. D. Iuliis, V. Manta und G. Zizak (2001). *Two-Dimensional Two-Wavelength Emission Technique for Soot Diagnostics*. Appl. Opt., 40(30):5370–5378.

[Cochran, 1954] Cochran, W. G. (1954). *Some methods for strengthening the common χ^2 test*. Biometrics, 10:417–451.

[Cootes et al., 1998] Cootes, Timothy F., G. J. Edwards und C. J. Taylor (1998). *Active Appearance Models*. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence*, S. 484–498.

[Cover und Thomas, 2006] Cover, Thomas M. und J. A. Thomas (2006). *Elements of Information Theory, Second Edition*. John Wiley and Sons, Inc.

[Darbellay und Vajda, 1999] Darbellay, G. A. und I. Vajda (1999). *Estimation of the Information by an Adaptive Partitioning of the Observation Space*. IEEE Transactions on Information Theory, 45(4):1315–1321.

[Das, 2001] Das, Sanmay (2001). *Filters, Wrappers and a Boosting-Based Hybrid for Feature Selection*. In: *Inter. Conf. on Machine Learning ICML*, S. 74–81.

[Debuse und Rayward-Smith, 1997] Debuse, Justin C. W. und V. J. Rayward-Smith (1997). *Feature Subset Selection within a Simulated Annealing DataMining Algorithm*. J. Intell. Inf. Syst., 9(1):57–81.

[Deisenroth, 2009] Deisenroth, Marc (2009). *Efficient Reinforcement Learning using Gaussian Processes*. Doktorarbeit, TU Karlsruhe.

[Deisenroth et al., 2008] Deisenroth, Marc P., C. E. Rasmussen und J. Peters (2008). *Approximate Dynamic Programming with Gaussian Processes*. In: *American Control Conference*.

[Dietterich, 2000] Dietterich, T.G. (2000). *Ensemble Methods in Machine Learning*. In: *Int. Workshop on Multiple Classifier Systems*, S. 1–15. Springer-Verlag.

[Doane, 1976] Doane, D.P. (1976). *Aesthetic frequency classification*. American Statistician, 30:181–183.

[Docquier und Candel, 2002] Docquier, Nicolas und S. Candel (2002). *Combustion control and sensors: a review*. Progress in Energy and Combustion Science, 28(2):107 – 150.

[Doya, 2000] Doya, Kenji (2000). *Reinforcement Learning In Continuous Time and Space*. Neural Computation, 12:219–245.

[Eisenbach, 2009] Eisenbach, M. (2009). *Rewarddekomposition für Multiagentensysteme bei komplexen Regelungsprozessen*. Diplomarbeit, Technische Universität Ilmenau, Fachgebiet Neuroinformatik und Kognitive Robotik.

[Engel et al., 2003] Engel, Yaakov, S. Mannor und R. Meir (2003). *Bayes Meets Bellman: The Gaussian Process Approach to Temporal Difference Learning*. In: *Proc. of the 20th International Conference on Machine Learning*, S. 154–161.

[Estevez et al., 2009] Estevez, P.A., M. Tesmer, C. Perez und J. Zurada (2009). *Normalized Mutual Information Feature Selection*. IEEE Transactions on Neural Notworks, 20:189–201.

[Fahlman und Lebiere, 1990] Fahlman, S. E. und C. Lebiere (1990). *The cascade-correlation learning architecture*. In: *Advances in neural information processing systems (NIPS) 2*, S. 524–532, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

[Fisher, 1936] Fisher, R.A. (1936). *The use of multiple measurements in taxonomic problems*. Annals of Eugenics, 7:179–188.

[Fleuret, 2004] Fleuret, F. (2004). *Fast Binary Feature Selection with Conditional Mutual Information*. Journal of Machine Learning Research, 5:1531–1555.

[Flynn, 2003] Flynn, D., Hrsg. (2003). *Thermal Power Plant Simulation and Control*. IEE London.

[Fraser und Swinney, 1986] Fraser, A. M. und H. L. Swinney (1986). *Independent coordinates for strange attractors from mutual information*. Physical Review A, 33(2):1134–1140.

[Freedman und Diaconis, 1981] Freedman, D. und P. Diaconis (1981). *On this histogram as a density estimator: L2 theory*. Probability Theory and Related Fields, 57(4):453–476.

[Freund und Schapire, 1995] Freund, Yoav und R. E. Schapire (1995). *A decision-theoretic generalization of on-line learning and an application to boosting*. In: *EuroCOLT '95: Proceedings of the Second European Conference on Computational Learning Theory*, S. 23–37, London, UK. Springer-Verlag.

[Fukunaga, 1990] Fukunaga, Keinosuke (1990). *Introduction to statistical pattern recognition (2nd ed.)*. Academic Press Professional, Inc., San Diego, CA, USA.

[Funkquist et al., 2009] Funkquist, J., V. Stephan, E. Schaffernicht und C. Rosner (2009). *SOFCOM - Self-Optimising Strategy for Control of the Combustion Process*. Technischer Bericht, Vattenfall Research and Development AB, Stockholm, Sweden.

[Funkquist et al., 2011] Funkquist, J., V. Stephan, E. Schaffernicht, C. Rosner und M. Berg (2011). *SOFCOM - Self-optimising strategy for control of the combustion process*. VGB PowerTech Journal, 8(3):48–54.

[Gomez et al., 2006] Gomez, F., J. Schmidthuber und R. Miikkulainen (2006). *Efficient Non-Linear Control through Neuroevolution*. In: *Proceedings of the European Conference on Machine Learning*, S. 654–662.

[Gomez et al., 2008] Gomez, F., J. Schmidthuber und R. Miikkulainen (2008). *Accelerated Neural Evolution through Cooperatively Coevolved Synapses*. Journal of Machine Learning Research, 9:937–965.

[Grancharova et al., 2008] Grancharova, Alexandra, J. Kocijan und T. A. Johansen (2008). *Explicit stochastic predictive control of combustion plants based on Gaussian process models*. Automatica, 44:1621–1631.

[Granger, 1969] Granger, C.W.J. (1969). *Investigating causal relations by econotetric models and cross-spectral methods*. Econometrica, 37(3):424–438.

[Görner, 2003] Görner, K. (2003). *Waste Incineration European State of the Art and New Developments*. IFRF Combustion Journal, 03.

[Gross et al., 2009] Gross, H.-M., H. Boehme, C. Schroeter, S. Mueller, A. Koenig, E. Einhorn, C. Martin, M. Merten und A. Bley (2009). *TOOMAS: interactive shopping guide robots in everyday use - final implementation and experiences from long-term field trials*. In: *Proceedings of the 2009 IEEE/RSJ international conference on Intelligent robots and systems*, IROS'09, S. 2005–2012, Piscataway, NJ, USA. IEEE Press.

[Guiasu, 1977] Guiasu, S. (1977). *Information Theory with Applications*. McGraw-Hill Inc., New York, USA.

[Guyon et al., 2006] Guyon, Isabell, S. Gunn, M. Nikravesh und L. Zadeh (2006). *Feature Extraction: Foundations and Applications*, Bd. 207 d. Reihe *Studies in fuzziness and soft computing*. Springer Verlag.

[Guyon und Elisseeff, 2003] Guyon, Isabelle und A. Elisseeff (2003). *An introduction to variable and feature selection*. Journal Machine Learning Research, 3:1157–1182.

[Guyon et al., 2002] Guyon, Isabelle, J. Weston, S. Barnhill und V. Vapnik (2002). *Gene Selection for Cancer Classification using Support Vector Machines*. Mach. Learn., 46(1-3):389–422.

[Hafner, 2009] Hafner, Roland (2009). *Dateneffiziente selbstlernende neuronale Regler*. Doktorarbeit, Universität Osnabrück.

[Hellwig, 2009] Hellwig, S. (2009). *Policy Iteration für die intelligente Regelung unter Berücksichtigung des Stabilitäts-Plastizitäts-Dilemmas*. Diplomarbeit, Technische Universität Ilmenau, Fachgebiet Neuroinformatik und Kognitive Robotik und Powitec GmbH.

[Hyvärinen et al., 2001] Hyvärinen, A., J. Karhunen und E. Oja (2001). *Independent Component Analysis*. Wiley, New York, USA.

[Hyvärinen et al., 2010] Hyvärinen, Aapo, K. Zhang und S. Shimizu (2010). *Estimation of a Strucutral Vector Autoregressive Model Using Non-Gaussianity*. J. Mach. Learn. Res., 11:1709–1731.

[Iwata et al., 2004] Iwata, K., K. Ikeda und H. Sakai (2004). *Asymptotic equipartition property on empirical sequence in reinforcement learning*. In: *Proceedings of the 2nd IASTED International Conference on Neural Networks and Computational Intelligence, Grindelwald, Switzerland*, S. 90–95.

[Jennings, 1994] Jennings, N. R. (1994). *Cooperation in industrial multi-agent systems*. World Scientific Publishing Co., Inc., River Edge, NJ, USA.

[Jordan, 1998] Jordan, M., Hrsg. (1998). *Learning in Graphical Models*. MIT Press.

[Jung und Stone, 2010] Jung, Tobias und P. Stone (2010). *Gaussian processes for sample efficient reinforcement learning with RMAX-like exploration*. In: *Proceedings of the 2010 European conference on Machine learning and knowledge discovery in databases: Part I*, ECML PKDD'10, S. 601–616, Berlin, Heidelberg. Springer-Verlag.

[Kaltenhäuser, 2010] Kaltenhäuser, R. (2010). *Schätzung von Transinformation aus Daten*. Diplomarbeit, Technische Universität Ilmenau, Fachgebiet Neuroinformatik und Kognitive Robotik.

[Kearns et al., 2002] Kearns, Michael, Y. Mansour und A. Y. Ng (2002). *A Sparse Sampling Algorithm for Near-Optimal Planning in Large Markov Decision Processes*. Machine Learning, 49:193–208.

[Khan et al., 2007] Khan, S., S. Bandyopadhyay, A. R. Ganguly, S. Saigal, D. J. Erickson, V. Protopopescu und G. Ostrouchov (2007). *Relative performance of mutual information estimation methods for quantifying the dependence among short and noisy data*. Physical Review E, 76:026209.

[Khare et al., 2005] Khare, V.-R., X. Yao, B. Sandhoff, Y. Jin und H. Wersing (2005). *Co-evolutionary Modular Neural Networks for Automatic Problem Decomposition*. In: *Proceedings of IEEE Conference on Evolutionary Computation*, S. 2691–2698.

[Kleppmann, 2006] Kleppmann, Wilhelm (2006). *Taschenbuch Versuchsplanung*. Carl Hanser Verlag München Wien.

[Ko et al., 2007] Ko, J., D. Klein, D. Fox und D. Haehnel (2007). *Gaussian Processes and Reinforcement Learning for Identification and Control of an Autonomous Blimp*. In: *Robotics and Automation, 2007 IEEE International Conference on*, S. 742–747.

[Kohavi und John, 1997] Kohavi, Ron und G. H. John (1997). *Wrappers for feature subset selection*. Artificial Intelligence, 97(1-2):273–324.

[Koller und Sahami, 1996] Koller, Daphne und M. Sahami (1996). *Toward Optimal Feature Selection*. In: *International Conference on Machine Learning*, S. 284–292.

[Kortenkamp und Simmons, 2008] Kortenkamp, D. und R. Simmons (2008). *Springer Handbook of Robotics*, Kap. Robotic Systems Architectures and Programming, S. 187–206. Springer Verlag.

[Kozachenko und Leonenko, 1987] Kozachenko, L. F. und N. N. Leonenko (1987). *Sample Estimate of the Entropy of a Random Vector*. Problems of Information Transmission, 23(2):95–101.

[Kramer, 1991] Kramer, M.A. (1991). *Nonlinear principal component analysis using autoassociative neural networks*. AIChE Journal, 37:233–243.

[Kraskov et al., 2004] Kraskov, Alexander, H. Stögbauer und P. Grassberger (2004). *Estimating mutual information*. Phys. Rev. E, 69(6):066138.

[Krause und Guestrin, 2007] Krause, Andreas und C. Guestrin (2007). *Nonmyopic active learning of Gaussian processes: an exploration-exploitation approach*. In: *Proceedings of the 24th international conference on Machine learning*, ICML '07, S. 449–456, New York, NY, USA. ACM.

[Kruskal, 1956] Kruskal, Joseph B. (1956). *On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem*. Proceedings of the American Mathematical Society, 7(1):48–50.

[Kschischang et al., 2001] Kschischang, F. R., B. J. Frey und H. Loeliger (2001). *Factor Graphs and the Sum-Product Algorithm*. IEEE Transactions on Information Theory, 47(2):498–519.

[Kuss, 2006] Kuss, Malte (2006). *Gauß-Prozess Modelle zur Robusten Regressionsanalyse, Klassifikation und Reinforcement Lernen*. Doktorarbeit, TU Darmstadt.

[Kullback, 1959] Kullback, S. (1959). *Information Theory and Statistics*. Wiley, New York.

[Kwak und Choi, 1999] Kwak, N. und C. Choi (1999). *Information Feature Selector for Neural Networks in Supervised Learning*. In: *Int. Joint Conf. on Neural Networks (IJCNN 99)*, S. 1313–1318.

[Kwak und Choi, 2002] Kwak, N. und C. H. Choi (2002). *Input feature selection by mutual information based on Parzen window*. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 24(12):1667 – 1671.

[Langley et al., 2009] Langley, P., J. Laird und S. Rogers (2009). *Cognitive Architectures: Research Issues and Challenges*. Cognitive Systems Research, 10:141–160.

[Langley, 1994] Langley, Pat (1994). *Selection of Relevant Features in Machine Learning*. In: *In Proceedings of the AAAI Fall Symposium on Relevance*, S. 140–144. AAAI Press.

[Le Cun et al., 1990] Le Cun, Yann, J. S. Denker und S. A. Solla (1990). *Optimal brain damage*. In: *Advances in neural information processing systems (NIPS) 2*, S. 598–605, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

[Lee und Seung, 2000] Lee, Daniel und H. S. Seung (2000). *Algorithms for Non-negative Matrix Factorization*. In: *Advances in neural information processing systems (NIPS)*, Bd. 13, S. 556–562. MIT Press (2001).

[Leung und Hung, 2010] Leung, Yukyee und Y. Hung (2010). *A Multiple-Filter-Multiple-Wrapper Approach to Gene Selection and Microarray Data Classification*. IEEE/ACM Transactions on Computational Biology and Bioinformatics, 7:108–117.

[Lu et al., 2005] Lu, G., G. Gilabert und Y. Yan (2005). *Vision based monitoring and characterisation of combustion flames*. Journal of Physics: Conference Series, 15(1):194.

[Marques und Jorge, 2000] Marques, Jorge S. und P. M. Jorge (2000). *Visual inspection of a combustion process in a thermoelectric plant*. Signal Processing, 80(8):1577–1589.

[Marthi, 2007] Marthi, Bhaskara (2007). *Automatic shaping and decomposition of reward functions*. In: *Proceedings of the 24th international conference on Machine learning*, ICML '07, S. 601–608.

[Martin et al., 2006] Martin, Christian, E. Schaffernicht, A. Scheidig und H.-M. Gross (2006). *Multi-modal sensor fusion using a probabilistic aggregation scheme for people detection and tracking.*. Robotics and Autonomous Systems, 54(9):721–728.

[Martinez und Kak, 2001] Martinez, A.M. und A. Kak (2001). *PCA versus LDA*. IEEE Transactions on Pattern Analysis and Machine Intelligence 23, 23:228–233.

[Mataric und Michaud, 2008] Mataric, M. und F. Michaud (2008). *Springer Handbook of Robotics*, Kap. Behaviour-Based Systems, S. 891–909. Springer Verlag.

[Metz et al., 2005] Metz, B., O. Davidson, H. de Coninck, M. Loos und L. Meyer, Hrsg. (2005). *Carbon Dioxide Capture and Storage*. Intergovernmental Panel on Climate Change, Cambridge University Press, New York, USA.

[Mühlhaus et al., 1999] Mühlhaus, R., K. Görner, R. Heitmüller, W. Moll und K. Pflipsen (1999). *Feuerungsanalyse und -optimierung mit Neuronalen Netzen*. In: *VDI-Gesellschaft Energietechnik: Verbrennungen und Feuerungen - 19. Flammtag*, S. 1321–28.

[Müller, 2000] Müller, Bernd (2000). *Innovative Prozeßführung in der thermischen Abfallbehandlung mit Künstlichen Neuronalen Netzen*. Doktorarbeit, Universität Karlsruhe(TH).

[Möller, 2009] Möller, Ch. (2009). *Prädiktion von Schnittregisterfehlern an Illustrationsmaschinen auf Basis von Messdaten einer Buchdruckmaschine*. Diplomarbeit, Technische Universität Ilmenau, Fachgebiet Neuroinformatik und Kognitive Robotik und MANroland Augsburg.

[Müller et al., 2008] Müller, St., S. Hellbach, E. Schaffernicht, A. Ober, A. Scheidig und H.-M. Gross (2008). *Whom to talk to? Estimating user interest from movement trajectories*. In: *Proc. of the 17th IEEE Int. Symposium on Robot and Human Interactive Communication, (RO-MAN 08)*, S. 532–538, Munich, Germany. IEEE Omnipress.

[Montgomery, 2004] Montgomery, Douglas C. (2004). *Design and Analysis of Experiments*. Wiley, New York.

[Moody und Darken, 1989] Moody, John und C. J. Darken (1989). *Fast learning in networks of locally-tuned processing units*. Neural Comput., 1(2):281–294.

[Moon et al., 1995] Moon, Young-Il, B. Rajagopalan und U. Lall (1995). *Estimation of mutual information using kernel density estimators*. Phys. Rev. E, 52(3):2318–2321.

[Moore und Atkeson, 1995] Moore, Andrew W. und C. G. Atkeson (1995). *The Parti-game Algorithm for Variable Resolution Reinforcement Learning in Multidimensional State-spaces*. Machine Learning, 21(3):199–233.

[Moriarty und Miikkulainen, 1996] Moriarty, David E. und R. Miikkulainen (1996). *Efficient reinforcement learning through symbiotic evolution*. Machine Learning, 22:11–32.

[Narendra und Thathachar, 1989] Narendra, Kumpati S. und M. A. L. Thathachar (1989). *Learning Automata: An Introduction*. Prentice Hall.

[Neal und Zhang, 2006] Neal, R. M. und J. Zhang (2006). *High dimensional classification with Bayesian neural networks and Dirichlet diffusion trees*, Bd. 207 d. Reihe *Studies in Fuzziness and Soft Computing*, S. 265–295. Springer Berlin / Heidelberg.

[Neal, 1996] Neal, Radford M. (1996). *Bayesian Learning for Neural Networks*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

[Neal, 2003] Neal, Radford M. (2003). *Density Modeling and Clustering Using Dirichlet Diffusion Trees*. In: *Bayesian Statistics 7: Proceedings of the Seventh Valencia International Meeting*, S. 619–629.

[Nguyen-Tuong et al., 2008] Nguyen-Tuong, Duy, M. Seeger und J. Peters (2008). *Local Gaussian Process Regression for Real Time Online Model Learning*. In: *NIPS*, S. 1193–1200.

[Niegowski, 2007] Niegowski, R. (2007). *Selbstorganisierende Merkmalsextraktion durch adaptive Datenfilter*. Diplomarbeit, Technische Universität Ilmenau, Fachgebiet Neuroinformatik und Kognitive Robotik.

[Nissen, 1997] Nissen, Volker (1997). *Einführung in Evolutionäre Algorithmen - Optimierung nach dem Vorbild der Evolution*. Vieweg Verlag.

[Nof, 2009] Nof, Shimon Y., Hrsg. (2009). *Springer Handbook of Automation*. Springer.

[Ober, 2007] Ober, A. (2007). *Analyse von Bewegungstrajektorien zur nutzerangepassten Dialoginitiierung*. Diplomarbeit, Technische Universität Ilmenau, Fachgebiet Neuroinformatik und Kognitive Robotik.

[Ogunnaike und Ray, 1994] Ogunnaike, B.A. und W. Ray (1994). *Process Dynamics, Modeling and Control*. Oxford University Press.

[Paeschke, 2003] Paeschke, Astrid (2003). *Prosodische Analyse emotionaler Sprechweise*. Logos Verlag, Berlin.

[Panait und Luke, 2005] Panait, Liviu und S. Luke (2005). *Cooperative Multi-Agent Learning: The State of the Art*. Autonomous Agents and Multi-Agent Systems, 11(3):387–434.

[Paninski, 2003] Paninski, Liam (2003). *Estimation of entropy and mutual information*. Neural Computation, 15(6):1191–1253.

[Pearl, 1988] Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann.

[Pearson, 1901] Pearson, K. (1901). *On lines and planes of closest fit to systems of points in space*. Philosophical Magazine, 2:559–572.

[Peters und Schaal, 2008] Peters, Jan und S. Schaal (2008). *Natural Actor-Critic*. Neurocomputing, 71(7-9):1180–1190.

[Pólya, 1930] Pólya, G. (1930). *Sur quelques points de la théorie des probabilités*. Annals of the Institute of Henri Poincaré, 1:117 – 161.

[Poupart et al., 2006] Poupart, Pascal, N. Vlassis, J. Hoey und K. Regan (2006). *An analytic solution to discrete Bayesian reinforcement learning*. In: *Proceedings of the 23rd international conference on Machine learning*, ICML '06, S. 697–704, New York, NY, USA. ACM.

[Prüger, 2008] Prüger, T. (2008). *Audiobasierte Merkmale für die multimodale Nutzermodellierung*. Diplomarbeit, Technische Universität Ilmenau, Fachgebiet Neuroinformatik und Kognitive Robotik.

[Prim, 1957] Prim, R. C. (1957). *Shortest connection networks and some generalizations*. Bell System Technology Journal, 36:1389–1401.

[Principe et al., 2000] Principe, J., D. Xu und J. Fisher (2000). *Unsupervised Adaptive Filtering*, Kap. Information Theoretic Learning, S. 265–319. Wiley.

[Rajagopalan et al., 1997] Rajagopalan, B., U. Lall und D. Tarboton (1997). *Evaluation of kernel density estimation methods for daily precipitation resampling*. Stochastic Hydrology and Hydraulics, 11:523–547.

[Rasmussen und Williams, 2005] Rasmussen, Carl E. und C. K. I. Williams (2005). *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press.

[Rasmussen und Kuss, 2004] Rasmussen, Carl Edward und M. Kuss (2004). *Gaussian Processes in Reinforcement Learning*. In: *Advances in Neural Information Processing Systems 16*, S. 751–759. MIT Press.

[Reinhardt, 2007] Reinhardt, M. (2007). *Stellgrößenbewertung und Komposition von Makrooperationen für die intelligente Feuerungsführung*. Diplomarbeit, Technische Universität Ilmenau, Fachgebiet Neuroinformatik und Kognitive Robotik und Powitec GmbH.

[Renyi, 1961] Renyi, Alfred (1961). *On measures of information and entropy*. In: *Proceedings of the 4th Berkeley Symposium on Mathematics, Statistics and Probability 1960*, S. 547–561.

[Reunanen, 2003] Reunanen, J. (2003). *Overfitting in Making Comparisons Between Variable Selection Methods*. Journal of Machine Learning Research, 3:1371–1382.

[Reunanen, 2006] Reunanen, Juha (2006). *Feature Extraction: Foundations and Applications*, Bd. 207 d. Reihe *Studies in fuzziness and soft computing*, Kap. Search Strategies, S. 119–136. Springer Verlag.

[Riedmiller, 2005] Riedmiller, Martin (2005). *Neural Fitted Q Iteration - First Experiences with a Data Efficient Neural Reinforcement Learning Method*. In: Gama, João, R. Camacho, P. Brazdil, A. Jorge und L. Torgo, Hrsg.: *Machine Learning: ECML 2005*, Bd. 3720 d. Reihe *Lecture Notes in Computer Science*, S. 317–328. Springer Berlin / Heidelberg.

[Riedmiller und Braun, 1993] Riedmiller, Martin und H. Braun (1993). *A Direct Adaptive Method for Faster Backpropagation Learning: The RPROP Algorithm*. In: *IEEE International Conference on Neural Networks*, S. 586–591.

[Riedmiller et al., 2009] Riedmiller, Martin, T. Gabel, R. Hafner und S. Lange (2009). *Reinforcement learning for robot soccer*. Autonomous Robots, 27:55–73.

[Riedmiller et al., 2007] Riedmiller, Martin, M. Montemerlo und H. Dahlkamp (2007). *Learning to Drive a Real Car in 20 Minutes*. Frontiers in the Convergence of Bioscience and Information Technologies, 0:645–650.

[Rosner et al., 2008] Rosner, Claus, H. Roepell, F. Wintrich, V. Stephan und E. Schaffernicht (2008). *Wirkungsgradverbesserung an steinkohlebefeuerten Dampferzeugern mittels lernfähiger, videogestützter Luftverteilungsoptimierung*. VGB Powertech, (12):94–99.

[Ross et al., 2008] Ross, S., B. Chaib-Draa und J. Pineau (2008). *Bayesian reinforcement learning in continuous POMDPs with application to robot navigation*. In: *IEEE International Conference on Robotics and Automation (ICRA'08)*, S. 2845–2851.

[Rumelhart et al., 1986] Rumelhart, D. E., G. E. Hinton und R. J. Williams (1986). *Learning internal representations by error propagation*, S. 318–362. MIT Press, Cambridge, MA, USA.

[Sanger, 1989] Sanger, Terence David (1989). *Optimal Unsupervised Learning in a Single-Layer Linear Feedforward Neural Network*. Neural Networks, 2:459–473.

[Schaffernicht et al., 2010] Schaffernicht, E., R. Kaltenhäuser, S. S. Verma und H.-M. Gross (2010). *Adaptive Feature Transformation for Image Data from Non-stationary Processes*. In: *Int. Conference on Artificial Neural Networks (ICANN10)*, S. 362–367.

[Schaffernicht und Gross, 2011] Schaffernicht, Erik und H.-M. Gross (2011). *Weighted Mutual Information for Feature Selection*. In: *ICANN (2)*, S. 181–188.

[Schaffernicht et al., 2009a] Schaffernicht, Erik, C. Moeller, K. Debes und H.-M. Gross (2009a). *Forward feature selection using Residual Mutual Information*. In: *17th European Symposium on Artificial Neural Networks (ESANN09)*, S. 583–588.

[Schaffernicht et al., 2009b] Schaffernicht, Erik, V. Stephan, K. Debes und H.-M. Gross (2009b). *Machine Learning Techniques for Selforganizing Combustion Control*. In: *32nd Annual Conference on Artificial Intelligence (KI)*, S. 395–402.

[Schaffernicht et al., 2007] Schaffernicht, Erik, V. Stephan und H.-M. Gross (2007). *An Efficient Search Strategy for Feature Selection Using Chow-Liu Trees*. In: *Int. Conference on Artificial Neural Networks ICANN07*, S. 190–199.

[Schaffernicht et al., 2009c] Schaffernicht, Erik, V. Stephan und H.-M. Gross (2009c). *Adaptive Feature Transformation for Image Data from Non-stationary Processes*. In: *Int. Conference on Artificial Neural Networks (ICANN09)*, S. 735–744.

[Scheidig et al., 2006] Scheidig, A., S. Mueller, C. Martin und H.-M. Gross (2006). *Generating Person's Movement Trajectories on a Mobile Robot*. In: *15th IEEE Int. Symposium on Robot and Human Interactive Communication (RO-MAN)*, RO-MAN 06, S. 747–752, Piscataway, NJ, USA. IEEE Press.

[Schölkopf et al., 1998] Schölkopf, Bernhard, A. Smola und K.-R. Müller (1998). *Nonlinear Component Analysis as a Kernel Eigenvalue Problem*. Neural Computation, 10(5):1299–1319.

[Schmid et al., 2006] Schmid, D., M.-S. Oh und D.-H. Kim (2006). *Reduction of UBC (Unburned Carbon-in-Ash) using an innovative combustion controller to increase efficiency*. In: *PowerGen Europe*.

[Scott, 1979] Scott, D. W. (1979). *On optimal and data-based histograms*. Biometrika, 66(3):605–610.

[Scott, 1992] Scott, D. W. (1992). *Multivariate density estimation: theory, practice, and visualization*. John Wiley & Sons: New York.

[Scott, 2009] Scott, D.W. (2009). *Sturges' rule*. Wiley Interdisciplinary Reviews: Computational Statistics, 1:303–306.

[Sebban und Nock, 2002] Sebban, Marc und R. Nock (2002). *A Hybrid Filter/Wrapper Approach of Feature Selection using Information Theory*. Pattern Recognition, 35(4):835 – 846.

[Shannon, 1948] Shannon, C. E. (1948). *A mathematical theory of communication*. The Bell System Technical Journal, 27:379–423.

[Si et al., 2004] Si, Jennie, A. G. Barto, W. B. Powell und D. Wunsch (2004). *Handbook of Learning and Approximate Dynamic Programming (IEEE Press Series on Computational Intelligence)*. Wiley-IEEE Press.

[Silverman, 1986] Silverman, B. W. (1986). *Density Estimation for Statistics and Data Analysis*. Chapman and Hall, London.

[Snelson und Ghahramani, 2006] Snelson, Edward und Z. Ghahramani (2006). *Sparse Gaussian Processes using Pseudo-inputs*. In: *NIPS*, S. 1257–1264. MIT press.

[Somol et al., 2006] Somol, Petr, J. Novovicová und P. Pudil (2006). *Flexible-Hybrid Sequential Floating Search in Statistical Feature Selection*, Bd. 4109 d. Reihe *Lecture Notes in Computer Science*, S. 632–639. Springer Berlin / Heidelberg.

[Souza et al., 2005] Souza, J., N. Japkowicz und S. Matwin (2005). *Feature Selection with a General Hybrid Algorithm*. In: *International Workshop on Feature Selection for Data Mining*.

[Stadler et al., 2011] Stadler, Konrad S., J. Poland und E. Gallestey (2011). *Model predictive control of a rotary cement kiln*. Control Engineering Practice, 19(1):1 – 9.

[Stanley und Miikkulainen, 2002] Stanley, Kenneth O. und R. Miikkulainen (2002). *Evolving Neural Networks through Augmenting Topologies*. Evolutionary Computation, 10(2):99–127.

[Steege et al., 2010] Steege, Frank-Florian, A. Hartmann, E. Schaffernicht und H.-M. Gross (2010). *Reinforcement learning based neural controllers for dynamic processes without exploration*. In: *Proceedings of the 20th international conference on Artificial neural networks: Part II*, ICANN'10, S. 222–227, Berlin, Heidelberg. Springer-Verlag.

[Stephan et al., 2001] Stephan, V., K. Debes, H.-M. Gross, F. Wintrich und H. Wintrich (2001). *A New Control Scheme for Combustion Processes using Reinforcement Learning based on Neural Networks*. International Journal on Computational Intelligence and Applications, 1(2):121–136.

[Stephan et al., 2004] Stephan, V., F. Wintrich, A. König und K. Debes (2004). *Application of Action Dependant Heuristic Dynamic Programming to Control an Industrial Waste Incineration Plant*. In: *3rd Workshop on Self-Organization of AdaptiVE Behavior*, SOAVE, S. 262–270. VDI-Verlag.

[Steuer et al., 2002] Steuer, R., J. Kurths, C. Daub, J. Weise und S. J. (2002). *The mutual information: Detecting and evaluating denpendencies between variables*. Bioinformatics, 18(2):231–240.

[Strehl und Littman, 2005] Strehl, Alexander L. und M. L. Littman (2005). *A theoretical analysis of Model-Based Interval Estimation*. In: *Proceedings of the 22nd international conference on Machine learning (ICML '05)*, S. 856–863.

[Sturges, 1926] Sturges, H. A. (1926). *The Choice of a Class Interval*. Journal of the American Statistical Association, 21(153):65–66.

[Sun et al., 2001] Sun, Ron, E. Merrill und T. Peterson (2001). *From implicit skills to explicit knowledge: a bottom-up model of skill learning*. Cognitive Science, 25(2):203–244.

[Sutton und Barto, 1998] Sutton, Richard S. und A. G. Barto (1998). *Reinforcement Learning: An Introduction*. MIT Press.

[Suzuki et al., 2008a] Suzuki, T., M. Sugiyama, J. Sese und T. Kanamori (2008a). *Approximating Mutual Informaton by Maximum Likelihood Density Ratio Estimation*. JMLR workshop and conference proceedings, 4:5–20.

[Suzuki et al., 2008b] Suzuki, T., M. Sugiyama, J. Sese und T. Kanamori (2008b). *A Least-squares Approach to Mutual Information Estimation with Application in Variable Selection*. Proceedings of the 3rd workshop on new challenges for feature selection in data mining and knowledge discovery (FSDM2008). Antwerp, Belgium.

[Taylor et al., 2006] Taylor, Matthew, S. Whiteson und P. Stone (2006). *Comparing Evolutionary and Temporal Difference Methods for Reinforcement*

Learning. In: *Proceedings of the Genetic and Evolutionary Computation Conference*, S. 1321–28.

[Terrell und Scott, 1985] Terrell, G.R. und D. Scott (1985). *Oversmoothed nonparametric density estimates*. Journal of the American Statistical Association, 80:209–214.

[Thrun, 1992] Thrun, Sebastian B. (1992). *Efficient Exploration In Reinforcement Learning*. Technischer Bericht, CMU, Pittsburgh, PA, USA.

[Tokic und Palm, 2011] Tokic, Michel und G. Palm (2011). *Value-difference based exploration: adaptive control between epsilon-greedy and softmax*. In: *Proceedings of the 34th Annual German conference on Advances in artificial intelligence*, KI'11, S. 335–346.

[Topalov und Kaynak, 2004] Topalov, Andon Venelinov und O. Kaynak (2004). *Neural network modeling and control of cement mills using a variable structure systems theory based on-line learning mechanism*. Journal of Process Control, 14(5):581 – 589.

[Torkkola, 2001] Torkkola, Kari (2001). *Nonlinear Feature Transforms Using Maximum Mutual Information*. In: *In Proc. of Int. Joint Conference on Neural Networks (IJCNN)*, S. 2756–2761.

[Torkkola, 2002] Torkkola, Kari (2002). *Learning Feature Transforms Is an Easier Problem Than Feature Selection*. In: *Inter. Conf. on Pattern Recognition ICPR(2)*, S. 104–107.

[Torkkola, 2003] Torkkola, Kari (2003). *Feature extraction by non parametric mutual information maximization*. J. Mach. Learn. Res., 3:1415–1438.

[Torkkola, 2006] Torkkola, Kari (2006). *Feature Extraction: Foundations and Applications*, Bd. 207 d. Reihe *Studies in fuzziness and soft computing*, Kap. Information-Theoretic Methods, S. 167–186. Springer Verlag.

[Trafton et al., 2005] Trafton, J. Gregory, N. L. Cassimatis, M. D. Bugajska, D. P. Brock, F. E. Mintz und A. C. Schultz (2005). *Enabling effective human-robot interaction using perspective-taking in robots*. IEEE Transactions on Systems, Man, and Cybernetics, 35:460–470.

[Troccaz, 2009] Troccaz, Jocelyne (2009). *Computer and Robot-Assisted Medical Intervention*, S. 1451–1466.

[Turlach, 1993] Turlach, Berwin A. (1993). *Bandwidth Selection in Kernel Density Estimation: A Review*. Technischer Bericht, CORE and Institut de Statistique, Voie du Roman Pays 34, B-1348 Louvain-la-Neuve, Belgium.

[Uschold und Grüninger, 1996] Uschold, Mike und M. Grüninger (1996). *Ontologies: principles, methods, and applications*. Knowledge Engineering Review, 11(2):93–155.

[Vafaie und Jong, 1992] Vafaie, Haleh und K. D. Jong (1992). *Genetic Algorithms as a Tool for Feature Selection in Machine Learning*. In: *in Machine Learning. In Proceedings of the 1992 IEEE Int. Conf. on Tools with AI*, S. 200–204. Society Press.

[Van Dijck und Van Hulle, 2006] Van Dijck, Gert und M. M. Van Hulle (2006). *Speeding Up the Wrapper Feature Subset Selection in Regression by Mutual Information Relevance and Redundancy Analysis*. In: *Int. Conference on Artificial Neural Networks ICANN*, S. 31–40.

[Van Hulle, 2005] Van Hulle, H. M. (2005). *Edgeworth Approximation of Multivariate Differential Entropy*. Neural Computation, 17(2):1903–1910.

[Vera et al., 2010] Vera, Pablo A., P. A. Estévez und J. C. Príncipe (2010). *Linear Projection Method Based on Information Theoretic Learning*. In: *ICANN (3)*, S. 178–187.

[Vollmer, 2009] Vollmer, Christian (2009). *Reinforcement Learning in kontinuierlichen Aktionsräumen mit Diffusionsbäumen unter Berücksichtigung des Exploration-Exploitation-Dilemmas*. Diplomarbeit, Technische Universität Ilmenau, Fachgebiet Neuroinformatik und Kognitive Robotik.

[Vollmer et al., 2010] Vollmer, Christian, E. Schaffernicht und H.-M. Gross (2010). *Exploring Continuous Action Spaces with Diffusion Trees for Reinforcement Learning*. In: *ICANN (2)*, S. 190–199.

[Wardana, 2004] Wardana, A.N.I. (2004). *PID-fuzzy controller for grate cooler in cement plant*. In: *Control Conference, 2004. 5th Asian (3)*, S. 1563 – 1567.

[Whiteson et al., 2009] Whiteson, Shimon, M. E. Taylor und P. Stone (2009). *Critical Factors in the Empirical Performance of Temporal Difference and Evolutionary Methods for Reinforcement Learning*. Journal of Autonomous Agents and Multi-Agent Systems, 21(1):1–27.

[Wiering und Schmidhuber, 1998] Wiering, Marco und J. Schmidhuber (1998). *Efficient Model-Based Exploration*. In: *Proceedings of the Sixth International Conference on Simulation of Adaptive Behavior: From Animals to Animats 6*, S. 223–228. MIT Press/Bradford Books.

[Wirtschaftsministerium, 2010] Wirtschaftsministerium (2010). *Energie in Deutschland - Trends und Hintergründe zur Energieversorgung*. Technischer Be-

richt, Referat für Öffentlichkeitsarbeit, Bundesministeriumg für Wirtschaft und Technologie, Berlin, Germany.

[Wolpert, 1996] Wolpert, David H. (1996). *The Lack of A Priori Distinctions Between Learning Algorithms*. Neural Computation, 8(7):1341–1390.

[Wolpert und Macready, 1997] Wolpert, David H. und W. G. Macready (1997). *No free lunch theorems for optimization*. IEEE Transactions on Evolutionary Computation, 1(1):67–82.

[Xing et al., 2001] Xing, Eric P., M. I. Jordan und R. M. Karp (2001). *Feature selection for high-dimensional genomic microarray data*. In: *ICML*, S. 601–608.

[Yang und Honavar, 1998] Yang, Jihoon und V. Honavar (1998). *Feature Subset Selection Using a Genetic Algorithm*. IEEE Intelligent Systems, 13:44–49.

[Zell, 1994] Zell, Andreas (1994). *Simulation neuronaler Netze*. R. Oldenbourg Verlag, München.

[Zhu et al., 2007] Zhu, Zexuan, Y.-S. Ong und M. Dash (2007). *Markov blanket-embedded genetic algorithm for gene selection*. Pattern Recognition, 40(11):3236–3248.

[Zipser et al., 2006] Zipser, S., A. Gommlich, J. Matthes und H. Keller (2006). *Combustion plant monitoring and control using infrared and video cameras*. In: *Power Plants and Power Systems Control*, International Federation of Automatic Control IFAC.