Über lernende optische Inspektion am Beispiel der Schüttgutsortierung

Zur Erlangung des akademischen Grades
Doktor der Ingenieurwissenschaften
der Fakultät für Informatik
Karlsruher Institut für Technologie (KIT)

genehmigte
Dissertation
von

Dipl.-Inform. Matthias Richter

Tag der mündlichen Prüfung:	19. Juli 2018
Referent:	Prof. Dr.-Ing. Jürgen Beyerer
Korreferent:	Prof. Dr.-Ing. Michael Heizmann

To "programme a machine to carry out the operation A" means to put the appropriate instruction table into the machine so that it will do A.

— Alan Turing [Tur50]

Programming computers to learn from experience should eventually eliminate the need for much of this detailed programming effort.

— Arthur L. Samuel [Sam59]

Kurzfassung

Die automatische optische Inspektion spielt als zerstörungsfreie Analysemethode in modernen industriellen Fertigungsprozessen eine wichtige Rolle. Typische, kommerziell eingesetzte automatische Inspektionssysteme sind dabei speziell an die jeweilige Aufgabenstellung angepasst und sind sehr aufwendig in der Entwicklung und Inbetriebnahme. Außerdem kann mangelndes Systemwissen der Anwender die Inspektionsleistung im industriellen Einsatz verschlechtern. Maschinelle Lernverfahren bieten eine Alternative: Die Anwender stellen lediglich eine Stichprobe bereit und das System konfiguriert sich von selbst. Ebenso können diese Verfahren versteckte Zusammenhänge in den Daten aufdecken und so den Entwurf von Inspektionssystemen unterstützen.

Diese Arbeit beschäftigt sich mit geeigneten lernenden Verfahren für die optische Inspektion. Die als Beispiel dienende Schüttgutsortierung setzt dabei die Rahmenbedingungen: Die Aufnahmebedingungen sind kontrolliert und die Objekterscheinung einfach. Gleichzeitig zeigen die Objekte mitunter nur wenige diskriminative Merkmale. Die Lernstichproben sind klein, unbalanciert und oft unvollständig in Bezug auf die möglichen Defektklassen. Zusätzlich ist die verfügbare Rechenzeit stark begrenzt. Unter Berücksichtigung dieser Besonderheiten werden in der vorliegenden Arbeit lernende Methoden für die Mustererkennungs-Schritte Bilderfassung, Merkmalsextraktion und Klassifikation entwickelt.

Die Auslegung der Bilderfassung wird durch die automatische Selektion optischer Filter zur Hervorhebung diskriminativer Merkmale unterstützt.

Anders als vergleichbare Methoden erlaubt die hier beschriebenen Methode die Selektion optische Filter mit beliebig komplizierten Transmissionskurven. Da relevante Merkmale die Grundvoraussetzung für eine erfolgreiche Klassifikation sind, nimmt die Merkmalsextraktion einen großen Teil der Arbeit ein. Solche Merkmale können beispielsweise aus einer Menge an Standardmerkmalen identifiziert werden. In der Schüttgutsortierung ist dabei neben der Relevanz aber auch der Rechenaufwand der Merkmalsextraktion von Bedeutung. In dieser Arbeit wird daher ein Merkmalsselektionsverfahren beschrieben, welches diesen Aufwand mit einbezieht. Daneben werden auch Verfahren untersucht, mit denen sich Merkmale mit Hilfe einer Lernstichprobe an ein gegebenes Sortierproblem anpassen lassen. Im Rahmen dieser Arbeit werden dazu zwei Methoden zum Lernen von Formmerkmalen bzw. von Farb- und Texturmerkmalen beschrieben. Mit beiden Verfahren werden einfache, schnell berechenbare, aber wenig diskriminative Merkmale zu hochdiskriminativen Deskriptoren kombiniert. Das Verfahren zum Lernen der Farb- und Texturdeskriptoren erlaubt außerdem die Detektion und Rückweisung unbekannter Objekte. Diese Rückweisungsoption wird im Sinne statistischer Tests für Anwender leicht verständlich parametriert.
Die Detektion unbekannter Objekte ist auch das Ziel der Einklassenklassifikation. Hierfür wird in dieser Arbeit ein Verfahren beschrieben, das den Klassifikator anhand einer Lernstichprobe mit lediglich Beispielen der Positivklasse festlegt. Die Struktur dieses Klassifikators wird außerdem ausgenutzt, um sicher unbekannte Objekte um Größenordnungen schneller zurückzuweisen als dies mit alternativen Verfahren möglich ist.
Alle vorgestellten Verfahren werden anhand von synthetischen Datensätzen und Datensätzen aus der Lebensmittelinspektion, Mineralsortierung und Inspektion technischer Gegenstände quantitativ evaluiert. In einer Gegenüberstellung mit vergleichbaren Methoden aus der Literatur werden die Stärken und Einschränkungen der Methoden herausgestellt. Hierbei zeigten sich alle vorgestellten Verfahren gut für die Schüttgutsortierung geeignet.

Die vorgestellten Verfahren ergänzen sich außerdem gegenseitig. Sie können genutzt werden, um ein komplettes Sortiersystem auszulegen oder um einzeln als Komponenten in einem bestehenden System eingesetzt zu werden. Die Methoden sind dabei nicht auf einen bestimmten Anwendungsfall zugeschnitten, sondern für eine großen Palette an Produkten einsetzbar. Somit liefert diese Arbeit einen Beitrag zur Anwendung maschineller Lernverfahren in optischen Inspektionssystemen.

Danksagung

Die vorliegende Dissertation entstand im Rahmen meiner Tätigkeit als wissenschaftlicher Mitarbeiter am Lehrstuhl für Interaktive Echtzeitsysteme des Karlsruher Instituts für Technologie und in enger Kooperation mit der Abteilung Sichtprüfsysteme des Fraunhofer-Instituts für Optronik, Systemtechnik und Bildauswertung. An dieser Stelle möchte ich denjenigen Personen danken, die mich auf meiner Reise begleitet und unterstützt haben:

Meinem Doktorvater Herrn Professor Dr.-Ing. Jürgen Beyerer für die Förderung und die Betreuung meiner Arbeit, die fachlichen Diskussionen und das Schaffen kreativer Freiräume.

Professor Dr.-Ing. Michael Heizmann für sein Interesse an meiner Arbeit und die Übernahme des Korreferats.

Den alten und neuen Kollegeninnen und Kollegen des Lehrstuhls und der Abteilung Sichtprüfsysteme für die anregenden Gespräche, das kritische Hinterfragen von und das Mitentwicklen der Ideen dieser Arbeit.

Danke an Johannes Meyer, Jonathan Wehrle und Lars Sommer für das Lesen und Korrigieren dieser Arbeit. Ich habe es euch nicht leicht gemacht.

Meinen Studenten für die Konkretisierung und Umsetzung von vielen meiner halbgaren Ideen.

Ein besonderer Dank gilt auch meiner Frau Alexandra für Unterstützung, Geduld und Verständnis, für die konstruktive Kritik und natürlich für die zahlreichen Korrekturen.

Inhaltsverzeichnis

Symbolverzeichnis

Allgemeine Bezeichner

$a, b, c, A, B, C, \ldots$	Skalar, Abbildung auf Skalar (kursiv)
$\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, \ldots$	Vektor, Abbildung auf Vektor (klein, fett, kursiv)
$\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \ldots$	Matrix (groß, fett, kursiv)
$\mathcal{A}, \mathcal{B}, \mathcal{C}, \ldots$	Menge (groß, kalligraphisch)
$[\boldsymbol{a}]_i$	i-ter Eintrag des Vektors $\boldsymbol{a}$
$[\boldsymbol{a}]_{\mathcal{I}}$	Vektor der von $i \in \mathcal{I} \subset \mathbb{N}$ benannten Elemente von $\boldsymbol{a}$
$\overline{\mathcal{A}}$	Komplement der Menge $\mathcal{A}$

Stochastik

$\overline{a}, \overline{b}, \overline{c}, \ldots$	Stichprobenmittel
$\mathfrak{B}(N,p)$	Binomialverteilung mit Anzahl an Experimenten N und Eintrittswahrscheinlichkeit p
$\mathfrak{Dir}(\boldsymbol{\alpha})$	Dirichlet-Verteilung mit Konzentrationsparametern $\boldsymbol{\alpha}$
$\mathbb{E}\{\cdot\}, \mathbb{E}_X\{\cdot\}$	Erwartungswert, Erwartungswert in Bezug auf die Zufallsvariable X
$\mathfrak{L}(\boldsymbol{\theta} \mid \mathcal{D})$	Likelihoodfunktion von $\boldsymbol{\theta}$ in Bezug zu $\mathcal{D}$
$\ell(\boldsymbol{\theta} \mid \mathcal{D})$	log-Likelihoodfunktion von $\boldsymbol{\theta}$ in Bezug zu $\mathcal{D}$

$\mathrm{Logit}\,(\omega \mid \boldsymbol{x})$	Logarithmus des Verhältnisses von $P(\omega_1 \mid \boldsymbol{x})$ und $P(\omega_2 \mid \boldsymbol{x})$
$\mathcal{N}(\mu, \sigma), \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$	Normalverteilung (Gauß-Verteilung) mit Mittelwert μ bzw. $\boldsymbol{\mu}$ und Varianz σ^2 bzw. Kovarianzmatrix $\boldsymbol{\Sigma}$
$P(\cdot), p(\cdot)$	Wahrscheinlichkeitsfunktion, Wahrscheinlichkeitsdichtefunktion
$P(\cdot, \cdot), p(\cdot, \cdot)$	Verbundwahrscheinlichkeit, Verbunddichte
$P(\cdot \mid \cdot)\,, p(\cdot \mid \cdot)$	Bedingte Wahrscheinlichkeitsfunktion, bedingte Wahrscheinlichkeitsdichtefunktion
$\mathfrak{U}(a; b)$	Gleichverteilung im Bereich $[a, b]$
$\mathrm{Var}\{\cdot\}\,, \mathrm{Var}_X\,\{\cdot\}$	Varianz, Varianz in Bezug auf die Zufallsvariable X
$\mathfrak{W}_p(\boldsymbol{V}, n)$	Wishart-Verteilung über positiv definite $p \times p$ Matrizen mit $n > p-1$ Freiheitsgraden und Skalierungmatrix $\boldsymbol{V}$

Lineare Algebra und Vektoranalysis

$\|\cdot\|, \|\cdot\|_p$	Norm, p-Norm
$\mathbf{0}$	Nullvektor, d. h. $[\mathbf{0}]_q = 0$ für alle q
$\mathbf{1}$	Einsvektor, d. h. $[\mathbf{1}]_q = 1$ für alle q
$\boldsymbol{a}^\top, \boldsymbol{A}^\top, \ldots$	Transponierte eines Vektors bzw. einer Matrix
$\boldsymbol{e}_q$	q-ter Standardbasisvektor, d. h. $[\boldsymbol{e}_q]_r = \mathbb{1}[q = r]$
$\mathrm{diag}(\boldsymbol{x})$	Diagonalmatrix mit Diagonale $\boldsymbol{x}$
∇_x	Partielle Ableitung nach x, d. h. $\nabla_x = \frac{\partial}{\partial x}$
$\nabla_{\boldsymbol{x}}$	Gradient nach $\boldsymbol{x}$, d. h. $\nabla_{\boldsymbol{x}} = \left(\frac{\partial}{\partial [\boldsymbol{x}]_1}, \ldots, \frac{\partial}{\partial [\boldsymbol{x}]_D}\right)^\top$
$\boldsymbol{I}$	Identitätsmatrix, d. h. $\boldsymbol{I} = \mathrm{diag}(\mathbf{1})$
$\mathrm{spur}(\boldsymbol{A})$	Spur der Matrix $\boldsymbol{A}$

Spezielle Bezeichner

$\mathbb{1}[p]$	Indikatorfunktion über das Prädikat p, d. h. $\mathbb{1}[p] = 1$, wenn p wahr und $\mathbb{1}[p] = 0$ sonst
C	Anzahl der Klassen
D	Dimension des Merkmalsraums
$d(\cdot, \cdot)$	Distanzmaß, Metrik
$\mathcal{D}$	Lernstichprobe
$\mathcal{H}$	Menge der Hypothesenfunktionen
$h(\boldsymbol{x}), H(\boldsymbol{x})$	Hypothesenfunktion
$i(\mathcal{D})$	Unreinheitsmaß
$\mathfrak{J}(\boldsymbol{\theta})$	Zielfunktion
$l(\omega_i, \omega_k)$	Verlustfunktion
$\mathbb{M}$	Merkmalsraum
N	Umfang der Lernstichprobe
$\mathcal{O}(f)$	Landau-Symbol, Komplexitätsklasse
$o \in \Omega$	Objekt
Ω	Welt bzw. Klassifikationsdomäne
$\omega, \omega_c \subseteq \Omega$	Klasse
$\hat{\omega}(o), \hat{\omega}(\boldsymbol{x})$	Klassifikator
$\mathbb{P}$	Musterraum
$\boldsymbol{p} \in \mathbb{P}$	Muster
Θ	Parameterraum
$\boldsymbol{\theta} \in \Theta$	Parametervektor
Ψ	Menge der Merkmalsextraktionsverfahren

$\psi \in \Psi$	Merkmalsextraktionsverfahren
$\mathrm{sign}(x)$	Vorzeichen von x
$\mathrm{supp}\{f\}$	Support der Funktion f, d. h. $\mathrm{supp}\{f\} = \{x \mid f(x) \neq 0\}$
$\mathcal{R}_c \subseteq \mathbb{M}$	(Entscheidungs-)Region
$\mathfrak{R}(\hat{\omega})$	Risiko eines Klassifikators
$\mathcal{S} \subseteq \Psi$	(Merkmals-)Selektion
$s(\cdot, \cdot)$	Ähnlichkeitsmaß
$\mathcal{T}$	Teststichprobe
$U(\mathcal{S}; \mathcal{D})$	Gütemaß einer Merkmalsselektion in Bezug auf $\mathcal{D}$
$\mathcal{X} = \mathcal{D} \cup \mathcal{T}$	Gesamtstichprobe
$\boldsymbol{x} \in \mathbb{M}$	Merkmal, Merkmalsvektor
$\boldsymbol{x}_n, y_n, \omega_n$	Stichprobenelement
y	Klassenlabel

Abkürzungsverzeichnis

BOW	Bag of Visual Words
CCA	Connected Component Analysis
CFS	Correlation-based Feature Selection
CLM	Conditional Likelihood Maximization
CNN	Convolutional Neural Network
DCT	diskrete Kosinustransformation
EM	Expectation Maximization
ERM	Empirical Risk Minimization
FPR	False Positive Rate, auch: Fallout
FV-Kodierung	Fisher-Vektorkodierung
GA	genetischer Algorithmus
GMM	Gauß-Mischmodell
GMB	Gauß-Mixturbaum
HOG	Histogram of Oriented Gradients
i. i. d.	unabhängig und identisch verteilt (engl.: independent and identically distributed)
JMI	Joint Mutual Information
kNN	k nächste Nachbarn
LBP	Local Binary Patterns

LDA	Lineare Diskriminanzanalyse, Latent Dirichlet Allocation
MAP	maximum a-posteriori
MCC	Matthews' Correlation Coefficient
MDA	Multiple Diskriminanzanalyse
MRMR	Maximum-Relevance Minimum-Redundancy
NIR	Nahinfrarot
NSGA-II	Nondominated Sorting Genetic Algorithm II
ocSVM	One Class SVM
PCA	Hauptkomponentenanalyse
PLSR	Partial Least Squares Regression
PSO	Particle Swarm Optimization
RBF	radiale Basisfunktion
RF	Random Forest
ROC	Receiver Operating Characteristics
SIFT	Scale-Invariant Feature Transform
SRM	Structural Risk Minimization
SVM	Support Vektor Maschine
SVDD	Support Vector Data Description
SWIR	Kurzwelliges Infrarot (short wave infrared)
TPR	True Positive Rate, auch: Recall, Sensitivity

1 Einleitung

Optische Inspektion, auch Sichtprüfung genannt, bezeichnet die optische Kontrolle eines Produkts auf Fehler bzw. auf die Einhaltung von Qualitätsstandards. Als zerstörungsfreies Prüfverfahren findet Sichtprüfung vielseitige Anwendung, z. B. bei der Fertigung von Automobilen, bei der Inspektion von Lebensmitteln und beim Recycling von Abfällen. Moderne industrielle Fertigungsprozesse beinhalten fast immer eine Sichtprüfung zur Eingangskontrolle der benötigten Ausgangsmaterialien, zur Endkontrolle des fertigen Produkts oder zur Überwachung von Zwischenschritten.
Sichtprüfung wird dabei häufig von Menschen durchgeführt, da Menschen komplexe Situationen sehr schnell erfassen und passende Entscheidungen ableiten können. Doch auch unter Verwendung von Hilfsmitteln wie optischer Vergrößerung und speziellen Beleuchtungen ist diese Art der Prüfung fehleranfällig. Grund hierfür sind Konzentrationsschwankungen, die etwa durch Ermüdung, Langeweile oder Ablenkung verursacht werden. Schoonahd et al. [SGM73] fassen es so zusammen:

> In general, human visual inspection is characterized by three facts. First, inspectors look for many things at once. Second, they must do this very fast. Third, they are not very accurate.

Zudem ist die Leistungsfähigkeit und Einsatzmöglichkeit menschlicher Inspekteure begrenzt: Eine lückenlose Qualitätsprüfung von Materialien geringen Werts – z. B. von Getreide oder Granulaten – ist nicht umsetzbar, da die ökonomischen Gegebenheiten einen hohen Durchsatz erfordern, der von Menschen nicht erreichbar ist. Zudem können Menschen nicht in feindlichen

Umgebungen, etwa bei extremen Temperaturen, in einer Schutzatmosphäre oder in einem radioaktiven Umfeld arbeiten.
Es liegt also Nahe, die optische Inspektion zu automatisieren. Automatische optische Inspektion, bzw. automatische Sichtprüfung, ersetzt die menschlichen Inspekteure durch computergestützte Systeme. Diese Systeme werten Kamerabilder oder Daten anderer bildgebender Verfahren mit Hilfe von Bildverarbeitungsverfahren aus, um die Qualität der Produkte zu bewerten, Defekte zu lokalisieren und um Objekte zu sortieren. Computer arbeiten schneller als Menschen und machen, korrekte Programmierung vorausgesetzt, keine Fehler.
Es stellt sich aber die Frage, wie die Computer programmiert werden müssen, um das Ziel einer konsistenten, vollständigen und schnellen automatischen optischen Inspektion zu erreichen. Diese Frage entspricht einem klassischen Entscheidungsfindungsproblem: Zeigt ein gegebenes Werkstück einen Defekt? Ist die Qualität des Materials ausreichend? Muss ein Teil aussortiert werden?

1.1 Algorithmische Entscheidungsfindung

Ein erster Durchbruch in der computergestützten Entscheidungsfindung gelang in den 1970er und 1980er Jahren mit den sogenannten Expertensystemen. Expertensysteme bestehen aus einer Wissensbasis und einer Inferenzmaschine zum Ableiten von Entscheidungen. Anstelle von üblichem Programmcode ist die Wissensbasis dabei in Form von Fallbeispielen oder `WENN-DANN`-Regeln repräsentiert. Die Inferenzmaschine gleicht die gegebenen Fakten mit dieser Wissensbasis ab, um mittels logischen Schlussfolgerungen Entscheidungen zu treffen. Durch die Trennung von Wissensbasis und Inferenzmaschine kann das Wissen unabhängig vom eigentlichen Programm erweitert werden, um die Leistung des Expertensystems stetig zu verbessern.
Prominente Beispiele solcher Expertensysteme sind MYCIN von Shortliffe [Sho74], PROSPECTOR von Hart et al. [HDE78] und R1/XCON von McDermott [McD82]. MYCIN ist ein Assistenzsystem, das Ärzte bei der

Diagnose ansteckender bakterieller Infektionen und der Auswahl passender Antibiotika unterstützen soll. Die Wissensbasis besteht aus über 600 Regeln, die mit Hilfe von Experten zusammengetragen wurden. MYCIN versteht einfache englische Sätze und kann getroffene Diagnosen begründen und die zugehörige Konfidenz angeben.
PROSPECTOR unterstützt Geologen bei der Exploration von Mineralvorkommen und der Bewertung der vorhandenen Ressourcen. Die Wissensbasis besteht aus `WENN-DANN`-Regeln, die zusätzlich auch die Eintrittswahrscheinlichkeit einer Hypothese quantifizieren. Die Inferenz erfolgt probabilistisch auf Grundlage eines Bayes'schen Netzes und kann somit auch mit unvollständigen oder unsicheren Eingaben erfolgen.
R1/XCON ist ein Expertensystem, das zur Konfiguration von VAX-11 Computersystemen verwendet wurde. Auf Grundlage eines Anforderungskatalogs wurde sowohl die Vollständigkeit der bestellten Bauteile geprüft, als auch ein Bauplan für den Aufbau des Systems erstellt. Die Wissensbasis besteht aus Regeln zur Kombination von Bauteilen sowie aus Spezifikationen dieser Bauteile. McDermott [McD82] geben den Umfang des Systems mit 292 generellen und 480 domänenspezifischen Regeln an. Wenige Jahre später war das Regelwerk auf über 10000 Regeln angewachsen [Bar+89].
Zwar wurden mit diesen und anderen Systemen große kommerzielle Erfolge erzielt, jedoch verdeutlicht insbesondere R1/XCON einen entscheidenden Nachteil von Expertensystemen: Komplexe Problemstellungen ziehen komplexe Regelwerke nach sich. Je größer das Regelwerk, desto schwieriger wird es, die Korrektheit desselben sicherzustellen und redundante oder widersprüchliche Regeln zu erkennen. Außerdem ist das Aufstellen einer Wissensbasis durch Befragung von Experten sehr zeitaufwendig. Hart et al. kumulierten beispielsweise 50 Stunden Interviews für ein einziges Modell [HDE78]. Die Aufbereitung dieser Interviews und die Überführung in Regeln der Wissensbasis ist ebenfalls sehr zeitaufwendig. Zudem sind Expertensysteme zur Lösung bestimmter Probleme, wie der Spracherkennung, der Bildkategorisierung oder das Finden optimaler Spielzüge, ungeeignet. Menschen

können hier zwar sehr gute Entscheidungen treffen und einzelne Schritte der Entscheidungen begründen, aber meist kein allgemeingültiges Vorgehen zur Entscheidungsfindung angeben.
Arthur Samuel verfolgte schon früh einen anderen Ansatz: Anstatt aus formalisiertem Expertenwissen zu schließen, sollte der Computer selbst aus Erfahrung lernen und so selbst zum Experten werden. Bereits 1955 präsentierte Samuel ein Programm, welches gelernt hatte, Checkers (eine Variante des Brettspiels Dame) zu spielen [Sam59]. Dieses Programm lernte, indem es immer wieder gegen sich selbst spielte und die eigenen Züge bewertete. Sogar mit der begrenzten Rechenleistung und Speicherressourcen der damaligen Zeit konnte dieses Programm bereits menschliche Spieler besiegen. Der verfolgte Ansatz ist heute als Reinforcement Learning (siehe Sutton und Barto [SB98]) bekannt und führte jüngst zu großer medialer Aufmerksamkeit, als ein Computer den bisherigen Go-Weltmeister in diesem hoch komplexen Strategiespiel besiegte [Sil+16; Sil+17].
Dieses Vorgehen – Verhalten zu lernen, statt es explizit zu kodieren – ist als maschinelles Lernen bekannt und findet heute Anwendung in nahezu allen Bereichen des täglichen Lebens: Filterung unerwünschter E-Mails, Spracherkennung, automatische Übersetzung, Analyse und Komposition von Musikstücken, Zusammenfassung und Erstellung journalistischer Texte, Analyse und Generierung von Bild und Videomaterial, etc.

1.2 Automatische Sichtprüfung

Trotz dieser breiten Anwendung und trotz der Erfolge maschinellen Lernens stützen sich kommerzielle Sichtprüfsysteme noch immer fast ausschließlich auf regelbasierten Architekturen. Damit gehen auch die genannten Probleme von Expertensystemen mit einher: Die Übersetzung von Expertenwissen in Regeln ist schwierig und zeitaufwendig, das Regelsystem für komplexe Sichtprüfaufgaben ist umfangreich und schwer überschaubar und einmal

eingerichtete Systeme sind unflexibel und lassen sich ohne Modifikation nicht auf neue oder veränderte Produkte übertragen.
Warum also nicht den von Samuel vorgeschlagenen Weg des maschinellen Lernens gehen und die Regeln aus Daten lernen? Solche Systeme müssten nicht mehr aufwendig von Bildverarbeitungsexperten eingerichtet werden, sondern könnten direkt von den Anwendern in Betrieb genommen werden. Die Anwender müssen lediglich eine Stichprobe der zu inspizierenden Materialien bereitstellen. Ändern sich die Anforderungen, etwa indem ein neuer Defekttyp hinzukommt oder sich das Produkt verändert, kann das System einfach neu eingelernt werden, ohne dass zusätzliche Analysen notwendig sind.
In dieser Arbeit werden Methoden untersucht, um diesem Ziel näher zu kommen. Exemplarisch dient hierzu die Schüttgutsortierung, also die visuelle Inspektion von stückigen bis körnigen Gemengen, mit dem Ziel der Sortierung in verschiedene Materialklassen. Beispiele für Schüttgutsortierung sind die schon oben erwähnte Getreidesortierung, bei der gesunde Körner von gebrochenen, infizierten oder sortenfremden Körnern und sonstigen Verunreinigungen getrennt werden sollen, die Inspektion von Bruch im Bergbau zur Entdeckung wertvoller Metalle und Minerale oder die Sortierung von Glasscherben zum Recycling.

1.3 Schüttgutsortiersysteme

Automatische Schüttgutsortierer sind komplexe mechatronische Systeme, in denen der mechanische Transport des Schüttguts, die Bildverarbeitung und die Ausschleusung defekter Objekte aufeinander abgestimmt sind.
Abbildung 1.1 zeigt einen typischen Aufbau eines solchen Systems: Ein Förderband beschleunigt das Schüttgut, um so ungewünschte laterale Bewegungen zu unterdrücken und um einen definierten Zustand für die folgende Inspektion und Ausschleusung sicherzustellen. Am Ende des Förderbands geht das Schüttgut in den freien Fall über, wo es von einer Kamera, meist eine Zeilenkamera, beobachtet wird. Sowohl die Beleuchtung als auch der

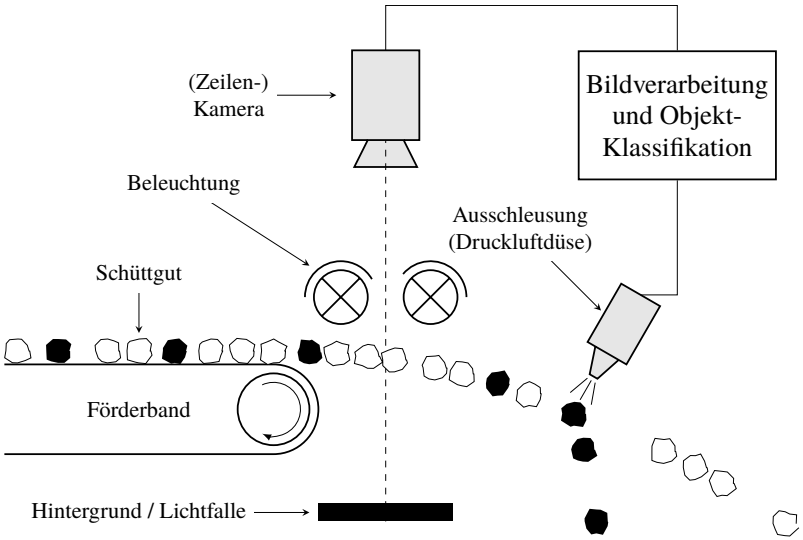


Abbildung 1.1: Schematischer Aufbau eines typischen Schüttgutsortiersystems.

Hintergrund sind so gewählt, dass die Objekte auf dem Bild möglichst gut identifizierbar sind. Ein Computer detektiert einzelne Objekte im Bild und ordnet diese der Gut- oder Schlechtfraktion zu. Diese Zuordnung wird schließlich mittels ansteuerbaren Druckluftdüsen physikalisch umgesetzt. Je nach Geschwindigkeit des Förderbands und Abstand zwischen Inspektion und Ausschleusung bleiben nur wenige Millisekunden, um das Material zu klassifizieren. Üblich sind dabei Bandgeschwindigkeiten von ca. $3\,\frac{\mathrm{m}}{\mathrm{s}}$ und geforderte Reaktionszeiten von ca. 20 ms. Abbildung 1.2 zeigt ein solches Schüttgutsortiersystem, das zu Demonstrations- und Forschungszwecken am Fraunhofer-IOSB aufgebaut ist.

In manchen Systemen wird das Material statt durch ein Förderband mit einer Rutsche transportiert oder die Sortierentscheidung durch mechanische Elemente statt durch Druckluft umgesetzt. Auch kann die Anordnung von Beleuchtung, Hintergrund und Kamera variieren oder es können mehrere

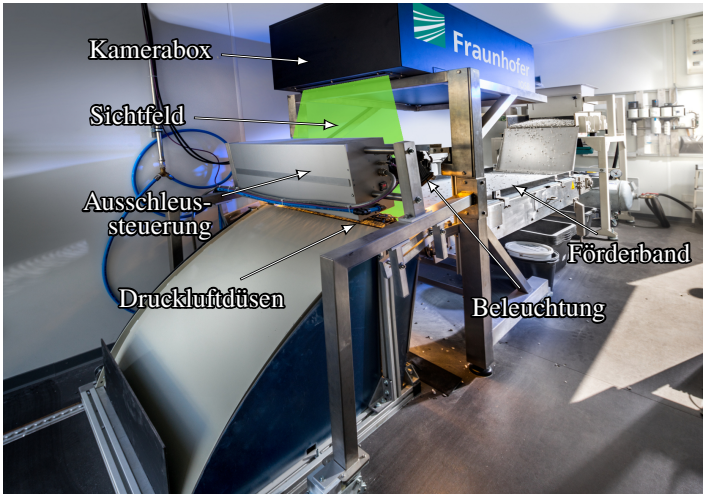


Abbildung 1.2: Beispiel eines Schüttgutsortiersystems am Fraunhofer-IOSB.

bzw. andere bildgebende Sensoren verwendet werden. Das Grundprinzip der Sortierung bleibt jedoch stets das gleiche. Obwohl auch die Optimierung des Gesamtsystems, etwa die Art, Anzahl und Positionierung der Beleuchtung und verwendeten Sensoren, ein spannendes wissenschaftliches Thema ist, soll es im Folgenden nur um die Auswertung der bereits gewonnen Bilder gehen (Abschnitt 3.1 bildet eine Ausnahme: Hier werden Teile der Auswertung in die Bilderfassung verlagert).

Abbildung 1.3 zeigt die Bildverarbeitungskette eines typischen Schüttgutsortiersystems: Zunächst werden die Objekte im Kamerabild mittels Segmentierungsverfahren gefunden und vereinzelt. Für jedes Objekt werden Deskriptoren berechnet, die etwa die Farbe, Textur, oder Geometrie des Objekts beschreiben. Die Einordnung in Gut- bzw. Schlechtprodukt erfolgt in kommerziell eingesetzten Systemen wie oben beschrieben meist durch regelbasierte Klassifikation.

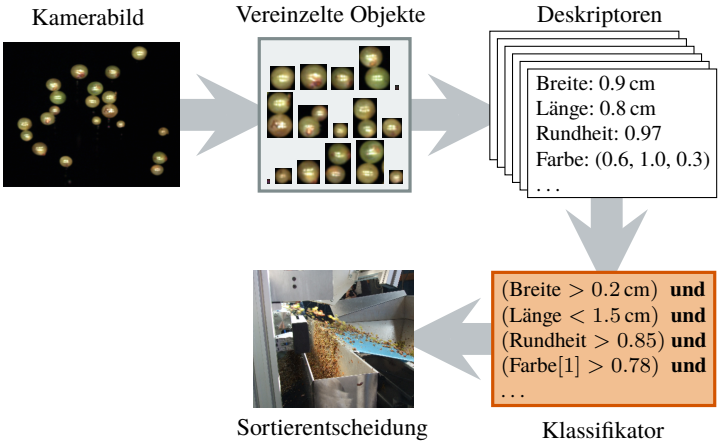


Abbildung 1.3: Bildverarbeitungskette eines typischen Schüttgutsortiersystems.

Abstrakt betrachtet handelt es sich hierbei um die klassische Verarbeitungskette der Mustererkennung: Sensierung ⇒ Detektion ⇒ Merkmalsextraktion ⇒ Klassifikation. Es liegt nahe, die Komponenten dieser Kette durch andere, auch lernende Verfahren aus dem Maschinensehen zu ersetzen.
Im Vergleich zu anderen Fragestellungen des Maschinensehens erscheint die Schüttgutsortierung sogar als ein verhältnismäßig einfaches Problem: Die Umgebung, d. h. die Beleuchtungssituation und der Hintergrund, ist kein Störfaktor, sondern eine Designgröße. Zudem haben die zu inspizierenden Objekte ein relativ einfaches Erscheinungsbild. Allerdings zeigen die Objekte oft wenig diskriminative Merkmale und das Erscheinungsbild variiert insbesondere bei natürlichen Objekten mitunter stark. Zudem steht, wie oben erwähnt, nur sehr wenig Zeit zur Bilderfassung und -auswertung zur Verfügung. Schließlich sind die verwendeten Datensätze oft sehr klein und beinhalten deutlich mehr Beispiele der Gut- als der Schlechtfraktion. Die Datensätze sind also unbalanciert und nicht notwendigerweise repräsentativ. Schlimmer: Die Schlechtfraktion ist in der Regel nicht vollständig bekannt,

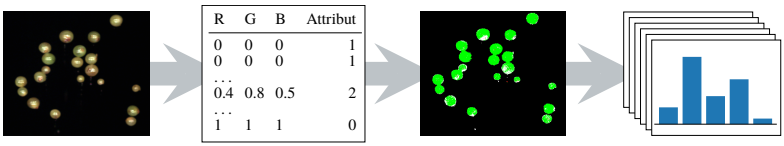


Abbildung 1.4: Merkmalsextraktion des Teach'N'Sort Systems: Jedem Pixel wird ein Attributwert zugeordnet; Objekte werden anhand der so erhaltenen Attribute beschrieben.

was bedeutet, dass bei der Sortierung Objekte auftauchen können, die während der Inbetriebnahme nicht beachtet wurden. Ein Sortiersystem sortiert immer unter der Annahme einer offenen Welt und ein lernendes System muss diese unbekannten Fälle ebenso klassifizieren können wie bekannte Klassen.
Es ist also nicht klar, ob lernende, datengetriebene Ansätze wirklich ein Verbesserungspotential gegenüber klassischen regelbasierten Systemen bergen.

1.4 Fallstudie: Teach'N'Sort

Um diese Frage zu klären, wurden die für die Klassifikation anhand von Farbe relevanten Teile des vom Fraunhofer-IOSB entwickelten Systems „Teach'N'Sort" zunächst mathematisch modelliert. In einem zweiten Schritt wurden die so identifizierten freien Parameter automatisch anhand einer Lernstichprobe festgelegt. Teach'N'Sort ist als klassifizierende Komponente in einer Reihe von kommerziell eingesetzten Systemen enthalten.
Die Merkmalsextraktion der Farbsortierung des Systems ist in Abbildung 1.4 zusammengefasst. Jedem Pixel des Kamerabilds wird mit Hilfe einer Attribut-Tabelle ein Attributwert zugewiesen. Diese Attribut-Tabelle ordnet jedem RGB-Farbwert ein Attributwert zu, wobei die Zuordnung wie unten beschrieben anhand der Farbverteilungen von Referenzaufnahmen gewonnen wird. Die Attribute kodieren somit semantisch bedeutsame Konzepte wie „Farbe reifer Beeren" oder „Farbe von Pflanzenteilen". Anhand des so gewonne-

nen Attributbilds werden die Objekte detektiert und die Objektdeskriptoren errechnet. Der Deskriptor eines gegebenen Objekts entspricht dabei einer Zählstatistik der im zugehörigen Bildausschnitt auftretenden Attributwerte. Dieses System lässt sich folgendermaßen formalisieren: Die Farbe eines Bildpixels wird durch den Vektor $\boldsymbol{c} = (r, g, b)^\top \in [0,1]^3$ beschrieben, wobei r, g und b die Farbwerte des Rot-, Grün- und Blaukanals bezeichnen. Die Menge der J_i Vordergrundpixel das i-ten Objekts der Lernstichprobe sei $\mathcal{O}_i = \{\boldsymbol{c}_{i,j} \,|\, j = 1, \ldots, J_i\}$ mit $i = 1, \ldots, N$. Weiterhin sei

$$a(\boldsymbol{c}) : [0,1]^3 \to \{0, 1, 2, \ldots, D\} \tag{1.1}$$

die Attributabbildung, die jedem Farbtupel genau ein Attribut zuweist. Das Attribut 0 bezeichnet hierbei „unbekannte" Farben, also Farben, die nicht in den Referenzaufnahmen auftauchen. Für ein gegebenes Objekt $\mathcal{O}$ ist der zugehörige Objektdeskriptor $\boldsymbol{x} = ([\boldsymbol{x}]_0, [\boldsymbol{x}]_1, \ldots, [\boldsymbol{x}]_D)^\top \in [0,1]^{D+1}$ durch

$$[\boldsymbol{x}]_d = \frac{1}{|\mathcal{O}|} \sum_{\boldsymbol{c} \in \mathcal{O}} \mathbb{1}[a(\boldsymbol{c}) = d], \quad d = 0, 1, 2, \ldots, D \tag{1.2}$$

gegeben. Das Merkmal $[\boldsymbol{x}]_d$ bezeichnet also den Anteil der Vordergrundpixel, die das Attribut d tragen.

Die wichtigste Komponente in diesem Ansatz ist die Attributabbildung $a(\boldsymbol{c})$. Diese wird in mehreren Stufen aus Aufnahmen des zu sortierenden Materials gewonnen. Zunächst werden K empirische Farbverteilungen $\hat{p}_f(\boldsymbol{c} \,|\, k)$, $k = 1, \ldots, K$ der Vordergrundpixel aus den K Referenzaufnahmen ermittelt. Bei Teach'N'Sort kommen dabei dreidimensionale Histogramme, also Zählstatistiken, zum Einsatz. Im zweiten Schritt werden durch

$$\text{cut}\left(\hat{p}_f(\boldsymbol{c} \,|\, k), \beta_k\right) := \begin{cases} \frac{1}{Z_k} \hat{p}_f(\boldsymbol{c} \,|\, k) & \text{wenn } \hat{p}_f(\boldsymbol{c} \,|\, k) \geq \beta_k \\ 0 & \text{sonst} \end{cases} \tag{1.3}$$

selten vorkommende Farben verworfen. Hier ist Z_k eine Normalisierungskonstante, die sicherstellt, dass cut$(\hat{p}, \beta)$ eine gültige Wahrscheinlichkeitsdichte darstellt. Dieser Schritt dient der Bereinigung der Farbhistogramme, da die verwendeten Referenzaufnahmen auch Schmutzpartikel oder anderes, nicht erwünschtes Material zeigen können. Die so bearbeiteten Farbverteilungen werden zu $D < K$ „Farbklassen" zusammengefasst:

$$\hat{p}_c(\boldsymbol{c} \,|\, d) = \sum_{k=1}^{K} \alpha_{kd} \,\mathrm{cut}\left(\hat{p}_f(\boldsymbol{c} \,|\, k)\,, \beta_k\right). \tag{1.4}$$

Hierbei ist durch $\alpha_{kd} \geq 0$ mit $\sum_k \alpha_{kd} = 1$ für $d = 1, \ldots, D$ sichergestellt, dass die $\hat{p}_c(\boldsymbol{c} \,|\, d)$ wohldefinierte Wahrscheinlichkeitsdichtefunktionen sind. Da die so gewonnenen Farbklassen auf einer nicht notwendigerweise repräsentativen Stichprobe des Materials basieren, das System aber dennoch robust gegenüber kleinen Veränderungen der Farbe sein soll, werden die Farbklassen im vierten Schritt aufgeweitet. Diese Operation wird durch eine Faltung mit einem parametrischen Rausch-Kernel $e(\boldsymbol{c}|\boldsymbol{\theta}_d)$ realisiert,

$$\hat{p}_\nu(\boldsymbol{c} \,|\, d) := \hat{p}(\boldsymbol{c} \,|\, d) * e(\boldsymbol{c} \,|\, \boldsymbol{\theta}_d) = \int \hat{p}(\boldsymbol{\gamma} \,|\, d)\, e(\boldsymbol{c} - \boldsymbol{\gamma} \,|\, \boldsymbol{\theta}_d)\, \mathrm{d}\boldsymbol{\gamma}, \tag{1.5}$$

was wiederum der Annahme von unkorreliertem, additivem Rauschen im Farbraum entspricht. Um die Anzahl der zu schätzenden Parameter zu reduzieren, wurde hier isotropes, mittelwertfreies, normalverteiltes Rauschen angenommen, d. h.

$$e(\boldsymbol{c} \,|\, \sigma_d) := \frac{1}{\sqrt{(2\pi)^3 \sigma_d^3}} \exp\left(-\frac{\boldsymbol{c}^\top \boldsymbol{c}}{2\sigma_d^3}\right). \tag{1.6}$$

Die vollständige Verarbeitungskette der Farbklassengenerierung ist in Abbildung 1.5 exemplarisch dargestellt.

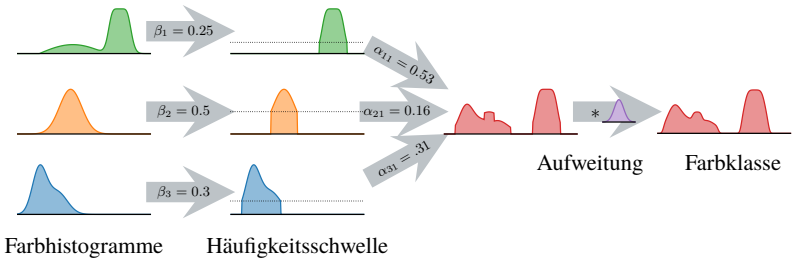


Abbildung 1.5: Generierung der Farbklassen im Teach'N'Sort-System.

Die Attributabbildung $a(\boldsymbol{c})$ wird schließlich durch maximum a-posteriori Klassifikation mit Rückweisungsoption (siehe Abschnitt 2.1) aus den $\hat{p}_\nu(\boldsymbol{c} \,|\, d)$ gewonnen, d. h. mit $\hat{p}_\nu(\boldsymbol{c}, d) = \hat{p}_\nu(\boldsymbol{c} \,|\, d)\, P(d)$:

$$a(\boldsymbol{c}) := \begin{cases} 0 & \text{wenn } \max_d \hat{p}_\nu(\boldsymbol{c}, d) < \epsilon \\ \arg\max_d \hat{p}_\nu(\boldsymbol{c}, d) & \text{sonst.} \end{cases} \tag{1.7}$$

Die $\hat{p}_\nu(\boldsymbol{c} \,|\, d)$ werden hier als klassenbedingte Merkmalsdichten der Farbklassen aufgefasst. Die a-priori Klassenwahrscheinlichkeiten $P(d) \propto \gamma_d \geq 0$ sind wie die anderen Parameter α_{kd} und β_k vom Benutzer einstellbar und steuern so das Verhalten des Systems.

In der Praxis wird die Abbildung $a(\boldsymbol{c})$ nicht zur Laufzeit ausgewertet, sondern für jeden Farbwert vorberechnet und als Lookup-Tabelle abgelegt. Dies ist möglich, da die Farbwerte $\boldsymbol{c}$ ohnehin durch die Hardware quantisiert werden. Gegebenenfalls kann der Farbraum noch weiter unterabgetastet werden, um Speicherplatz zu sparen.

Die Deskriptoren $\boldsymbol{x}$ werden mit Hilfe von regelbasierten Klassifikatoren der Form `WENN` $[\boldsymbol{x}]_d < \tau_d$ `DANN (Defekt|Gut)` klassifiziert. In hier nicht reproduzierten Experimenten zeigte sich allerdings, dass die Attributabbildung $a(\boldsymbol{c})$ einen deutlich größeren Einfluss auf die Klassifikationsgüte hat, als die eigentlichen Klassifikationsregeln. Dies kann dadurch erklärt

werden, dass die Attributabbildung den gesamten Farbraum auf eine kleine Menge nominaler Merkmale überführt. Die im Bild enthaltene Information wird dabei sehr stark reduziert, sodass der Objektklassifikator nur wenig nicht-diskriminative Information transportiert. Der Klassifikator verrichtet nur noch wenig zusätzliche Arbeit.

1.4.1 Datengetriebene Festlegung der Parameter

Die oben beschriebene Farbklassifizierung wird durch folgende Parameter beeinflusst:

- Gewichte α_{kd}, mit $\alpha_{kd} \geq 0$ und $\sum_k \alpha_{kd} = 1$ ($D \cdot (K-1)$ Parameter),
- Häufigkeitsschwellen β_k mit $0 < \beta_k \leq 1$ (K Parameter),
- Kernelbreiten σ_d mit $\sigma_d > 0$ (D Parameter) und
- Gewichte γ_d mit $\gamma_d \geq 0$ (D Parameter).

Es ergeben sich also $D\,(K-1) + K + D + D = D\,(K+1) + K$ freie Parameter, wobei K die Anzahl der zugrunde liegenden Farbverteilungen und D die Anzahl der semantisch bedeutsamen Farbklassen ist. In einem typischen Sortierproblem ist üblicherweise $K \approx 10$ und $D \approx 5$, womit in der Regel insgesamt etwa 65 Parameter einzustellen sind. Die manuelle Einstellung dieser Parameter ist ein sehr zeitaufwendiger, oft mehrtägiger Prozess. Dieser Prozess wird auch dadurch erschwert, dass bereits kleine Änderungen eines Parameters zu großen Änderungen der Klassifikationsgüte führen können.
Mit einem geeigneten Klassifikator und einem Maß $U\,(\boldsymbol{\theta}; \mathcal{D})$ zur Beurteilung der Klassifikationsgüte auf einem Datensatz $\mathcal{D}$, kann die Suche nach den Pa-

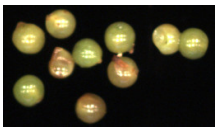
rametern $\boldsymbol{\theta} = (\alpha_{11}, \ldots, \alpha_{KD}, \beta_1, \ldots, \beta_K, \sigma_1, \ldots, \sigma_D, \gamma_1, \ldots, \gamma_D)^\top$ aber auch als Optimierungsproblem aufgefasst werden:

$$\begin{aligned} \boldsymbol{\theta}^* &= \arg\max_{\boldsymbol{\theta}} U(\boldsymbol{\theta}; \mathcal{D}) \\ \text{s.t.} \quad & \alpha_{kd} \geq 0, \quad \textstyle\sum_{k=1}^{K} \alpha_{kd} = 1, \quad 0 < \beta_k \leq 0, \quad \sigma_d, \gamma_d > 0. \end{aligned} \tag{1.8}$$

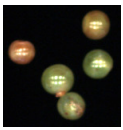
Die große Menge an Parametern, die Nebenbedingungen und die komplizierte Lösungsfläche machen eine geschlossene Optimierung von $U(\boldsymbol{\theta}; \mathcal{D})$ allerdings schwierig. Glücklicherweise muss hier – wie bei vielen maschinellen Lernverfahren – nicht eine optimale Parameterkombination $\boldsymbol{\theta}^*$ gefunden werden. Vielmehr reicht es, die Aufgabe „gut genug" zu lösen. Im Gegenteil ist es sogar vorteilhaft *nicht* eine global optimale Lösung zu finden, da diese neben den Parametern auch von der Stichprobe $\mathcal{D}$ abhängt. Eine optimale Lösung in Bezug zu $\mathcal{D}$ ist mit neuen Daten nicht notwendigerweise ebenfalls optimal. Perfekte Klassifikation auf $\mathcal{D}$ bedeutet im Regelfall sogar eine geringere Klassifikationsleistung auf neuen Daten – in diesem Fall liegt Overfitting vor. Dieses Thema wird in Abschnitt 2.4 erneut aufgegriffen.
Hier wure ein genetischer Algorithmus zur Lösung des Optimierungsproblems verwendet. Genetische Algorithmen sind heuristische Optimierungsverfahren, die auch in hoch- und sogar unendlich-dimensionalen Suchräumen mit nichtkonvexen Lösungsflächen gute Lösungen finden können [Mit98]. Da genetische Algorithmen in Abschnitt 2.3.2 genauer behandelt werden, soll an dieser Stelle auf Details verzichtet werden. Hier ist es nur wichtig, dass ein genetischer Algorithmus eine gerichtete randomisierte Suche ist, die approximativ optimale Lösungen findet, also suboptimale Lösungen $\tilde{\boldsymbol{\theta}}$ mit $U(\boldsymbol{\theta}^*; \mathcal{D}) - U\left(\tilde{\boldsymbol{\theta}}; \mathcal{D}\right) < \varepsilon$ für $\varepsilon > 0$.

1.4.2 Validierung

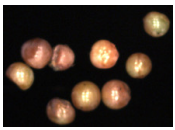
Um den beschriebenen Ansatz zu validieren, wurde ein Sortierproblem aus der Lebensmittelinspektion herangezogen: die Sortierung von gesunden



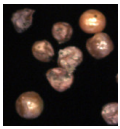
(a) Reife Beeren



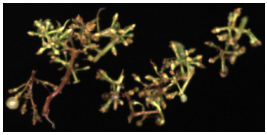
(b) Unreife Beeren



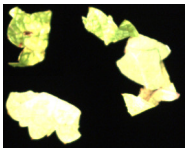
(c) Essigbeeren



(d) Verfaulte Beeren



(e) Rappen



(f) Blätter

Abbildung 1.6: Beispielaufnahmen von Rieslingbeeren und Pflanzenteilen.

Weinbeeren gegen unreife und kranke Beeren sowie Pflanzenteile. Die Stichprobe enthielt Beeren und Pflanzenteile der Weinsorten Spätburgunder, Weißburgunder und Riesling. Das Schüttgut wurde mit weißem, diffusem Licht beleuchtet und gegen einen dunklen Hintergrund aufgenommen. Da Spätburgunderbeeren ebenfalls sehr dunkel sind, wurde hier der Blaukanal der Kamera gegen einen Nahinfrarotkanal ausgetauscht, wodurch die Bilder einen deutlichen Blaustich bekommen. In allen drei Fällen waren mehr Beispiele der Gut- als der Schlechtklasse vorhanden. Eine Übersicht über die Datensätze findet sich in Tabelle 1.1. Beispielaufnahmen der Rieslingbeeren finden sich in Abbildung 1.6.

Die automatische Parametersuche wurde mit den Einstellungen eines menschlichen Experten verglichen. Die Merkmale der manuellen Parametereinstellung wurden mit einem regelbasierten Klassifikator klassifiziert. Die Regeln dieses Klassifikators wurden ebenfalls vom menschlichen Experten eingestellt. Bei der automatischen Suche wurde stattdessen ein Entscheidungsbaumklassifikator (Abschnitt 2.4.5) verwendet, der ebenfalls anhand der Trainingsdaten

Tabelle 1.1: Übersicht über die zur Evaluation verwendeten Datensätze.

Weinsorte	**Farbraum**	**Anzahl Beispiele**		
		Gut	**Schlecht**	**Verhältnis**
Spätburgunder	RG-NIR	2 416	641	3,77 : 1
Weißburgunder	RGB	332	291	1,14 : 1
Riesling	RGB	1 061	235	4,51 : 1

konstruiert wurde. In beiden Fällen wurde die Klassifikationsgüte mit Matthews' Correlation Coefficient (MCC) [Mat75] bewertet:

$$\mathrm{MCC} = \frac{N_{\mathrm{tp}}\, N_{\mathrm{tn}} - N_{\mathrm{fp}}\, N_{\mathrm{fn}}}{\sqrt{(N_{\mathrm{tp}} + N_{\mathrm{fp}})\,(N_{\mathrm{tp}} + N_{\mathrm{fn}})\,(N_{\mathrm{tn}} + N_{\mathrm{fp}})\,(N_{\mathrm{tn}} + N_{\mathrm{fn}})}}. \tag{1.9}$$

Hier bezeichnen N_{tp}, N_{tn}, N_{fp} und N_{fn} die Anzahl der richtig positiv, richtig negativ, falsch positiv und falsch negativ klassifizierten Beispiele. Matthews' Correlation Coefficient (MCC) kann als Korrelationskoeffizient zwischen der Vorhersage des Klassifikators und der Ground Truth interpretiert werden. $\mathrm{MCC} = 1$ bedeutet perfekte Klassifikation, $\mathrm{MCC} = 0$ ist äquivalent mit zufälliger Klassenzuweisung und $\mathrm{MCC} = -1$ bedeutet, dass der Klassifikator perfekt, aber mit vertauschten Klassen klassifiziert. MCC und andere Methoden der Klassifikatorbewertung werden in Abschnitt 2.5 genauer behandelt. Das zu optimierende Gütemaß wurde ebenfalls mit MCC realisiert,

$$U\left(\boldsymbol{\theta}; \mathcal{D}\right) := \overline{\mathrm{MCC}}_{\boldsymbol{\theta}} - s_{\mathrm{MCC}_{\boldsymbol{\theta}}}, \tag{1.10}$$

wobei $\overline{\mathrm{MCC}}_{\boldsymbol{\theta}}$ den mittleren MCC und $s_{\mathrm{MCC}_{\boldsymbol{\theta}}}$ die Streuung des MCC über eine 10-fold Cross Validation (siehe Abschnitt 2.5) der Klassifikation unter Verwendung von $\boldsymbol{\theta}$ bezeichnet.
Die Ergebnisse sind in Tabelle 1.2 zusammengefasst. Hier zeigt sich deutlich,

Tabelle 1.2: Vergleich der Klassifikationsgüte der manuellen und automatischen Parametersuche. Bei der automatischen Parametersuche ist zusätzlich die empirische Standardabweichung der Klassifikationsgüte über eine stratifizierte 5-fold Cross Validation angegeben.

Weinsorte	**MCC / Parameterselektion**	
	manuell	**automatisch**
Spätburgunder	0.47	0.71 $\pm$0.04
Weißburgunder	0.86	0.93 $\pm$0.08
Riesling	0.88	0.99 $\pm$0.20

dass die automatische Parametereinstellung der manuellen Parametereinstellung überlegen ist. Einzig die Klassifikationsgüte bei Rieslingbeeren schwankt sehr stark, da der Datensatz hier deutlich mehr Positiv- als Negativbeispiele enthält, was wiederum einen systematischen Klassifikationsfehler des Entscheidungsbaums nach sich zieht. In diesem Fall hat ein menschlicher Experte einen Vorteil, da der Mensch diese Faktoren bei der Parametereinstellung beachten und unter Verwendung von Vorwissen korrigieren kann. Bei der Klassifikation von Spätburgunder erreicht die Maschine hingegen eine deutlich bessere Klassifikationsgüte als der Mensch. Der Grund hierfür sind die Bilddaten, bei denen der Blaukanal durch einen Infrarot-Kanal ausgetauscht wurde. Der Mensch hat Probleme, diese ungewohnten Bildinformationen richtig zu interpretieren und in geeignete Klassifikationsregeln zu überführen. Da ein Computer die Daten nicht semantisch interpretiert, sondern gewissermaßen blind optimiert, ist die automatische Parametereinstellung hiervon nicht betroffen.

Das oben beschriebene Verfahren und die Ergebnisse wurden in [RB14b] und [RLB15a] veröffentlicht. Diese Ergebnisse zeigen, dass datengetriebene, lernende Verfahren auch in der automatischen Sichtprüfung großes Potential haben.

1.5 Zielsetzung der Arbeit

Diese Arbeit bietet einen Beitrag zur Auslotung dieses Potentials und stellt konkrete Ansätze zur Lösung verschiedener Inspektionsaufgaben vor. Es wird untersucht, wie der Stand der Technik in der optischen Inspektion mit Verfahren des maschinellen Lernens verbessert werden kann. Exemplarisch dient hierzu die Schüttgutsortierung. Insbesondere werden folgende drei Fragestellungen untersucht:

1. Wie kann das Design der Bilderfassung unterstützt bzw. automatisiert werden?
2. Wie können für ein gegebenes Sortierproblem diskriminative Merkmale selektiert bzw. aus Daten gelernt werden?
3. Welche Klassifikationsverfahren eignen sich für den Einsatz in der Schüttgutsortierung?

Damit werden drei wesentliche Teile der Bildverarbeitungskette – Bilderfassung, Merkmalsextraktion und Klassifikation – abgedeckt. Für jeden dieser Bereiche wurden speziell auf die Bedürfnisse und Rahmenbedingungen der visuellen Inspektion angepasste Verfahren entwickelt und anhand von Datensätzen aus der Schüttgutsortierung evaluiert.

1.6 Aufbau der Arbeit

Der Rest dieser Arbeit gliedert sich wie folgt: In Kapitel 2 wird zunächst auf die Theorie der Mustererkennung als das Fundament dieser Arbeit eingegangen. Dies umfasst die Definition der Grundbegriffe der Mustererkennung in Abschnitt 2.1, die Definition von Klassifikationsmerkmalen und verschiedene Verfahren zur Extraktion und Selektion dieser Merkmale in Abschnitt 2.2, einen kurzen Überblick der für diese Arbeit relevanten Optimierungsverfahren in Abschnitt 2.3 sowie die Beschreibung von Klassifikationsmethoden in Abschnitt 2.4 und deren empirische Bewertung in Abschnitt 2.5.

In Kapitel 3 wird die Theorie unter Beachtung der besonderen Rahmenbedingungen der Schüttgutsortierung angewendet. Dabei werden alle relevanten Teile der Mustererkennungskette beachtet: Abschnitt 3.1 stellt ein Verfahren zur automatischen Selektion optischer Filter mit dem Ziel der Optimierung der Bilderfassung vor. Abschnitt 3.2 behandelt die Merkmalsselektion unter Beachtung der für die Extraktion benötigten Rechenzeit. Abschnitt 3.3 und Abschnitt 3.4 stellen Methoden vor, mit denen problemangepasste Form- bzw. Farb- und Texturmerkmale aus einer Stichprobe abgeleitet werden können. Abschnitt 3.5 stellt schließlich ein Verfahren zur schnellen Einklassenklassifikation vor.
Kapitel 4 schließt mit einer Zusammenfassung und einem Ausblick auf zukünftige Arbeiten.

2 Theorie

Bevor in Kapitel 3 die Anwendung maschineller Lernverfahren im Hinblick der Zielsetzung dieser Arbeit beschrieben wird, sollen hier zunächst die zugrunde liegenden Begriffe und das mathematische Rahmenwerk erläutert werden. Da es sich bei der Schüttgutsortierung um eine Klassifikationsaufgabe handelt, ist dieses Rahmenwerk durch die Theorie der Mustererkennung gegeben. In diesem Kapitel wird daher zunächst kurz auf die Grundbegriffe der Mustererkennung – Objekt, Muster, Merkmal, Klasse, Klassifikator, etc. – eingegangen. Im Anschluss werden die für diese Arbeit zentralen Konzepte Merkmale und Klassifikator eingehender behandelt. Da viele Klassifikationsverfahren auf der Optimierung von Parametern beruhen und Optimierung auch in Kapitel 3 eine Rolle spielt, wird kurz auf im maschinellen Lernen gängige, numerische Optimierungsverfahren eingegangen. Das Kapitel schließt mit Methoden zur empirischen Bewertung der Klassifikationsleistung anhand von Datensätzen. Die Anwendung dieser Methoden im Kontext der visuellen Inspektion wird in Kapitel 3 erörtert.

2.1 Grundbegriffe der Mustererkennung

Das Ziel der Mustererkennung ist es, Objekte anhand ihrer Ähnlichkeit zueinander in Klassen einzuteilen. Ein Objekt kann dabei sowohl ein physikalischer Gegenstand oder ein immaterielles Konzept sein. Formal ist ein Objekt o ein Element der Welt bzw. Domäne Ω. Die Ähnlichkeit zwischen zwei Objekten $o, p \in \Omega$ wird durch eine Äquivalenzrelation $o \sim p$ kodiert. Eine Klasse ω ist eine von einem Prototypen $p \in \Omega$ und einer über Ω

definierten Äquivalenzrelation $o \sim p$ induzierte Untermenge der Domäne, d. h. $\Omega \supset \omega(p) = \{o \in \Omega \,|\, o \sim p\}$. Die verschiedenen Klassen $\omega_1, \ldots, \omega_C$ partitionieren dabei die Domäne, d. h. die Klassen sind paarweise disjunkt, $\omega_i \cap \omega_k = \emptyset$ für $\omega_i \neq \omega_k$ und die Domäne wird vollständig von den Klassen überdeckt, d. h. $\bigcup_{c=1}^{C} \omega_c = \Omega$.
Der Schritt vom Objekt o zur zugehörigen Klasse ω geschieht dabei wie in Abbildung 2.1 dargelegt: Ein Objekt $o \in \Omega$ wird durch Sensierung in ein Muster $\boldsymbol{p} \in \mathbb{P}$ überführt. Ein Muster ist dabei eine Ansammlung von beobachteten oder gemessenen Eigenschaften und mithin eine Repräsentation des Objekts und $\mathbb{P}$ bezeichnet die Menge bzw. der Raum aller möglichen Muster. Das gleiche Objekt wird dabei nicht immer auf das gleiche Muster abgebildet, da die Sensierung in der Regel mit Unsicherheiten belegt ist und geometrische Transformationen des Objekts in unterschiedlichen Mustern resultieren kann.
Im nächsten Schritt werden aus dem Muster $\boldsymbol{p}$ eine Reihe von Merkmalen extrahiert und in einem Vektor $\boldsymbol{x} \in \mathbb{M}$ des Merkmalsraums $\mathbb{M}$ zusammengefasst. Merkmale sind also aus dem Muster ableitbare Eigenschaften, die das Muster und somit das Objekt charakterisieren. Ebenso wie beim Übergang von Objekten zu Mustern ist nicht garantiert, dass ein Objekt immer mit dem gleichen Merkmal assoziiert wird. Die Merkmalsextraktion $\boldsymbol{x} = \psi(\boldsymbol{p})$ wird aber als deterministisch vorausgesetzt, sodass ein Muster immer auf die gleichen Merkmale abgebildet wird. Der Übergang von Mustern zu Merkmalen ist dabei nicht klar definiert. Insbesondere können die Muster selbst als Merkmale verwendet werden oder mehrere Stufen der Merkmalsextraktion zwischengeschaltet werden. Die Merkmale bilden schließlich die Basis für die Klassifikation: Ein Klassifikator $\hat{\omega}(o)$ weist einem Objekt anhand der vom zugehörigen Muster extrahierten Merkmale eine der Klassen $\omega_1, \ldots, \omega_C$ zu.
Im Kontext dieser Arbeit ist die Domäne durch das Sortierproblem gegeben. Die Objekte sind die zu sortierenden Objekte des Materialstroms und die Klassen sind die Gut- und Schlechtfraktion. Die korrespondierende Ähnlichkeitsbeziehung ist dadurch implizit gegeben. Die Muster sind hier fast

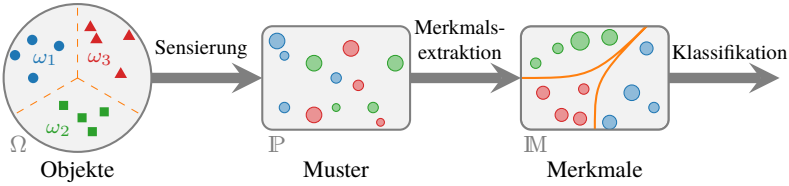


Abbildung 2.1: Abstrahierter Prozess der Mustererkennung.

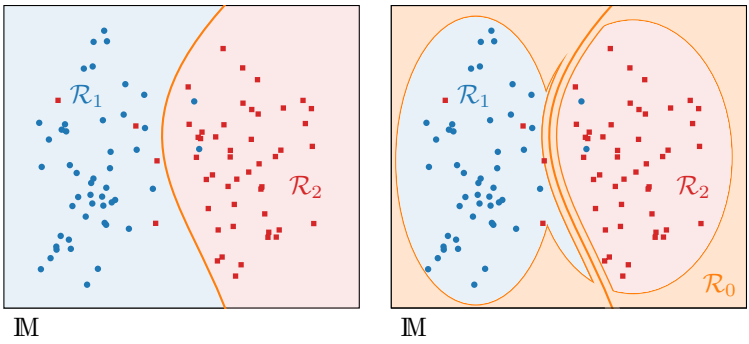
immer Bilddaten, allerdings können diese Bilddaten durch ein beliebiges bildgebendes Verfahren wie etwa Röntgen- oder auch Ultraschallbildgebung gewonnen werden. Die Merkmale und Klassifikatoren sind Hauptgegenstand dieser Arbeit.

Die Festlegung der Merkmale und des Klassifikators erfolgt auf Grundlage einer (endlichen) Menge von N Objekten $o_1, \ldots, o_N$ mit bekannten Klassenzugehörigkeiten $\omega(o_n)$, kurz ω_n, für $n = 1, \ldots, N$. Ziel ist es, einen Klassifikator $\hat{\omega}(o)$ zu finden, der die Objekte mit möglichst geringem Fehler der korrekten Klasse zuordnet, d. h. $\hat{\omega}(o_n) = \omega(o_n)$ für fast alle $n = 1, \ldots, N$. Gleichzeitig soll dieser Klassifikator auch generalisieren, d. h. neue, bisher ungesehene Objekte korrekt klassifizieren.

Wie erwähnt und in Abbildung 2.1 angedeutet, geschieht dies auf Grundlage der von den Objekten abgeleiteten Merkmalen $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$. Da der Klassifikator jedem Objekt o und damit auch jedem möglichen Merkmal $\boldsymbol{x}$ eine Klasse zuweist, ergibt sich implizit eine Partitionierung des Merkmalsraums $\mathbb{M}$ in C disjunkte, nicht notwendigerweise zusammenhängende Regionen $\mathcal{R}_1, \ldots, \mathcal{R}_C$ (siehe Abbildung 2.2a). Dabei ist die Region $\mathcal{R}_c$ mit einer Klasse ω_c assoziiert.

Durch Umkehren dieser Argumentation ist ein Klassifikator durch eine Partitionierung des D-dimensionalen Merkmalsraums $\mathbb{M}$ in die Regionen $\mathcal{R}_c$ gegeben:

$$\hat{\omega}(o) = \omega_c \quad \Leftrightarrow \quad \boldsymbol{x} \in \mathcal{R}_c. \tag{2.1}$$



(a) Ohne Rückweisungsoption **(b)** Mit Rückweisungsoption

Abbildung 2.2: Entscheidungsgebiete und -grenzen im Merkmalsraum.

Ein Objekt o wird also diejenige Klasse ω_c zugewiesen, die mit der Region $\mathcal{R}_c$ assoziiert ist, welche die von o abgeleiteten Merkmale $\boldsymbol{x}$ enthält. In diesem Kontext werden die $\mathcal{R}_c$ auch Entscheidungsgebiete genannt. Die Ränder $\partial\mathcal{R}_c$ der Entscheidungsgebiete heißen Entscheidungsgrenzen.
Diese Formulierung ist unabhängig vom verwendeten Klassifikator gültig. Verschiedene Klassifikationsverfahren unterscheiden sich im Wesentlichen durch die Definition bzw. Parametrierung der Entscheidungsgrenzen. In Abschnitt 2.4 werden verschiedene Ansätze für die Definition der Entscheidungsgrenzen diskutiert. Eine manuelle Festlegung der Entscheidungsgrenzen ist nur bei niedrigdimensionalen Merkmalsräumen praktikabel. Stattdessen werden die Parameter anhand einer Lernstichprobe $(\boldsymbol{x}_n, \omega_n)_{n=1}^N$ von N Merkmalsvektoren und zugehörigen Klassen ermittelt. Die Festlegung der Parameter ist in der Regel mit hohem Rechenaufwand verbunden und geschieht deshalb vorab in einer dedizierten Lernphase. Die Auswertung des Klassifikators in der darauf folgenden Betriebsphase ist hingegen mit verhältnismäßig geringem Aufwand verbunden. Mathematisch unsauber wird die Menge der Merkmalsvektoren der Lernstichprobe $\mathcal{D} = \{\boldsymbol{x}_n \mid n = 1, \ldots, N\}$ genau wie die Menge an Tupeln aus Merkmalsvektoren und assoziierter

Klasse $\mathcal{D} = \{(\boldsymbol{x}_n,\omega_n) \,|\, n = 1,\ldots,N\}$ in der Literatur häufig ebenfalls als Lernstichprobe bezeichnet. Zugunsten der besseren Lesbarkeit wird hier ebenso verfahren, wobei sich die jeweilige Bedeutung aus dem Kontext ergibt.
Eine fundamentale Annahme der Mustererkennung ist, dass die Entstehung der Objekte, und damit auch die Entstehung der Muster und der Merkmale, durch einen Zufallsprozess modelliert werden kann. Die Lernstichprobe wird also durch eine Verbunddichte $p(\boldsymbol{x}_1,\omega_1,\ldots,\boldsymbol{x}_N,\omega_N)$ beschrieben. Hierbei wird vereinfachend angenommen, dass die Paare $(\boldsymbol{x}_n,\omega_n)$ unabhängig und identisch verteilt (i. i. d. – independent and identically distributed) sind,wodurch die Verbunddichte faktorisiert werden kann:

$$p(\boldsymbol{x}_1,\omega_1,\ldots,\boldsymbol{x}_N,\omega_N) = \prod_{n=1}^{N} p(\boldsymbol{x}_n,\omega_n)\,. \tag{2.2}$$

In der Praxis trifft diese Annahme in den allermeisten Fällen nicht zu. Zum Beispiel stimmt die Annahme einer identischen Verteilung nicht, wenn die Objekte aus unterschiedlichen Quellen, etwa verschiedenen Produktionsanlagen, stammen. Dennoch wird die i. i. d.-Annahme oft *ungefähr* eingehalten. Im Beispiel mit den unterschiedlichen Produktionsanlagen sind die Daten bedingt unabhängig. Zudem erleichtert, bzw. ermöglicht diese Annahme erst die theoretische Behandlung der Mustererkennung. Insbesondere kann gezeigt werden, dass eine optimale Klassifikation im Sinne der geringsten Fehlerwahrscheinlichkeit durch

$$\hat{\omega}(o) = \arg\max_{\omega} p(\boldsymbol{x},\omega) \quad \text{mit } \boldsymbol{x} = \psi(\boldsymbol{p}(o)) \tag{2.3}$$

erreicht wird [BRN17]. Gleichzeitig zeigt dieses Ergebnis einen Mechanismus zur praktischen Implementierung eines Klassifikators auf: Ziel ist es, die Dichte $p(\boldsymbol{x},\omega)$ anhand der Stichprobe $\mathcal{D}$ festzulegen. Dieser Klassifikator wird in Abschnitt 2.4 erneut aufgegriffen.

Die probabilistische Formulierung deutet aber auch auf eine wichtige Tatsache der Mustererkennung hin: Eine perfekte, d. h. fehlerfreie Klassifikation ist nur dann möglich, wenn sich die klassenbedingten Merkmalsdichten nicht überlagern, wenn also $\text{supp}\{p(\boldsymbol{x} \mid \omega_i)\} \cap \text{supp}\{p(\boldsymbol{x} \mid \omega_k)\} = \emptyset$ für jedes Paar von Klassen $\omega_i \neq \omega_k$ gilt. Wenn umgekehrt $p(\boldsymbol{x} \mid \omega_i) \equiv p(\boldsymbol{x} \mid \omega_k)$ gilt, können die Klassen ω_i und ω_k anhand der Merkmale $\boldsymbol{x}$ nicht unterschieden werden. In der Praxis trifft in der Regel keines dieser beiden Extreme zu. Das bedeutet, dass in Regionen von $\mathbb{M}$, in denen sich die Dichten überschneiden, Fehlklassifikationen auftreten können. Dieser Fehler ist irreduzibel.
Die Gründe für diesen irreduziblen Fehler liegen im Prozess der Mustererkennung (siehe Abbildung 2.1): Die Sensierung der Objekte ist mit Unsicherheiten behaftet und die Extraktion der Merkmale geht zwangsläufig mit Informationsverlust einher. Zwei Objekte unterschiedlicher Klassen können sehr ähnliche Muster erzeugen und aus zwei unterschiedlichen Mustern können sehr ähnliche Merkmale extrahiert werden. Beispielsweise sind die visuellen Erscheinungen (Muster) von Äpfeln und Nashi-Birnen ebenso wie die Formparameter (Merkmale) von Äpfeln und Orangen sehr ähnlich. Zur Unterscheidung wären andere Muster (etwa spektrale Messungen) bzw. andere Merkmale (z. B. Farbe) nötig. Andererseits kann bereits die Klasseneinteilung uneindeutig sein: Sind Schnabeltiere Säuger, Reptilien oder Vögel?[1]
Dieses Beispiel verdeutlicht auch einen weiteren Aspekt der Mustererkennung, der bisher nicht betrachtet wurde: Wenn die Entscheidung zwischen zwei Klassen mit hoher Unsicherheit verbunden ist, sollte es die Möglichkeit geben, die Klassifikation zurückzuweisen. Formal wird diese Rückweisungsoption durch eine zusätzliche Klasse ω_0 modelliert. Wie die anderen Klassen korrespondiert nach Gleichung (2.1) auch ω_0 mit einem Entscheidungsgebiet $\mathcal{R}_0$. Eine Rückweisungsregion um die Entscheidungsgrenzen zwischen zwei Regionen $\mathcal{R}_i, \mathcal{R}_k$ mit $i,k > 0$ entspricht beispielsweise der Rückweisung von unsicheren Klassifikationsentscheidungen.

[1] Es sind Säugetiere [Grü+04].

Es gibt noch eine weitere wichtige Motivation für die Rückweisungsoption. Bei der Einteilung der Objekte $o \in \Omega$ in Klassen ω_c wurde angenommen, dass alle Klassen bekannt sind, bzw. dass alle möglichen Klassen in der Stichprobe repräsentiert sind. Diese Annahme wird auch als Annahme der geschlossenen Welt bezeichnet: Alle Objekte werden genau einer der definierten Klassen zugeordnet. Mit der Rückweisungsoption kann die Klasse ω_0 als zusätzliche Klasse betrachtet werden, die alle unbekannten Objekte fasst und somit unvollständiges Wissen über die Klassenstruktur der Welt Ω modelliert. Klassifikation mit Rückweisungsoption ω_0 wird deshalb auch als Klassifikation unter der Annahme einer offenen Welt bezeichnet.
Diese Annahme ist in der visuellen Inspektion und insbesondere bei der Schüttgutsortierung enorm wichtig, da hier die Defektklassen oft nur teilweise bekannt sind. Außerdem können während des Betriebs neue Defektklassen auftauchen, wenn beispielsweise eine neuartige Fertigungsanlage in Betrieb genommen wird oder der Zulieferer von Rohmaterial gewechselt wurde.
In der Praxis kann die Rückweisungsoption durch einen zusätzlichen Schritt nach der eigentlichen Klassifikation implementiert werden. Typische Ansätze dazu finden sich z. B. in Beyerer et al. [BRN17]. Alternativ kann die Rückweisungsoption auch – wie in den Abschnitten 3.4 und 3.5 gezeigt – direkt in den Klassifikator eingebaut werden. Unabhängig von der Implementierung entspricht die Rückweisungsoption einer Entscheidungsregion $\mathcal{R}_0$, wie sie in Abbildung 2.2b skizziert ist: Objekte werden zurückgewiesen, wenn die zugehörigen Merkmale nahe an Entscheidungsgrenzen zwischen zwei bekannten Klassen liegen oder wenn die Merkmale in dünn besetzten Regionen des Merkmalsraums liegen.
Die hier diskutierte Formalisierung gilt unabhängig von den vorliegenden Objekten, Mustern, Merkmalen und Klassifikatoren und bietet ein Rahmenwerk für die Analyse von Mustererkennungsproblemen. Insbesondere die Interpretation von Klassifikatoren als Entscheidungsgrenzen bzw. -regionen im Merkmalsraum ist hilfreich. Im Folgenden werden die beiden Schlüsselelemente Merkmale und Klassifikator genauer betrachtet.

2.2 Merkmale

Wie eingangs erläutert, beschreiben Merkmale die charakterisierenden Eigenschaften eines zu klassifizierenden Objekts und bilden damit die Grundlage zur Klassifikation. Merkmale können von Menschen interpretierbare Beschreibungen, z. B. die Farbe eines Objekts oder dessen Länge in m, aber auch abstrakte Kodierungen, wie z. B. die Energie innerhalb eines Spektralbereichs eines Audiosignals oder aus einem Bild berechnete Texturcodes, darstellen.
In der deskriptiven Statistik werden Merkmale in verschiedene Skalenniveaus eingeteilt. Stevens [Ste46] gruppiert Merkmale in vier Klassen mit zunehmender Aussagekraft und zunehmender Einschränkung der zulässigen Transformationen: Nominalskala, Ordinalskala, Intervallskala und Verhältnisskala. Nominalskalierte Merkmale nehmen diskrete Werte ein, die als Bezeichner verstanden werden können (z. B. Automarken, Postleitzahlen). Ordinalskalierte Merkmale sind zusätzlich mit einer Ordnung ausgestattet (z. B. Schulnoten, Güteklassen). Intervallsaklierte Merkmale erlauben weiterhin einen Abstand zwischen zwei Merkmalen zu berechnen, besitzen aber keinen natürlichen Nullpunkt (z. B. Temperatur in °C, Höhe über dem Meeresspiegel). Verhältnisskalierte Merkmale hingegen besitzen einen solchen und erlauben es zwei Merkmale ins Verhältnis zu setzen (z. B. Temperatur in K, Kontostand). Gelegentlich, z. B. in Beyerer et al. [BRN17], wird die Absolutskala als zusätzliche höchstrangige Skala definiert. Absolutskalierte Merkmale besitzen alle Eigenschaften von verhältnisskalierten Merkmalen und sind zusätzlich mit einer natürlichen Einheit ausgestattet. Diese Merkmale repräsentieren in der Regel Zählwerte wie Einwohnerzahlen, die Euler-Charakteristik etc.
In der praktischen Anwendung der Mustererkennung wird allerdings selten so feingliedrig unterschieden. Merkmale werden lediglich als metrisch oder nichtmetrisch kategorisiert. Metrische Merkmale werden durch reelle Zahlen kodiert, besitzen eine Ordnung und erlauben die Berechnung von Abständen zwischen Merkmalen. Metrische Merkmale werden dabei wie

verhältnisskalierte Merkmale behandelt. Insbesondere werden Operationen wie die Mittelwertbildung oder Normierung angewendet, auch wenn diese streng genommen nicht notwendigerweise definiert sind. Alle verbleibenden Arten von Merkmalen werden als nichtmetrisch kategorisiert. Merkmale der Nominal- und Ordinalskala sind nach dieser Typisierung nichtmetrisch. Merkmale der Intervall-, Verhältnis- und Absolutskala sind dagegen (in der Regel) metrisch.

Die Sammlung der Merkmale, die von einem gegebenen Objekt extrahiert wurden, wird als ein (Merkmals-)Vektor $\boldsymbol{x}$ in einem hochdimensionalen Merkmalsraum $\mathbb{M}$ beschrieben. Im Allgemeinen wird dabei von einem Hilbertraum $\mathbb{M} = \mathbb{R}^D$, bzw. einer Einbettung in einen solchen ausgegangen. Diese Annahme erleichtert die mathematische Formulierung der gängigen Klassifikationsverfahren wie etwa der logistischen Regression (siehe Abschnitt 2.4.2) oder der Support Vektor Maschine (siehe Abschnitt 2.4.4). Im Folgenden werden die beiden Begriffe „Merkmal" und „Merkmalsvektor" synonym verwendet. Die jeweilige Bedeutung ergibt sich aus dem Kontext.

Nichtmetrische Merkmale können durch die sogenannte One-Hot-Kodierung ebenfalls in diesem Raum repräsentiert werden. Angenommen, ein nichtmetrisches Merkmal habe K mögliche Ausprägungen. Dann kann das Merkmal $x = k$ durch den Vektor

$$\boldsymbol{x} = \boldsymbol{e}_k = \left(\mathbb{1}[k = 1], \mathbb{1}[k = 2], \ldots, \mathbb{1}[k = K]\right)^\top \tag{2.4}$$

kodiert werden, also den Vektor mit einer 1 an der k-ten Stelle und 0 an allen anderen Stellen. Natürlich geht durch diese Kodierung eine mögliche Ordnung der Merkmale verloren. Andererseits wird eine Struktur auferlegt, die die Merkmale eigentlich nicht besitzen.

Unabhängig von der Typisierung und Kodierung sollen Merkmale bestimmte Eigenschaften besitzen. Die wichtigste Forderung ist die Forderung der Relevanz der Merkmale. Formal ist ein Merkmal $\boldsymbol{x}$ relevant für die Klasse ω, wenn $\boldsymbol{x}$ und ω stochastisch abhängig sind, d. h. wenn $p(\boldsymbol{x}, \omega) \neq p(\boldsymbol{x})\, P(\omega)$.

Die Relevanz der Merkmale $\boldsymbol{x}$ kann durch die Transinformation

$$I(\boldsymbol{x}\,;\,\omega) = \mathbb{E}_{\boldsymbol{x},\omega}\left\{\log\frac{p(\boldsymbol{x},\omega)}{p(\boldsymbol{x})P(\omega)}\right\} \tag{2.5}$$

quantifiziert werden. In diesem Sinne ist $I(\boldsymbol{x}\,;\,\omega)$ ein Maß der Abhängigkeit von $\boldsymbol{x}$ und ω. Weiterhin sollten die individuellen Merkmale $[\boldsymbol{x}]_1, \ldots, [\boldsymbol{x}]_D$ möglichst wenig redundant sein. Wie die Relevanz kann auch die Redundanz durch informationstheoretische Maße, hier durch die bedingte Transinformation, quantifiziert werden:

$$I([\boldsymbol{x}]_i\,;\,[\boldsymbol{x}]_k\,|\,\omega) = \mathbb{E}_{[\boldsymbol{x}]_i,[\boldsymbol{x}]_k,\omega}\left\{\log\frac{p([\boldsymbol{x}]_i\,,[\boldsymbol{x}]_k\,|\,\omega)}{p([\boldsymbol{x}]_i\,|\,\omega)\,p([\boldsymbol{x}]_k\,|\,\omega)}\right\}. \tag{2.6}$$

Geringe Redundanz ist wünschenswert, da dadurch die Dimensionalität des Merkmalsraums $\mathbb{M}$ minimiert wird. Ein niedrigdimensionaler Merkmalsraum ist einerseits einfacher zu interpretieren und zu inspizieren. Andererseits benötigen die Entscheidungsgrenzen und damit die Klassifikatoren so weniger zu schätzende Parameter, was in der Regel die Klassifikationsgüte erhöht. Ein hochdimensionaler Merkmalsraum führt hingegen meist zu einer Verminderung der Klassifikationsleistung. Dieses Phänomen ist als Fluch der Dimensionalität (nach Bellman [Bel61]) oder Hughes Phänomen [Hug68] bekannt und betrifft nicht nur die Mustererkennung.

Da die Gewinnung der Muster selbst bei bestmöglicher Auslegung der Bilderfassung mit geometrischen Verzerrungen und Rauschen einhergeht, ist es zudem wünschenswert, dass die Merkmale gegenüber projektiven, linearen Transformationen, mindestens aber gegenüber Ähnlichkeitsabbildungen (d. h. Verschiebung, Rotation und Skalierung) invariant sind: Muster, die mit solchen Abbildungen transformiert wurden, sollen (annähernd) das gleiche Merkmal liefern. Auch sollten die Merkmale gegenüber Rauscheinflüssen und Verdeckungen robust sein, damit eine korrekte Klassifikation auch in diesen Fällen möglich ist. Im Folgenden wird auf verschiedene Techniken

für die Gewinnung von Merkmalen eingegangen, die zur Beschreibung von Bildmustern geeignet sind.

2.2.1 Visuelle Merkmale

Auch wenn die Zuordnung nicht immer eindeutig ist, lassen sich visuelle Merkmale in drei Klassen einteilen, die verschiedene Aspekte der Erscheinung beschreiben: Formmerkmale, Farbmerkmale und Texturmerkmale.
Formmerkmale beschreiben die geometrischen Eigenschaften der den Objekten zugehörigen Muster und können ihrerseits wieder in Flächen- und Konturmerkmale unterteilt werden. Flächenmerkmale beschreiben in der Regel globale Eigenschaften wie das Objektvolumen und topologische Eigenschaften der Muster. Typische Flächenmerkmale sind Durchmesser, Längen der Hauptachsen, Kompaktheit, Füllgrad, Konvexität, Anisometrie, Euler-Charakteristik oder Hu-Momente [Hu62]. Konturmerkmale werden hingegen aus dem Objektprofil abgeleitet und sind daher oft Beschreibungen lokaler Strukturen. Typische Konturmerkmale sind der Objektumfang, Zentroiddistanzen, Tangentenwinkel, Krümmung, Kettencodes [Fre61] oder Fourier-Konturdeskriptoren. Details der Berechnung dieser und weiterer Formmerkmale finden sich z. B. in der Übersicht von Zhang und Lu [ZL04] oder in den Lehrbüchern von Steger et al. [SUW08] und Beyerer et al. [BRN17].
Farbmerkmale beschreiben die Verteilung der Farben der Objekte. In der optischen Inspektion spielen diese Merkmale eine große Rolle, da die Farbe Rückschlüsse auf mechanische, thermische, chemische und biologische Beanspruchung von Materialen zulässt. Dies betrifft insbesondere die Inspektion organischer Materialien. Eine Verfärbung der Oberfläche deutet bei Äpfeln beispielsweise auf eine Druckstelle oder eine verfaulte Frucht hin. Farbmerkmale eignen sich aber auch, um etwa Schweißnähte oder Brandflecken auf nichtorganischen Oberflächen zu detektieren [DCL00]. Die meisten Farbmerkmale beschreiben die Verteilung der Objektfarbe in Form einer Statistik. Einfache Möglichkeiten sind Farbmomente (mittlere Objektfarbe,

Farbstreuung, höhere Momente) oder Zählstatistiken (Histogramme) [SB91; Sha08]. Auch in Abschnitt 1.4 wurden Farbhistogramme als grundlegende Farbstatistik herangezogen. Neben der Wahl der Statistik ist hier auch die Wahl eines geeigneten Farbraums bedeutsam, da sich hierdurch die Nachbarschaftsbeziehungen der Farben verändern. So unterscheiden sich die Farbmomente eines Musters im RGB-Farbraum im Allgemeinen von den Farbmomenten des gleichen Musters im HSV-Farbraum.
Die letzte Gruppe visueller Merkmale sind schließlich Texturmerkmale, welche die Oberflächendetails der Objekte beschreiben. Da die Textur in vielen Aufgaben des Computersehens das diskriminativste Merkmal zur Klassifikation darstellt, sind hier eine Vielzahl von Extraktionsverfahren beschrieben worden. Wie bei den Farbmerkmalen kann Textur über Statistiken wie Haralicks Grauwertmatrixmerkmale [HSD73] beschrieben werden. Eine gute Übersicht solcher Merkmale findet sich z. B. in Jähne et al. [JHG99].
Eine andere Möglichkeit ist die Verwendung einer Kodierung der Oberflächentextur, beispielsweise wie in Beyerer et al. [BRN17] beschrieben durch die Koeffizienten eines autoregressiven Texturmodells. Alternativen sind Texturcodes wie Local Binary Patterns (LBP) [OPM02] oder die modifizierten Zensustransformation [FE04] sowie Beschreibungen in der Frequenzdomäne, etwa die Energieanteile einer diskrete Kosinustransformation (DCT) [HL01] oder die Filterantworten von Gabor-Filterbanken [Lee96]. Diese Beschreibungen lassen sich auch mit statistischen Beschreibungen, z. B. durch ein Histogramm über die Texturcodes des Musters, kombinieren.
Einen ähnlichen Ansatz verfolgen auch Lowes vielseitig einsetzbare Scale-Invariant Feature Transform (SIFT) [Low04] oder die Histogram of Oriented Gradients (HOG)-Deskriptoren von Dalal und Triggs [DT05]. In beiden Verfahren wird die Textur durch Histogramme der Kanten im Muster beschrieben. SIFT-Deskriptoren beschreiben dabei lokale Regionen um bestimmte Schlüsselpunkte, die in einem vorherigen Schritt detektiert wurden. HOG-Deskriptoren sind dagegen eine globale Beschreibung des Bildinhalts, die

durch eine Aufteilung des gesamten Bildes in überlappende Zellen gewonnen wird.
Die hier aufgeführten Merkmale sind nur ein kleiner Ausschnitt der bekannten und veröffentlichten visuellen Merkmalen. Nicht alle diese Merkmale sind für jede Mustererkennungsaufgabe gleich relevant. In der Praxis müssen also zunächst passende Merkmale identifiziert werden. Dies kann manuell durch Inspektion der jeweiligen Merkmalsräume geschehen, was aber aufgrund der großen Menge an verfügbaren Verfahren ein hohes Maß an Wissen und Erfahrung erfordert.

2.2.2 Merkmalsselektion

Die Selektion relevanter Merkmale kann aber auch automatisiert werden. Dafür muss das Problem formalisiert werden:
Gegeben ist eine Stichprobe $(\boldsymbol{p}_n, \omega_n)_{n=1}^{N}$ von N Mustern $\boldsymbol{p}_n$ mit bekannten Klassenzugehörigkeiten ω_n sowie eine Menge $\Psi = \{\psi_d(\cdot) \,|\, d = 1, \ldots, D\}$ von D Merkmalsextraktionsverfahren. Diese Menge induziert einen Merkmalsraum $\boldsymbol{x} = \Psi(\boldsymbol{p}) \in [\mathbb{M}]_\Psi$, wobei $\Psi(\boldsymbol{p}) = (\psi_1(\boldsymbol{p}), \ldots, \psi_D(\boldsymbol{p}))^\top$ der von $\boldsymbol{p}$ extrahierte Merkmalsvektor aller Merkmale in Ψ ist. Gesucht ist eine Untermenge $\mathcal{S}^* \subseteq \Psi$ der Merkmale, die ein Gütemaß $U(\mathcal{S}^*; \mathcal{D})$ maximiert, d. h.

$$\mathcal{S}^* = \underset{\mathcal{S} \subseteq \Psi}{\arg\max}\ U(\mathcal{S}; \mathcal{D})\,. \tag{2.7}$$

Leider ist die Lösung dieses Optimierungsproblems NP-hart, zumindest wenn $\mathcal{S}^*$ gleichzeitig minimalen Umfangs sein soll [AK98]. Die optimale Lösung kann also nur durch einen Brute-Force-Ansatz gefunden werden. Solch ein Ansatz ist in der Praxis aber nur mit kleinen Mengen Ψ von Kandidaten realistisch, da das Gütemaß $U(\mathcal{S}; \mathcal{D})$ für 2^D Untermengen $\mathcal{S} \subseteq \Psi$ ausgewertet werden muss.

Als Alternative haben sich Greedy-Verfahren, in denen entweder sukzessive Merkmale entfernt (Greedy-Elimination) oder hinzugefügt werden (Greedy-Selektion). In der Greedy-Elimination werden ausgehend von der Menge aller Merkmale $\mathcal{F}$ sukzessive Merkmale entfernt, deren Auslassung die Güte am wenigsten verringert,

$$\begin{aligned} \mathcal{S}_{t+1} &:= \mathcal{S}_t \setminus \{\psi_{t+1}\}, \quad \text{mit} \quad \mathcal{S}_0 := \Psi \quad \text{und} \\ \psi_{t+1} &:= \underset{\psi \in \mathcal{S}_t}{\arg\min} \big(U\left(\mathcal{S}_t; \mathcal{D}\right) - U\left(\mathcal{S}_t \setminus \{\psi\}; \mathcal{D}\right) \big). \end{aligned} \tag{2.8}$$

Merkmale werden entfernt, bis die Selektion eine gewünschte Güte erreicht hat, bzw. bis die Elimination eines weiteren Merkmals das Gütemaß zu stark verringert. Bei der Greedy-Selektion werden hingegen ausgehend von der leeren Menge sukzessive Merkmale hinzugefügt, deren Hinzunahme die Güte maximal positiv beeinflusst,

$$\begin{aligned} \mathcal{S}_{t+1} &:= \mathcal{S}_t \cup \{\psi_{t+1}\} \quad \text{mit} \quad \mathcal{S}_0 := \emptyset \quad \text{und} \\ \psi_{t+1} &:= \underset{\psi \in \overline{\mathcal{S}_t}}{\arg\max} \big(U\left(\mathcal{S}_t \cup \{\psi\}; \mathcal{D}\right) - U\left(\mathcal{S}_t; \mathcal{D}\right) \big). \end{aligned} \tag{2.9}$$

Hierbei bezeichnet $\overline{\mathcal{S}_t} := \mathcal{F} \setminus \mathcal{S}_t$ die Menge aller nicht selektierten Merkmale. Wie bei der Greedy-Elimination wird abgebrochen, wenn ein bestimmtes Gütemaß erreicht ist oder bis die Hinzunahme eines weiteren Merkmals die Güte der Selektion nicht signifikant erhöht. Mit beiden Ansätzen benötigt die Gütefunktion im schlimmsten Fall statt $\mathcal{O}(2^D)$ nur noch $\mathcal{O}(D^2)$ Auswertungen.

Unabhängig davon, ob ein Greedy-Ansatz oder eine alternative Selektionsmethode verwendet wird, gibt es grundsätzlich drei Ansätze das Gütemaß zu definieren: Wrapper, Filter und eingebettete Verfahren.

Im Wrapperansatz entspricht das Gütemaß der Selektion $\mathcal{S}_t$ der Klassifikationsgüte eines Klassifikators im induzierten Merkmalsraum $[\mathbb{M}]_{\mathcal{S}_t}$. Als Grundlage dienen dabei meist Verfahren zur Klassifikatorbewertung, die

in Abschnitt 2.5 diskutiert werden. Da die Gütemaße unabhängig vom Klassifikator definiert sind, funktioniert der Wrapperansatz mit jedem Klassifikationsverfahren. Allerdings muss für jede Auswertung des Gütemaßes ein Klassifikator trainiert und evaluiert werden, was den Ansatz verhältnismäßig rechenaufwendig macht. Zudem ist die Selektion an den verwendeten Klassifikator angepasst und führt mit anderen Klassifikatoren möglicherweise zu suboptimalen Ergebnissen. Dies ist insbesondere dann der Fall, wenn zur Selektion ein leistungsfähigerer Klassifikator verwendet wurde als für die anschließende Klassifikation.

Filteransätze beheben dieses Problem durch die Verwendung von Gütemaßen, die die Selektion unabhängig vom Klassifikator bewerten. Ein geeignetes Gütemaß kann beispielsweise aus dem Relevanz- und Redundanzkriterium aus den Gleichungen (2.5) und (2.6) konstruiert werden. Der Filteransatz lässt sich aber auch theoretisch fundierter motivieren. Brown et al. zeigen in ihrem Conditional Likelihood Maximization (CLM) Framework ein allgemeines Vorgehen, aus dem eine Reihe von bekannten Gütemaßen abgeleitet werden können [Bro+12].

Hierfür wird ein hypothetisches Klassifikationsmodell $q(\omega \mid \mathcal{S}(\boldsymbol{p}),\boldsymbol{\theta})$ angenommen. Unter der Annahme, dass die optimalen Modellparameter $\boldsymbol{\theta}$ bekannt sind, kann die optimale Merkmalsselektion $\mathcal{S}$ durch Maximierung der Likelihoodfunktion $\mathfrak{L}\left(\mathcal{S}, \boldsymbol{\theta} \mid \mathcal{D}\right)$ gewonnen werden. Die Autoren zeigen, dass sich die (skalierte) log-Likelihood dieses Modells bezüglich $\mathcal{D}$ im Limit, d. h. mit einer unendlich großen Stichprobe, in drei Terme zerlegen lässt:

$$\begin{aligned}\lim_{N\to\infty} -\frac{1}{N} \log \mathfrak{L}\left(\mathcal{S}, \boldsymbol{\theta} \mid \mathcal{D}\right) &= -\frac{1}{N} \sum_{n=1}^{N} \log q\left(\omega_n \mid \mathcal{S}(\boldsymbol{p}_n), \boldsymbol{\theta}\right) \\ = \mathbb{E}_{\boldsymbol{p},\omega} \left\{ \log \frac{P(\omega \mid \mathcal{S}(\boldsymbol{p}))}{q\left(\omega \mid \mathcal{S}(\boldsymbol{p}), \boldsymbol{\theta}\right)} \right\} &+ I\left(\overline{\mathcal{S}}(\boldsymbol{p})\,;\, \omega \,\middle|\, \mathcal{S}(\boldsymbol{p})\right) + H\left(\omega \mid \Psi(\boldsymbol{p})\right).\end{aligned} \tag{2.10}$$

Die Wahrscheinlichkeit $P(\omega \mid \mathcal{S}(\boldsymbol{p}))$ ist die wahre, aber unbekannte a-posteriori Klassenverteilung. Der erste Term in Gleichung (2.10) ist die Likelihood-ratio zwischen der wahren und der vorhergesagten Klassenverteilung und somit vom verwendeten Klassifikationsmodell q abhängig. Der letzte Term hängt weder vom Klassifikator q, noch von der Selektion $\mathcal{S}$ ab und quantifiziert den nicht reduzierbaren Fehler. Der mittlere Term ist die bedingte Transinformation zwischen den nicht selektierten Merkmalen $\overline{\mathcal{S}}$ und der Klasse ω bei Kenntnis der selektierten Merkmale in $\mathcal{S}$. Dieser Term hängt ausschließlich von der Selektion $\mathcal{S}$ ab und kann somit unabhängig vom Klassifikator q minimiert werden. Somit ist der Filteransatz theoretisch motivierbar.
In diesem Fall ist das Gütemaß also die bedingte Transinformation. Unglücklicherweise ist dieses Gütemaß in der Praxis nicht berechenbar bzw. schätzbar. Brown et al. zeigen aber, dass durch bestimmte vereinfachende Annahmen viele bekannte Heuristiken wie die Mutual Information Feature Selection [Bat94], die Conditional Mutual Information Maximization [Fle04] und das Maximum-Relevance Minimum-Redundancy-Kriterium [PLD05] auf CLM zurückgeführt werden können [Bro+12]. Da die Details des CLM-Ansatzes in Abschnitt 3.1.5 genauer erläutert werden, soll an dieser Stelle auf eine weitere Ausführung verzichtet werden.
Ein wesentlicher Nachteil dieser und anderer auf informationstheoretischen Maßen basierenden Verfahren ist, dass sie mit verhältnismäßig hohem Rechenaufwand verbunden sind. Zur Schätzung der benötigten Merkmalsdichten müssen bei kontinuierlichen Merkmalen zudem entweder Modellannahmen getroffen oder die Merkmale auf diskrete Werte quantisiert werden. Beides geht mit Informationsverlust einher.
Aus diesem Grund wurden auch Filterkriterien entwickelt, die sich nicht auf Maße der Informationstheorie stützen. Ein bekanntes solches Verfahren ist die Correlation-based Feature Selection (CFS) [Hal99]. Das Gütemaß verfolgt die in den Gleichungen (2.5) und (2.6) formulierten Ziele, indem kleine, stark mit der Klassenvariable korrelierte, wenig redundante Selektionen bevorzugt

werden. Mit der mittleren Korrelation zwischen den selektierten Merkmalen und der Klassenvariable $\overline{r_{\psi,\omega}}$, sowie der mittleren Korrelation zwischen den Merkmalen der Selektion $\overline{r_{\psi,\psi}}$ ist das CFS-Kriterium als

$$U_{\text{CFS}}\left(\mathcal{S};\mathcal{D}\right) := \frac{k \cdot \overline{r_{\psi,\omega}}}{\sqrt{k + k\left(k+1\right)\overline{r_{\psi,\psi}}}}, \quad \text{mit } k := |\mathcal{S}| \tag{2.11}$$

definiert. Der Zähler belohnt stark relevante Selektionen, während der Nenner redundante Merkmale bestraft. Die mittleren Korrelationen werden dabei anhand des empirischen Korrelationskoeffizienten $r_{s,t}$ der Stichproben $(s_n)_{n=1}^N$ und $(t_n)_{n=1}^N$ bestimmt, d. h.

$$\overline{r_{\psi,\omega}} = \frac{1}{k}\sum_{\psi_q \in \mathcal{S}} r_{\psi_q,\omega} \quad \text{und} \quad \overline{r_{\psi,\psi}} = \frac{2}{k(k+1)}\sum_{\substack{\psi_q,\psi_p \in \mathcal{S} \\ \psi_q \neq \psi_p}} r_{\psi_q,\psi_p}. \tag{2.12}$$

Die letzte Gruppe von Merkmalsselektionsverfahren bilden schließlich die eingebetteten Verfahren. Hier ist die Merkmalsselektion in einen Klassifikator integriert. Die Merkmalsselektion wird dabei implizit während des Trainingsalgorithmus oder unter Ausnutzung von Zwischenergebnissen des Trainings durchgeführt. Die Selektionen sind somit an den Klassifikator angepasst, benötigen aber keine Auswertung der Klassifikationsgüte. Eingebettete Verfahren bilden also einen Mittelweg zwischen Wrapper- und Filteransätzen. Beispiele für eingebettete Verfahren sind L_1-regularisierte logistische Regression (siehe Abschnitt 2.4.2), Entscheidungsbäume und Random Forests (Abschnitte 2.4.5 und 2.4.6) sowie AdaBoost (Abschnitt 2.4.7).
Auch wenn mit den in diesem Abschnitt besprochenen Verfahren die relevanten Merkmale automatisch aus einer großen Menge von Kandidaten identifiziert werden können, sind diese Merkmale doch noch immer generisch und nicht auf die Mustererkennungsaufgabe zugeschnitten. Wenn die Klassifikationsgüte nicht ausreicht, müssen aufwendigere Methoden wie Feature Engineering, Feature Learning oder Merkmale höherer Ordnung bemüht werden.

2.2.3 Feature Engineering

Feature Engineering bezeichnet die Entwicklung von an das gegebene Problem angepassten Verfahren zur Merkmalsextraktion. Hier ist das Ziel, das vorhandene Domänenwissen auszunutzen, um hochdiskriminative, robuste, störungsinvariante und möglichst niedrigdimensionale Merkmale zu definieren. Diese Merkmale enthalten lediglich die zur Klassifikation benötigte Information und führen somit auch mit sehr einfachen Klassifikatoren zu guten Ergebnissen. In der Praxis ist dieser Ansatz hoch relevant. Pedro Domingos merkt an:

> Feature engineering is the key. At the end of the day, some machine learning projects succeed and some fail. What makes the difference? Easily the most important factor is the features used. — Domingos [Dom12]

Andrew Ng ergänzt in seinem Vortrag „Deep Learning: Machine learning via Large-scale Brain Simulations"[2]

> When we walk around Silicon Valley and look at people doing applied machine learning, most of that effort is [...] spent designing features. — Ng

Leider gilt das No Free Lunch Theorem [Wol96] auch für Feature Engineering: Es gibt keine standardisierte Methode, mit der für alle Fragestellungen gleichermaßen geeignete Merkmale definiert werden können. Jedes Mustererkennungsproblem muss individuell behandelt werden. Dennoch gibt es bewährte Vorgehensweisen des Feature Engineering. Ein typischer Ansatz ist es, den Entstehungsprozess der Muster durch ein parametrisches generatives Modell („Vorwärtsmodell") abzubilden. Dieses Modell bündelt das Wissen über die Problemdomäne. Zur Klassifikation eines Objekts werden zunächst die Modellparameter $\hat{\boldsymbol{\theta}}$ geschätzt, die das Muster $\boldsymbol{p}$ bestmöglich

[2] Video online: `http://youtu.be/Vzg1vESvIBw?t=11m7s`, abgerufen am 06.08.2018.

annähern. Bei statistischen Modellen bietet sich hierfür die Maximierung der (log-)Likelihood an. Mit anderen Modellen kann beispielsweise die mittlere quadratische Abweichung des wahren und des generierten Musters minimiert werden. Die so geschätzten Modellparameter $\hat{\boldsymbol{\theta}}$ dienen schließlich als Klassifikationsmerkmale.

Beyerer [Bey96] liefert ein Beispiel für diesen Ansatz: Zur Beurteilung der Qualität gehonter Zylinderlaufbahnen in Motorblöcken und zur Erkennung unerwünschter Prozesszustände beim Stirnplanfräsen wurde hier ein physikalisch motiviertes, statistisches Modell der Riefentexturen entworfen. Eine Textur wird dabei durch Scharen von Texturprimitiven zusammengesetzt, die einzelne Riefen repräsentieren. Die Riefentexturen einer Schar werden wiederum durch einen mit der Schar assoziierten Zufallsprozess generiert. Die Modellparameter – Anzahl der Riefenscharen, sowie Richtung, Dichte und Amplitude der Riefen einer Schar – sind anschaulich interpretierbar, erzeugen realistische synthetische Texturen und eignen sich außerdem gut zur Bewertung der Qualität der Oberflächen.

Diese Eigenschaften haben viele modellbasierte Ansätze gemein: Die konstruierten Merkmale besitzen eine eindeutige Interpretation und sind gleichzeitig hochdiskriminativ. Oft reicht bereits ein einfacher regelbasierter Klassifikator aus, um eine sehr gute Klassifikationsgüte zu erzielen. Da sich diese Klassifikatoren ebenfalls inspizieren bzw. interpretieren lassen, kann die Leistung des Gesamtsystems quantifiziert und im Betrieb manipuliert werden. Der Entwurf solcher Systeme ist somit gewissermaßen der Goldstandard des Feature Engineerings.

Allerdings hat Feature Engineering zwei entscheidende Nachteile: Erstens sind die Merkmale hochspezialisiert und damit nicht für den Einsatz in anderen Domänen geeignet. Das Riefenmodell von Beyerer [Bey96] lässt sich beispielsweise ohne erhebliche Anpassungen nicht zur Klassifikation von hölzernen Oberflächen einsetzen. Zweitens benötigt Feature Engineering ein hohes Maß an Wissen und Erfahrung, sowohl in der eigentlichen Problemdomäne, als auch in der mathematischen Modellierung solcher Phänomene. Die

Analyse des Problems ist zudem sehr zeitaufwendig und damit kostenintensiv. Besonders bei komplizierten Fragestellungen mit einer hohen Anzahl an Klassen wie etwa der Objektkategorisierung oder Personenreidentifikation skaliert Feature Engineering schlecht. In der visuellen Inspektion ist die Klassifikationsdomäne zwar meist starken Einschränkungen unterworfen, dennoch ist auch hier die Entwicklung problemangepasster Merkmale aufwendig und teuer. Aus diesem Grund wurden Verfahren entwickelt, die den Prozess des Feature Engineering – zumindest in Teilen – automatisieren.

2.2.4 Feature Learning

Wie beim Feature Engineering sollen beim Feature Learning an die Problemdomäne angepasste Merkmale definiert werden. Anders als beim Feature Engineering werden diese Merkmale hier jedoch automatisch aus einer Lernstichprobe abgeleitet. Statt der eigentlichen Merkmalsextraktion wird also ein Verfahren zur Ableitung der Merkmalsextraktion aufgestellt.
Teilweise können dafür auch Methoden der Merkmalsselektion angewendet werden. Dazu werden generische, meist einfache Merkmale definiert und die relevanten Kandidaten einer zufällig generierten Menge dieser Merkmale zu komplexeren Deskriptoren zusammengesetzt. Der Objektdetektor von Viola und Jones [VJ01] ist ein Beispiel dieses Ansatzes. Hier werden einfache Merkmale, die Helligkeitsunterschiede zwischen benachbarten Bildregionen repräsentieren, mittels Boosting kaskadiert, um einen hochdiskriminativen Objektdetektor zu erzeugen. Dollár et al. [Dol+09] verwenden neben Helligkeitsunterschieden weitere primitive Merkmale, die sich aus Bildregionen extrahieren lassen. Beide Ansätze können als eingebettete Verfahren der Merkmalsselektion betrachtet werden. Es gibt auch Verfahren, die einen Wrapperansatz nutzen. Im Ansatz von Burla et al. [Bur+12] werden Bildverarbeitungsketten mit Hilfe eines genetischen Algorithmus (siehe Abschnitt 2.3.2) zusammengesetzt. Die Güte einer Bildverarbeitungskette wird dabei über die Leistung bei der Detektion von Oberflächenfehlern wie Kratzern und

Schmutz bewertet. Eine ähnliche Methode wird von Lillywhite et al. [Lil+13] verwendet, um Merkmalsextraktionsverfahren für die Objektdetektion zu lernen. Wie in der Arbeit von Zimmerer [Zim14] gezeigt wurde, eignet sich dieses Verfahren auch für den Einsatz in der Schüttgutsortierung.
Es gibt jedoch auch Ansätze, um geeignete Merkmale ohne die vorherige Definition primitiver Merkmale direkt aus den Mustern abzuleiten. Solche Verfahren stützen sich meist auf Methoden der Dimensionsreduzierung. Ein früher Ansatz sind die Eigenfaces von Sirovich und Kirby [SK87]. Hier wird der Musterraum mittels Hauptkomponentenanalyse (principal component analysis, PCA) auf einen D-dimensionalen, linearen Unterraum projiziert, der die Muster $\boldsymbol{p}_1, \ldots, \boldsymbol{p}_N$ der Lernstichprobe im Sinne der kleinsten Quadrate optimal repräsentiert,

$$\boldsymbol{p} \approx \overline{\boldsymbol{p}} + \sum_{d=1}^{D} \alpha_d \boldsymbol{b}_d = \overline{\boldsymbol{p}} + \boldsymbol{B}\boldsymbol{x} \quad \text{mit} \quad \overline{\boldsymbol{p}} = \frac{1}{N} \sum_{n=1}^{N} \boldsymbol{p}_n. \tag{2.13}$$

Der Merkmalsvektor $\boldsymbol{x} = (\alpha_1, \alpha_2, \ldots, \alpha_D)^\top$ entspricht der Repräsentation von $\boldsymbol{p} - \overline{\boldsymbol{p}}$ in Bezug auf die PCA-Basisvektoren $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_D$. Neben Bildmustern wie bei Sirovich und Kirby [SK87] ist dieser Ansatz auch für andere Arten von Mustern geeignet. Blanz und Vetter [BV03] verwenden beispielsweise Laserscandaten, um Personen wiederzuerkennen. Die Active Shape Models von Cootes et al. [Coo+95], mit denen sich beliebige Strukturen in Bildern lokalisieren und klassifizieren lassen, basieren ebenfalls auf PCA. Mit Fisherfaces [BHK97] wird PCA durch eine multiple Diskriminanzanalyse ersetzt. Anders als bei PCA wird hier also Klasseninformation in die Basiswechselmatrix mit einbezogen.
Beim Sparse Dictionary Learning [OF97] werden Muster auch durch eine Linearkombination repräsentiert,

$$\boldsymbol{p} \approx \sum_{d=1}^{D'} \beta_d \boldsymbol{d}_d = \boldsymbol{D}\boldsymbol{x} \quad \text{mit} \quad \boldsymbol{x} = (\beta_1, \beta_2, \ldots, \beta_{D'})^\top. \tag{2.14}$$

Wie bei PCA sind die $\boldsymbol{d}_d$ so gewählt, dass der quadratische Rekonstruktionsfehler minimiert wird. Im Unterschied zu PCA sind die $\boldsymbol{d}_d$ hier allerdings nicht notwendigerweise orthogonal und bilden insbesondere keine Basis. Die $\boldsymbol{d}_d$ werden hier auch als Wörter eines Wörterbuches mit D' Einträgen bezeichnet. Dieses Wörterbuch $\{\boldsymbol{d}_1, \ldots, \boldsymbol{d}_{D'}\}$ ist so gewählt, dass die Muster $\boldsymbol{p}$ mit möglichst wenigen Wörtern $\boldsymbol{d}_d$ rekonstruiert werden können. Daraus folgt, dass die zugehörigen Merkmalsvektoren $\boldsymbol{x}$ dünn besetzt sind, dass also für viele der Koeffizienten $\beta_d = 0$ gilt. Das gelingt nur, wenn das Wörterbuch viele spezialisierte Wörter enthält und somit übervollständig ist. Die Dimension D' des Merkmalsraums $\mathbb{M}$ ist damit in der Regel *größer* als die Dimension des Musterraums $\mathbb{P}$. Da allerdings die Merkmale und damit auch $\mathbb{M}$ dünn besetzt sind, lassen sich unterschiedliche Klassen in diesem Merkmalsraum gut voneinander trennen – zumindest, wenn ein geeignetes Wörterbuch verwendet wird.
Im Allgemeinen kann die Suche nach einem geeigneten Wörterbuch als ein Optimierungsproblem beschrieben werden:

$$\min_{\boldsymbol{D}} \sum_{i=1}^{N} \|\boldsymbol{p}_i - \boldsymbol{D}\boldsymbol{x}_i\|_2 + \lambda \|\boldsymbol{x}_i\|_0. \tag{2.15}$$

Hierbei bezeichnet $\|\boldsymbol{x}\|_0$ die L_0-Norm von $\boldsymbol{x}$, d. h. die Anzahl der von Null verschiedenen Einträge von $\boldsymbol{x}$. Der erste Term des Optimierungsfunktionals minimiert also den Rekonstruktionsfehler im Sinne der kleinsten Quadrate und der zweite Term sorgt für einen dünn besetzten Merkmalsraum. Als Resultat kodieren die Merkmale vor allem die semantisch bedeutsamen Teile des Musters. Leider ist die Lösung von Gleichung (2.15) ein NP-hartes Problem [Til15], weswegen der Regularisierungsterm oft durch eine L_1-Regularisierung ersetzt wird. Zwar existieren für dieses modifizierte Optimierungsproblem effiziente Algorithmen (z. B. [Mai+09b; Mai+09a]), jedoch wird dadurch kein dünn besetzter Merkmalsraum erzwungen. Um dies zu erreichen, kann die Lösung mit L_1-Regularisierung als Ausgangspunkt

für eine anschließende Optimierung von Gleichung (2.15) verwendet werden. Statt die $\boldsymbol{x}$ direkt zur Klassifikation einzusetzen, können auch weitere Merkmale aus den Koeffizienten abgeleitet werden. Carrera et al. [Car+15] nutzen beispielsweise aus, dass anomale Strukturen mit deutlich mehr Wörtern rekonstruiert werden als normale Strukturen. Das Klassifikationsmerkmal ist also die Anzahl der zur Rekonstruktion verwendeten Wörter.
Sowohl PCA als auch Dictionary Learning nutzen lineare Transformationen, um Muster in Merkmale zu überführen. Dieses Vorgehen wird durch die Mannigfaltigkeitsannahme motiviert, die besagt, dass sich die hochdimensionalen Muster auf (bzw. verursacht durch Messfehler: auf eine Region um) eine niedrigdimensionale Mannigfaltigkeit in $\mathbb{P}$ beschränken [CSZ06]. Diese Mannigfaltigkeit ist aber im Allgemeinen nicht linear und ist demnach mit den oben genannten Methoden nicht abbildbar. Verfahren wie Principal Curves [HS89], Hebb'sche Netzwerke [Oja82], Autoencoder [BH89] oder Kernel PCA [SSM98] erlauben hingegen auch nichtlineare Transformationen, sind aber mit einem höheren Rechenaufwand verbunden.
In den letzten Jahren wurden vermehrt tiefe neuronale Netze (Deep Learning, siehe Schmidhuber [Sch15] und LeCun et al. [LYG15] für eine Übersicht) eingesetzt. Diese Verfahren kombinieren, ähnlich wie die eingebettete Merkmalsselektion, das Lernen von Merkmalen und Klassifikator aus einer großen Menge an Daten. Im Bereich des Maschinensehens sind vor allem die Convolutional Neural Networks (CNNs) [Fuk80; Lec+98] von Bedeutung. CNNs sind Feed-Forward Netze, d. h. Netze, die das Eingangssignal – hier das Muster – in mehreren Schichten verarbeiten. Die Neuronen einer gegebenen Schicht geben die Ergebnisse dabei nur an die (bzw. eine) folgende Schicht weiter, niemals aber an die vorherige Schicht. In CNNs sind die ersten Schichten des Netzes durch Kaskaden von Faltungsblöcken gegeben, die jeweils aus einer Faltungs- und einer Pooling-Schicht bestehen. Eine Faltungsschicht realisiert dabei die diskrete Faltung des Eingangssignals mit mehreren Faltungsmatrizen. Die Einträge der Faltungsmatrizen sind dabei nicht vorgegeben, sondern werden beim Training des Netzes festgelegt.

Die Pooling-Schichten reduzieren die Dimensionalität der Faltungsergebnisse, indem diese beispielsweise in Blöcke aufgeteilt und nur die maximale Filterantwort jedes Blocks weitergegeben wird (max-pooling). Neben Reduzierung der Datenmenge wird durch die Pooling-Schichten eine gewisse Toleranz gegenüber Translation und Rauschen der Eingangsdaten erreicht. Auf diese Faltungsblöcke folgen schließlich mehrere voll vernetzte Schichten, die die Rolle eines klassischen Feed-Forward Netzes übernehmen. Weitere Details zur Architektur von CNNs finden sich beispielsweise in Beyerer et al. [BRN17].
Der kaskadierte Aufbau von CNNs kann durch ähnliche Strukturen im primären visuellen Kortex von Säugetieren motiviert werden [LYG15]. Wie die L1-Zellen des primären visuellen Kortex spricht der erste Faltungsblock eines trainierten CNNs auf einfache Strukturen wie Kanten, Ecken und Farbverläufe an. In den folgenden Blöcken werden diese einfachen Strukturen zu immer komplexeren Strukturen zusammengesetzt, bis schließlich einzelne Neuronen des CNNs auf komplexe Objekte wie Gesichter ansprechen [ZF13].
CNNs werden vor allem wie oben beschrieben als Ende-zu-Ende-Klassifikatoren eingesetzt: Das System lernt die komplette Mustererkennung inklusive Merkmalsextraktion und Klassifikation. Es ist aber auch möglich, das Ergebnis einer Zwischenschicht abzugreifen und als Merkmalsvektor für ein anderes Klassifikationsverfahren einzusetzen [Don+13; Oqu+14]. Je nachdem, wo die Merkmale abgegriffen werden, kodieren diese einfache Strukturen bis hin zu semantisch bedeutsamen Objekten. In diesem Kontext können CNNs ebenfalls als nichtlineare Merkmalsextraktionsverfahren angesehen werden.

2.2.5 Merkmale höherer Ordnung

Eine andere Methode, die semantische Lücke zwischen einfachen und interpretierbaren Merkmalen zu schließen, ist die Generierung von Merkmalen höherer Ordnung. Bei diesen Ansätzen werden einfache Merkmale eines Objekts, etwa DCT, LBP, SIFT oder HOG (siehe Abschnitt 2.2.1), extrahiert

und in einen semantisch bedeutsamen Objektdeskriptor überführt. Meist geht dies zusätzlich mit einer Dimensionsreduzierung einher. Die bereits genannten Verfahren von Viola und Jones [VJ01] und Dollár et al. [Dol+09] können als solche Verfahren interpretiert werden. Allerdings wird hier kein Deskriptor erzeugt, der mit einem beliebigen Klassifikator verwendet werden kann.

Eine bekannte Methode zur Generierung solcher klassifikatorunabhängiger Deskriptoren ist das von Sivic [Siv06] und Csurka et al. [Csu+04] popularisierte Bag of Visual Words (BOW) Verfahren. Das Merkmal höherer Ordnung ist hierbei eine Statistik über eine Menge von einfachen, semantisch wenig bedeutsamen Deskriptoren. Da in Abschnitt 3.4 genauer auf diese Methode eingegangen wird, sollen an dieser Stelle nur die Grundlagen erörtert werden. Das Ziel von Bag of Visual Words (BOW) ist es, eine Menge von lokalen Merkmalen $\boldsymbol{x} \in \mathbb{M}$, die lokale Strukturen des Musters beschreiben, in einen Deskriptor $\boldsymbol{m}$ zu überführen, der die globale Struktur des Musters repräsentiert. Dazu wird der Merkmalsraum $\mathbb{M}$ in K disjunkte Regionen $z_1, \ldots, z_K$ mit $z_i \cap z_k = \emptyset$ für $i \neq k$ und $\bigcup_{k=1}^{K} z_k = \mathbb{M}$ unterteilt. Der Deskriptor $\boldsymbol{m}$ entspricht dann einer Statistik über die Zugehörigkeiten der lokalen Deskriptoren $\boldsymbol{x}_t$ eines Objekts zu den z_k. Meist wird hierfür ein Histogramm verwendet. Es gibt aber auch Ansätze, die eine Statistik höherer Ordnung berechnen [PD07]. Ein Vorteil dieses Verfahrens ist, dass die Anzahl sowie der Ort der lokalen Merkmale bezogen auf das Muster nicht in den Objektdeskriptor eingeht. Bei geeigneten lokalen Deskriptoren ist der Objektdeskriptor somit robust gegenüber Skalierung und Translation des zu klassifizierenden Objekts. Zudem übertragen sich die Invarianzeigenschaften der $\boldsymbol{x}_t$ auf den Objektdeskriptor. Das Verwerfen des räumlichen Zusammenhangs der lokalen Deskriptoren kann sich aber auch nachteilig auswirken, wenn der räumliche Zusammenhang zwischen den $\boldsymbol{x}_t$ diskriminative Information trägt.

Der BOW-Ansatz weist starke Parallelen mit der Grays Vektorquantisierung [Gra84] zur Datenkompression und Signalübertragung auf. Anders

als bei der Vektorquantisierung liegt der Fokus bei BOW allerdings nicht auf einer möglichst originalgetreuen Rekonstruktion, sondern auf einer diskriminativen Repräsentation der Daten: »[We] are not even interested in a "correct clustering" in the sense of feature distributions, but rather in accurate categorization.« (Csurka et al. [Csu+04]).
In der Textverarbeitung ist der Bag of Words-Ansatz auch als ein generatives probabilistisches Dokumentenmodell interpretierbar. Dieses Dokumentenmodell modelliert dabei den Entstehungsprozess eines Dokuments als eine Aneinanderreihung von zufällig gezogenen Wörtern des Vokabulars und ist somit lediglich von den individuellen Auftrittswahrscheinlichkeiten der Wörter abhängig. Solch ein Modell ist natürlich eine starke Vereinfachung. Die Latent Dirichlet Allocation (LDA) [BNJ03; RV13] ist ein detaillierteres Modell, in dem nicht nur Wörter, sondern auch latente Themengebiete modelliert werden, die wiederum die Auftrittswahrscheinlichkeiten der Wörter beeinflussen. Die Themen werden dabei nicht a-priori definiert, sondern vom Lernalgorithmus ermittelt. Der LDA-Ansatz kann wie BOW auf Bilddaten übertragen werden [RV13]. Ein gegebenes Bild kann dann über die im Bild enthaltenen Themen kategorisiert werden. Allerdings sind Lernalgorithmus und Extraktion der Themen aus Bilddaten verhältnismäßig aufwendig.
Farhadi et al. [Far+09] klassifizieren ebenfalls anhand der in den Bildern enthaltenen Themen. Anders als bei LDA werden hier die semantisch bedeutsamen Themen bzw. Attribute aber vorgegeben. Für jedes Attribut wird ein Klassifikator trainiert, der das Vorhandensein des Attributs im Bild unabhängig von der Objektklasse vorhersagt. Die eigentliche Objektklasse wird dann mit Hilfe dieses Attributdeskriptors vorgenommen. Die Extraktion des Attributdeskriptors ist dabei, je nach verwendeten Klassifikatoren, sehr schnell. Allerdings ist der Ansatz unflexibel, da die Attribute vorgegeben werden müssen, und er benötigt zusätzliche Annotation der Daten.
Zudem ist nicht klar, ob die explizite Kodierung der Attribute überhaupt nötig ist, da die Attribute bereits implizit in den Klassen kodiert sind. Dies kann ausgenutzt werden, indem K binäre Klassifikationshypothesen

$h_1(\boldsymbol{x}), \ldots, h_K(\boldsymbol{x})$ – sogenannte Classemes [TSF10] – für $K \leq C$ der C Klassen im Datensatz trainiert werden. Der Klassifikator $h_k(\boldsymbol{x})$ trennt dabei die Klasse ω_k von allen anderen Klassen $\overline{\omega}_k$. Da das Ziel der $h_k(\boldsymbol{x})$ nicht die fehlerfreie Klassifikation, sondern eine Kodierung der im Bild enthaltenen Konzepte ist, werden hierfür in der Regel einfache, schnell auswertbare lineare Klassifikatoren verwendet. Der zur eigentlichen Klassifikation verwendete Deskriptor ergibt sich aus den Vorhersagen der $h_k(\boldsymbol{x})$,

$$\boldsymbol{f} = (h_1(\boldsymbol{x}), \ldots, h_K(\boldsymbol{x}))^\top \tag{2.16}$$

mit $h_k(\boldsymbol{x}) \in \{0, 1\}$ oder $h_k(\boldsymbol{x}) \in \mathbb{R}$ für $k = 1, \ldots K$. Der Classeme-Deskriptor $\boldsymbol{f}$ erlaubt dabei nicht nur die Klassifikation in eine der K bekannten Klassen, sondern auch die Kategorisierung in neue Klassen.
Dieser Ansatz hat jedoch noch immer zwei entscheidende Nachteile: Erstens ist die Wahl der K Basisklassen noch immer dem Nutzer überlassen, jedoch sind keinerlei Kriterien bekannt, die diese Wahl erleichtern. Zweitens werden die einzelnen Classemes unabhängig von dem eigentlichen Ziel, also der Objektkategorisierung, trainiert. Unsuprevised Classemes [CSS12] lösen das erste Problem durch die Verwendung von Clusteranalyse, die implizite Klassen definiert und den Nutzer so entlastet. Dieser Ansatz erzielt bessere Ergebnisse als die überwachten Classemes [CSS12], jedoch fehlt eine theoretische Motivation für dieses Vorgehen. Zudem bleibt der zweite Nachteil – getrennte Optimierung von Classeme und Klassifikator – bestehen.
Dieses Problem wird durch den Ansatz von Bergamo et al. [BTF11] angegangen. Wie bei Cusano et al. [CSS12] wird der Classeme-Ansatz hier um eine automatische Entdeckung der latenten Klassen erweitert. Diese werden allerdings nicht mit einer vorgeschalteten Clusteranalyse gefunden, sondern durch die Konstruktion einer komplizierten Verlustfunktion (siehe Abschnitt 2.4.3) modelliert, die das Training des Klassifikators und die Suche nach den Classemes kombiniert. Der Benutzer muss lediglich die Anzahl der gewünschten Classemes vorgeben, nicht aber die eigentlichen Classemes.

2.3 Optimierung

Da sich die Lernalgorithmen vieler Klassifikationsverfahren als Optimierungsproblem fassen lassen (siehe Abschnitt 2.4.3), soll hier kurz auf zwei im maschinellen Lernen häufig verwendete Optimierungsverfahren eingegangen werden: Gradientenabstieg und genetische Algorithmen. In der Optimierungstheorie sind eine große Anzahl an leistungsfähigeren Algorithmen bekannt, welche die Struktur des Optimierungsproblems ausnutzen. Details zu solchen Verfahren finden sich beispielsweise in den Büchern von Nocedal und Wright [NW06] und Boyd und Vandenberghe [BV09].
Unabhängig von der Wahl des Optimierungsverfahrens ist das Ziel hier stets die Minimierung einer Zielfunktion $\mathfrak{J}(\boldsymbol{\theta})$ (Maximierung von $\mathfrak{J}(\boldsymbol{\theta})$ entspricht Minimierung von $-\mathfrak{J}(\boldsymbol{\theta})$) durch die Wahl geeigneter Parameter $\boldsymbol{\theta} \in \Theta$,

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \mathfrak{J}(\boldsymbol{\theta})\,. \tag{2.17}$$

Außer bei konvexen Funktionen gibt es in der Regel nicht nur ein globales Minimum[3], sondern mehrere lokale Minima. Lokale Minima sind Funktionswerte $\mathfrak{J}(\tilde{\boldsymbol{\theta}})$, für die $\mathfrak{J}(\tilde{\boldsymbol{\theta}}) < \mathfrak{J}(\boldsymbol{\eta})$ für alle Parameter $\boldsymbol{\eta}$ in einer Region um $\tilde{\boldsymbol{\theta}}$ gilt. Anders als bei anderen Optimierungsproblemen ist es beim maschinellen Lernen aber nicht wichtig, das globale Optimum zu treffen. Vielmehr reicht es, eine Lösung $\tilde{\boldsymbol{\theta}}$ zu finden, die das Problem „gut genug" löst. Im Gegenteil kann eine global optimale Lösung sogar unerwünscht sein, da diese Overfitting an die Lernstichprobe nach sich ziehen kann (siehe Abschnitt 2.4).
Im Folgenden wird vereinfachend von einer skalaren Zielfunktion $\mathfrak{J} : \Theta \to \mathbb{R}$ ohne Beschränkung der zulässigen $\boldsymbol{\theta} \in \Theta$ ausgegangen. Verfahren zur Optimierung mit Nebenbedingungen, wie sie bereits in Abschnitt 1.4 aufgetreten sind und beispielsweise auch bei der Support Vektor Maschine eine Rolle spielen, werden im Rahmen dieser Arbeit nicht tiefergehend behandelt. Ebenso wird an dieser Stelle die multikriterielle Optimierung, d. h. die gleichzeitige

[3] Es kann auch mehrere globale Minima geben, z. B. $\arg\min_x \sin(x)$ mit $x \in \mathbb{R}$.

Optimierung mehrerer gleichwertiger Ziele, an dieser Stelle ebenfalls nicht behandelt. Dieses Thema wird allerdings in Abschnitt 3.2 erneut aufgegriffen.

2.3.1 Gradientenabstieg

Das Gradientenabstiegsverfahren ist ein einfaches numerisches Verfahren zur Lösung allgemeiner Optimierungsprobleme ohne Nebenbedingungen. Ausgehend von einer initialen Schätzung $\boldsymbol{\theta}_0$ wird ein Minimum von $\mathfrak{J}$ iterativ durch

$$\boldsymbol{\theta}_t = \boldsymbol{\theta}_{t-1} + \alpha_t \, \Delta\boldsymbol{\theta}_t \tag{2.18}$$

mit $\alpha_t > 0$ für $t > 0$ angenähert. Es kann gezeigt werden, dass das Verfahren konvergiert, wenn $\mathfrak{J}(\boldsymbol{\theta}_t) < \mathfrak{J}(\boldsymbol{\theta}_{t-1})$ für alle $t > 0$ (ausgenommen optimale Parameter $\boldsymbol{\theta}^*$) gilt [BV09]. Eine natürliche Wahl der Richtung $\Delta\boldsymbol{\theta}_t$ zur Erfüllung dieser Bedingungen ist der negative Gradient der Zielfunktion,

$$\Delta\boldsymbol{\theta}_t := -\nabla_{\boldsymbol{\theta}}\mathfrak{J}(\boldsymbol{\theta}_{t-1}), \tag{2.19}$$

da dieser in die Richtung des steilsten Abstiegs von $\mathfrak{J}$ zeigt. Die Konvergenz zum globalen Minimum ist dabei nur bei konvexen Funktionen garantiert [BV09]. Bei nichtkonvexen Funktionen kann, je nach Wahl des Startpunkts $\boldsymbol{\theta}_0$, ein unterschiedliches lokales Optimum erreicht werden.
Das Gradientenabstiegsverfahren kann überall dort eingesetzt werden, wo der Gradient der Zielfunktion $\nabla_{\boldsymbol{\theta}}\mathfrak{J}$ berechnet werden kann. Wenn der Gradient nicht analytisch bestimmt werden kann, kann dieser numerisch mit Hilfe des zentralisierten Differenzenquotienten

$$\frac{\partial\mathfrak{J}(\boldsymbol{\theta})}{\partial\theta_i} \approx \frac{\mathfrak{J}(\boldsymbol{\theta} + h\,\boldsymbol{e}_i) - \mathfrak{J}(\boldsymbol{\theta} - h\,\boldsymbol{e}_i)}{2h} \tag{2.20}$$

mit Schrittweite h approximiert werden. Dieses Verfahren ist aber bei kleinen Schrittweiten numerisch instabil und bei zu großen Schrittweiten inexakt.

Zudem muss die Zielfunktion $\mathfrak{J}$ bei der Verwendung eines P-dimensionalen Parametervektors $2P$ mal ausgewertet werden. Als Alternative bieten sich Methoden der automatischen Differenzierung an, siehe z. B. Baydin et al. [Bay+15]. Mit automatischer Differenzierung kann der Gradient $\nabla_{\boldsymbol{\theta}}\mathfrak{J}(\boldsymbol{\theta})$ an $\boldsymbol{\theta}$ mit geringem zusätzlichen Rechenaufwand exakt berechnet werden. Dies funktioniert sogar dann, wenn $\mathfrak{J}$ Diskontinuitäten aufweist oder der Funktionswert von Zufallsereignissen abhängt.

Neben der Berechnung des Gradienten ist die Wahl der Schrittweite α_t von Bedeutung. Hierfür gibt es mehrere Ansätze. Im einfachsten Fall wird eine konstante Lernrate $\alpha_t = \eta > 0$ gewählt. Die Wahl einer passenden Lernrate ist dabei nicht einfach: Ist η zu klein, konvergiert der Algorithmus langsam. Ist η hingegen zu groß, kann eine Konvergenz nicht mehr garantiert werden, da $\mathfrak{J}(\boldsymbol{\theta}_t) < \mathfrak{J}(\boldsymbol{\theta}_{t-1})$ nicht mehr sichergestellt ist. Dieses Problem ist in Abbildung 2.3 illustriert. Hier ist die Zielfunktion $\mathfrak{J}(\boldsymbol{\theta}) = (\boldsymbol{\theta} - \boldsymbol{p})^\top \boldsymbol{A}\,(\boldsymbol{\theta} - \boldsymbol{p})$ mit $\boldsymbol{\theta}, \boldsymbol{p} \in \mathbb{R}^2$ und positiv definiter Matrix $\boldsymbol{A} \in \mathbb{R}^{2\times 2}$ strikt konvex mit dem Minimum bei $\boldsymbol{p}$. Mit $\eta = 0{,}7$ wird das Optimum innerhalb weniger Iterationen gefunden. Mit $\eta = 0{,}2$ konvergiert der Algorithmus ebenfalls, benötigt aber deutlich mehr Iterationen. Für $\eta = 1{,}0$ konvergiert der Algorithmus nicht, sondern oszilliert um das globale Minimum. Die Konvergenzgeschwindigkeit ist allerdings auch von der Wahl des Startpunkts $\boldsymbol{\theta}_0$ abhängig.

Abbildung 2.3 zeigt ein weiteres Verhalten des Gradientenabstiegsverfahrens: Je weiter $\boldsymbol{\theta}_t$ vom Optimum entfernt ist, desto größer ist der Betrag des Gradienten und damit die Änderung der Parameterschätzung. Dieses Verhalten kann durch den Einsatz einer abklingenden Lernrate (engl.: learning rate decay) verstärkt werden. In frühen Iterationen führt eine große Lernrate zu schneller Annäherung an das Optimum, während in der späteren Iterationen kleinere Lernraten eine Oszillation um das Optimum vermeiden. Typische Implementierungen einer abklingenden Lernrate sind der exponentielle Verfall $\alpha_t = \eta \cdot \gamma^t$ mit $\gamma \in (0, 1)$ (rote Linie in Abbildung 2.3) sowie der hyperbolische Verfall $\alpha_t = \frac{\eta}{(1+\gamma\, t)}$. Eine weitere Alternative ist die Liniensuche, bei

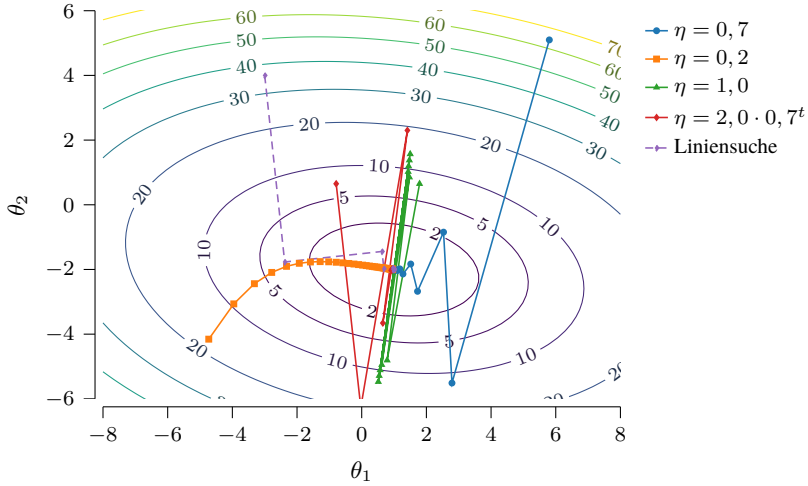


Abbildung 2.3: Optimierung der Zielfunktion $\mathfrak{J}(\boldsymbol{\theta}) = (\boldsymbol{\theta} - \boldsymbol{p})^\top \boldsymbol{A} (\boldsymbol{\theta} - \boldsymbol{p})$ mit $\boldsymbol{p} = (1, -2)^\top$ durch Gradientenabstieg mit fester und variabler Lernrate sowie durch Liniensuche.

der α_t in jedem Iterationsschritt anhand einer Minimumsuche entlang der Richtung des Gradienten festgelegt wird,

$$\alpha_t = \arg\min_{\alpha} \mathfrak{J}(\boldsymbol{\theta} - \alpha \nabla_{\boldsymbol{\theta}} \mathfrak{J}(\boldsymbol{\theta}_{t-1})) \,. \tag{2.21}$$

Dieses Verfahren ist mit zusätzlichem Rechenaufwand verbunden, beschleunigt aber die Konvergenz des Gradientenabstiegs deutlich. Im Beispiel in Abbildung 2.3 wird das Optimum mit der Liniensuche bereits nach fünf Iterationen erreicht.

Für Zielfunktionen, die sich additiv aus mehreren Teilzielen $\mathfrak{J}_q$ zusammensetzen, also $\mathfrak{J}(\boldsymbol{\theta}) = \sum_q \mathfrak{J}_q(\boldsymbol{\theta})$, kann zudem in jedem Iterationsschritt nur eine zufällige Teilmenge der $\mathfrak{J}_q$ minimiert werden. Dieser stochastische Gradientenabstieg findet insbesondere beim Training von tiefen neuronalen Netzen Anwendung [Bot10]. Einerseits wird dadurch die benötigte Rechenzeit

trotz geringerer Konvergenzgeschwindigkeit erheblich reduziert. Andererseits generalisieren die gelernten Klassifikatoren besser, da breite lokale Minima gegenüber schmalen Minima bevorzugt werden und es die Randomisierung ermöglicht, aus flachen lokalen Minima und von Sattelpunkten zu entkommen. Weitere Erweiterungen des Gradientenabstiegsverfahrens sind die Momentum-Methode [RHW86], bei der vergangene Iterationsschritte berücksichtigt werden, sowie Verfahren, bei der jede Dimension des Parameterraums mit einer eigenen adaptiven Lernrate gewichtet wird [DHS11; Zei12; KB14].
Bei nichtkonvexen Optimierungsproblemen hängt die Konvergenz zu einem lokalen oder globalen Optimum mit lokalen Optimierungsverfahren wie dem Gradientenabstieg von der Wahl der initialen Parameterschätzung $\boldsymbol{\theta}_0$ ab. Zudem können solche Verfahren auch auf einem Sattelpunkt der Lösungsfläche stecken bleiben. Um dennoch ein globales Optimum zu finden, kann die Optimierung mit mehreren unterschiedlichen Startpunkten wiederholt und die Lösungskandidaten miteinander verglichen werden.

2.3.2 Genetische Algorithmen

Die globale Optimierung mit genetischen Algorithmen (GAs) funktioniert nach einem ähnlichen Prinzip: Statt einer Lösung werden viele Lösungskandidaten vorgehalten und in mehreren Runden optimiert. Bei einer genügend großen Menge an Kandidaten ist die Wahrscheinlichkeit groß, dass so auch bei nichtkonvexen Problemen ein globales Optimum gefunden wird.
GAs orientieren sich an der Evolutionstheorie. Jeder Lösungskandidat $\boldsymbol{\theta}_n \in \Theta, i = 1, \ldots, N$ wird als Individuum bezeichnet. Die Gesamtheit aller Individuen einer Optimierungsrunde $\mathcal{P}_t$ heißt Population. In der $(t+1)$-ten Runde werden durch Rekombination und Mutation der Individuen der vorherigen Population M neue Lösungskandidaten erzeugt und der Population hinzugefügt. Die Lösungskandidaten werden anhand der Zielfunktion $\mathfrak{J}(\boldsymbol{\theta})$ bewertet und die schlechtesten M Individuen werden verworfen. In diesem Kontext heißt $\mathfrak{J}$ auch die Fitness-Funktion und jede Runde wird auch als Ge-

Algorithmus 1 Genetischer Algorithmus.

function GA($\mathfrak{J}, N, M, t_{\max}$)
  $\mathcal{P}_0 \leftarrow \{\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_N\}$ zufällig
  for $t = 1, \ldots, t_{\max}$ **do**
    $\mathcal{K} \leftarrow \mathcal{P}_{t-1}$
    for $m = 1, \ldots, M$ **do**
      $(\boldsymbol{\theta}_i, \boldsymbol{\theta}_k) \leftarrow$ SELECTPARENTS$(\mathcal{P}_{t-1})$
      $\boldsymbol{\theta}'_c \leftarrow$ RECOMBINE$(\boldsymbol{\theta}_i, \boldsymbol{\theta}_k)$
      $\boldsymbol{\theta}_c \leftarrow$ MUTATE$(\boldsymbol{\theta}'_c)$
      $\mathcal{K} \leftarrow \mathcal{K} \cup \{\boldsymbol{\theta}_c\}$
    $\mathcal{P}_t \leftarrow \{\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_N\} \subset \mathcal{K}$ mit $\mathfrak{J}(\boldsymbol{\theta}_1) \leq \mathfrak{J}(\boldsymbol{\theta}_2) \leq \ldots \leq \mathfrak{J}(\boldsymbol{\theta}_N)$.
    if $|\mathcal{P}_t \cap \mathcal{P}_{t-1}| = N$ **then break**
  return $\mathcal{P}_t$

neration bezeichnet. Dieses Vorgehen wird wiederholt, bis eine vorgegebene Menge an Iterationen erreicht ist oder bis sich die Population nicht mehr ändert. Der Algorithmus ist in Algorithmus 1 aufgeführt.

Die wesentlichen Freiheitsgrade dieses Algorithmus sind die drei Methoden SELECTPARENTS, RECOMBINE und MUTATE zur Erzeugung neuer Individuen. Die Methode SELECTPARENTS$(\mathcal{P})$ wählt dazu zwei Eltern-Kandidaten $\boldsymbol{\theta}_i$ und $\boldsymbol{\theta}_k$ aus der Population $\mathcal{P}$ aus. Hierfür gibt es mehrere Ansätze. Ein naiver Ansatz ist die zufällige Wahl von $\boldsymbol{\theta}_i$ und $\boldsymbol{\theta}_k$. Dadurch wird aber vorhandenes Wissen über den Parameterraum nicht ausgenutzt und eine Konvergenz ist unwahrscheinlich. Alternativ könnten die beiden fittesten Individuen, also die $\boldsymbol{\theta}_i, \boldsymbol{\theta}_k \in \mathcal{P}_t$ mit kleinstem $\mathfrak{J}(\boldsymbol{\theta})$, verwendet werden. Dadurch würden allerdings alle neuen Lösungskandidaten von den gleichen Eltern erzeugt und der Parameterraum nur sehr langsam exploriert.

Eine gängige Alternative, die zwischen diesen Extremen liegt, ist die Tournament Selection. Hier werden beide Eltern nacheinander als die fittesten Indi-

viduen zweier zufälliger Untermengen $\mathcal{P}'_i, \mathcal{P}'_k \subset \mathcal{P}$ mit $|\mathcal{P}'_i| = |\mathcal{P}'_k| \ll |\mathcal{P}|$ festgelegt, d. h.

$$\boldsymbol{\theta}_i = \underset{\boldsymbol{\theta} \in \mathcal{P}'_i}{\arg\min}\, \mathfrak{J}(\boldsymbol{\theta}) \quad \text{und} \quad \boldsymbol{\theta}_k = \underset{\boldsymbol{\theta} \in \mathcal{P}'_k \setminus \{\boldsymbol{\theta}_i\}}{\arg\min}\, \mathfrak{J}(\boldsymbol{\theta})\,. \tag{2.22}$$

Durch die Größe der Untermengen kann zwischen Exploration des Lösungsraums und Exploitation vorhandenen Wissens abgewägt werden.
Die Generierung neuer Individuen wird in der Methode RECOMBINE implementiert. Je nach Problemstellung kann hier Vorwissen über die Struktur des Lösungsraums eingebracht werden. Ein gängiger Ansatz zur Kombination der $\boldsymbol{\theta}_i, \boldsymbol{\theta}_k$ ist, jeden Parameter $[\boldsymbol{\theta}_c]_d$ zufällig von einem der Eltern zu übernehmen,

$$\left[\boldsymbol{\theta}'_c\right]_d = \textsc{Random}\left(\left\{[\boldsymbol{\theta}_i]_d\,, [\boldsymbol{\theta}_k]_d\right\}\right) \quad d = 1, \ldots, \dim(\Theta), \tag{2.23}$$

wobei die Methode $\textsc{Random}(\mathcal{M})$ zufällig und mit gleicher Wahrscheinlichkeit ein Element der gegebenen Menge $\mathcal{M}$ auswählt.
Da der Parameterraum Θ durch Rekombination der bestehenden Lösungen nicht vollständig abgetastet werden kann, wird ein durch Rekombination gewonnenes Individuum zusätzlich durch die Methode MUTATE verändert. Gängige Methoden sind das Hinzufügen kleiner zufälliger Störungen $\boldsymbol{e}$,

$$\boldsymbol{\theta}_c = \boldsymbol{\theta}'_c + \boldsymbol{e} \quad \text{mit} \quad \|\boldsymbol{e}\| < \varepsilon, \tag{2.24}$$

oder eine zufällige Veränderung einzelner Parameter. Ein übliches Vorgehen hierfür ist es, jeden Parameter $[\boldsymbol{\theta}_c]_d$ mit einer vorgegebenen Wahrscheinlichkeit ρ neu festzulegen oder mit Wahrscheinlichkeit $(1 - \rho)$ zu übernehmen, also

$$[\boldsymbol{\theta}_c]_d = \begin{cases} \textsc{Random}\left([\Theta]_d\right) & \text{mit Wahrscheinlichkeit } \rho \\ \left[\boldsymbol{\theta}'_c\right]_d & \text{mit Wahrscheinlichkeit } (1-\rho). \end{cases} \tag{2.25}$$

Hierbei bezeichnet $[\Theta]_d$ die Menge der zulässigen Parameter in der d-ten Dimension. Wie bei der RECOMBINE-Methode, kann in der MUTATE-Methode Vorwissen über das Optimierungsproblem eingebracht werden. Zusätzlich können hier Nebenbedingungen geprüft und gegebenenfalls erzwungen werden.
Die Methoden RECOMBINE und MUTATE übernehmen gewissermaßen entgegengesetzte Rollen: RECOMBINE nutzt vorhandenes Wissen über gute Lösungen aus, um neue Kandidaten in die Nähe der Minima zu platzieren (Exploitation); MUTATE randomisiert die Lösungen (Exploration) und erlaubt es so, aus lokalen Minima zu entkommen.
Genetische Algorithmen sind äußerst flexibel und können für die verschiedensten Aufgaben eingesetzt werden. Neben der skalaren Optimierung erlaubt die Wahl einer geeigneten Fitness-Funktion auch die multikriterielle Optimierung (siehe Deb et al. [Deb+02] und Abschnitt 3.2). Nebenbedingungen können einfach integriert und erzwungen werden. Auch erlauben genetische Algorithmen die Optimierung in unscharf definierten und unendlichdimenisonalen Parameterräumen. Beispielsweise können die Individuen die Struktur neuronaler Netze [Xin99], Algorithmen [Bur+12; Lil+13] oder sogar physikalische Systeme [Hor+06] repräsentieren.
Genetische Algorithmen haben gegenüber anderen Optimierungsverfahren jedoch auch Nachteile. Der größte Nachteil ist der verhältnismäßig hohe Rechenaufwand: In jeder Iteration muss die Zielfunktion für jedes neue Individuum ausgewertet werden. Bei einem hochdimensionalen Suchraum muss zudem eine große Population verwendet werden, um diesen genügend zu explorieren. Die Wahl eines Abbruchkriteriums ist schwierig und es gibt keine Garantie der Konvergenz zu einem globalen Minimum. Vielmehr werden in der Regel lediglich Lösungen in der Nähe der Minima gefunden. Die Konvergenzeigenschaften werden maßgeblich von den RECOMBINE- und MUTATE-Methoden beeinflusst, jedoch gibt es hierfür lediglich Heuristiken und keine allgemeingültigen Implementierungsrichtlinien.

2.4 Klassifikation

Die probabilistische Formulierung und die Annahme der i. i. d. Stichprobe legen eine ebenfalls probabilistische Behandlung der Klassifikation nahe. In Abschnitt 2.1 wurde bereits vorgegriffen, dass die Klassifikation anhand der Verbunddichte $p(\boldsymbol{x}, \omega)$ durchgeführt werden kann. Dies ist äquivalent zur Klassifikation anhand der maximalen a-posteriori Klassenwahrscheinlichkeit $P(\omega \mid \boldsymbol{x})$:

$$\hat{\omega}(o) = \arg\max_{\omega} p(\boldsymbol{x}, \omega) \qquad \text{mit } \boldsymbol{x} = \psi(\boldsymbol{p}(o)) \tag{2.26}$$

$$= \arg\max_{\omega} P(\omega)\, p(\boldsymbol{x} \mid \omega) \tag{2.27}$$

$$= \arg\max_{\omega} \frac{P(\omega)\, p(\boldsymbol{x} \mid \omega)}{p(\boldsymbol{x})} \qquad (\text{da } p(\boldsymbol{x}) \text{ unabhängig von } \omega) \tag{2.28}$$

$$= \arg\max_{\omega} P(\omega \mid \boldsymbol{x}) \qquad (\text{Satz von Bayes}). \tag{2.29}$$

Da er anhand der maximalen a-posteriori Wahrscheinlichkeit entscheidet, wird dieser Klassifikator als der maximum a-posteriori (MAP) Klassifikator bezeichnet. In diesem Kontext heißt $P(\omega)$ auch die a-priori (Klassen-)Wahrscheinlichkeit, $p(\boldsymbol{x} \mid \omega)$ die klassenbedingte Merkmalsdichte bzw. Likelihood[4] und $P(\omega \mid \boldsymbol{x})$ die a-posteriori (Klassen-)Wahrscheinlichkeit. In der folgenden Diskussion wird statt $\hat{\omega}(o)$ mit $\boldsymbol{x} = \psi(\boldsymbol{p}(o))$ vereinfachend $\hat{\omega}(\boldsymbol{x})$ geschrieben. Es kann gezeigt werden, dass dieser Klassifikator bei Kenntnis der Wahrscheinlichkeitsverteilungen die Fehlerwahrscheinlichkeit

$$P_e = \mathbb{E}_{\boldsymbol{x},\omega}\left\{\mathbb{1}[\hat{\omega}(\boldsymbol{x}) \neq \omega]\right\} = \sum_{\omega} \int_{\mathbb{M}} p(\boldsymbol{x},\omega) \mathbb{1}[\hat{\omega}(\boldsymbol{x}) \neq \omega]\, \mathrm{d}\boldsymbol{x} \tag{2.30}$$

minimiert. Ein Beweis findet sich beispielsweise in Beyerer et al. [BRN17]. Wie schon eingangs erwähnt, ist allerdings auch mit diesem Klassifikator und bei voller Kenntnis der beteiligten Verteilungen eine fehlerfreie Klassifikation

[4] Nicht zu verwechseln mit der *Likelihood-Funktion.* $\mathfrak{L}(\boldsymbol{\theta} \mid \mathcal{D})$

im Allgemeinen unmöglich, d. h. im Allgemeinen ist die Fehlerwahrscheinlichkeit $P_e > 0$. Dieser irreduzierbare Fehler entsteht in den Regionen des Merkmalsraums, in denen $P(\omega \mid \boldsymbol{x}) > 0$ für mehr als eine Klasse ω.
In der Praxis ist die Verbunddichte $p(\boldsymbol{x}, \omega)$ jedoch nicht bekannt und muss stattdessen anhand einer Lernstichprobe geschätzt werden. Ein naheliegender Ansatz ist die Verwendung von parametrischen Dichten $p(\boldsymbol{x} \mid \omega, \boldsymbol{\theta})$, deren Parameter $\boldsymbol{\theta}$ beispielsweise mittels Maximum-Likelihood-Schätzung

$$\arg\max_{\boldsymbol{\theta}} \mathfrak{L}\left(\boldsymbol{\theta} \mid \mathcal{D}\right) = \arg\max_{\boldsymbol{\theta}} \prod_{\boldsymbol{x} \in \mathcal{D}} p(\boldsymbol{x} \mid \omega, \boldsymbol{\theta}) \tag{2.31}$$

festgelegt werden. Unabhängig von der Wahl der Dichteschätzung entsteht hierdurch ein zusätzlicher reduzierbarer Klassifikationsfehler. Die noch benötigten a-priori Wahrscheinlichkeiten $P(\omega)$ werden nicht aus der Stichprobe geschätzt, sondern vom Benutzer festgelegt. Sie kodieren also Vorwissen über das Klassifikationsproblem und können zur Beeinflussung des Klassifikationsverhaltens genutzt werden.
Neben den a-priori Wahrscheinlichkeiten bietet die Bayes'sche Entscheidungstheorie aber noch ein weiteres, besser interpretierbares Werkzeug zur Umsetzung diese Ziele: Risikominimierung. Die Risikominimierung erlaubt es, Fehlklassifikationen verschiedener Klassen unterschiedlich zu bewerten. Zum Beispiel ist bei der Sortierung von Edelsteinen das Ausschleusen eines Diamanten schwerwiegender als eine Fehldetektion eines wertlosen Steins. Letzteres verursacht zusätzliche Arbeit oder einen Imageverlust, während ersteres große finanzielle Einbußen nach sich zieht.
Diese Kosten werden mit einer Kostenfunktion $l\left(\omega_i, \omega_k\right)$ kodiert. Hierbei sind $l\left(\omega_i, \omega_k\right)$ die verursachten Kosten der Zuweisung $\hat{\omega}(\boldsymbol{x}) = \omega_i$ bei vorliegender wahrer Klasse ω_k. Kosten sind hier jedoch nicht nur monetär zu verstehen, sondern können beliebige (messbare) Bewertungen darstellen.

Bei gegebener Kostenfunktion $l(\omega_i, \omega_k)$ werden die Klassifikationsparameter nun nicht mehr zur Minimierung der Fehlerwahrscheinlichkeit P_e, sondern zur Minimierung des Risikos, d. h. des erwarteten Verlusts

$$\mathfrak{R}(\hat{\omega}) = \mathbb{E}_{\boldsymbol{x},\omega}\left\{l\left(\hat{\omega}(\boldsymbol{x}), \omega\right)\right\} = \sum_{\omega} \int_{\mathbb{M}} p(\boldsymbol{x}, \omega)\, l\left(\hat{\omega}(\boldsymbol{x}), \omega\right)\, \mathrm{d}\boldsymbol{x} \tag{2.32}$$

festgelegt. Hier zeigt sich, dass die Risikominimierung eine Verallgemeinerung der Fehlerminimierung darstellt: Gleichung (2.30) entspricht Risikominimierung mit der Kostenfunktion $l(\omega_i, \omega_k) = \mathbb{1}[\omega_i \neq \omega_k]$. Diese spezielle Kostenfunktion wird auch 0-1-loss genannt, da angenommen wird, dass bei einer korrekten Klassifikation keine, und bei einer Fehlklassifikation die Kosten $l(\omega_i, \omega_k) = 1$ entstehen. Ein nach Risikominimierung klassifizierender Klassifikator, bei dem alle beteiligten Wahrscheinlichkeitsverteilungen bekannt sind, wird auch der Bayes'sche Optimalklassifikator genannt. Der MAP-Klassifikator ist der Bayes'sche Optimalklassifikator für den 0-1-loss. Egal, ob parametrische Dichten oder nichtparametrische Verfahren wie die Parzen-Fenstermethode [Par62] verwendet werden, und egal ob der Klassifikator die Fehlerwahrscheinlichkeit oder das Risiko minimiert, entsteht neben dem irreduzierbaren Klassifikationsfehler ein zusätzlicher Modellfehler. Es kann gezeigt werden, dass sich dieser Fehler aus einem Bias- und einem Varianzterm zusammensetzt [HTF09]. Der Biasterm repräsentiert dabei den vom Modell induzierten systematischen Fehler. Der Varianzterm repräsentiert den stochastischen Fehler der Parameterschätzung anhand der (endlichen) Stichprobe. Bias und Varianz verhalten sich dabei gegenläufig. Einfache Modelle mit wenigen Parametern zeigen im Allgemeinen einen hohen Bias, da sie die Komplexität des Klassifikationsproblems unterschätzen. Die Varianz ist hingegen klein, da kleine Veränderungen der Lernstichprobe die einfachen Entscheidungsregionen des Klassifikators nur wenig beeinflussen. Im Gegensatz dazu zeigen komplexe Modelle mit vielen Parametern geringen Bias, aber eine hohe Varianz, da sich die Entscheidungsregionen sehr gut an

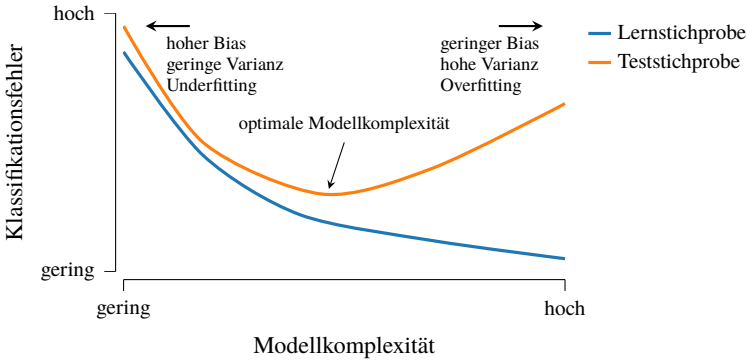


Abbildung 2.4: Qualitativer Zusammenhang von Modellkomplexität, Klassifikationsfehler, Bias und Varianz eines Klassifikators. Grafik nach Hastie et al. [HTF09].

die Lernstichprobe anpassen. Die Wahl eines geeigneten Klassifikators ist auch immer eine Abwägung zwischen Bias und Varianz des Modells.
Diese Abwägung ist auch bei der empirischen Klassifikatorbewertung (siehe Abschnitt 2.5) beobachtbar. Abbildung 2.4 skizziert den typischen Verlauf des Klassifikationsfehlers auf der zur Festlegung der Modellparameter verwendeten Lernstichprobe und auf einer bisher ungesehenen Teststichprobe in Abhängigkeit der Modellkomplexität. Zu einfache Modelle resultieren in hohen Fehlerraten mit beiden Stichproben. Hier spricht man auch von Underfitting. Zu komplexe Modelle minimieren den Fehler auf der Lernstichprobe, machen aber viele Fehler bei der Klassifikation der Teststichprobe. In diesem Fall hat der Klassifikator die Lernstichprobe gewissermaßen auswendig gelernt und man spricht von Overfitting. Die optimale Modellkomplexität liegt zwischen diesen beiden Extremen und ist vom Klassifikationsproblem abhängig.

2.4.1 Generative und Diskriminative Modelle

Klassifikatoren, die anhand von Gleichung (2.26), also unter Verwendung der Verbunddichte $p(\boldsymbol{x}, \omega)$ bzw. von $p(\boldsymbol{x} \mid \omega)$ und $P(\omega)$, klassifizieren, werden auch als generative Modelle bezeichnet. Diese Bezeichnung rührt daher, dass diese Verteilungen die Daten vollständig beschreiben und somit nicht nur zur Klassifikation von gegebenen Daten, sondern auch zur Generierung neuer Daten $(\boldsymbol{x},\omega)$ genutzt werden können.
Wie eingangs erwähnt wurde, werden für $p(\boldsymbol{x}, \omega)$ bzw. $p(\boldsymbol{x} \mid \omega)$ üblicherweise parametrische Modelle verwendet, deren Parameter mit Hilfe einer endlichen Lernstichprobe festgelegt werden. Die Parameterschätzung wird umso unsicherer, je höher die Anzahl der Parameter bzw. je geringer der Umfang der Lernstichprobe ist [BRN17].
Für einen einfachen Gauß-Klassifikator, also Klassifikation unter der Normalverteilungsannahme $p(\boldsymbol{x} \mid \omega_c) = \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)$ für $c = 1, \ldots, C$, sind beispielsweise $C \cdot \left(\frac{1}{2}D(D+1) + D\right)$ Parameter zu schätzen, wobei D die Dimensionalität von $\mathbb{M}$ bezeichnet. Die Anzahl der Parameter wächst hier also quadratisch mit der Dimensionalität des Merkmalsraums. Mit der Anzahl der Parameter wächst aber auch der für eine verlässliche Schätzung benötigte Umfang der Lernstichprobe.
Diesem Fluch der Dimensionalität kann durch Verwendung von Regularisierung, hier beispielsweise durch Shrinkage-Verfahren [SS05] oder durch andere Einschränkung des Parameterraums entgegengewirkt werden. Hierdurch wird aber in der Regel der Bias des Klassifikators, und somit der erwartete Klassifikationsfehler erhöht.
Statt der Verbunddichte $p(\boldsymbol{x}, \omega)$ kann aber auch die a-posteriori Wahrscheinlichkeit $P(\omega \mid \boldsymbol{x})$ direkt modelliert werden. Diese diskriminativen Modelle eignen sich lediglich zur Klassifikation, jedoch nicht zur Generierung neuer Stichproben. Ohne Kenntnis der klassenunabhängigen Dichte der Merkmale $p(\boldsymbol{x})$ enthält ein diskriminatives Modell also weniger Information über die Struktur des Merkmalsraums als ein generatives Modell. Im Umkehrschluss

bedeutet das aber auch, dass zur Festlegung solcher Modelle eine geringere Anzahl an Trainingsbeispielen nötig ist. In der Tat werden in der Praxis diskriminative Modelle oft gegenüber generativen Modellen bevorzugt:

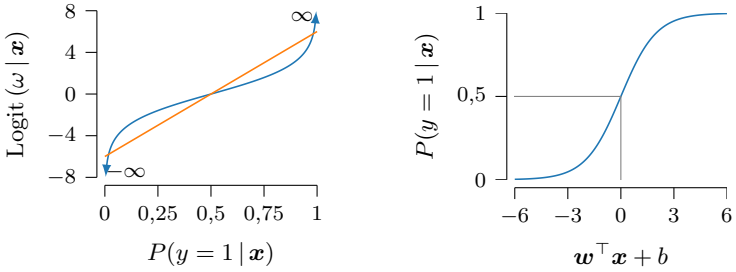
> [...] leaving aside computational issues and matters such as handling missing data, the prevailing consensus seems to be that discriminative classifiers are almost always to be preferred to generative ones. — Ng und Jordan [NJ01]

Diese Bevorzugung kann dadurch begründet werden, dass diskriminative Modelle in einer Evaluation oft bessere Klassifikationsgüte zeigen als generative Modelle mit vergleichbarer Komplexität [NJ01].

2.4.2 Logistische Regression

Die logistische Regression ist ein typischer Vertreter solcher diskriminativer Modelle. Statt der Verbunddichte $p(\boldsymbol{x}, \omega)$ wird hier die a-posteriori Wahrscheinlichkeit $P(\omega \mid \boldsymbol{x})$ direkt modelliert. Zur Vereinfachung wird im Folgenden von binärer Klassifikation, also der Unterscheidung der Positivklasse ω_1 von der Negativklasse ω_2 ausgegangen. Das Verfahren ist allerdings auf beliebig viele Klassen erweiterbar. Zudem werden Indikatorvariablen $y = \mathbb{1}[\omega = \omega_1]$ anstelle der Klassen ω_1 und ω_2 verwendet. Dabei kodiert $y = 1$ die Positiv- und $y = 0$ die Negativklasse.
Bei einer solchen binären Klassifikation kann anstatt anhand der a-posteriori Klassenwahrscheinlichkeit anhand des Logits entschieden werden. Der Logit ist der Logarithmus des Verhältnisses der beiden a-posteriori Wahrscheinlichkeiten,

$$\text{Logit}\,(\omega \mid \boldsymbol{x}) = \log \frac{P(y = 1 \mid \boldsymbol{x})}{P(y = 0 \mid \boldsymbol{x})} = \log \frac{P(y = 1 \mid \boldsymbol{x})}{1 - P(y = 1 \mid \boldsymbol{x})}, \tag{2.33}$$



(a) Logit $(\omega \mid \boldsymbol{x})$ in Abhängigkeit von $P(y = 1 \mid \boldsymbol{x})$ (blau) und lineare Approximation $\boldsymbol{w}^\top \boldsymbol{x} + b$ (orange)

(b) A-Posteriori Wahrscheinlichkeit für $y = 1$ in Abhängigkeit der linearen Approximation von Logit $(\omega \mid \boldsymbol{x})$

Abbildung 2.5: Zugrundeliegende Funktionen der logistischen Regression.

und bildet den Bereich $[0, 1]$ auf die erweiterten reellen Zahlen $\mathbb{R} \cup \{-\infty, \infty\}$ ab. Damit kann eine zur MAP-Klassifikation äquivalente Klassifikationsvorschrift formuliert werden:

$$\hat{\omega}(\boldsymbol{x}) = \begin{cases} \omega_1 & \text{wenn } \text{Logit}\,(\omega \mid \boldsymbol{x}) > 0 \\ \omega_2 & \text{sonst.} \end{cases} \tag{2.34}$$

Diese Vorschrift gilt, da aus $P(y = 1 \mid \boldsymbol{x}) > P(y = 0 \mid \boldsymbol{x})$ folgt, dass $\text{Logit}\,(\omega \mid \boldsymbol{x}) > 0$ und andernfalls $\text{Logit}\,(\omega \mid \boldsymbol{x}) \leq 0$. Dabei ist $\text{Logit}\,(\omega \mid \boldsymbol{x})$ als Funktion über $\boldsymbol{x}$ zu verstehen. Die Entscheidungsgrenze $P(y = 1 \mid \boldsymbol{x}) = P(y = 0 \mid \boldsymbol{x})$ ist die Menge $\partial\mathcal{R}_1 = \{\boldsymbol{x} \in \mathbb{M} \mid \text{Logit}\,(\omega \mid \boldsymbol{x}) = 0\}$ und die sichere Entscheidung $P(y = 1 \mid \boldsymbol{x}) = 1$ bzw. $P(y = 0 \mid \boldsymbol{x}) = 1$ entspricht $\text{Logit}\,(\omega \mid \boldsymbol{x}) = \infty$ bzw. $\text{Logit}\,(\omega \mid \boldsymbol{x}) = -\infty$.
Unter der Billigung von Approximationsfehlern, insbesondere für große Absolutbeträge $|\text{Logit}\,(\omega \mid \boldsymbol{x})|$, kann diese Funktion nun durch

$$\text{Logit}\,(\omega \mid \boldsymbol{x}) \approx \boldsymbol{w}^\top \boldsymbol{x} + b \tag{2.35}$$

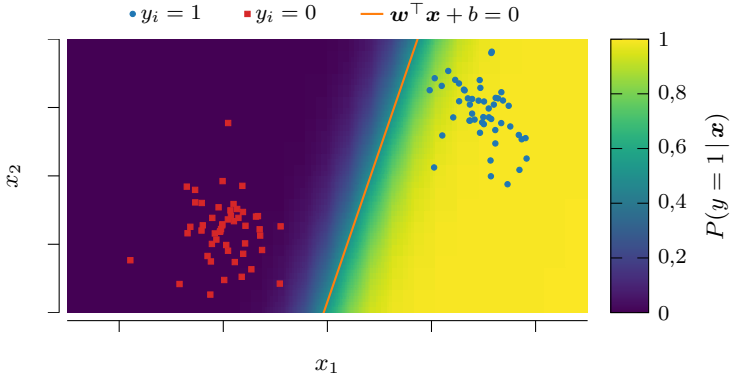


Abbildung 2.6: A-Posteriori Wahrscheinlichkeit der logistischen Regression und zugehörige Entscheidungsgrenze (orange) im Merkmalsraum.

linear approximiert werden. Da für die Klassifikation besonders die Werte um $P(y = 1 \mid \boldsymbol{x}) = 0{,}5$ wichtig sind, fällt der Approximationsfehler hier allerdings nicht so stark ins Gewicht. Abbildung 2.5a zeigt den Zusammenhang von Logit $(\omega \mid \boldsymbol{x})$ und $P(y = 1 \mid \boldsymbol{x})$ sowie die lineare Approximation aus Gleichung (2.35).

Die lineare Approximation des Logits kann durch Gleichsetzen der Gleichungen (2.33) und (2.35) und Auflösen nach $P(y = 1 \mid \boldsymbol{x})$ in eine Wahrscheinlichkeit überführt werden,

$$\log \frac{P(y = 1 \mid \boldsymbol{x})}{1 - P(y = 1 \mid \boldsymbol{x})} \approx \boldsymbol{w}^\top \boldsymbol{x} + b \tag{2.36}$$

$$\Leftrightarrow \quad P(y = 1 \mid \boldsymbol{x}) \approx \frac{\mathrm{e}^{\boldsymbol{w}^\top \boldsymbol{x} + b}}{1 - \mathrm{e}^{\boldsymbol{w}^\top \boldsymbol{x} + b}} = \frac{1}{1 + \mathrm{e}^{-\boldsymbol{w}^\top \boldsymbol{x} - b}}. \tag{2.37}$$

Somit wird die a-posteriori Wahrscheinlichkeit implizit durch die Parameter $\boldsymbol{\theta} := (\boldsymbol{w}^\top, b)^\top$ modelliert. Abbildung 2.5b zeigt den Zusammenhang der linearen Approximation $\boldsymbol{w}^\top \boldsymbol{x} + b$ und der a-posteriori Wahrscheinlichkeit $P(y = 1 \mid \boldsymbol{x})$. Anhand dieser Schätzung kann die MAP-Klassifikation wie in Gleichung (2.29) durchgeführt werden. Abbildung 2.6 zeigt ein Beispiel

der Schätzung der a-posteriori Wahrscheinlichkeit $P(y = 1 \mid \boldsymbol{x})$ mit logistischer Regression und der dazugehörigen Entscheidungsgrenze in einem zweidimensionalen Merkmalsraum.
Zur Schätzung der Parameter $\boldsymbol{\theta}$ bietet sich ein Maximum-Likelihood-Ansatz an [HTF09]. Anstelle der Likelihoodfunktion wird hier die log-Likelihoodfunktion

$$\ell\left(\boldsymbol{\theta} \mid \mathcal{D}\right) = \log \mathfrak{L}\left(\boldsymbol{\theta} \mid \mathcal{D}\right) \tag{2.38}$$

$$= \log\left(\prod_{\boldsymbol{x}_n \in \mathcal{D}} p(\boldsymbol{x}_n \mid \boldsymbol{\theta})\right) = \sum_{\boldsymbol{x}_n \in \mathcal{D}} \log p(\boldsymbol{x}_n \mid \boldsymbol{\theta}) \tag{2.39}$$

maximiert. Die benötigte, zu maximierende Größe $p(\boldsymbol{x}_n \mid \boldsymbol{\theta})$ ist dabei die geschätzte a-posteriori Wahrscheinlichkeit der Klasse des zugehörigen Beispiels. Mit den Indikatorvariablen y_n kann diese Wahrscheinlichkeit durch

$$p(\boldsymbol{x}_n \mid \boldsymbol{\theta}) = P(y = 1 \mid \boldsymbol{x}_n)^{y_n} \; P(y = 0 \mid \boldsymbol{x}_n)^{1-y_n} \tag{2.40}$$

ausgedrückt werden. Die Indikatorvariable $y \in \{0, 1\}$ selektiert also die für ein gegebenes Beispiel $\boldsymbol{x}_n$ relevante a-posteriori Klassenwahrscheinlichkeit. Insgesamt ergibt sich für die log-Likelihoodfunktion somit:

$$\begin{aligned}
\ell\left(\boldsymbol{\theta} \mid \mathcal{D}\right) &= \sum_{\boldsymbol{x}_n \in \mathcal{D}} \Big(y_n \log P(y{=}1 \mid \boldsymbol{x}_n) + (1{-}y_n) \log\left(1{-}P(y{=}1 \mid \boldsymbol{x}_n)\right) \Big) \\
&= \sum_{\boldsymbol{x}_n \in \mathcal{D}} \left(\log\left(1 - P(y = 1 \mid \boldsymbol{x}_n)\right) + y_n \log \frac{P(y = 1 \mid \boldsymbol{x}_n)}{1 - P(y = 1 \mid \boldsymbol{x}_n)} \right) \\
&= \sum_{\boldsymbol{x}_n \in \mathcal{D}} \left(-\log\left(\mathrm{e}^{\boldsymbol{\theta}^\top \boldsymbol{x}'_n} + 1\right) + y_n \boldsymbol{\theta}^\top \boldsymbol{x}'_n \right).
\end{aligned} \tag{2.41}$$

Zur Vereinfachung der Notation wurde hier $\boldsymbol{\theta}^\top \boldsymbol{x}'_n$ mit ${\boldsymbol{x}'_n}^\top := (\boldsymbol{x}^\top, 1)^\top$ und $\boldsymbol{\theta}^\top := (\boldsymbol{w}^\top, b)^\top$ anstelle von $\boldsymbol{w}^\top \boldsymbol{x} + b$ verwendet.

Eine notwendige Bedingung für die Maximierung der log-Likelihood in Gleichung (2.41) ist, dass der Gradient in Bezug zu $\boldsymbol{\theta}$ verschwindet, also dass $\nabla_{\boldsymbol{\theta}} \ell\left(\boldsymbol{\theta} \,|\, \mathcal{D}\right) = \mathbf{0}$. Dies ist allerdings nur gegeben, wenn die a-posteriori Klassenwahrscheinlichkeiten $P(y = 1 \,|\, \boldsymbol{x})$ mit den wahren Klassen bzw. den zugehörigen Indikatorvariablen übereinstimmen:

$$\nabla_{\boldsymbol{\theta}} \ell\left(\boldsymbol{\theta} \,|\, \mathcal{D}\right) = \sum_{\boldsymbol{x}_n \in \mathcal{D}} \left(\nabla_{\boldsymbol{\theta}} \left(y_n \boldsymbol{\theta}^\top \boldsymbol{x}'_n - \log\left(\mathrm{e}^{\boldsymbol{\theta}^\top \boldsymbol{x}'_n} + 1 \right) \right) \right) \tag{2.42}$$

$$= \sum_{\boldsymbol{x}_n \in \mathcal{D}} \left(y_n - \frac{1}{1 + \mathrm{e}^{-\boldsymbol{\theta}^\top \boldsymbol{x}'_n}} \right) \boldsymbol{x}'_n \tag{2.43}$$

$$= \sum_{\boldsymbol{x}_n \in \mathcal{D}} \left(y_n - P(y = 1 \,|\, \boldsymbol{x}_n) \right) \boldsymbol{x}'_n. \tag{2.44}$$

Da dies, wie aus Gleichung (2.37) ersichtlich, nur für $\boldsymbol{w}^\top \boldsymbol{x} + b = \pm\infty$ gilt, ist die Bedingung nicht erfüllbar. Dies gilt auch dann, wenn die Klassen im Merkmalsraum linear separierbar sind, d. h. mittels einer linearer Funktion ohne Fehler voneinander getrennt werden können. Es existiert also keine analytische Lösung zur Parameterschätzung der logistischen Regression. Stattdessen muss die (log-)Likelihoodfunktion hier numerisch minimiert werden. Neben den Verfahren aus Abschnitt 2.3 wird in der Praxis dazu häufig der Iteratively Reweighted Least Squares Algorithmus verwendet, der auf dem iterativen Newton-Raphson Verfahren zur Optimierung konvexer Probleme basiert:

$$\boldsymbol{\theta}_{t+1} := \boldsymbol{\theta}_t - \left(\frac{\partial^2 \ell\left(\boldsymbol{\theta}_t \,|\, \mathcal{D}\right)}{\partial \boldsymbol{\theta}\, \partial \boldsymbol{\theta}^\top} \right)^{-1} \nabla_{\boldsymbol{\theta}} \ell\left(\boldsymbol{\theta}_t \,|\, \mathcal{D}\right). \tag{2.45}$$

Eine Herleitung dieses Algorithmus findet sich z. B. in Hastie et al. [HTF09], S. 120 ff. Wie in Abbildung 2.6 zu sehen ist, entspricht die so gefundene Entscheidungsgrenze jedoch nicht immer der Lösung, die ein Mensch gewählt hätte. Das Finden einer solchen Lösung erfordert zusätzliche Ein-

schränkung des Parameterraums, die üblicherweise durch einen zusätzlichen Regularisierungsterm $R(\boldsymbol{\theta})$ im Optimierungsproblems kodiert wird:

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} \ell\left(\boldsymbol{\theta} \mid \mathcal{D}\right) - \lambda\, R(\boldsymbol{\theta}). \tag{2.46}$$

Typische Regularisierungen sind die L_2-Regularisierung $R(\boldsymbol{\theta}) = \|\boldsymbol{w}\|_2$ und die L_1-Regularisierung $R(\boldsymbol{\theta}) = \|\boldsymbol{w}\|_1$. Erstere bevorzugt kleine Parameter und vermeidet somit große Koeffizienten für nicht relevante Merkmale. Dieses auch als „Ridge Regression" bekannte Verfahren weist starke Parallelen mit der in Abschnitt 2.4.4 diskutierten Support Vektor Maschine auf. Die L_1-Regularisierung verstärkt diesen Effekt zusätzlich, sodass die Einträge des Gewichtsvektors $\boldsymbol{w}$ nur dann von Null verschieden sind, wenn das zugehörige Merkmal für die Klassifikation relevant ist. Solche Modelle lassen sich üblicherweise gut von Menschen interpretieren. Da der Betrag der Koeffizienten von $\boldsymbol{w}$ umso größer ist, je relevanter das dazugehörige Merkmale, kann dieses als „LASSO" bekannte Verfahren auch als eingebettetes Verfahren zur Merkmalsselektion verwendet werden [Tib96].
Beide Regularisierungen verringern die Varianz der logistischen Regression, erhöhen aber notwendigerweise den Bias. Eine Bayes'sche Interpretation dieser und anderer Regularisierungen liefert die Theorie der Sparse Linear Models [HTW15]. Eine Diskussion dieses Themas geht allerdings über den Rahmen dieser Arbeit hinaus.

2.4.3 Hypothesenfunktionen

Wenn eine Schätzung der Klassenwahrscheinlichkeit nicht benötigt wird, kann mit der logistischen Regression auch direkt anhand des Vorzeichens

von $\boldsymbol{w}^\top \boldsymbol{x} + b$ klassifiziert werden, anstatt nach der maximalen a-posteriori Wahrscheinlichkeit:

$$\hat{\omega}(\boldsymbol{x}) = \begin{cases} \omega_1 & \text{wenn } h(\boldsymbol{x}) := \boldsymbol{w}^\top \boldsymbol{x} + b > 0 \\ \omega_2 & \text{sonst.} \end{cases} \tag{2.47}$$

Diese Formulierung motiviert eine alternative Sicht auf das Klassifikationsproblem: Gesucht ist eine Funktion $h(\boldsymbol{x})$, anhand derer ein Klassifikator $\hat{\omega}(\boldsymbol{x})$ definiert werden kann. In diesem Kontext heißt $h(\boldsymbol{x})$ die Hypothesenfunktion des Klassifikators. Wie bei der MAP-Klassifikation soll die Hypothesenfunktion so gewählt werden, dass der korrespondierende Klassifikator $\hat{\omega}$ die Fehlerwahrscheinlichkeit P_e bzw. das Risiko $\mathfrak{R}(\hat{\omega})$ minimiert.
Im Gegensatz zur MAP-Klassifikation können die Hypothesenfunktionen aber beliebiger Art sein und unterliegen insbesondere auch nicht den durch die Kolmogorov'schen Axiome auferlegten Einschränkungen. Die Formulierung des Klassifikators in Gleichung (2.47), also der Klassifikation anhand des Vorzeichens von $h(\boldsymbol{x})$, ist dabei typisch für binäre Klassifikationsprobleme. Neben einer linearen Hypothesenfunktion sind aber auch polynomiale, hyperbolische oder sonstige nichtlineare Funktionen möglich. In der Tat lassen sich viele parametrische probabilistische Klassifikatoren durch algebraische Umformungen in nichtlineare Hypothesenfunktionen überführen.
Die Hypothesenfunktionen sind auch nicht notwendigerweise skalar. Ein Beispiel ist der k nächste Nachbarn (kNN) Klassifikator, der einem gegebenen Objekt die unter den k nächsten Nachbarn im Merkmalsraum am häufigsten vertretene Klasse zuweist. Die Hypothesenfunktion des Klassifikators ist

$$\boldsymbol{h}(\boldsymbol{x}) = \left(N_k\left(\boldsymbol{x}, \omega_1\right), \ldots, N_k\left(\boldsymbol{x}, \omega_C\right)\right)^\top \in \mathbb{N}^C. \tag{2.48}$$

Hierbei zählt $N_k(\boldsymbol{x}, \omega_c)$ die Anzahl der k nächsten Nachbarn von $\boldsymbol{x}$ in der Lernstichprobe $\mathcal{D}$, die der Klasse ω_c zugeordnet sind. Anhand dieser Hypothesenfunktion wird die Klasse durch

$$\hat{\omega}(\boldsymbol{x}) = \omega_{c^*} \quad \text{mit} \quad c^* = \arg\max_c \left[\boldsymbol{h}(\boldsymbol{x})\right]_c \tag{2.49}$$

zugewiesen. Eine solche Hypothesenfunktion und Klassifikationsvorschrift ist typisch für eine Klassifikation mit mehr als zwei Klassen.
Wie oben erwähnt, kann die Suche nach einer geeigneten Hypothese wie bei der MAP-Klassifikation als Optimierungsproblem formuliert werden. In der allgemeinsten Formulierung ist eine Menge an Hypothesenfunktionen $\mathcal{H}$ gegeben. Aus dieser (möglicherweise überabzählbar unendlichen) Menge soll nun diejenige Hypothese $h^* \in \mathcal{H}$ ausgewählt werden, die den Erwartungswert einer ebenfalls gegebene Verlustfunktion $l\left(h(\boldsymbol{x}), y\right)$ minimiert,

$$h^* = \arg\min_{h \in \mathcal{H}} \mathbb{E}_{\boldsymbol{x},y} \left\{ l\left(h(\boldsymbol{x}), y\right) \right\} = \arg\min_{h \in \mathcal{H}} \mathfrak{R}\left(h\right). \tag{2.50}$$

Diese Formulierung ist analog zur Risikominimierung aus Gleichung (2.32). In der Tat wurde hier lediglich der Klassifikator $\hat{\omega}(\boldsymbol{x})$ durch die Hypothesenfunktion $h(\boldsymbol{x})$ und die Klasse ω durch die Indikatorvariable y ausgetauscht. Unglücklicherweise kann das Risiko in Gleichung (2.50) in der Praxis nicht berechnet werden, da die dafür benötigte Verteilung $p(\boldsymbol{x}, y)$ nicht bekannt ist. Mit einer ausreichend umfangreichen und repräsentativen Stichprobe $\mathcal{D}$ mit bekannten Klassenzugehörigkeiten kann das Risiko allerdings durch den mittleren Verlust auf der Stichprobe

$$\mathfrak{R}\left(h\right) \approx \mathfrak{R}_{\text{emp}}\left(h\right) = \frac{1}{N} \sum_{\boldsymbol{x}_n \in \mathcal{D}} l\left(h(\boldsymbol{x}_n), y_n\right) \tag{2.51}$$

angenähert werden. Dieses Vorgehen ist als Empirical Risk Minimization (ERM) bekannt und ein grundlegendes Prinzip Vapniks statistischer Lerntheorie [Vap95].

In Verbindung mit geeigneten Optimierungsverfahren lassen sich mittels ERM beliebige Klassifikatoreigenschaften erwirken, indem eine passende Verlustfunktion gewählt wird. Beispielsweise führt der sogenannte Hinge-Loss $l\left(h(\boldsymbol{x}), y\right) = \max\left(0, 1 - h(\boldsymbol{x}) \cdot y\right)$ mit $h(\boldsymbol{x}) \in \mathbb{R}$ und $y = \pm 1$ zu einer Klassifikation mit maximalem Rand, d. h. mit maximalem Abstand von der Entscheidungsgrenze zu den nächsten Beispielen der Lernstichprobe. Ebenso kann durch die Verlustfunktion Vorwissen über das Klassifikationsproblem eingebracht werden. Durch klassenbedingte Gewichtung der Verlustfunktion können etwa die a-priori Klassenwahrscheinlichkeiten kodiert werden, ohne dass diese a-priori Wahrscheinlichkeiten bei der eigentlichen Klassifikation verwendet werden.
Vapnik und Chervonenkis entwickelten neben dem ERM-Prinzip auch das Prinzip der Structural Risk Minimization (SRM) [Vap95]. SRM ist eng mit dem Bias-Varianz Trade-off, bzw. des in Abbildung 2.4 skizzierten Zusammenhang von Modellkomplexität und Klassifikationsfehler verbunden. Dieser Zusammenhang wird abgeschätzt, indem die Modellkomplexität anhand der sogenannten VC-Dimension ν quantifiziert wird. Unter Zuhilfenahme der daraus abgeleiteten VC-Konfidenz $\Phi(\nu, N, \eta)$ kann eine untere Schranke für das wahre Risiko $\mathfrak{R}\left(h\right)$ angegeben werden. Weitere Details finden sich in der Arbeit von Vapnik [Vap95].

2.4.4 Support Vektor Maschine

Die Überlegungen dieser Theorie führten zur Entwicklung der Support Vektor Maschine (SVM), die noch immer eines der wichtigsten Klassifikationsverfahren der Mustererkennung darstellt. Die Gründe hierfür sind zum einen das solide theoretische Fundament – VC-Lerntheorie sowie reproduzierbare Kernel-Hilberträume (siehe z. B. Schölkopf und Smola [SS01]) – und die daraus ableitbaren Garantien der Klassifikation sowie die Anwendbarkeit auf viele verschiedene Problemdomänen. Zum anderen erzielen SVMs bereits mit kleinen Stichproben und auch in hochdimensionalen Merkmalsräumen

sehr gute Klassifikationsergebnisse. Der hier vorgestellte Zugang geht auf die Formulierung von Boser et al. [BGV92] zurück [5].
Wie die logistische Regression (Abschnitt 2.4.2) ist die SVM ein linearer, binärer Klassifikator mit der Hypothesenfunktion

$$h(\boldsymbol{x}) = \boldsymbol{w}^\top \phi(\boldsymbol{x}) + b. \tag{2.52}$$

Im Gegensatz zur logistischen Regression werden die Merkmale $\boldsymbol{x} \in \mathbb{M}$ hier jedoch zusätzlich mit einer Abbildung $\phi : \mathbb{M} \to \Phi$ in einen höherdimensionalen Raum ($\dim(\Phi) \geq \dim(\mathbb{M})$) transformiert. Der Klassifikator trennt also linear in Φ, ermöglicht aber nichtlineare Entscheidungsgrenzen in $\mathbb{M}$. Ebenfalls anders als bei der logistischen Regression sind die Indikatorvariablen hier $y = \pm 1$, wobei $y = 1$ für die Positivklasse ω_1 und $y = -1$ für die Negativklasse ω_2 steht.
Zur Motivation der SVM dient erneut die Abbildung 2.6. Anders als der Computer hätte ein Mensch die Trennebene hier intuitiv so gewählt, dass der Abstand zu den nächsten Trainingsbeispielen maximiert würde. Eine solche Trennebene minimiert die Wahrscheinlichkeit einer Fehlklassifikation ungesehener Daten. Somit generalisiert der Klassifikator besser auf ungesehenen Daten. Dies ist das Ziel der SVM: Die Parameter $\boldsymbol{w}$ und b sollen nicht nur so gewählt werden, dass die Trainingsdaten korrekt klassifiziert werden, sondern so, dass zusätzlich der Rand, also der Abstand von Hyperebene und Trainingsdaten $\phi(\boldsymbol{x}_n)$ mit $\boldsymbol{x}_n \in \mathcal{D}$, maximiert wird. Das ist natürlich nur möglich, wenn die Daten in Φ linear separierbar sind. Zur Herleitung der SVM soll – für den Moment – angenommen werden, dass dies der Fall ist. Diese Beschränkung wird später wieder aufgehoben.
Damit der Rand maximiert werden kann, muss dieser zunächst quantifiziert werden. Für einen gegebenen Merkmalsvektor $\boldsymbol{x}$ entspricht der Betrag der Hypothesenfunktion $h(\boldsymbol{x}) = \boldsymbol{w}^\top \phi(\boldsymbol{x}) + b$ den um $\|\boldsymbol{w}\|$ skalierten Abstand

[5] Weitere Meilensteine der Entwicklungsgeschichte der SVM sind auf der Website `http://www.svms.org/history.html` (abgerufen am 06.08.2018) zusammengefasst.

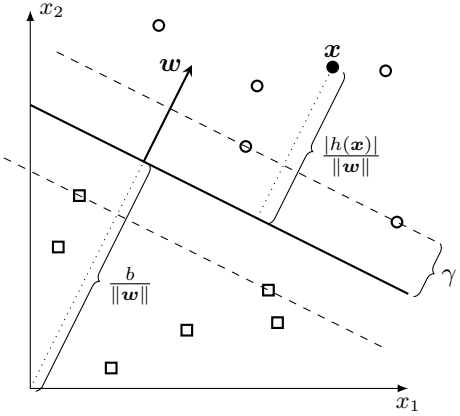


Abbildung 2.7: Zusammenhang von Lernstichprobe, Abstand der Hyperebene zu Merkmalen und Nullpunkt und dem Rand γ. Hier ist $\mathbb{M} = \mathbb{R}^2$ und $\phi(\boldsymbol{x}) = \boldsymbol{x}$. Grafik nach Boser et al. [BGV92].

von $\phi(\boldsymbol{x})$ zur Hyperebene (siehe Abbildung 2.7). Unter der Annahme, dass $h(\boldsymbol{x}) = 0$ eine gültige Trennebene ist, gilt für Beispiele der Positivklasse $h(\boldsymbol{x}) > 0$ und $y = 1$ und für Beispiele der Negativklasse $h(\boldsymbol{x}) < 0$ und $y = -1$. Weiterhin liegen alle transformierten Merkmalsvektoren der Lernstichprobe auf oder außerhalb des Randes der separierenden Hyperebene, d. h. mit γ als der Größe des Rands gilt für alle $\boldsymbol{x}_n \in \mathcal{D}$

$$\frac{y_n h(\boldsymbol{x}_n)}{\|\boldsymbol{w}\|} \geq \gamma. \tag{2.53}$$

Da der Betrag von $h(\boldsymbol{x})$ für die Klassifikation unerheblich ist, gilt ohne Beschränkung der Allgemeinheit

$$\min_{\boldsymbol{x}_n \in \mathcal{D}} y_n \, h(\boldsymbol{x}_n) = 1. \tag{2.54}$$

Zusammen mit Gleichung (2.53) ergibt sich für den Rand also:

$$\gamma = \frac{1}{\|\boldsymbol{w}\|}. \tag{2.55}$$

Der Rand wird maximal, wenn $\|\boldsymbol{w}\|$ ein Minimum annimmt. Das führt zu folgendem quadratischem Optimierungsproblem mit Nebenbedingungen zur Festlegung der Parameter $\boldsymbol{w}, b$:

$$\min_{\boldsymbol{w},b} \frac{1}{2}\|\boldsymbol{w}\|^2 \qquad \text{s.t.} \quad y_n(\boldsymbol{w}^\top \phi(\boldsymbol{x}_n) + b) \geq 1 \quad \forall \boldsymbol{x}_n \in \mathcal{D}. \tag{2.56}$$

Hierbei erzwingt das Optimierungziel die Maximierung des Rands, während die Nebenbedingungen sicherstellen, dass alle Beispiele richtig klassifiziert werden. Mittels Lagrange-Multiplikatoren kann dieses Optimierungsproblem in ein Optimierungsproblem ohne Nebenbedingungen überführt werden. Es ergibt sich das Optimierungsfunktional

$$L(\boldsymbol{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_{\boldsymbol{x}_n \in \mathcal{D}} \alpha_n \left(y_n \left(\boldsymbol{w}^\top \phi(\boldsymbol{x}_n) + b \right) - 1 \right) \tag{2.57}$$

mit $\alpha_n \geq 0$ für $n = 1, \ldots, N$.
Die optimale Lösung $(\boldsymbol{w}^*, b^*)$ von Gleichung (2.56) liegt auf einem Sattelpunkt von $L(\boldsymbol{w}, b, \boldsymbol{\alpha})$. Dieser Sattelpunkt ist ein Minimum in Bezug auf die Parameter $\boldsymbol{w}$ und b und ein Maximum in Bezug auf $\boldsymbol{\alpha}$ [BGV92]. An dieser Lösung gilt:

$$\nabla_{\boldsymbol{w}} L(\boldsymbol{w}^*, b^*, \boldsymbol{\alpha}^*) = \boldsymbol{w}^* - \sum_{\boldsymbol{x}_n \in \mathcal{D}} \alpha_n^* y_n \phi(\boldsymbol{x}_n) = \mathbf{0} \tag{2.58}$$

$$\Rightarrow \quad \boldsymbol{w}^* = \sum_{\boldsymbol{x}_n \in \mathcal{D}} \alpha_n^* y_n \phi(\boldsymbol{x}_n). \tag{2.59}$$

Der Normalenvektor der Hyperebene ist also eine Linearkombination der Trainingsbeispiele. Eine ähnliche Beobachtung ergibt sich beim Perzeptron-

Algorithmus [Ros61], allerdings wird hier der Gewichtsvektor $\boldsymbol{w}$ explizit als Linearkombination der Trainingsbeispiele konstruiert. Weiterhin gilt für den Gradienten in Bezug auf b:

$$\nabla_b L(\boldsymbol{w}^*, b^*, \boldsymbol{\alpha}^*) = \sum_{\boldsymbol{x}_n \in \mathcal{D}} \alpha_n^* y_n = 0. \tag{2.60}$$

Einsetzen von Gleichungen (2.59) und (2.60) in Gleichung (2.57) sowie Umformung ergibt das duale Optimierungsproblem, das nun nur noch von $\boldsymbol{\alpha}$ abhängig ist [BGV92]:

$$\begin{aligned} &\max_{\boldsymbol{\alpha}} \sum_{\boldsymbol{x}_n \in \mathcal{D}} \alpha_n - \frac{1}{2} \sum_{\boldsymbol{x}_n \in \mathcal{D}} \sum_{\boldsymbol{x}_k \in \mathcal{D}} y_n y_k \, \alpha_n \alpha_k \, \phi(\boldsymbol{x}_n)^\top \phi(\boldsymbol{x}_k) \\ &\text{s.t.} \quad \sum_{\boldsymbol{x}_n \in \mathcal{D}} y_n \alpha_n = 0 \quad \text{und} \quad \alpha_n \geq 0 \quad \forall \boldsymbol{x}_n \in \mathcal{D}. \end{aligned} \tag{2.61}$$

Dieses quadratische Optimierungsproblem kann mit Standardverfahren oder mit speziell für die SVM entwickelten Algorithmen, wie der Sequential Minimal Optimization [Pla98], gelöst werden. Unter Verwendung von Gleichung (2.59) und der optimalen Lösung $\boldsymbol{\alpha}^*$ des dualen Problems ergibt sich die Hypothesenfunktion als

$$h(\boldsymbol{x}) = {\boldsymbol{w}^*}^\top \phi(\boldsymbol{x}) + b^* = b^* + \sum_{\boldsymbol{x}_n \in \mathcal{D}} \alpha_n^* y_n \phi(\boldsymbol{x}_n)^\top \phi(\boldsymbol{x}), \tag{2.62}$$

wobei der Parameter b^* nach Optimierung des dualen Problems so gewählt wird, dass die Hyperebene mittig zwischen den Klassen im transformierten Merkmalsraum liegt,

$$2\, b^* = \max_{y_n = -1} \sum_{\boldsymbol{x}_k \in \mathcal{D}} \alpha_k^* y_k \phi(\boldsymbol{x}_k)^\top \phi(\boldsymbol{x}_n) - \min_{y_n = 1} \sum_{\boldsymbol{x}_k \in \mathcal{D}} \alpha_k^* y_k \phi(\boldsymbol{x}_k)^\top \phi(\boldsymbol{x}_n). \tag{2.63}$$

Somit ist die Hyperebene vollständig durch die Beispiele $\boldsymbol{x}_n$ mit $\alpha_n \neq 0$ bestimmt [BGV92]. Aus diesem Grund werden diese $\boldsymbol{x}_n$ Support Vektoren genannt. Geometrisch betrachtet sind die Support Vektoren die $\boldsymbol{x}_n$, deren Bild $\phi(\boldsymbol{x}_n)$ am nächsten an der trennenden Hyperebene liegen. Die Anzahl der zu bestimmenden Parameter α_n^* ist damit unabhängig von der Dimension des Merkmalsraums und wird vielmehr durch die Komplexität der Daten bzw. des Klassifikationsproblems bestimmt [BGV92]. Im Prinzip können dadurch Transformationen in sehr hochdimensionale Räume Φ verwendet werden, was wiederum nahezu beliebig komplizierte Entscheidungsgrenzen im Merkmalsraum $\mathbb{M}$ ermöglicht. Allerdings ist eine solche Transformation und die Auswertung der Hypothesenfunktion unter Umständen mit hohem Rechenaufwand verbunden.
Der Kernel-Trick schafft Abhilfe: Sowohl im Optimierungsfunktional, als auch in der Hypothesenfunktion in Gleichungen (2.61) und (2.62) tauchen die transformierten Merkmale $\phi(\boldsymbol{x})$ nur innerhalb von Skalarprodukten auf. Sie können also durch eine Funktion

$$K(\boldsymbol{u}, \boldsymbol{v}) = \phi(\boldsymbol{u})^\top \phi(\boldsymbol{v}) \quad \forall \boldsymbol{u}, \boldsymbol{v} \in \mathbb{M} \tag{2.64}$$

ersetzt werden, die die Transformation und das Skalarprodukt implizit berechnet. Solche Funktionen werden als Kernfunktionen bzw. Kernel bezeichnet. Die Hypothesenfunktion in Gleichung (2.65) ist eine gewichtete Summe von Kernfunktionen,

$$h(\boldsymbol{x}) = b^* + \sum_{\boldsymbol{x}_n \in \mathcal{D}} y_n \alpha_n^* K(\boldsymbol{x}_n, \boldsymbol{x}). \tag{2.65}$$

Ob eine gegebene Funktion $K(\boldsymbol{u}, \boldsymbol{v})$ einem Skalarprodukt in Φ entspricht, kann mit den Bedingungen aus dem Satz von Mercer überprüft werden [Mer09]: Erstens muss $K : \mathbb{M}^2 \to \mathbb{R}$ symmetrisch sein, d. h. für alle $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{M}$ muss $K(\boldsymbol{u}, \boldsymbol{v}) = K(\boldsymbol{v}, \boldsymbol{u})$ gelten. Zweitens muss für alle quadratin-

tegrablen Funktionen $g : \mathbb{M} \to \mathbb{R}$, d. h. Funktionen mit $\int_{\mathbb{M}} \left(g(\boldsymbol{x})\right)^2 \, \mathrm{d}\boldsymbol{x} < \infty$, gelten, dass

$$\iint\limits_{\mathbb{M}\times\mathbb{M}} g(\boldsymbol{u})\, K(\boldsymbol{u},\boldsymbol{v})\, g(\boldsymbol{v}) \, \mathrm{d}\boldsymbol{u} \, \mathrm{d}\boldsymbol{v} \geq 0. \tag{2.66}$$

Diese Bedingung kann als Verallgemeinerung positiv semidefiniter Matrizen $\boldsymbol{K} \in \mathbb{R}^{D\times D}$ auf den Raum der symmetrischen bivariaten Funktionen aufgefasst werden. Für eine gegebene Kernfunktion $K(\boldsymbol{u},\boldsymbol{v})$ kann die implizit berechnete Transformation ϕ wiederum durch Lösung des Eigenfunktionsproblems

$$\int_{\mathbb{M}} K(\boldsymbol{u},\boldsymbol{v})\phi_n(\boldsymbol{v}) \, \mathrm{d}\boldsymbol{v} = \lambda_n \phi_n(\boldsymbol{u}) \tag{2.67}$$

gefunden werden. Die Vektorabbildung $\phi(\boldsymbol{u})$ setzt sich dabei durch alle (reellen) Eigenfunktionen $\phi_n(\boldsymbol{u})$ zusammen,

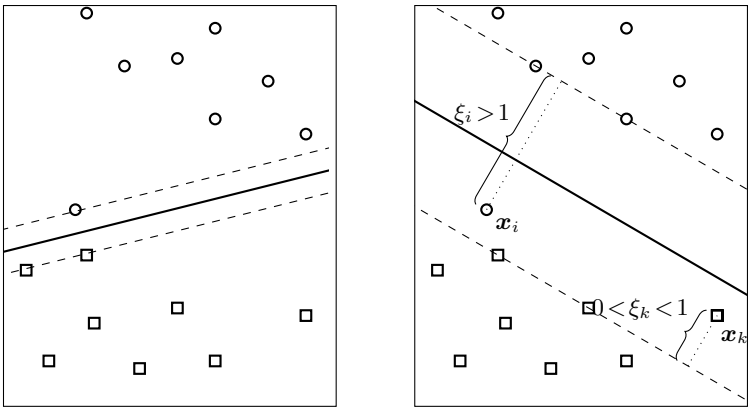
$$\phi(\boldsymbol{u}) = \left(\sqrt{\lambda_1}\phi_1(\boldsymbol{u}), \sqrt{\lambda_2}\phi_2(\boldsymbol{u}), \sqrt{\lambda_3}\phi_3(\boldsymbol{u}), \ldots\right)^\top. \tag{2.68}$$

Die Dimension von Φ entspricht der Anzahl der Eigenfunktionen. Insbesondere kann auch $\dim(\Phi) = \infty$ gelten, wenn der Kernel unendlich viele Eigenfunktionen besitzt. Da die Gleichung (2.65) aber nur von $\boldsymbol{\alpha}$ und b abhängt, müssen für die SVM dennoch nur endlich viele Parameter geschätzt werden.

Die meisten praktischen Klassifikationsprobleme lassen sich mit wenigen parametrischen Standardkerneln behandeln. Die triviale lineare Kernfunktion $K(\boldsymbol{u},\boldsymbol{v}) = \boldsymbol{u}^\top\boldsymbol{v}$ entspricht der Identitätsfunktion $\phi(\boldsymbol{x}) = \boldsymbol{x}$ und erzeugt eine lineare Entscheidungsgrenze in $\mathbb{M}$. Dieser Kernel ist insbesondere dann nützlich, wenn die Klassen linear separierbar und $\dim(\mathbb{M})$ groß ist. Nichtlineare Entscheidungsgrenzen in $\mathbb{M}$ können mit dem Polynomialkernel $K(\boldsymbol{u},\boldsymbol{v}) = \left(\boldsymbol{u}^\top\boldsymbol{v} + \alpha\right)^\beta$ und dem radiale Basisfunktion (RBF) bzw.

Gauß-Kernel $K(\boldsymbol{u}, \boldsymbol{v}) = \mathrm{e}^{-\gamma \|\boldsymbol{u}-\boldsymbol{v}\|^2}$ erzeugt werden. Der Polynomialkernel entspricht dabei der Transformation in den Raum aller Monome der einzelnen Merkmale von Grad $\leq \beta$ [BRN17]. Der RBF-Kernel entspricht einer impliziten Transformation in einen unendlich-dimensionalen Raum Φ, da diese Kernfunktion unendlich vielen Eigenfunktionen besitzt.
Kernfunktionen, und insbesondere der RBF-Kernel, lassen sich auch als Ähnlichkeitsmaße interpretieren. Mit dieser Interpretation lassen sich auch Kernfunktionen für beliebige Objekte wie Zeichenketten [Lod+02] oder Graphen [LK02] angeben. Hier wirkt der Kernel dann direkt auf den Mustern $\boldsymbol{p}$, ohne dass vorher eine Merkmalsextraktion $\psi(\boldsymbol{p})$ angewendet wird.

Nicht linear separierbare Daten. Bisher wurde angenommen, dass die Daten in Φ linear separierbar sind. Durch die Wahl eines geeigneten Kernels, z. B. des RBF-Kernels, kann dies durch Erhöhung der Dimension von Φ immer garantiert werden. Dies erhöht allerdings die Varianz des Klassifikators und somit das Risiko von Overfitting und führt insgesamt zu schlechterer Generalisierungsfähigkeit. Zudem kann, wie in Abbildung 2.8a gezeigt, der Rand durch Ausreißer in linear separierbaren Daten sehr klein werden. Um dem entgegenzuwirken, ist es sinnvoll, kleine Randverletzungen und soger Fehlklassifikationen zuzulassen (siehe Abbildung 2.8b).
Formal lassen sich Randverletzungen durch Einführung von Schlupfvariablen ξ_n für die $\boldsymbol{x}_n \in \mathcal{D}$ modellieren. Mit diesen Schlupfvariablen werden Nebenbedingungen in Gleichung (2.56) aufgeweicht, indem statt $y_n h(\boldsymbol{x}_n) \geq 1$ nun $y_n h(\boldsymbol{x}_n) \geq 1 - \xi_n$ gefordert wird. Die ξ_n messen somit den Grad der Randverletzung. Für $0 < \xi_n \leq 1$ wird $\boldsymbol{x}_n$ zwar richtig klassifiziert, $\phi(\boldsymbol{x}_n)$ liegt aber innerhalb des Randes um die Hyperebene. Für $\xi_n > 1$ liegt das zugehörige $\phi(\boldsymbol{x}_n)$ auf der falschen Seite der Entscheidungsgrenze (siehe



(a) SVM ohne Randverletzung **(b)** SVM mit Randverletzung

Abbildung 2.8: Zulassen von Randverletzungen und Fehlklassifikation kann die Generalisierung des Klassifikators verbessern. Linearer Kernel, $\dim(\mathbb{M}) = 2$ und gleiche Lernstichprobe in beiden Fällen.

Abbildung 2.8b). Um die Zahl der Randverletzungen und Fehlklassifikationen zu minimieren, wird das Optimierungsproblem um einen Strafterm erweitert:

$$\begin{aligned} &\min_{\boldsymbol{w},b} \frac{1}{2}\|\boldsymbol{w}\|^2 + C \sum_{\boldsymbol{x}_n \in \mathcal{D}} \xi_n \\ &\text{s.t.} \quad y_n(\boldsymbol{w}^\top \phi(\boldsymbol{x}_n) + b) \geq 1 - \xi_n \quad \forall \boldsymbol{x}_n \in \mathcal{D}. \end{aligned} \tag{2.69}$$

Das entsprechende duale Optimierungsproblem lautet:

$$\begin{aligned} &\max_{\boldsymbol{\alpha}} \sum_{\boldsymbol{x}_n \in \mathcal{D}} \alpha_n - \frac{1}{2} \sum_{\substack{\boldsymbol{x}_n \in \mathcal{D} \\ \boldsymbol{x}_m \in \mathcal{D}}} y_n y_m \, \alpha_n \alpha_m \left(K(\boldsymbol{x}_n, \boldsymbol{x}_m) + \frac{\mathbb{1}[n=m]}{C} \right) \\ &\text{s.t.} \quad \sum_{\boldsymbol{x}_n \in \mathcal{D}} y_n \alpha_n = 0 \quad \text{und} \quad \alpha_n \geq 0 \quad \forall \boldsymbol{x}_n \in \mathcal{D}. \end{aligned} \tag{2.70}$$

Der Parameter b^* wird so gewählt, dass $y_n h(\boldsymbol{x}_n) = 1 - \frac{\alpha_n^*}{C}$ für alle $\boldsymbol{x}_n \in \mathcal{D}$ mit $\alpha_n^* \neq 0$ gilt. Geometrisch entspricht dies der Bedingung, dass die

Support Vektoren mit $\xi_n = 0$ auf dem Rand um die Entscheidungsgrenze liegen [SS01].
Eine so festgelegte SVM wird als soft margin SVM bezeichnet, da die Randbedingung nicht strikt durchgesetzt wird. Im Gegensatz dazu steht die hard margin SVM. Eine soft margin SVM verfügt üblicherweise über deutlich mehr Support Vektoren als eine hard margin SVM und verursacht entsprechend höheren Rechenaufwand.
Der Hyperparameter C steuert die Anzahl der erlaubten Randverletzungen. Es können auch zwei unabhängige Hyperparameter C_1 und C_{-1} eingeführt werden, mit denen Randverletzungen je nach Klasse unterschiedlich bewertet werden. Das primale Optimierungsproblem lautet dann

$$\min_{\boldsymbol{w},b} \frac{1}{2}\|\boldsymbol{w}\|^2 + C_1 \sum_{\substack{\boldsymbol{x}_n \in \mathcal{D} \\ y_n = 1}} \xi_n + C_{-1} \sum_{\substack{\boldsymbol{x}_n \in \mathcal{D} \\ y_n = -1}} \xi_n \tag{2.71}$$

mit Nebenbedingungen wie in Gleichung (2.69). Mit diesem Vorgehen kann Vorwissen über die a-priori Klassenwahrscheinlichkeit oder die Klassifikationskosten eingebracht werden. Ebenso kann eine unbalancierte Stichprobe mit deutlich mehr Beispielen der Klasse ω_1 bzw. ω_2 ausgeglichen werden. In diesem Fall wird üblicherweise $C_1 = C\frac{N}{N_1}$ und $C_{-1} = C\frac{N}{N_{-1}}$ gewählt, wobei N_1 und N_{-1} die Anzahl der Beispiele der Positiv- bzw. Negativklasse sind.

Optimierung der primalen Form. Die duale Form der SVM ergibt sich natürlich aus der Behandlung der Nebenbedingungen der hard margin SVM mit Lagrange-Multiplikatoren. Mit einer passenden Verlustfunktion $l\left(h(\boldsymbol{x}), y\right)$ lässt sich die soft margin SVM aber auch als ein Optimierungsproblem ohne Nebenbedingungen formulieren:

$$\min_{\boldsymbol{w},b} \|w\|^2 + C \sum_{n=1}^{N} l\left(h(\boldsymbol{x}_n), y_n\right). \tag{2.72}$$

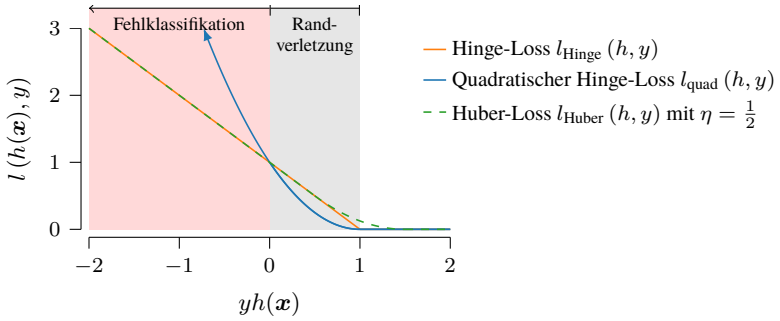


Abbildung 2.9: Vergleich der Verlustfunktionen Hinge-Loss, quadratischer Hinge-Loss und Huber-Loss.

Diese Formulierung entspricht dem ERM-Prinzip mit Regularisierungsterm $R(\boldsymbol{w}) = \frac{1}{C \cdot N} \|\boldsymbol{w}\|$. Die Regularisierung misst wiederum die Komplexität des Klassifikators.

Wie Chapelle [Cha07] zeigt, führt diese Formulierung zu den gleichen Ergebnissen wie die duale Formulierung in Gleichung (2.70). Chapelle formuliert weiterhin ein Newton-Verfahren zur effizienten Optimierung unter Verwendung des quadratischen Hinge-Loss und des Huber-Loss mit Bandbreite η (eine differenzierbare Approximation des Hinge-Loss, siehe Abbildung 2.9):

$$l_{\text{quad}}(h(\boldsymbol{x}), y) = l_{\text{Hinge}}(h(\boldsymbol{x}), y)^2 = (\max(0, 1 - yh(\boldsymbol{x})))^2, \tag{2.73}$$

$$l_{\text{Huber}}(h(\boldsymbol{x}), y) = \begin{cases} 0 & \text{wenn } yh(\boldsymbol{x}) > 1 + \eta \\ \frac{(1+\eta-yh(\boldsymbol{x}))^2}{4\eta} & \text{wenn } |1 - yh(\boldsymbol{x})| \leq \eta \\ 1 - yh(\boldsymbol{x}) & \text{wenn } yh(\boldsymbol{x}) < 1 - \eta \end{cases}. \tag{2.74}$$

Die Optimierung mit diesem Verfahren führt zwar zu den gleichen Ergebnissen wie die duale Optimierung, konvergiert aber insbesondere dann schneller, wenn die Anzahl N der Trainingsbeispiele deutlich größer ist als die Dimension $\dim(\Phi)$ des induzierten Merkmalsraums. Der Grund hierfür ist, dass

in diesem Fall die Anzahl der Support Vektoren hoch ist und die exakte Lösung der SVM nicht berechnet werden kann. In diesem Fall konvergiert die primale Optimierung schneller zu einer guten Näherung als die duale Optimierung [Cha07].
Ein weiterer Vorteil der primalen Optimierung ist, dass der Hyperparameter C und die Kernelparameter gleichzeitig optimiert werden können, statt wie bei der dualen Formulierung alternierend [Cha07].

Mehrklassen-SVM. Ein entscheidender Nachteil der SVM ist die Beschränkung auf binäre Klassifikation. Zwar gibt es z. B. mit der Formulierung von Crammer und Singer [CS01] echte Mehrklassen-SVMs, in der Praxis haben sich jedoch zwei Heuristiken durchgesetzt, die das Mehrklassenproblem in mehrere binäre Entscheidungsprobleme überführen. Diese Ansätze sind nicht spezifisch für die SVM, sondern mit beliebigen binären Klassifikatoren einsetzbar.
Im one-vs-all-Ansatz wird für jede Klasse ω_c ein Klassifikator mit Hypothesenfunktion $h_c(\boldsymbol{x})$, für $c = 1, \ldots, C$, trainiert, der diese Klasse von allen anderen Klassen trennt. Soll ein unbekanntes Objekt klassifiziert werden, werden alle Klassifikatoren ausgewertet und diejenige Klasse zugewiesen, deren Hypothesenfunktion den höchsten Wert annimmt, d. h.

$$\hat{\omega}(\boldsymbol{x}) = \arg\max_{\omega_c} h_c(\boldsymbol{x}). \tag{2.75}$$

Dieses Schema entspricht der Klassifikationsvorschrift aus Gleichung (2.49) in Abschnitt 2.4.3. Hier kann es außerdem sinnvoll sein, eine Rückweisungsoption einzubauen, falls die beiden sichersten Hypothesen h_{c_1}, h_{c_2} ähnliche Werte annehmen, also $h_{c_1}(\boldsymbol{x}) - h_{c_2}(\boldsymbol{x}) < \varepsilon$ oder falls keine Hypothese zugunsten der zugehörigen Klasse entscheidet, $h_c(\boldsymbol{x}) \leq 0$ für alle $c = 1, \ldots, C$.
Im one-vs-one-Ansatz wird ein Klassifikator $\hat{\omega}_k(\boldsymbol{x})$ für jedes Paar von Klassen trainiert. Die Anzahl der Klassifikatoren ist mit $K = \frac{1}{2} \cdot C \cdot (C-1)$

also deutlich höher als beim one-vs-all-Ansatz. Bei der Klassifikation eines unbekannten Objekts wird jeder Klassifikator ausgewertet und diejenige Klasse zugewiesen, die die meisten Stimmen auf sich vereint, d. h.

$$\hat{\omega}(\boldsymbol{x}) = \arg\max_{\omega_c} \sum_{k=1}^{K} \mathbb{1}[\hat{\omega}_k(\boldsymbol{x}) = \omega_c] \,. \tag{2.76}$$

Auch hier kann die Einführung einer Rückweisungsoption sinnvoll sein, wenn die beiden Klassen mit den meisten Stimmen eine ähnliche Anzahl an Stimmen erhalten.

2.4.5 Entscheidungsbäume

Neben der SVM sind Entscheidungsbäume und die darauf aufbauenden Random Forest Klassifikatoren wichtige Verfahren der Mustererkennung. Anders als SVMs können Entscheidungsbäume ohne Anpassungen wie problemspezifische Kernfunktionen mit nichtmetrischen Merkmalen oder einer Mischung aus metrischen und nichtmetrischen Merkmalen umgehen.
Anschaulich sind Entscheidungsbäume hierarchische Regelsysteme, in denen die nächste auszuwertende Regel von den Ergebnissen vorheriger Regeln abhängt. Die naheliegende Struktur solcher Systeme ist eine Baumstruktur, in der innere Knoten die Regeln, die ausgehenden Kanten jedes Knotens die möglichen Antworten und die Blattknoten das Klassifikationsergebnis bzw. die Entscheidung repräsentieren. Diese Strukturen werden daher auch Entscheidungsbäume genannt. Wenn jede Regel einer Ja/Nein-Frage entspricht, also jeder innere Knoten zwei Kindknoten besitzt, spricht man von einem binären Entscheidungsbaum. Da sich jeder Entscheidungsbaum in einen binären Entscheidungsbaum überführen lässt [BRN17], werden im Folgenden nur diese Strukturen betrachtet.
Prinzipiell ist es möglich, einen Entscheidungsbaum manuell festzulegen. Allerdings ist hier, wie bei den in Kapitel 1 beschriebenen Expertensystemen, die Wissensakquisition und die Handhabung großer Regelsysteme schwierig.

Eine naheliegende Frage ist daher, wie ein Entscheidungsbaum automatisch anhand einer Lernstichprobe $\mathcal{D}$ aufgestellt werden kann. Die wesentliche Schwierigkeit ist dabei die kombinatorische Vielfalt: Bei n binären Merkmalen und zwei möglichen Entscheidungen bzw. Klassen gibt es 2^{2^n} mögliche Entscheidungsbäume. Somit kommt ein Brute-Force-Ansatz nicht in Frage. Abhängig von den bereits verwendeten Regeln müssen aber nicht alle zur Verfügung stehenden Regeln in Betracht gezogen werden. Wenn z. B. ein Merkmal stark mit einem anderen Merkmal korreliert ist, müssen nur Regeln betrachtet werden, die eines dieser Merkmale beinhalten. Diese Beobachtung wird in einem Greedy-Ansatz ausgenutzt, indem der Baum sukzessive aus denjenigen Regeln aufgebaut wird, die den Raum der diskriminierenden Regeln am stärksten einschränken. Bevor dieser Algorithmus erklärt wird, soll der Entscheidungsbaum jedoch im Sinne der bisherigen Betrachtung geometrisch interpretiert werden.

Geometrische Interpretation. Der in Abschnitt 2.4.3 vorgestellte kNN-Klassifikator ist trotz seiner einfachen Klassifikationsvorschrift insbesondere mit großen Lernstichproben ein erstaunlich leistungsfähiger Klassifikator. Über die Anzahl der zu betrachtenden Nachbarn k kann die Komplexität des Klassifikators und damit der Bias-Varianz Trade-off gesteuert werden. Durch geeignete Wahl eines Distanzmaßes können zudem Merkmale beliebiger Skalen verwendet sowie Vorwissen über die Struktur des Klassifikationsproblems eingebracht werden. Ein erheblicher Nachteil ist jedoch, dass bei der Klassifikation eines Objekts die Distanz des zugehörigen Merkmalsvektors $\boldsymbol{x}$ zu allen Merkmalsvektoren der Lernstichprobe berechnet werden muss. Der Rechenaufwand ist somit linear im Umfang der Lernstichprobe $\mathcal{D}$. Wegen des Fluchs der Dimensionalität steigt allerdings mit der Dimension des Merkmalsraums auch der für eine fehlerarme Klassifikation benötigte Umfang der Lernstichprobe $\mathcal{D}$ und auch damit die für die Auswertung benötigte Rechenzeit stark an. Im Gegensatz dazu ist der Rechenaufwand mit der

logistischen Regression in $\mathcal{O}(D)$ und mit der SVM in $\mathcal{O}(S)$ mit der Anzahl der Support Vektoren S.

Da aber nur die nächsten k Nachbarn der Lernstichprobe relevant sind, kann auch der Rechenaufwand des kNN-Klassifikators durch Verwendung von geeigneten, vorab berechneten Datenstrukturen erheblich reduziert werden. Wie bei den anderen Klassifikatoren wird also ein Teil der zur Auswertung benötigten Rechenzeit in eine Lernphase verlagert. Eine naheliegende Datenstruktur ist eine Unterteilung des Merkmalsraums in K disjunkte Regionen $z_1, \ldots, z_k \subset \mathbb{M}$. Für einen gegebenen Merkmalsvektor $\boldsymbol{x} \in z_k$ müssen dann lediglich die Trainingsbeispiele in z_k und in den benachbarten Regionen betrachtet werden. Dieses Schema ist natürlich nur dann sinnvoll, wenn die Berechnung $\boldsymbol{x} \in z_k$ und die Abfrage benachbarter Regionen effizient berechenbar ist.

Aus der algorithmischen Geometrie sind eine Reihe von Techniken zur Lösung dieser Aufgabe bekannt. k-d-Bäume scheinen hier besonders gut geeignet, da sich diese Datenstruktur an die Verteilung der Daten anpasst und die Suche nach den k nächsten Nachbarn eines Punktes im Mittel mit $\mathcal{O}(k \log N)$ Operationen vollzogen werden kann [Ben75; FBF77]. Dies wird erreicht, indem der Merkmalsraum iterativ entlang achsenparalleler Hyperebenen $\boldsymbol{e}_q^\top \boldsymbol{x} - b = 0$ getrennt wird, wobei $\boldsymbol{e}_q$ der q-te Basisvektor von $\mathbb{M}$ ist, d. h.

$$\boldsymbol{e}_q = (0, \ldots, 0, \underbrace{1}_{q\text{-te Stelle}}, 0, \ldots, 0)^\top. \tag{2.77}$$

Bei k-d-Bäumen wird der Parameter b als der Median der $[\boldsymbol{x}_n]_q$, $\boldsymbol{x}_n \in \mathcal{D}$ gewählt. Somit teilt die Hyperebene die Stichprobe $\mathcal{D}$ in zwei Hälften

$$\mathcal{D}^- = \left\{ \boldsymbol{x}_n \in \mathcal{D} \,\middle|\, \boldsymbol{e}_q^\top \boldsymbol{x}_n - b < 0 \right\} \quad \text{und} \tag{2.78}$$

$$\mathcal{D}^+ = \left\{ \boldsymbol{x}_n \in \mathcal{D} \,\middle|\, \boldsymbol{e}_q^\top \boldsymbol{x}_n - b \geq 0 \right\}. \tag{2.79}$$

Die Unterteilung wird rekursiv mit den beiden Teildatensätzen fortgesetzt, bis alle Trainingsdaten verwendet wurden.
Ein so konstruierter k-d-Baum ist optimal bezüglich der Baumtiefe [Ben75], aber nicht notwendigerweise optimal für einen kNN-Klassifikator geeignet, da die Klasseninformation der $\boldsymbol{x}_n$ bei der Konstruktion nicht beachtet wurde. Da die Klassifikationsentscheidung von den k nächsten Nachbarn abhängt, entscheidet sich dieser Klassifikator in Regionen, in denen (fast) alle enthaltenen $\boldsymbol{x}_n \in \mathcal{D}$ der gleichen Klasse zugeordnet sind, (fast) immer für die gleiche Klasse. Die Aufteilung sollte also so erfolgen, dass die Hyperebenen die Klassenreinheit der entstehenden Regionen maximieren. Mit so einer Struktur kann die Suche nach den nächsten Nachbarn früher abgebrochen werden, was die Auswertung des Klassifikators weiter beschleunigt. Diese Überlegungen können weiter generalisiert werden und führen schließlich zum Entscheidungsbaum.

Verallgemeinerung. Formal ist ein binärer Entscheidungsbaum $\mathcal{T}$ ein gerichteter, azyklischer Graph mit inneren Knoten $\mathcal{K}$, Blattknoten $\mathcal{B}$ und Kanten $\mathcal{E}$. Die inneren Knoten $k \in \mathcal{K}$ sind mit Tests $h_k(\boldsymbol{x}) \in \{0, 1\}$ mit zwei Ausgängen assoziiert. Mit metrischen Merkmalen beruhen diese Tests oft auf Schwellwerten, d. h. $h_k(\boldsymbol{x}) = \mathbb{1}\left[[\boldsymbol{x}]_{q_k} < \tau_k\right]$, wobei q_k eine mit dem Knoten assozierte Dimension des Merkmalsraums und τ_k ein Schwellwert ist. Mit nichtmetrischen Merkmalen wird in der Regel auf Gleichheit oder Mengenzugehörigkeit geprüft, d. h. mit einer Menge $\mathcal{Q}_k$ an relevanten Merkmalsausprägungen ist $h_k(\boldsymbol{x}) = \mathbb{1}\left[[\boldsymbol{x}]_{q_k} \in \mathcal{Q}_k\right]$. Unabhängig vom verwendeten Test entsprechen die ausgehenden Kanten von k den möglichen Ausgängen von $h_k(\boldsymbol{x})$.
Die Blattknoten $b \in \mathcal{B}$ des Baumes sind schließlich mit Klassifikationshypothesen $h_b(\boldsymbol{x})$ assoziiert. Wie bei den $h_k(\boldsymbol{x})$ können hierfür beliebige Funktionen und insbesondere auch andere Klassifikatoren wie eine SVM angesetzt werden. Oft wird hierfür aber die konstante Zuweisung eines Klassenlabels verwendet, also $h_b(\boldsymbol{x}) = y_b$. Alternativ kann für eine an-

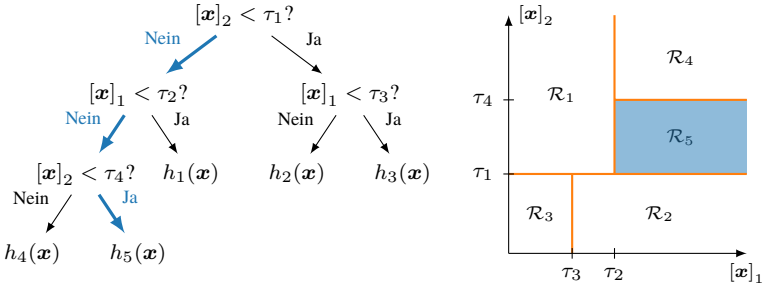


Abbildung 2.10: Beispiel eines Entscheidungsbaums und die dazugehörige Partitionierung des Merkmalsraums.

schließende MAP-Klassifikation eine Dichteschätzung oder eine a-posteriori Wahrscheinlichkeit hinterlegt werden, d. h. $\boldsymbol{h}_b(\boldsymbol{x}) = (q_b(\omega_1), \ldots, q_b(\omega_C))^\top$ mit

$$q_b(\omega) := p_b(\boldsymbol{x} \,|\, \omega) \quad \text{bzw.} \quad q_b(\omega) := P_b(\omega \,|\, \boldsymbol{x})\,. \tag{2.80}$$

Gleichzeitig steht jeder Blattknoten für eine Entscheidungsregion $z_b \subseteq \mathbb{M}$ im Merkmalsraum. Da die Entscheidungsregionen $z_b, b \in \mathcal{B}$ disjunkt sind, ergibt sich die Hypothesenfunktion bzw. Dichteschätzung des gesamten Baumes formal als

$$h(\boldsymbol{x}) = \sum_{b \in \mathcal{B}} \mathbb{1}[\boldsymbol{x} \in \mathcal{R}_b]\, h_b(\boldsymbol{x}). \tag{2.81}$$

Mit der Klassifikationshypothese $h_b(\boldsymbol{x}) = y_b$ approximiert ein Entscheidungsbaum somit die Entscheidungsgrenzen eines kNN-Klassifikators. In der Praxis werden natürlich nicht alle Blätter ausgewertet, sondern anhand der $h_k(\boldsymbol{x})$ der inneren Knoten nur ein Pfad durch den Baum verfolgt.
Ein Beispiel eines binären Entscheidungsbaum und die zugehörige Partitionierung des Merkmalsraums ist in Abbildung 2.10 zu sehen. Die orangefarbenen Linien deuten die Grenzen der Entscheidungsregionen $\mathcal{R}_b$ an. Das grün

hinterlegte Entscheidungsregion $\mathcal{R}_4$ entspricht dem grün markierten Pfad durch den Entscheidungsbaum.

Entscheidungsbaumtraining. Der Algorithmus zur Festlegung der Tests und Struktur eines Entscheidungsbaums anhand einer Stichprobe enthält zwei wesentliche Schritte. Erstens müssen die Tests der inneren Knoten $h_k(\boldsymbol{x})$ und zweitens müssen die Hypothesenfunktionen der Blattknoten $h_b(\boldsymbol{x})$ festgelegt werden. Der erste Schritt ähnelt der rekursiven Konstruktion eines k-d-Baums, jedoch mit der oben beschriebenen Einbeziehung der Klasseninformation und den Verallgemeinerungen der $h_k(\boldsymbol{x})$. Ausgehend von einer Menge $\mathcal{H}$ an Testfunktionen wird in jedem rekursiven Schritt derjenige Test $h_k(\boldsymbol{x})$ ausgewählt, der die Lernstichprobe in maximal klassenreine Teilstichproben $\mathcal{D}^-$ und $\mathcal{D}^+$ aufteilt. Die Rekursion wird beendet, wenn sich durch die Unterteilung die Klassenreinheit nicht mehr signifikant erhöht, der Umfang der Lernstichprobe eine Grenze unterschreitet oder der Baum eine vordefinierte Tiefe erreicht.
Die Klassenreinheit wird dabei mittels eines Unreinheitsmaßes $i(\mathcal{D})$ gemessen. Dieses Unreinheitsmaß wird minimal, wenn der Datensatz klassenrein ist und maximal, wenn $\mathcal{D}$ gleich viele Beispiele aller Klassen enthält. Hierfür haben sich drei Heuristiken bewährt [DHS01]: Gini-impurity i_G, das Entropiemaß i_H und das Fehlklassifikationsmaß i_M,

$$i_G(\mathcal{D}) = \sum_{c \neq e} f_c f_e = \sum_{c=1}^{C} f_c(1 - f_c) = 1 - \sum_{c=1}^{C} f_c^2, \tag{2.82}$$

$$i_H(\mathcal{D}) = -\frac{1}{2} \sum_{c=1}^{C} f_c \log f_c \quad \text{und} \tag{2.83}$$

$$i_M(\mathcal{D}) = \frac{1}{|\mathcal{D}|} \sum_{\boldsymbol{x}_n \in \mathcal{D}} \mathbb{1}\left[y_n \neq \underset{c=1,\ldots,C}{\arg\max} f_c \right], \tag{2.84}$$

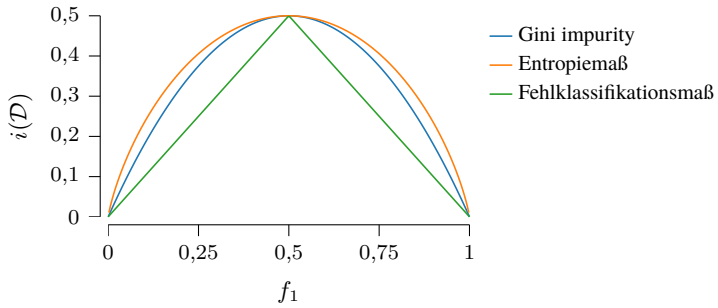


Abbildung 2.11: Unreinheitsmaße für zwei Klassen in Abhängigkeit der Zusammensetzung der Lernstichprobe. f_1 ist der Anteil der Beispiele der Klasse ω_1 in $\mathcal{D}$ (siehe Text).

In allen drei Fällen ist

$$f_c = \frac{1}{|\mathcal{D}|} \sum_{\boldsymbol{x}_n \in \mathcal{D}} \mathbb{1}[y_n = c] \tag{2.85}$$

der Anteil der Beispiele der Klasse ω_c in $\mathcal{D}$. Abbildung 2.11 zeigt diese Maße für $C = 2$ Klassen. In jeder Iteration des Trainings wird derjenige Test ausgewählt, der die Gesamtunreinheit der Teilstichproben $\mathcal{D}^+$ und $\mathcal{D}^-$ minimiert, also

$$h_k = \arg\min_{h \in \mathcal{H}} \Big(\overbrace{i\big(\mathcal{D}^+\big) + i\big(\mathcal{D}^-\big)}^{=: i_{\text{ges}}(\mathcal{D})} \Big) \quad \text{mit} \tag{2.86}$$

$$\mathcal{D}^+ = \{\boldsymbol{x} \in \mathcal{D} \,|\, h(\boldsymbol{x}) = 1\} \quad \text{und} \quad \mathcal{D}^- = \{\boldsymbol{x} \in \mathcal{D} \,|\, h(\boldsymbol{x}) = 0\}\,. \tag{2.87}$$

Im Prinzip lässt die Menge der binären Tests $\mathcal{H}$ beliebige binäre Entscheidungsfunktionen zu, die beispielsweise durch logistische Regression oder SVM festgelegt werden können. In diesem Sinne kann ein Entscheidungsbaum als Meta-Klassifikator angesehen werden. Mit metrischen Merkmalen

beruhen die Tests in Anlehnung an k-d-Bäume aber wie oben beschrieben oft auf achsenparallelen Hyperebenen, d. h.

$$\mathcal{H} := \left\{ \boldsymbol{e}_d^\top \boldsymbol{x} = [\boldsymbol{x}]_d < \tau \,\middle|\, d = 1, \ldots, D \text{ und } \tau \in \mathbb{R} \right\}. \tag{2.88}$$

Mit solchen Tests reduziert sich die Suche nach dem optimalen $h \in \mathcal{H}$ auf die Suche nach einem optimalen Schwellwert τ für jede Dimension d des Merkmalsraums. Hierfür gibt es mehrere Ansätze, siehe z. B. die Arbeit von Bayer [Bay16]. In der Praxis hat sich die Quantilsheuristik bewährt. Hier wird für jede Dimension $1 \leq d \leq D$ der Schwellwert τ aus der Menge der $\frac{1}{Q}$-Quantile der $[\boldsymbol{x}]_d$ so gewählt, dass die Gesamtunreinheit $i_{\text{ges}}([\mathcal{D}]_d)$ (siehe Gleichung 2.86) minimiert wird. Damit muss das Unreinheitsmaß pro Iteration $D \cdot Q$ mal ausgewertet werden, was auch bei hochdimensionalen Merkmalsräumen einen vertretbaren Aufwand bedeutet.
Die Klassifikationshypothese jedes Blattknotens b wird schließlich aus den Trainingsdaten $\mathcal{D}_b$, die das Blatt erreichen, abgeleitet. Oft wird die am häufigsten in $\mathcal{D}$ vorkommende Klasse ermittelt und das zugehörige Klassenlabel als konstante Hypothese verwendet. Eine Dichteschätzung wie in Gleichung (2.80) wird in der Regel durch Maximum-Likelihood-Schätzung ermittelt, also z. B. für die a-posteriori Klassenwahrscheinlichkeit

$$q_b(\omega) = \frac{|\{\boldsymbol{x}_n \in \mathcal{D}_b \,|\, \omega_n = \omega\}|}{|\mathcal{D}_b|}. \tag{2.89}$$

Der vollständige Trainingsalgorithmus ist in Algorithmus 2 gezeigt. Hier ist die Baumstruktur implizit durch die rekursiven Aufrufe und die verschachtelten Listen gegeben. Die Entscheidungsfunktion der Blattknoten $h_b(\boldsymbol{x})$ wird mit der Funktion LearnLeaf($\mathcal{D}$) festgelegt.
Ein wesentlicher Nachteil von Entscheidungsbäumen ist die hohe Varianz des Klassifikators: Entscheidungsbäume neigen zum Overfitting [HTF09]. Dieser Nachteil kann durch Zurückstutzen des Baums nach dem Training ausgeglichen werden, siehe z. B. Duda et al. [DHS01]. Alternativ kann die

Algorithmus 2 Entscheidungsbaumtraining.

function LEARN($\mathcal{D}, \mathcal{H}, d, N_{\min}, g_{\min}$)
  if $d \leq 0$ **or** $|\mathcal{D}| \leq N_{\min}$ **then**
    return LEARNLEAF($\mathcal{D}$)
  $h \leftarrow \arg\min_{h \in \mathcal{H}} i_{\text{ges}}(\mathcal{D})$
  $\mathcal{D}_t \leftarrow \{\boldsymbol{x} \in \mathcal{D} \mid h(\boldsymbol{x}) = 1\}$
  $\mathcal{D}_f \leftarrow \{\boldsymbol{x} \in \mathcal{D} \mid h(\boldsymbol{x}) = 0\}$
  if $i(\mathcal{D}) - (i(\mathcal{D}_t) + i(\mathcal{D}_f)) < g_{\min}$ **then**
    return LEARNLEAF($\mathcal{D}$) ▷ Datensatz genügend klassenrein
  $l_t \leftarrow$ LEARN $(\mathcal{D}_t, \mathcal{H}, d-1, N_{\min}, g_{\min})$
  $l_f \leftarrow$ LEARN $(\mathcal{D}_f, \mathcal{H}, d-1, N_{\min}, g_{\min})$
  return (h, l_t, l_f)

Varianz des Klassifikators durch Mittelung über ein Ensemble aus mehreren Entscheidungsbäumen reduziert werden. In den nächsten zwei Abschnitten werden zwei Techniken vorgestellt, um dies zu erreichen: Random Forests und Boosting.

2.4.6 Random Forests

Die von Breiman [Bre01] vorgeschlagenen Random Forests (RFs) sind, wie der Name suggeriert, ein Ensemble aus randomisierten Entscheidungsbäumen. Die Randomisierung wird durch zwei Mechanismen realisiert: Bagging (engl.: **b**ootstrap **agg**regat**ing**) und zufällige Unterabtastung des Merkmalsraums während des Entscheidungsbaumtrainings.

Bagging bezeichnet ein Verfahren, in dem jeder Klassifikator eines Ensembles mit Hilfe einer zufälligen Untermenge der Trainingsdaten trainiert wird (bootstrapping) und in der Betriebsphase das Klassifikationsergebnis aller Klassifikatoren zusammengefasst (aggregiert) wird. Bootstrapping bezeichnet dabei ein statistisches Verfahren zur Bestimmung der Genauigkeit einer Schätzung auf Grundlage einer Stichprobe mit unbekannter Verteilung, in dem der Schätzwert wiederholt anhand von Stichproben berechnet wird,

die durch zufälliges Ziehen mit Zurücklegen aus der originalen Stichprobe generiert werden [Efr79].
Im Kontext der RFs wird Bootstrapping verwendet, indem das Ensemble aus T Entscheidungsbäumen anhand von zufällig generierten Lernstichproben, den Bootstrap Samples $\tilde{\mathcal{D}}_t = \{\boldsymbol{x}_{r_t(1)}, \ldots, \boldsymbol{x}_{r_t(B)}\}$, $t = 1, \ldots, T$ trainiert wird. Hierbei ist $B \leq N$ die Größe der Bootstrap Samples, typischerweise $B \approx 0.7N$, und $r_t(b)$ eine zufällige, nicht notwendigerweise surjektive Abbildung mit $1 \leq r_t(b) \leq N$. Insbesondere kann $r_t(b_1) = r_t(b_2)$ für $b_1 \neq b_2$ gelten und mithin ein Merkmalsvektor $\boldsymbol{x}_n$ mehrfach in einem Bootstrap Sample $\tilde{\mathcal{D}}_t$ enthalten sein.
Bagging nutzt effektiv die hohe Varianz der Entscheidungsbäume: Jeder Entscheidungsbaum sieht einen anderen Teil des Klassifikationsproblems, findet andere Entscheidungsgrenzen und verursacht unterschiedliche Klassifikationsfehler. Diese (stochastischen) Klassifikationsfehler werden anschließend durch die Aggregation der Entscheidungen des Ensembles wieder korrigiert [HTF09].
Für sich genommen bewirkt Bagging zwar eine Diversifizierung des Ensembles, aber dennoch werden in den ersten Ebenen der Bäume mit hoher Wahrscheinlich die gleichen oder zumindest ähnliche Testfunktionen verwendet, da die Unreinheit der anfänglichen Bootstrap Samples durch diese Tests mit hoher Wahrscheinlichkeit am stärksten reduziert wird. Die Unterschiede der Bootstrap Samples werden erst in den späteren Iterationen des Lernalgorithmus relevant. Um dieses Problem zu beheben, wird bei der Wahl der Testfunktionen in Gleichung (2.86) nicht ganz $\mathbb{M}$, sondern nur ein zufällig gewählter Unterraum von $\mathbb{M}$ betrachtet. Mit den kanonischen Testfunktionen aus Gleichung (2.88) wird statt der Menge aller D möglichen Merkmale nur eine zufällige Untermenge von $D' \leq D$ Merkmalen, typischerweise $D' \approx \sqrt{D}$, ausgewertet. Die Einschränkung des Merkmalsraums wird bei jeder Iteration des Trainingsalgorithmus neu gewählt.

Es gibt mehrere Optionen, um die Klassifikationsentscheidung aus den Entscheidungsbäumen abzuleiten. Wie bei dem one-vs-all-Ansatz der Mehrklassen-SVM kann ein Mehrheitsentscheid herangezogen werden,

$$h(\boldsymbol{x}) = \arg\max_{y} \sum_{t=1}^{T} \mathbb{1}[h_t(\boldsymbol{x}) = y]\,, \tag{2.90}$$

wobei die $h_t(\boldsymbol{x})$ die Hypothesenfunktionen der Bäume des Ensembles sind. Alternativ können a-posteriori Klassenwahrscheinlichkeiten geschätzt werden,

$$\hat{P}(\omega_c \,|\, \boldsymbol{x}) = \frac{1}{T} \sum_{t=1}^{T} \mathbb{1}[h_t(\boldsymbol{x}) = c]\,, \quad c = 1, \ldots, C. \tag{2.91}$$

Wenn die einzelnen Bäume bereits eine Schätzung der Klassenwahrscheinlichkeit liefern, ist es gängig, über diese Schätzungen zu mitteln, d. h.

$$\hat{P}(\omega_c \,|\, \boldsymbol{x}) = \frac{1}{T} \sum_{t=1}^{T} \hat{P}_t(\omega_c \,|\, \boldsymbol{x})\,. \tag{2.92}$$

Wie bereits erwähnt führt Bagging zur Diversifizierung des Ensembles. Man könnte nun schließen, dass dadurch die Varianz des RF-Klassifikators erhöht wird, doch das Gegenteil ist der Fall. In der Tat kann gezeigt werden, dass die Kombination aus Bagging und Unterabtastung des Merkmalsraums die Varianz des Klassifikators reduziert, ohne dabei den Bias zu erhöhen [HTF09]. Dieses Ergebnis und die Tatsache, dass RFs bei vielen praktischen Problemen bereits mit Standardparametern sehr gute Ergebnisse erzielen, haben diese Klassifikatoren in der Praxis sehr beliebt gemacht. Ein weiterer Vorteil von RFs ist, dass sowohl das Training als auch die Klassifikation sehr einfach parallelisierbar sind, indem jeder Baum getrennt ausgewertet wird.
Neben der Klassifikation werden RFs auch zur eingebetteten Merkmalsselektion (siehe Abschnitt 2.2.2) verwendet. Dazu wird die Relevanz der Merkmale für jeden Baum des Ensembles anhand der Lerndaten bewertet,

die nicht in den Bootstrap Samples zum Training verwendet wurden. Die Relevanz der Merkmale in Bezug auf das Ensemble wird schließlich wieder durch Mittelung bewertet. Details dieser Methode werden z. B. von Breiman [Bre01] erläutert.

2.4.7 Boosting

Random Forests sind ein Beispiel für sogenannte Ensemblemethoden, bei denen eine Entscheidung von einem Ensemble, anstatt von einem einzelnen Klassifikator getroffen wird. Dieses Vorgehen ähnelt dabei der Strategie, die Meinung mehrerer Experten einzuholen, um sich ein Urteil zu bilden. In diesem Abschnitt wird eine weitere, nicht auf Entscheidungsbäume beschränkte Ensemblemethode vorgestellt: Boosting. Zunächst soll aber kurz skizziert werden, warum ein solches Vorgehen theoretisch gerechtfertigt ist.
Bei der MAP-Klassifikation wird anhand der a-posteriori Wahrscheinlichkeit $P(\omega \mid \boldsymbol{x})$ entschieden. Dabei liegt immer ein explizites oder implizites Modell M zugrunde. Da dieses Modell anhand einer zufälligen Stichprobe festgelegt wird, muss es streng genommen ebenfalls als eine Zufallsvariable angesehen werden und korrekterweise müsste die Klassenwahrscheinlichkeit als $P(\omega \mid M, \boldsymbol{x})$ notiert werden. Die wahre, von M unabhängige a-posteriori Klassenwahrscheinlichkeit wird dann durch Marginalisierung über alle möglichen Modelle gewonnen, d. h. mit der Menge $\mathcal{M}$ aller möglichen Modelle

$$P(\omega \mid \boldsymbol{x}) = \int_{\mathcal{M}} P(\omega \mid M,\boldsymbol{x})\, p(M \mid \boldsymbol{x})\ \mathrm{d}M. \tag{2.93}$$

Natürlich ist diese Marginalisierung nicht berechenbar, da hier eine unendliche Anzahl an Modellen ausgewertet werden müsste. Ähnlich wie bei ERM (siehe

Abschnitt 2.4.3) kann das Integral aber mit einer endlichen Menge an Modellen approximiert werden,

$$P(\omega \mid \boldsymbol{x}) \approx \sum_{t=1}^{T} P(\omega \mid M_t, \boldsymbol{x})\, P(M_t \mid \boldsymbol{x})\,. \tag{2.94}$$

Bei der Klassifikation mit dieser a-posteriori Wahrscheinlichkeit ergibt sich die endgültige Klassifikationsentscheidung also aus der gewichteten Mittelung der verschiedenen Modelle. Dieses Vorgehen ist nicht auf MAP-Klassifikation beschränkt, sondern kann unter der Annahme $P(M_t \mid \boldsymbol{x}) = \text{const}$ auch mit Hypothesenfunktionen h_t verwendet werden. Hier ergibt sich die Gesamthypothese H als gewichtete Summe der h_t,

$$H(\boldsymbol{x}) = \sum_{t=1}^{T} \alpha_t h_t(\boldsymbol{x}) \tag{2.95}$$

mit Gewichten $\alpha_t \in \mathbb{R}$. Die Gesamtheit der Modelle M_t bzw. Hypothesen $h_t(\boldsymbol{x})$ mit $t = 1, \ldots, T$ wird dabei auch ein Ensemble genannt. Verfahren, die über ein Ensemble mitteln, heißen Ensemblemethoden.
Ein linearer Klassifikator mit der Hypothesenfunktion $h(\boldsymbol{x}) = \boldsymbol{\alpha}^\top \boldsymbol{x} - \tau$ kann als Ensemble von $T = D$ Hypothesenfunktionen $h_t(\boldsymbol{x}) = \boldsymbol{e}_t^\top \boldsymbol{x} - \tau_k$, also Vergleiche eines einzelnen Merkmals mit einem Schwellwert, aufgefasst werden. Die Gesamthypothese des Ensembles ergibt sich als

$$H(\boldsymbol{x}) = \sum_{t=1}^{D} \alpha_t h_t(\boldsymbol{x}) = \sum_{t=1}^{D} \alpha_t \left(\boldsymbol{e}_t^\top \boldsymbol{x} - \tau_t\right) \tag{2.96}$$

$$= \left(\sum_{t=1}^{D} \alpha_t \boldsymbol{e}_t^\top\right) \boldsymbol{x} - \sum_{t=1}^{D} \alpha_t \tau_t = \boldsymbol{\alpha}^\top \boldsymbol{x} - \tau, \tag{2.97}$$

wobei die Gewichte α_t im Vektor $\boldsymbol{\alpha}$ zusammengefasst wurden.
Dieses Beispiel zeigt die zwei wesentlichen Freiheitsgrade von Ensemblemethoden: die Wahl der Hypothesen $h_t(\boldsymbol{x})$ und die Wahl der Gewichte. Bei

Random Forests waren die Hypothesen durch Entscheidungsbäume gegeben, deren Entscheidungen gleich stark gewichtet wurden. Boosting weicht diese Einschränkungen auf, indem beliebige Klassifikatoren zugelassen werden, deren Gewichte beim Training festgelegt werden. Die einzige Voraussetzung an die Klassifikatoren ist, dass diese besser klassifizieren als ein Zufallsentscheid. Im Kontext des Boostings werden Klassifikatoren, die diese Bedingung gerade so erfüllen, schwache Klassifikatoren genannt. Boosting vereint viele schwache Klassifikatoren zu einem starken Klassifikator.

AdaBoost. Eines der bekanntesten Boostingverfahren ist AdaBoost (engl.: **ada**ptive **boost**ing), welches von Freund und Schapire [FS96] vorgestellt wurde. Zwar wurde AdaBoost von Freund und Schapire bereits zur Klassifikation mehrerer Klassen formuliert und es existieren weitere Formulierungen mit dem gleichen Ziel (siehe z. B. Mukherjee und Schapire [MS11]), doch zur Vereinfachung der Notation soll hier nur auf die binäre Klassifikation eingegangen werden. Wie bei der SVM wird dabei die Klasse ω_1 wird mit $y = 1$ und die Klasse ω_2 mit $y = -1$ kodiert.
AdaBoost erstellt einen starken Klassifikator, indem in mehreren Runden jeweils derjenige schwache Klassifikator $h_t \in \mathcal{H}$ zum Ensemble hinzugefügt wird, der den gewichteten Klassifikationsfehler auf der Lernstichprobe minimiert. Das Gewicht α_t des Klassifikators h_t wird anhand dieses Klassifikationsfehlers berechnet. Anschließend wird die Gewichtung der Lernstichprobe so angepasst, dass falsch klassifizierte Beispiele höher gewichtet werden als richtig klassifizierte Beispiele. Somit wird die Wahrscheinlichkeit erhöht, in der nächsten Selektionsrunde einen Klassifikator auszuwählen, der diese Beispiele richtig klassifiziert. Das Training wird abgebrochen, wenn der selektierte schwache Klassifikator nicht besser klassifiziert als ein Zufallsentscheid oder wenn eine maximale Anzahl an Iterationen erreicht ist. Der Trainingsalgorithmus in der Version nach Friedman et al. [FHT00] ist in Algorithmus 3 gezeigt.

Algorithmus 3 Diskreter AdaBoost Algorithmus.

1: **function** AdaBoost($\mathcal{D}, \mathcal{H}$)
2: $\quad w_{1,n} \leftarrow \frac{1}{N}$ **for** $\boldsymbol{x}_n \in \mathcal{D}$
3: $\quad$ **for** $t = 1, \ldots, T$ **do**
4: $\quad\quad h_t = \arg\min_{h \in \mathcal{H}} \varepsilon_t = \arg\min_{h \in \mathcal{H}} \sum_{\boldsymbol{x}_n \in \mathcal{D}} w_{t,n} \mathbb{1}[h(\boldsymbol{x}_n) \neq y_n]$
5: $\quad\quad$ **if** $\varepsilon_t \geq \frac{1}{2}$ **then break** $\quad \triangleright\ h_t$ ist schlechter als Zufallsentscheid
6: $\quad\quad \alpha_t \leftarrow \log \frac{1 - \varepsilon_t}{\varepsilon_t}$
7: $\quad\quad w_{t+1,n} \leftarrow \frac{w_{t,n}}{\sum_k w_{t+1,k}} \exp\left(\alpha_t \mathbb{1}[h_t(\boldsymbol{x}_n) \neq y_n]\right)$ **for** $\boldsymbol{x}_n \in \mathcal{D}$
8: $\quad$ **return** $H(\boldsymbol{x}) := \operatorname{sign}\left(\sum_t \alpha_t h_t(\boldsymbol{x})\right)$

Die wesentlichen Freiheitsgrade des AdaBoost Algorithmus sind die Anzahl der Iterationen T als obere Schranke für die Größe des Ensembles und die Art bzw. die Menge der schwachen Klassifikatoren $\mathcal{H}$. Diese Menge kann eine Klasse parametrischer Klassifikatoren repräsentieren, beispielsweise die Menge aller SVMs mit beliebigen, aber festen Hyperparametern. In diesem Fall würde die Klassifikatorauswahl in Zeile 4 des Algorithmus 3 durch Training mit individuellen Hyperparametern $C_n := C\, w_{t,n}$ für jedes $\boldsymbol{x}_n \in \mathcal{D}$ ersetzt. Ebenso könnte $\mathcal{H}$ die Menge aller Entscheidungsbäume sein. Hierbei müsste die Gewichtung der Lernstichprobe bei der Berechnung der Unreinheitsmaße beachtet werden. Der starke Klassifikator wäre dann wie ein Random Forest ein Ensemble von Entscheidungsbäumen. In der Tat zieht Breiman starke Parallelen zwischen Boosting und Random Forests: Unter bestimmten Voraussetzungen lässt sich AdaBoost als ein Random Forest interpretieren [Bre01].
Im einfachsten Fall ist $\mathcal{H}$ aber eine endliche Menge an Klassifikatorkandidaten. In diesem Fall können die $h \in \mathcal{H}$ als binäre Merkmale und AdaBoost als eingebettete Methode zur Merkmalsselektion (siehe Abschnitt 2.2.2)

aufgefasst werden. Die Gewichte α_t des Ensembles geben dabei einen Hinweis auf die Relevanz eines Merkmals. Außerdem kann der Algorithmus beschleunigt werden, indem eine Indikatormatrix

$$\boldsymbol{E} := \begin{pmatrix} e_{1,1} & \cdots & e_{1,N} \\ \vdots & \ddots & \vdots \\ e_{|\mathcal{H}|,1} & \cdots & e_{|\mathcal{H}|,N} \end{pmatrix} \quad \text{mit} \quad e_{k,n} = \mathbb{1}[h_k(\boldsymbol{x}_n) = y_n] \tag{2.98}$$

vorberechnet wird. Mit $\boldsymbol{w}_i := (w_{i,1}, \ldots, w_{i,N})^\top$ reduziert sich Zeile 4 des Algorithmus 3 auf die Suche nach der kleinsten Komponente von $\boldsymbol{\varepsilon} = \boldsymbol{E}\boldsymbol{w}$. Auch ohne diese Optimierung ist AdaBoost ein sehr schneller, einfach zu implementierender und gut interpretierbarer Algorithmus. AdaBoost besitzt bis auf die maximale Anzahl der Iterationen T und die Basisklassifikatoren $\mathcal{H}$ keine einzustellenden Hyperparameter und kann durch geeignete Wahl von $\mathcal{H}$ auf eine Vielzahl von Aufgabenstellungen angewendet werden. Beispielsweise stützen sich die Objektdetektionsverfahren von Viola und Jones [VJ01] und von Dollár et al. [Dol+09] (siehe Abschnitt 2.2.4) auf AdaBoost. AdaBoost zeigt im Allgemeinen eine gute Generalisierungsfähigkeit und kann, wenn die Hypothesenmenge $\mathcal{H}$ hinreichend simpel ist, Klassifikatoren erzeugen, die der menschlichen Interpretation zugänglich sind.
Allerdings neigt AdaBoost bei zu diskriminativen schwachen Klassifikatoren $h \in \mathcal{H}$ zu Overfitting, da die Gewichtung falsch klassifizierter Beispiele den Lernalgorithmus sehr anfällig gegenüber Ausreißern im Datensatz macht. Das kann theoretisch begründet werden: Friedman et al. [FHT00] zeigen, dass AdaBoost ein additives logistisches Regressionsmodell aufstellt, also den Logit (siehe Gleichung 2.33) durch ein additives Modell approximiert,

$$\operatorname{Logit}(\omega \mid \boldsymbol{x}) \approx \sum_{t=1}^{T} h_t(\boldsymbol{x}). \tag{2.99}$$

Mit dieser Sichtweise kann gezeigt werden, dass AdaBoost den exponentiellen Verlust

$$\mathfrak{R}(H) = \mathbb{E}_{\boldsymbol{x},y}\left\{\exp\left(-y\,H(\boldsymbol{x})\right)\right\} \tag{2.100}$$

minimiert. Dieses Fehlermaß gewichtet falsch klassifizierte Ausreißer sehr hoch. Dieses Verhalten kann aber auch genutzt werden, um Ausreißer in einem Datensatz zu erkennen, indem nach erfolgtem Training die Gewichte $w_{T,n}$ inspiziert werden.

Gradient Boosting. Die Interpretation von AdaBoost als additives Modell zur Fehlerminimierung motiviert eine weitere Sichtweise auf Boosting. Wie oben beschrieben sind die schwachen Klassifikatoren oft parametrische Funktionen, d. h. $\mathcal{H} = \{h(\boldsymbol{x};\boldsymbol{\theta}_t) \,|\, \boldsymbol{\theta}_t \in \Theta\}$. Somit ist der starke Klassifikator

$$H(\boldsymbol{x};\boldsymbol{\theta}_G) = \sum_{t=1}^{T} \alpha_t h(\boldsymbol{x};\boldsymbol{\theta}_t) \tag{2.101}$$

durch $\boldsymbol{\theta}_G = \left(\boldsymbol{\theta}_1^\top, \ldots, \boldsymbol{\theta}_T^\top\right)^\top$ parametriert. Nach dem ERM-Prinzip sollen diese Parameter nun so bestimmt werden, dass der Erwartungswert eines Fehlermaßes minimiert wird,

$$\boldsymbol{\theta}_G^* = \arg\min_{\boldsymbol{\theta}_G} \mathfrak{J}(\boldsymbol{\theta}_G) = \arg\min_{\boldsymbol{\theta}_G} \mathbb{E}_{\boldsymbol{x},y}\left\{l\left(H(\boldsymbol{x};\boldsymbol{\theta}_G), y\right)\right\}. \tag{2.102}$$

Diese Optimierung kann beispielsweise numerisch mittels Gradientenabstieg (siehe Abschnitt 2.3) durchgeführt werden.
Friedman [Fri01] überträgt das Konzept des Gradientenabstiegs zur Optimierung im Parameterraum Θ in den Raum der Funktionen: Anstatt eines Parameters $\boldsymbol{\theta}^*$ wird die Funktion $H^*(\boldsymbol{x})$ gesucht, die den erwarteten Fehler $\mathbb{E}_{\boldsymbol{x},y}\left\{l\left(H(\boldsymbol{x}), y\right)\right\}$ direkt, also ohne Umweg über eine Parametrierung, minimiert. Analog zur numerischen Optimierung wird H^* schrittweise durch

$H_T(\boldsymbol{x}) = \sum_{t=0}^{T} \delta_t(\boldsymbol{x})$ mit initialer Schätzung $\delta_0(\boldsymbol{x})$ angenähert. Die Schritte $\delta_t(\boldsymbol{x})$ sind dabei durch einen Gradientenabstieg im Funktionenraum gegeben,

$$\delta_t(\boldsymbol{x}) = -\alpha_t g_t(\boldsymbol{x}) = -\alpha_t \nabla_H \mathfrak{J}(H_{t-1}(\boldsymbol{x})) \quad \text{mit} \tag{2.103}$$

$$\alpha_t = \arg\min_{\alpha} \mathfrak{J}(H_{t-1}(\boldsymbol{x}) - \alpha g_t(\boldsymbol{x}))\,. \tag{2.104}$$

In der Praxis kann $l\left(H(\boldsymbol{x}), y\right)$ nur an den Datenpunkten der Lernstichprobe evaluiert und damit die Lösung von Gleichung (2.102) nur angenähert werden. Dafür wird der Gradient an den $\boldsymbol{x}_n \in \mathcal{D}$ ausgewertet und dasjenige $h_t \in \mathcal{H}$ gesucht, dass den quadratischen Abstand an diesen Stellen minimiert,

$$h_t = \arg\min_{h \in \mathcal{H}, \beta} \sum_{\boldsymbol{x}_n \in \mathcal{D}} \left(\tilde{y}_n - \beta h(\boldsymbol{x}_n)\right)^2 \quad \text{mit} \tag{2.105}$$

$$\tilde{y}_n := -\nabla_H \mathfrak{J}(H_{t-1}(\boldsymbol{x}_n)). \tag{2.106}$$

Alternativ könnte durch Verwendung von Glattheitsannahmen über $\mathcal{H}$ der erwartete Verlust in Gleichung (2.102) direkt minimiert werden. Dieser Ansatz ist jedoch deutlich rechenaufwendiger als der Ansatz in Gleichung (2.105) [Fri01].

Dieses sehr allgemeine Gradient Boosting Verfahren wird von Friedman auf verschiedene Arten spezialisiert [Fri01]. Besonderen Raum nimmt dabei $\mathcal{H}$ als Menge von Regressions- bzw. Entscheidungsbäumen ein. Wie bei Random Forests bestehen die Ensembles hier aus verschiedenen Entscheidungsbäumen. Diese sind allerdings nicht voneinander unabhängig, sondern vielmehr aufeinander abgestimmt, sodass sich später hinzugefügte Bäume vor allem auf falsch klassifizierte Beispiele konzentrieren, also Beispiele $\boldsymbol{x}_n$ mit großen $\tilde{y}_n$. Gradient Boosting wird in Abschnitt 3.3 erneut eine Rolle spielen.

2.5 Klassifikatorbewertung

Jedes Klassifikationssystem hängt von einer Reihe von Hyperparametern ab, die nicht anhand der Lernstichprobe festgelegt werden. Werden z. B. Bag of Visual Words Merkmale und eine lineare SVM eingesetzt, müssen die lokalen Deskriptoren und eventuelle Hyperparameter, die Größe des Vokabulars, Hyperparameter des Clusteralgorithmus, sowie der SVM-Parameter C festgelegt werden. Ebenso könnte zwischen verschiedenen Klassifikatoren wie einer SVM und einem Random Forest abgewägt werden. Um zu entscheiden, welche Hyperparameter bzw. Komponenten schlussendlich verwendet werden sollen, müssen die verschiedenen Konfigurationen quantitativ miteinander verglichen werden. In diesem Abschnitt wird deswegen auf verschiedene Bewertungsmaße eingegangen.

Diese Bewertungsmaße können anhand der Lernstichprobe $\mathcal{D}$ berechnet werden. Allerdings sagt dieses Vorgehen nichts über die Generalisierungsfähigkeit des Klassifikationssystems aus. Wie schon beim Bias-Varianz Trade-off (Abschnitt 2.4) erwähnt wurde, werden mit ausreichend komplexen Klassifikatoren auf der Lernstichprobe gute Werte erzielt, da in den meisten Lernverfahren die Minimierung eines Fehlermaßes explizites Ziel ist. Auf ungesehenen Daten ist hingegen eine schlechtere Klassifikationsgüte zu erwarten. Die Generalisierungsfähigkeit muss daher mit Hilfe einer nicht für das Training verwendeten Teststichprobe $\mathcal{T}$ evaluiert werden.

Ein wichtiges, naheliegendes Maß zur Bewertung von Klassifikatoren ist die Fehlerwahrscheinlichkeit P_e aus Gleichung (2.30) oder allgemeiner die erwarteten Kosten $\mathfrak{R}(\hat{\omega})$. In der Praxis können beide Größen nur mit Hilfe einer endlichen Stichprobe approximiert werden und sind deswegen mit Unsicherheiten behaftet. Zudem ist die Fehlerwahrscheinlichkeit ein stark vereinfachendes Gütemaß, das wichtige Aspekte des Problems wie die Fehlklassifikation in Bezug auf eine bestimmte Klasse ausblenden kann. Zusätzlich sind insbesondere bei diskriminativen Klassifikationsansätzen die

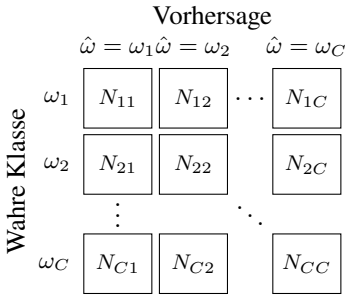
zur Berechnung benötigten a-priori Klassenwahrscheinlichkeiten in der Regel nicht bekannt.
Folgendes Beispiel verdeutlicht die Problematik: Zur Bewertung eines binären Klassifikators steht eine Teststichprobe mit 100 Beispielen zur Verfügung. Fünf Beispiele gehören zur Klasse ω_2, die restlichen 95 Beispiele gehören zu ω_1. Der Klassifikator ordnet alle Beispiele der Klasse ω_1 zu und ist damit in der Praxis unbrauchbar. Die Schätzung der Fehlerwahrscheinlichkeit P_e entspricht allerdings der a-priori Wahrscheinlichkeit von ω_2:

$$\hat{P}_e = \underbrace{\hat{P}(\omega_1 \,|\, \omega = \omega_2)}_{=1} P(\omega_2) + \underbrace{\hat{P}(\omega_2 \,|\, \omega = \omega_1)}_{=0} P(\omega_1) = P(\omega_2)\,. \quad (2.107)$$

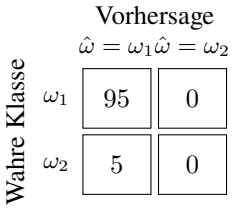
Ist $P(\omega_2)$ klein, wird der Fehler ebenfalls als klein eingeschätzt.
Eine Konfusionsmatrix liefert ein genaueres Bild. Die Konfusionsmatrix ist eine Tabelle, in der die Spalten für die Klassifikationsergebnisse (Vorhersagen) und die Zeilen für die wahren Klassen stehen. In jeder Zelle der Konfusionsmatrix wird die Anzahl der Beispiele N_{ik} eingetragen, die der Klasse ω_i angehören und der Klasse ω_k zugeordnet wurden, siehe Abbildung 2.12a. Die sich aus obigem Beispiel ergebende Konfusionsmatrix ist in Abbildung 2.12b zu sehen. Hier wird auch für Laien deutlich, dass der Klassifikator unbrauchbar ist.
Aus der Konfusionsmatrix lassen sich eine Reihe von Fehlermaßen ableiten. Die Genauigkeit (Accuracy) ist das Gegenstück zur Fehlerwahrscheinlichkeit P_e unter Vernachlässigung der a-priori Klassenwahrscheinlichkeiten:

$$\mathrm{accuracy} = \frac{1}{N} \sum_{c=1}^{C} N_{cc} \quad (2.108)$$

Ein Wert von $\mathrm{accuracy} = 1$ bedeutet perfekte Klassifikation, ein Wert von $\mathrm{accuracy} = \frac{1}{C}$ entspricht dem Zufallsentscheid. Wie die Fehlerwahrscheinlichkeit kann die Accuracy allerdings einen falschen Eindruck der Klassifikationsleistung vermitteln: Im Beispiel ist $\mathrm{accuracy} = 0{,}95$.



(a) Konfusionmatrix mit C Klassen



(b) Beispiel einer Konfusionmatrix mit zwei Klassen

Abbildung 2.12: Konfusionsmatrix zur empirischen Klassifikatorbewertung.

2.5.1 Maße für binäre Klassifikatoren

Weitere oft verwendete Maße beziehen sich auf binäre Klassifikation. Im Folgenden wird daher zunächst ebenfalls von einer binären Klassifikation ausgegangen. Die Verallgemeinerung auf Mehrklassenprobleme folgt im Anschluss. Bei binärer Klassifikation wird die Klasse ω_1 als Positiv- und die Klasse ω_2 als Negativklasse bezeichnet. Daher sind die Formelzeichen $N_{\text{tp}} = N_{11}$ (true positives), $N_{\text{fp}} = N_{21}$ (false positives), $N_{\text{fn}} = N_{12}$ (false negatives) und $N_{\text{tn}} = N_{22}$ (true negatives) gebräuchlich.
Recall (auch True Positive Rate, TPR oder Sensitivity) ist das Verhältnis der richtig als positiv klassifizierten Beispiele zu der Gesamtzahl der Positivbeispiele der Teststichprobe,

$$\text{recall} = \frac{N_{\text{tp}}}{N_{\text{tp}} + N_{\text{fn}}} \in [0, 1]. \tag{2.109}$$

Diese Zahl gibt also an, wie sicher ein Klassifikator eine vorliegende Positivklasse erkennt, bzw. wie viele Beispiele der Klasse ω_1 vom Klassifikator gefunden werden. Der Recall des Beispiels in Abbildung 2.12b ist $\text{recall} = \frac{95}{95} = 1$.

Precision ist das Verhältnis der richtig positiv klassifizierten Beispiele zur Gesamtzahl der als positiv klassifizierten Beispiele,

$$\text{precision} = \frac{N_{\text{tp}}}{N_{\text{tp}} + N_{\text{fp}}} \in [0, 1] \tag{2.110}$$

und gibt damit eine Abschätzung über die Sicherheit einer Klassifikation zu ω_1. Im Beispiel in Abbildung 2.12b ist $\text{precision} = \frac{95}{100} = 0{,}95$.
Precision und Recall geben unterschiedliche Aspekte der Klassifikation der Positivklasse wieder und dürfen daher nie isoliert voneinander betrachtet werden. Zum Beispiel kann $\text{recall} = 1$ erreicht werden, indem alle Beispiele der Klasse ω_1 zugeordnet werden. Daraus folgt allerdings, dass die Precision minimal wird, da neben N_{tp} auch N_{fp} den Maximalwert annimmt. Umgekehrt kann eine hohe Precision erreicht werden, indem fast keine Beispiele als ω_1 klassifiziert werden. Da dadurch auch viele Positivbeispiele aussortiert werden, sinkt allerdings der Recall.
Oft ist es dennoch wünschenswert verschiedene Klassifikatoren mit einer einzigen Kennzahl zu vergleichen. Das harmonische Mittel aus Precision und Recall ist eine naheliegende Möglichkeit, diese beiden Werte zu kombinieren. Dieses Maß ist als F_1-score bekannt und ein Spezialfall des allgemeineren F_β-score. Die F_β-score ist das mit β gewichtete harmonischen Mittel aus Precision und Recall,

$$F_\beta = \frac{\left(1 + \beta^2\right) \cdot \text{precision} \cdot \text{recall}}{\beta^2 \cdot \text{precision} + \text{recall}} \in [0, 1]. \tag{2.111}$$

Das Beispiel in Abbildung 2.12b liefert einen F_1-score von $F_1 = 0{,}97$. Hieran wird deutlich, dass der F_β-score die Klassifikationsleistung lediglich in Bezug auf die Positivklasse bewertet. Dieses Maß ist somit für unbalancierte Datensätze anfällig und ist in solchen Fällen nicht zur umfassenden Bewertung geeignet. Wird eine umfassende Aussage über die Gesamtklassifikationsleistung gewünscht, bietet sich Matthews' Correlation Coefficient

(MCC) an [Mat75]. MCC kann als Korrelation der Klassifikation mit der Ground Truth interpretiert werden und berechnet sich anhand von

$$\mathrm{MCC} = \frac{N_{\mathrm{tp}}\, N_{\mathrm{tn}} - N_{\mathrm{fp}}\, N_{\mathrm{fn}}}{\sqrt{\left(N_{\mathrm{tp}} + N_{\mathrm{fp}}\right)\left(N_{\mathrm{tp}} + N_{\mathrm{fn}}\right)\left(N_{\mathrm{tn}} + N_{\mathrm{fp}}\right)\left(N_{\mathrm{tn}} + N_{\mathrm{fn}}\right)}} \tag{2.112}$$

mit $\mathrm{MCC} \in [-1, 1]$. Das Maß ist symmetrisch und wird von unbalancierten Datensätzen wenig beeinflusst. Aus diesen Gründen ist dieses Maß für die Klassifikatorbewertung im Kontext der visuellen Inspektion besonders gut geeignet. Im Beispiel in Abbildung 2.12b ist $\mathrm{MCC} = 0$, was auf einen nutzlosen Klassifikator hindeutet.

2.5.2 ROC Kurven

Analog zum Recall wird der Fallout (auch False Positive Rate, FPR) als Verhältnis der positiv klassifizierten Negativbeispiele zu allen Negativbeispielen der Stichprobe definiert,

$$\mathrm{fallout} = \frac{N_{\mathrm{fp}}}{N_{\mathrm{fp}} + N_{\mathrm{tn}}} \in [0, 1]. \tag{2.113}$$

Ergänzend zur Precision ist der Fallout also ein Maß für die Sicherheit einer Klassifikation in ω_1. Ähnlich wie bei der Precision gibt es einen systematischen Zusammenhang zwischen Fallout und Recall: Eine geringe Anzahl an falsch-positiv klassifizierten Beispielen, d. h. ein geringer Fallout, geht mit geringem Recall einher. Umgekehrt bedeutet hoher Recall auch hohen Fallout. Der Zusammenhang ist allerdings nicht linear.
Wenn das Klassifikationsergebnis von einem skalaren Parameter wie beispielsweise der Parameter b der logistischen Regression in Gleichung (2.37) abhängt, kann ein Plot von Recall gegen Fallout für verschiedene Parameterwerte Aufschluss über das allgemeine Verhalten des Klassifikators geben. Dieser Plot heißt Receiver Operating Characteristics (ROC) Kurve und wurde ursprünglich zur Bewertung von Radarsystemen entwickelt. Idealisierte

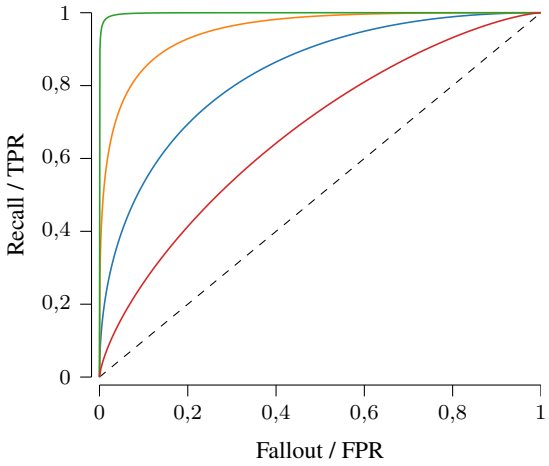


Abbildung 2.13: Typische Verläufe von ROC Kurven.

Beispiele von Receiver Operating Characteristics (ROC) Kurven sind in Abbildung 2.13 zu sehen.

ROC Kurven haben eine anschauliche Interpretation. Je weiter sich die Kurve der linken oberen Ecke anschmiegt, desto besser ist der Klassifikator: Ein perfekter Klassifikator erreicht 100% Recall bei 0% Fallout. Im Gegenteil dazu wird mit dem schlechtesten möglichen Klassifikator, dem Zufallsentscheid, $\text{recall} \approx \text{fallout}$ erwartet. Die Kennlinie dieses Klassifikators ist also die Diagonale. Reale Klassifikatoren bewegen sich immer zwischen diesen beiden Extremen. Falls eine ROC Kurve unter die Diagonal fällt, bedeutet dies, dass der Klassifikator die beiden Klassen vertauscht.

ROC Kurven werden zwar zur Klassifikatorbewertung verwendet, sagen aber eigentlich mehr über den Informationsgehalt der Merkmale aus. Nicht-diskriminative Merkmale führen effektiv zum Zufallsentscheid, während informative Merkmale eine bessere Klassifikation zulassen. Ein Beispiel für diese Interpretation findet sich in Beyerer et al. [BRN17].

Aus einer ROC Kurve kann wiederum die AUC-Kennzahl abgeleitet werden, die der Fläche unter der ROC Kurve (engl: **a**rea **u**nder **c**urve) entspricht. Eine $\mathrm{AUC} = 1$ bedeutet perfekte Klassifikation, der Zufallsentscheid entspricht $\mathrm{AUC} = 0{,}5$.
Neben der Klassifikatorbewertung können ROC Kurven auch verwendet werden, um den Klassifikationsparameter so festzulegen, dass ein gewünschter Fallout nicht überschritten, bzw. ein gewünschter Recall nicht unterschritten wird.

2.5.3 Erweiterung auf Mehrklassenklassifikation

Bis auf die Fehlerwahrscheinlichkeit und die Accuracy sind die besprochenen aus der Konfusionsmatrix abgeleiteten Bewertungsmaße nur die für binäre Klassifikation definiert. Diese Maße können aber auch in einem Mehrklassenszenario eingesetzt werden, indem jede Klasse getrennt betrachtet und bewertet wird. Eine Klassifikation unter C Klassen $\omega_1, \ldots, \omega_C$ wird also als C getrennte binäre Klassifikationsprobleme behandelt. Ziel des c-ten Klassifikationsproblem ist es, die Klasse ω_c von allen anderen Klassen $\overline{\omega}_c = \bigcup_{q \neq c} \omega_q$ zu unterscheiden. Dadurch kann für jede Klasse eine eigene Konfusionsmatrix abgeleitet werden (siehe Abbildung 2.14):

$$N_{\mathrm{fn},c} = \sum_{q \neq c} N_{cq}, \quad N_{\mathrm{fp},c} = \sum_{q \neq c} N_{qc}, \quad N_{\mathrm{tn},c} = \sum_{q,r \neq c} N_{qr} \tag{2.114}$$

und $N_{\mathrm{tp},c} = N_{cc}$. Mit dieser Konfusionmatrix können alle oben genannten binären Maße berechnet werden. Eine Bewertung der Klassifikationsleistung der Gesamtklassifikation wird durch Mittelung der individuellen Maße erreicht. Hierbei kann zusätzlich die Standardabweichung bzw. der Standardfehler über die verschiedenen Klassen angegeben werden.
Ebenso können ROC Kurven für jede einzelne Klasse erstellt und in einem gemeinsamen Diagramm aufgeführt werden, um die Klassifikationsleistung der verschiedenen Klassen feingliedriger zu vergleichen. Wenn das Klassifi-

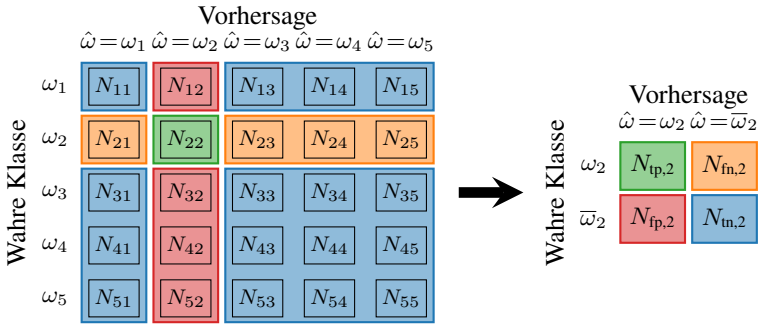


Abbildung 2.14: Überführung einer Konfusionsmatrix mit C Klassen in C binäre Konfusionsmatrizen. Beispiel hier: Reduzierung einer Konfusionsmatrix mit $C = 5$ Klassen auf die Konfusionsmatrix der Klasse ω_2.

kationsverfahren dies zulässt, können diese Kurven auch verwendet werden, um unterschiedliche Parameter für jede Klasse festzulegen.

2.5.4 Cross Validation

Wie eingangs beschrieben wird für die Evaluierung in der Regel eine von der Lernstichprobe $\mathcal{D}$ disjunkte Teststichprobe $\mathcal{T}$ verwendet, da nur so die Generalisierungsfähigkeit des Klassifikators abgeschätzt werden kann. Oft ist der Umfang der Gesamtstichprobe $\mathcal{X} = \mathcal{D} \cup \mathcal{T}$ aber gering, sodass die Bewertung der Klassifikationsgüte nicht zuverlässig möglich ist.
Die Kreuzvalidierung (engl: Cross Validation) kann hier Abhilfe schaffen. In einer K-fold Cross Validation wird die Gesamtstichprobe $\mathcal{X}$ in K annähernd gleich große Untermengen aufgeteilt. In K Runden, den sogenannten Folds, werden jeweils $(K-1)$ Teile als Lernstichprobe $\mathcal{D}_k$ $(k = 1, \ldots, K)$ verwendet und die verbleibende Untermenge als $\mathcal{T}_k$ zur Evaluation verwendet. Am Ende der K-fold Cross Validation werden die in den Folds berechneten Gütemaße meist durch Mittelung zusammengefasst. Zusätzlich kann die Standardabweichung oder der Standardfehler des Gütemaßes und damit die

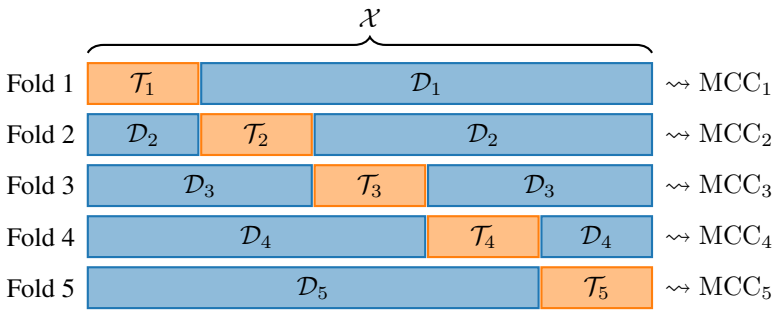


Abbildung 2.15: Prinzip einer 5-fold Cross Validation mit Gütemaß MCC.

Varianz des Klassifikators abgeschätzt werden. Das Prinzip einer 5-fold Cross Validation mit MCC als Gütemaß ist in Abbildung 2.15 zu sehen.

Kreuzvalidierung nutzt den Datensatz vollständig aus: Jedes Beispiel wird $K-1$ mal zum Training und einmal zur Evaluation verwendet. Ein Spezialfall tritt auf, wenn $K = |\mathcal{X}|$, d. h. wenn die Anzahl der Folds der Anzahl der Beispiele entspricht. In diesem Fall werden in jedem Fold alle bis auf ein Beispiel zum Training verwendet, weswegen man hier auch von Leave One Out Cross Validation spricht. In der Statistik ist dieses Verfahren als Jackknife bekannt und eng mit Bootstrapping verwandt (siehe z. B. Efron und Hastie [EH16], S. 155–159).

Eine weitere im Kontext der visuellen Inspektion wichtige Variante der Kreuzvalidierung ist die stratifizierte Kreuzvalidierung. Hier werden die Untermengen so gewählt, dass die Klassenverteilung in jedem Fold annähernd gleich ist. Dies ist insbesondere bei unbalancierten Datensätzen wichtig, da sonst die Evaluationsergebnisse verzerrt werden können.

2.6 Zusammenfassung

In diesem Kapitel wurde auf die Theorie der Mustererkennung als Grundlage der im nächsten Kapitel vorgestellten Verfahren eingegangen. Dazu wurden zunächst die Grundbegriffe der Mustererkennung definiert und auf die zentralen Konzepte Merkmalsextraktion und Klassifikation eingegangen. Die wichtigsten Ziele der Merkmalsextraktion – relevante, robuste und wenig redundante Merkmale – wurden erläutert und Standardmerkmale der visuellen Inspektion wurden vorgestellt. Im Anschluss wurde auf Merkmalsselektion sowie Feature Engineering, Feature Learning und Merkmale höherer Ordnung eingegangen. Im Hinblick auf die Klassifikation wurden kurz die für das maschinelle Lernen wichtigen Optimierungsverfahren Gradientenabstieg und genetische Algorithmen vorgestellt. Die Klassifikation wurde probabilistisch und in der Interpretation des Empirical Risk Minimization beleuchtet. Anschließend wurden die bedeutsamen Klassifikationsverfahren SVM und Entscheidungsbäume vorgestellt, sowie auf die Ensemblemethoden Random Forests und Boosting eingegangen. Das Kapitel schloss mit Verfahren zur empirischen Bewertung der Klassifikationsleistung.

3 Praxis

Nachdem im letzten Kapitel die Grundbegriffe der Mustererkennung eingeführt wurden sowie ein Überblick über Verfahren zur Merkmalsextraktion und Klassifikation gegeben wurde, wird in diesem Kapitel die Anwendung dieser Verfahren im Kontext der automatischen visuellen Inspektion am Beispiel der Schüttgutsortierung diskutiert. Dabei werden die Besonderheiten und Rahmenbedingungen der Schüttgutsortierung bei der Betrachtung mit einbezogen:

- Die Umgebung, d. h. insbesondere Beleuchtung und Hintergrund, ist kein Störfaktor, sondern eine Designgröße. Die Bilderfassung wird ausgelegt, um die anschließende Bildverarbeitung zu vereinfachen.
- Die zu klassifizierenden Objekte zeigen, verglichen mit den Objekten aus anderen Aufgaben des Maschinensehens, ein verhältnismäßig einfaches Erscheinungsbild.
- Die Klassen liegen oft nah beieinander, d. h. insbesondere bei Objekten natürlichen Ursprungs kann das Erscheinungsbild der Objekte wenig diskriminativ für die Klasseneinteilung sein.
- Besonders bei der Schüttgutsortierung unterliegen die Systeme harten Echtzeitanforderungen. Für die Verarbeitung der gesamten Kette aus Bildaufnahme, Verarbeitung und Ausschleusung stehen nur wenige Millisekunden zur Verfügung.

- Die Lernstichproben sind in der Regel nicht balanciert, da deutlich mehr Beispiele von Gutprodukten als von Schlechtprodukten verfügbar sind.
- Die Klassifikation findet in der Regel unter der Annahme einer offenen Welt (siehe Abschnitt 2.1) statt, da nicht alle Defektklassen bekannt sind.

Die Diskussion kann anhand der Bildverarbeitungskette eines Schüttgutsortiersystems, wie sie in der Einleitung motiviert und in Abbildung 1.3 gezeigt wurde, systematisiert werden. Diese Bildverarbeitungskette ist in Abbildung 3.1 abstrahiert dargestellt und zeigt vier mögliche Ansatzpunkte für Verfahren der Mustererkennung auf: Bilderfassung, Objektdetektion, Merkmalsextraktion und Klassifikation. Die Objektdetektion wird dabei in den allermeisten Fällen durch Hintergrundsubtraktion und Connected Component Analysis (CCA) (siehe Steger et al. [SUW08]) vollzogen. Da die Hintergrundsubtraktion durch die Kontrolle der Umgebung einfach ist und sehr effiziente Algorithmen für CCA existieren (z. B. von Zhao et al. [ZLZ13]), wird dieser Punkt hier nicht weiter behandelt.
Für jeden der verbleibenden Bausteine werden im Folgenden geeignete Verfahren vorgestellt. In Abschnitt 3.1 wird die Bilderfassung durch die Selektion optischer Filter optimiert, sodass diskriminative Merkmale bereits im Kamerabild hervorgehoben werden. Der Großteil dieses Kapitels beschäftigt sich mit verschiedenen Aspekten der Merkmalsextraktion, da hier großes Verbesserungspotential gegenüber dem Stand der Technik besteht [Dom12]. In Abschnitt 3.2 wird ein neues Verfahren zur Merkmalsselektion vorgestellt, welches die Kosten der Merkmalsgewinnung bei der Selektion berücksichtigt. Weiterhin wird in Abschnitt 3.3 ein Verfahren gezeigt, mit dem Formmerkmale automatisch aus einem Datensatz abgeleitet werden können. Zur Klassifikation anhand von Farbe und Textur wird der Bag of Visual Words-Ansatz in Abschnitt 3.4 für den Einsatz in der Schüttgutsortierung angepasst und um eine Rückweisungsoption unbekannter Objekte erweitert. In Abschnitt 3.5

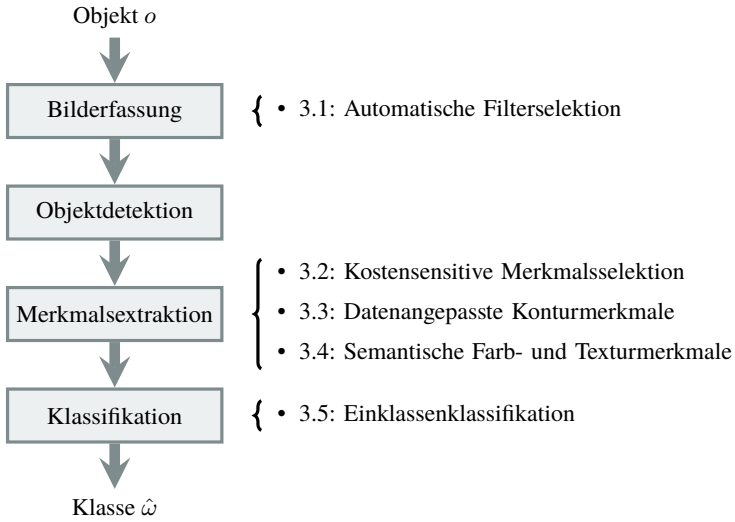


Abbildung 3.1: Abstrahierte Bildverarbeitungskette eines Schüttgutsortiersystems und Einordnung der in dieser Arbeit vorgestellten Ansätze.

wird schließlich ein Verfahren beschrieben, mit dem Entscheidungsbäume zur Dichteschätzung und schnellen Einklassenklassifikation anhand einer Stichprobe mit nur einer Klasse gelernt werden können.

3.1 Automatische Auswahl optischer Filter

Eine gut ausgelegte Bilderfassung vereinfacht die nachfolgende Merkmalsextraktion und Klassifikation erheblich. Die Qualität der Bilderfassung wird dabei vor allem durch die Wahl geeigneter Beleuchtung, Optik und Kamerachips, sowie die Anordnung dieser Komponenten beeinflusst [SUW08; BPF12]. Aber auch die Wahl des betrachteten Spektralbereichs ist relevant und muss insbesondere nicht auf das visuelle elektromagnetische Spektrum (VIS, $\lambda = 380\,\text{nm}$ bis $\lambda = 780\,\text{nm}$) begrenzt werden. Je nach Material finden sich diskriminative Informationen z. B. auch im ultravioletten Spektralbe-

reich (UV, $\lambda = 200\,\text{nm}$ bis $\lambda = 380\,\text{nm}$) und, besonders bei organischen Materialien, im nahen und kurzwelligen infraroten Spektralbereich (NIR und SWIR, $\lambda = 780\,\text{nm}$ bis $\lambda = 3000\,\text{nm}$).
Multi- und Hyperspektralkameras lösen den betrachteten Spektralbereich fein auf und ermöglichen somit beispielsweise die Detektion verschiedener chemischer Verbindungen. Daher findet diese Technik besonders im Bereich der Lebensmittelinspektion zunehmend Einsatz [FS12; Sch+15]. Daneben gibt es aber auch Anwendungen in anderen Bereichen wie Recycling, Bergbau oder Forensik [BPL15; BPL17]. Die Kameratechnik hat zwar in den letzten Jahren beträchtliche Fortschritte erzielt, jedoch ist das Datenvolumen hyperspektraler Bilder bei gleicher räumlicher Auflösung deutlich höher als bei herkömmlichen RGB- oder Grauwertkameras, was wiederum zu erhöhtem Zeitbedarf für die Übertragung und Verarbeitung dieser Daten führt. Zudem benötigen Hyperspektralkameras typischerweise eine stärkere Beleuchtung oder längere Integrationszeiten, da das verfügbare Licht auf viele schmale, statt auf wenige breite Kanäle aufgeteilt wird. Diese Faktoren machen den Einsatz in der Schüttgutsortierung momentan noch schwierig.
Um die Vorteile der spektralen Bilderfassung dennoch nutzen zu können, bietet sich ein Mittelweg an. Grauwertkameras werden mit passenden optischen Filtern kombiniert, die charakteristische Merkmale in spektralen Signaturen (optisch) extrahieren. Die Analyse der spektralen Signaturen erfolgt dabei im Labor. Die Signaturen der zu unterscheidenden Materialien werden mit hochauflösenden Spektrometern oder Hyperspektralkameras erfasst und anschließend auf diskriminative Merkmale untersucht. Im nächsten Schritt werden passende optische Filter angefertigt oder aus einem Katalog bestellt. Das fertige System verwendet nur noch das reduzierte, meist ein- bis vierkanalige Bild zur Klassifikation. Dieses Vorgehen ist allerdings sehr zeitaufwendig und erfordert ein gewisses Maß an Vorwissen und Erfahrung seitens der Systemdesigner, da aufgrund von kombinatorischer Explosion eine Auswertung aller möglichen Kombinationen nicht praktikabel ist. Eine zumindest teilweise Automatisierung der Filterselektion ist daher wünschenswert.

3.1.1 Stand der Technik

Automatische Filterselektion ist schon länger ein Thema in der visuellen Inspektion. Zur Detektion von Druckstellen auf Jonagold-Äpfeln nutzen Kleynen et al. [KLD03] einen Brute-Force-Ansatz, um 4 optimal zur Klassifikation geeignete Bandpassfilter aus 26 Filterkandidaten auszuwählen. Dazu wurden die Filterantworten aller 14950 Kombinationen der Kandidaten simuliert und anhand des Trainingsfehlers einer quadratischen Diskriminanzanalyse bewertet. Piron et al. [Pir+08] nutzen den gleichen Ansatz, um 2 bis 4 Filter für die Unterscheidung von Nutzpflanze und Unkraut aus einem Katalog von 22 Kandidaten auszuwählen. Die erschöpfende Suche funktioniert allerdings nur, solange die Anzahl der Filterkandidaten und der benötigten Filter klein ist.

Andere Ansätze suchen nicht nach den besten Filterkombinationen, sondern nach diskriminativen Spektralbändern. Diese werden bei einer nachgeschalteten manuellen Filterauswahl genutzt, um diese Auswahl zu unterstützen. Als Kriterium für die Bandselektion können z. B. die Koeffizienten einer Partial Least Squares Regression (PLSR) dienen [Osb+97]. PLSR ist ein Verfahren der multivariaten Statistik, in dem ein linearer Zusammenhang zwischen einer Beobachtungsmatrix $\boldsymbol{X}$ und einer Matrix von latenten Variablen $\boldsymbol{Y}$ (z. B. Stoffkonzentrationen, Klassenzugehörigkeit etc.) aufgestellt wird. Konkret wird ein fehlerbehafteter linearer Zusammenhang zwischen den Matrizen $\boldsymbol{X}$ und $\boldsymbol{Y}$ angenommen:

$$\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{B} + \boldsymbol{E}. \tag{3.1}$$

Hier ist $\boldsymbol{E}$ der (nichtlineare) Rekonstruktionsfehler und $\boldsymbol{B}$ enthält die Regressionskoeffizienten. Die Zerlegung wird dabei so gewählt, dass die Kovarianz zwischen $\boldsymbol{X}$ und $\boldsymbol{Y}$ maximiert ist. Für die spektrale Bandselektion kann PLSR eingesetzt werden, um sowohl eine vorgegebene Anzahl, als auch für die gegebene Aufgabe optimale Anzahl an Spektralbändern auszuwählen [Osb+97].

Alternative Ansätze bewerten Spektralbänder mittels des Fisher-Kriteriums, also dem Verhältnis der Inter- und Intraklassenvarianz dieser Bänder [FG01; Cha+01].
Ähnliche Ideen finden sich in der Fernerkundung. Pal experimentiert mit linearen SVMs und regularisierter logistischer Regression zur Bewertung individueller Wellenlängen [Pal09; Pal12]. Sowohl bei der SVM, als auch bei der logistischen Regression kodieren die Koeffizienten des Gewichtsvektors dabei die Relevanz der entsprechenden Spektralbänder. Guo et al. [Guo+06] nutzen hingegen die Transinformation zwischen den individuellen Spektralbändern und einer Menge von Basisspektren, die in ungesehenen Aufnahmen gefunden werden sollen.
Wieder andere Methoden bedienen sich Ideen des Filterdesigns, um Bandpassfilter zu generieren. In DeBacker et al. [DeB+05] werden diese Bandpassfilter mittels zentraler Wellenlänge und Filterbreite parametriert. De Backer et al. nutzen dann ein adaptives Simulated Annealing Verfahren, um die Parameter einer vorgegebenen Menge von Filtern geschlossen zu optimieren. Als Optimierungskriterium dient hierbei die Bhattacharya-Schranke, die eine obere Schranke für die Fehlerwahrscheinlichkeit (Gleichung 2.30) darstellt. Nakauchi et al. [NNY12] parametrieren Bandpassfilter durch die kleinste und größte durchgelassene Wellenlänge und optimieren diese Parameter durch eine globale stochastische Suche, gefolgt von einer lokalen Optimierung. Sowohl die globale Suche, als auch die lokale Optimierung nutzen das Fisher-Kriterium als Gütemaß. Beide Verfahren machen allerdings keine Aussagen darüber, wie die modellierten Filter physikalisch umgesetzt werden sollen.
Eine Möglichkeit hierfür ist das Verfahren von Taphanel [Tap15]. In seiner Arbeit stellt Taphanel ein Verfahren zur Simulation von Interferenzfiltern vor und nutzt diese Simulation, um die Schichtdicken der Filter so zu optimieren, dass eine gegebene Gütefunktion maximiert wird. Bei geeigneter Formulierung des Optimierungsfunktionals kann dieser Ansatz also zur Umsetzung der gefundenen Bandpassfilter eingesetzt werden. Allerdings ist die Fertigung solcher Filter verhältnismäßig teuer.

3.1.2 Beitrag zum Stand der Technik

Alle diese Verfahren zur Bandselektion bzw. zur Parameteroptimierung zeigen gute Ergebnisse in ihren jeweiligen Anwendungen, allerdings gibt es keine Garantie, dass die gefundenen Parameter auch durch verfügbare Filter physikalisch umgesetzt werden können. Es lohnt sich also, einen Schritt zurückzutreten und das Problem der Filterselektion aus einer anderen Perspektive zu betrachten.
In diesem Abschnitt wird ein allgemeines Framework für die automatische Selektion optischer Filter aus einer großen Menge an Kandidaten vorgestellt. Die Filterselektion wird dabei als Merkmalsselektion formuliert. Anders als bei anderen Verfahren ist die Art der Filter dabei beliebig, womit auch komplizierte Filter, beispielsweise Farbfilter, in die Selektion einbezogen werden können. Zusätzlich werden verschiedene Ansätze für die Implementierung der Selektion vorgestellt und anhand zweier Datensätze aus dem Bereich der Mineralsortierung und der Lebensmittelinspektion quantitativ verglichen. Das beschriebene Verfahren wurde in Richter und Beyerer [RB14a] veröffentlicht.

3.1.3 Filterselektion als Merkmalsselektion

Die Filterselektion kann folgendermaßen formalisiert werden: Gegeben ist eine Menge $\mathcal{F}$ von D Filtern sowie eine Menge $\mathcal{D} = \{(\boldsymbol{p}_n, \omega_n) \,|\, n = 1, \ldots, N\}$ von N spektralen Signaturen $\boldsymbol{p}_n$ mit bekannten Klassenzugehörigkeiten ω_n. Gesucht ist eine Menge von Filtern $\mathcal{S}^* \subseteq \mathcal{F}$, die ein nicht weiter spezifiziertes Gütemaß $U\left(\mathcal{S}^*; \mathcal{D}\right)$ maximiert,

$$\mathcal{S}^* = \arg\max_{\mathcal{S} \subseteq \mathcal{F}} U\left(\mathcal{S}; \mathcal{D}\right). \tag{3.2}$$

Diese Formalisierung entspricht fast wörtlich der Formalisierung der Merkmalsselektion aus Abschnitt 2.2.2. In der Tat lässt sich die Filterselektion auf die Merkmalsselektion abbilden, indem die Filter $\psi_k \in \mathcal{F}$ anhand der $\boldsymbol{p}_n$ simuliert werden: Die Filter extrahieren die Merkmale $[\boldsymbol{x}_n]_k = \psi_k(\boldsymbol{p}_n)$,

anhand derer klassifiziert werden soll. Das bedeutet, dass alle bekannten Methoden der Merkmalsselektion zur Filterselektion genutzt werden können. Implizit wurden solche Verfahren in den oben genannten Ansätzen auch schon verwendet: Kleynen et al. [KLD03] nutzen einen Wrapperansatz in Verbindung mit einer quadratischen Diskriminanzanalyse. Die Verfahren von Feyaerts und Gool [FG01], Chao et al. [Cha+01] und Guo et al. [Guo+06] entsprechen Filtermethoden der Merkmalsselektion; und die Ansätze von Osborne et al. [Osb+97] und Pal [Pal09; Pal12] sind eingebettete Verfahren. Bis auf den Ansatz von Kleynen et al. sind diese Verfahren aber dadurch limitiert, dass sie nur Bandpassfilter zulassen, obwohl die Formulierung als Merkmalsselektion im Prinzip beliebige Filter, etwa Notch- oder Farbfilter, zulässt.
Im Folgenden werden Verfahren für die drei Ansätze – Wrapper, Filter und eingebettet – formuliert und anhand von Beispieldatensätzen ausgewertet. Alle Ansätze sind Greedy-Verfahren, selektieren die Filter also in mehreren Runden, wobei jeweils das Filter gewählt wird, welches das jeweilige Gütekriterium optimiert. Zudem wird hier von einem binären Klassifikationsproblem ausgegangen. Die Verfahren lassen sich aber auch auf Mehrklassenprobleme erweitern.

3.1.4 Wrapperansatz: Lineare Diskriminanzanalyse

Fishers Lineare Diskriminanzanalyse (LDA) dient als Basis für ein einfaches Wrapper-Kriterium. Wie die logistische Regression und die SVM ist LDA ein linearer Klassifikator mit $h(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{x} - b$. Der Vektor $\boldsymbol{w}$ ist dabei so gewählt, dass die Klassentrennung der projizierten Merkmalsvektoren $x_n := \boldsymbol{w}^\top \boldsymbol{x}_n$ mit $n = 1, \ldots, N$ maximiert wird. Erreicht wird dies durch die Maximierung des Fisher-Kriteriums

$$J_{\text{LDA}}(\boldsymbol{w}) = \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2}, \tag{3.3}$$

wobei m_1, m_2 und s_1^2, s_2^2 die empirischen Mittelwerte und Varianzen der projizierten Merkmale x_n der Klasse ω_1 bzw. ω_2 sind. Dieses Kriterium maximiert also den Abstand der Klassenmittelpunkte im projizierten Merkmalsraum mit dem Ziel, dort eine optimale Klassentrennbarkeit zu erreichen. Gleichzeitig soll aber auch die Varianz innerhalb der Klassen minimiert werden, damit diese möglichst wenig überlappen.
In Gleichung (3.3) ist der Zusammenhang mit $\boldsymbol{w}$ nur implizit. Durch Umformung wird dieser Zusammenhang explizit (siehe z. B. Beyerer et al. [BRN17] oder Bishop [Bis13]):

$$J_{\text{LDA}}(\boldsymbol{w}) = \frac{\boldsymbol{w}^\top \boldsymbol{S}_B \boldsymbol{w}}{\boldsymbol{w}^\top \boldsymbol{S}_W \boldsymbol{w}}, \tag{3.4}$$

Hier ist $\boldsymbol{S}_B$ die Streumatrix der Lernstichprobe zwischen den Klassen (**b**etween classes) und $\boldsymbol{S}_W$ die Summe der Streumatrizen innerhalb der Klassen (**w**ithin classes). Mit den mittleren Merkmalsvektoren $\overline{\boldsymbol{x}}_1$ und $\overline{\boldsymbol{x}}_2$ der Klassen ω_1 bzw. ω_2 berechnen sich diese Streumatrizen als

$$\boldsymbol{S}_B := (\overline{\boldsymbol{x}}_2 - \overline{\boldsymbol{x}}_1)(\overline{\boldsymbol{x}}_2 - \overline{\boldsymbol{x}}_1)^\top \tag{3.5}$$

$$\boldsymbol{S}_W := \sum_{\substack{(\boldsymbol{x}_n, \omega_n) \in \mathcal{D} \\ \omega_n = \omega_1}} (\boldsymbol{x}_n - \overline{\boldsymbol{x}}_1)(\boldsymbol{x}_n - \overline{\boldsymbol{x}}_1)^\top + \sum_{\substack{(\boldsymbol{x}_n, \omega_n) \in \mathcal{D} \\ \omega_n = \omega_2}} (\boldsymbol{x}_n - \overline{\boldsymbol{x}}_2)(\boldsymbol{x}_n - \overline{\boldsymbol{x}}_2)^\top. \tag{3.6}$$

Differenzierung des Fisher-Kriteriums im Bezug zu $\boldsymbol{w}$ zeigt schließlich, dass

$$\boldsymbol{w} \propto \boldsymbol{S}_W^{-1} (\overline{\boldsymbol{x}}_2 - \overline{\boldsymbol{x}}_1) \tag{3.7}$$

das Kriterium aus Gleichung (3.4) maximiert [Bis13]. Der Parameter b wird anschließend so festgelegt, dass der Klassifikationsfehler minimiert wird.
Wie in Abschnitt 2.2.2 erwähnt, ist der Hauptnachteil aller Wrapperansätze die für das Training und die Auswertung benötigte Rechenzeit. Mit LDA kann diese Einschränkung aber umgangen werden: Da das Fisher-Kriterium bereits die Klassentrennbarkeit quantifiziert, kann es als Stellvertreter der Klassifika-

tionsgüte verwendet werden. Genauer: Für zwei Unterräume $\mathbb{M}_1, \mathbb{M}_2 \subset \mathbb{M}$ gleicher Dimension gilt, dass ein durch $\boldsymbol{w}_1, b_1$ parametrierter Klassifikator in $\mathbb{M}_1$ eine höhere Klassifikationsgüte erzielt als ein durch $\boldsymbol{w}_2, b_2$ parametrierter Klassifikator in $\mathbb{M}_2$, wenn $J(\boldsymbol{w}_1) > J(\boldsymbol{w}_2)$.
Insgesamt ergibt sich folgendes Greedy-Wrapperverfahren: Beginnend mit der initial leeren Selektion $\mathcal{S}_0 := \emptyset$ wird iterativ dasjenige Merkmal $\psi \in \overline{\mathcal{S}_t}$ hinzugefügt, das das Fisher-Kriterium über den von $\mathcal{S}_t \cup \{\psi\}$ induzierten Merkmalsraum maximiert,

$$\mathcal{S}_{t+1} := \mathcal{S}_t \cup \{\psi_{t+1}\} \quad \text{mit} \quad \psi_{t+1} := \underset{\psi \in \overline{\mathcal{S}_t}}{\arg\max}\, J_{\text{LDA}}\left(\boldsymbol{w}_t\right). \tag{3.8}$$

Die Selektion wird abgebrochen, wenn sich die Güte nicht mehr signifikant verbessert oder wenn die Selektion eine vorgegebene Anzahl an Merkmalen enthält.

3.1.5 Filteransatz: Conditional Likelihood Maximization

In Abschnitt 2.2.2 wurde bereits das Conditional Likelihood Maximization (CLM) Framework von Brown et al. [Bro+12] vorgestellt. Die Autoren leiten über einen Maximum-Likelihood-Ansatz die bedingte Transinformation (Conditional Mutual Information, CMI) zwischen Merkmal ψ und Klasse ω bei Kenntnis der bisher selektierten Merkmale $\mathcal{S}$ her und nutzen diese als Selektionskriterium:

$$J_{\text{CMI}}\left(\psi \,|\, \mathcal{S}\right) = I\left(\psi(\boldsymbol{p})\,;\, \omega \,|\, \mathcal{S}(\boldsymbol{p})\right). \tag{3.9}$$

Wie beim Wrapperansatz wird dieses Kriterium genutzt, um ausgehend von der leeren Menge sukzessive Merkmale hinzuzufügen,

$$\mathcal{S}_{t+1} := \mathcal{S}_t \cup \{\psi_{t+1}\} \quad \text{mit} \quad \psi_{t+1} := \underset{\psi \in \overline{\mathcal{S}_t}}{\arg\max}\, J_{\text{CMI}}\left(\psi \,|\, \mathcal{S}_t\right). \tag{3.10}$$

Die Selektion wird gestoppt, wenn $J_{\text{CMI}}\left(\psi_{t+1} \,|\, \mathcal{S}_t\right) < \tau$ oder eine maximale Anzahl an selektierten Merkmalen erreicht ist. Ebenso können nach jeder Selektionsrunde die redundanten Merkmale entfernt werden.
Leider ist dieses Kriterium aufgrund kombinatorischer Explosion bei der Schätzung der zugrunde liegenden Merkmalsdichten in der Praxis ohne zusätzliche Annahmen nicht einsetzbar. Unter der Annahme, dass die selektierten Merkmale $\psi_k \in \mathcal{S}$ bei Kenntnis eines beliebigen nicht-selektierten Merkmals $\tilde{\psi} \in \overline{\mathcal{S}_t}$ voneinander unabhängig sind, und dass die $\psi_k \in \mathcal{S}$ bei Kenntnis von $\tilde{\psi}$ und der Klasse ω voneinander unabhängig sind, d. h. dass

$$p\left(\mathcal{S} \,\middle|\, \tilde{\psi}\right) = \prod_{\psi_k \in \mathcal{S}} p\left(\psi_k \,\middle|\, \tilde{\psi}\right) \quad \text{und} \tag{3.11}$$

$$p\left(\mathcal{S} \,\middle|\, \tilde{\psi}, \omega\right) = \prod_{\psi_k \in \mathcal{S}} p\left(\psi_k \,\middle|\, \tilde{\psi}, \omega\right) \tag{3.12}$$

gilt, lässt sich das CMI-Selektionskriterium jedoch folgendermaßen vereinfachen [Bro+12]:

$$J_{\text{CMI}}\left(\psi \,|\, \mathcal{S}_t\right) = I\left(\psi\,;\,\omega\right) - \sum_{\psi_k \in \mathcal{S}_t} I\left(\psi_k\,;\,\psi\right) + \sum_{\psi_k \in \mathcal{S}_t} I\left(\psi_k\,;\,\psi \,|\, \omega\right). \tag{3.13}$$

Jeder dieser Terme hat eine eindeutige Interpretation: Der erste Term quantifiziert die Relevanz eines Merkmals, also wie gut das Merkmal zur Klassifikation geeignet ist. Der zweite Term bestraft die Redundanz mit den bereits selektierten Merkmalen. Der dritte Term belohnt wiederum schwache Relevanz des Merkmals, also Relevanz in Verbindung mit den bereits selektierten Merkmalen. Mit dieser Interpretation können nun auch andere bekannte Verfahren analysiert werden.

Das Maximum-Relevance Minimum-Redundancy (MRMR) Verfahren von Peng et al. [PLD05] und das Joint Mutual Information (JMI) Verfahren von Yang und Moody [YM99] lassen sich im Kontext von CLM durch

$$J_{\text{MRMR}}\left(\psi \,|\, \mathcal{S}_t\right) = I\left(\psi\,;\,\omega\right) - \frac{1}{|\mathcal{S}_t|}\sum_{\psi_k \in \mathcal{S}_t} I\left(\psi_k\,;\,\psi\right) \quad \text{und} \tag{3.14}$$

$$J_{\text{JMI}}\left(\psi \,|\, \mathcal{S}_t\right) = I\left(\psi\,;\,\omega\right) - \frac{1}{|\mathcal{S}_t|}\sum_{\psi_k \in \mathcal{S}_t} I\left(\psi_k\,;\,\psi\right) + \frac{1}{|\mathcal{S}_t|}\sum_{\psi_k \in \mathcal{S}_t} I\left(\psi_k\,;\,\psi \,|\, \omega\right) \tag{3.15}$$

ausdrücken. MRMR implementiert also die implizite Annahme, dass die selektierten Merkmale paarweise unabhängig gegeben der Klasse sind, wodurch der dritte Term in Gleichung (3.13) verschwindet. Zudem schrumpft der Redundanz-Term mit zunehmender Anzahl selektierter Merkmale. Es wird also zusätzlich angenommen, dass der Grad der Unabhängigkeit der Merkmale einer Selektion mit der Größe der Selektion zunimmt. JMI fügt MRMR den dritten Term von Gleichung (3.13) hinzu, macht also nicht die Annahme einer paarweise bedingten Unabhängigkeit. Wie bei MRMR wächst jedoch mit der Anzahl der selektierten Merkmale auch der Glaube in die paarweise und paarweise bedingte Unabhängigkeit der Merkmale.
Mit dieser Formulierung kann das kontextuelle Vorwissen eingebracht werden, dass die Merkmale hier optischen Filtern entsprechen: Stark überlappende Filter erzeugen ähnliche Bilder und somit stochastisch abhängige Merkmale, während zwei Filter in völlig verschiedenen Spektralbereichen stochastisch

unabhängige Merkmale extrahieren sollten. Diese Überlegung führt zu den similarity-MRMR (sMRMR) und similarity-JMI (sJMI) Maßen

$$J_{\text{sMRMR}}\left(\psi \,|\, \mathcal{S}_t\right) = I\left(\psi\,;\,\omega\right) - \sum_{\psi_k \in \mathcal{S}_t} s(\psi,\psi_k) I\left(\psi_k\,;\,\psi\right) \quad \text{und} \tag{3.16}$$

$$J_{\text{sJMI}}\left(\psi \,|\, \mathcal{S}_t\right) = I\left(\psi\,;\,\omega\right) + \sum_{\psi_k \in \mathcal{S}_t} s(\psi,\psi_k) \left(I\left(\psi_k\,;\,\psi \,|\, \omega\right) - I\left(\psi_k\,;\,\psi\right)\right) \tag{3.17}$$

Hier quantifiziert $0 \leq s(\psi,\psi_k) \leq 1$ die Ähnlichkeit zwischen den zu den Filtern ψ und ψ_k gehörenden Transmissionskurven $f(\lambda)$ und $f_k(\lambda)$. Dieses Maß ist 1, wenn $f \equiv f_k$ und strebt gegen 0, je unähnlicher f und f_k sind. Ein einfaches Maß, das diese Bedingungen erfüllt, ist die auf Funktionen erweiterte Kosinusähnlichkeit von f und f_k,

$$s(\psi,\psi_k) := \frac{\langle f, f_k\rangle}{\sqrt{\langle f, f\rangle \cdot \langle f_k, f_k\rangle}} \quad \text{mit} \quad \langle g, h\rangle = \int_{\lambda_0}^{\lambda_1} g(\lambda) h(\lambda)\,\mathrm{d}\lambda. \tag{3.18}$$

Dieses Maß beachtet allerdings nicht den spektralen „Abstand" zwischen den Filtern: Filter, die über benachbarte Spektralbereiche integrieren, tendieren eher zur stochastischen Abhängigkeit als Filter, deren Wellenlängen weit auseinander liegen. Für zwei Bandpassfilter ψ und ψ_k kann dieser Abstand durch den Abstand der zentralen Wellenlängen $|\lambda_{c,f} - \lambda_{c,f_k}|$ berücksichtigt werden,

$$s(\psi,\psi_k) := \frac{\langle f, f_k\rangle}{\sqrt{\langle f, f\rangle \cdot \langle f_k, f_k\rangle}} \cdot \frac{1}{1 + |\lambda_{c,f} - \lambda_{c,f_k}|}. \tag{3.19}$$

Alternativen für ein Ähnlichkeitsmaß können beispielsweise aus der Wasserstein-Distanz abgeleitet werden. Hierzu müssen allerdings die Filterfunktionen normalisiert werden, wodurch Information über die Unterschiede der Signalstärke zweier Filter verworfen wird.

3.1.6 Eingebetteter Ansatz: AdaBoost

Wie bereits in Abschnitt 2.4.7 erwähnt wurde, kann AdaBoost als eingebettetes Verfahren der Merkmalsselektion eingesetzt werden, indem jedes Merkmal ψ_k mit einem schwachen Klassifikator h_k assoziiert wird. Hier wird eine MAP-Klassifikation mit Gauß-Dichten und gleichen a-priori Klassenwahrscheinlichkeiten verwendet, d. h. mit $x_k := \psi_k(\boldsymbol{p})$ ist

$$h_k(\boldsymbol{p}) := \begin{cases} 1 & \text{wenn } p\left(x_k \mid \mu_{1,k}, \sigma_{1,k}\right) > p\left(x_k \mid \mu_{2,k}, \sigma_{2,k}\right) \\ -1 & \text{sonst .} \end{cases} \tag{3.20}$$

Die Parameter der Gauß-Dichten werden durch Maximum-Likelihood-Schätzung festgelegt. Nach dem Training werden die T selektierten Klassifikatoren anhand der Gewichtung $|\alpha_t|$ sortiert, da diese die Relevanz des zugehörigen Merkmals kodiert. Anders als bei den anderen beiden Verfahren kann es hier vorkommen, dass ein Filter mehrfach selektiert wird.

3.1.7 Filtersimulation

Die von einer Kamera gemessene Antwort $g_f(s)$ eines Filters $f(\lambda)$ auf ein Reflektanzspektrum eines Stoffes $s(\lambda)$ entspricht dem Standardskalarprodukt der Filterfunktion und des Spektrums,

$$g_f(s) := \langle f, s \rangle = \int_{-\infty}^{\infty} f(\lambda) s(\lambda) \, \mathrm{d}\lambda. \tag{3.21}$$

Mögliche spektrale Verzeichnungseffekte der Optik, die Quanteneffizienz der Kamera sowie das Spektrum der Beleuchtung werden hier ebenfalls in der Filterfunktion $f(\lambda)$ modelliert. Weitere Effekte, wie eine winkelabhängige Transmissionsfunktion des Filters, werden hier allerdings nicht beachtet.
In der Praxis werden die Spektren $s(\lambda)$ nur an bestimmten Stützstellen $\lambda_1, \ldots, \lambda_D$ gemessen. Diese Messungen werden in einen Vektor $\boldsymbol{s} := (s(\lambda_1), \ldots, s(\lambda_D))^\top$ zusammengefasst. Die Abtastung der Filterfunk-

tion $f(\lambda)$ an den gleichen Stützstellen ergibt die vektorielle Repräsentation der Filterantwort $\boldsymbol{f} := (f(\lambda_1), \ldots, f(\lambda_D))^\top$. Die Filterantwort lässt sich schließlich durch das Skalarprodukt der beiden Vektoren berechnen, d. h. $g_f(\boldsymbol{s}) \approx \boldsymbol{f}^\top \boldsymbol{s}$.
Bei realen Kameras wird dieser gemessene Grauwert zudem noch quantisiert. Eine solche b-Bit Quantisierung kann hier ebenfalls mittels

$$g'(\boldsymbol{s}) = \left\lfloor \frac{g(\boldsymbol{s})}{g_{\max}} \cdot 2^b \right\rfloor \cdot \frac{1}{2^b} \tag{3.22}$$

simuliert werden, wobei $g_{\max}$ die maximale simulierte Intensität in der Lernstichprobe bezeichnet:

$$g_{\max} := \max_{(\boldsymbol{p}_n, \omega_n) \in \mathcal{D}} \left\{ \max_{\psi_f \in \mathcal{S}_t} g_f(\boldsymbol{p}_n) \right\}. \tag{3.23}$$

Die Filtersensitivitäten $f(\lambda)$ stammen entweder aus Messungen realer Filter oder werden künstlich durch ein parametrisches Modell erzeugt. Ersteres bietet sich vor allem bei komplizierten Filterkurven, wie z. B. Farbfiltern einer RGB-Kamera, an. Für Bandpassfilter mit zentraler Wellenlänge λ_c und Halbwertsbreite w wird hier ein einfaches Modell angenommen:

$$f^{\mathrm{BP}}_{\lambda_c, w}(\lambda) := \exp\left\{ -(\log 2) \cdot \left(\frac{\lambda - \lambda_c}{\frac{1}{2} w} \right)^{2r} \right\}. \tag{3.24}$$

Der Parameter $r \in \mathbb{N}$ steuert hierbei die Flankensteilheit des Filters. Abbildung 3.2 zeigt Beispiele für die so generierten Transmissionskurven.

3.1.8 Experimente und Ergebnisse

Die vorgestellte Methode zur Filterselektion wurde an zwei unterschiedlichen Sortierproblemen aus der Praxis erprobt: (a) Klassifikation von Mineralen als Magnesit oder Talk und (b) Trennung von getrockneten, zerkleinerten

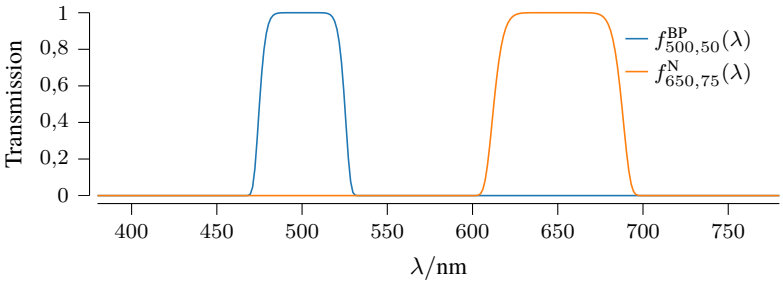


Abbildung 3.2: Synthetisch generierte Transmissionskurven zweier Bandpassfilter nach Gleichung (3.24). In beiden Fällen ist $r = 3$.

Paprikafrüchten von Stielresten, Haaren und anderen Fremdkörpern. In beiden Experimenten wurden die spektralen Signaturen der Objekte aus Bildern einer Hyperspektralkamera extrahiert. Das Aufnahmesystem und die Vorverarbeitung der Aufnahmen wird von Irgenfried und Negara [IN13] beschrieben. Der Spektralbereich betrug $1050\,\mathrm{nm}$ – $2450\,\mathrm{nm}$ (SWIR) in Experiment (a) und $405\,\mathrm{nm}$ – $1025\,\mathrm{nm}$ (VIS-NIR) in Experiment (b). In Experiment (a) standen 78906 Spektren der Positiv- und 136520 Spektren der Negativklasse, in Experiment (b) standen 69754 Spektren der Positiv- und 164549 Spektren der Negativklasse zur Verfügung. Es wurden pro Klasse jeweils 40% der Spektren zur Filterselektion verwendet, die restlichen 60% dienten der Evaluation.

In beiden Experimenten wurden Bandpassfilter gemäß Gleichung (3.24) simuliert. Innerhalb des Spektralbereichs der Messungen wurden alle $10\,\mathrm{nm}$ je ein Filter der Breite $w = 50\,\mathrm{nm}$, $w = 75\,\mathrm{nm}$ und $w = 100\,\mathrm{nm}$ generiert. Im ersten Experiment wurden so 423 Filterkandidaten, im zweiten Experiment 189 Filterkandidaten generiert.

Die Selektion mittels CLM benötigt eine Schätzungen der (bedingten) Transinformation. Diese Schätzungen wurden mittels eines Maximum-Likelihood-Verfahrens ermittelt, d. h.

$$I(X\,;\,Y) \approx \sum_{x,y \in X \times Y} \hat{P}(x,y) \cdot \log_2 \frac{\hat{P}(x,y)}{\hat{P}(x)\hat{P}(y)} \tag{3.25}$$

$$I(X\,;\,Y \mid Z) \approx \sum_{z \in Z} \hat{P}(z) \sum_{x,y \in X \times Y} \hat{P}(x,y \mid z) \cdot \log_2 \frac{\hat{P}(x,y \mid z)}{\hat{P}(x \mid z)\hat{P}(y \mid z)}, \tag{3.26}$$

wobei die zugrunde liegenden Wahrscheinlichkeiten durch Histogramme geschätzt wurden wurden. Dafür wurden die Filterantworten auf 16 Wertebereiche diskretisiert und die Auftrittswahrscheinlichkeiten $\hat{P}$ durch Zählstatistiken ermittelt.

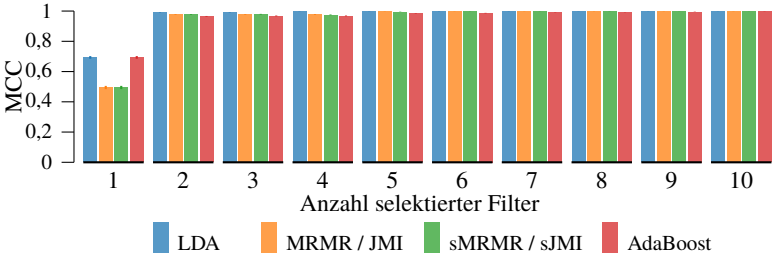
Tabelle 3.1 zeigt die Ergebnisse einer Selektion von drei Filtern. In beiden Experimenten stimmen die ersten beiden selektierten Filter der CLM-Verfahren überein. Die Auswahl der kanonischen und der modifizierten Kriterien unterscheidet sich erst ab dem dritten Filter. In beiden Experimenten selektieren MRMR und JMI bzw. sMRMR und sJMI die gleichen Filter. Im ersten Experiment stimmt das erste von LDA und AdaBoost selektierte Filter überein, im zweiten Experiment werden gänzlich unterschiedliche Filter selektiert. Im zweiten Experiment wird durch das AdaBoost-Verfahren außerdem das gleiche Filter zweimal selektiert. Dies ist dem Verfahren geschuldet: Im Unterschied zu LDA und CLM werden bei AdaBoost bereits selektierte Merkmale nicht aus der Kandidatenliste entfernt.

Um die Güte der Selektion zu bewerten, wurden die nicht zur Filterselektion verwendeten Daten eingesetzt. Für jedes Verfahren wurden ein bis zehn Filter selektiert und die Simulationsergebnisse in einer 10-fold Cross Validation mit einer linearen soft margin SVM mit $C = 1$ klassifiziert. Abbildung 3.3 zeigt den mit jedem Verfahren erreichten MCC in Abhängigkeit der Anzahl selektierter Filter.

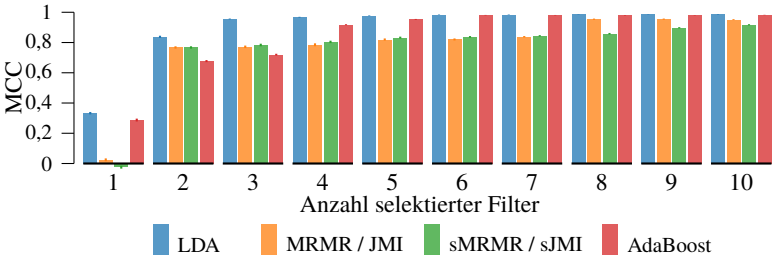
Tabelle 3.1: Ergebnisse der Selektion von drei Filtern mit den verschiedenen Selektionsmethoden.

Methode	**Filter λ_c/w in Reihenfolge der Selektion**		
Experiment (a): Mineralsortierung im SWIR			
LDA	2371 nm / 50 nm	2231 nm / 50 nm	1711 nm / 50 nm
MRMR	2451 nm / 100 nm	2431 nm / 75 nm	2301 nm / 50 nm
JMI	2451 nm / 100 nm	2431 nm / 75 nm	2301 nm / 50 nm
sMRMR	2451 nm / 100 nm	2431 nm / 75 nm	2281 nm / 50 nm
sJMI	2451 nm / 100 nm	2431 nm / 75 nm	2281 nm / 50 nm
AdaBoost	2371 nm / 50 nm	1231 nm / 50 nm	2381 nm / 50 nm
Experiment (b): Paprika gegen Fremdkörper im VIS			
LDA	745 nm / 50 nm	755 nm / 100 nm	655 nm / 50 nm
MRMR	985 nm / 50 nm	975 nm / 100 nm	775 nm / 100 nm
JMI	985 nm / 50 nm	975 nm / 100 nm	775 nm / 100 nm
sMRMR	985 nm / 50 nm	975 nm / 100 nm	885 nm / 50 nm
sJMI	985 nm / 50 nm	975 nm / 100 nm	885 nm / 50 nm
AdaBoost	665 nm / 50 nm	735 nm / 50 nm	655 nm / 50 nm

In Experiment (a) sind, unabhängig vom Verfahren, bereits zwei Filter für eine fast perfekte Klassifikation ausreichend. Die LDA-Selektion liefert bessere Ergebnisse als die anderen Verfahren, allerdings ist der Unterschied der Klassifikationsgüte ab zwei Filtern sehr gering. Der größte Unterschied liegt hier in der Selektion des ersten Filters. LDA und AdaBoost wählen gut geeignete Filter, während die CLM-Verfahren eine schlechtere Wahl treffen. Der Grund könnte sein, dass die SVM Klassifikation mit einem Merkmal der impliziten Klassifikation in LDA und AdaBoost entspricht. Das selektierte



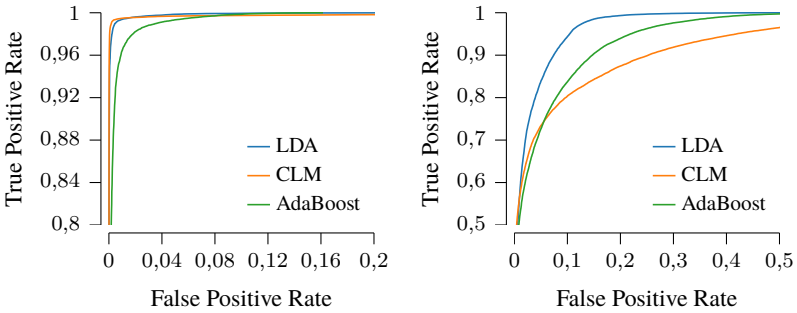
(a) Mineralsortierung im SWIR



(b) Paprika gegen Fremdkörper im VIS

Abbildung 3.3: Klassifikationsgüte der selektierten Filterkombinationen.

Filter ist also auf den Klassifikator abgestimmt. Im Unterschied dazu wird bei CLM der Klassifikator explizit nicht betrachtet.
Experiment (b) zeigt ein differenzierteres Bild. Auch hier ist die Selektion des ersten Filters durch CLM der Selektion durch LDA und AdaBoost unterlegen. Allerdings wird dies durch den zweiten Filter ausgeglichen. Eine signifikante Verbesserung der Klassifikationsleistung mit drei Merkmalen wird nur durch LDA erreicht. AdaBoost erreicht eine signifikant bessere Klassifikation mit vier Merkmalen, die hier aber nur drei unterschiedlichen Filtern entsprechen (siehe Tabelle 3.1).
Unabhängig vom Experiment werden die besten Ergebnisse mit dem LDA-Verfahren erzielt. Hierfür können zwei Gründe ausschlaggebend sein: Erstens trifft LDA eine implizite Annahme normalverteilter Daten, die bei der



(a) Mineralsortierung im SWIR **(b)** Paprika gegen Fremdkörper im VIS

Abbildung 3.4: ROC Kurven für die Klassifikation mit zwei Filtern.

Filtersimulation mit dem zentralen Grenzwertsatz begründbar ist. Zweitens ist sowohl bei AdaBoost als auch bei CLM eine Diskretisierung der Daten nötig, die mit Informationsverlust einhergeht. Diese Information wird bei der Selektion mit LDA nicht vernichtet und kann entsprechend genutzt werden. Abbildung 3.4 zeigt die ROC Kurven unter Verwendung zweier Filter. Da sich die ersten beiden selektierten Filter bei den CLM-Verfahren in beiden Experimenten nicht unterscheiden, wird hier nur eine ROC Kurve für alle diese Verfahren gezeigt. In Experiment (a) erreichen alle Methoden eine fast perfekte Selektion, jedoch ist die Selektion von AdaBoost den anderen Methoden leicht unterlegen. In Experiment (b) ist LDA den anderen Verfahren weit überlegen. Ab einer FPR von $\geq 5\,\%$ erreicht hier die Selektion durch AdaBoost bessere TPR als die von CLM.

3.1.9 Zusammenfassung

In diesem Abschnitt wurde gezeigt, wie das Problem der Filterselektion auf ein bekanntes Problem des maschinellen Lernens abgebildet werden kann: die Merkmalsselektion. Jedes Filter entspricht hierbei einem Merkmal, wobei die Filterantworten auf Grundlage von gemessenen spektralen Signaturen

simuliert werden. Anders als vergleichbare Verfahren ist der Ansatz nicht auf Bandpassfilter beschränkt. Vielmehr lassen sich beliebige Filter mit hochkomplizierten Transmissionskurven selektieren.
Im nächsten Schritt wurden Vertreter des Wrapper-, Filter- und des eingebetteten Ansatzes zur Merkmalsselektion vorgeschlagen. Der Wrapperansatz verwendet LDA als Klassifikator. Dadurch ist es möglich, die rechenaufwendige Evaluation des Klassifikators durch die Auswertung des Fisher-Kriteriums zu ersetzen. Als Filteransatz wurde das CLM Framework gewählt, das eine Modellierung von Vorwissen über die statistischen Abhängigkeiten zwischen Filtern erlaubt. Als eingebettetes Verfahren wurde AdaBoost verwendet.
Die Ansätze wurden anhand von zwei Sortierproblemen aus dem Bereich Bergbau sowie Lebensmittelanalyse evaluiert. In beiden Fällen liefert das LDA-Verfahren die besten Ergebnisse. Dies kann damit begründet werden, dass sowohl CLM als auch AdaBoost die Daten diskretisieren, wodurch nutzbare Information verloren geht. Verwendung von Entropieschätzverfahren für kontinuierliche Daten bei CLM oder eingebettete Merkmalsselektion mittels Real AdaBoost [FHT00] oder eines anderen nicht diskreten Boostingverfahrens könnten diese Informationen erhalten. Allerdings sind diese Ansätze mit teilweise erheblichem Rechenaufwand verbunden. Im Gegensatz dazu liefert der LDA-Ansatz bereits sehr gute Ergebnisse und kann durch Vorberechnung der Streumatrizen erheblich beschleunigt werden.
Der gezeigte Ansatz bezieht sich auf Selektion optischer Filter bzw. passender Beleuchtung. Im Prinzip können die hier gezeigten Verfahren aber auch auf andere Aspekte der Bilderfassung ausgeweitet werden, solange ein Klassifikationsproblem zugrunde liegt und sich die Effekte simulieren lassen.

3.2 Kostensensitive Merkmalsselektion

Wie bereits in Abschnitt 2.2 diskutiert wurde, ist die Verwendung diskriminativer Merkmale Voraussetzung für eine erfolgreiche Klassifikation. Bevor komplizierte Verfahren entworfen werden, sollte in der Praxis zunächst

auf bewährte Standardmerkmale (siehe Abschnitt 2.2.1) und Standardklassifikatoren wie der SVM gesetzt werden. Da aber zu viele Merkmale die Modellkomplexität erhöhen und somit die Interpretierbarkeit des Systems vermindern, und weil ein zu hochdimensionaler Merkmalsraum bei nicht ausreichendem Stichprobenumfang zur Verminderung der Klassifikationsgüte führt [Hug68], ist eine Auswahl geeigneter Merkmale unerlässlich. Für die Automatisierung dieser Auswahl wurden neben den Methoden, die im letzten Abschnitt vorgestellt wurden, eine große Anzahl an Verfahren vorgeschlagen. Eine Übersicht und eine systematische Einteilung vieler solcher Ansätze findet sich z. B. in Chandrashekar und Sahin [CS14].
Ziel der meisten dieser Methoden ist neben der Auswahl relevanter Merkmale die Minimierung des Umfangs der Selektion. Bei Correlation-based Feature Selection (CFS) [Hal99] (siehe Abschnitt 2.2.2) ist dieses Ziel explizit in der Gütefunktion kodiert:

$$U_{\text{CFS}}\left(\mathcal{S};\mathcal{D}\right) := \frac{k \cdot \overline{r_{\psi,\omega}}}{\sqrt{k + k\left(k+1\right)\overline{r_{\psi,\psi}}}}, \quad \text{mit } k := \left|\mathcal{S}\right|. \tag{3.27}$$

Hier bezeichnen $\overline{r_{\psi,\omega}}$ und $\overline{r_{\psi,\psi}}$ die zu maximierende mittlere Korrelation der Merkmale mit der Klasse bzw. die zu minimierende mittlere Korrelation zwischen den selektierten Merkmalen (siehe Gleichung 2.12). Da die Anzahl k der selektierten Merkmale stärker in den Nenner als in den Zähler eingeht, werden kleine Selektionen gegenüber ähnlich relevanten Selektionen mit mehr Merkmalen bevorzugt.
Eine kleine Selektion bedeutet allerdings nicht notwendigerweise auch eine Minimierung des benötigten Rechenaufwands. In Echtzeitsystemen, wie bei der visuellen Inspektion von Schüttgütern, wird durch dieses Vorgehen daher nicht notwendigerweise das eigentliche Ziel einer diskriminativen, möglichst schnell berechenbaren Selektion erreicht. Angenommen eine Selektion $\mathcal{S}_a$ bestehe aus einer geringeren Anzahl an Merkmalen als eine Selektion $\mathcal{S}_b$, also $|\mathcal{S}_a| < |\mathcal{S}_b|$. Die Relevanz der enthaltenen Merkmale sei gleich, aber die Merkmale in $\mathcal{S}_b$ verursachen geringeren Rechenaufwand als

die Merkmale in $\mathcal{S}_a$. Die meisten Selektionsverfahren, auch CFS, würden daher $\mathcal{S}_a$ bevorzugen, obwohl bei einem Echtzeitsystem $\mathcal{S}_b$ zum Einsatz kommen sollte. Bei der Schüttgutsortierung kann unter hoher Systemlast sogar eine geringere Klassifikationsleistung mit $\mathcal{S}_b$ in Kauf genommen werden, wenn dadurch die Echtzeitbedingungen eingehalten werden können. In solchen Systemen sollte neben dem mittleren Rechenaufwand auch die Varianz der Rechenzeit einbezogen werden, um die Wahrscheinlichkeit einer Deadlineverletzung zu minimieren. Das betrifft besonders Merkmale, bei denen die Objektgröße die Rechenzeit beeinflusst, also Farbmomente, Kompaktheit, Symmetriemerkmale etc.

3.2.1 Stand der Technik

Selektion mit Kosten. Die Idee, die Kosten der Merkmalsgewinnung bei der Selektion mit einzubeziehen, kam schon früh auf. Yang und Honavar [YH98] nutzen einen genetischen Algorithmus (siehe Abschnitt 2.3.2) zur Merkmalsselektion. Jedes Individuum der Population entspricht dabei einer Selektion, und die Fitness eines Individuums setzt sich aus der Klassifikationsgüte eines mit den Merkmalen trainierten neuronalen Netzes und den Kosten der Selektion zusammen. „Kosten" ist hierbei ein sehr allgemeiner Begriff, der z. B. monetäre Beschaffungskosten oder das Risiko einer medizinischen Behandlung kodiert. Eine Verbindung zu der für die Extraktion benötigten Rechenzeit der Merkmale stellen die Autoren allerdings nicht her. Dagegen ist die Minimierung der benötigten Rechenzeit das explizite Ziel von Iswandy und Koenig [IK06]. In Experimenten vergleichen die Autoren einen GA-Ansatz mit Particle Swarm Optimization (PSO) und kommen zum Schluss, dass PSO schneller konvergiert als der GA. Wie beim Verfahren von Yang und Honavar wird für beide Ansätze statt multikriterieller Optimierung ein einziges (skalares) Optimierungskriterium aus den beiden Zielen – Relevanzmaximierung und Kostenminimierung – abgeleitet.

Paclík et al. [Pac+02] beschreiben ein Greedy-Verfahren zur kostensensitiven Merkmalsselektion. Wie bei den GA- und PSO-Ansätzen berechnet sich die Güte eines Merkmals im Bezug auf die bereits selektierten Merkmale nicht nur aus der erwarteten Klassifikationsleistung, sondern auch aus den durch das Merkmal verursachten zusätzlichen Kosten. Dabei werden Abhängigkeiten zwischen Merkmalen beachtet: Merkmale, die sich aus anderen Merkmalen der Selektion ableiten lassen, verursachen geringere Kosten als Merkmale, die keine bereits berechneten Zwischenergebnisse nutzen können. Das Verfahren bevorzugt somit effektiv weniger relevante Merkmale, die geringe Gesamtkosten erzeugen, gegenüber teuren, hochrelevanten Merkmalen. Plasberg und Kleijn [PK09] verfolgen einen ähnlichen Ansatz, indem sie das MRMR-Verfahren von Peng et al. [PLD05] um einen Term erweitern, der die Komplexität der Berechnung der Selektion bewertet. Die Gewichtung der verschiedenen Terme wird in einer vorgeschalteten Gittersuche festgelegt.

Multikriterielle Formulierung. All diesen Ansätzen ist gemein, dass die verschiedenen Optimierungsziele in einer einzigen skalaren Gütefunktion kombiniert werden. Es gibt aber auch Merkmalsselektionsverfahren, die sich eine gleichzeitige, multikriterielle Optimierung dieser Ziele vornehmen.
Morita et al. [Mor+03] formulieren zwei Kriterien, die mittels NSGA multikriteriell optimiert werden. Das erste Kriterium maximiert den Abstand von Clustern im Merkmalsraum als Stellvertreter für die erwartete Klassifikationsleistung, während das zweite Kriterium die Anzahl der selektierten Merkmale minimiert. Hamdani et al. [Ham+07] minimieren den Klassifikationsfehler eines kNN-Klassifikators und die Größe der Selektion mittels Nondominated Sorting Genetic Algorithm II (NSGA-II). Dieser Ansatz wird auch von Tekguc et al. [TSD09] für die Klassifikation von Gesichtsausdrücken verwendet. Allerdings wird hier statt der Klassifikationsgüte eines Klassifikators das Fisher-Kriterium, also das Verhältnis aus Inter- und Intraklassendistanz, maximiert.

Xue et al. [XZB13] vergleichen verschiedene multikriterielle Optimierungsverfahren, darunter GA- und PSO-Methoden, im Kontext der Merkmalsselektion. Wie bei den anderen hier vorgestellten Ansätzen werden zwei Optimierungsziele verfolgt: eine hohe Klassifikationsgüte und eine kleine Selektion. Saroj und Jyoti [SJ14] formulieren hingegen drei Optimierungsziele: Informationsgewinn, Redundanzfreiheit und Minimierung der Anzahl der selektierten Merkmale. Diese Ziele werden mittels NSGA-II optimiert. Der Vergleich der Selektionsergebnisse mit einer skalaren, GA-basierten Optimierung brachte allerdings keine eindeutigen Ergebnisse.

3.2.2 Beitrag zum Stand der Technik

Alle oben vorgestellten Methoden beziehen entweder die Kosten der Merkmalsgewinnung in ein skalares Optimierungsziel ein oder verwenden multikriterielle Optimierung ohne Beachtung der Kosten. Kein Verfahren kombiniert die multikriterielle Optimierung mit dem Ziel der Kostenminimierung. In diesem Abschnitt wird daher ein Merkmalsselektionsverfahren vorgestellt, das diese Lücke schließt. Neben der Relevanz der Merkmale wird auch der mittlere Rechenaufwand und die Streuung des Rechenaufwands in die Optimierung einbezogen. Das multikriterielle Optimierungsproblem wird mit Hilfe von NSGA-II [Deb+02] optimiert. Das Ergebnis der Optimierung ist nicht eine einzige Selektion, sondern eine Menge von Pareto-optimalen Selektionen, also Selektionen, die alle anderen Selektionen in mindestens einem Gütemaß übertreffen. Aus dieser Menge an Selektionen kann je nach Anforderungen oder sogar anhand der momentanen Systemlast eine passende Selektion ausgewählt werden. Dieses Verfahren wurde in Richter et al. [Ric+16a] veröffentlicht.

3.2.3 Kostenminimierung

Zur Minimierung der Kosten muss der Begriff zunächst noch präziser gefasst werden. „Kosten” ist hier ein allgemeiner Begriff und kann beispielsweise

den gemessenen Rechenaufwand in der Anzahl benötigter CPU-Zyklen oder den monetären Aufwand eines Versuchs (Materialen und Arbeitszeit) repräsentieren. Formal bezeichnet $c\left(\psi_d\left(\boldsymbol{p}_n\right)\right)$ allgemein die nicht weiter spezifizierten, aber messbaren Kosten der Extraktion von Merkmal ψ_d aus dem Muster $\boldsymbol{p}_n$.
Anhand der Kosten $c\left(\psi_d\left(\boldsymbol{p}_n\right)\right)$ der Lernstichprobe $\mathcal{D}$ können die mittleren Kosten sowie die Streuung der Kosten geschätzt werden:

$$\overline{c}\left(\psi_d\right) = \frac{1}{N}\sum_{n=1}^{N} c\left(\psi_d\left(\boldsymbol{p}_n\right)\right) \quad \text{und} \tag{3.28}$$

$$\sigma_c\left(\psi_d\right) = \sqrt{\frac{1}{N-1}\sum_{n=1}^{N}\left(c\left(\psi_d\left(\boldsymbol{p}_n\right)\right) - \overline{c}\left(\psi_d\right)\right)^2}. \tag{3.29}$$

Zur Vereinfachung der Notation sei $\overline{\boldsymbol{c}} = \left(\overline{c}\left(\psi_1\right), \ldots, \overline{c}\left(\psi_D\right)\right)^\top \in \mathbb{R}^D$ der Vektor der mittleren Kosten und $\boldsymbol{\sigma}_c = \left(\sigma_c\left(\psi_1\right), \ldots, \sigma_c\left(\psi_D\right)\right)^\top \in \mathbb{R}^D$ der Vektor der Kostenstreuung der Merkmale. Die Selektion $\mathcal{S}$ wird ebenfalls durch einen Vektor

$$\boldsymbol{s} = \left(\mathbb{1}[\psi_1 \in \mathcal{S}], \ldots, \mathbb{1}[\psi_D \in \mathcal{S}]\right)^\top \in \{0,1\}^D \tag{3.30}$$

repräsentiert, wobei $\mathbb{1}[\psi \in \mathcal{S}]$ die Mengenzugehörigkeit zu $\mathcal{S}$ anzeigt, d. h. es gilt $\mathbb{1}[\psi \in \mathcal{S}] = 1$, wenn ψ selektiert wurde und $\mathbb{1}[\psi \in \mathcal{S}] = 0$ sonst.
Mit diesen Definitionen und mit der Annahme nicht untereinander korrelierter Kosten lassen sich die erwarteten Gesamtkosten und die Gesamtkostenstreuung kompakt als $\boldsymbol{s}^\top\overline{\boldsymbol{c}}$ bzw. $\boldsymbol{s}^\top\boldsymbol{\sigma}_c$ schreiben. Zusammen mit einem Kriterium $u(\boldsymbol{s};\mathcal{D})$, das die Relevanz der Selektion $\mathcal{S}$ ohne Berücksichtigung der Anzahl oder der Kosten der Merkmale in $\mathcal{S}$ bewertet, lässt sich ein zu maximierendes Gesamtkriterium definieren:

$$U\left(\mathcal{S};\mathcal{D}\right) = u(\boldsymbol{s};\mathcal{D}) - \lambda\boldsymbol{s}^\top\overline{\boldsymbol{c}} - \mu\boldsymbol{s}^\top\boldsymbol{\sigma}_c, \quad \text{mit } \lambda,\mu \geq 0. \tag{3.31}$$

Eine Maximierung dieses Kriteriums bedeutet eine Maximierung der Relevanz der Merkmale bei gleichzeitiger Minimierung der erwarteten Kosten sowie die Kostenstreuung und damit des Risikos einer Deadlineverletzung. Passende Relevanzkriterien für $u(\boldsymbol{s}; \mathcal{D})$ werden in Abschnitt 3.2.6 definiert. Dieser Ansatz umfasst neben der kostensensitiven Merkmalsselektion auch die konventionelle Minimierung der Anzahl der selektierten Merkmale. Dafür werden die Kosten $c\left(\psi_d\left(\boldsymbol{p}_n\right)\right)$ für alle Merkmale ψ_d und Muster $\boldsymbol{p}_n$ gleich gewählt, wodurch $\overline{\boldsymbol{c}} \propto \mathbf{1}$ und $\boldsymbol{\sigma}_c = \mathbf{0}$ gilt.

Offen bleibt jedoch, wie die Parameter λ und μ der Gleichung (3.31) sinnvoll festgelegt werden können. Diese Wahl ist nicht intuitiv, da passende Werte von der verwendeten Gütefunktion sowie vom zugrunde liegenden Klassifikationsproblem abhängen. Soll eine möglichst schnell berechenbare, aber möglicherweise weniger relevante Selektion einer relevanteren, aber aufwendigeren Selektion vorgezogen werden? Wie hoch soll das Risiko einer Deadlineverletzung bewertet werden?

Eine allgemeingültige Lösung kann nicht gefunden werden. Stattdessen wird in dieser Arbeit das kombinierte Kriterium in die drei Bestandteile – Bewertung der Relevanz, der Kosten und des Risikos der Deadlineverletzung – aufgeteilt. Da diese Ziele als gleichwertig betrachtet werden, gibt es nicht eine optimale Selektion, sondern eine Menge Pareto-optimaler Selektionen. Pareto-optimale Selektionen sind Selektionen, die in mindestens einem Kriterium eine bessere Bewertung erzielen als alle anderen Selektionen (siehe Abbildung 3.5). Es obliegt den Benutzern, nach der automatischen Selektion eine für die Situation angemessene Wahl zu treffen. Diese Wahl kann auch teilweise automatisiert erfolgen, indem z. B. die Systemlast zur Laufzeit bewertet und eine Selektion verwendet wird, mit der Deadlineverletzungen vermieden werden.

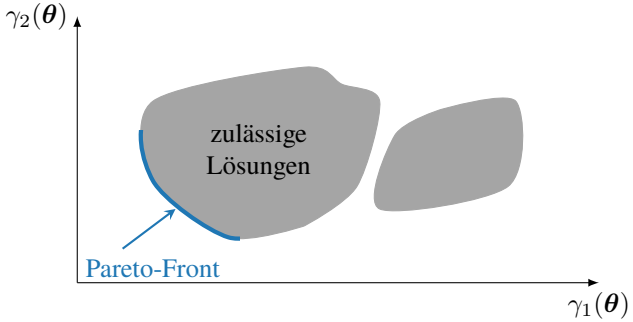


Abbildung 3.5: Skizze einer Lösungsmenge und Pareto-Front bei der multikriteriellen Minimierung von $\boldsymbol{\gamma}(\boldsymbol{\theta}) = (\gamma_1(\boldsymbol{\theta}), \gamma_2(\boldsymbol{\theta}))^\top$ mit Nebenbedingungen. Anders als hier gezeigt ist die Pareto-Front ist nicht notwendigerweise zusammenhängend.

3.2.4 Multikriterielle Optimierung mit NSGA-II

Zur multikriteriellen Optimierung gibt es eine Reihe von Lösungsansätzen. Die Maximierung von Gleichung (3.31) mit verschiedenen Werten für λ und μ ist ein naheliegender Ansatz. Dabei stellt sich allerdings die Frage, wie der Wertebereich dieser Parameter sinnvoll begrenzt und abgetastet werden kann. Auch kann eine nichtkonvexe Pareto-Front, d. h. die Menge der Pareto-optimalen Lösungen, durch diese Linearisierung nicht vollständig repräsentiert werden. Eine globale Optimierung mit GAs generiert dagegen eine Reihe von Lösungskandidaten, die sich der Pareto-Front annähern. Statt einer Linearisierung wird hier deswegen der vielfach erprobte genetische Algorithmus NSGA-II von Deb et al. [Deb+02] zur multikriteriellen Optimierung eingesetzt.

Die Grundidee von NSGA-II ist, die Fitness eines Individuums anhand des Nondomination Rank und der Crowding Distance zu bewerten. Der Nondomination Rank gibt an, wie weit ein Individuum von der Pareto-Front entfernt ist. Dafür wird für jeden Kandidaten $\boldsymbol{\theta}_c \in \mathcal{P}$ die Anzahl der

Individuen der Population $\boldsymbol{\theta}_o \in \mathcal{P}, \boldsymbol{\theta}_o \neq \boldsymbol{\theta}_c$, gezählt, die den Kandidaten in allen Kriterien dominieren,

$$N_{\boldsymbol{\theta}_c}^{\text{dom}} = \sum_{\substack{\boldsymbol{\theta}_o \in \mathcal{P} \\ \boldsymbol{\theta}_o \neq \boldsymbol{\theta}_c}} \mathbb{1}[\boldsymbol{\gamma}(\boldsymbol{\theta}_o) \prec \boldsymbol{\gamma}(\boldsymbol{\theta}_c)]\,, \tag{3.32}$$

wobei $\boldsymbol{\gamma}(\cdot)$ die multikriterielle Zielfunktion bezeichnet und $\boldsymbol{p} \prec \boldsymbol{q}$ genau dann gilt, wenn $[\boldsymbol{p}]_d < [\boldsymbol{q}]_d$ für alle $d = 1, \ldots, D$. Die Individuen einer Population werden anhand von $N_{\boldsymbol{\theta}_c}^{\text{dom}}$ sortiert und schließlich mit einem Nondomination Rank assoziiert. Individuen mit $N_{\boldsymbol{\theta}_c}^{\text{dom}} = 0$ erhalten den Nondomination Rank 1, Individuen mit $N_{\boldsymbol{\theta}_c}^{\text{dom}} = 1$ erhalten den Nondomination Rank 2 usw.
Der Nondomination Rank wäre ausreichend, um eine Fitnessfunktion zu definieren. Damit die Population aber einen möglichst großen Bereich der Pareto-Front abdeckt, werden Individuen gleichen Nondomination Ranks jedoch zusätzlich anhand der Populationsdichte um die Individuen bewertet. Individuen in dünn besetzten Regionen des Parameterraums werden dadurch als fitter bewertet als Individuen in dicht besiedelten Regionen. Die Crowding Distance ist ein empirisches Dichtemaß um $\boldsymbol{\theta}_c$ und proportional zur Summe der Kantenlängen des Hyperquaders, das von dem im Parameterraum elementweise nächstkleineren bzw. -größeren Individuum $\boldsymbol{\theta}_b$ und $\boldsymbol{\theta}_d$ gleichen Nondomination Ranks, also von $\boldsymbol{\theta}_b \prec \boldsymbol{\theta}_c \prec \boldsymbol{\theta}_d$ mit $N_{\boldsymbol{\theta}_b}^{\text{dom}} = N_{\boldsymbol{\theta}_c}^{\text{dom}} = N_{\boldsymbol{\theta}_b}^{\text{dom}}$, aufgespannt wird.
Deb et al. definieren effiziente Algorithmen zur Berechnung von Nondomination Rank und Crowding Distance, die im wesentlichen auf einer schnellen Berechnung von $N_{\boldsymbol{\theta}_c}^{\text{dom}}$ und einer Sortierung der Individuen anhand dieser Maße basieren. Details und eine Diskussion der Laufzeit dieser Algorithmen finden sich in Deb et al. [Deb+02].

3.2.5 Kodierung und genetische Operationen

Die Kodierung der Selektion für NSGA-II ist einfach: Das Genom eines Individuums ist der Selektionsvektor aus Gleichung (3.30), also $\boldsymbol{\theta} = \boldsymbol{s}$. Die

genetischen Operationen RECOMBINE und MUTATE entsprechen den kanonischen Definitionen in Gleichungen (2.23) und (2.25) aus Abschnitt 2.3.2. Zur Selektion der Eltern wird Tournament Selection (siehe Gleichung 2.22) verwendet.
Die Zielfunktion $\boldsymbol{\gamma}(\cdot)$ umfasst die gleichen Kriterien wie Gleichung (3.31), d. h. Maximierung der Relevanz und Minimierung der mittleren Kosten und Kostenstreuung. Da NSGA-II alle Kriterien minimiert, ist die multikriterielle Zielfunktion

$$\boldsymbol{\gamma}(\boldsymbol{s}) := \left(-u(\boldsymbol{s};\mathcal{D}), \boldsymbol{s}^\top \overline{\boldsymbol{c}}, \boldsymbol{s}^\top \boldsymbol{\sigma}_c\right)^\top. \tag{3.33}$$

In dieser Zielfunktion wird implizit angenommen, dass die Kosten eines Merkmals unabhängig von den anderen Merkmalen in der Selektion sind. Das ist natürlich eine Vereinfachung, da in einem realen System Zwischenergebnisse bei der Berechnung eines Merkmals wiederverwendet werden können. Um die Notation einfach zu halten, wurde hier jedoch auf eine Modellierung dieser Abhängigkeiten verzichtet. Sollen diese Abhängigkeiten beachtet werden, bietet sich beispielsweise ein Berechnungsgraph oder ein empirischer Ansatz wie in Paclík et al. [Pac+02] an.

3.2.6 Gütefunktionen

Zur Bewertung der Relevanz einer Selektion bieten sich verschiedene Gütemaße $u(\boldsymbol{s};\mathcal{D})$ an. Hier werden ein Filter- und ein Wrapperansatz vorgestellt.

Filter. Der Filteransatz orientiert sich an CFS [Hal99]: Die Relevanz der Merkmale wird anhand der Korrelation zwischen den selektierten Merkmalen und der Klassen bewertet. Anders als bei CFS wird hierzu jedoch der multiple Korrelationskoeffizient verwendet, also

$$u_{\text{corr}}(\boldsymbol{s};\mathcal{D}) := \sqrt{R^2} = \sqrt{\boldsymbol{r}_{\mathcal{S},\omega}^\top \boldsymbol{R}_{\mathcal{S},\mathcal{S}}^{-1} \boldsymbol{r}_{\mathcal{S},\omega}}. \tag{3.34}$$

Mit dem empirischen Korrelationskoeffizienten $r_{s,t}$ der Stichproben $(s_n)_{n=1}^N$ und $(t_n)_{n=1}^N$ beschreibt der Vektor

$$\boldsymbol{r}_{\mathcal{S},\omega} = \left(r_{\psi_1,\omega}, \ldots, r_{\psi_K,\omega}\right)^\top \tag{3.35}$$

dabei die Korrelation der selektierten Merkmale $\psi_1, \ldots, \psi_K \in \mathcal{S}$ mit der Klasse ω und die Matrix

$$\boldsymbol{R}_{\mathcal{S},\mathcal{S}} = \begin{pmatrix} r_{\psi_1,\psi_1} & \cdots & r_{\psi_1,\psi_K} \\ \vdots & \ddots & \vdots \\ r_{\psi_K,\psi_1} & \cdots & r_{\psi_K,\psi_K} \end{pmatrix}, \tag{3.36}$$

beschreibt die paarweise Korrelation aller selektierten Merkmale in $\mathcal{S}$. Anders als CFS und andere Filteransätze bestraft dieses Kriterium explizit nicht die Korrelation zwischen den selektierten Merkmalen. Das bedeutet, dass redundante Merkmale selektiert werden können, solange sie zumindest schwach relevant sind und solange der Einfluss auf die anderen Optimierungskriterien gering ist.

Wrapper. Der Wrapperansatz orientiert sich an dem Wrapperansatz aus Abschnitt 3.1.4. Damit das Selektionsverfahren nicht auf binäre Klassifikation beschränkt ist, stützt sich das Maß hier jedoch nicht auf LDA, sondern auf die multiple Diskriminanzanalyse (MDA, siehe z. B. Beyerer et al. [BRN17], S. 84–86).
Anders als bei LDA gibt es für MDA mehrere mögliche Optimierungsfunktionale. Hier wurde das Kriterium aus Bishop [Bis13] verwendet. Mit den Streumatrizen innerhalb der Klassen $\boldsymbol{S}_W$ und zwischen den Klassen $\boldsymbol{S}_B$

sowie der Projektionsmatrix $\boldsymbol{W}$ als zu optimierender Parameter, ist dieses Kriterium folgendermaßen definiert:

$$u_{\text{MDA}}(\boldsymbol{s};\mathcal{D}) := \text{spur}\left(\left(\boldsymbol{W}\boldsymbol{S}_W\boldsymbol{W}^\top\right)^{-1}\left(\boldsymbol{W}\boldsymbol{S}_B\boldsymbol{W}^\top\right)\right). \tag{3.37}$$

Dieses Kriterium wird von der Matrix $\boldsymbol{W}^*$ maximiert, die aus den C Eigenvektoren zusammengesetzt ist, die zu den größten Eigenwerten von $\boldsymbol{S}_W^{-1}\boldsymbol{S}_B$ gehören [Bis13]. Wie beim Filterkriterium wird auch hier die Größe der Selektion und Redundanz zwischen den Merkmalen nicht bestraft.

3.2.7 Experimente und Ergebnisse

Der beschriebene Ansatz wurde mit Hilfe von einem synthetischen Datensatz und zwei realen Datensätzen evaluiert, die für typische Sortierprobleme stehen.

Der synthetische Datensatz wurde aus dem MADELON-Datensatz der 2003 NIPS Feature Selection Challenge von Guyon et al. [Guy+05] gewonnen. Der Datensatz besteht aus 2600 Beispielen mit jeweils 500 Merkmalen und zwei Klassen. Guyon et al. konstruierten diesen Datensatz dabei so, dass $96\,\%$ der Merkmale keine Klasseninformation tragen und dass die restlichen $4\,\%$ der Merkmale nur in Kombination mit anderen relevanten Merkmalen informativ sind. Das bedeutet, dass nur eine Kombination aus den 20 informativen Merkmalen zur Klassifikation befähigt. Weitere Informationen über die Erstellung und Zusammensetzung des Datensatzes finden sich in Guyon et al. [Guy+05].

Da der MADELON-Datensatz keine Selektionskosten enthält, wurden diese für die Experimente zufällig generiert. Die Kosten $c(\psi_d(\boldsymbol{p}_n)) \sim \mathcal{N}(\mu_d, \sigma_d^2)$ wurden anhand einer Normalverteilung mit Erwartungswert μ_d und Varianz σ_d^2 generiert. Die Parameter der Normalverteilungen wurden wiederum aus Gleichverteilungen gezogen, $\mu_d \sim \mathfrak{U}(0;100)$ und $\sigma_d \sim \mathfrak{U}(0{,}01;5)$ für $d = 1,\ldots,D$. Die Relevanz der Merkmale wurde dabei nicht betrachtet.

Tabelle 3.2: Zugrundeliegende Rohdaten der realen Datensätze für die Evaluation. Die Größe der Lego-Steine entspricht der Anzahl und Anordnung der Noppen.

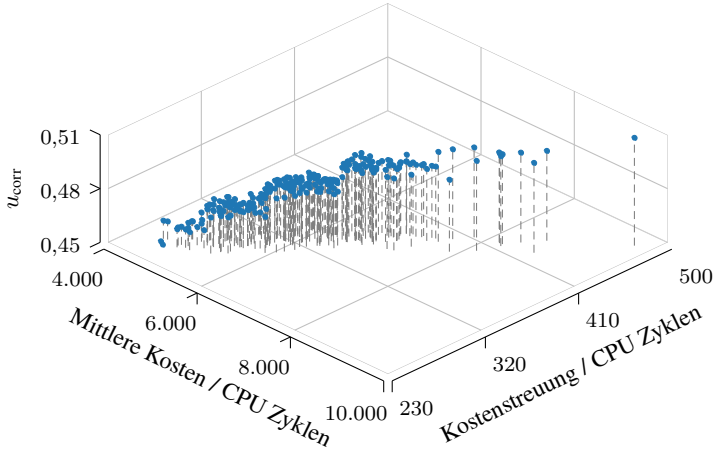
Datensatz	Klasse	#	Beispiel
Lego	Basic Steine, Größe 2×2	380	
Lego	Basic Steine, Größe 2×3	330	
Lego	Basic Steine, Größe 2×4	438	
Lego	Basic Steine, Größe 1×1 bis 1×8	367	
Lego	Technik Kreuzprofilachsen verschiedener Länge	468	
Lego	Technik Steine, Größe 1×1 bis 1×16	371	
Lego	Technik Verbinder symmetrisch	415	
Lego	Technik Verbinder asymmetrisch	366	
Kiesel	Runde graue Steine mit dunklen Flecken	1213	
Kiesel	Blau-graue Steine mit hellen Flecken	3418	

Zusätzlich wurden auf Grundlage realer Daten zwei Datensätze erstellt, die typische Sortierprobleme repräsentieren. Dafür wurden mit einem Schüttgutsortiersystem des Fraunhofer-IOSB Bilder von Lego-Steinen und von Kieselsteinen aufgenommen. Eine Übersicht dieser Datensätze und Beispiele der aufgenommenen Muster finden sich in Tabelle 3.2. Für jedes Objekt wurden 44 Standardmerkmale wie die Fläche, Kompaktheit, Länge und Breite, Farbmomente etc. extrahiert und jeweils die dafür benötigte Rechenzeit in CPU-Zyklen gemessen.

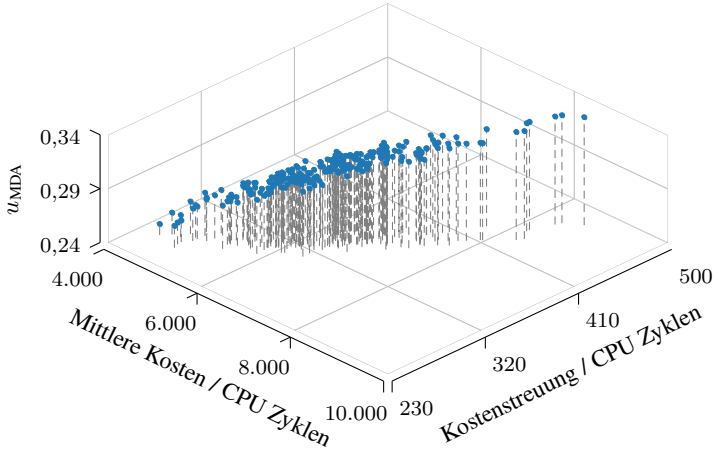
Der Lego-Datensatz enthält insgesamt 3135 Beispiele aus acht Klassen etwa gleicher Größe und dient als Repräsentant der Sortierung maschinell hergestellter Gegenstände. Bei der Sortierung solcher Objekte ist die Varianz

innerhalb der Klassen in der Regel klein. Allerdings ist die Anzahl der Klassen groß und es besteht eine hohe Ähnlichkeit zwischen manchen Klassen. Bei ungünstigem Blickwinkel lassen sich die Objekte dieser Klassen nicht unterscheiden. Hier sind z. B. die ersten drei Klassen (siehe Tabelle 3.2) nicht unterscheidbar, wenn die Steine entlang der kurzen Seite betrachtet werden. Ebenso können die einreihigen Lego Basic Steine von oben betrachtet nicht von einreihigen Lego Technik Steinen unterschieden werden, da sich diese nur durch quer verlaufende Löcher in den Steinen unterscheiden.
Der Kiesel-Datensatz enthält 4630 Beispiele zweier Klassen von Steinen, wobei die zweite Klasse etwa dreimal so groß ist wie die erste Klasse. Dieser Datensatz dient als Beispiel für die Sortierung natürlicher Objekte, bei denen die Datensätze oft stark unbalanciert sind. Ebenso ist die Variation innerhalb der Klassen sowie der Abstand zwischen den Klassen bei solchen Objekten oft größer als bei technisch hergestellten Objekten.

Ergebnisse. Der Ansatz wurde mit variierender Populationsgröße und Anzahl an Generationen getestet. Da sich bei den Versuchen herausstellte, dass die Populationsgröße nur geringen Einfluss auf die Ergebnisse hat (ausgenommen Konvergenzgeschwindigkeit), wird im Folgenden lediglich auf die Ergebnisse von Versuchen mit 250 Individuen eingegangen.
Abbildung 3.6 zeigt die Pareto-Front der Selektion im MADELON-Datensatz nach 500 Generationen des genetischen Algorithmus. In Abbildung 3.6a wurde das Korrelationskriterium verwendet, in Abbildung 3.6b das MDA-Kriterium. Die Pareto-Fronten beider Kriterien ähneln sich im Sinne der Verteilung der mittleren Kosten und Kostenstreuung. Allerdings ist beim Korrelationskriterium eine treppenartige Struktur des Gütemaßes u_{corr} erkennbar, die beim MDA-Kriterium deutlich schwächer ausgeprägt ist. In beiden Fällen gibt es jedoch keine klare beste Lösung.
Die Pareto-Fronten der beiden anderen Datensätze sind in Abbildung 3.7 zu sehen. Für eine bessere Übersichtlichkeit wurde auf die Visualisierung der Kostenstreuung verzichtet – diese hängen hier ohnehin annähernd linear mit

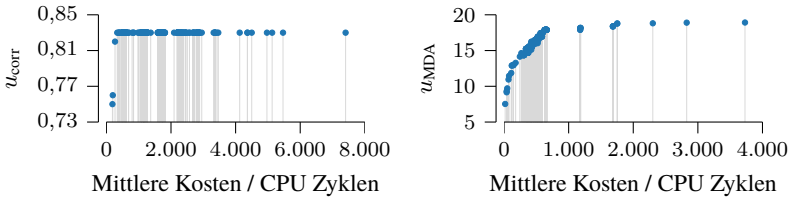


(a) u_{corr}

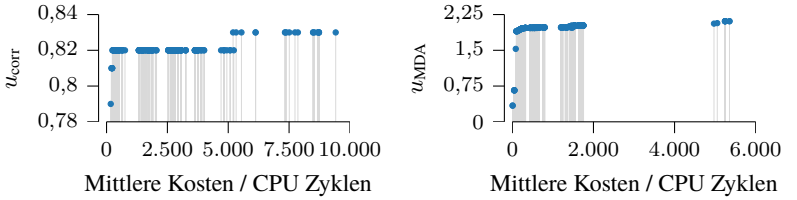


(b) u_{MDA}

Abbildung 3.6: Pareto-Front aus 250 Individuen nach 500 Generationen (MADELON-Datensatz).



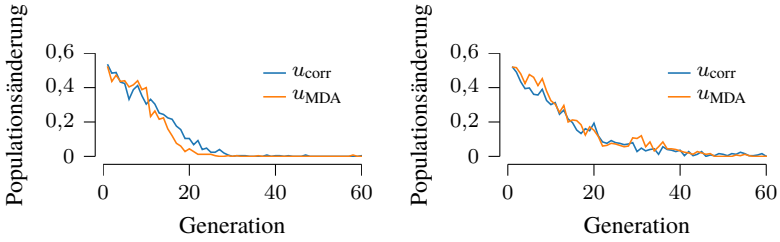
(a) Lego-Datensatz



(b) Kiesel-Datensatz

Abbildung 3.7: Auf zwei Kriterien reduzierte Pareto-Fronten aus 250 Individuen nach 100 Generationen.

den mittleren Kosten zusammen. Wie beim MADELON-Datensatz sind mit dem Korrelationskriterium Diskontinuitäten bzw. Sprünge in der Pareo-Front erkennbar. Mit diesem Kriterium gibt es beim Lego-Datensatz eine klar zu bevorzugende Selektion. Beim Kieseldatensatz ist die Wahl effektiv auf zwei Lösungen mit unterschiedlichen mittleren Kosten beschränkt. Das MDA-Kriterium bewertet die unterschiedlichen Lösungen insbesondere beim Lego-Datensatz deutlich differenzierter. Zusätzlich werden mit diesem Kriterium auch günstigere Selektionen gefunden als mit dem Korrelationskriterium.
Abbildung 3.8 zeigt den Anteil der ausgetauschten Individuen bzw. die relative Änderung der Population pro Generation für die Experimente mit realen Daten. Mit beiden Datensätzen konvergiert der Algorithmus nach weniger als 100 Iterationen, wobei die Pareto-Front mit dem Lego-Datensatz schneller gefunden wird als mit dem Kiesel-Datensatz. Mit beiden Kriterien konvergiert der Algorithmus annähernd gleich schnell, obwohl sich die



(a) Lego-Datensatz **(b)** Kiesel-Datensatz

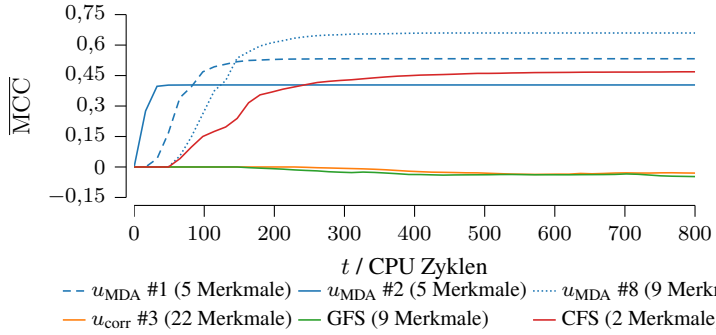
Abbildung 3.8: Relative Änderung der Population pro Generation mit einer Populationsgröße von 250 Individuen.

Pareto-Fronten und auch die Individuen der jeweiligen Populationen deutlich unterscheiden. Mit dem MADELON-Datensatz, der hier nicht gezeigt wird, konvergiert die Selektion langsamer. Hier ändert sich die Population erst nach ca. 150 Generationen nicht mehr signifikant. Ein Grund könnte der mit 500 Merkmalen etwa 11 mal größere Parameterraum als bei den Experimenten mit realen Daten und 44 Merkmalen sein. Zudem ist dieser Datensatz sehr komplex, da er, wie oben beschrieben, so konstruiert wurde, dass die Selektion der relevanten Merkmale erschwert wird.

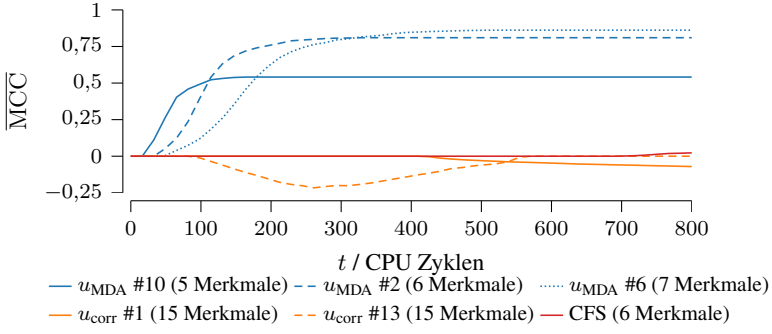
Unter Beachtung des Anwendungsfalls wurde zur Bewertung der Selektionsgüte eine Klassifikation unter Echtzeitbedingungen simuliert. Dafür wurden in einer stratifizierten 5-fold Cross Validation Klassifikation lineare soft margin SVMs mit $C = 1$ für jeden Lösungskandidaten trainiert. Zur Mehrklassenklassifikation im Lego-Datensatz wurde das one-vs-all Schema verwendet. Die Echtzeitanforderungen wurden durch eine variable Deadline simuliert. Für eine gegebene Deadline t (in CPU-Zyklen) wurden Beispiele $\boldsymbol{p}_n$ der Lernstichprobe als falsch klassifiziert gewertet, wenn die Gesamtkosten $c_{\boldsymbol{s},n} = (c(\psi_1(\boldsymbol{p}_n)), \ldots, c(\psi_D(\boldsymbol{p}_n)))\boldsymbol{s}$ der Selektion $\boldsymbol{s}$ größer als die Deadline sind, d. h. wenn $c_{\boldsymbol{s},n} > t$.

Zusätzlich zu der multikriteriellen Selektion mit NSGA-II wurde zum Vergleich auch Merkmalsselektion mit CFS implementiert. Dafür wurden, ähnlich

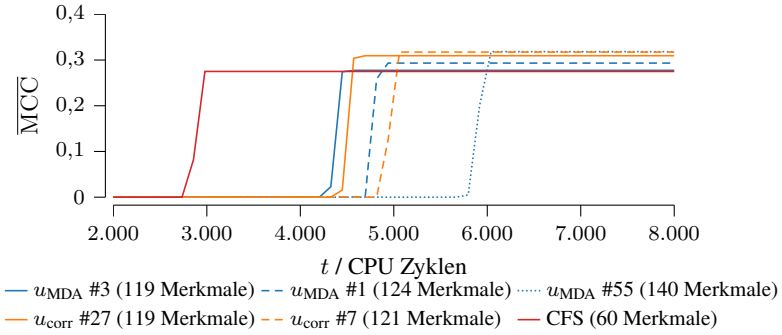
wie bei der Greedy-Selektion, der Selektion iterativ diejenigen noch nicht selektierten Merkmale hinzugefügt, die das Gütemaß aus Gleichung (3.27) maximieren. Die Iteration wurde nach einer vordefinierten Anzahl an Selektionsrunden abgebrochen. Im Anschluss wurde diejenige Selektion verwendet, die das Gütemaß maximierte. Da bei diesem Verfahren die Kosten der Merkmale nicht betrachtet wurden, ist hier ein schlechteres Deadlineverhalten wahrscheinlich. Der Vergleich dient somit als zu schlagende Baseline.
Für den Lego-Datensatz wurde zusätzlich die Selektion nach dem GFS-Ansatz von Paclík et al. [Pac+02] implementiert. Da keine Informationen über die Abhängigkeiten zwischen den Merkmalen oder geeignete Messungen zum Ableiten dieser Abhängigkeiten vorlagen, wurde auf die von Paclík et al. beschriebene Gruppierung der Merkmale verzichtet. Stattdessen wurden wie beim CFS-Verfahren Merkmale in einem Greedy-Verfahren hinzugefügt, bis die Selektion eine vorgegebene Größe erreichte. Für die Evaluation wurden allerdings alle Zwischenergebnisse und nicht nur die finale Selektion betrachtet.
Abbildung 3.9 zeigt das simulierte Deadlineverhalten einer Auswahl der Selektionsergebnisse. Wie sich bereits in Abbildung 3.7 abzeichnete, sind die Selektionen mit dem MDA- und dem Korrelationskriterium sehr unterschiedlich. Mit den realen Datensätzen werden mit dem MDA-Kriterium gute Lösungen mit unterschiedlichem Deadlineverhalten gefunden: schnell berechenbare, aber weniger diskriminative Selektionen und relevantere Selektionen, die aber kurze Deadlines deutlich häufiger verletzen. Die mit dem Korrelationskriterium gefundenen Selektionen wurden allerdings mit beiden Datensätzen erst bei langen Deadlines von $t > 800$ Zyklen (nicht gezeigt) relevant. Hier wurden also deutlich teurere Selektionen gewählt als mit dem MDA-Kriterium. Überraschenderweise sind die mit dem GFS-Verfahren gefundenen Selektionen ebenfalls gänzlich ungeeignet. Bis zu einer Deadline von $t = 800$ Zyklen ist die Klassifikation mit diesen Selektionen nicht besser als ein Zufallsentscheid.



(a) Lego-Datensatz



(b) Kiesel-Datensatz



(c) MADELON-Datensatz

Abbildung 3.9: Mittlererer MCC in Abhängigkeit der simulierten Deadline.

Die Selektion mit CFS ist bei beiden Datensätzen der multikriteriellen Selektion unterlegen. Dies war zu erwarten, da hier die Selektionskosten nicht bewertet wurden. Dass Minimierung der Selektionsgröße nicht notwendigerweise die Rechenzeit minimiert, wird in Abbildung 3.9a illustriert. Obwohl durch CFS beim Lego-Datensatz nur zwei Merkmale selektiert wurden, werden mit der hier beschriebenen Methode Selektionen mit fünf bzw. neun Merkmalen gefunden, die schneller berechenbar und damit bei kurzen Deadlines besser für die Klassifikation geeignet sind. Beim Kiesel-Datensatz wird die Selektion mit CFS erst ab einer Deadline von $t \approx 700$ Zyklen relevant, während die Selektion mit dem MDA-Kriterium bereits bei $t \approx 180$ ein $\mathrm{MCC} > 0{,}75$ erzielt wird.

Mit beiden Datensätzen zeigt sich aber auch, dass je nach Deadlinesituation eine andere Selektion bevorzugt werden sollte. In Abbildung 3.9a sollte beispielsweise bei einer hohen Systemlast die Selektion $\#2$ mit fünf Merkmalen und bei geringer Systemlast die Selektion $\#3$ mit neun Merkmalen verwendet werden. Durch eine vorgeschaltete Deadlinesimulation und eine Messung der CPU-Last im Betrieb lässt sich eine solche Auswahl auch zur Laufzeit automatisieren.

Die Selektion mit dem synthetischen MADELON-Datensatz zeigt ein deutlich anderes Bild. Zum einen werden hier mit dem Korrelationskriterium ähnlich gute Selektionen gefunden wie mit dem MDA-Kriterium. Aus diesen Experimenten wird also nicht klar, ob das MDA-Kriterium dem Korrelationskriterium überlegen ist. Zum anderen ist CFS den anderen Verfahren deutlich überlegen, obwohl diese Verfahren im Unterschied zu CFS die Kosten der Selektion mit einbeziehen. Diese Verfahren finden erst ab einer Deadline von $t \approx 4500$ Zyklen ähnlich geeignete Selektionen, und diese Selektionen enthalten deutlich mehr Merkmale als die mit CFS gefundene Selektion.

Allerdings ist bei diesem Datensatz keine der gefundenen Selektionen zur Klassifikation geeignet. Mit allen Selektionen wird selbst bei $t = 8000$ Zyklen kein mittlerer $\overline{\mathrm{MCC}} > 0{,}35$ erreicht. Der Grund liegt vermutlich in der oben beschriebenen Struktur des Datensatzes: Nur 20 der 500 Merkmale

sind relevant und kein Merkmal ist für sich genommen informativ. Außerdem wurden die Kosten der Merkmalsgewinnung zufällig und unabhängig von der Relevanz der Merkmale erzeugt. Bei den Verfahren mit Bewertung der Kosten entspricht dies einer Rauschquelle, die die Selektion zusätzlich erschwert.

3.2.8 Zusammenfassung

In diesem Abschnitt wurde ein Verfahren zur Merkmalsselektion unter Beachtung der benötigten Rechenzeit vorgestellt. Die Selektion wurde als multikriterielles Optimierungsproblem formuliert und mittels des genetischen Algorithmus NSGA-II implementiert. Die Implementierung kann leicht um weitere Optimierungskriterien, wie der Messunsicherheit der Merkmale, sowie um weitere Gütefunktionen, z. B. basierend auf Transinformation, erweitert werden.

Der Ansatz wurde mit einem synthetischen und zwei realen Datensätzen aus dem Bereich der Schüttgutsortierung evaluiert. Bei der Selektion mit den realen Datensätzen wurden Selektionen gefunden, die die Motivation der Methode bestätigen: Die Minimierung des Umfangs der Selektion entspricht nicht notwendigerweise der Minimierung der benötigten Rechenzeit. Insbesondere bei kurzen Deadlines kann eine weniger relevante, aber schneller berechenbare Selektion einer relevanteren, aber teureren Selektion vorgezogen werden. Die hier gezeigte multikriterielle Optimierung zeigt hier bessere Ergebnisse als eine Selektion ohne Bewertung der Rechenzeit mit CFS, als auch bessere Ergebnisse einer Selektion mit GFS, bei der die Rechenzeit in einer skalaren Bewertungsfunktion mit eingeht. Bei der Selektion mit den synthetischen Daten wurden durch kein getestetes Verfahren geeignete Kombinationen von Merkmalen gefunden. Dies ist teilweise durch die Konstruktion des Datensatzes erklärbar.

Anders als andere Ansätze produziert der hier vorgestellte Ansatz eine Reihe von Pareto-optimalen, also gleichwertigen Selektionen. Es obliegt dem Benutzer, die für die jeweilige Situation passende Selektion zu wählen. Diese

Abwägung der verschiedenen Kriterien kann durch Deadlinesimulation und Messung der CPU-Last aber auch automatisiert werden. Bei kurzen Deadlines würden dann Selektionen verwendet, die bei langen Deadlines schlechtere Klassifikationsergebnisse produzieren, kurze Deadlines aber deutlich weniger häufig verletzen.
Dieses Vorgehen könnte weiter verbessert werden, indem die Abhängigkeiten zwischen Merkmalen, z. B. bei der Berechnung der Farbmomente, ebenfalls berücksichtigt würden. Diese Abhängigkeiten könnten durch einen Berechnungsgraphen modelliert und bei der Berechnung der mittleren Kosten und der Kostenstreuung einbezogen werden. Diese Berechnung wäre allerdings aufwendiger als die lineare Form in Gleichung (3.33). Alternativ könnten die Kosten von Gruppen von Merkmalen wie von Paclík et al. [Pac+02] beschrieben gemessen werden, um die Abhängigkeiten implizit zu modellieren. In diesem Fall müssten allerdings deutlich mehr Geschwindigkeitsmessungen der Merkmale durchgeführt werden.

3.3 Datenangepasste Konturmerkmale mit Shape Ferns

Wenn die Selektion von Standardmerkmalen keine guten Ergebnisse liefert, können Methoden des Feature Learning Abhilfe schaffen. In den folgenden zwei Abschnitten werden daher zwei Methoden vorgestellt, die visuelle Merkmale für die Charakterisierung der Form bzw. der Farbe und Textur aus Daten ableiten. Die Form ist besonders bei der Inspektion technisch hergestellter Objekte ein wichtiges Merkmal. Farbe und Textur der Objekte spielt hier oft eine untergeordnete Rolle. Anwendungsbeispiele für solche Inspektionsprobleme sind die Sortierung von Schrauben und Muttern oder die Klassifikation elektronischer Bauelemente beim Recycling. In diesem Abschnitt werden daher ausschließlich Formmerkmale behandelt. Die Klassifikation anhand von Farbe und Textur ist Thema des nächsten Abschnitts.

3.3.1 Stand der Technik

Die Objektform wird üblicherweise anhand einer Beschreibung der Objektkontur oder Beschreibung der Objektfläche bzw. des Objektvolumens charakterisiert. Für beide Ansätze gibt es eine Reihe von etablierten Extraktionsverfahren, die auch schon in Abschnitt 2.2.1 aufgegriffen wurden.
Zur Charakterisierung der Objektkontur eignen sich beispielsweise die Fourier-Konturdeskriptoren. Wie in Beyerer et al. [BRN17] beschrieben, wird die Kontur hierfür als komplexe periodische Funktion $c(t) = x(t) + i\,y(t)$ mit $i = \sqrt{-1}$ und $c(t) = c(t + T)$ (d. h. Periodenlänge T) aufgefasst. Dabei bezeichnen $x(t)$ und $y(t)$ die Bildkoordinaten des Konturpunkts mit Parameter t. Der Fourier-Konturdeskriptor entspricht den Koeffizienten der Fouriertransformation $C(f)$ von $c(t)$. Durch Normalisierung (siehe Beyerer et al. [BRN17], S. 53 ff.) kann zudem Invarianz gegenüber Translation, Rotation und Skalierung der Objekte erreicht werden. Die Entfernung von hochfrequenten Anteilen macht den Deskriptor zudem robust gegenüber kleinen Veränderungen der Kontur, während die Entfernung niederfrequenter Anteile diese feinen Unterschiede hervorhebt, aber die grobe Form vernachlässigt.
Weiterführende Ansätze beschreiben die Kontur mit Hilfe von Wavelet-Transformationen [TB97; YH98] oder über Statistiken der topologischen Struktur der Kontur [BMP01; AY03]. Diese und andere Beschreibungen der Kontur haben allerdings den Nachteil, dass sie keine Beschreibung des Objektvolumens liefern. Zusammenhängende Objekte mit Löchern, wie die Lego Technik Steine aus Tabelle 3.2, können allein mit Konturmerkmalen nicht von Objekten ohne Löcher, aber gleicher Kontur, wie die Lego Basic Steine, unterschieden werden.
Deskriptoren zur Beschreibung des Volumens stützen sich häufig auf Statistiken der Vordergrundpixel wie z. B. Hu- oder Zernike-Momente [Hu62; TC88]. Alternative Ansätze beschreiben die lokale Struktur der Fläche, beispielsweise durch Generalisierung der Fourier-Konturdeskriptoren auf die Objektfläche [ZL02] oder indem die Objektfläche durch eine hierarchische

Struktur repräsentiert wird [KK00]. Eine gute Übersicht und eine Kategorisierung dieser und anderer Verfahren findet sich in der Arbeit von Zhang und Lu [ZL04].

3.3.2 Beitrag zum Stand der Technik

Alle hier vorgestellten Verfahren beschreiben generische Verfahren zur Beschreibung der Form und sind damit nicht notwendigerweise optimal für ein gegebenes, spezielles Sortierproblem geeignet. Zudem beschreiben die meisten Verfahren entweder die Kontur oder die Fläche der Objekte. Bei vielen Sortierproblemen, beispielsweise bei der Sortierung von Lego-Steinen, ist aber eine Kombination dieser Merkmale nötig, da die diskriminative Information je nach Objekt eher in der Kontur oder eher in der Fläche kodiert ist.
In diesem Abschnitt wird ein Feature Learning Verfahren vorgestellt, mit dem sich ein komplexer Formdeskriptor an eine Lernstichprobe anpassen lässt. Der Deskriptor setzt sich aus einfachen, schnell berechenbaren Kontur- und Flächenmerkmalen zusammen. Gleichzeitig wird ein diskriminativer, ebenfalls schnell auswertbarer Klassifikator gelernt. Das Verfahren erzeugt hochdiskriminative Deskriptoren und Klassifikatoren. Außerdem lässt sich das Ergebnis in eine von Menschen einfach interpretierbare Beschreibung überführen und somit zur Unterstützung von Feature Engineering nutzen.

3.3.3 Shape Ferns

Die hier vorgestellte Methode orientiert sich am Keypoint-Matching Ansatz von Ozuysal et al. [OFL07], in dem einfache Grauwertdifferenzen zwischen zufällig gewählten Bildpositionen zur Klassifikation mit Random Ferns (s. u.) genutzt werden. Die Merkmale und Klassifikatoren lassen sich sehr schnell berechnen und der Ansatz kann durch Hinzufügen oder Entfernen von Klassifikatoren an die verfügbare Rechenleistung angepasst werden. In dieser Arbeit werden ebenfalls Random Ferns verwendet, die sich hier allerdings statt

auf Grauwertdifferenzen auf zwei Typen primitiver Formmerkmale stützen. Diese Merkmale beschreiben sowohl die Kontur als auch das Volumen der Objekte. Im Gegensatz zu Ozuysal et al. [OFL07] werden die Random Fern Klassifikatoren hier mittels Gradient Boosting (siehe Abschnitt 2.4.7) zu einer robusten, hochdiskriminativen Klassifikationskaskade zusammengesetzt. Ein ähnliches Verfahren wurde bereits in Richter et al. [RHE14] verwendet, wenn auch in einem anderen Kontext. Im Folgenden werden die drei Komponenten der Methode beschrieben: Random Ferns, Fern Boosting und die primitiven Formmerkmale.

Random Ferns. Random Ferns wurden von Ozuysal et al. [OFL07] als schnelles Verfahren zum Keypoint-Matching vorgestellt. Sie teilen dabei Eigenschaften mit Entscheidungsbäumen und können geometrisch ähnlich interpretiert werden (siehe Ozuysal et al. [Ozu+10]). Die Methode lässt sich aber auch als eine semi-naive Bayes-Methode mit binären Merkmalen motivieren. Eine semi-naive Bayes-Methode bezeichnet dabei eine Methode zur MAP-Klassifikation, d. h. zur Klassifikation anhand der Vorschrift

$$\hat{\omega} = \arg\max_{\omega} P(\omega \,|\, \boldsymbol{d}) = \arg\max_{\omega} P(\omega)\, P(\boldsymbol{d} \,|\, \omega)\,. \tag{3.38}$$

Die Merkmale $\boldsymbol{d}$ sind hier binär, d. h. $\boldsymbol{d} \in \{0,1\}^D$. Wie bei Entscheidungsbäumen können diese Merkmale mittels Testfunktionen $h(\boldsymbol{x})$ aus beliebigen anderen Merkmalen $\boldsymbol{x}$ abgeleitet werden. Mit binären Merkmalen kann die klassenbedingte Merkmalsdichte $P(\boldsymbol{d} \,|\, \omega)$ durch eine Zählstatistik über die Lernstichprobe festgelegt werden,

$$P(\boldsymbol{d} \,|\, \omega) \approx \frac{1}{N} \sum_{\boldsymbol{d}_n \in \mathcal{D}} \mathbb{1}[\boldsymbol{d}_n = \boldsymbol{d} \wedge \omega_n = \omega]\,. \tag{3.39}$$

Da sich mit D-dimensionalen Merkmalen aber 2^D mögliche Kombinationen bilden lassen, ist bereits für kleine D eine sehr umfangreiche Stichprobe nötig, um $P(\boldsymbol{d}\,|\,\omega)$ mit hinreichender Genauigkeit zu schätzen.
Der naive Bayes-Ansatz reduziert diese kombinatorische Vielfalt durch die Annahme, dass die Merkmale $[\boldsymbol{d}]_d$ paarweise klassenbedingt unabhängig voneinander sind. Dadurch lässt sich die klassenbedingte Merkmalsdichte faktorisieren,

$$P(\boldsymbol{d}\,|\,\omega) \approx \prod_{d=1}^{D} P([\boldsymbol{d}]_d\,|\,\omega)\,, \tag{3.40}$$

und es müssen nur noch die $P([\boldsymbol{d}]_d\,|\,\omega)$ festgelegt werden. Statt 2^D Wahrscheinlichkeiten müssen pro Klasse also nur noch $2D$ Wahrscheinlichkeiten aus der Lernstichprobe geschätzt werden.
Auch wenn nur wenige Merkmale paarweise stochastisch abhängig sind, trifft die naive Bayes-Annahme in der Praxis fast nie zu, wodurch die Klassifikationsleistung sinkt. Stattdessen finden sich häufig Cliquen von (klassenbedingt) abhängigen Merkmalen, wobei die Cliquen wiederum paarweise unabhängig voneinander sind. Diese Gruppierung ist nicht bekannt, kann aber durch einen randomisierten Ansatz angenähert werden. Hierfür werden die D Merkmale zufällig in T Gruppen von jeweils S Merkmalen eingeteilt. Mit zufälligen Permutationen $1 \leq \sigma_t(s) \leq D$ lassen sich diese Gruppierung durch Indexmengen

$$\mathcal{F}_t = \{\sigma_t(1), \ldots, \sigma_t(S)\} \subset \{1, \ldots, D\}, \quad t = 1, \ldots, T \tag{3.41}$$

ausdrücken und die klassenbedingte Merkmalsdichte kann durch

$$P(\boldsymbol{d}\,|\,\omega) \approx \prod_{t=1}^{T} P\big([\boldsymbol{d}]_{\mathcal{F}_t}\,\big|\,\omega\big) \tag{3.42}$$

approximiert werden. Durch die zufällig gewählten Permutationen kann ein Merkmal hier mehreren Gruppen angehören, aber nicht mehrfach in der gleichen Gruppe vertreten sein. Mit diesem Ansatz müssen $(T \cdot 2^S)$ Wahrscheinlichkeiten pro Klasse bestimmt werden. Mit $S \ll D$ wird die Anzahl der zu schätzenden Parameter somit deutlich reduziert.
Dieser Ansatz kann also als ein Mittelweg zwischen den Gleichungen (3.39) und (3.40) angesehen werden und wird daher als eine semi-naive Bayes-Methode bezeichnet [ZW05]. Der Parameter S steuert dabei den „Grad der Naivität". Die durch die $\mathcal{F}_t$ induzierten Gruppierungen der Merkmale werden wiederum als Random Ferns bezeichnet [OFL07].

Fern Boosting. Bei Ozuysal et al. [OFL07] entsprechen die binären Merkmale einer Schwellwertbildung über Grauwertdifferenzen, d. h. mit zwei Pixelpositionen $\boldsymbol{u}_d, \boldsymbol{v}_d \in \mathbb{R}^2$, dem Grauwertbild $g : \mathbb{R}^2 \to [0, 1]$ und einem Schwellwert τ_d ist $[\boldsymbol{d}]_d = \mathbb{1}[g(\boldsymbol{u}_d) - g(\boldsymbol{v}_d) < \tau_d]$. Die Grauwertdifferenzen $[\boldsymbol{x}]_d = (g(\boldsymbol{u}_d) - g(\boldsymbol{v}_d))$ lassen sich dabei wiederum als metrische Merkmale aus einem Merkmalsraum $\tilde{\mathbb{M}}$ auffassen, also $\boldsymbol{x} \in \tilde{\mathbb{M}} = [-1, 1]^D$. Diese Betrachtung ermöglicht eine geometrische Interpretation von semi-naive Bayes: Ein Random Fern $\mathcal{F}_t$ mit S Merkmalen partitioniert den D-dimensionalen Merkmalsraum $\tilde{\mathbb{M}}$ in 2^S disjunkte Regionen $\mathcal{R}_{t,s}$, $s = 1, \ldots, 2^S$. Analog zu Entscheidungsbäumen (siehe Abschnitt 2.4.5) wird jede dieser Regionen mit einer Schätzung $P_{t,s}([\boldsymbol{d}]_{\mathcal{F}} \,|\, \omega) = q_{t,s}(\omega)$ assoziiert, wobei $q_{t,s}(\omega)$ die Wahrscheinlichkeit ist, dass ein $\boldsymbol{x}$ der Klasse ω in der Region $\mathcal{R}_{t,s}$ liegt.
Wie bei Entscheidungsbäumen können statt Wahrscheinlichkeitsfunktionen auch beliebige andere Hypothesenfunktionen $h_{t,s}(\boldsymbol{x})$ mit den Regionen assoziiert werden. Die Klassifikation würde dann nicht mehr durch MAP mit Gleichung (3.42) zur Schätzung von $p(\boldsymbol{x} \,|\, \omega)$ durchgeführt, sondern wie bei Random Forests durch Mittelung der Vorhersagen des Ensembles aus T Random Ferns erreicht. Alternativ kann das Ensemble auch mit Hilfe von Friedmans Gradient Boosting Verfahren (siehe Abschnitt 2.4.7) erstellt werden.

Der Lernalgorithmus ist dabei analog zu Friedmans L_2_TreeBoost Algorithmus zur binären Klassifikation [Fri01]. Die zu minimierende Verlustfunktion ist die negative binomiale log-Likelihood des Modells

$$l\left(H(\boldsymbol{x}), y\right) = \log\left(1 + \exp\left(-2yH(\boldsymbol{x})\right)\right) \tag{3.43}$$

mit $y = \pm 1$, wobei $H(\boldsymbol{x})$ ähnlich wie bei der logistischen Regression (siehe Abschnitt 2.4.2) die (skalierte) Logit-Funktion

$$H(\boldsymbol{x}) := H_T(\boldsymbol{x}) = \sum_{t=1}^{T} h_t(\boldsymbol{x}) \approx \frac{1}{2} \log \frac{P(y = 1 \,|\, \boldsymbol{x})}{P(y = -1 \,|\, \boldsymbol{x})} \tag{3.44}$$

approximiert. Die Stufen $h_t(\boldsymbol{x})$ setzen sich dabei wie bei Entscheidungsbäumen (vgl. Gleichung 2.81) aus den mit den Regionen $\mathcal{R}_{t,s}$ assoziierten Hypothesenfunktionen zusammen, also

$$h_t(\boldsymbol{x}) = \sum_{s=1}^{2^S} \mathbb{1}[\boldsymbol{x} \in \mathcal{R}_{t,s}]\, h_{t,s}(\boldsymbol{x}). \tag{3.45}$$

Nach L_2_TreeBoost ergeben sich die $h_{s,t}(\boldsymbol{x})$ schließlich durch stufenweise Minimierung der Verlustfunktion, wobei jede Stufe eine Konstante $\gamma^*_{t,s}$ hinzufügt,

$$h_{t,s}(\boldsymbol{x}) = \gamma^*_{t,s} = \arg\min_{\gamma} \sum_{\boldsymbol{x}_n \in \mathcal{D}} \mathbb{1}[\boldsymbol{x}_n \in \mathcal{R}_{t,s}]\, l\left(H_{t-1}(\boldsymbol{x}_n) + \gamma, y_n\right) \tag{3.46}$$

$$\approx \frac{\sum_{\boldsymbol{x}_n \in \mathcal{D}} \mathbb{1}[\boldsymbol{x}_n \in \mathcal{R}_{t,s}]\, \tilde{y}_n}{\sum_{\boldsymbol{x}_n \in \mathcal{D}} \mathbb{1}[\boldsymbol{x}_n \in \mathcal{R}_{t,s}]\, |\tilde{y}_n|(2 - |\tilde{y}_n|)}. \tag{3.47}$$

Da diese Minimierung nicht geschlossen lösbar ist, wurde sie in der zweiten Zeile durch einen Schritt einer numerischen Minimierung mit dem Newton–

Raphson-Verfahren approximiert [Fri01]. Die Pseudoantworten $\tilde{y}_n$ sind dabei durch

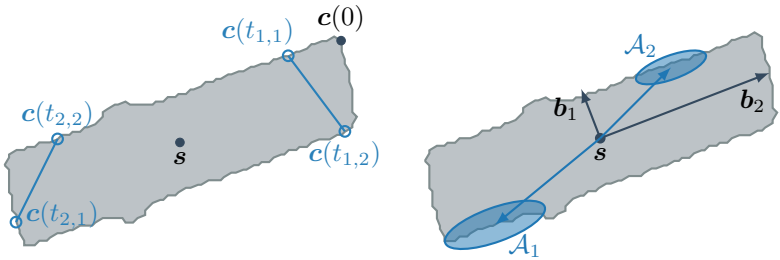
$$\tilde{y}_n := -\nabla_H l\left(H_{t-1}(\boldsymbol{x}_n), y_n\right) = \frac{2y_n}{1+\exp\left(2y_n H_{t-1}(\boldsymbol{x}_n)\right)} \tag{3.48}$$

definiert, also der benötigten Änderung der Hypothesenfunktion an $\boldsymbol{x}_n$ zur Minimierung der Verlustfunktion.
Die Klassifikation kann wie bei der logistischen Regression anhand der Hypothesenfunktion $H_T(\boldsymbol{x})$ oder durch Überführung von $H_T(\boldsymbol{x})$ in eine a-posteriori Wahrscheinlichkeit,

$$\hat{P}(y=1 \,|\, \boldsymbol{x}) = \frac{1}{1+\exp\left(-2H_T(\boldsymbol{x})\right)}, \tag{3.49}$$

vorgenommen werden. Neben binärer Klassifikation kann der hier gezeigte Ansatz auch analog zu Friedmans L_K_TreeBoost auf eine Mehrklassenklassifikation ausgeweitet werden [Fri01].

Primitive Formmerkmale für Shape Ferns. Nachdem das Klassifikationsverfahren und die Erstellung der Fern-Kaskade geklärt ist, bleibt noch offen, welche Merkmale zur Klassifikation verwendet werden sollen. Wie bei Ozuysal et al. [OFL07] könnten hier Pixeldifferenzen eines Maskenbildes $m(\boldsymbol{q}) \in \{0,1\}$ genutzt werden. Allerdings sind solche Merkmale wenig aussagekräftig, da sich der Wertebereich auf $-1, 0$ und 1 beschränkt, wobei die allermeisten möglichen Differenzen innerhalb und außerhalb des Objekts den Wert 0 erzeugen. Stattdessen kommen hier zwei Merkmale zum Einsatz, welche beide Formaspekte – Kontur und Fläche – beschreiben: lokale Konturdistanzen und lokale Masse.
Für die Konturmerkmale wird die Kontur in Anlehnung an die Fourier-Konturdeskriptoren als periodische Funktion $\boldsymbol{c}(t) \in \mathbb{R}^2$ mit $\boldsymbol{c}(t) = \boldsymbol{c}(t+1)$ aufgefasst. Das Merkmal $[\boldsymbol{d}]_d$ wird dann durch Schwellwertbildung über



(a) Konturdistanzen **(b)** Lokale Masse

Abbildung 3.10: Primitive Formmerkmale für Shape Ferns.

den Abstand von zwei Konturpunkten an $t_{d,1}$ und $t_{d,2}$ gebildet (siehe Abbildung 3.10a),

$$[\boldsymbol{d}]_d = \mathbb{1}[\|\boldsymbol{c}(t_{d,1}) - \boldsymbol{c}(t_{d,2})\| < \tau_d] \,. \tag{3.50}$$

Dieses Merkmal ist invariant bezüglich Translation der Objekte. Um zusätzlich eine Invarianz gegenüber Rotation zu erreichen, wird $\boldsymbol{c}(0)$ als der am weitesten vom Schwerpunkt $\boldsymbol{s}$ des Objekts entfernte Konturpunkt festgelegt. Eine Skalierungsinvarianz kann beispielsweise durch Normalisierung anhand des Abstands von $\boldsymbol{c}(0)$ und $\boldsymbol{s}$ erreicht werden.
Die Flächenmerkmale bewerten die lokale Masse in einer Region $\mathcal{A}_d$, also die Anzahl der Vordergrundpixel in $\mathcal{A}_d$ (siehe Abbildung 3.10b),

$$[\boldsymbol{d}]_d = \mathbb{1}\left[\left(\int_{\mathcal{A}_d} m(\boldsymbol{q})\,\mathrm{d}\boldsymbol{q}\right) < \tau_d\right], \tag{3.51}$$

wobei $m(\boldsymbol{q})$ das Maskenbild bezeichnet, also $m(\boldsymbol{q}) = 1$ genau dann gilt, wenn $\boldsymbol{q}$ ein Vordergrundpixel ist und $m(\boldsymbol{q}) = 0$ sonst.

Die Region $\mathcal{A}_d$ ergibt sich wiederum aus dem Inhalt eines Kreises in Bezug zu einer Metrik $d(\boldsymbol{q}, \boldsymbol{p})$ mit Radius r_d um den Punkt $\boldsymbol{p}_d$,

$$\mathcal{A}_d = \{\boldsymbol{q} \,|\, d(\boldsymbol{q}, \boldsymbol{p}_d) \leq r_d\} \,. \tag{3.52}$$

Damit diese Flächenmerkmale gegenüber Translation, Rotation und Skalierung invariant sind, werden die Regionen in Bezug auf ein lokales Koordinatensystem mit Ursprung im Objektschwerpunkt $\boldsymbol{s}$ definiert, bei dem die Koordinatenachsen den Hauptachsen $\boldsymbol{b}_1$ und $\boldsymbol{b}_2$ des Objekts entsprechen. Mit der Basiswechselmatrix $\boldsymbol{B} = \begin{pmatrix} \boldsymbol{b}_1 & \boldsymbol{b}_2 \end{pmatrix}^\top$ kann die Transformation der lokalen in globale Koordinaten implizit durch Verwendung der quadrierten Mahalanobis-Distanz

$$d(\boldsymbol{q},\boldsymbol{p}) = (\boldsymbol{q} - \boldsymbol{p})^\top \boldsymbol{B}^\top \boldsymbol{B} \, (\boldsymbol{q} - \boldsymbol{p}) \tag{3.53}$$

durchgeführt werden. Mit dieser Transformation beschränken sich sinnvolle Werte für den Kreismittelpunkt $\boldsymbol{p}_d$ auf den Bereich $[-1, 1] \times [-1, 1]$.

Shape Fern Training. Diese primitiven Formmerkmale sind zwar aussagekräftiger als Pixeldifferenzen des Maskenbilds $m(\boldsymbol{q})$, isoliert betrachtet aber dennoch nur bedingt zur Klassifikation geeignet. In Kombination mit Random Fern Boosting ergibt sich aber ein starker Klassifikator. Für den Lernalgorithmus müssen allerdings noch die Regionen $\mathcal{R}_{t,s}$ bzw. die generierenden Random Ferns $\mathcal{F}_t$ definiert werden. Prinzipiell können die Merkmale der $\mathcal{F}_t$, wie von Ozuysal et al. [OFL07] vorgeschlagen, zufällig generiert werden. Mit den hier verwendeten Merkmalen wäre jedoch die Wahrscheinlichkeit der Entdeckung diskriminativer Merkmale relativ gering.
Diese Wahrscheinlichkeit kann erhöht werden, indem eine Menge von K zufälligen Merkmalskandidaten generiert wird, aus denen die S relevantesten Merkmale ausgewählt werden. Solch eine Selektion lässt sich mit beliebigen Merkmalsselektionsverfahren durchführen. Im Rahmen dieser

Algorithmus 4 Shape Fern Training.

function LEARN($\mathcal{D}, T, S, K$)
  $H_0(\boldsymbol{x}) := \mathbb{E}_{\mathcal{D}}\{y\}$
  for t = 1, . . . , T **do**
    $\tilde{y}_n \leftarrow 2y_n(1 + \exp(2y_n H_{t-1}(\boldsymbol{x}_n)))^{-1}\ \forall \boldsymbol{x}_n \in \mathcal{D}$
    $\Psi_t \leftarrow \{\text{RANDOMFEATURE}() \mid k = 1, \ldots, K\}$
    $\mathcal{F}_t \leftarrow \emptyset$
    for s = 1, . . . , S **do**
      $\psi_s \leftarrow \arg\max_{\psi \in \Psi_t} \frac{s \cdot \overline{r_{\mathcal{F}_t \cup \psi, \tilde{y}}}}{\sqrt{s + s(s+1)\overline{r_{\mathcal{F}_t \cup \psi, \mathcal{F}_t \cup \psi}}}}$
      $\mathcal{F}_t \leftarrow \mathcal{F}_t \cup \{\psi_s\}$
    $\gamma_{t,s} \leftarrow \frac{\sum_{\boldsymbol{x}_n \in \mathcal{D} \cap \mathcal{R}_{s,t}} \tilde{y}_n}{\sum_{\boldsymbol{x}_n \in \mathcal{D} \cap \mathcal{R}_{s,t}} |\tilde{y}_n|(2 - |\tilde{y}_n|)}, \quad s = 1, \ldots, 2^S$
    $H_t(\boldsymbol{x}) := H_{t-1}(\boldsymbol{x}) + \sum_{s=1}^{2^S} \mathbb{1}[\boldsymbol{x} \in \mathcal{R}_{t,s}]\, \gamma_{t,s}$
  return $H_T(\boldsymbol{x})$

Arbeit wird jedoch ein modifizierten CFS-Verfahren genutzt, bei dem die Zwischenergebnisse des Fern Boosting Algorithmus verwendet werden. Statt den Merkmalen, die am stärksten mit dem Klassenlabel y korrelieren, werden hier diejenigen Merkmale selektiert, die die stärkste Korrelation mit den Pseudoantworten $\tilde{y}_n$ aufweisen. Dadurch wird sichergestellt, dass sich die Merkmale aufeinanderfolgender Shape Ferns unterscheiden bzw. optimal ergänzen. Der vollständige Lernalgorithmus ist in Algorithmus 4 gezeigt.

3.3.4 Experimente und Ergebnisse

Der Ansatz wurde mit Hilfe des im vorherigen Abschnitt vorgestellten Lego-Datensatzes (siehe S. 141) evaluiert. Abweichend von den Experimenten des letzten Abschnitts war hier die Trennung von Basic Steinen (Zeilen 1–4 in Tabelle 3.2) und Technik Steinen (Zeilen 5–8 in Tabelle 3.2) das Klassifikationsziel. Diese Gruppierung wurde gewählt, um die Klassifikation durch größere Variation innerhalb der Klassen zu erschweren und so eine aussagekräftigere Evaluation zu ermöglichen.

Die Shape Ferns wurden wie oben beschrieben implementiert, wobei die lokalen Konturmerkmale nicht normalisiert wurden, also nicht invariant bezüglich der Objektgröße waren. Auf diese Normalisierung wurde verzichtet, da die Größe der Steine absehbar ein wichtiges Klassifikationsmerkmal darstellt.
Dadurch, dass sich einige Lego Technik Steine nur durch Löcher von den Lego Basic Steinen unterscheiden, ist zudem zu erwarten, dass Konturmerkmale alleine nicht zur Klassifikation ausreichen. Um dies zu verdeutlichen, wurde der hier vorgestellte Ansatz mit den einleitend beschriebenen translations- und rotationsinvarianten Fourier-Konturdeskriptoren verglichen. Diese Deskriptoren wurden mit einer soft margin SVM mit RBF-Kernel klassifiziert. Die SVM- und Kernel-Hyperparameter wurden dabei durch Parametersuche in einer 10-fold Cross Validation über die Lernstichprobe festgelegt.

Ergebnisse. Abbildung 3.11 zeigt einen Vergleich der Klassifikationsleistung von Fourier-Konturdeskriptoren mit dem hier vorgestellten Ansatz. Die Klassifikationsleistung wurde dabei in einer 5-fold Cross Validation ermittelt. Wie erwartet wird mit Fourier-Konturdeskriptoren keine gute Klassifikation erreicht. Überraschenderweise führt hier auch eine geringere Anzahl an Fourier-Koeffizienten D zu einer besseren Klassifikationsgüte, obwohl dadurch die Kontur weniger genau repräsentiert wird.
Zur Evaluation der Shape Ferns wurden die Anzahl der Random Ferns T der Kaskade, die Anzahl S von Merkmalen pro Random Fern sowie die Anzahl K von in jeder Stufe zufällig generierten Merkmalskandidaten variiert. Die Experimente deuten darauf hin, dass eine größere Kaskade das Klassifikationsergebnis verbessert. Dies ist nachvollziehbar, da hierdurch der Approximationsfehler $l\left(H_T(\boldsymbol{x}), y\right)$ minimiert wird. Die Anzahl S der Merkmale jedes Random Ferns hat einen deutlich höheren Einfluss auf die Klassifikationsleistung. Diese darf nicht zu groß gewählt werden, da sonst die Anzahl der Regionen $\mathcal{R}_{t,s}$ zu groß wird und die assoziierten $\gamma^*_{t,s}$ nicht genau genug geschätzt werden können. Die Wahl von S hängt also von der

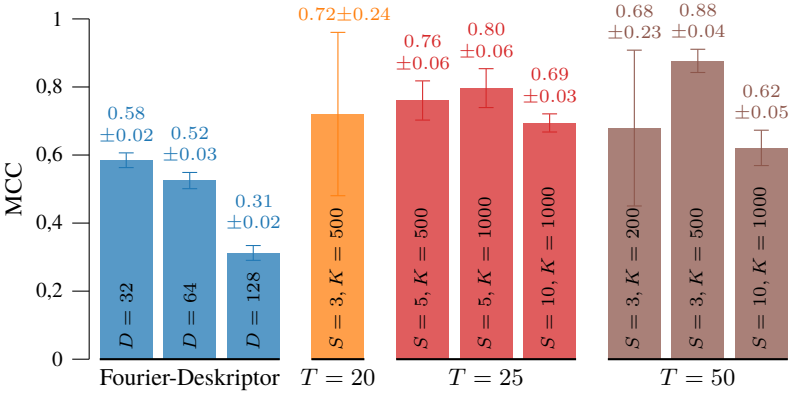


Abbildung 3.11: Vergleich der Klassifikationsleistung von Shape Ferns mit verschiedenen Parametern und Fourier-Konturdeskriptoren. Die Fehlerbalken zeigen die Standardabweichung über eine 5-fold Cross Validation.

Lernstichprobe ab. Bei dem hier verwendeten Datensatz führt $S = 5$ zu guten Ergebnissen.

Ebenfalls wichtig ist die Anzahl K von zufälligen Merkmalskandidaten. Hier führen größere Werte zu besseren Ergebnissen, da damit die Wahrscheinlichkeit erhöht wird, dass sich unter den K Kandidaten genügend stark mit $\tilde{y}_n$ korrelierende Merkmale finden.

Die besten Ergebnisse wurden mit einer Kaskade von $T = 50$ Random Ferns mit je $S = 3$ ausgewählten Merkmalen aus $K = 500$ Kandidaten erreicht. Abbildung 3.12 zeigt die Zusammensetzung der von dieser Kaskade gelernten Merkmale. Die Konturdistanzmerkmale stellen 63 % der Merkmale, die restlichen 37 % sind Flächenmerkmale. Die Konturdistanzen bewerten dabei vor allem die Breite der Objekte. Dies ist durch den Datensatz begründbar: Die Mehrzahl der Basic Steine ist zweireihig und damit breiter als die einreihigen Technik Steine, und insbesondere breiter als die Kreuzprofilachsen und die Verbindersteine. Die Flächenmerkmale bewerten vor allem die lokale Masse

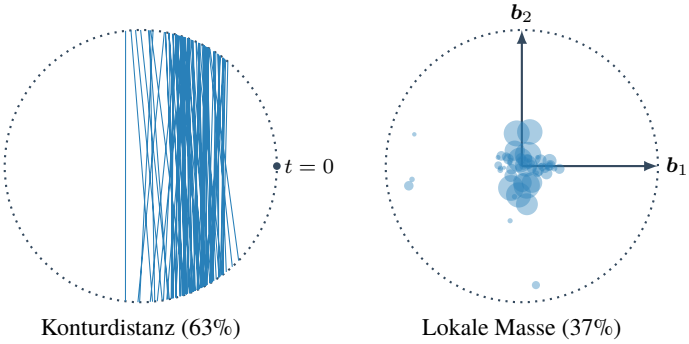


Abbildung 3.12: Inspektion der selektierten Formmerkmale der Fern-Kaskade mit $T = 50, S = 3$ und $K = 500$.

im Zentrum der Steine. Diese Merkmale heben den Unterschied zwischen einreihigen Technik Steinen mit Löchern und Basic Steinen ohne Löcher hervor.

Aufgrund dieser Analyse können die gelernten Merkmale prinzipiell auch auf wenige Merkmale zur Bewertung der Breite und der lokalen Masse im Zentrum der Steine reduziert werden. Das Verfahren eignet sich also nicht nur zum Lernen von Formmerkmalen, sondern kann auch als Unterstützung im Feature Engineering verwendet werden.

3.3.5 Zusammenfassung

In diesem Abschnitt wurden Shape Ferns zur Beschreibung der Objektform vorgestellt. Mit diesem Verfahren lassen sich an die jeweilige Problemstellung angepasste Formmerkmale und Klassifikatoren gleichzeitig aus einer Lernstichprobe ableiten. Dazu werden primitive Merkmale zur Bewertung der Kontur und zur Bewertung der Fläche mittels Gradient Fern Boosting zu leistungsstarken Klassifikatoren zusammengesetzt. Der Ansatz wurde mit dem Lego-Datensatz aus Abschnitt 3.2 evaluiert und mit Fourier-Konturdeskriptoren verglichen. Shape Ferns zeigten dabei mit allen

evaluierten Parameterkombinationen bessere Ergebnisse als die Klassifikation mit Fourier-Konturdeskriptoren. Dieses Ergebnis war zu erwarten, da reine Konturmerkmale bei diesem Datensatz nicht ausreichend diskriminativ sind. Die Shape Ferns zeigten sich sensitiv gegenüber der Wahl der Parameter. Insbesondere die Anzahl der Merkmale pro Random Fern und die Anzahl der in jeder Stufe zufällig generierten Merkmalskandidaten beeinflussten das Klassifikationsergebnis stark.
Ein Vorteil des hier vorgestellten Ansatzes ist, dass die gelernten Merkmale der Kaskade nach dem Training inspiziert werden können. Im Experiment mit dem Lego-Datensatz zeigte sich, dass hier vor allem die Breite und der Füllgrad im Zentrum der Objekte relevante Merkmale sind. Statt für das Feature Learning können Shape Ferns also auch zur Unterstützung im Feature Engineering eingesetzt werden.
In diesem Kontext ist auch die Erweiterbarkeit des hier gezeigten Ansatzes um ergänzende Typen primitiver Merkmale interessant. Neben den beschriebenen Kontur- und Formmerkmalen können weitere lokale Merkmale, wie beispielsweise die Krümmung der Kontur oder die Distanz zum nächsten Konturpunkt, verwendet werden. Ebenso ist es denkbar, die Formmerkmale mit Textur- und Farbmerkmalen zu verbinden. Erste Experimente, bei denen ausschließlich die von Ozuysal et al. [OFL07] vorgeschlagenen Texturmerkmale verwendet wurden, zeigten allerdings nur geringen Erfolg. Zur Beschreibung von Farbe und Textur ist das im nächsten Abschnitt vorgestellte Verfahren besser geeignet.

3.4 Farb- und Texturmerkmale mit Bag of Visual Words

Die Objektfarbe ist für die Inspektion natürlicher Materialien wie Pflanzen und Lebensmittel ein hochdiskriminatives Merkmal [DS06; Zha+14a]. Farbe wird z. B. durch innere biochemische Abläufe und externe Beanspruchung beeinflusst und lässt somit Rückschlüsse auf Reifegrad, Schädlingsbefall,

Transportbedingungen etc. zu. Einfache Inspektionssysteme nutzen Standardmerkmale wie die mittlere Objektfarbe oder das davon abgeleitete mittlere Verhältnis zweier Farbkanäle [Bla+09; MKG15]. Komplexere Deskriptoren beschreiben die Verteilung der im Objekt vorkommenden Farben, beispielsweise mittels Farbhistogrammen [ElB+15]. Das in Abschnitt 1.4 beschriebene Verfahren verwendet ebenfalls Histogramme, auf deren Grundlage semantisch bedeutsame Merkmale gewonnen werden.
Diese Idee, also das Ableiten semantisch bedeutsamer Merkmale aus einer empirischen Verteilung der Objektfarben, wird in diesem Abschnitt wieder aufgegriffen, im Kontext von Bag of Visual Words (BOW) neu interpretiert und um weitere Kanäle zur Beschreibung von Textur und Form sowie um eine Rückweisungsoption erweitert.

3.4.1 Stand der Technik

Wenn auch nicht in Verbindung mit BOW, gibt es dennoch eine Reihe von Verfahren, die einfache Merkmale, meist Farbe, zu semantisch bedeutsamen Merkmalen fusionieren. Duffy et al. [DCL00] beschreiben beispielsweise ein Verfahren zur Detektion von Brandmarken auf Luftfiltern. Aus Bildaufnahmen beschädigter und intakter Beispiele werden Farbhistogramme gesammelt und zu einem für Defekte charakteristischen Histogramm fusioniert. Mit Hilfe dieses Histogramms wird eine Funktion zur Schätzung der Defektwahrscheinlichkeit eines Pixels definiert. Anhand dieser Funktion werden zu klassifizierende Bilder zunächst in eine Wahrscheinlichkeitskarte transformiert, in der jeder Pixel die Defektwahrscheinlichkeit kodiert. Diese Wahrscheinlichkeitskarte wird dann durch einen vom Benutzer einstellbaren Schwellwert binarisiert und das resultierende Binärbild schließlich zur Lokalisierung von Defekten verwendet. In einer späteren Arbeit verwenden Bergasa et al. [Ber+00] ein Gauß-Mischmodell anstelle von Farbhistogrammen. Der Rest des Verfahrens bleibt unverändert.

Zhang et al. [Zha+14b] verwenden einen ähnlichen Ansatz, um den Reifegrad und die Qualität von Dattelfrüchten zu bestimmen. Dafür wird ein Datensatz von 40 Beispielen in vier Güteklassen 1–4 unterteilt und für jede Klasse ein Histogramm der Rot- und Grünwerte gebildet. Mit Hilfe von MAP-Klassifikation wird eine Rückprojektionstabelle gebildet, die jedem Farbwert eine Güteklasse zuweist. Die Güteklassen werden dabei durch Grauwerte kodiert, wobei Güteklasse 4 durch den Grauwert 1 und Güteklasse 1 durch den Grauwert 255 repräsentiert wird. Fehlende Einträge in der Rückprojektionstabelle, also Farbwerte, die in der Lernstichprobe nicht auftreten, werden durch lineare Interpolation zwischen den nächsten Nachbarn in der Tabelle synthetisiert. Zur Klassifizierung einer ungesehenen Frucht wird zunächst (wie in Abschnitt 1.4) durch Rückprojektion ein Attributbild gewonnen und dann anhand einer Statistik über die Vordergrundpixel eine Güteklasse zugewiesen.

Ein von Li et al. [LCG09] vorgestelltes Verfahren zur Bestimmung des Reifegrads von Tomatenfrüchten verwendet ebenfalls Rückprojektion. Li et al. verwenden, anders als Duffy et al. und Zhang et al., allerdings eine halb-manuelle Unterteilung des HSV-Farbraums. Hierzu wird der HSV-Farbraum in 49 Bänder unterteilt, die dem menschlichen Farbempfinden von Tomatenfrüchten entsprechen sollen. Dabei wird die Helligkeitskomponente des Farbraums verworfen, d. h. das Verfahren betrachtet nur den Farbton und die Sättigung der Früchte. Durch eine komplizierte, auf den Anwendungsfall zugeschnittene Clusteranalyse werden schließlich vier dominante Farben des Datensatzes bestimmt. Der zur Klassifikation verwendete Deskriptor ist schließlich eine Zählstatistik bzw. ein Histogramm über die im Bild auftretenden dominanten Farben.

Alle diese Ansätze modellieren die charakteristischen Farben der zu untersuchenden Objekte, um semantische Merkmale abzuleiten. Dieses Vorgehen ist aber nicht zwingend notwendig. Mit dem Ziel der Tablettenblisterinspektion stellen Derganc et al. [Der+03] eine nichtparametrische Methode zur Bestimmung von Entscheidungsgebieten im Farbraum vor. Dazu werden zunächst die

Moden der Farbverteilung von Beispielbildern gesucht. Anschließend werden die Entscheidungsgebiete für intakte und für defekte Farben mittels Region-Growing festgelegt, wobei die Moden der Farbverteilung als Ausgangspunkte für das Region-Growing dienen. Ein Bild wird, wie bei den anderen Verfahren auch, durch Rückprojektion der Farbwerte und anschließender Berechnung einer Statistik der resultierenden Pixel klassifiziert.

3.4.2 Beitrag zum Stand der Technik

Zwar erzielen alle hier aufgeführten Ansätze gute Ergebnisse in den jeweiligen Anwendungsfeldern, es ist jedoch fraglich, ob die Verfahren auch für andere Problemdomänen geeignet sind: Die Ansätze von Duffy et al. [DCL00; Ber+00] lassen die Behandlung mehrerer Defektklassen offen. Die Konstruktion der Rückprojektionstabelle von Zhang et al. [Zha+14b] ist auf ihren Anwendungsfall zugeschnitten und nicht ohne Anpassung auf andere Sichtprüfaufgaben übertragbar. Li et al. [LCG09] betreiben einen hohen Aufwand zur Ableitung problemangepasster Merkmale, doch insbesondere das ungewöhnliche Verfahren zur Clusteranalyse scheint schlecht auf andere Domänen übertragbar. Der Ansatz von Derganc et al. [Der+03] macht schließlich die implizite Annahme, dass die zu untersuchenden Objekte annähernd uniform gefärbt sind bzw. dass die korrespondierenden Regionen im Farbraum zusammenhängen. Diese Annahme wird mit mehrfarbigen oder texturierten Objekten verletzt.

Überhaupt verwendet keiner der Ansätze Texturmerkmale, die bei der Klassifikation inhomogener Oberflächen – z. B. Bohnen, Holz, Steine etc. – durchaus diskriminative Informationen tragen. Auch gehen alle Ansätze von der Klassifikation in einer geschlossenen Welt aus. Nicht in der Lernstichprobe repräsentierte Klassen können somit nicht erkannt und zurückgewiesen werden.

In diesem Abschnitt wird die Farbklassifizierung abstrahiert und im Kontext von Bag of Visual Words neu formuliert. Abweichend vom üblichen BOW-

Ansatz werden für die visuelle Inspektion dichte Abtastung und primitive Deskriptoren verwendet. Neben Farbmerkmalen erlaubt das so formulierte Verfahren auch die Beschreibung der Textur und – in Grenzen – der Form der zu inspizierenden Objekte und ist somit für eine Vielzahl an Sortieraufgaben einsetzbar. Da bei der visuellen Inspektion und insbesondere bei der Schüttgutsortierung eine offene Welt, d. h. ein unvollständiges Wissen über die Klassen, angenommen werden muss, wird das BOW-Verfahren zudem um eine Rückweisungsoption für unbekannte Objekte erweitert. Die hier vorgestellten Methoden wurden in Richter et al. [RLB15c], Richter et al. [Ric+16b] und Richter et al. [RLB16] veröffentlicht.

3.4.3 Bag of Visual Words

Der BOW-Ansatz wurde bereits in Abschnitt 2.2.5 kurz vorgestellt, für ein besseres Verständnis der in diesem Abschnitt beschriebenen Modifikationen wird das Verfahren hier jedoch zunächst ausführlicher beschrieben.
Die Ursprünge von BOW finden sich in der Textverarbeitung. Das so genannte Unigramm-Modell ist ein generatives Modell $P(\boldsymbol{p})$ eines Textes $\boldsymbol{p} = (w_1, \ldots, w_N)$, in dem die Wörter w_n als stochastisch unabhängig angesehen werden, d. h. $P(\boldsymbol{p}) = \prod_{n=1}^{N} P(w_n)$. Dieses Modell kann zur MAP-Klassifikation verwendet werden, indem eine Schätzung $P(w \,|\, \omega)$ der klassenbedingten Auftrittswahrscheinlichkeit der Wörter, und damit auch $P(\boldsymbol{p} \,|\, \omega)$, erstellt wird. Alternativ kann aus einer Schätzung der dokumentabhängigen Auftrittswahrscheinlichkeit $P(w \,|\, \boldsymbol{p})$ ein Deskriptor zur Klassifikation mit beliebigen Klassifikatoren verwendet werden. In beiden Fällen spielt die Reihenfolge und der Kontext der Wörter ebenso wie die Länge der Texte keine Rolle – daher der Name „Bag of Words".
Dieser Ansatz kann auf Bilder übertragen werden, indem ein Bild als Dokument aufgefasst wird, das sich aus visuellen Wörtern eines nicht weiter definierten Vokabulars zusammensetzt. Wenn dieses visuelle Vokabular bekannt ist, kann ein Bild durch die im Bild vorkommenden Worte charakte-

risiert werden. Wie bei der Textverarbeitung wird auch hier die Reihenfolge bzw. der Ort und der (räumliche) Zusammenhang der visuellen Wörter nicht in den Deskriptor einbezogen. Die Größe der Bilder ist ebenfalls unerheblich. Insbesondere können Bilder unterschiedlicher Größe ohne Anpassung der Größe durch Skalierung oder Padding verglichen werden. Diese Eigenschaften sind für den Einsatz in der Schüttgutsortierung ideal, da die Rotation und Größe der zu sortierenden Objekte hier stark variiert und für eine vorherige Bildnormalisierung nur wenig Rechenzeit verfügbar ist.
Im Allgemeinen kann die BOW-Methode in zwei Phasen unterteilt werden: In der Lernphase wird das visuelle Vokabular anhand einer Lernstichprobe $\mathcal{D}$ festgelegt. In der Betriebsphase wird der Bilddeskriptor unter Zuhilfenahme des Vokabulars extrahiert. Im Folgenden werden beide Schritte formalisiert.

Visuelles Vokabular. Gegeben ist eine Stichprobe $\mathcal{D} = \{\boldsymbol{p}_1, \ldots, \boldsymbol{p}_N\}$ von N Beispielbildern $\boldsymbol{p}_n$. Aus diesen Bildern werden D-dimensionale lokale Deskriptoren $\psi(\boldsymbol{p}_n, \boldsymbol{u}_{tn})$ an den Bildkoordinaten $\boldsymbol{u}_{tn}$ extrahiert,

$$\boldsymbol{x}_{tn} = \psi(\boldsymbol{p}_n, \boldsymbol{u}_{tn}) \in \mathbb{M} = \mathbb{R}^D, \quad t = 1, \ldots, T_n. \tag{3.54}$$

Die Anzahl der von $\boldsymbol{p}_n$ extrahierten lokalen Deskriptoren T_n ist dabei von $\boldsymbol{p}_n$ abhängig und unterscheidet sich in der Regel für verschiedene $\boldsymbol{p}_n$. Der lokale Deskriptor $\psi(\boldsymbol{p}_n, \boldsymbol{u}_{tn})$ beschreibt das Bild $\boldsymbol{p}_n$ am, bzw. in einer Region um den Pixel $\boldsymbol{u}_{tn}$. Die Wahl von ψ ist beliebig, jedoch sollten die Invarianzeigenschaften, z. B. gegenüber Rotationen und Beleuchtungsunterschieden, bedacht werden, da sich diese auf den globalen Bilddeskriptor ausweiten. Im Kontext der Bildkategorisierung wird beispielsweise oft SIFT verwendet (z. B. in Csurka et al. [Csu+04], Perronnin et al. [PSM10], Chatfield et al. [Cha+11]), wodurch sich die Invarianzeigenschaften gegenüber Rotation und Skalierung und die Robustheit gegenüber Beleuchtungsunterschieden auf die Bilddeskriptoren übertragen.

Auf Grundlage der extrahierten lokalen Deskriptoren $\boldsymbol{x}_{tn}$ wird der Merkmalsraum $\mathbb{M}$ dann in K paarweise disjunkte, nicht notwendigerweise zusammenhängende Regionen $z_1, \ldots, z_K$ mit $z_i \cap z_k = \emptyset$ für $i \neq k$ unterteilt (siehe Abbildung 3.13a). Jede dieser Regionen entspricht einem visuellen Wort im Vokabular $\mathcal{V}$, d. h.

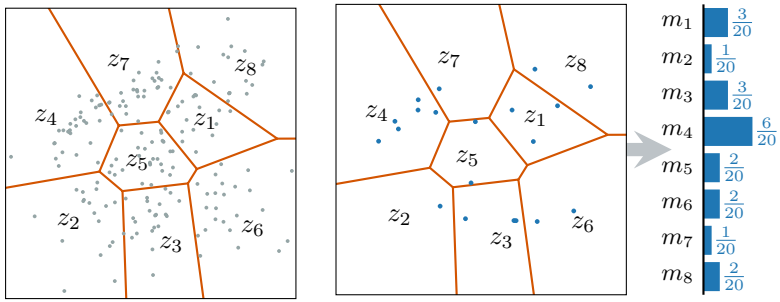
$$\mathcal{V} = \{z_1, \ldots, z_K\}. \tag{3.55}$$

Csurka et al. [Csu+04] und viele andere verwenden K-Means, um $\mathbb{M}$ zu partitionieren. Die Regionen sind dann durch die von den K Clusterzentren $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K$ definierten Voronoi-Gebiete festgelegt,

$$z_k = \left\{ \boldsymbol{x} \in \mathbb{M} \,\middle|\, k = \operatorname*{arg\,min}_{j=1,\ldots,K} \|\boldsymbol{\mu}_j - \boldsymbol{x}\| \right\}. \tag{3.56}$$

Andere Verfahren verwenden ein Gauß-Mischmodell (GMM) und MAP-Klassifikation, um die Regionen festzulegen (z. B. Chatfield et al. [Cha+11]). In diesem Kontext können die z_k auch als Entscheidungsregionen einer Vorklassifikation der lokalen Deskriptoren $\boldsymbol{x}_t \in \mathbb{M}$ angesehen werden.
Mit dieser Formulierung geht keine Klasseninformation in das Vokabular ein. Wenn diese vorhanden ist, kann sie aber auch genutzt werden, um klassenspezifische Vokabulare zu ermitteln. Im einfachsten Fall wird für jede Klasse ω_c ein eigenes, spezialisiertes Vokabular $\mathcal{V}_{\omega_c}$ ermittelt, indem für die Erstellung nur die aus den zu ω_c gehörenden $\boldsymbol{p}_n$ extrahierten $\boldsymbol{x}_{tn}$ betrachtet werden. Die Wörter des vollständigen Vokabulars $\mathcal{V} = \bigcup_{\omega_i} \mathcal{V}_{\omega_i}$ sind dann aber nicht mehr notwendigerweise paarweise disjunkt. Es gibt aber auch Methoden, die die Klasseninformation effizienter nutzen, siehe z. B. Winn et al. [WCM05] oder López-Sastre et al. [Lóp+11].

Globaler Deskriptor. Nachdem das visuelle Vokabular bestimmt ist, kann ein globaler Deskriptor $\boldsymbol{m}$ für ein ungesehenes Bild $\boldsymbol{p}$ bestimmt werden. Dazu werden zunächst wieder T lokale Deskriptoren $\boldsymbol{x}_t = \psi(\boldsymbol{p}, \boldsymbol{u}_t)$ extrahiert.



(a) Lernphase: Bestimmung des visuellen Vokabulars

(b) Betriebsphase: Extraktion eines globalen Deskriptors

Abbildung 3.13: Die beiden Phasen des BOW-Ansatzes mit zweidimensionalen lokalen Deskriptoren. Die orangefarbenen Linien zeigen die Grenzen der visuellen Wörter $z_1, \ldots, z_8$. Die grauen Punkte (links) sind die lokalen Deskriptoren der Lernstichprobe, die blauen Punkte (rechts) sind die lokalen Deskriptoren eines zu klassifizierenden Objekts. Die blauen Balken (rechts) zeigen eine visuelle Darstellung des entsprechenden globalen Objektdeskriptors $\boldsymbol{m}$.

Der globale Deskriptor ist dann eine Statistik der $\boldsymbol{x}_t$ in Bezug auf das visuelle Vokabular $\mathcal{V}$. Csurka et al. [Csu+04] verwenden dazu eine einfache Zählstatistik bzw. ein Histogramm: für jedes Wort z_k wird gezählt, wie viele der lokalen Deskriptoren $\boldsymbol{x}_t$ dem Wort angehören (siehe Abbildung 3.13b). Formal ist der Objektdeskriptor $\boldsymbol{m} = \left([\boldsymbol{m}]_1, \ldots, [\boldsymbol{m}]_K\right)^\top \in \mathbb{R}^K$ mit

$$[\boldsymbol{m}]_k = \frac{T_k}{T} = \frac{1}{T} \sum_{t=1}^{T} \mathbb{1}[\boldsymbol{x}_t \in z_k] \tag{3.57}$$

definiert, wobei T_k die Anzahl der $\boldsymbol{x}_t \in z_k$ ist. Wenn das Vokabular wie von Csurka et al. mittels K-Means bestimmt wurde, kann die Zugehörigkeit

zu z_k anhand von Gleichung (3.56) bestimmt werden. Der Objektdeskriptor berechnet sich dann durch

$$T_k = \sum_{t=1}^{T} \mathbb{1}\left[k = \arg\min_j \|\boldsymbol{x}_t - \boldsymbol{\mu}_j\|\right]. \tag{3.58}$$

Im Gegensatz zu dieser harten Zuweisung, in der jeder lokale Deskriptor $\boldsymbol{x}_t$ genau einem z_k zugewiesen wird, kann auch eine weiche Zuweisung

$$[\boldsymbol{m}]_k \propto \sum_{t=1}^{T} k(\boldsymbol{x}_t - \boldsymbol{\mu}_{z_k}) \tag{3.59}$$

mit einem Ähnlichkeitsmaß $k : \mathbb{M} \to \mathbb{R}_+$ definiert werden. Ein solcher Ansatz wird beispielsweise von Gemert et al. [Gem+08] und Philbin et al. [Phi+08] verfolgt. In Anlehnung an dieses Verfahren wird hier der mittlere Abstand zu jedem $\boldsymbol{\mu}_{z_k}$, also

$$[\boldsymbol{m}]_k = \frac{1}{T} \sum_{t=1}^{T} \|\boldsymbol{x}_t - \boldsymbol{\mu}_{z_k}\|, \tag{3.60}$$

als zusätzliche Statistik herangezogen. Dieses Vorgehen kann als „inverse" weiche Zuweisung interpretiert werden.

Fisher-Vektorkodierung. Die Fisher-Vektorkodierung (FV-Kodierung) ist eine von Perronnin und Dance [PD07] vorgestellte alternative Kodierung des globalen Deskriptors und kann als eine Statistik höherer Ordnung aufgefasst werden. Die Grundannahme der FV-Kodierung ist, dass die $\boldsymbol{x}_t$ durch ein GMM

$$p(\boldsymbol{x} \mid \boldsymbol{\theta}) = \sum_{k=1}^{K} P(z_k)\ p(\boldsymbol{x} \mid z_k, \boldsymbol{\theta}) \approx \sum_{k=1}^{K} P(z_k)\ \mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{3.61}$$

mit Parametern $\boldsymbol{\theta} = (\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1, P(z_1), \ldots, \boldsymbol{\mu}_K, \boldsymbol{\Sigma}_k, P(z_K))^\top$ erzeugt werden. Die $\boldsymbol{\mu}_k$ und $\boldsymbol{\Sigma}_k$ sind dabei der Erwartungswert bzw. die Kovarianzmatrix der k-ten Gauß-Dichte $p(\boldsymbol{x} \mid z_k, \boldsymbol{\theta})$.
In Anlehnung an die Fisher-Kernel-SVM [JH99] wird die FV-Kodierung aus dem Fisher-score, also dem Gradienten der log-Likelihood des Modells aus Gleichung (3.61) in Bezug auf $\boldsymbol{\theta}$, abgeleitet:

$$\nabla_{\boldsymbol{\theta}} \left(\frac{1}{T} \sum_t \log p(\boldsymbol{x}_t \mid \boldsymbol{\theta}) \right) = \frac{1}{T} \sum_t \nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{x}_t \mid \boldsymbol{\theta}). \tag{3.62}$$

Da der Fisher-score ein Maß für die Abweichung der Verteilung der $\boldsymbol{x}_t$ von der angenommenen Verteilung $p(\boldsymbol{x} \mid \boldsymbol{\theta})$ ist, kodiert auch der globale Deskriptor $\boldsymbol{m}$ diese Abweichung. Mit GMMs kann der globale Deskriptor geschlossen berechnet werden [PSM10]:

$$\boldsymbol{m} = \left(\boldsymbol{u}_1^\top, \boldsymbol{v}_1^\top, \ldots, \boldsymbol{u}_K^\top, \boldsymbol{v}_K^\top \right)^\top \in \mathbb{R}^{2KD} \quad \text{mit} \tag{3.63}$$

$$\gamma_{kt} = P(z_k \mid \boldsymbol{x}_t) = \frac{P(z_k)\ p(\boldsymbol{x}_t \mid z_k)}{p(\boldsymbol{x}_t \mid \boldsymbol{\theta})}, \tag{3.64}$$

$$\boldsymbol{u}_k = \frac{1}{N\sqrt{P(z_k)}} \sum_{t=1}^{T} \gamma_{kt}\, \boldsymbol{\Sigma}_k^{-\frac{1}{2}} \left(\boldsymbol{x}_t - \boldsymbol{\mu}_k \right) \quad \text{und} \tag{3.65}$$

$$\boldsymbol{v}_k = \frac{1}{N\sqrt{2P(z_k)}} \sum_{t=1}^{T} \gamma_{kt} \left(\operatorname{diag} \left(\boldsymbol{x}_t - \boldsymbol{\mu}_k \right) \boldsymbol{\Sigma}_k^{-1} \left(\boldsymbol{x}_t - \boldsymbol{\mu}_k \right) - \mathbf{1} \right). \tag{3.66}$$

Hier ist γ_{kt} die Wahrscheinlichkeit, dass $\boldsymbol{x}_t$ von der Komponente z_k generiert wurde, $\boldsymbol{u}_k$ ist die mittlere Abweichung der $\boldsymbol{x}_t$ zum Erwartungswert $\boldsymbol{\mu}_k$ und $\boldsymbol{v}_k$ ist die Varianz der Abweichung von $\boldsymbol{\mu}_k$.
Perronnin et al. konnten zeigen, dass die Klassifikation dieser Deskriptoren mit einer linearen SVM einer Klassifikation der Menge $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T\}$ mit Hilfe einer Fisher-Kernel-SVM entspricht [PSM10]. In der Praxis profitiert die Klassifikation mit einer linearen SVM oft von vorheriger Normalisierung

der Merkmale. Perronnin et al. schlagen dafür eine Kombination aus L_2-Normalisierung und komponentenweiser Potenz-Normalisierung vor:

$$[\boldsymbol{m}]'_j = \operatorname{sign}([\boldsymbol{m}]_j)|\,[\boldsymbol{m}]_j\,|^{\alpha}, \quad \alpha \in (0,1] \quad \text{gefolgt von} \tag{3.67}$$

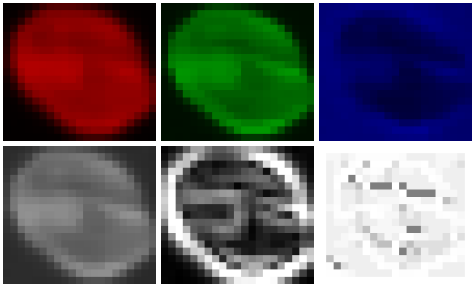
$$\boldsymbol{m}'' = \frac{\boldsymbol{m}'}{\|\boldsymbol{m}'\|}, \quad . \tag{3.68}$$

Die zweite Normalisierung entfernt vom Bild unabhängige Information aus dem Deskriptor, während die erste Normalisierung kleine Werte $[\boldsymbol{m}]_j$ anhebt und den Deskriptor somit entdünnt, was wiederum die Konvergenz des SVM-Trainings beschleunigt [PSM10].

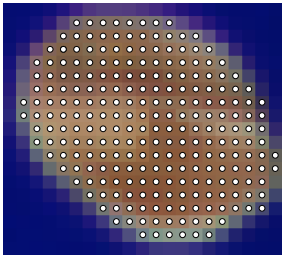
3.4.4 Anpassung für die visuelle Inspektion von Schüttgütern

Im Kontext der visuellen Inspektion kann das BOW-Modell direkt angewendet werden, indem die vereinzelten Objekte aus Abbildung 1.3 die Rolle der Muster $\boldsymbol{p}$ übernehmen. Wie am Anfang des Kapitels erwähnt, unterscheiden sich hier jedoch die Rahmenbedingungen von anderen Aufgaben des maschinellen Sehens wie der Bildkategorisierung oder der inhaltsbasierten Bildsuche. Das betrifft insbesondere die verfügbare Rechenzeit sowie die Größe und die Komplexität der zu klassifizierenden Bildern. Um diesen Unterschieden Rechnung zu tragen, wird vom traditionellen BOW-Ansatz abgewichen, indem primitive Deskriptoren und dichte Abtastung verwendet werden.

Primitive Deskriptoren. Als lokale Deskriptoren dienen typischerweise SIFT, HOG oder ähnliche Texturdeskriptoren. Im Kontext der Schüttgutsortierung sind diese Deskriptoren allerdings ungeeignet, da sie zum einen verhältnismäßig rechenaufwendig sind und die Oberflächen der zu sortierenden Objekte zum anderen wenig Struktur aufweisen. Zudem sind die



(a) Primitive Deskriptoren: Einfache, schnell berechenbare Merkmale wie Farbe, Gradientenbetrag oder Texturcodes



(b) Dichte Abtastung: Jeder Vordergrundpixel ist ein Keypoint zur Extraktion lokaler Merkmale

Abbildung 3.14: Anpassung von BOW für die Schüttgutsortierung: Primitive Deskriptoren und dichte Abtastung.

Objekte in der Regel klein und bieten so wenig Grundlage zur Berechnung der Deskriptoren.

Andererseits werden viele der Eigenschaften dieser komplexen Deskriptoren, wie die Robustheit gegenüber Beleuchtungsunterschieden und geometrischen Verzerrungen des Bildes, nicht benötigt, da die Aufnahmebedingungen beim Systemdesign beachtet werden und somit unter Kontrolle stehen.

Hier werden stattdessen primitive Deskriptoren verwendet, deren Extraktion wenig Rechenaufwand benötigt. Wie bei den Shape Ferns (Abschnitt 3.3) sind die primitiven Deskriptoren für sich betrachtet wenig aussagekräftig. Eine große Menge primitiver Deskriptoren ist jedoch diskriminativ.

Ein naheliegendes primitives Merkmal ist die Farbe des Objekts an Position $\boldsymbol{u}$ (erste Reihe in Abbildung 3.14a), d. h. mit dem Bild $\boldsymbol{p} : \mathbb{R}^2 \rightarrow \mathbb{R}^3$ als Funktion, die jedem Pixel einen Farbwert zuweist, ist $\psi(\boldsymbol{p}, \boldsymbol{u}) := \boldsymbol{p}(\boldsymbol{u})$. Dieser primitive Deskriptor benötigt nur einen Bildzugriff und gegebenenfalls eine Transformation in einen anderen Farbraum. Die Wahl des Farbraums hat hier Einfluss auf das Vokabular, da die verwendeten Clustering-Verfahren K-Means oder GMM Distanzen zwischen zwei Farbwerten messen. Insbesondere

nichtlineare Transformationen, wie RGB zu HSL oder RGB zu L*a*b*, ändern die Nachbarschaftsbeziehungen zwischen den Farbwerten und führen damit zu unterschiedlichen visuellen Wörtern. In jedem Fall können die visuellen Wörter als Bezeichner bzw. Prototypen bestimmter Farben oder auch als Farbklassen aufgefasst werden. Im Unterschied zum Ansatz aus Abschnitt 1.4 werden diese Farbklassen jedoch ohne Einbringung von Vorwissen ermittelt. Primitive Deskriptoren sind allerdings nicht auf Farbmerkmale beschränkt. Die Textur eines Objekts kann beispielsweise durch den Grauwert, den Gradientenbetrag oder durch Texturcodes[1] wie LBP [OPM02] kodiert werden (zweite Reihe in Abbildung 3.14a). Alle diese Merkmale sind mit wenigen Bildzugriffen sehr schnell berechenbar.
Die Form eines Objekts ist schwieriger zu kodieren, da der BOW-Ansatz den örtliche Zusammenhang der primitiven Deskriptoren verwirft. In gewissen Grenzen kann die Größe und die globale Struktur (länglich, rund etc.) eines Objekts durch die Distanztransformation, d. h. den (örtlichen) Abstand von $\boldsymbol{u}$ zum nächsten Hintergrundpixel, kodiert werden. Die Distanztransformation ist allerdings verhältnismäßig rechenaufwendig und kann die lokale Struktur und insbesondere die Kontur nicht beschreiben. Die Form sollte also besser mit Hilfe von Standardmerkmalen oder des Ansatzes aus Abschnitt 3.3 behandelt werden.

Dichte Abtastung. Die Positionen $\boldsymbol{u}_t$ zur Extraktion der lokalen Deskriptoren ψ werden üblicherweise mit Keypoint-Detektoren wie dem Harris Affine Detector [HS88] oder, wie bei SIFT, dem Difference of Gaussians Operator [Low04] gefunden. Da die zu sortierenden Objekte in der Schüttgutsortierung in der Regel klein sind (vgl. Tabelle 3.3 auf S. 187) und die Oberfläche wenig Struktur aufweist, finden die Keypoint-Detektoren hier nur wenige oder sogar gar keine Keypoints.

[1] Achtung: Texturcodes sind nominale Merkmale, die hier mathematisch unsauber wie metrische Merkmale behandelt werden.

Das hat Einfluss sowohl auf die Lern- als auch auf die Betriebsphase: In der Lernphase wird eine große Anzahl an Trainingsbeispielen $\boldsymbol{p}_n$ benötigt, um ein geeignetes Vokabular zu bestimmen; in der Betriebsphase können nicht genügend lokale Deskriptoren extrahiert werden, um eine zuverlässige Statistik zu berechnen und diskriminative globale Deskriptoren zu erhalten. Die geringe Objektgröße und Kontrolle der Umgebung kann aber auch zum Vorteil genutzt werden, indem jeder Vordergrundpixel als Keypoint aufgefasst wird (siehe Abbildung 3.14b). Die dafür notwendige Trennung von Vorder- und Hintergrund wurde bereits für die Vereinzelung der Objekte vorgenommen (siehe Abbildung 1.3 auf S. 8) und verursacht dadurch keine zusätzlichen Kosten. Diese dichte Abtastung ist jedoch nur praktikabel, wenn die primitiven Deskriptoren schnell genug berechenbar sind.
Hier zeigt sich wieder ein Zusammenhang mit dem Ansatz aus Abschnitt 1.4: Das Vokabular $\mathcal{V}$ übernimmt die Funktion der Attributabbildung und der globale Deskriptor entspricht dem Deskriptor aus Gleichung (1.2). Allerdings ist mit den hier gezeigten Anpassungen von BOW noch keine Detektion unbekannter bzw. anomaler lokaler Deskriptoren und damit keine Rückweisung unbekannter Objekte möglich. Diese Lücke wird im nächsten Abschnitt geschlossen.

3.4.5 Erweiterung um eine Rückweisungsoption

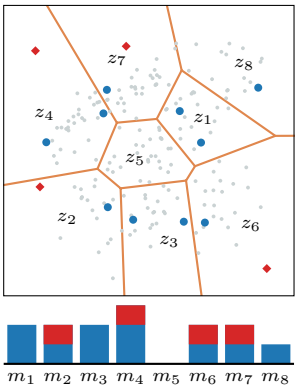
So wie beschrieben, macht der BOW-Ansatz die implizite Annahme einer geschlossenen Welt: Unabhängig von der Kodierung wird bei der Erstellung des globalen Deskriptors jeder lokale Deskriptor $\boldsymbol{x}_t$ betrachtet. Dies betrifft auch Ausreißer in Bezug auf die Lernstichprobe $\mathcal{D}$ (siehe Abbildungen 3.15a und 3.15c).
Das hat zwei Nachteile: Erstens können Verschmutzungen oder andere Störungen den globalen Deskriptor verrauschen, was wiederum die Klassifikationsleistung des Systems mindert. Zweitens geht beim Übergang von den lokalen Deskriptoren zum globalen Objektdeskriptor zwangsweise Informati-

on verloren, die zur Objektrückweisung auf Klassifikatorebene (siehe z. B. Abschnitt 3.5) nicht mehr zur Verfügung steht.
Aus diesen Gründen sollten anomale lokale Deskriptoren vor der Bestimmung des globalen Deskriptors erkannt und herausgefiltert werden. Dazu wird in Anlehnung an eine Klassifikation mit Rückweisung ein zusätzliches visuelles Wort z_0 – das Unbekannt-Wort – eingeführt (siehe Abbildungen 3.15b und 3.15d), welches die anomalen lokalen Deskriptoren kodiert. Neben der Verbesserung des Signal-Rausch-Verhältnisses des globalen Deskriptors kann der Anteil der als Ausreißer zurückgewiesenen lokalen Deskriptoren als Merkmal für die Rückweisungsoption auf Objektebene herangezogen werden.

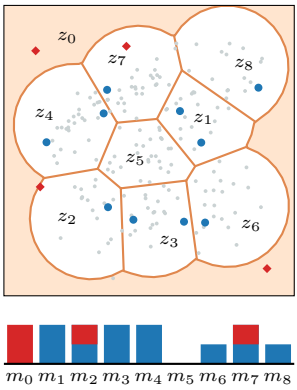
Naiver Ansatz. Die Verwendung von K-Means und der kanonischen Kodierung aus Gleichung (3.58) legt folgenden Ansatz für die Definition des Unbekannt-Wortes z_0 nahe: Ein lokaler Deskriptor $\boldsymbol{x}_t$ wird zurückgewiesen, wenn der Abstand zum nächsten Clusterzentrum $\boldsymbol{\mu}_k$ eine Schwelle $d_{\max}$ überschreitet (siehe Abbildung 3.15b),

$$z_0 := \left\{ \boldsymbol{x} \in \mathbb{M} \,\middle|\, \min_{k=1,\dots,K} \|\boldsymbol{x} - \boldsymbol{\mu}_k\| > d_{\max} \right\}. \tag{3.69}$$

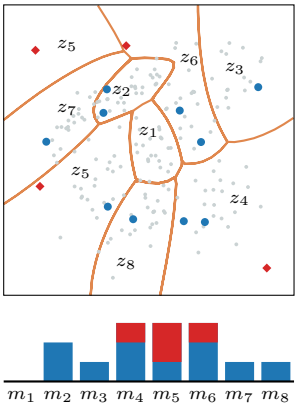
Dieser Ansatz wird auch von Tanaka et al. [TTA13] verfolgt. Das Ziel von Tanaka et al. ist allerdings nicht die Rückweisung unbekannter Objekte, sondern die Verbesserung der Bildkategorisierung durch Entrauschen der Deskriptoren.
Das Hauptproblem bei diesem Vorgehen ist die Wahl eines geeigneten Schwellwerts $d_{\max}$. Ist $d_{\max}$ zu klein, werden zu viele nicht anomale lokale Deskriptoren aussortiert und der globale Deskriptor ist nicht länger diskriminativ. Ist der Schwellwert zu groß, werden Ausreißer nicht länger erkannt und der Deskriptor ist weiterhin verrauscht. Zudem ist $d_{\max}$ schwer zu interpretieren, da geeignete Schwellwerte von der Dimensionalität des Merkmalsraums sowie von der Verteilung der lokalen Deskriptoren in $\mathbb{M}$ abhängen.



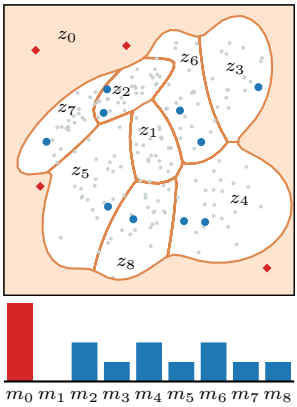
(a) K-Means Clustering, ohne Rückweisung von Ausreißern



(b) K-Means Clustering, Rückweisung nach Gleichung (3.69)



(c) GMM Clustering, ohne Rückweisung von Ausreißern



(d) GMM Clustering, Rückweisung wenn $p(\boldsymbol{x}) < \kappa$

Abbildung 3.15: Annahme der geschlossenen und offenen Welt im BOW Ansatz. Die orangefarbenen Linien zeigen die Grenzen der Regionen z_k, die hellgrauen Punkte zeigen lokale Deskriptoren der Lernstichprobe. Die orange hinterlegte Fläche zeigt das Unbekannt-Wort z_0. Blaue Kreise und rote Rauten stehen für lokale Deskriptoren eines zu klassifizierenden Objekts. Die Rauten stehen dabei für Ausreißer.

Dieser Nachteil kann, analog zu dem weiter unten und in Gleichung (3.79) beschriebenen Vorgehen, durch die Verwendung eines Hilfsparameters ausgeglichen werden. Doch selbst dann ist ein gemeinsamer Schwellwert für alle z_k ungeeignet, da dadurch die lokale Dichte der $\boldsymbol{x}_t$ nicht beachtet wird. Wenn diese lokale Dichte in den z_k wie in Abbildung 3.15b nicht ausreichend homogen ist, ist die Rückweisungsgrenze für manche Regionen nicht eng genug, während sie für andere Regionen zu lose ist (in Abbildung 3.15b z. B. in z_4 und z_6). Diesem Problem könnte durch die Verwendung von wortabhängigen Schwellwerten, d. h.

$$z_0 := \left\{ \boldsymbol{x} \in \mathbb{M} \;\middle|\; \left(\min_{k=1,\ldots,K} \|\boldsymbol{x} - \boldsymbol{\mu}_k\| - d_k \right) > 0 \right\}, \tag{3.70}$$

begegnet werden. Allerdings muss dann für die Festlegung dieser Schwellwerte durch einen Hilfsparameter ein kompliziertes Optimierungsproblem gelöst werden. Außerdem kann auch dann die Isotropie der Euklid'schen Distanz zu losen Rückweisungsgrenzen führen (siehe z. B. z_4 in Abbildung 3.15b).

Dichtebasierter Ansatz. Statt dieses distanzbasierten Ansatzes wird daher eine Methode vorgestellt, die auf einer Schätzung der Merkmalsdichte $p(\boldsymbol{x})$ beruht. Diese Methode nutzt implizit sowohl wortabhängige Schwellwerte als auch ein anisotropes Distanzmaß. Wie bei der FV-Kodierung wird dazu angenommen, dass die lokalen Deskriptoren von einem GMM mit Dichtefunktion

$$p(\boldsymbol{x}) = \sum_{k=1}^{K} P(z_k)\; p(\boldsymbol{x} \mid z_k) \approx \sum_{k=1}^{K} P(z_k)\; \mathcal{N}\left(\boldsymbol{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right) \tag{3.71}$$

erzeugt werden. Zur Vereinfachung der Notation wurde hier auf den Parametervektor $\boldsymbol{\theta} = \left(P(z)\,, \boldsymbol{\mu}, \boldsymbol{\Sigma}\right)^\top$ verzichtet. Diese Dichteschätzung wird sowohl für die Partitionierung des Merkmalsraums und die Definition des globalen

Deskriptors, als auch für die Definition von z_0 und die Rückweisungsoption verwendet.

Objektdeskriptor in der geschlossenen Welt. Wie beim kanonischen Ansatz von Csurka et al. wird der Objektdeskriptor durch harte Zuweisung der lokalen Deskriptoren zu den z_k gewonnen (Gleichung 3.57). Ein lokaler Deskriptor $\boldsymbol{x}_t$ wird dabei derjenigen GMM-Komponente zugewiesen, welche $\boldsymbol{x}_t$ mit höchster a-posteriori Wahrscheinlichkeit generiert hat (siehe Abbildung 3.15c),

$$[\boldsymbol{m}]_k = \frac{T_k}{T} = \frac{1}{T}\sum_{t=1}^{T} \mathbb{1}\left[z_k = \arg\max_z P(z \mid \boldsymbol{x}_t)\right] \tag{3.72}$$

$$= \frac{1}{T}\sum_{t=1}^{T} \mathbb{1}\left[z_k = \arg\max_z P(z)\, p(\boldsymbol{x}_t \mid z)\right]. \tag{3.73}$$

Diese Art der Zuweisung zeigt starke Ähnlichkeit mit der harten Zuweisung mit K-Means in Gleichung (3.58). Der Zusammenhang wird deutlich, wenn der Logarithmus von $P(z_k \mid \boldsymbol{x}_t)$ betrachtet wird,

$$\arg\max_z \{\log P(z \mid \boldsymbol{x})\} = \arg\min_z \beta_z \left(d_{\boldsymbol{\Sigma}_z^{-1}}(\boldsymbol{x}_t, \boldsymbol{\mu}_z)\right)^2 \quad \text{mit} \tag{3.74}$$

$$\left(d_{\boldsymbol{\Sigma}_z^{-1}}(\boldsymbol{x}_t, \boldsymbol{\mu}_z)\right)^2 = (\boldsymbol{x}_t - \boldsymbol{\mu}_z)^\top \boldsymbol{\Sigma}_z^{-1} (\boldsymbol{x}_t - \boldsymbol{\mu}_z) \quad \text{und} \tag{3.75}$$

$$\beta_z = \log\left(P(z)\, |\boldsymbol{\Sigma}_z|^{-1/2}\right). \tag{3.76}$$

Hierbei bezeichnet $\boldsymbol{\mu}_z$ den Erwartungswert und $\boldsymbol{\Sigma}_z$ die Kovarianzmatrix der Gauß-Dichte $p(\boldsymbol{x} \mid z)$. Die MAP-Zuweisung entspricht also einer Zuweisung anhand der (gewichteten) Mahalanobis-Distanz $d_{\boldsymbol{\Sigma}^{-1}}(\boldsymbol{x}, \boldsymbol{\mu})$. Diese Art der Zuweisung ist nur unwesentlich langsamer berechenbar als die harte Zuweisung mit der Euklid'schen Distanz (nach Gleichung 3.58), da sich die Werte von β_z und $\boldsymbol{\Sigma}_z^{-1}$ nach der Bestimmung des Vokabulars nicht mehr ändern und somit vorberechnet werden können.

Statt für die harte Zuweisung können die a-posteriori Wahrscheinlichkeiten aber auch für eine weiche Zuweisung verwendet werden,

$$[\boldsymbol{m}]_k = \frac{T_k}{T} = \frac{1}{T} \sum_{t=1}^{T} P(z_k \,|\, \boldsymbol{x}_t) \,. \tag{3.77}$$

Die a-posteriori-Wahrscheinlichkeit $P(z_k \,|\, \boldsymbol{x}_t)$ übernimmt damit die Funktion des Ähnlichkeitsmaßes aus Gleichung (3.59). Mit dieser Zuweisung kann $[\boldsymbol{m}]_k$ als eine Schätzung der a-posteriori Wahrscheinlichkeit, dass die $\boldsymbol{x}_t$ eines Objekts von der k-ten GMM-Komponente generiert wurden, interpretiert werden.

Das Unbekannt-Wort. Um das visuelle Wort z_0 zu definieren, kann wie in Gleichung (3.69) ein globaler Schwellwert $d_{\max}$ für die gewichtete quadrierte Mahalanobis-Distanz aus Gleichung (3.74) verwendet werden. Da die Mahalanobis-Distanz anisotrop ist, wird die lokale Dichte der Merkmale in den z_k mit einbezogen. Die Gewichtung mit β_{z_k} entspricht wortabhängigen Schwellwerten d_z und beachtet somit auch Inhomogenitäten der lokalen Dichte.
Da zur Berechnung des globalen Deskriptors aber ohnehin jede Komponente des GMMs ausgewertet werden muss, bietet es sich an, die vollständige Dichteschätzung $p(\boldsymbol{x})$ (Gleichung 3.71) zur Definition von z_0 heranzuziehen,

$$z_0 := \{\boldsymbol{x} \in \mathbb{M} \,|\, p(\boldsymbol{x}) < \kappa\} \,. \tag{3.78}$$

Mit diesem Rückweisungskriterium ergibt sich auch eine klare Definition der Ausreißer: Ein lokaler Deskriptor $\boldsymbol{x}_t$ ist ein Ausreißer, wenn er in einer Region geringer Dichte in Bezug auf die Lernstichprobe $\mathcal{D}$ liegt (siehe Abbildung 3.15d).
Die Rückweisungsschwelle κ, unter der lokale Deskriptoren als Ausreißer interpretiert werden, muss dabei vom Benutzer eingestellt werden. Die

Bedeutung dieses Parameters ist allerdings schwer zu interpretieren, da $p(\boldsymbol{x})$ nicht die Auftrittswahrscheinlichkeit von Ausreißern darstellt, sondern die Dichte der lokalen Deskriptoren der Lernstichprobe $\mathcal{D}$ ist und insbesondere $p(\boldsymbol{x}) > 1$ sein kann. Außerdem ist die Wahl eines geeigneten κ, ähnlich wie die eines geeigneten $d_{\max}$, von der Dimensionalität des Merkmalsraums und der Verteilung der lokalen Deskriptoren abhängig.
Hier wird stattdessen ein Hilfsparameter ρ verwendet, um die Rückweisungsschwelle κ festzulegen: Für ein gegebenes ρ wird κ so gewählt, dass

$$\mathbb{E}_{\mathcal{D}}\left\{\mathbb{1}[p(\boldsymbol{x}) < \kappa]\right\} \approx \rho. \tag{3.79}$$

Der Hilfsparameter ρ ist also der vom Benutzer erwartete Anteil der Ausreißer in $\mathcal{D}$ und kann damit als Ausreißerwahrscheinlichkeit $P(p(\boldsymbol{x}) < \kappa)$ lokaler Deskriptoren bekannter Objekte interpretiert werden. Diese Wahrscheinlichkeit repräsentiert somit vor allem Wissen über die Variabilität des Produkts sowie Wissen über den Prozess und die Fehlerquellen in der Bilderfassung.

Klassifikation mit Rückweisungsoption. Unabhängig von der Berechnung der $[\boldsymbol{m}]_1, \ldots, [\boldsymbol{m}]_K$ durch harte oder weiche Zuweisung wird der Anteil der Ausreißer in den lokalen Deskriptoren eines Objekts durch

$$[\boldsymbol{m}]_0 := \frac{T_0}{T} = \frac{1}{T}\sum_{t=1}^{T} \mathbb{1}[\boldsymbol{x}_t \in z_0] \tag{3.80}$$

gemessen. Ein Objekt wird als unbekannt zurückgewiesen, wenn zu viele seiner lokalen Deskriptoren unbekannt sind, bzw. wenn der Anteil der unbekannten Deskriptoren eine Grenze τ überschreitet, $[\boldsymbol{m}]_0 > \tau$ mit $0 \leq \tau \leq 1$.

Mit einem beliebigen Klassifikator $\hat{\omega}'(\boldsymbol{x})$ lautet die Klassifikationsregel mit Rückweisung

$$\hat{\omega}(\boldsymbol{x}) = \begin{cases} \omega_0 & \text{wenn } [\boldsymbol{m}]_0 > \tau \\ \hat{\omega}'(\boldsymbol{x}) & \text{sonst.} \end{cases} \tag{3.81}$$

Der Parameter τ ist hier als Ausreißerwahrscheinlichkeit der lokalen Deskriptoren zu verstehen, nicht aber als die Wahrscheinlichkeit, dass das Objekt selbst ein Ausreißer ist. Insbesondere ist $[\boldsymbol{m}]_0$ und damit auch die Wahl eines geeigneten Schwellwerts τ von der Ausreißerrate ρ aus Gleichung (3.79) abhängig.
In Abschnitt 3.4.7 werden zwei alternative, von statistischen Hypothesentests inspirierte Methoden zur Rückweisung unbekannter Objekte vorgestellt, die diese Abhängigkeit lösen. Da sich die verschiedenen Parametrierungen aber ineinander überführen lassen, wird für die Analyse der Methode im nächsten Abschnitt das hier vorgestellte, einfache Rückweisungsschema verwendet.
Neben der Rückweisung unbekannter Objekte wird das Unbekannt-Wort z_0 wie in Tanaka et al. [TTA13] auch zur Bereinigung des Objektdeskriptors verwendet. Dazu werden bei der Berechnung von $\boldsymbol{m} = \left([\boldsymbol{m}]_1, \ldots, [\boldsymbol{m}]_K\right)^\top$ nur die nicht als Ausreißer erkannten lokalen Deskriptoren mit einbezogen (vergleiche Abbildungen 3.15c und 3.15d).

3.4.6 Experimente und Ergebnisse

Zur Evaluation der hier vorgestellten Methoden wurden acht Datensätze verwendet. Sieben der Datensätze stehen exemplarisch für Sortieraufgaben in der Lebensmittelinspektion und wurden im Rahmen der Inbetriebnahme realer Sortiersysteme gewonnen. Der achte Datensatz ist der Kieselsteindatensatz aus Abschnitt 3.2 und steht exemplarisch für die Mineralsortierung. Dieser Datensatz wurde ebenfalls mit einem Schüttgutsortiersystem des Fraunhofer IOSB aufgenommen. Die Datensätze wurden folgendermaßen eingeteilt:

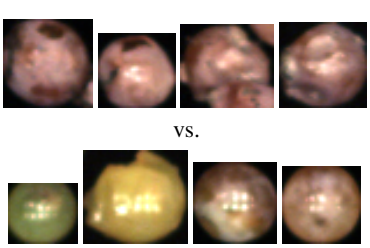
(A) Trennung gesunder Weinbeeren von unreifen und verfaulten Beeren sowie von Pflanzenteilen. Der Datensatz enthält die drei Sorten Riesling, Weißburgunder und Spätburgunder, die hier getrennt voneinander betrachtet wurden und im Folgenden mit A-1, A-2 und A-3 bezeichnet werden. Da die Spätburgunderbeeren relativ dunkel sind und sich nicht vom schwarzen Hintergrund abheben, wurde bei der Aufnahme der Blaukanal der RGB-Kamera durch einen Nahinfrarotkanal (NIR-Kanal) ausgetauscht. Der gleiche Datensatz wurde bereits in Abschnitt 1.4 verwendet.

(B) Bestimmung des Zuckergehalts von Weinbeeren der Sorte Gewürztraminer als entweder „hoch" oder „niedrig". Der Zuckergehalt der Beeren wurde dabei vor der Bildaufnahme bestimmt. Wie bei den Spätburgunderbeeren wurde der Blaukanal der Kamera durch einen NIR-Kanal ausgetauscht, da die in Zucker enthaltenen OH-Gruppen im NIR-Spektrum sichtbar werden.

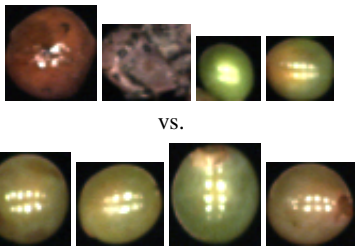
(C) Trennung gesunder Weizenkörner von mit einem Pilz befallenen Weizenkörnern (C-1), von Fremdkulturen wie Hafer oder Mais (C-2), sowie von Besatz, d. h. von kleinen Steinen, verschrumpelten Körnern und sonstigem Schmutz (C-3). In Datensatz C-1 unterscheiden sich die Positiv- und Negativklasse lediglich durch eine dunkle Verfärbung am Ende des Korns.

(D) Unterscheidung von grauen Kieselsteinen mit dunklen Flecken und blau-grauen Kieselsteinen mit hellen Flecken. Da die Steine eine sehr ähnliche Farbverteilung haben, wird erwartet, dass lediglich Farbmerkmale zur Klassifikation nicht ausreichen. Dieser Datensatz wurde auch in Abschnitt 3.2 (siehe S. 142) verwendet.

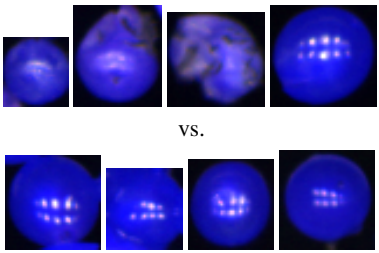
Eine Übersicht und Beispiele der Datensätze finden sich in Tabelle 3.3 und Abbildung 3.16.



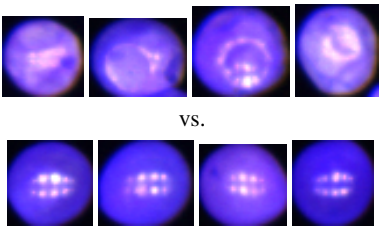
(a) Datensatz A-1: Riesling



(b) Datensatz A-2: Weißburgunder

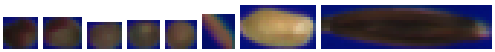


(c) Datensatz A-3: Spätburgunder



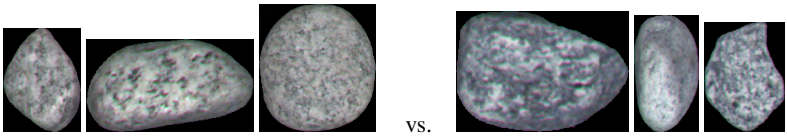
(d) Datensatz B: Gewürztraminer

Gesunde Körner Pilzbefall Fremdkulturen



Besatz

(e) Datensätze C-1 bis C-3: Weizen



(f) Datensatz D: Kieselsteine

Abbildung 3.16: Beispielbilder der in den Experimenten verwendeten Datensätze. Die Bildgröße spiegelt nicht die Objektgröße wieder. Nicht gezeigt: Pflanzenteile in Datensätzen A-1 bis A-3.

Tabelle 3.3: Überblick über die zur Evaluation der BOW-Methoden verwendeten Datensätze.

Datensatz		**# Beispiele**		**Objektfläche T / Pixel**		
		positiv	negativ	min	max	median
A-1	Riesling	235	1061	98	6035	1008
A-2	Weißburgunder	291	332	25	56267	1212
A-3	Spätburgunder	641	2416	61	28489	1089
B	Gewürztraminer	211	248	147	2936	1578
C-1	Pilzbefall	396	1145	25	822	332
C-2	Fremdkulturen	396	1625	25	822	372
C-3	Besatz	396	1293	25	822	185
D	Kieselsteine	1213	3418	3019	70686	12272

Die verwendeten lokalen Deskriptoren kodieren Farbe und Textur. Farbe wurde dabei mit Hilfe der Farbkanäle im RGB-, XYZ- oder L*a*b*-Farbraum repräsentiert. Zur Beschreibung der Textur wurden Grauwert, rotationsinvariante uniforme LBP [OPM02] und Gradientenbeträge verschiedener Skalen herangezogen. Für Letztere wurde das Grauwertbild zunächst mit Gauß-Kerneln mit den Filterbreiten $\sigma \in \{0; 1; 1{,}5; 2; 2{,}5\}$ gefaltet und der Gradientenbetrag mit Hilfe des Differenzenquotienten der benachbarten Pixel berechnet. Somit ergab sich ein 10-dimensionaler lokaler Deskriptor $\psi(\boldsymbol{p})$. Die lokalen Deskriptoren wurden vor der Bestimmung des Vokabulars bzw. Berechnung des globalen Deskriptors durch eine Whitening-Transformation dekorreliert. Die Parameter dieser Transformation wurden in der Lernphase nach der Bestimmung des Vokabulars anhand der globalen Deskriptoren des Lerndatensatzes festgelegt. In allen Experimenten wurde die Klasseninformation genutzt, um klassenspezifische Vokabulare zu erstellen.

Die globalen Deskriptoren wurden mit Hilfe einer linearen soft margin SVM klassifiziert. Der Hyperparameter C wurde mit einer randomisierten Suche im Intervall $C \in [2^{-5}, 2^5]$ festgelegt. Zum Training wurde das primale Optimierungsproblem verwendet, da der Umfang der Lernstichprobe die Dimensionalität des Merkmalsraums in den Experimenten deutlich überstieg. Alle Datensätze und Parameterkombinationen wurden mit Hilfe einer stratifizierten 5-fold Cross Validation evaluiert. Dabei wurde jeweils die Hälfte der Lernstichprobe zur Bestimmung des Vokabulars und die andere Hälfte der Lernstichprobe zur Festlegung des SVM-Klassifikators verwendet.

Klassifikation in der geschlossenen Welt. Die besten Ergebnisse für die Klassifikation in der geschlossenen Welt, d. h. mit globalen Deskriptoren ohne Verwendung des Unbekannt-Worts, sind in Tabelle 3.4 aufgeführt. Mit Ausnahme von Datensatz C-1 (Weizen mit Pilzbefall) werden in allen Fällen sehr gute Klassifikationsergebnisse erzielt. In allen Fällen ist die Fisher-Vektorkodierung der kanonischen Kodierung mit harter oder weicher Zuweisung überlegen. Bis auf die Datensätze A-3, B und D ist der Güteunterschied zwischen FV-Kodierung und harter Zuweisung allerdings gering. Da die harte Zuweisung aber schneller berechenbar ist und kompaktere und einfacher zu interpretierende globale Deskriptoren erzeugt, ist diese Kodierung der FV-Kodierung in der Praxis zu bevorzugen. Weiche Zuweisung nach Gleichung (3.60) ist in allen Fällen der harten Zuweisung und der FV-Kodierung unterlegen und wird deshalb in der Tabelle nicht gezeigt.
Mit den Datensätzen A-2, C-3 und D wird eine annähernd perfekte Klassifikation erreicht. Tabelle 3.5 zeigt aber, dass hier mit den Ansätzen aus den Abschnitten 1.4 und 3.2 sowie mit Farbmomenten und einer SVM mit RBF-Kernel ebenfalls gute Ergebnisse erzielt werden können; diese Datensätze sind verhältnismäßig einfach. Mit dem Teach'N'Sort-Ansatz aus Abschnitt 1.4 wird mit dem Datensatz A-1 im Mittel sogar eine bessere Klassifikationsgüte erreicht als mit dem hier gezeigten Ansatz. Allerdings ist die Klassifikation mit dem Teach'N'Sort-System deutlich instabiler.

Tabelle 3.4: Beste Ergebnisse der Klassifikation in der geschlossenen Welt. MCC mit Angabe der Standardabweichung über eine stratifizierte 5-fold Cross Validation.

DS	Kodierung	Farbraum	Normalisierung	MCC
A-1	Fisher-Vektor	XYZ	Potenz	0,91 ±0,016
A-2		XYZ	L2, Potenz	0,99 ±0,014
A-3		XYZ	Potenz	0,88 ±0,024
B		XYZ	L2, Potenz	0,95 ±0,012
C-1		L*a*b*	Potenz, L2	0,73 ±0,027
C-2		L*a*b*	Potenz	0,91 ±0,058
C-3		XYZ	Potenz	0,99 ±0,011
D		L*a*b*	L1	0,99 ±0,014
A-1	Harte Zuweisung	XYZ	—	0,90 ±0,052
A-2	(Gleichung 3.58)	XYZ	L2, Potenz	0,98 ±0,016
A-3		XYZ	L2, Potenz	0,80 ±0,025
B		XYZ	L2, Potenz	0,90 ±0,045
C-1		L*a*b*	Potenz, L2	0,70 ±0,043
C-2		L*a*b*	L2, Potenz	0,90 ±0,035
C-3		XYZ	L2, Potenz	0,99 ±0,011
D		L*a*b*	Potenz	0,96 ±0,006

Da fast alle Ergebnisse in Tabelle 3.5 nur auf Grundlage von Farbmerkmalen erzielt wurden, stellt sich die Frage, wie stark die Klassifikationsleistung durch Verwendung der Texturmerkmale beeinflusst wird. Zur Beantwortung dieser Frage wurden die Datensätze unter Verwendung von ausschließlich Farbmerkmalen und harter Zuweisung nach Gleichung (3.58) evaluiert.

Tabelle 3.5: Klassifikationsgüte mit anderen Ansätzen. MCC mit Angabe der Standardabweichung über eine stratifizierte 5-fold Cross Validation.

DS	Ansatz	MCC
A-1	Farbmomente und SVM mit RBF-Kernel	0,80 ±0,149
A-2		0,52 ±0,246
A-3		0,40 ±0,312
B		0,71 ±0,121
C-1		0,29 ±0,079
C-2		0,58 ±0,028
C-3		0,96 ±0,020
D		0,88 ±0,009
A-1	Teach'N'Sort (Abschnitt 1.4)	0,99 ±0,20
A-2		0,93 ±0,08
A-3		0,71 ±0,04
D	Merkmalsselektion (Abschnitt 3.2)	0,86

Die besten Klassifikationsergebnisse sind in Abbildung 3.17 zusammengefasst. Ausschließlich bei Datensatz D verbessert sich die Klassifikationsgüte durch Hinzunahme von Texturmerkmalen signifikant. Mit den Datensätzen A-2 und C-3 scheint die Klassifikation von den Texturmerkmalen zu profitieren, allerdings kann hier nicht ausgeschlossen werden, dass die Verbesserung der Klassifikationsgüte zufällig bedingt ist. Alle verbleibenden Datensätze profitieren nicht von Texturmerkmalen. Zur Einsparung von Rechenzeit könnte hier also auf Texturmerkmale verzichtet werden.
Die Größe des visuellen Vokabulars beeinflusst ebenfalls die erreichbare Klassifikationsgüte und den benötigten Rechenaufwand. Abbildung 3.18 zeigt den MCC in Abhängigkeit der Größe des visuellen Vokabulars für

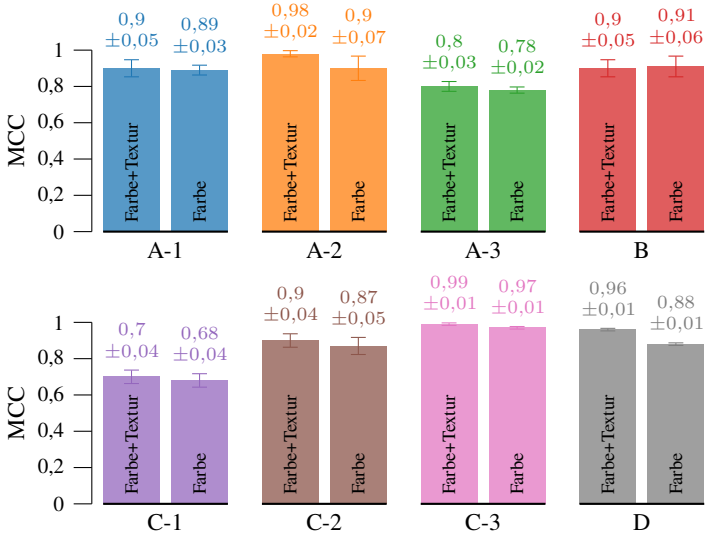
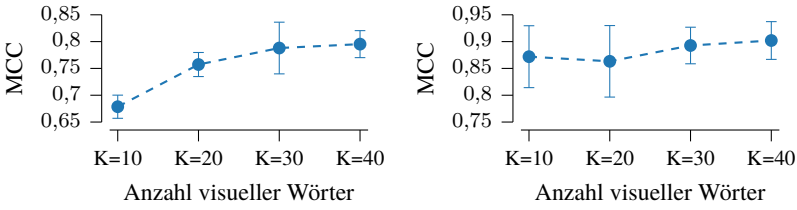


Abbildung 3.17: Klassifikationsgüte mit unterschiedlichen Merkmalskanälen und harter Zuweisung. Die Fehlerbalken zeigen die Standardabweichung über eine stratifizierte 5-fold Cross Validation.

die Datensätze A-3 und C-2. Datensatz A-3 profitiert von einem größeren Vokabular, es besteht jedoch kein linearer Zusammenhang zwischen der Größe des Vokabulars und der Klassifikationsgüte. Ab $K = 20$ ist keine signifikante Verbesserung sichtbar. Bei Datensatz C-2 scheint sich die Größe des Vokabulars nur auf die Varianz der Klassifikationsgüte auszuwirken, nicht aber auf die mittlere Klassifikationsgüte. Die Hinzunahme weiterer visueller Worte reduziert hier die Varianz. Mit allen Datensätzen ist ab einer Größe von $K \geq 30$ keine signifikante Verbesserung der Klassifikationsgüte erkennbar. Im Vergleich zu anderen Anwendungen von BOW sind die Vokabulare hier also sehr klein. Die Gründe hierfür sind die verhältnismäßig einfachen Datensätze und die geringe Dimension von $\mathbb{M}$.

In der praktischen Anwendung sind die Lernstichproben in der Regel deutlich kleiner als die hier verwendeten Datensätze. Verfahren für den Einsatz in



(a) Datensatz A-3 **(b)** Datensatz C-2

Abbildung 3.18: Klassifikationsgüte in Abhängigkeit der Größe des visuellen Vokabulars. Kodierung nach Gleichung (3.58), Farb- und Texturmerkmale. Die Fehlerbalken zeigen die empirische Standardabweichung über eine 5-fold Cross Validation.

der visuellen Inspektion sollten aber auch mit kleinen Lernstichproben gute Ergebnisse erzielen. Abbildung 3.19 zeigt die Güte der Klassifikation in Abhängigkeit der Anzahl an Trainingsdaten auf dem schwierigsten Datensatz C-1. Wie bei den anderen Experimenten wurde die Hälfte der Trainingsdaten zur Generierung des Vokabulars und die andere Hälfte zum Training der SVM verwendet. Die Klassifikationsgüte nimmt bis ca. 400 Trainingsbeispiele schnell zu, danach flacht der Zuwachs ab. Ab ca. 800 Beispielen nimmt die Klassifikationsgüte nicht mehr signifikant zu. Ähnliche Beobachtungen wurden mit den anderen Datensätzen gemacht. Diese Beobachtung legt nahe, dass für den praktischen Einsatz dieser Methode die Bereitstellung weniger hundert Trainingsbeispiele ausreicht.

Ein weiterer Faktor für die praktische Anwendung ist die benötigte Rechenzeit. Das Verfahren wurde in der JIT-kompilierten Programmiersprache Julia [Bez+17] implementiert und auf einem handelsüblichen Rechner mit 2,4 GHz Intel i7 CPU evaluiert. Die im Mittel benötigte Zeit für die Merkmalsextraktion und Klassifikation pro Objekt beträgt weniger als 65 ms. Für den praktischen Einsatz dauert die Merkmalsextraktion jedoch noch zu lange: Je nach Objektgeschwindigkeit stehen hier üblicherweise ca. 20 ms zur Bilderfassung, Objektdetektion, Klassifikation und Ausschleusung zur

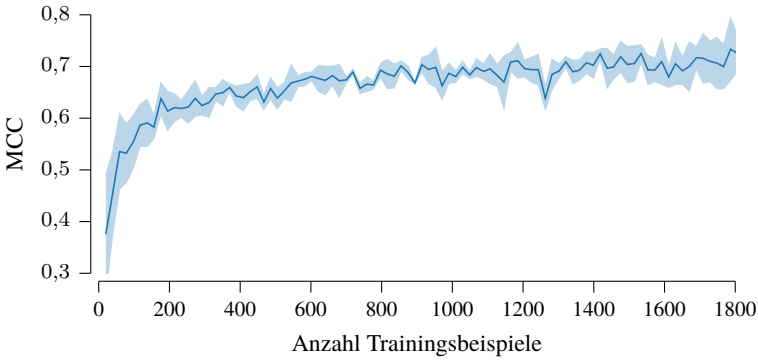
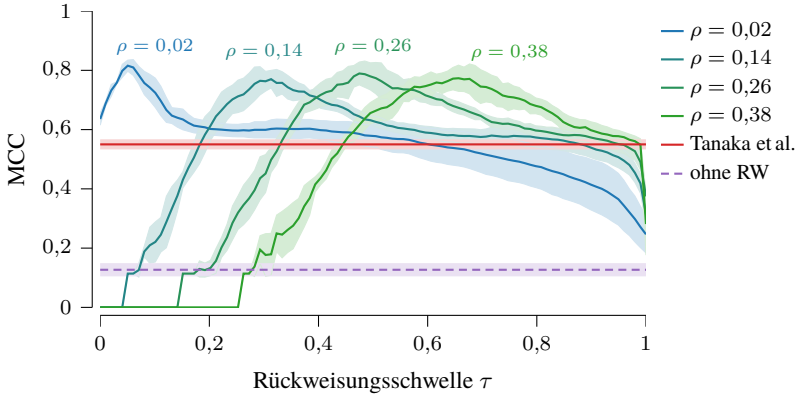


Abbildung 3.19: Klassifikationsgüte in Abhängigkeit der Anzahl an Trainingsdaten. Datensatz C-1, Kodierung nach Gleichung (3.58), Farb- und Texturmerkmale. Die schattierte Fläche zeigt die empirische Standardabweichung über eine 5-fold Cross Validation.

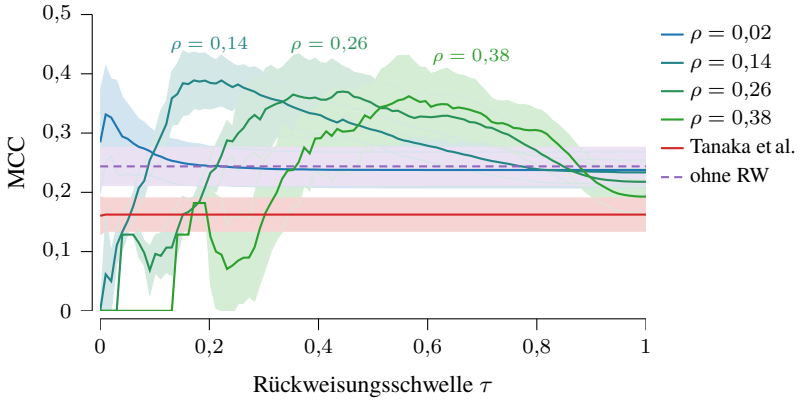
Verfügung. Allerdings besteht durch die prototypische Implementierung noch Optimierungspotential. Beispielsweise können die lokalen Deskriptoren bei der Berechnung des globalen Deskriptors unterabgetastet, die Erstellung der globalen Deskriptoren durch Verwendung von Lookup-Tabellen beschleunigt, oder Berechnung der Distanzen auf mehrere CPU-Threads verteilt werden. Ebenfalls können Teile des Verfahrens in Hardware, z. B. auf einer Framegrabber-Karte, implementiert werden.

Klassifikation in der offenen Welt. Zur Evaluation der Rückweisungsoption aus Abschnitt 3.4.5 wurden die Datensätze A-3 (Spätburgunder) und C-2 (Weizen und Fremdkulturen) verwendet. In der Betriebsphase wurden die unbekannten Objekte in beiden Fällen aus den Datensätzen A-1 (Riesling) und A-2 (Weißburgunder) entnommen. Da für Datensatz A-3 der Blaukanal durch einen IR-Kanal ausgetauscht wurde, haben die unbekannten Objekte ein deutlich anderes Erscheinungsbild als die bekannten Objekte (siehe Abbildung 3.16). Hier sollte die Anomaliedetektion also gut funktionieren.

Dagegen ist bei Datensatz C-2 die Farbe und Textur der bekannten und unbekannten Objekten ähnlich, wodurch die Detektion unbekannter Objekte erschwert wird. Zur Berechnung der Klassifikationsgüte wurden unbekannte Objekte der Negativklasse zugeordnet. Dies ist durch die Anwendung in der Schüttgutsortierung motiviert, da unbekannte Objekte hier in der Regel aussortiert werden sollen.
Abbildung 3.20 zeigt die Klassifikationsgüte in Abhängigkeit der Rückweisungsschwelle τ für verschiedene Ausreißerraten ρ (siehe Gleichungen 3.81 und 3.79). Wie erwartet ist die Güte der Klassifikation in der offenen Welt im Allgemeinen geringer als die Güte der Klassifikation in der geschlossenen Welt. Einerseits werden unbekannte Objekte fehlklassifiziert, andererseits werden eigentlich bekannte Objekte als unbekannt zurückgewiesen.
Mit Datensatz A-3 ist der maximale $\mathrm{MCC} \approx 0{,}8$ für alle getesteten Ausreißerraten ρ, aber dieses Maximum wird mit verschiedenen Rückweisungsschwellen τ erreicht. Das ist naheliegend, da bei kleinerem ρ mehr lokale Deskriptoren zum Unbekannt-Wort z_0 gezählt werden. Anders als beim Datensatz A-3 ist die maximale Klassifikationsgüte beim Datensatz C-2 abhängig von ρ und τ. Die maximale Güte von $\mathrm{MCC} \approx 0{,}4$ wird mit $\rho = 0{,}14$ und $\tau \approx 0{,}18$ erreicht. Das ist deutlich schlechter als die Klassifikation in der geschlossenen Welt, wo $\mathrm{MCC} \approx 0{,}9$ erzielt wurde.
Bei beiden Datensätzen ist aber die Klassifikation mit Rückweisung der Klassifikation ohne Rückweisungsoption überlegen. Hier wird bei Datensatz C-2 die Güte $\mathrm{MCC} \approx 0{,}24$ und mit Datensatz A-3 sogar nur $\mathrm{MCC} \approx 0{,}13$ erreicht.
Der hier vorgestellte Ansatz ist auch dem Ansatz von Tanaka et al. [TTA13], d. h. mit einer Rückweisungsregion nach Gleichung (3.69), überlegen. Der Parameter $d_{\max}$ wurde dabei wie κ durch die Ausreißerrate ρ bestimmt. Hier wurde $\rho = 0{,}15$ gewählt, sodass ca. $15\,\%$ der lokalen Deskriptoren der Lernstichprobe als Ausreißer gewertet wurden. Dieser Parameter hatte in den Experimenten allerdings wenig Einfluss auf das qualitative Ergebnis; die Klassifikationsgüte ist nahezu unabhängig von τ. Mit Datensatz A-3 wird die



(a) Datensatz A-3, unbekannte Objekte aus A-1 und A-2



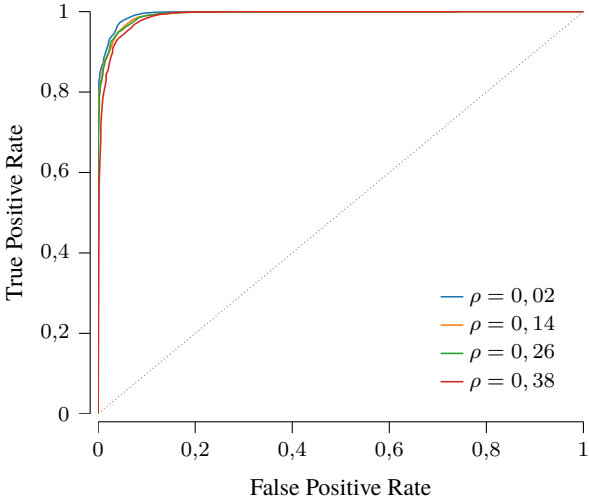
(b) Datensatz C-2, unbekannte Objekte aus A-1 und A-2

Abbildung 3.20: MCC in Abhängigkeit der Rückweisungsschwelle τ und der Ausreißerrate ρ. Vergleich mit Tanaka et al. [TTA13] und Klassifikation ohne Rückweisung (ohne RW). Die schattierten Flächen zeigen die Standardabweichung über eine 5-fold Cross Validation.

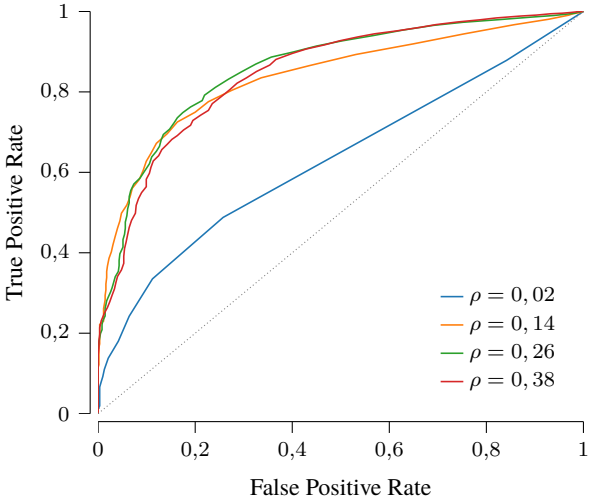
Güte $\mathrm{MCC} \approx 0{,}55$ und mit Datensatz C-2 die Güte $\mathrm{MCC} \approx 0{,}17$ erreicht. Letzteres ist sogar schlechter als eine Klassifikation ohne Rückweisung. Ein Grund für dieses Ergebnis kann sein, dass die lokalen Deskriptoren sehr inhomogen im Merkmalsraum verteilt sind, wodurch ein gemeinsamer Schwellwert $d_{\max}$ wie oben beschrieben für visuelle Worte in dünn besiedelten Regionen von $\mathbb{M}$ zu eng und für visuelle Worte in dichteren Regionen zu lose ist. Als Resultat wurden für den gezeigten Parameter $\rho = 0{,}15$ während der Auswertung nur sehr wenige lokale Deskriptoren als Ausreißer markiert.
Abschließend zeigt Abbildung 3.21 die ROC Kurven für die Rückweisungsoption der beiden Datensätze. Hierfür wurde die Detektion der Klasse ω_0 als binäres Klassifikationsproblem aufgefasst und der Schwellwert τ aus Gleichung (3.81) verschoben. Mit Datensatz A-3, d. h. bei visuell unterschiedlichen Objekten, ist der Anteil an Ausreißern $[\boldsymbol{m}]_0$ ein sehr gutes Detektionsmerkmal. Sind die unbekannten Objekte den bekannten Objekten visuell ähnlich, wie in Datensatz C-2, sinkt die Güte des Rückweisungsmerkmals. Das ist insbesondere dann der Fall, wenn die durch ρ festgelegte Grenze der Ausreißerdetektion in den lokalen Deskriptoren zu weit ist (hier $\rho = 0{,}02$).

3.4.7 Alternative Parametrierung der Rückweisungsoption

Wie in Abschnitt 3.4.5 bereits erwähnt wurde, beeinflusst der Parameter ρ auch die Wahl des Objektrückweisungsparameters τ, wodurch die Wahl einer geeigneten Parameterkombination erschwert wird. Mit dem Ziel, diese Abhängigkeit zu lösen, wird in diesem Abschnitt daher der theoretische Zusammenhang von ρ und τ analysiert. Mit Hilfe des gewonnenen Wissens werden zwei alternative, von ρ entkoppelte Parametrierungen der Rückweisungsoption entwickelt.
Ein Zugang findet sich in der Analyse des Rückweisungsmerkmals $m_0 := [\boldsymbol{m}]_0$ im Kontext der Theorie der Mustererkennung: In Abschnitt 2.1 wurden die Objekte $o \in \Omega$ als Zufallsvariablen eines Zufallsprozesses $p(o, \omega)$ modelliert.



(a) Datensatz A-3



(b) Datensatz C-2

Abbildung 3.21: ROC Kurven für die Detektion unbekannter Objekte mit variierendem Rückweisungsparameter τ.

Damit sind die vom Objekt extrahierten lokalen Deskriptoren $\mathcal{O} = \{\boldsymbol{x}_t\}_{t=1}^{T}$ und die Anzahl T der Vordergrundpixel ebenfalls Zufallsvariablen. Daraus folgt, dass das Rückweisungsmerkmal

$$m_0 := [\boldsymbol{m}]_0 = \frac{1}{T} \sum_{\boldsymbol{x}_t \in \mathcal{O}} \mathbb{1}[p(\boldsymbol{x}_t) < \kappa] \approx \mathbb{E}_{\mathcal{O}} \{\mathbb{1}[p(\boldsymbol{x}) < \kappa]\} \tag{3.82}$$

ebenfalls eine Zufallsvariable ist, die einer bestimmten Verteilung folgt. Unter der Annahme, dass die $\boldsymbol{x}_t$ i. i. d. gegeben $\mathcal{O}$ sind, sind auch die $p(\boldsymbol{x}_t)$ unabhängige und identisch verteilte Zufallsvariablen. Somit lässt sich der Test $\mathbb{1}[p(\boldsymbol{x}_t) < \kappa]$ als Zufallsexperiment mit zwei Ausgängen ($\boldsymbol{x}_t$ ist ein Ausreißer oder nicht) auffassen. Die Anzahl der Erfolge dieses Experiments, also die Anzahl an Ausreißern in $\mathcal{O}$, ist

$$M_0 := \sum_{\boldsymbol{x}_t \in \mathcal{O}} \mathbb{1}[p(\boldsymbol{x}_t) < \kappa] = T \cdot m_0. \tag{3.83}$$

Da M_0 also die Anzahl der Erfolge einer Reihe von gleichartigen, aber unabhängigen Tests ist, folgt diese Zufallsvariable einer Binomialverteilung, also $M_0 \sim \mathfrak{B}(T, p)$, mit den Verteilungsparametern Anzahl T der Experimente bzw. der lokalen Deskriptoren und der Erfolgs- bzw. Ausreißerwahrscheinlichkeit $p = P(p(\boldsymbol{x}) < \kappa)$.
Der Parameter T ist durch das Objekt vorgegeben (und somit selbst auch eine Zufallsvariable), aber der Parameter p ist im Allgemeinen nicht bekannt. Da über unbekannte Objekte $o \in \omega_0$ nichts bekannt ist, kann hier keine Aussage getroffen werden. Für bekannte Objekte $o \notin \omega_0$ (kurz $o \in \overline{\omega_0}$) indes schon.
Eine Grundannahme der Mustererkennung ist, dass die Lernstichprobe $\mathcal{D}$ die in der Zukunft angetroffenen zu klassifizierenden Objekte repräsentiert. Da diese Lernstichprobe nur bekannte Objekte enthält, folgt, dass

$$P(p(\boldsymbol{x}) < \kappa \,|\, \overline{\omega_0}) \approx P(p(\boldsymbol{x}) < \kappa \,|\, \mathcal{D}) = \mathbb{E}_{\mathcal{D}} \{\mathbb{1}[p(\boldsymbol{x}) < \kappa]\} \overset{(*)}{\approx} \rho. \tag{3.84}$$

Die Annäherung $(*)$ ergibt sich aus Gleichung (3.79): κ wurde so gewählt, dass ρ diesem Erwartungswert entspricht. Gleichzeitig bestätigt die obige Gleichung die Interpretation von ρ als Vorwissen über die Ausreißerwahrscheinlichkeit der lokalen Merkmale. Für bekannte Objekte $o \in \overline{\omega_0}$ folgt M_0 also einer Binomialverteilung mit Parametern T und ρ, also $M_0 \,|\, \overline{\omega_0} \sim \mathfrak{B}\,(T,\rho)$. Da $M_0 \,|\, \omega_0$ ebenfalls binomialverteilt ist, könnte mit zusätzlichen Annahmen der Verteilungsparameter ein MAP-Klassifikator für die Rückweisungsoption konstruiert werden. Da solche Annahmen in der Praxis aber schwer begründbar sind, soll hier darauf verzichtet werden. Stattdessen wird hier das Rückweisungskriterium aus Gleichung (3.81),

$$\hat{\omega}\,(\mathcal{O}) = \begin{cases} \omega_0 & \text{für } m_0 > \tau \;\Leftrightarrow M_0 > T\,\tau, \\ \overline{\omega_0} & \text{sonst} \end{cases} \tag{3.85}$$

mit dem Wissen über $M_0 \,|\, \overline{\omega_0}$ weiter untersucht. Die Fehlerwahrscheinlichkeit dieses Klassifikators setzt sich aus der Wahrscheinlichkeit der Rückweisung eines bekannten Objekts und der Wahrscheinlichkeit der Akzeptanz eines unbekannten Objekts zusammen, also

$$P_e = \underbrace{P(m_0 > \tau \,|\, \overline{\omega_0})\; P(\overline{\omega_0})}_{=P_{e1}} + \underbrace{P(m_0 \leq \tau \,|\, \omega_0)\; P(\omega_0)}_{=P_{e2}}. \tag{3.86}$$

Im Kontext statistischer Tests werden die Terme P_{e1} und P_{e2} auch als Fehler der 1. und Fehler der 2. Art eines Hypothesentests mit der Null-Hypothese $\hat{\omega}(\mathcal{O}) = \overline{\omega_0}$ bezeichnet. Damit können also gängige Verfahren aus der Praxis statistischer Tests auf das Parametrierungsproblem übertragen werden.

z-score. Da eine Binomialverteilung mit großem T sich gut durch eine Normalverteilung approximieren lässt, kann angenommen werden, dass auch das normalisierte Rückweisungskriterium $m_0|\overline{\omega_0}$ annähernd einer Normalverteilung mit Mittelwert $\mathbb{E}_{\mathcal{D}}\,\{m_0\}$ und Varianz $\mathrm{Var}_{\mathcal{D}}\,\{m_0\}$ folgt. Zur Bewertung

der Normalität solcher Variablen wird gelegentlich der Standardscore bzw. der z-score

$$z = \frac{m_0 - \mathbb{E}_{\mathcal{D}}\{m_0\}}{\sqrt{\mathrm{Var}_{\mathcal{D}}\{m_0\}}}, \tag{3.87}$$

herangezogen. Der z-score misst die Abweichung von Erwartungswert in Anzahl der Standardabweichungen und damit implizit die Wahrscheinlichkeit, dass m_0 aus der vorgegebenen Verteilung generiert wurde. Im Umkehrschluss kann der z-score aber auch verwendet werden, um ein τ für einen vorgegebenen erwarteten Fehler 1. Art zu ermitteln:

$$\tau = \mathbb{E}_{\mathcal{D}}\{m_0\} + z\sqrt{\mathrm{Var}_{\mathcal{D}}\{m_0\}}. \tag{3.88}$$

Hier ist z also ein Parameter, mit dem die erwartete Fehlerwahrscheinlichkeit P_{e1} kodiert wird: Für $z = 0$ ist der Fehler $P_{e1} = 0{,}5$, für $z = 1$ ist $P_{e1} \approx 0{,}16$, für $z = 3$ ist $P_{e1} < 0{,}01$ etc.
Die beiden benötigten Größen $\mathbb{E}_{\mathcal{D}}\{m_0\}$ und $\mathrm{Var}_{\mathcal{D}}\{m_0\}$ können aus den Daten geschätzt, aber auch analytisch bestimmt werden. Für den Erwartungswert gilt

$$\begin{aligned}
\mathbb{E}_{\mathcal{D}}\{m_0\} &= \mathbb{E}_T\left\{\mathbb{E}_{\mathcal{D}}\left\{\frac{M_0}{T}\,\middle|\,T\right\}\right\} \quad \text{(Law of Total Expectation)} && (3.89)\\
&= \mathbb{E}_T\left\{\frac{\mathbb{E}_{\mathcal{D}}\{M_0 \mid T\}}{T}\right\} && (3.90)\\
&= \mathbb{E}_T\left\{\frac{\rho\, T}{T}\right\} = \rho \qquad (M_0 \sim \mathfrak{B}(T,\rho)). && (3.91)
\end{aligned}$$

Hier wurde $m_0 = \frac{M_0}{T}$ und das Law of Total Expectation (siehe Weiss et al. [WHH05]), $\mathbb{E}\{X\} = \mathbb{E}\{\mathbb{E}\{X \mid Y\}\}$, sowie der Erwartungswert der Binomialverteilung verwendet. Für die Varianz von m_0 gilt mit dem Law of

Total Variance, $\mathrm{Var}\{X\} = \mathbb{E}\{\mathrm{Var}\{X \mid Y\}\} + \mathrm{Var}\{\mathbb{E}\{X \mid Y\}\}$ (siehe ebenfalls [WHH05]), weiterhin

$$\mathrm{Var}_{\mathcal{D}}\{m_0\} = \mathbb{E}_T\left\{\mathrm{Var}_{\mathcal{D}}\left\{\frac{M_0}{T}\,\middle|\, T\right\}\right\} + \overbrace{\mathrm{Var}_T\{\mathbb{E}_{\mathcal{D}}\{m_0 \mid T\}\}}^{=\mathrm{Var}_T\{\rho\}=0} \tag{3.92}$$

$$= \mathbb{E}_T\left\{\frac{\mathrm{Var}_{\mathcal{D}}\{M_0 \mid T\}}{T^2}\right\} = \mathbb{E}_T\left\{\frac{T\,\rho\,(1-\rho)}{T^2}\right\} \tag{3.93}$$

$$= \mathbb{E}_T\left\{\frac{1}{T}\right\}\rho\,(1-\rho). \tag{3.94}$$

Unglücklicherweise kann der Erwartungswert $\mathbb{E}\{\frac{1}{T}\}$ im Allgemeinen nicht bestimmt werden. Mit Hilfe der Jensen'schen Ungleichung, also $\mathbb{E}\{f(X)\} \geq f\left(\mathbb{E}\{X\}\right)$ für konvexe f, ist allerdings

$$\frac{1}{\mathbb{E}_{\mathcal{D}}\{T\}}\,\rho\,(1-\rho) \leq \mathrm{Var}_{\mathcal{D}}\{m_0\} \qquad (\text{da } \tfrac{1}{T}\text{konvex für } T>0) \tag{3.95}$$

eine untere Schranke der Varianz. Der Erwartungswert $\mathbb{E}_{\mathcal{D}}\{T\}$ ist die mittlere Größe bekannter Objekte in Pixeln und kann entweder als Vorwissen parametriert oder aus der Lernstichprobe geschätzt werden.
Insgesamt kann der Schwellwert τ durch den Hilfsparameter z also durch

$$\tau := \rho + z\sqrt{\frac{\rho(1-\rho)}{\mathbb{E}\{T\}}} \tag{3.96}$$

festgelegt werden. Diese Gleichung macht auch den bereits beobachteten Zusammenhang der optimalen Wahl (in Bezug auf Klassifikationsleistung) von ρ und τ explizit. Anders als der Parameter τ ist der Parameter z von der Wahl von ρ unabhängig. Mit $\rho = 0{,}14$ und $z = 3$ wird mit diesem Verfahren in den obigen Experimenten für Datensatz C-2 ein annähernd optimaler Schwellwert $\tau = 0{,}19$ gewählt. Für Datensatz A-3 wird mit $\tau = 0{,}17$ die optimale Schwelle allerdings deutlich unterschätzt.

Signifikanzniveau. Eine weitere Rückweisungsoption kann durch einen Perspektivwechsel bei der Interpretation von $P_{e1} = P(m_0 > \tau \,|\, \overline{\omega_0})$ entwickelt werden. Dazu wird nicht m_0, sondern τ als Veränderliche angesehen: Für ein Objekt o mit $0 \leq q \leq T$ Ausreißern ist $P(M_0 > q \,|\, \overline{\omega_0})$ die Wahrscheinlichkeit, ein bekanntes Objekt mit mehr als q Ausreißern anzutreffen. Umgekehrt quantifiziert diese Wahrscheinlichkeit das Vertrauen in die Nullhypothese $o \in \overline{\omega_0}$. Ein Objekt sollte zurückgewiesen werden, wenn dieses Vertrauen ein Signifikanzniveau α unterschreitet, also

$$\hat{\omega}(\mathcal{O}) = \begin{cases} \omega_0 & \text{wenn } P(M_0 > q \,|\, \overline{\omega_0}) \leq \alpha \\ \overline{\omega_0} & \text{sonst.} \end{cases} \tag{3.97}$$

Da M_0 binomialverteilt ist, ist die benötigte Wahrscheinlichkeit durch die kumulierte Wahrscheinlichkeitsfunktion gegeben,

$$P(M_0 > q \,|\, \overline{\omega_0}) = 1 - P(M_0 \leq q \,|\, \overline{\omega_0}) \tag{3.98}$$

$$= 1 - \sum_{j=0}^{q} \binom{T}{j} \rho^j \, (1-\rho)^{T-j}. \tag{3.99}$$

Die Auswertung dieses Zusammenhangs ist allerdings mit verhältnismäßig hohem Rechenaufwand verbunden. Zur Verringerung der benötigten Rechenzeit kann für ein gegebenes Signifikanzniveau α aber auch ein Schwellwert $\tau = \frac{q}{T}$ mit

$$P(M_0 > q \,|\, \overline{\omega_0}) \approx \alpha \tag{3.100}$$

analog zu Gleichung (3.79) numerisch anhand der Lernstichprobe bestimmt werden. In den obigen Experimenten würden so für $\rho = 0{,}14$ und $\alpha = 0{,}01$ mit $\tau = 0{,}18$ für Datensatz C-2 und $\tau = 0{,}17$ für Datensatz A-3 ähnliche Schwellwerte wie bei Verwendung des z-score festgelegt.

Wie beim z-score ist der Parameter α von der Wahl von ρ unabhängig. Vielmehr geht ρ direkt in die Definition von α ein. Anders als der Parameter z repräsentiert α hier aber eine Wahrscheinlichkeit und ist damit für Anwender ohne tiefergehende Kenntnis über Statistik besser interpretierbar.

3.4.8 Zusammenfassung

Die Hauptbeiträge der hier vorgestellten Methode sind zweierlei:

1. Durch dichtes Abtasten und primitive Deskriptoren wird das Bag of Visual Words Verfahren für die Schüttgutsortierung einsatzfähig. Die resultierenden Objektdeskriptoren können je nach Wahl der primitiven Deskriptoren Farbe, Textur und (mit Einschränkungen) die Form des Objekts beschreiben. Zudem sind die Objektdeskriptoren gegenüber Rotation und Objektgröße invariant[2]. Diese beiden Eigenschaften sind für die Schüttgutsortierung von großer Bedeutung.

2. Die Erweiterung um eine Detektion anomaler primitiver Deskriptoren entrauscht die Objektdeskriptoren und erlaubt eine Rückweisung unbekannter Objekte. Die Analyse der zugrunde liegenden Größen resultierte in zwei Verfahren, diese Rückweisungsoption für Anwender interpretierbar zu parametrieren. Solche Verfahren sind für die visuelle Inspektion von großer Bedeutung, da in der Praxis grundsätzlich von der Klassifikation in der offenen Welt ausgegangen werden muss.

In umfangreichen Versuchen wurde die Methode mit realen Datensätzen evaluiert, die unterschiedliche Sortierprobleme aus der Lebensmittelinspektion und Mineralsortierung repräsentieren. Es wurden drei Kodierungsverfahren – harte und weiche Zuweisung und die Fisher-Vektorkodierung – verglichen. Die komplexere FV-Kodierung ist einfachen Häufigkeitsstatistiken hier nur geringfügig überlegen. Die Kombination aus Farb- und Texturmerkmalen hat nur in zwei Fällen die Klassifikationsleistung verbessert, in den restlichen

[2] Es sei denn, die Objektrotation oder -größe wäre in den primitiven Deskriptoren kodiert.

sechs Experimenten war kein signifikanter Vorteil gegenüber ausschließlicher Farbklassifikation nachzuweisen. Im Gegensatz zum typischen Einsatz von BOW reichten zur Beschreibung der Objekte in den Experimenten bereits kleine Vokabulare von deutlich unter 100 visuellen Worten. Auch mit schwierigen Datensätzen wurde bereits mit einer kleinen Zahl an Trainingsbeispielen eine hohe Klassifikationsgüte erreicht.
Die Güte der Rückweisungsoption ist von den Erscheinungsbildern bekannter und unbekannter Objekte abhängig. Sind diese ähnlich, sinkt die Klassifikationsgüte. Trotzdem wurde mit der hier vorgestellten Methode ein besseres Ergebnis erzielt als sowohl die Klassifikation ohne Rückweisung als auch ein vergleichbarer Ansatz aus der Literatur.
Die für die Experimente verwendete Implementierung ist noch zu langsam für den Einsatz in Sortiermaschinen. Neben naheliegenden softwaretechnischen Optimierungen und der teilweisen Auslagerung in die Hardware könnte das Verfahren durch zufälliges statt dichtes Abtasten der Vordergrundpixel beschleunigt werden. Dabei muss allerdings untersucht werden, wie sich diese Unterabtastung auf die Klassifikationsleistung und insbesondere die Rückweisungsoption auswirkt. Eine weitere Möglichkeit zur Reduzierung der Rechenzeit ist die Reduzierung des visuellen Vokabulars durch Verschmelzung ähnlicher oder redundanter visueller Worte.
Neben der Klassifikation in der offenen Welt könnte die Rückweisungsoption auch genutzt werden, um während der Sortierung einen Drift der Daten, z. B. zwischen oder sogar während Ernteperioden, zu erkennen. Dieser Drift könnte korrigiert werden, indem die Vokabulare im Betrieb mit neuen Daten angepasst werden, ohne das ganze Vokabular neu zu bestimmen. Solche Verfahren könnten auch mit Ansätzen des Active Learning (siehe z. B. Röder [Röd13]) kombiniert werden, um die Anzahl der dazu benötigten Annotationen durch die Anwender gering zu halten.

3.5 Einklassenklassifikation mit Gauß-Mixturbäumen

In dieser Arbeit wurden Verfahren zur semi-automatischen Optimierung der Bilderfassung, zur Selektion relevanter Merkmale und zum Lernen von Merkmalen anhand einer Lernstichprobe behandelt. Diese Verfahren decken also zwei der drei für diese Arbeit relevanten Teilaufgaben der Mustererkennungskette aus Abbildung 3.1 ab. Die dritte Aufgabe – Klassifikation – wurde dabei durch Standardverfahren, insbesondere die SVM übernommen oder direkt in das Verfahren eingebaut.
In den Experimenten waren für das Training und die Auswertung der Methoden jeweils Datensätze mit mehreren hundert Beispielen jeder Klasse vorhanden. Der Zeitaufwand bei der Erstellung solcher Datensätze kann durch geeignete Werkzeuge erleichtert werden, ist aber auch dann ein zeitaufwendiger Prozess. Zudem ist die Erstellung geeigneter Datensätze mit hohem Aufwand verbunden. Die Lernstichprobe sollte das Sortierproblem gut repräsentieren, doch dies ist in der Praxis oft schwierig, da die Defektklasse deutlich seltener auftritt als die Positivklasse. Die Lernstichprobe kann, wie in den Experimenten, durch eine klassenabhängige Gewichtung oder (äquivalent) durch Verkleinerung oder Resampling balanciert werden, jedoch sind die Defektklassen auch dann oft nicht ausreichend repräsentativ, insbesondere wenn die Klasse mehrere Defekttypen vertritt und nur wenige Beispiele in der Lernstichprobe umfasst.
Außerdem werden die Defektbeispiele in der Praxis oft so gewählt, dass die Defekte gut erkennbar und damit einfach zu klassifizieren sind. Mit dem Ziel eines leistungsfähigen Klassifikators sollte die Stichprobe aber mehr schwer zu klassifizierende Beispiele enthalten, da diese näher an den Entscheidungsgrenzen liegen und somit relevanter für die Erstellung des Klassifikators sind als die einfachen Beispiele. Die SVM wird sogar ausschließlich durch die schwierig zu klassifizierenden Beispiele der Stichprobe definiert.

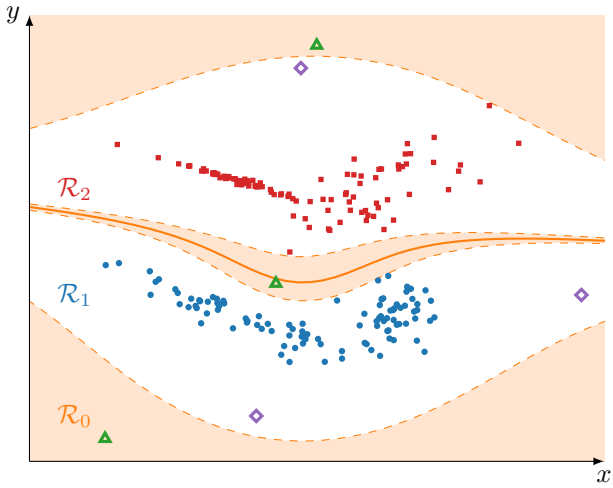


Abbildung 3.22: Klassifikation mit Rückweisung durch erweiterte logistische Regression mit $h(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{x} + b$. Die Merkmale sind alle Monome von x und y von Grad ≤ 3, d. h. $\boldsymbol{x} = (1, x, y, x^2, xy, y^2, x^3, x^2y, xy^2, y^3)^\top$. Rückweisung, wenn $|h(\boldsymbol{x})| < \varepsilon$ oder $|h(\boldsymbol{x})| > \tau$. Die schattierte Fläche zeigt die Rückweisungsregion $\mathcal{R}_0$. Dreiecke zeigen korrekt zurückgewiesene Beispiele, Rauten zeigen inkorrekt nicht zurückgewiesene Merkmale in Regionen geringer Dichte.

Doch selbst wenn die Stichprobe balanciert ist und ausreichend relevante Defektbeispiele enthält, ist sie in den seltensten Fällen repräsentativ für alle Defekttypen, die während des Betriebs auftauchen können. Insbesondere wenn die Fertigungskette durch einen Wechsel der Rohstoffquelle oder durch Einbindung einer neuen Maschine verändert wird, kann sich die Defektklasse im Laufe der Zeit verändern. Diese Defekte sind dem Klassifikator unbekannt und sollten als solche erkannt und entsprechend gesondert behandelt werden. Es muss also in der offenen Welt klassifiziert werden. In Abschnitt 2.1 wurde bereits auf Verfahren verwiesen, mit denen beliebige Klassifikatoren um eine Rückweisungsoption ergänzt werden können [HW06; YW10; BRN17].

Solche Methoden weisen mit diskriminativen Modellen wie der logistischen Regression Beispiele zurück, wenn die Klassenzuweisung zu unsicher ist, die Merkmale $\boldsymbol{x}$ also zu nah an der Entscheidungsgrenze liegen, oder wenn die Entscheidung zu sicher ist, d. h. wenn die $\boldsymbol{x}$ zu weit von der Entscheidungsgrenze entfernt sind (Dreiecke in Abbildung 3.22). Merkmale in dünn besiedelten Regionen des Merkmalsraums, aber im richtigen Abstand zur Entscheidungsgrenze, werden nicht zurückgewiesen (Rauten in Abbildung 3.22). Für die Rückweisung dieser Merkmale wäre Information über die Dichten $p(\boldsymbol{x} \mid \omega)$ nötig, die bei diskriminativen Modellen aber nicht vorhanden ist.
Andere Ansätze integrieren die Rückweisungsoption in den Klassifikator, indem Informationen über die Dichte mit betrachtet werden. Die lokale Dichte ist dabei aber nicht notwendigerweise explizit kodiert. Bei der prototypbasierten Klassifikation von Dubuisson und Masson [DM93] und Fischer et al. [Fis+14] wird die Dichte beispielsweise implizit berücksichtigt, indem ein Objekt zurückgewiesen wird, wenn der Abstand zum nächsten Prototypen zu groß ist. In der Arbeit von Bayer [Bay16] werden Entscheidungsbäume um eine Rückweisungsoption erweitert, indem in jedem inneren Knoten zusätzlich zum Test $h_k(\boldsymbol{x})$ eine Rückweisungsregion assoziiert wird. Diese Rückweisungsregion kann einerseits über Konfidenzbereiche der betrachteten Merkmale oder andererseits über die Distanz zum empirischen Mittel der Stichprobe definiert werden.
Da die Defektklasse in der Schüttgutsortierung aber ohnehin oft nicht repräsentativ ist, stellt sich die Frage, ob sich der Klassifikator nicht anhand von lediglich Positivbeispielen definieren ließe. Defekte Objekte würden dann statt einer Defektklasse der Unbekanntklasse ω_0 zugeordnet. Die in diesem Abschnitt vorgestellten Gauß-Mixturbäume (GMBs) sind ein Verfahren für eine solche Einklassenklassifikation. GMBs kombinieren Entscheidungsbäume und GMMs, um einerseits eine beliebig genaue Schätzung der Dichte zu erhalten und andererseits eine schnelle Rückweisung unbekannter Objekte zu ermöglichen.

3.5.1 Stand der Technik

Diese Einklassenklassifikation entspricht formal der Unterscheidung der Klassen ω_0 und $\overline{\omega_0}$, wobei die Lernstichprobe lediglich Beispiele der Klasse $\overline{\omega_0}$ enthält. Wie bei der normalen Klassifikation kann wieder zwischen diskriminativen und generativen Ansätzen unterschieden werden.
Diskriminative Methoden nutzen (explizit oder implizit) die a-posteriori Wahrscheinlichkeit $P(\overline{\omega_0} \mid \boldsymbol{x})$, also die Wahrscheinlichkeit, dass $\boldsymbol{x}$ nicht anomal ist, zur Definition der Rückweisungsregion $\mathcal{R}_0$. Die a-posteriori Wahrscheinlichkeit wird dabei nicht immer explizit verwendet. Mit SVM-Methoden wie Tax und Duins Support Vector Data Description (SVDD) [TD99] oder die One Class SVM (ocSVM) von Schölkopf et al. [Sch+01] wird statt der a-posteriori Verteilung die Rückweisungsgrenze $P(\omega_0 \mid \boldsymbol{x}) = \alpha$ direkt angegeben. SVDD modelliert diese Grenze als Hyperkugel im induzierten Merkmalsraum Φ. Die Hyperkugel ist so konstruiert, dass sie die transformierten Merkmale $\phi(\boldsymbol{x}_n)$ für $\boldsymbol{x}_n \in \mathcal{D}$ mit kleinstem Radius umschließt. Bei der ocSVM wird hingegen eine Hyperebene in Φ konstruiert, die die $\phi(\boldsymbol{x}_n)$ mit maximalem Rand vom Ursprung von Φ trennt. Obwohl die beiden Ansätze verschiedene Annahmen über die Lage der anomalen Daten machen – SVDD vermutet diese außerhalb der Hyperkugel, ocSVM nahe des Ursprungs von Φ – führen beide Ansätze zu ähnlichen Entscheidungsgrenzen im originalen Merkmalsraum $\mathbb{M}$. Im Falle der Verwendung von RBF-Kerneln sind die beiden Verfahren sogar äquivalent [Sch+01]. Ein Überblick über weitere SVM-Verfahren findet sich in der Arbeit von Khan und Madden [KM14].
Diskriminative Ansätze sind aber nicht auf SVM-Methoden beschränkt. Desir et al. verwenden beispielsweise Random Forests zur Einklassenklassifikation [Dés+13]. Da Random Forests Stichproben von mindestens zwei Klassen benötigen, werden die Beispiele der Hintergrundklasse ω_0 unter Ausnutzung von Wissen über die Verteilung der Lernstichprobe mit Beispielen der Klasse $\overline{\omega_0}$, sowie durch das Bagging und die zufällige Unterraumprojektion im Lernalgorithmus der Random Forests zufällig generiert.

Generative Methoden schätzen hingegen die Dichte $p(\boldsymbol{x} \mid \overline{\omega_0})$ der Lernstichprobe direkt. Ein Objekt wird zurückgewiesen, wenn dessen Merkmale in eine dünn besiedelte Region des Merkmalsraums fallen, also

$$\hat{\omega} = \omega_0 \quad \Leftrightarrow \quad p(\boldsymbol{x} \mid \overline{\omega_0}) < \kappa. \tag{3.101}$$

Im Kontext der MAP-Klassifikation entspricht dieses Vorgehen der Annahme, dass die unbekannten Merkmale gleichverteilt sind, dass also

$$p(\boldsymbol{x} \mid \omega_0) = \text{const} > 0. \tag{3.102}$$

Wie bei der MAP-Klassifikation wird die Dichte $p(\boldsymbol{x} \mid \overline{\omega_0})$ mittels nichtparametrischen Verfahren wie der Parzen-Fenstermethode [Par62] oder durch parametrische Ansätze geschätzt. Mit einem GMM lassen sich sogar beliebige Dichtefunktionen beliebig genau approximieren [Maz96].
Die Güte der Dichteschätzung hängt dabei aber sehr stark von der vorgegebenen Anzahl von GMM-Komponenten ab. Die Anzahl dieser Komponenten kann durch den Bayes'schen Ansatz von Görür und Rasmussen [GR10] automatisch geschätzt werden, jedoch werden dafür neue Hyperparameter eingeführt, die die Anzahl der Komponenten implizit steuern. Mit hierarchischen Gauß-Mischmodellen kann die Anzahl der Komponenten nach dem Training festgelegt werden, indem nur ein Querschnitt der Hierarchie betrachtet wird. Ein solches hierarchisches GMM kann beispielsweise mit der Methode von Vasconcelos und Lippman [VL98] erstellt werden. In Anlehnung an die agglomerative Clusteranalyse werden hierzu, ausgehend von einer Komponente pro Beispiel in der Lernstichprobe, ähnliche Verteilungen sukzessive miteinander verschmolzen, bis nur noch eine Komponente vorhanden ist. Die Parameter der Komponenten einer Schicht ergeben sich dabei direkt aus den Parametern der darunterliegenden Schicht und müssen nicht aus den Lerndaten geschätzt werden. Williams [Wil99] konstruiert ebenfalls ein hierarchisches Gauß-Mischmodell, allerdings werden die Parameter dieses

Modells – ein Strukturvektor und eine Konnektivitätsmatrix – mittels eines Markov-Chain-Monte-Carlo-Verfahrens festgelegt.
Neben diesen Ansätzen gibt es auch andere hierarchische Dichteschätzer, die sich nicht auf GMMs stützen. Ram und Gray [RG11] schätzen die Dichte wie Breimans Regression Trees [Bre+84] durch eine stückweise konstante Funktion. Diese Density Estimation Trees sind zwar einfach zu trainieren und können schnell ausgewertet werden, jedoch wird die Form der Dichteschätzung durch die Struktur der Entscheidungsbäume stark eingeschränkt, was sich wiederum in einer relativ schwachen Klassifikationsleistung äußert [RG11].
Die größten Gemeinsamkeiten der im Folgenden beschriebenen Gauß-Mixturbäume finden sich in den bereits erwähnten Ansätzen von Bayer [Bay16] sowie in den Density Forests von Criminisi [Cri11]. Density Forests sind aus mehreren Entscheidungsbäumen zusammengesetzt, die wie klassische Entscheidungsbäume rekursiv aus Daten konstruiert werden. Die Tests der inneren Knoten werden dabei so gewählt, dass sie den Informationszuwachs in Bezug auf die Lerndaten maximieren. Unter der Annahme normalverteilter Merkmale entspricht dieses Ziel der Minimierung des Volumens der Ellipsoiden, die die Lernstichproben der Kindknoten umschließen [Cri11]. Die Blattknoten des Entscheidungsbaums werden schließlich mit multivariaten Normalverteilungen assoziiert. Da der Baum den Merkmalsraum in disjunkte Regionen partitioniert, setzt sich die Dichteschätzung also aus mehreren abgeschnittenen Normalverteilungen zusammen. Die resultierende Partitionierungsfunktion ist allerdings sehr kompliziert und im Allgemeinen nicht analytisch bestimmbar. Dieser Nachteil kann durch die Verwendung von achsenparallelen Schnitten in den inneren Knoten (siehe Gleichung 2.88 auf S. 88) ausgeglichen werden [Cri11], allerdings wird die Form der Dichteschätzung dadurch wie bei Ram und Gray [RG11] erheblich eingeschränkt.

3.5.2 Beitrag zum Stand der Technik

Die Dichteschätzung mit Gauß-Mixturbäumen unterliegt solchen Einschränkungen nicht. GMBs sind hierarchische Gauß-Mischmodelle, wobei jede Schicht eines GMBs selbst als GMM aufgefasst werden kann. Tiefere Schichten beinhalten dabei mehr Komponenten und bilden damit mehr Details ab als frühe Schichten. Die hierarchische Struktur eines GMBs kann außerdem ausgenutzt werden, um den Rechenaufwand der Dichteschätzung erheblich zu reduzieren und um eine schnelle Einklassenklassifikation abzuleiten. Insbesondere die Rückweisung von sicher unbekannten Objekten beansprucht dabei nur wenige CPU-Zyklen. Die Algorithmen zum Training und zur Auswertung eines GMBs sind konzeptionell einfach und leicht zu implementieren. Die hier vorgestellten Methoden werden anhand synthetischer und realer Datensätze mit Einklassenklassifikation durch GMMs und ocSVMs verglichen. Das Verfahren wurde in Richter et al. [RLB17] veröffentlicht.

3.5.3 Gauß-Mixturbäume

Gauß-Mixturbäume vereinen die Dichteschätzung durch GMMs mit der hierarchischen Struktur und Konzepten des Trainings von Entscheidungsbäumen. Ein GMM beschreibt die Dichte der assoziierten Zufallsvariable als eine Konvexkombination von K Gauß-Dichten,

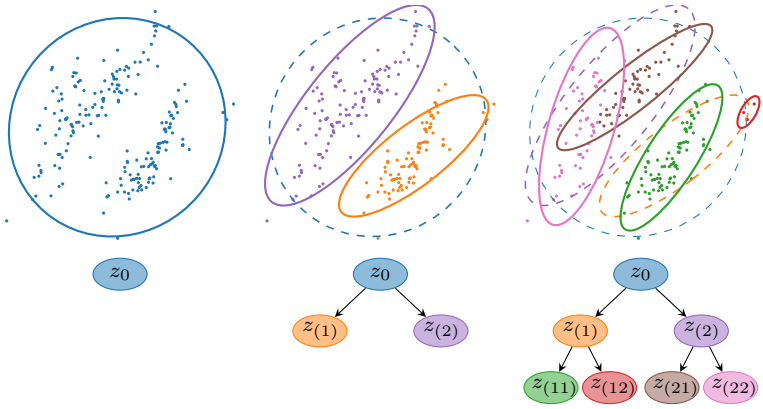
$$p(\boldsymbol{x}) = \sum_{k=1}^{K} P(z_k)\ p(\boldsymbol{x} \mid z_k) \quad \text{mit} \quad p(\boldsymbol{x} \mid z_k) = \mathcal{N}\left(\boldsymbol{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right). \tag{3.103}$$

Die Parameter des GMMs – $P(z_k)$, $\boldsymbol{\mu}_k$ und $\boldsymbol{\Sigma}_k$ mit $k = 1, \ldots, K$ – werden üblicherweise durch Expectation Maximization (EM) [DLR77] bestimmt, was einer Maximum-Likelihood-Schätzung anhand des Datensatzes $\mathcal{D}$ entspricht. Ein GMM kann auch als generatives Modell aufgefasst werden. Die Indikatorvariable z_k gibt dabei an, ob ein Merkmalsvektor $\boldsymbol{x}$ von der

dazugehörigen k-ten Komponente generiert wurde. In diesem Sinne wird z_k auch als Bezeichner dieser Komponente verwendet.
Die Dichteschätzung in Gleichung (3.103) kann für die Einklassenklassifikation verwendet werden, indem die Schätzung wie oben beschrieben mit einem Schwellwert verglichen wird. Allerdings müssen dazu alle K Komponenten ausgewertet werden, auch wenn nur wenige Komponenten zur Dichte $p(\boldsymbol{x})$ an $\boldsymbol{x}$ beitragen. Mit Density Estimation Trees von Criminisi [Cri11] ist dies nicht der Fall. Hier wird nur die für $\boldsymbol{x}$ „zuständige" Komponente ausgewertet. Allerdings ist hier die Dichteschätzung aufgrund der komplizierten Partitionierungsfunktion verhältnismäßig rechenaufwendig.
GMBs befinden sich zwischen diesen beiden Extremen: Wie ein Density Estimation Tree kann ein GMB als Entscheidungsbaum zur Dichteschätzung angesehen werden. Im Unterschied zu einem Density Estimation Tree ist in einem GMB allerdings jeder Knoten, also auch innere Knoten, mit einer Normalverteilung und einer a-priori Wahrscheinlichkeit assoziiert, die die Dichte mit verschiedenen Detailstufen repräsentieren. Abbildung 3.23 zeigt drei GMBs mit null (d. h. nur der Wurzelknoten z_0), einer und zwei Schichten, sowie die mit den Knoten assoziierten Normalverteilungen.
Diese Baumstruktur kann unendlich fortgesetzt werden, wobei jeder Knoten $z_{(k_1 \cdots k_d)}$ mit einer Normalverteilung $\mathcal{N}\left(\boldsymbol{x} \,\middle|\, \boldsymbol{\mu}_{z_{(k_1 \cdots k_d)}}, \boldsymbol{\Sigma}_{z_{(k_1 \cdots k_d)}}\right)$ und einer a-priori Wahrscheinlichkeit $P\left(z_{(k_1 \cdots k_d)}\right)$ assoziiert ist. Letztere beschreibt die Wahrscheinlichkeit, dass der Knoten an der Generierung eines Merkmalsvektors $\boldsymbol{x}$ beteiligt ist und ersteres beschreibt die Dichte $p\left(\boldsymbol{x} \,\middle|\, z_{(k_1 \cdots k_d)}\right)$ der Merkmale dieses Knotens. Die $K_{(k_1 \cdots k_d)}$ Kindknoten von $z_{(k_1 \cdots k_d)}$,

$$\mathcal{C}\left(z_{(k_1 \cdots k_d)}\right) := \left\{ z_{(k_1 \cdots k_d k_{d+1})} \,\middle|\, k_{d+1} = 1, \ldots, K_{(k_1 \cdots k_d)} \right\}, \tag{3.104}$$

beschreiben die Dichte detaillierter als der Knoten $z_{(k_1 \cdots k_d)}$. Ausgehend von der groben Dichteschätzung im Wurzelknoten z_0 durch eine einzige Normalverteilung wird die Schätzung also durch jede zusätzliche Schicht $\mathcal{S}_d$ verfeinert.



(a) GMB der Tiefe 0. **(b)** GMB der Tiefe 1. **(c)** GMB der Tiefe 2

Abbildung 3.23: GMBs zunehmender Tiefe beschreiben die zugrunde liegenden Daten mit zunehmend detaillierteren Dichtefunktionen. Die Ellipsen über den Bäumen zeigen eine Standardabweichung der mit den Knoten assoziierten Normalverteilungen.

Zur Vereinfachung der Notation bezeichne $z_{\boldsymbol{k}(d)}$ einen Knoten $z_{(k_1 \cdots k_d)} \in \mathcal{S}_d$ der Tiefe d und $z_{\overrightarrow{\boldsymbol{k}(d)}} := (z_{\boldsymbol{k}(d)}, z_{\boldsymbol{k}(d-1)}, \ldots, z_0)$ den Pfad vom Wurzelknoten zu diesem Knoten. Somit steht $P\left(z_{\overrightarrow{\boldsymbol{k}(d)}}\right) = P\left(z_{\boldsymbol{k}(d)}, z_{\boldsymbol{k}(d-1)}, \ldots, z_0\right)$ für die Wahrscheinlichkeit, den Knoten $z_{\boldsymbol{k}(d)}$ bei der Generierung eines Merkmals $\boldsymbol{x}$ zu besuchen. Die Dichteschätzung der Schicht $\mathcal{S}_d$ ergibt sich durch Marginalisierung über alle Knoten der Schicht,

$$\begin{aligned} p(\boldsymbol{x}) \approx p(\boldsymbol{x} \,|\, \mathcal{S}_d) &= \sum_{z_{\boldsymbol{k}(d)} \in \mathcal{S}_d} p\left(\boldsymbol{x}, z_{\overrightarrow{\boldsymbol{k}(d)}}\right) \\ &= \sum_{z_{\boldsymbol{k}(d)} \in \mathcal{S}_d} p\left(\boldsymbol{x} \,\middle|\, z_{\overrightarrow{\boldsymbol{k}(d)}}\right) P\left(z_{\overrightarrow{\boldsymbol{k}(d)}}\right). \end{aligned} \tag{3.105}$$

Dabei kann $P\left(z_{\overrightarrow{\boldsymbol{k}(d)}}\right)$ als die Wahrscheinlichkeit verstanden werden, dass der Knoten $z_{\boldsymbol{k}(d)}$ an der Generierung des Merkmals $\boldsymbol{x}$ beteiligt ist. Damit

jede Schicht eine gültige Wahrscheinlichkeitsdichte $p(\boldsymbol{x} \mid \mathcal{S}_d)$ repräsentiert, müssen drei Bedingungen erfüllt sein:

1. Bei der Generierung von $\boldsymbol{x}$ wird ein Knoten jeder Schicht besucht,

$$\sum_{z_{\boldsymbol{k}(d)} \in \mathcal{S}_d} P\left(z_{\overrightarrow{\boldsymbol{k}(d)}}\right) = 1. \tag{3.106}$$

2. Die Wahrscheinlichkeit, einen Knoten zu besuchen, verteilt sich vollständig auf seine Kindknoten,

$$\begin{aligned} P\left(z_{\overrightarrow{\boldsymbol{k}(d-1)}}\right) &= \sum_{z_{\boldsymbol{k}(d)} \in \mathcal{C}\left(z_{\boldsymbol{k}(d-1)}\right)} P\left(z_{\boldsymbol{k}(d)}, z_{\overrightarrow{\boldsymbol{k}(d-1)}}\right) \\ &= \sum_{z_{\boldsymbol{k}(d)} \in \mathcal{C}\left(z_{\boldsymbol{k}(d-1)}\right)} P\left(z_{\overrightarrow{\boldsymbol{k}(d)}}\right). \end{aligned} \tag{3.107}$$

3. Die Wahrscheinlichkeitsmasse der Dichte eines Knotens verteilt sich vollständig auf seine Kindknoten,

$$\int_{\mathbb{M}} p\left(\boldsymbol{x}, z_{\overrightarrow{\boldsymbol{k}(d)}}\right) \mathrm{d}\boldsymbol{x} = \sum_{z \in \mathcal{C}\left(z_{\boldsymbol{k}(d)}\right)} \int_{\mathbb{M}} p\left(\boldsymbol{x}, z, z_{\overrightarrow{\boldsymbol{k}(d)}}\right) \mathrm{d}\boldsymbol{x} \tag{3.108}$$

Ein GMB ist ein endlicher Teilbaum dieser unendlich genauen Beschreibung der Dichte. Neben dem Wurzelknoten enthält dieser Teilbaum entweder alle oder keine Kinder eines gegebenen Knotens $z_{\boldsymbol{k}(d)}$. Alternativ betrachtet ist ein GMB eine endliche, gestutzte Annäherung des unendlichen Baums, die durch Ersetzen beliebiger Teilbäume durch die Wurzelknoten dieser Teilbäume entsteht. Insbesondere enthält ein GMB immer den Wurzelknoten z_0. Abbildung 3.24 skizziert den Zusammenhang zwischen dieser unendlichen hierarchischen Dichteschätzung mit Schichten S_d und einem möglichen GMB dieser Hierarchie.

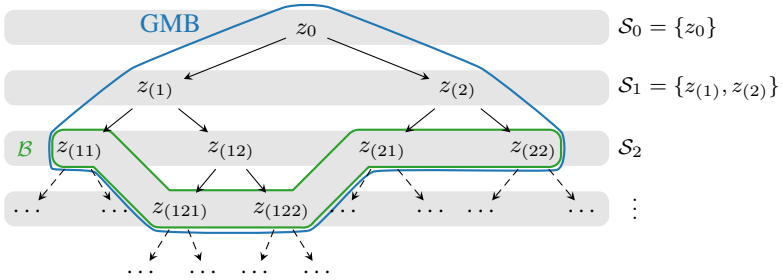


Abbildung 3.24: Ein GMB als endlicher Teilbaum eines unendlichen Baums, der die Dichte $p(\boldsymbol{x})$ mit beliebiger Genauigkeit repräsentiert.

Da sich die Blattknoten $\mathcal{B}$ eines GMB über mehrere Schichten $\mathcal{S}_d$ erstrecken können, gilt Gleichung (3.105) im Allgemeinen nicht. Wegen den Bedingungen aus den Gleichungen (3.106) bis (3.108) kann die Dichte $p(\boldsymbol{x})$ an einem Punkt $\boldsymbol{x}$ aber durch Marginalisierung der Verbunddichte $p\left(\boldsymbol{x}, z_{\overrightarrow{\boldsymbol{k}(d)}}\right)$ über alle Pfade zu den Blattknoten $z_{\boldsymbol{k}(d)} \in \mathcal{B}$ des GMB angenähert werden,

$$p(\boldsymbol{x}) = \sum_{z_{\boldsymbol{k}(d)} \in \mathcal{B}} p\left(\boldsymbol{x}, z_{\overrightarrow{\boldsymbol{k}(d)}}\right) = \sum_{z_{\boldsymbol{k}(d)} \in \mathcal{B}} P\left(z_{\overrightarrow{\boldsymbol{k}(d)}}\right) p\left(\boldsymbol{x} \,\middle|\, z_{\overrightarrow{\boldsymbol{k}(d)}}\right). \quad (3.109)$$

Ein GMB mit B Blattknoten entspricht also einem GMM mit B Komponenten. Mit diesen Definitionen kann auch der generative Prozess eines GMBs beschrieben werden: In der ersten Schicht wird zufällig ein Knoten $z_{\boldsymbol{k}(1)} \in \mathcal{C}\left(z_0\right)$ mit Wahrscheinlichkeit $P\left(z_{\boldsymbol{k}(1)}\right)$ gewählt. In der zweiten Schicht wird $z_{\boldsymbol{k}(2)} \in \mathcal{C}\left(z_{\boldsymbol{k}(1)}\right)$ mit Wahrscheinlichkeit

$$P\left(z_{\boldsymbol{k}(2)} \,\middle|\, z_{\boldsymbol{k}(1)}\right) = \frac{P\left(z_{\boldsymbol{k}(2)}, z_{\boldsymbol{k}(1)}\right)}{P\left(z_{\boldsymbol{k}(1)}\right)} = \frac{P\left(z_{\overrightarrow{\boldsymbol{k}(2)}}\right)}{P\left(z_{\boldsymbol{k}(1)}\right)} \quad (3.110)$$

gewählt. Dieser Prozess wird fortgesetzt, bis ein Blattknoten $z_{\boldsymbol{k}(d)} \in \mathcal{B}$ erreicht ist. Der Merkmalsvektor $\boldsymbol{x}$ wird schließlich anhand der mit $z_{\boldsymbol{k}(d)}$ assoziierten Normalverteilung $p\left(\boldsymbol{x} \,\middle|\, z_{\overrightarrow{\boldsymbol{k}(d)}}\right)$ generiert.

GMB Training. Das Lernverfahren eines GMBs orientiert sich am Entscheidungsbaumtraining (siehe Algorithmus 2, S. 89). Die Struktur des Baumes, die Parameter der Normalverteilungen und die a-priori Wahrscheinlichkeiten der Knoten werden rekursiv anhand einer Lernstichprobe $\mathcal{D} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$ festgelegt. Die Parameter $\boldsymbol{\mu}_0$ und $\boldsymbol{\Sigma}_0$ des Wurzelknotens werden mittels Maximum-Likelihood geschätzt, also

$$\boldsymbol{\mu}_0 := \hat{\boldsymbol{\mu}}_{\text{ML}}(\mathcal{D}) = \frac{1}{N} \sum_{n=1}^{N} \boldsymbol{x}_n \quad \text{und} \tag{3.111}$$

$$\boldsymbol{\Sigma}_0 := \hat{\boldsymbol{\Sigma}}_{\text{ML}}(\mathcal{D}) = \frac{1}{N} \sum_{n=1}^{N} (\boldsymbol{x}_n - \boldsymbol{\mu}_0)(\boldsymbol{x}_n - \boldsymbol{\mu}_0)^\top. \tag{3.112}$$

Für einen gegebenen Knoten $z_{\boldsymbol{k}(d)}$ werden die Parameter der Kindknoten $z \in \mathcal{C}\left(z_{\boldsymbol{k}(d)}\right)$ über ein GMM mit $K = \left|\mathcal{C}\left(z_{\boldsymbol{k}(d)}\right)\right|$ Komponenten festgelegt. Training des GMMs geschieht mit EM oder dem Ansatz von Görür und Rasmussen [GR10].
Damit die Bedingung aus Gleichung (3.107) erfüllt wird, müssen die durch das GMM ermittelten a-priori Wahrscheinlichkeiten $P\left(z_{\boldsymbol{k}(d)} \,\middle|\, z_{\overrightarrow{\boldsymbol{k}(d-1)}}\right)$ der Knoten $z_{\boldsymbol{k}(d)} \in \mathcal{C}\left(z_{\boldsymbol{k}(d-1)}\right)$ mit den a-priori Wahrscheinlichkeiten der Elternknoten gewichtet werden, d. h.

$$P\left(z_{\overrightarrow{\boldsymbol{k}(d)}}\right) := P\left(z_{\overrightarrow{\boldsymbol{k}(d-1)}}\right) \cdot P\left(z_{\boldsymbol{k}(d)} \,\middle|\, z_{\overrightarrow{\boldsymbol{k}(d-1)}}\right). \tag{3.113}$$

Es lässt sich leicht nachprüfen, dass mit diesem Vorgehen die anderen Bedingung aus Gleichungen (3.106) und (3.108) ebenfalls erfüllt werden.
Damit sich die Knoten der zweiten Schicht bzw. die mit den Knoten assoziierten Dichteschätzungen unterscheiden, wird das GMM für einen gegebenen

Algorithmus 5 Gauß-Mixturbaum Training.

function LEARN(node, $p, \mathcal{D}, d$)
  if $d \leq d_{\max}$ **and** $|\mathcal{D}| \geq N_{\min}$ **then**
    for $(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k, P(z_k)) \in \texttt{GMM}(\mathcal{D}, K)$ **do**
      $\texttt{c} \leftarrow \texttt{Node}\left(p \cdot P(z_k), \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right)$
      $\mathcal{D}' \leftarrow \{\boldsymbol{x} \in \mathcal{D} \,|\, P(z_k \,|\, \boldsymbol{x}) \geq \varphi\}$
      $\texttt{node.child}_k \leftarrow \text{LEARN}\left(\texttt{c}, p \cdot P(z_k), \mathcal{D}', d+1\right)$
  return node

function LEARNTREE($\mathcal{D}$)
  return LEARN $\left(\texttt{Node}\left(\hat{\boldsymbol{\mu}}_{\text{ML}}(\mathcal{D}), \hat{\boldsymbol{\Sigma}}_{\text{ML}}(\mathcal{D}), 1\right), 1, \mathcal{D}, 1\right)$

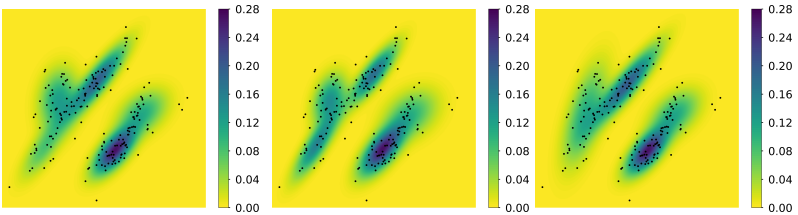
Knoten $z_{\boldsymbol{k}(2)} \in \mathcal{C}\left(z_{\boldsymbol{k}(1)}\right)$ nur anhand der Trainingsdaten geschätzt, die wahrscheinlich durch die Elternkomponente $z_{\boldsymbol{k}(1)}$ erzeugt wurden, d. h.

$$\mathcal{D}(z_{\boldsymbol{k}(1)}) := \left\{\boldsymbol{x} \in \mathcal{D} \,\middle|\, z_{\boldsymbol{k}(1)} = \underset{z \in \mathcal{C}(z_0)}{\arg\max}\, P(z \,|\, \boldsymbol{x})\right\}. \tag{3.114}$$

Diese Prozedur wird rekursiv fortgesetzt, wobei für die Kindknoten $z_{\boldsymbol{k}(d+1)} \in \mathcal{C}\left(z_{\boldsymbol{k}(d)}\right)$ nur die Trainingsdaten $\mathcal{D}(z_{\boldsymbol{k}(d)})$ verwendet werden, die wahrscheinlich von $z_{\boldsymbol{k}(d)}$ generiert wurden.

Dieses Vorgehen entspricht der Aufteilung der Lernstichprobe während des Entscheidungsbaumtrainings, wobei die Testfunktion eines inneren Knotens einer MAP-Klassifikation entspricht. Im Gegensatz zum Entscheidungsbaumtraining wird durch die Aufteilung allerdings nicht die Klassenreinheit, sondern die Likelihoodfunktion der Dichteschätzung maximiert.

Wie beim Entscheidungsbaumtraining wird die Rekursion abgebrochen, wenn eine maximale Baumtiefe $d_{\max}$ erreicht ist oder weniger als $N_{\min}$ Trainingsbeispiele zur Schätzung des GMMs zur Verfügung stehen. Die zweite Bedingung kann auch bei großen Lernstichproben schnell erreicht werden. Um zu flachen Gauß-Mixturbäumen entgegenzuwirken, kann die Aufteilung der Lernstichprobe aufgeweicht werden, sodass $\mathcal{D}(z_{\boldsymbol{k}(d)})$ alle



(a) Ground Truth **(b)** GMM mit $K = 5$ **(c)** GMB mit $d_{\max} = 2$

Abbildung 3.25: Vergleich der Dichteschätzung mit GMMs und GMBs. Die Ground Truth Verteilung zur Generierung der Lernstichprobe entspricht einem GMM mit $K = 5$ Komponenten.

Beispiele enthält, die mit einer Wahrscheinlichkeit von mindestens φ von $z_{\boldsymbol{k}(d)}$ generiert wurden,

$$\mathcal{D}(z_{\boldsymbol{k}(d)}) := \left\{ \boldsymbol{x} \in \mathcal{D}(z_{\boldsymbol{k}(d-1)}) \;\middle|\; P\left(z_{\boldsymbol{k}(d)} \;\middle|\; \boldsymbol{x}, z_{\overrightarrow{\boldsymbol{k}(d-1)}}\right) \geq \varphi \right\}. \quad (3.115)$$

Mit dieser aufgeweichten Aufteilung der Lernstichprobe ist ein Merkmalsvektor $\boldsymbol{x}$ in der Regel in mehreren $\mathcal{D}(z_{\boldsymbol{k}(d)})$ enthalten.
Der Pseudocode dieses Lernalgorithmus ist in Algorithmus 5 zu sehen. Ein Vergleich der Dichteschätzung anhand eines GMMs mit $K = 5$ Komponenten und einem GMB der Tiefe $d_{\max} = 2$ (d. h. 4 Komponenten) sowie die zur Generierung der Lernstichprobe verwendete Ground Truth (GMM mit $K = 5$ Komponenten) ist in Abbildung 3.25 zu sehen.

3.5.4 Untere Schranke der Dichteschätzung

Für die Dichteschätzung nach Gleichung (3.109) müssen alle mit den Blattknoten des Baumes assoziierten Dichten ausgewertet werden, also auch die, die wenig zur marginalisierten Dichte $p(\boldsymbol{x})$ an $\boldsymbol{x}$ beitragen. Der Rechenaufwand entspricht also dem Rechenaufwand der Dichteschätzung mit einem äquivalenten GMM.

Die Baumstruktur des GMB kann allerdings ausgenutzt werden, um wenig relevante Teilbäume früh von der Berechnung auszuschließen. In der Regel unterschätzt die daraus resultierende Approximation die wahre Dichte. Da sich diese Approximation nicht zu 1 integriert, ist sie keine echte Dichte, sondern stellt vielmehr eine untere Schranke von $p(\boldsymbol{x})$ dar. Mit dem Ziel der Einklassenklassifikation ist diese untere Schranke allerdings ausreichend, da die Dichteschätzung hier ohnehin nur zum Vergleich mit einem Schwellwert genutzt wird (siehe Gleichung 3.101).
Um eine Formel für diese Schranke herzuleiten, wird zunächst ein beliebiger Knoten $z_{\boldsymbol{k}(d)}$ betrachtet. Da ein gegebenes von $z_{\boldsymbol{k}(d)}$ generiertes Merkmal $\boldsymbol{x}$ ebenfalls von einem der Kindknoten $z \in \mathcal{C}\left(z_{\boldsymbol{k}(d)}\right)$ generiert werden kann, gilt mit $\mathcal{C}_d := \mathcal{C}\left(z_{\boldsymbol{k}(d)}\right)$

$$1 = \sum_{z \in \mathcal{C}_d} P\left(z \,\middle|\, \boldsymbol{x}, z_{\overrightarrow{\boldsymbol{k}(d)}}\right) \tag{3.116}$$

$$\Leftrightarrow \quad p\left(\boldsymbol{x}, z_{\overrightarrow{\boldsymbol{k}(d)}}\right) = \sum_{z \in \mathcal{C}_d} P\left(z \,\middle|\, \boldsymbol{x}, z_{\overrightarrow{\boldsymbol{k}(d)}}\right) p\left(\boldsymbol{x}, z_{\overrightarrow{\boldsymbol{k}(d)}}\right) \tag{3.117}$$

$$= \sum_{z \in \mathcal{C}_d} P\left(z \,\middle|\, \boldsymbol{x}, z_{\overrightarrow{\boldsymbol{k}(d)}}\right) \sum_{z' \in \mathcal{C}_d} p\left(\boldsymbol{x}, z', z_{\overrightarrow{\boldsymbol{k}(d)}}\right), \tag{3.118}$$

wobei in der letzten Zeile erneut über alle Kinder von $z_{\boldsymbol{k}(d)}$ marginalisiert wurde.
Diese Gleichung kann genutzt werden, um die Dichte $p(\boldsymbol{x}) = p\left(\boldsymbol{x}, z_{\overrightarrow{\boldsymbol{k}(0)}}\right)$ beginnend am Wurzelknoten z_0 rekursiv zu berechnen. Die Rekursion endet beim Erreichen eines Blattknotens, also wieder mit $\mathcal{C}_d := \mathcal{C}\left(z_{\boldsymbol{k}(d)}\right)$

$$p\left(\boldsymbol{x}, z_{\overrightarrow{\boldsymbol{k}(d)}}\right) = \begin{cases} \sum_{z,z' \in \mathcal{C}_d} P\left(z \,\middle|\, \boldsymbol{x}, z_{\overrightarrow{\boldsymbol{k}(d)}}\right) p\left(\boldsymbol{x}, z', z_{\overrightarrow{\boldsymbol{k}(d)}}\right) & \text{für } z_{\boldsymbol{k}(d)} \notin \mathcal{B} \\ p\left(\boldsymbol{x} \,\middle|\, z_{\overrightarrow{\boldsymbol{k}(d)}}\right) P\left(z_{\overrightarrow{\boldsymbol{k}(d)}}\right) & \text{sonst.} \end{cases} \tag{3.119}$$

Die a-posteriori Wahrscheinlichkeit $P\left(z \,\middle|\, \boldsymbol{x}, z_{\overrightarrow{\boldsymbol{k}(d)}}\right)$, also die Wahrscheinlichkeit, dass $\boldsymbol{x}$ von dem Kindknoten $z \in \mathcal{C}\left(z_{\boldsymbol{k}(d)}\right)$ generiert wurde, ist durch das mit $z_{\boldsymbol{k}(d)}$ assoziierte GMM (Zeile 3 des Algorithmus 5) gegeben.
Die Rekursionsgleichung (3.119) liefert die gleiche Dichteschätzung wie Gleichung (3.109). Da hier zusätzlich zu den Blattknoten aber auch alle inneren Knoten des Baumes berücksichtigt werden, bedeutet diese Dichteschätzung einen höheren Rechenaufwand als Gleichung (3.109). Allerdings zeigt die Rekursionsgleichung auch eine Möglichkeit zur schnellen Approximation der Dichte auf: Für fast alle $\boldsymbol{x} \in \mathbb{M}$ ist die maximale a-posteriori Wahrscheinlichkeit $\max_z P\left(z \,\middle|\, \boldsymbol{x}, z_{\overrightarrow{\boldsymbol{k}(d)}}\right) \approx 1$. In diesen Regionen ist die Dichte $p\left(\boldsymbol{x} \,\middle|\, z', z_{\overrightarrow{\boldsymbol{k}(d)}}\right)$ der anderen Kindknoten z von $z_{\boldsymbol{k}(d)}$ vernachlässigbar klein und damit

$$P\left(z \,\middle|\, \boldsymbol{x}, z_{\overrightarrow{\boldsymbol{k}(d)}}\right) \approx 1 \quad \Rightarrow \quad p\left(\boldsymbol{x}, z_{\overrightarrow{\boldsymbol{k}(d)}}\right) \approx p\left(\boldsymbol{x}, z, z_{\overrightarrow{\boldsymbol{k}(d)}}\right). \tag{3.120}$$

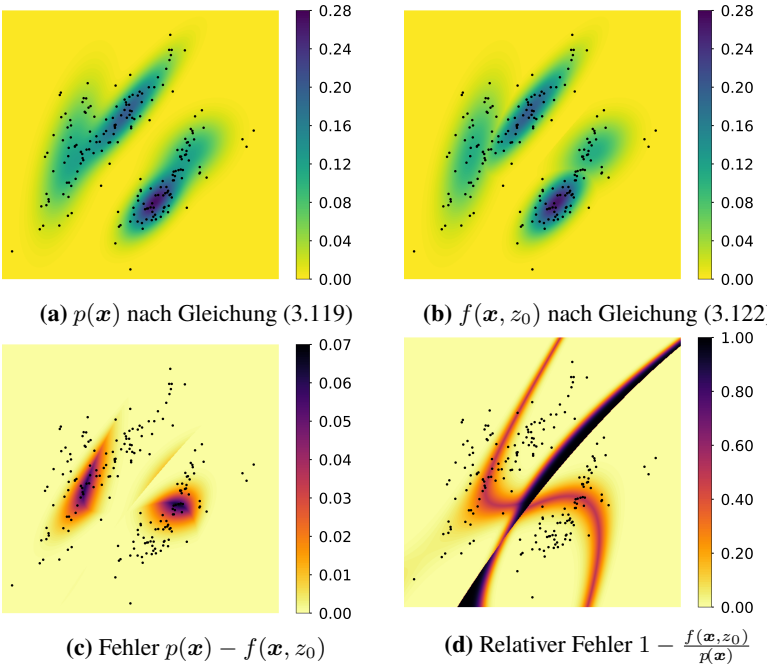
In diesen Fällen muss also – wie beim generativen Prozess des GMB – jeweils nur der Pfad durch den Baum betrachtet werden, durch den $\boldsymbol{x}$ mit hoher Wahrscheinlichkeit generiert wurde. Wenn Approximationsfehler in Regionen mit $P\left(z \,\middle|\, \boldsymbol{x}, z_{\overrightarrow{\boldsymbol{k}(d)}}\right) \not\approx 1$ hingenommen werden, kann die Rekursionsgleichung (3.119) durch

$$f(\boldsymbol{x}, z_{\boldsymbol{k}(d)}) = \begin{cases} f(\boldsymbol{x}, z^*) & \text{für } z_{\boldsymbol{k}(p)} \notin \mathcal{B} \\ p\left(\boldsymbol{x} \,\middle|\, z_{\overrightarrow{\boldsymbol{k}(d)}}\right)\, P\left(z_{\overrightarrow{\boldsymbol{k}(d)}}\right) & \text{sonst} \end{cases} \tag{3.121}$$

$$\text{mit} \quad z^* = \operatorname*{arg\,max}_{z \in \mathcal{C}\left(z_{\boldsymbol{k}(d)}\right)} P\left(z \,\middle|\, \boldsymbol{x}, z_{\overrightarrow{\boldsymbol{k}(d)}}\right) \tag{3.122}$$

angenähert werden. Da hier nur der Beitrag des Pfades mit der maximalen a-posteriori Wahrscheinlichkeit der Generierung von $\boldsymbol{x}$ beachtet wird, ist

$$f(\boldsymbol{x}, z_0) = \max_{z_{\boldsymbol{k}(d)} \in \mathcal{B}} p\left(\boldsymbol{x}, z_{\overrightarrow{\boldsymbol{k}(d)}}\right), \tag{3.123}$$



(a) $p(\boldsymbol{x})$ nach Gleichung (3.119)

(b) $f(\boldsymbol{x}, z_0)$ nach Gleichung (3.122)

(c) Fehler $p(\boldsymbol{x}) - f(\boldsymbol{x}, z_0)$

(d) Relativer Fehler $1 - \frac{f(\boldsymbol{x}, z_0)}{p(\boldsymbol{x})}$

Abbildung 3.26: Dichteschätzung und untere Schranke der Dichte mit einem binären GMB der Tiefe 2. Der Approximationsfehler ist in Regionen zwischen zwei benachbarten Komponenten am höchsten.

woraus unmittelbar folgt, dass

$$f(\boldsymbol{x}, z_0) \leq p\left(\boldsymbol{x}, z_{\overrightarrow{\boldsymbol{k}(0)}}\right) \approx p(\boldsymbol{x})\,. \tag{3.124}$$

Mit der erwarteten Baumtiefe $\mathbb{E}\{d\}$ und der erwarteten Anzahl von Kindknoten $\mathbb{E}\{K\}$ der inneren Knoten werden hier im Mittel also statt $\mathbb{E}\{K\}^{\mathbb{E}\{d\}}$ lediglich $\mathbb{E}\{K\} \cdot \mathbb{E}\{d\}$ Knoten ausgewertet. Insbesondere bei tiefen, vollständigen Bäumen ist die rekursive Approximation also erheblich schneller berechenbar als die genaue Dichteschätzung nach Gleichung (3.119). Diese

Approximation entspricht, bis auf eine Skalierung, der Dichteschätzung mit Criminisis Density Forests [Cri11].
Abbildungen 3.26a und 3.26b zeigen die Dichteschätzung eines vollständigen binären GMBs der Tiefe $d_{\max} = 2$ und die schnelle Approximation mit Gleichung (3.122). Es ist deutlich erkennbar, dass die Dichte in Regionen unterschätzt wird, die zwischen den Mittelwerten zweier Normalverteilungen von Knoten aus der gleichen Ebene des Baums liegen (Abbildung 3.26d). Der Approximationsfehler wird größer, je mehr sich die Normalverteilungen der Blätter überlappen (Abbildung 3.26c). In dünn besetzten Regionen des Merkmalsraums ist dieser Fehler aber gering. Da bei Einklassenklassifikation Merkmale aus diesen Bereichen zurückgewiesen werden sollen, kann der Approximationsfehler in den dicht besiedelten Regionen vernachlässigt werden.
Da die Knoten hier mit multivariaten Normalverteilungen assoziiert sind, lässt sich die Berechnung von $f(\boldsymbol{x}, z_0)$, bzw. des wahrscheinlich für die Generierung von $\boldsymbol{x}$ verantwortlichen Pfades, durch Umformung weiter beschleunigen. Mit $\mathcal{C}_{d-1} := \mathcal{C}\left(z_{\boldsymbol{k}(d-1)}\right)$ ist

$$z^* = \underset{z_{\boldsymbol{k}(d)} \in \mathcal{C}_{d-1}}{\arg\max} \; P\left(z_{\boldsymbol{k}(d)} \,\middle|\, \boldsymbol{x}, z_{\overrightarrow{\boldsymbol{k}(d-1)}}\right) \tag{3.125}$$

$$= \underset{z_{\boldsymbol{k}(d)} \in \mathcal{C}_{d-1}}{\arg\max} \; \log P\left(z_{\boldsymbol{k}(d)} \,\middle|\, \boldsymbol{x}, z_{\overrightarrow{\boldsymbol{k}(d-1)}}\right) \tag{3.126}$$

$$= \underset{z_{\boldsymbol{k}(d)} \in \mathcal{C}_{d-1}}{\arg\min} \Big\{ \left(\boldsymbol{x} - \boldsymbol{\mu}_{z_{\overrightarrow{\boldsymbol{k}(d)}}}\right)^\top \boldsymbol{\Sigma}^{-1}_{z_{\overrightarrow{\boldsymbol{k}(d)}}} \left(\boldsymbol{x} - \boldsymbol{\mu}_{z_{\overrightarrow{\boldsymbol{k}(d)}}}\right) + \underbrace{\log |\boldsymbol{\Sigma}_{z_{\overrightarrow{\boldsymbol{k}(d)}}}| - \log \left(P\left(z_{\boldsymbol{k}(d)} \,\middle|\, z_{\overrightarrow{\boldsymbol{k}(d-1)}}\right)\right)^2}_{=:\gamma(z_{\boldsymbol{k}(d)})} \Big\}. \tag{3.127}$$

Der Term $\gamma(z_{\boldsymbol{k}(d)})$ hängt nicht von $\boldsymbol{x}$ ab und kann somit in der Lernphase vorberechnet werden. Dadurch reduziert sich die Suche nach z^* auf die Auswertung schnell berechenbarer quadrierter Mahalanobis-Distanzen.

3.5.5 Einklassenklassifikation mit früher Rückweisung

Da GMBs Dichteschätzer sind, ist eine Einklassenklassifikation durch die eingangs beschriebene Schwellwertbildung

$$\hat{\omega}(\boldsymbol{x}) := \begin{cases} \omega_0 & \text{wenn } p(\boldsymbol{x}) < \kappa \\ \overline{\omega_0} & \text{sonst} \end{cases} \tag{3.128}$$

möglich. Der Schwellwert κ kann hier, wie bei den BOW-Deskriptoren aus Abschnitt 3.4.5, durch einen Hilfsparameter $\rho \in [0, 1]$ mit

$$\mathbb{E}_{\mathcal{D}} \left\{ \mathbb{1}[p(\boldsymbol{x}) < \kappa] \right\} \approx \rho \tag{3.129}$$

festgelegt werden. Mit einer repräsentativen Stichprobe steuert dieser Parameter also den erwarteten Fehler erster Ordnung $P(\hat{\omega}(\boldsymbol{x}) = \omega_0 \,|\, \overline{\omega_0}) \approx \rho$. Die Methode ist zwar einfach zu implementieren und mit der unteren Schranke aus dem vorherigen Abschnitt auch schnell berechenbar. Die hierarchische Struktur eines GMB kann aber noch weiter ausgenutzt werden, um sicher unbekannte Objekte mit $p(\boldsymbol{x}) \approx 0$ ohne vollständige Berechnung von $p(\boldsymbol{x})$ zurückzuweisen. Dazu wird jeder Knoten $z_{\boldsymbol{k}(d)}$ zusätzlich mit einem Rückweisungsparameter $\kappa_{\boldsymbol{k}(d)}$ assoziiert. Ein Objekt wird zurückgewiesen, wenn $p\left(\boldsymbol{x} \,\middle|\, z_{\overrightarrow{\boldsymbol{k}(d)}}\right) < \kappa_{\boldsymbol{k}(d)}$. Da bei der Approximation mit Gleichung (3.122) ohnehin die mit den inneren Knoten verknüpften Dichten ausgewertet werden müssen, entsteht hier kein zusätzlicher Rechenaufwand. Insgesamt ergibt sich der Klassifikator

$$\hat{\omega}_{z_{\boldsymbol{k}(d)}}(\boldsymbol{x}) = \begin{cases} \omega_0 & \text{wenn } p\left(\boldsymbol{x}, z_{\overrightarrow{\boldsymbol{k}(d)}}\right) < \kappa_{\boldsymbol{k}(d)} \\ \overline{\omega_0} & \text{wenn } p\left(\boldsymbol{x}, z_{\overrightarrow{\boldsymbol{k}(d)}}\right) \geq \kappa_{\boldsymbol{k}(d)} \text{ und } z_{\boldsymbol{k}(d)} \in \mathcal{B} \\ \hat{\omega}_{z^*}(\boldsymbol{x}) & \text{sonst} \end{cases} \tag{3.130}$$

mit z^* wie in Gleichung (3.122).

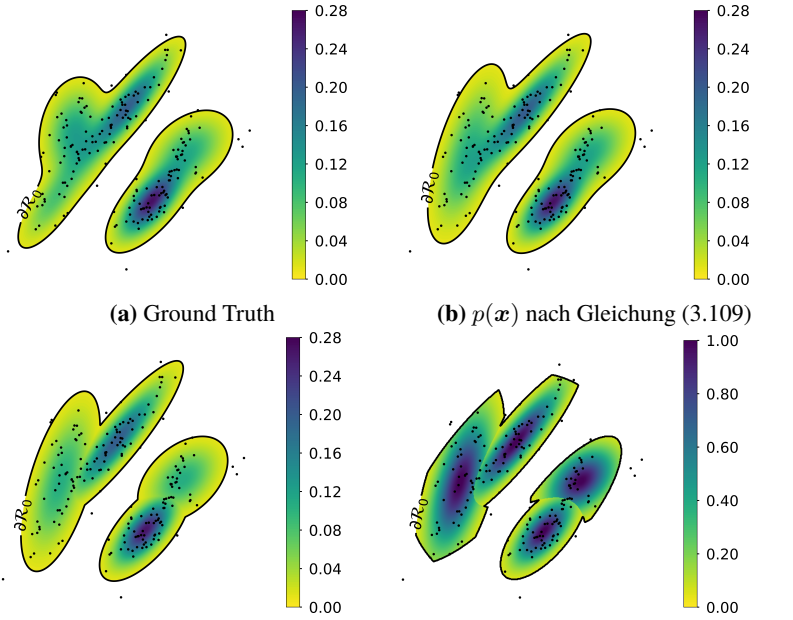
Wie bei der Berechnung von $f(\boldsymbol{x}, z_0)$ kann hier außerdem ausgenutzt werden, dass die Knoten mit Normalverteilungen assoziiert sind. Statt anhand der Verbunddichte $p\left(\boldsymbol{x}, z_{\overrightarrow{\boldsymbol{k}(d)}}\right)$ zurückzuweisen, kann also auch hier die quadrierte Mahalanobis-Distanz verwendet werden. Die erste Bedingung aus Gleichung (3.130) kann damit durch

$$\left(\boldsymbol{x} - \boldsymbol{\mu}_{z_{\boldsymbol{k}(d)}}\right)^\top \boldsymbol{\Sigma}^{-1}_{z_{\boldsymbol{k}(d)}} \left(\boldsymbol{x} - \mu_{z_{\boldsymbol{k}(d)}}\right) > \delta_{z_{\boldsymbol{k}(d)}} \tag{3.131}$$

ersetzt werden, wobei $\boldsymbol{\mu}_{z_{\boldsymbol{k}(d)}}$ und $\boldsymbol{\Sigma}_{z_{\boldsymbol{k}(d)}}$ die Parameter der Gauß-Dichte am Knoten $z_{\boldsymbol{k}(d)}$ bezeichnen.
Die Rückweisungsparameter $\delta_{z_{\boldsymbol{k}(d)}}$ können wiederum durch statistische Tests festgelegt werden: Die quadrierte Mahalanobis-Distanz folgt einer χ^2-Verteilung mit $\dim \mathbb{M} = D$ Freiheitsgraden. Da die Verteilungsfunktion dieser Verteilung bekannt ist, kann die Rückweisungsschwelle wie in Abschnitt 3.4.7 anhand eines Signifikanzniveaus α festgelegt werden. Sollen im Mittel nicht mehr als α der Beispiele zurückgewiesen werden, wird $\delta_{z_{\boldsymbol{k}(d)}}$ so gewählt, dass

$$P\left(d_{\Sigma_{z_{\boldsymbol{k}(d)}}}(\boldsymbol{x}, \mu_{z_{\boldsymbol{k}(d)}}) \leq \delta_{z_{\boldsymbol{k}(d)}}\right) = 1 - \alpha. \tag{3.132}$$

Da hier wieder lediglich Matrixmultiplikationen durchgeführt werden, ist diese Rückweisungsoption sehr schnell berechenbar. Die weitaus größere Reduzierung der Rechenzeit wird aber durch die frühe Rückweisung unbekannter Objekte erzielt. Insbesondere Objekte, deren Merkmalsvektoren weit von den Beispielen entfernt liegen, können bereits am Wurzelknoten oder in der ersten Schicht zurückgewiesen werden. Allerdings weicht die Form der Rückweisungsgrenze von den Rückweisungsgrenzen unter Verwendung von $p(\boldsymbol{x})$ nach Gleichung (3.109) und der unteren Schranke $f(\boldsymbol{x}, z_0)$ nach Gleichung (3.122) ab. Abbildung 3.27 zeigt exemplarisch die Rückweisungsgrenzen mit der vollständigen Dichteschätzung (Abbildung 3.27b), mit der unteren Schranke (Abbildung 3.27c) und mit der frühen Rückweisung (Ab-



(a) Ground Truth

(b) $p(\boldsymbol{x})$ nach Gleichung (3.109)

(c) $f(\boldsymbol{x}, z_0)$ nach Gleichung (3.122)

(d) Rückweisung nach Gleichung (3.130)

Abbildung 3.27: Vergleich der Rückweisungsregion mit Dichteschätzung, unterer Schranke der Dichte, sowie mit schneller Rückweisung. Die Entscheidungsgrenze liegt bei $P(\overline{\omega_0} \mid \boldsymbol{x}) = 0{,}05$.

bildung 3.27d) im Vergleich mit der optimalen Rückweisungsgrenze, die von der Ground Truth Verteilung abgeleitet wurde (Abbildung 3.27a).

3.5.6 Experimente und Ergebnisse

Der GMB-Ansatz wurde mit Einklassenklassifikation mittels eines GMMs und mittels einer ocSVM mit RBF-Kernel verglichen. Dafür wurden synthetische und reale Daten verwendet.

In allen Experimenten wurden binäre GMBs mit $\varphi = 0{,}5$ und einer maximalen Tiefe von $d_{\max} = 3$ verwendet. Bei dichtebasierter Rückweisung wurden die Schwellwerte anhand der Akzeptanzrate $\rho = 0{,}01$ ermittelt. Mit der

schnellen Rückweisung wurden die Parameter $\delta_{\boldsymbol{k}(d)}$ mit $\alpha = 0{,}001$ nach Gleichung (3.132) festgelegt.
Der Rückweisungsparameter κ der GMM-basierten Einklassenklassifikation wurde wie beim GMB mit $\rho = 0{,}01$ berechnet. Die ocSVM wurde mit einem RBF-Kernel der Breite $\sigma = \frac{1}{D}$ und dem Lernparameter $\nu = 0{,}1$ trainiert. Der Lernparameter ν ist dabei sowohl eine obere Schranke des Lernfehlers, als auch eine untere Schranke des Anteils an Support Vektoren in $\mathcal{D}$ [Sch+01].
Die Implementierung des GMMs und der ocSVM wurden der erprobten scikit-learn Bibliothek von Pedregosa et al. [Ped+11] entnommen. Für die hier vorgestellten GMB-Ansätze wurde die JIT-compilierte Programmiersprache Julia [Bez+17] verwendet. Alle Experimente wurden auf einem handelsüblichen Rechner mit 2,4 GHz Intel i7 CPU und 8 GB RAM durchgeführt.

Synthetische Daten. Die Methoden wurden zunächst mit synthetischen Daten mit bekannten Verteilungen der Klassen $\overline{\omega_0}$ und ω_0 evaluiert. Die Beispiele der Positivklasse $\overline{\omega_0}$ wurden mit einem GMM mit $K = 5$ Komponenten generiert. Die Parameter des Mischmodells wurden dabei zufällig festgelegt:

- Die $\boldsymbol{\mu}_k \sim \mathcal{N}\left(\mathbf{0}, (2\sqrt{D})\boldsymbol{I}\right)$ folgen einer zentrierten Normalverteilung diagonaler Kovarianzmatrix und Varianz $2\sqrt{D}$.
- Die $\boldsymbol{\Sigma}_k \sim \mathfrak{W}_D\left(\boldsymbol{I}, D^2\right)$ folgen einer unskalierten Wishart-Verteilung mit D^2 Freiheitsgraden.
- Die $P(z_k) = \eta_k \sim \mathfrak{Dir}(\mathbf{1})$ folgen einer Dirichlet-Verteilung mit $p(\eta_i) = p(\eta_j)$ für alle $i,j \in \{1, \ldots, K\}$.

Die Beispiele der Hintergrundklasse ω_0 wurden mit Hilfe einer Gleichverteilung über ein Hyperquader in $\mathbb{M}$ generiert. Die Position und Seitenlängen des Hyperquaders wurden so gewählt, dass alle Beispiele der Positivklasse $\overline{\omega_0}$ umschlossen werden.

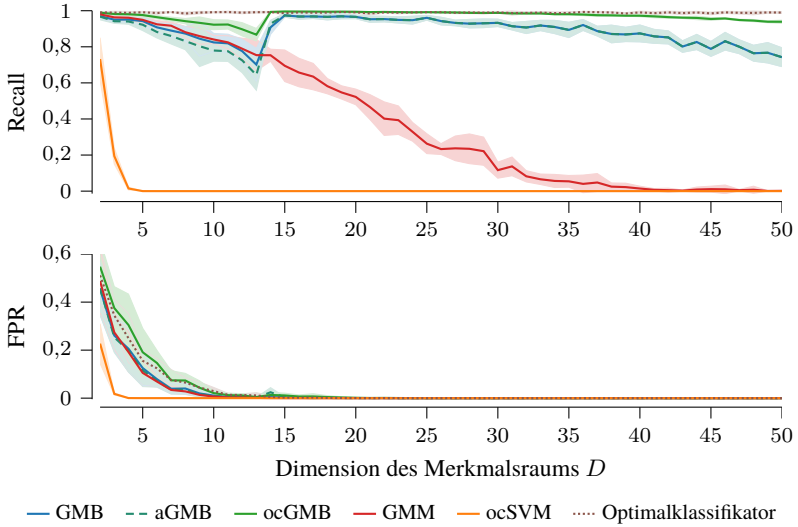


Abbildung 3.28: Richtigpositivrate (Recall) und Falschpositivrate (FPR) der Einklassenklassifikation in Abhängigkeit der Dimension des Merkmalsraums D. aGMB: approximierte Dichte nach Abschnitt 3.5.4. ocGMB: schnelle Rückweisung nach Abschnitt 3.5.5.

Anhand dieser Verteilungen wurden 250 Trainingsbeispiele der Positivklasse $\overline{\omega_0}$ sowie je 100000 Testbeispiele beider Klassen $\overline{\omega_0}$ und ω_0 generiert. Dabei wurde die Dimension des Merkmalsraums von $D = 2$ bis $D = 50$ variiert. Alle Auswertungen wurden je zehn mal mit neuen Verteilungsparametern für $\boldsymbol{x} \,|\, \overline{\omega_0}$ und $\boldsymbol{x} \,|\, \omega_0$ wiederholt.

Abbildung 3.28 zeigt die Richtigpositivrate (Recall) und Falschpositivrate (FPR) der getesteten Klassifikatoren in Abhängigkeit der Dimensionalität des Merkmalsraums. Der Optimalklassifikator ist hier nicht der Bayes'sche Optimalklassifikator nach Gleichung (2.26) aus Abschnitt 2.4, sondern die Rückweisung nach Gleichung (3.101) mit bekannter Dichte $p(\boldsymbol{x})$. Dieser Klassifikator findet fast alle Beispiele der Positivklasse ($\mathrm{recall} \approx 1$), klassifiziert aber bis $D = 10$ noch einige Hintergrundbeispiele fälschlicherweise

als bekannt (FPR > 0). Dieses Ergebnis war zu erwarten, da sich die Verteilungen der beiden Klassen bei kleinen Dimensionen stark überschneiden. Mit wachsender Dimension verteilt sich die Wahrscheinlichkeitsmasse der Hintergrundklasse jedoch auf einen größeren Raum als die Wahrscheinlichkeitsmasse der Positivklasse.
Erstaunlicherweise versagt der ocSVM-Klassifikator in diesem Experiment vollständig. Bereits ab $D = 5$ findet dieser Klassifikator kein einziges Beispiel der Positivklasse $\overline{\omega_0}$. Die Wahl von anderen Hyperparameter ν und σ veränderte die Klassifikationsleistung nicht signifikant. Dieses Ergebnis ist überraschend, da die ocSVM wie einleitend diskutiert der Quasistandard der Einklassenklassifikation ist.
Für die dichtebasierte Einklassenklassifikation wurden GMMs mit $K = 5$ Komponenten, also der gleichen Anzahl an Komponenten wie die gesuchte Ground Truth Verteilung und binäre GMBs mit einer maximalen Tiefe von $d_{\max} = 3$ verwendet. Bis $D = 13$ erzielen beide Methoden vergleichbaren Recall und FPR, wobei die Verwendung der unteren Schranke (aGMB) etwas schlechtere Ergebnisse liefert, als die genaue Dichteschätzung nach Gleichung (3.109). Mit der schnellen Rückweisung (ocGMB) wird ein höherer Recall erreicht, da die Entscheidungsgrenzen hier loser gewählt wurden als bei den dichtebasierten Methoden. Dadurch ist jedoch auch die Falschpositivrate etwas höher als bei den anderen Ansätzen. Der Trade-Off zwischen Recall und FPR kann mit dem Parameter α gesteuert werden. Allerdings werden durch ein kleineres Signifikanzniveau $(1 - \alpha)$ auch mehr bekannte Beispiele zurückgewiesen.
Ab $D = 13$ steigt die Klassifikationsleistung der GMB-Methoden wieder an, während sich die Rückweisung mit GMM weiter verschlechtert. Dieses Verhalten ist durch zwei Ursachen erklärbar. Erstens wächst mit der Dimension des Merkmalsraums auch der erwartete Abstand zwischen den Beispielen beider Klassen. Der Fluch der Dimensionalität hat hier also positive Auswirkungen auf die Klassifikationsgüte. Dies ist auch an der Entwicklung der

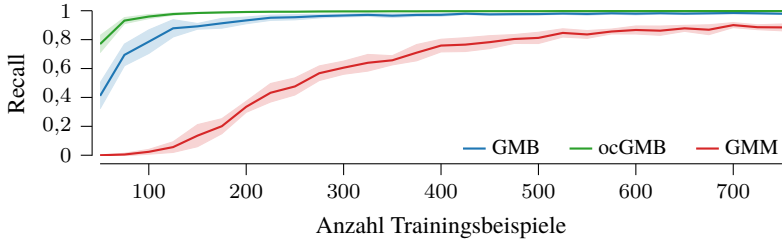


Abbildung 3.29: Recall in Abhängigkeit der Anzahl der Trainingsbeispiele mit $D = 50$.

Falschpositivrate sichtbar: ab $D = 10$ sinkt diese mit allen Methoden auf FPR ≈ 0 ab.

Zweitens müssen die $\left(5 \cdot (D + \frac{1}{2}\, D(D+1)) + 4\right)$ Parameter des GMMs anhand von nur 250 Trainingsbeispielen festgelegt werden. Somit sind bereits ab $D = 9$ mehr Parameter zu schätzen als Trainingsbeispiele zur Verfügung stehen. Beim binären GMB müssen für jeden inneren Knoten nur $\left(2 \cdot (D + \frac{1}{2}\, D(D+1)) + 1\right)$ Parameter bestimmt werden.

Ein GMB kommt also mit weniger Lerndaten aus als ein GMM. Das wird auch in Abbildung 3.29 bestätigt, in der die Anzahl der Trainingsbeispiele aus einem $D = 50$ dimensionalen Merkmalsraum variiert wurde. Dichtebasierte Rückweisung mit einem GMB erreicht bereits ab $N \approx 200$ maximalen Recall. Mit früher Rückweisung werden sogar nur etwa 100 Beispiele benötigt, um einen guten Klassifikator zu erhalten. Dieser Unterschied kann auf die unterschiedlichen Methoden zur Bestimmung der Rückweisungsschwellen zurückgeführt werden: Der Schwellwert κ für die dichtebasierte Rückweisung wird anhand der Lerndaten ermittelt, während die $\delta_{z_{\boldsymbol{k}(d)}}$ der schnellen Rückweisung aufgrund von Konfidenzbereichen der assoziierten Normalverteilungen festgelegt werden. Klassifikation mit einem GMM erreicht erst mit über 700 Trainingsbeispielen einen ähnlich hohen Recall wie die anderen Methoden.

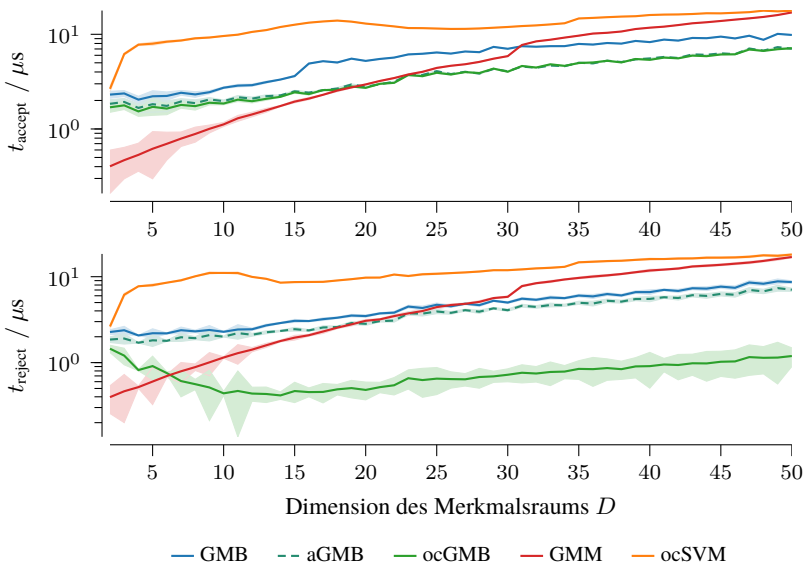


Abbildung 3.30: Benötigte Zeit für die Klassifikation eines Beispiels in Abhängigkeit der Dimension des Merkmalsraums (gemessen, logarithmischer Maßstab auf der Ordinate).

Für den praktischen Einsatz ist neben der Anzahl benötigter Trainingsbeispiele auch die für die Klassifikation benötigte Rechenzeit ausschlaggebend. Abbildung 3.30 zeigt die zur Klassifikation eines Beispiels benötigte Rechenzeit. Die Messungen der GMM- und ocSVM-Klassifikation sind nicht direkt mit den Messungen der GMB-Methoden vergleichbar, da diese mit unterschiedlichen Programmiersprachen[3] implementiert sind.
Qualitativ kann aber beobachtet werden, dass der Rechenaufwand der ocSVM schneller ansteigt als der Rechenaufwand der dichtebasierten Methoden. Eine mögliche Erklärung ist, dass in höheren Dimensionen eine größere Anzahl an Support Vektoren benötigt wird, um die Rückweisungsgrenze zu definieren.

[3] GMM und ocSVM: Cython (compiliert), GMB: Julia (JIT-compiliert).

Für das GMM müssen dagegen nur 5 und für den GMB maximal $2^{d_{\max}} = 8$ Komponenten ausgewertet werden. Hier wird der Rechenaufwand durch die Matrixmultiplikationen dominiert. Durch den Ausschluss von Komponenten mit geringem Einfluss kann die untere Schranke nur um wenige μs schneller berechnet werden als die volle Dichteschätzung mit GMBs. Die schnelle Rückweisung bringt im Falle der Klassifikation zu ω_0 jedoch einen erheblichen Geschwindigkeitszuwachs: Unbekannte Objekte können in weniger als $1\ \mu$s zurückgewiesen werden.
Hier zeigt sich der Fluch der Dimensionalität erneut von seiner positiven Seite: Bis etwa $D = 10$ sinkt die für die Rückweisung benötigte Rechenzeit, da die Wahrscheinlichkeit steigt, dass ein Beispiel bereits in der ersten Schicht zurückgewiesen wird. Ab $D = 10$ steigt die Rechenzeit durch die größere Dimension und damit aufwendigere Matrixmultiplikation wieder an.

Reale Daten. Neben synthetischen Daten mit bekannter Verteilung wurden auch Experimente mit realen Daten unbekannter Verteilung durchgeführt. Hierzu wurde der Lego-Datensatz aus Abschnitt 3.2.7 sowie der Wein-Datensatz A-3 (Schwarzburgunder) aus Abschnitt 3.4.6 verwendet. Beim Lego-Datensatz wurden neun geometrische Merkmale[4] verwendet. Auf Farb- und Texturmerkmale wurde verzichtet, da diese hier nur wenig diskriminative Information tragen. Das Mehrklassenproblem wurde in mehrere Einklassenprobleme umgewandelt, indem die Zielklasse (z. B. 2×2-Steine) von allen anderen Klassen unterschieden werden muss. Beim Datensatz A-3 wurden die um m_0 erweiterten BOW-Deskriptoren mit $\dim(\mathbb{M}) = 31$ aus Abschnitt 3.4 verwendet. Die Klassifikatoren wurden mit der Positivklasse trainiert und mit Positiv- und Defektklasse evaluiert.
In allen Experimenten wurde eine stratifizierte 5-fold Cross Validation durchgeführt. Für die Lego-Steine standen dadurch pro Fold je 377, 433 und 415 Beispiele für die 2×2-, 2×4- und Technik Verbindersteine zur

[4] Durchmesser, Dichte, Fläche, Fläche der konvexen Hülle, Kompaktheit, Ausdehnung, Rundheit, Umfang und Umfang der konvexen Hülle.

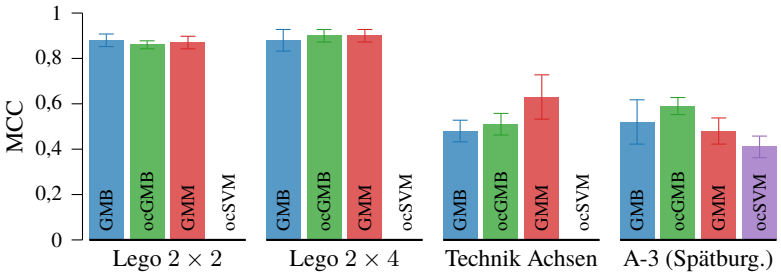


Abbildung 3.31: Klassifikationsgüte der Einklassenklassifikation mit realen Daten. Die Fehlerbalken zeigen die Standardabweichung über eine 5-fold Cross Validation.

Verfügung. Für die Weinbeerensortierung wurden 145 Beispiele pro Fold verwendet.

Der GMM-Klassifikator wurde mit $K = 6$ (Lego) bzw. $K = 8$ (Wein) ermittelt. Alle weiteren Parameter (ν, σ der ocSVM, $d_{\max}, \rho$ bzw. α der GMBs) wurden wie bei den Experimenten mit den synthetischen Daten festgelegt.

Die Klassifikationsgüte ist in Abbildung 3.31 zu sehen. Wie mit den synthetischen Daten schlägt beim Lego-Datensatz die Klassifikation mit einer ocSVM fehl. Die Klassifikation mit GMMs und GMBs zeigt mit Ausnahme der Technik Achsen keine signifikanten Unterschiede. Hier werden mit dem GMM bessere Ergebnisse erzielt als mit dem GMB. Allerdings schwankt die Klassifikationsgüte des GMMs auch deutlich stärker als die Klassifikationsgüte der GMB-Methoden. Insgesamt ist diese Klasse schwer von den anderen Steinen unterscheidbar, da insbesondere die Technik Verbinder (siehe Tabelle 3.2) ähnliche geometrische Merkmale bezüglich Länge und Breite aufweisen.

Beim Datensatz A-3 wird das beste Ergebnis mit einem GMB und schneller Rückweisung erzielt. Die ocSVM erreicht hier eine mit den anderen Methoden vergleichbare Klassifikationsgüte. Allerdings ist die Klassifikationsgüte mit

allen Methoden deutlich schlechter als die erreichbare Güte bei Klassifikation in der geschlossenen Welt. Dies kann durch die Beschaffenheit der BOW-Merkmale begründet werden: Da sich die Einträge der Deskriptoren (bis auf m_0) zu 1 addieren, sind diese auf das D-Einheitssimplex in $\mathbb{M}$ beschränkt und verletzen so die Normalverteilungsannahme des GMMs und der GMBs. Insbesondere ist der Support supp$\{p(\boldsymbol{x})\}$ der Dichteschätzungen nicht auf das Einheitssimplex beschränkt.

3.5.7 Zusammenfassung

In diesem letzten Abschnitt des Kapitels wurden Gauß-Mixturbäume (GMBs) zur effizienten Einklassenklassifikation vorgestellt. Ein GMB ist eine Baumstruktur, in der jeder Knoten mit einer Gauß-Dichte und einer a-priori Wahrscheinlichkeit assoziiert ist. Diese Baumstruktur repräsentiert eine hierarchische Dichteschätzung, wobei mit zunehmender Tiefe mehr Details in den Daten aufgelöst werden. Dabei kann jede Ebene des Baumes als GMM aufgefasst werden. Im Gegensatz zu vergleichbaren Ansätzen der Literatur ist dadurch sowohl die Dichteschätzung als auch der Lernalgorithmus sehr einfach zu implementieren.
GMBs eignen sich neben der Dichteschätzung auch zur effizienten Einklassenklassifikation. Hier kann die hierarchische Struktur des Baums ausgenutzt werden, um nicht relevante Teilbäume von der Berechnung auszuschließen und die Dauer der Berechnung so zu verkürzen. Durch die frühe Rückweisung sicher unbekannter Objekte kann die Klassifikation zusätzlich beschleunigt werden.
Im Vergleich mit Einklassenklassifikation durch GMMs und ocSVMs zeigten GMBs vergleichbare oder bessere Klassifikationsgüte. Dabei wurden zur Ableitung guter Klassifikatoren mit GMBs deutlich weniger Trainingsdaten benötigt als mit GMMs. Zudem war die Klassifikation mit GMBs und hier insbesondere die Rückweisung schneller berechenbar als mit den alternativen

Ansätzen. Beide Eigenschaften machen die Methode ideal für den Einsatz in optischen Sortiersystemen.
Die Experimente zeigten jedoch auch die Grenzen der Methode auf: Wenn die Merkmale die Normalverteilungsannahme der Knoten des GMBs verletzen, wird kein gutes Klassifikationsergebnis erzielt. Dies ist z. B. mit den BOW-Merkmalen aus Abschnitt 3.4 der Fall. In den Experimenten war der GMB Ansatz jedoch sowohl der dichtebasierten Einklassenklassifikation mit GMMs, als auch der Klassifikation mit ocSVM überlegen.

3.5.8 Ausblick

Der GMB-Ansatz kann aber auch noch verbessert werden. Dies betrifft insbesondere den Lernalgorithmus. Hier können die Abbruchsbedingungen um weitere Tests erweitert werden, welche die Normalität der Daten bewerten, d. h. Tests, die bewerten, wie sehr die Annahme einer Normalverteilung auf die Daten zutrifft. Hierzu kann wie von Criminisi [Cri11] vorgeschlagen der Information Gain benutzt werden. Alternativen ergeben sich aus statistischen Tests der Normalität der Daten, z. B. wie von Royston [Roy83] beschrieben. In Anlehnung an das SRM-Prinzip könnte auch abgebrochen werden, wenn der Zuwachs der Likelihood-Funktion des Baumes im Verhältnis zur Komplexität des Klassifikators unter eine Schranke sinkt. Weiterhin kann bei der Schätzung der Kovarianzen ein Regularisierungsterm oder ein Shrinkage-Verfahren (siehe z. B. Schäfer und Strimmer [SS05]) verwendet werden, um auch mit wenigen Trainingsbeispielen hochdimensionaler Daten auszukommen.
Eine naheliegende Idee ist, die GMBs und Ensemblemethoden zu vereinen. Wie bei Random Forests kann hierzu Bagging und zufällige Unterraumprojektion verwendet werden. Erste Experimente in diese Richtung zeigten allerdings keinen Vorteil gegenüber den hier beschriebenen GMBs.
Weiterhin kann der Ansatz auch zu einer Mehrklassenklassifikation mit Rückweisungsoption erweitert werden. Hierzu wird jeder Blattknoten mit einem zusätzlichen Klassifikator assoziiert. In hier nicht gezeigten Experimenten

konnte bereits beobachtet werden, dass sich die Klassen auf verschiedene Teilbäume verteilen. Die Blattknoten wiesen bereits eine hohe Klassenreinheit auf, was wiederum eine anschließende Klassifikation erleichtern würde. Das ist einleuchtend, da sich die Klassenstruktur auch in der Verteilung der Daten im Merkmalsraum widerspiegelt. Allerdings stehen für das Training der Klassifikatoren in den Blattknoten weniger umfangreiche Lernstichproben zur Verfügung als an den inneren Knoten.
Für die visuelle Inspektion und besonders die Schüttgutsortierung ist auch die Verbindung mit Approximate Computing (siehe z. B. Maier et al. [Mai+16]) interessant. Die hierarchische Struktur der GMBs kann hier genutzt werden, um hohe Systemlasten durch geringere Genauigkeit der Dichteschätzung bzw. Klassifikation auszugleichen und so Deadlineverletzungen vorzubeugen.
Schließlich ist die grundlegende Formulierung in Abschnitt 3.5.3 von der Wahl der $p(\boldsymbol{x} \mid z_{\boldsymbol{k}})$ unabhängig. Somit kann der GMB-Ansatz auch auf andere Mischverteilungen ausgeweitet werden. In Verbindung mit BOW-Merkmalen bieten sich insbesondere Dirichlet-Mixturen (siehe Togban und Ziou [TZ17]) an, um die Verteilung der Merkmale zu beschreiben. In diesem Fall ist aber die Beschleunigung durch Verwendung von Mahalanobis-Distanzen nicht mehr gültig.

4 Abschlussbemerkungen

Ziel dieser Arbeit war die Entwicklung und Untersuchung von lernenden Verfahren, die für die optische Inspektion geeignet sind. Da die optische Inspektion sehr viele Teilaufgaben umfasst – Vollständigkeits- und Lageprüfung, Vermessung der Form, Bewertung der Oberfläche, Qualitätsprüfung, Materialidentifikation und Defektdetektion (siehe Beyerer et al. [BPF12]) – wurde sich hier auf die Schüttgutsortierung, also die Klassifizierung von granularen Materialien konzentriert.

Hier können lernende Verfahren genutzt werden, um den Entwurf und die Inbetriebnahme von Sortiersystemen zu beschleunigen bzw. zu vereinfachen. Trotz des enormen Potentials solcher Verfahren gibt es vergleichsweise wenige wissenschaftliche Arbeiten, die sich mit diesem Thema beschäftigen. Der Großteil der Arbeiten konzentriert sich stattdessen auf die Verbesserung der Sensorik oder die Analyse der Materialeigenschaften der zu sortierenden Objekte [WH10; Cub+11; BPL15; BPL17]. Lernende Verfahren beschränken sich oft auf Standardmethoden oder sind auf einen Anwendungsfall zugeschnitten [DS06]. In dieser Arbeit wurden dagegen Verfahren entwickelt, die nicht an ein Produkt gebunden sind, sondern in einer Reihe von Problemdomänen zum Einsatz kommen können. Hierbei wurde sich auf die Bildverarbeitungskette konzentriert. Methoden zur (halb-)automatischen Konfiguration bzw. Auslegung ganzer Sortieranlagen wurden nicht betrachtet.

Die in dieser Arbeit vorgestellten Verfahren unterliegen den Rahmenbedingungen der Schüttgutsortierung: Die Inspektion wird dadurch erleichtert, dass kontrollierte Aufnahmebedingungen bestehen und dass die Erscheinung der zu sortierenden Objekte in der Regel einfach ist. Allerdings unterliegt

die Schüttgutsortierung harten Echtzeitanforderungen, wodurch nur wenig Rechenzeit für die gesamte Bildverarbeitungskette zur Verfügung steht. Für lernende Ansätze kommt erschwerend hinzu, dass nur wenige Lerndaten zur Verfügung stehen und die Lernstichproben zudem häufig unbalanciert sind. In den allermeisten Fällen ist außerdem eine Detektion und Rückweisung unbekannter, d. h. nicht in der Lernstichprobe vertretenen Klassen gefordert.

4.1 Zusammenfassung

In dieser Arbeit wurde zunächst an einem Fallbeispiel verifiziert, ob lernende Ansätze im Kontext der Schüttgutsortierung überhaupt einen Vorteil gegenüber dem Stand der Technik ermöglichen. Dazu wurden die Parameter eines bestehenden Systems zur Farbsortierung identifiziert und mit Hilfe eines Lerndatensatzes automatisch festgelegt. Im Vergleich zu den Einstellungen eines menschlichen Experten wurde so ein vergleichbares oder sogar besseres Sortierergebnis erreicht. Die Parametersuche dauerte dabei lediglich wenige Minuten, anstatt wie sonst mehrere Tage. Motiviert durch dieses Ergebnis wurde die Bildverarbeitungskette optischer Sortiersysteme analysiert und Ansatzpunkte für lernende Verfahren aufgedeckt. Diese sind die Bilderfassung, die Merkmalsextraktion und die eigentliche Klassifikation.
In Abschnitt 3.1 wurde die Bilderfassung durch Selektion optischer Filter optimiert, welche die diskriminativen Merkmale der Objekte hervorheben. Durch Abbildung von Filtern auf Merkmale im Sinne der Mustererkennung kann die Filterselektion dabei als Merkmalsselektion aufgefasst werden. In Experimenten mit Datensätzen aus der Lebensmittelinspektion und dem Bergbau wurden Vertreter der Wrapper-, Filter- und eingebetteten Verfahren zur Merkmalsselektion evaluiert. Im Gegensatz zu verwandten Ansätzen in der Literatur ist der hier vorgestellte Ansatz nicht auf die Selektion optischer Bandpassfilter beschränkt, sondern kann optische Filter mit beliebig komplizierten Transmissionskurven selektieren. Da die Filterantwort solcher Filter mit komplizierten spektralen Signaturen durch einen menschlichen

Experten nur schwer interpretierbar ist, bietet der hier vorgestellte Ansatz ein wichtiges Hilfsmittel zur Auslegung von Sortiersystemen.
Der Hauptteil der Arbeit konzentrierte sich auf die Gewinnung geeigneter Merkmale. Geeignete Merkmale sind von herausragender Bedeutung für die Klassifikation, da durch die Merkmalsextraktion der Großteil der in den Mustern enthaltenen Informationen verworfen wird. Die automatische Selektion geeigneter Merkmale aus einer Menge an Kandidaten ist zwar gut erforscht, jedoch wird die zur Extraktion der Merkmale benötigte Rechenzeit meist nicht beachtet.
In Abschnitt 3.2 wurde deswegen ein Verfahren entwickelt, welches diese Kosten bei der Selektion beachtet. Im Gegensatz zu ähnlichen Ansätzen wurde die Selektion dafür als multikriterielles Optimierungsproblem formuliert, wodurch mehrere gleichwertige Pareto-optimale Lösungen ermittelt werden. Die endgültige Abwägung zwischen Relevanz und Kosten der Merkmale obliegt dem Benutzer. Alternativ zu einer manuellen Festlegung kann die Wahl einer passenden Lösung aber auch automatisch zur Laufzeit durchgeführt werden. Um Deadlineverletzungen zu vermeiden, kann so bei hoher hoher Systemlast die Rechenzeit der Klassifikation reduziert werden, ohne dadurch zu stark an Genauigkeit zu verlieren. In Versuchen mit realen Daten wurden mit diesen Ansatz bessere Ergebnisse erzielt als mit alternativen Ansätzen.
Neben der Selektion von Standardmerkmalen wurden auch Verfahren untersucht, mit denen diskriminative Merkmale aus einem Datensatz gelernt werden können. Mit den Shape Ferns wurde in Abschnitt 3.3 ein Verfahren vorgestellt, das schnell berechenbare primitive Kontur- und Flächenmerkmale zu aussagekräftigen Merkmalen zusammenfasst. Im Vergleich zu den etablierten Fourier-Konturdeskriptoren wurde mit den Shape Ferns eine deutlich höhere Klassifikationsgüte erreicht. Die gelernten Merkmale lassen sich außerdem zur Inspektion durch menschliche Experten visuell darstellen. Dadurch können Zusammenhänge in den Daten aufgedeckt werden, die durch klassisches Feature Engineering ausgenutzt werden können.

Neben diesen Formmerkmalen wurde in Abschnitt 3.4 der Bag of Visual Words Ansatz verwendet, um diskriminative Farb- und Texturmerkmale anhand einer Lernstichprobe zu lernen. Der klassische Ansatz wurde durch die Verwendung von primitiven Merkmalen und dichter Abtastung für den Einsatz in der Schüttgutsortierung abgestimmt. In einem zweiten Schritt wurde der BOW-Ansatz um eine Detektion anomaler primitiver Deskriptoren erweitert. Dadurch wurden die Objektdeskriptoren einerseits entrauscht, andererseits wurde die Rückweisung unbekannter Objekte ermöglicht. Die Rückweisung wurde dabei im Sinne von statistischen Tests für Bediener einfach interpretierbar parametriert. In umfangreichen Experimenten mit realen Datensätzen aus der Lebensmittel- und Mineralsortierung wurde gezeigt, dass dieser Ansatz Standardverfahren überlegen ist. Außerdem erreicht die Methode auch mit kleinen Datensätzen und unabhängig von der Problemdomäne eine gute Klassifikationsleistung. Somit ist die Methode gut für den Einsatz in der Schüttgutsortierung geeignet.
Im letzten Abschnitt 3.5 des Anwendungskapitels wurde schließlich die Klassifikation in der offenen Welt, d. h. unter unvollständiger Kenntnis der vorkommenden Klassen, behandelt. Statt bestehende Klassifikatoren mit einer Rückweisungsoption auszustatten, wurde hierzu ein Verfahren zur Einklassenklassifikation entwickelt. Solche Ansätze sind besonders dann wertvoll, wenn Positivbeispiele einfach, Defektbeispiele aber schwer zu beschaffen sind. Dies ist bei der Schüttgutsortierung oft der Fall. Das hier vorgestellte GMB-Verfahren basiert auf einer Schätzung der Merkmalsdichte mit Hilfe hierarchischer Gauß-Mischmodelle. Unter Ausnutzung der hierarchischen Struktur wurde ein Verfahren entwickelt, mit dem die Dichte schnell nach unten abgeschätzt werden kann. Die hierarchische Struktur wurde weiter ausgenutzt, um sicher unbekannte Objekte früh zurückzuweisen. Durch diese Methode wurde die benötigte Rechenzeit im Vergleich zur vollständigen Dichteschätzung erheblich reduziert. Wie bei den BOW-Deskriptoren kann die Rückweisungsoption im Sinne von statistischen Tests bzw. Signifikanzniveaus für Bediener einfach verständlich parametriert werden. Im

Vergleich mit Einklassenklassifikation durch (flache) Gauß-Mischmodelle und One Class SVMs wurden mit den Gauß-Mixturbäumen ähnliche oder bessere Klassifikationsergebnisse erzielt. Gleichzeitig benötigten die Gauß-Mixturbäume deutlich weniger umfangreiche Lernstichproben zur Erzielung einer guten Klassifikationsgüte und die Rückweisung unbekannter Objekte konnte deutlich schneller berechnet werden als mit den anderen Methoden. Somit wurden für alle Teilaufgaben der Bildverarbeitungskette lernende Verfahren vorgestellt, die die Entwicklung von Sortiersystemen durch automatisierte Analysen unterstützen. Diese Analysen können auch nichtlineare, nicht intuitive Zusammenhänge aufdecken und so zu verbesserter Systemleistung führen. Aus Anwendersicht wird die Inbetriebnahme der Systeme vereinfacht, da sie keine Kenntnisse über die Interna des Systems benötigen, sondern lediglich annotierte Datensätze bereitstellen müssen. Da die Anwender Domänenexperten sind, ist diese Aufgabe für sie in der Regel einfach durchführbar.

4.2 Ausblick

Der Fokus dieser Arbeit lag auf der Schüttgutsortierung, also einer klassischen Klassifikationsaufgabe. Die vorgestellten Methoden lassen sich prinzipiell auch auf andere optische Inspektionsaufgaben übertragen, solange diese als Klassifikationsproblem formuliert werden können. Beispiele sind die Vollständigkeits- und Lageprüfung sowie die Defektdetektion, bei der die Klassifikation eine Teilaufgabe darstellt. Einige Ansätze lassen sich auch auf andere Aufgabenbereiche der optischen Inspektion übertragen. Die Selektion optischer Filter kann auch als Merkmalsselektion im Kontext eines Regressionsproblems formuliert werden, wodurch die Selektion von für die Oberflächenbewertung und Qualitätsprüfung optimierter Filter ermöglicht wird. Die Vermessung von Formparametern kann durch einen modifizierten Shape Fern-Ansatz realisiert werden, indem zum Boosting nicht L_2_TreeBoost, sondern die Regressionsansätze LAD_TreeBoost oder M_TreeBoost (siehe

Friedman [Fri01]) verwendet werden. Neben diesen und in den jeweiligen Abschnitten in Kapitel 3 beschriebenen Weiterentwicklungen ergeben sich aber auch allgemeinere Forschungsfragen.

Sensorik. Es ist absehbar, dass in der näheren Zukunft bildgebende Sensoren praxistauglich werden, die nicht notwendigerweise im visuellen Bereich des elektromagnetischen Spektrums operieren und eine höhere spektrale Auflösung aufweisen als übliche RGB-Kameras. Insbesondere hyperspektrale Bildgebung, also Bildgebung mit einer sehr hohen spektralen Auflösung, weist hohes Potential auf [GOW+07; FS12; Sch+15]. Es ist zu prüfen, welche Methoden zur Verarbeitung dieser Daten geeignet sind. Erste Arbeiten in diese Richtung wurden bereits veröffentlicht, siehe Richter et al. [RLB15b].

Deep Learning. Ebenso ist zu untersuchen, inwieweit sich Methoden des Deep Learning [LYG15; Sch15] im Kontext der visuellen Inspektion anwenden lassen. Problematisch scheint hier vor allem die relativ hohe Anzahl an benötigten Beispielen und das Fehlen von Richtlinien zur Erstellung geeigneter Architekturen sowie theoretischer Aussagen zur erwartbaren Klassifikationsleistung. Entsprechende Arbeiten existieren zwar (z. B. Kabkab et al. [KHC16] und Song et al. [Son+17]), sind jedoch noch schwer in die Praxis übertragbar.

Akzeptanz. Generelle Fragestellungen ergeben sich aus der Akzeptanz lernender Methoden durch die Endanwender. Diese Akzeptanz wird im Wesentlichen durch die Blackbox-Natur maschineller Lernverfahren behindert. Oft wird eine „Interpretierbarkeit" der Methoden gefordert, jedoch würden auch Möglichkeiten zur indirekten Beeinflussung des Systemverhaltens sowie theoretische Garantien über die erwartete Systemleistung die Akzeptanz steigern.
Der Begriff „Interpretierbarkeit" muss dabei genauer qualifiziert werden. Nach dem Modell von Lipton [Lip16] ist im Kontext der visuellen Inspektion damit

meist die Zerlegbarkeit (decomposability) und die Post-hoc Interpretierbarkeit gemeint. Zerlegbarkeit bedeutet, dass alle Teile der Bildverarbeitungskette – also Merkmale sowie Klassifikations- und Hyperparameter – eine intuitive Erklärung zulassen. Dies kann beispielsweise durch die Verwendung von einfachen Standardmerkmalen oder durch Feature Engineering erreicht werden. Wenn die damit verbundenen Nachteile, also geringere Relevanz der Merkmale bzw. hoher Modellierungsaufwand nicht akzeptiert werden können, bieten sich Ansätze an, die primitive Merkmale zu diskriminativen Deskriptoren zusammensetzen. Neben den in dieser Arbeit verwendeten Shape Ferns und BOW-Deskriptoren können hierzu auch Classemes und verwandte Methoden [BT14] oder auf Dictionary Learning [Mai+09b] basierende Ansätze verwendet werden.
Bei der Klassifikation ist Zerlegbarkeit grundsätzlich nur für einfache Modelle und niedrigdimensionale Merkmalsräume erreichbar, doch selbst hier ist diese nicht garantiert [Lip16].
Die Post-hoc Interpretierbarkeit, also die Erklärung einer bestimmten Klassifikationsentscheidung etwa durch Visualisierung ähnlicher Beispiele in der Lernstichprobe scheint erfolgsversprechender. Da die Erklärung nicht notwendigerweise direkt aus dem zugrunde liegenden Klassifikator abgeleitet werden muss, können hier auch kompliziertere Modelle angesetzt werden. Zhang und Goncalves [ZG15] approximieren etwa die Entscheidungsgrenze in der Nähe des zu erklärenden Merkmalsvektors durch ein lineares Modell und nutzen dieses einfachere Modell, um die gegebene Klassifikationsentscheidung zu erklären. Bei der Verwendung von Ensemble-Klassifikatoren bietet es sich auch an, die Entscheidung anhand der Einzelentscheidungen im Ensemble zu erklären. Hierzu müssen allerdings noch geeignete Methoden entwickelt werden.
Wenn weder Zerlegbarkeit noch Post-hoc Interpretierbarkeit herstellbar ist, kann die Akzeptanz lernender Methoden durch Bereitstellung von Parametern zur indirekten Beeinflussung der Systemleistung gesteigert werden. Wie bei den BOW-Deskriptoren und den GMBs bieten sich hierfür Methoden

an, die auf statistischen Tests basieren. Neben der Klasse würde dann eine Fehlerwahrscheinlichkeit der Entscheidung angegeben und die Klassifikationsleistung des Systems ließe sich durch Signifikanzniveaus steuern. Da diese Signifikanzniveaus Wahrscheinlichkeiten repräsentieren, lässt sich der Einfluss dieser Parameter auch von Laien gut vorhersagen. Der Nachteil an diesem Ansatz ist, dass die anwendbaren statistischen Tests vom verwendeten Klassifikator abhängen. Die in dieser Arbeit entwickelten Methoden lassen sich ohne Weiteres nicht auf andere Systeme übertragen.
Eine weitere Methode zur Bereitstellung von indirekten Klassifikationsparametern ergibt sich aus der Risikominimierung der Bayes'schen Entscheidungstheorie (siehe Abschnitt 2.4). Die Systemleistung ließe sich durch die Vorgabe von Kosten für jedes Klassifikationsergebnis steuern. Da die Anwender die Kosten von Fehlklassifikationen in der Regel gut kennen, ist eine solche Parametrierung gut für die Praxis geeignet.
Eine letzte Methode zur Steigerung der Akzeptanz lernender Methoden ist die Angabe gewisser Garantien in Bezug auf die erwartbare Systemleistung. Dies betrifft vor allem Fragen nach der für eine gewünschte Klassifikationsleistung benötigten Stichprobengröße, aber auch Fragen bezüglich der Dimension des Merkmalsraums und der Anzahl der Modellparameter. Da hierzu noch eine einheitliche, generell anwendbare Theorie fehlt, kann sich hier ebenfalls mit statistischen Methoden beholfen werden.

Adaptive Systeme. Weitere Forschungsfragen ergeben sich ebenfalls aus der Anwendung. Neben Methoden zur Klassifikation in der offenen Welt betrifft dies adaptive Systeme, die sich in Anlehnung an ein adaptives Filter [Hay86] im Betrieb an eine veränderte Datenlage anpassen können. Übertragen auf die Schüttgutsortierung bedeutet dies die Verwendung von Methoden des Online Learning, also Verfahren, bei denen der Klassifikator im Betrieb mit neuen Lerndaten laufend trainiert werden kann. Hier stellt sich die Frage, wie veraltete Lerndaten und deren Einfluss auf die Entscheidungs-

grenzen verworfen werden können, um aus lokalen Minima zu entkommen. Ein Beispiel für ein solches System findet sich in Sun und Sun [SS09].
Online Learning ist insbesondere dann wichtig, wenn sich die Merkmalsausprägungen mit der Zeit verändern (Feature Drift), wie es beispielsweise über die Ernteperioden bei natürlichen Produkten der Fall sein kann. Verfahren zur Detektion von Feature Drift können auch eingesetzt werden, um Systemdefekte, beispielsweise verschmutzte Linsen oder Sichtfenster oder defekte Kamerasensoren, frühzeitig zu erkennen.
In Verbindung mit Online Learning sind auch Methoden des aktiven Lernens, siehe z. B. Settles [Set12], interessant. Solche Methoden könnten eingesetzt werden, um ein gegebenes System zur Laufzeit zu überwachen und den Nutzer zur Nachklassifikation von unsicher klassifizierten Objekten aufzufordern. Neben der Verbesserung der Klassifikationsleistung können so auch neue Defektklassen in den Daten entdeckt werden. Die Überwachung zur Laufzeit entspricht dem Stream-Based Selective Sampling Szenario des aktiven Lernens [Set12]. Alternativ können auch im Vorfeld oder während des Betriebs Beispiele generiert und dem Benutzer zur Klassifikation angeboten werden. Diese Beispiele werden so generiert, dass sie maximalen Informationsgehalt in Bezug auf den Klassifikator haben (Membership Query Synthesis [Set12]). Im Kontext der visuellen Inspektion können diese Beispiele durch parametrische Computergrafikmodelle erzeugt werden. Entsprechende Vorarbeiten wurden in Retzlaff et al. [Ret+16] veröffentlicht. Der Nachteil dieser Methode ist, dass zunächst ein generatives Modell der zu klassifizierenden Objekte aufgestellt werden muss, was wiederum mit erhöhtem Aufwand beim Systementwurf verbunden ist.
Um dieses Problem zu umgehen, könnten diese Beispiele auch mit Generative Adversial Networks (GANs) [Goo+14] generiert werden. Entsprechende Ideen wurden auch von Zhu und Bento [ZB17] geäußert. Im Kontext der visuellen Inspektion kann hier allerdings die große Anzahl an benötigten Daten für das Training des GANs problematisch werden.

Schließlich eröffnen sich durch die Anwendung von Methoden des Transfer Learning [PY10] neue Geschäftsfelder. Sortiersysteme könnten offline anhand reichhaltiger, aber generischer Daten vortrainiert und erst bei der Inbetriebnahme an die jeweilige Zieldomäne angepasst werden. Wenn die Objekte der Zieldomäne nicht zu stark von den zum Vortraining verwendeten Objekten abweichen, kann dadurch eine erhebliche Reduktion der benötigten Lerndaten erreicht werden.

Abbildungsverzeichnis

Tabellenverzeichnis

Algorithmenverzeichnis

Literaturverzeichnis

[AK98] E. Amaldi und V. Kann. “On the approximability of minimizing nonzero variables or unsatisfied relations in linear systems”. *Theoretical Computer Science* 209.1-2 (1998), S. 237–260. DOI: 10.1016/S0304-3975(97)00115-1.

[AY03] N. Arica und F. T. Yarman Vural. “BAS: a perceptual shape descriptor based on the beam angle statistics”. *Pattern Recognition Letters* 24.9-10 (Juni 2003), S. 1627–1639. DOI: 10.1016/S0167-8655(03)00002-3.

[Bar+89] V. E. Barker, D. E. O’Connor, J. Bachant und E. Soloway. “Expert systems for configuration at Digital: XCON and beyond”. *Communications of the ACM* 32.3 (1989), S. 298–318. DOI: 10.1145/62065.62067.

[Bat94] R. Battiti. “Using Mutual Information for Selecting Features in Supervised Neural Net Learning”. *IEEE Transactions on Neural Networks* 5.4 (1994), S. 537–550. DOI: 10.1109/72.298224.

[Bay+15] A. G. Baydin, B. A. Pearlmutter, A. A. Randul und J. M. Siskind. “Automatic differentiation in machine learning: a survey”. *arXiv preprint* (2015), S. 28.

[Bay16] J. Bayer. “Klassifikation mit Rückweisung durch Erweiterung von Entscheidungsbäumen”. Bacherlorarbeit. Karlsruher Institut für Technologie, 2016.

[Bel61] R. Bellman. *Adaptive control processes—A guided tour*. 5. Aufl. Princeton University Press, 1961. DOI: 10.1002/nav.3800080314.

[Ben75] J. L. Bentley. "Multidimensional binary search trees used for associative searching". *Communications of the ACM* 18.9 (1975), S. 509–517. DOI: 10.1145/361002.361007.

[Ber+00] L. Bergasa, N. Duffy, G. Lacey und M. Mazo. "Industrial inspection using Gaussian functions in a colour space". *Image and Vision Computing* 18.12 (Sep. 2000), S. 951–957. DOI: 10.1016/S0262-8856(00)00035-4.

[Bey96] J. Beyerer. "Modellgestützte Sichtprüfung spanend bearbeiteter Oberflächen". *tm – Technisches Messen* 63.5 (1996), S. 182–190.

[Bez+17] J. Bezanson, A. Edelman, S. Karpinski und V. B. Shah. "Julia: A Fresh Approach to Numerical Computing". *SIAM Review* 59.1 (2017), S. 65–98. DOI: 10.1137/141000671.

[BGV92] B. E. Boser, I. M. Guyon und V. N. Vapnik. "A training algorithm for optimal margin classifiers". *Workshop on Computational learning theory (COLT)*. Bd. 232. 1992, S. 144–152. DOI: 10.1145/130385.130401.

[BH89] P. Baldi und K. Hornik. "Neural networks and principal component analysis: Learning from examples without local minima". *Neural Networks* 2.1 (Jan. 1989), S. 53–58. DOI: 10.1016/0893-6080(89)90014-2.

[BHK97] P. N. Belhumeur, J. P. Hespanha und D. J. Kriegman. "Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection". *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 19.7 (1997), S. 711–720. DOI: 10.1109/34.598228.

[Bis13] C. M. Bishop. *Pattern Recognition and Machine Learning*. Bd. 53. 9. 2013. DOI: 10.1117/1.2819119.

[Bla+09] J. Blasco, S. Cubero, J. Gómez-Sanchís, P. Mira und E. Moltó. "Development of a machine for the automatic sorting of pomegranate (Punica granatum) arils based on computer vision". *Journal of Food Engineering* 90.1 (Jan. 2009), S. 27–34. DOI: 10.1016/j.jfoodeng.2008.05.035.

[BMP01] S. Belongie, J. Malik und J. Puzicha. "Shape Context: A New Descriptor for Shape Matching and Object Recognition". *Advances in Neural Information Processing Systems (NIPS)*. Hrsg. von T. K. Leen, T. G. Dietterich und V. Tresp. MIT Press, Dez. 2001, S. 831–837.

[BNJ03] D. M. Blei, A. Y. Ng und M. I. Jordan. "Latent Dirichlet Allocation". *Journal of Machine Learning Research (JMLR)* 3 (2003), S. 993–1022. DOI: 10.1162/jmlr.2003.3.4-5.993.

[Bot10] L. Bottou. "Large-Scale Machine Learning with Stochastic Gradient Descent". *Proceedings of COMPSTAT'2010*. Heidelberg: Physica-Verlag HD, 2010, S. 177–186. DOI: 10.1007/978-3-7908-2604-3_16.

[BPF12] J. Beyerer, F. Puente Leon und C. Frese. *Automatische Sichtprüfung: Grundlagen, Methoden und Praxis der Bildgewinnung und Bildauswertung*. 2012. DOI: 10.1007/978-3-642-23966-3.

[BPL15] J. Beyerer, F. Puente León und T. Längle. "2nd International Conference on Optical Characterization of Materials". *Optical Characterization of Materials*. 2015. DOI: 10.5445/KSP/1000032143.

[BPL17] J. Beyerer, F. Puente León und T. Längle. “3rd International Conference on Optical Characterization of Materials”. *Optical Characterization of Materials*. 2017.

[Bre+84] L. Breiman, J. Friedman, C. J. Stone und R. A. Olshen. *Classification and regression trees*. CRC press, 1984.

[Bre01] L. Breiman. “Random forests”. *Machine Learning* 45.1 (Okt. 2001), S. 5–32. DOI: 10.1023/A:1010933404324.

[BRN17] J. Beyerer, M. Richter und M. Nagel. *Pattern Recognition: Introduction, Features, Classifiers and Principles*. De Gruyter, 2017.

[Bro+12] G. Brown, A. Pocock, M.-J. Zhao und M. Luján. “Conditional likelihood maximisation: a unifying framework for information theoretic feature selection”. *Journal of Machine Learning Research (JMLR)* 13.1 (März 2012), S. 27–27–66–66.

[BT14] A. Bergamo und L. Torresani. “Classemes and Other Classifier-Based Features for Efficient Object Categorization”. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 36.10 (Okt. 2014), S. 1988–2001. DOI: 10.1109/TPAMI.2014.2313111.

[BTF11] A. Bergamo, L. Torresani und A. W. Fitzgibbon. “PiCoDes: Learning a Compact Code for Novel-Category Recognition”. *Advances in Neural Information Processing Systems (NIPS)*. 2011, S. 2088–2096.

[Bur+12] A. Burla, T. Haist, W. Lyda und W. Osten. “Genetic programming applied to automatic algorithm design in multiscale inspection systems”. *Optical Engineering* 51.6 (2012), S. 67001–67013. DOI: doi:10.1117/1.oe.51.6.067001.

[BV03] V. Blanz und T. Vetter. “Face recognition based on fitting a 3D morphable model”. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 25.9 (Sep. 2003), S. 1063–1074. DOI: 10.1109/TPAMI.2003.1227983.

[BV09] S. Boyd und L. Vandenberghe. *Convex optimization*. 2009.

[Car+15] D. Carrera, G. Boracchi, A. Foi und B. Wohlberg. “Detecting anomalous structures by convolutional sparse models”. *International Joint Conference on Neural Networks (IJCNN)*. IEEE, Juli 2015, S. 1–8. DOI: 10.1109/IJCNN.2015.7280790.

[Cha+01] K. Chao, Y. R. Chen, W. R. Hruschka und B. Park. “Chicken heart disease characterization by multi-spectral imaging”. *Applied Engineering in Agriculture* 17.1 (2001), S. 99–106.

[Cha+11] K. Chatfield, V. Lempitsky, A. Vedaldi und A. Zisserman. “The devil is in the details: an evaluation of recent feature encoding methods”. *Proceedings of the British Machine Vision Conference (BMVC)*. Nov. 2011, S. 76.1–76.12.

[Cha07] O. Chapelle. “Training a support vector machine in the primal.” *Neural Computation* 19.5 (2007), S. 1155–1178. DOI: 10.1162/neco.2007.19.5.1155.

[Coo+95] T. Cootes, C. Taylor, D. Cooper und J. Graham. *Active Shape Models-Their Training and Application*. 1995. DOI: 10.1006/cviu.1995.1004.

[Cri11] A. Criminisi. “Decision Forests: A Unified Framework for Classification, Regression, Density Estimation, Manifold Learning and Semi-Supervised Learning”. *Foundations and Trends® in Computer Graphics and Vision* 7.2-3 (2011), S. 81–227. DOI: 10.1561/0600000035.

[CS01] K. Crammer und Y. Singer. “On the Algorithmic Implementation of Multiclass Kernel-based Vector Machines”. *Journal of Machine Learning Research (JMLR)* 2.12 (2001), S. 265–292.

[CS14] G. Chandrashekar und F. Sahin. “A survey on feature selection methods”. *Computers & Electrical Engineering* 40.1 (Jan. 2014), S. 16–28. DOI: 10.1016/j.compeleceng.2013.11.024.

[CSS12] C. Cusano, R. Satta und S. Santini. “Unsupervised Classemes”. *European Conference on Computer Vision (ECCV)*. Bd. 7585. 1. 2012, S. 406–415.

[Csu+04] G. Csurka, C. Dance, L. Fan, J. Willamowski und C. Bray. “Visual categorization with bags of keypoints”. *ECCV Workshop on Statistical Learning in Computer Vision*. 2004, S. 1–22.

[CSZ06] O. Chapelle, B. Schölkopf und A. Zien. *Semi-Supervised Learning*. 1st. The MIT Press, 2006.

[Cub+11] S. Cubero, N. Aleixos, E. Moltó, J. Gómez-Sanchis und J. Blasco. “Advances in Machine Vision Applications for Automatic Inspection and Quality Evaluation of Fruits and Vegetables”. *Food and Bioprocess Technology* 4.4 (Mai 2011), S. 487–504. DOI: 10.1007/s11947-010-0411-8.

[DCL00] N. Duffy, J. Crowley und G. Lacey. “Object detection using colour”. *International Conference on Pattern Recognition (ICPR)*. Bd. 1. IEEE Comput. Soc, 2000, S. 700–703. DOI: 10.1109/ICPR.2000.905483.

[Deb+02] K. Deb, A. Pratap, S. Agarwal und T. Meyarivan. “A fast and elitist multiobjective genetic algorithm: NSGA-II”. *IEEE Transactions on Evolutionary Computation* 6.2 (Apr. 2002), S. 182–197. DOI: 10.1109/4235.996017.

[DeB+05] S. DeBacker, P. Kempeneers, W. Debruyn und P. Scheunders. "A Band Selection Technique for Spectral Classification". *IEEE Geoscience and Remote Sensing Letters* 2.3 (Juli 2005), S. 319–323. DOI: 10.1109/LGRS.2005.848511.

[Der+03] J. Derganc, B. Likar, R. Bernard, D. Tomaževič und F. Pernuš. "Real-time automated visual inspection of color tablets in pharmaceutical blisters". *Real-Time Imaging* 9.2 (Apr. 2003), S. 113–124. DOI: 10.1016/S1077-2014(03)00018-4.

[Dés+13] C. Désir, S. Bernard, C. Petitjean und L. Heutte. "One class random forests". *Pattern Recognition* 46.12 (Dez. 2013), S. 3490–3506. DOI: 10.1016/j.patcog.2013.05.022.

[DHS01] R. O. Duda, P. E. Hart und D. G. Stork. *Pattern Classification*. 2. Aufl. John Wiley & Sons, 2001.

[DHS11] J. Duchi, E. Hazan und Y. Singer. "Adaptive subgradient methods for online learning and stochastic optimization". *Journal of Machine Learning Research (JMLR)* 12 (2011), S. 2121–2159.

[DLR77] A. P. Dempster, N. M. Laird und D. B. Rubin. "Maximum likelihood from incomplete data via the EM algorithm". *Journal of the Royal Statistical Society* 39.1 (1977), S. 1–38. DOI: 10.2307/2984875.

[DM93] B. Dubuisson und M. Masson. "A statistical decision rule with incomplete knowledge about classes". *Pattern Recognition* 26.1 (Jan. 1993), S. 155–165. DOI: 10.1016/0031-3203(93)90097-G.

[Dol+09] P. Dollár, Z. Tu, P. Perona und S. Belongie. "Integral Channel Features." *Proceedings of the British Machine Vision Conference (BMVC)*. 2009, S. 244.1–244.11.

[Dom12] P. Domingos. “A few useful things to know about machine learning”. *Communications of the ACM* 55.10 (2012), S. 78. DOI: 10.1145/2347736.2347755.

[Don+13] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng und T. Darrell. “DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition”. *arXiv preprint* (Okt. 2013).

[DS06] C.-J. Du und D.-W. Sun. “Learning techniques used in computer vision for food quality evaluation: a review”. *Journal of Food Engineering* 72.1 (Jan. 2006), S. 39–55. DOI: 10.1016/j.jfoodeng.2004.11.017.

[DT05] N. Dalal und B. Triggs. “Histograms of Oriented Gradients for Human Detection”. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Bd. 1. 4. IEEE, 2005, S. 886–893. DOI: 10.1109/CVPR.2005.177.

[Efr79] B. Efron. “Bootstrap Methods: Another Look at the Jackknife”. *The Annals of Statistics* 7.1 (1979), S. 1–26. DOI: 10.1214/aos/1176344552.

[EH16] B. Efron und T. Hastie. *Computer Age Statistical Inference*. Cambridge: Cambridge University Press, 2016. DOI: 10.1017/CBO9781316576533.

[ElB+15] N. El-Bendary, E. El Hariri, A. E. Hassanien und A. Badr. “Using machine learning techniques for evaluating tomato ripeness”. *Expert Systems with Applications* 42.4 (März 2015), S. 1892–1905. DOI: 10.1016/j.eswa.2014.09.057.

[Far+09] A. Farhadi, I. Endres, D. Hoiem und D. Forsyth. “Describing objects by their attributes”. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, Juni 2009, S. 1778–1785. DOI: 10.1109/CVPR.2009.5206772.

[FBF77] J. H. Friedman, J. L. Bentley und R. A. Finkel. “An Algorithm for Finding Best Matches in Logarithmic Expected Time”. *ACM Transactions on Mathematical Software* 3.3 (1977), S. 209–226. DOI: 10.1145/355744.355745.

[FE04] B. Froba und A. Ernst. “Face detection with the modified census transform”. *International Conference on Automatic Face and Gesture Recognition (FGR)*. 2004, S. 1–6.

[FG01] F. Feyaerts und L. van Gool. “Multi-spectral vision system for weed detection”. *Pattern Recognition Letters* 22.6-7 (Mai 2001), S. 667–674. DOI: 10.1016/S0167-8655(01)00006-X.

[FHT00] J. Friedman, T. Hastie und R. Tibshirani. “Additive logistic regression: a statistical view of boosting”. *The Annals of Statistics* 28.2 (2000), S. 337–407.

[Fis+14] L. Fischer, D. Nebel, T. Villmann, B. Hammer und H. Wersing. “Rejection Strategies for Learning Vector Quantization – A Comparison of Probabilistic and Deterministic Approaches”. *Advances in Intelligent Systems and Computing*. Bd. 295. 2014, S. 109–118. DOI: 10.1007/978-3-319-07695-9_10.

[Fle04] F. Fleuret. “Fast Binary Feature Selection with Conditional Mutual Information”. *Journal of Machine Learning Research (JMLR)* 5 (2004), S. 1531–1555.

[Fre61] H. Freeman. “On the Encoding of Arbitrary Geometric Configurations”. *IRE Transactions on Electronic Computers* EC-10.2 (1961), S. 260–268. DOI: 10.1109/TEC.1961.5219197.

[Fri01] J. H. Friedman. “Greedy function approximation: A gradient boosting machine.” *The Annals of Statistics* 29.5 (Okt. 2001), S. 1189–1232. DOI: 10.1214/aos/1013203451.

[FS12] Y.-Z. Feng und D.-W. Sun. “Application of Hyperspectral Imaging in Food Safety Inspection and Control: A Review”. *Critical Reviews in Food Science and Nutrition* 52.11 (Nov. 2012), S. 1039–1058. DOI: 10.1080/10408398.2011.651542.

[FS96] Y. Freund und R. E. Schapire. “Experiments with a New Boosting Algorithm”. *International Conference on Machine Learning (ICML)*. 1996, S. 148–156.

[Fuk80] K. Fukushima. “Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position”. *Biological Cybernetics* 36.4 (Apr. 1980), S. 193–202. DOI: 10.1007/BF00344251.

[Gem+08] J. C. van Gemert, J.-M. Geusebroek, C. J. Veenman und A. W. M. Smeulders. “Kernel Codebooks for Scene Categorization”. *Eccv*. Bd. 6893. Pt 3. 2008, S. 696–709. DOI: 10.1007/978-3-540-88690-7_52.

[Goo+14] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville und Y. Bengio. “Generative Adversarial Nets”. *Advances in Neural Information Processing Systems (NIPS)*. Hrsg. von Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence und K. Q. Weinberger. Curran Associates, Inc., Juni 2014, S. 2672–2680. DOI: 10.1001/jamainternmed.2016.8245.

[GOW+07] A. GOWEN, C. ODONNELL, P. CULLEN, G. DOWNEY und J. FRIAS. “Hyperspectral imaging – an emerging process analytical tool for food quality and safety control”. *Trends in Food Science & Technology* 18.12 (Dez. 2007), S. 590–598. DOI: 10.1016/j.tifs.2007.06.001.

[GR10] D. Görür und C. E. Rasmussen. “Dirichlet process gaussian mixture models: Choice of the base distribution”. *Journal of Computer Science and Technology* 25.4 (2010), S. 653–664. DOI: 10.1007/s11390-010-1051-1.

[Gra84] R. M. Gray. “Vector quantization”. *IEEE ASSP Magazine* 1.2 (1984), S. 4–29. DOI: 10.1109/MASSP.1984.1162229.

[Grü+04] F. Grützner, W. Rens, E. Tsend-Ayush, N. El-Mogharbel, P. C. M. O’Brien, R. C. Jones, M. A. Ferguson-Smith und J. A. Marshall Graves. “In the platypus a meiotic chain of ten sex chromosomes shares genes with the bird Z and mammal X chromosomes”. *Nature* 432 (Okt. 2004), S. 913. DOI: 10.1038/nature03021.

[Guo+06] B. Guo, S. Gunn, R. Damper und J. Nelson. “Band Selection for Hyperspectral Image Classification Using Mutual Information”. *IEEE Geoscience and Remote Sensing Letters* 3.4 (Okt. 2006), S. 522–526. DOI: 10.1109/LGRS.2006.878240.

[Guy+05] I. Guyon, S. Gunn, A. Ben-Hur und G. Dror. “Result Analysis of the NIPS 2003 Feature Selection Challenge”. *Advances in Neural Information Processing Systems (NIPS)*. Hrsg. von L. K. Saul, Y. Weiss und L. Bottou. MIT Press, 2005, S. 545–552.

[Hal99] M. A. Hall. “Correlation-based Feature Selection for Machine Learning”. Diss. University of Waikato Hamilton, 1999.

[Ham+07] T. M. Hamdani, J.-m. Won, A. M. Alimi und F. Karray. “Multi-objective Feature Selection with NSGA II”. *Adaptive and Natural Computing Algorithms*. Hrsg. von B. Beliczynski, A. Dzielinski, M. Iwanowski und B. Ribeiro. Bd. 4431. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2007, S. 240–247. DOI: 10.1007/978-3-540-71618-1_27.

[Hay86] S. Haykin. *Adaptive filter theory*. Prentice-Hall, Inc., 1986.

[HDE78] P. E. Hart, R. O. Duda und M. T. Einaudi. “PROSPECTOR-A computer-based consultation system for mineral exploration”. *Journal of the International Association for Mathematical Geology* 10.5 (Okt. 1978), S. 589–610. DOI: 10.1007/BF02461988.

[HL01] Z. Hafed und M. Levine. “Face recognition using the discrete cosine transform”. *International Journal of Computer Vision* 43.3 (2001), S. 167–188. DOI: 10.1023/A:1011183429707.

[Hor+06] G. Hornby, A. Globus, D. Linden und J. Lohn. “Automated Antenna Design with Evolutionary Algorithms”. *Space*. Bd. 5. September. Reston, Virigina: American Institute of Aeronautics und Astronautics, Sep. 2006, S. 1–8. DOI: 10.2514/6.2006-7242.

[HS88] C. Harris und M. Stephens. “A Combined Corner and Edge Detector”. *Proceedings of the Alvey Vision Conference*. Alvey Vision Club, 1988, S. 23.1–23.6. DOI: 10.5244/C.2.23.

[HS89] T. Hastie und W. Stuetzle. “Principal Curves”. *Journal of the American Statistical Association* 84.406 (Juni 1989), S. 502–516. DOI: 10.1080/01621459.1989.10478797.

[HSD73] R. M. Haralick, K. Shanmigam und I. Dinstein. “Textural Features for Image Classification”. *IEEE Transactions on Systems, Man, and Cybernetics* 3.6 (1973), S. 610–621.

[HTF09] T. Hastie, R. Tibshirani und J. Friedman. “The Elements of Statistical Learning”. *Elements* 1 (2009), S. 337–387. DOI: 10.1007/b94608.

[HTW15] T. Hastie, R. Tibshirani und M. Wainwright. “Statistical Learning with Sparsity”. (2015).

[Hu62] M.-K. Hu. “Visual Pattern Recognition by Moment Invariants”. *IRE Transactions on Information Theory* 8.2 (1962), S. 179–187. DOI: 10.1109/TIT.1962.1057692.

[Hug68] G. Hughes. “On the mean accuracy of statistical pattern recognizers”. *IEEE Transactions on Information Theory* 14.1 (Jan. 1968), S. 55–63. DOI: 10.1109/TIT.1968.1054102.

[HW06] R. Herbei und M. H. Wegkamp. “Classification with reject option”. *Canadian Journal of Statistics* 34.4 (Dez. 2006), S. 709–721. DOI: 10.1002/cjs.5550340410.

[IK06] K. Iswandy und a. Koenig. “Feature selection with acquisition cost for optimizing sensor system design”. *Advances in Radio Science* 4 (2006), S. 135–141. DOI: 10.5194/ars-4-135-2006.

[IN13] S. Irgenfried und C. Negara. “A framework for storage, visualization and analysis of multispectral data”. *Optical Characterization of Materials*. Karlsruhe: KIT Scientific Publishing, 2013, S. 203–214.

[JH99] T. Jaakkola und D. Haussler. “Exploiting generative models in discriminative classifiers”. *Advances in Neural Information Processing Systems (NIPS)*. 1999, S. 487–493.

[JHG99] B. Jähne, H. Haußecker und P. Geißler. *Handbook of Computer Vision and Applications*. Bd. 2. 4. 1999. DOI: 10.1007/s00138-006-0021-7.

[KB14] D. P. Kingma und J. Ba. “Adam: A Method for Stochastic Optimization”. *Proceedings of the 12th annual conference on Genetic and evolutionary computation - GECCO ’10* (Dez. 2014), S. 103.

[KHC16] M. Kabkab, E. Hand und R. Chellappa. “On the size of Convolutional Neural Networks and generalization performance”. *International Conference on Pattern Recognition (ICPR)*. IEEE, Dez. 2016, S. 3572–3577. DOI: 10.1109/ICPR.2016.7900188.

[KK00] H. K. Kim und J. D. Kim. “Region-based shape descriptor invariant to rotation, scale and translation”. *Signal Processing: Image Communication* 16.1 (2000), S. 87–93. DOI: 10.1016/S0923-5965(00)00018-7.

[KLD03] O. Kleynen, V. Leemans und M.-F. Destain. “Selection of the most efficient wavelength bands for ‘Jonagold’ apple sorting”. *Postharvest Biology and Technology* 30.3 (Dez. 2003), S. 221–232. DOI: 10.1016/S0925-5214(03)00112-1.

[KM14] S. S. Khan und M. G. Madden. “One-class classification: taxonomy of study and review of techniques”. *The Knowledge Engineering Review* 29.03 (Juni 2014), S. 345–374. DOI: 10.1017/S026988891300043X.

[LCG09] C. Li, Q. Cao und F. Guo. “A method for color classification of fruits based on machine vision”. *WSEAS TRANSACTIONS on SYSTEMS* 8.2 (Feb. 2009), S. 312–321.

[Lec+98] Y. Lecun, L. Bottou, Y. Bengio und P. Haffner. “Gradient-based learning applied to document recognition”. *Proceedings of the IEEE* 86.11 (1998), S. 2278–2324. DOI: 10.1109/5.726791.

[Lee96] T. S. Lee. “Image representation using 2D Gabor wavelets”. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 18.10 (1996), S. 959–971. DOI: 10.1109/34.541406.

[Lil+13] K. Lillywhite, D.-J. Lee, B. Tippetts und J. Archibald. “A feature construction method for general object recognition”.

Pattern Recognition 46.12 (Dez. 2013), S. 3300–3314. DOI: 10.1016/j.patcog.2013.06.002.

[Lip16] Z. C. Lipton. "The Mythos of Model Interpretability". *ICML Workshop on Human Interpretability in Machine Learning* (Juni 2016).

[LK02] J. Lafferty und R. I. Kondor. "Diffusion Kernels on Graphs and Other Discrete Input Spaces". *International Conference on Machine Learning (ICML)*. 2002, S. 315–322. DOI: 10.1.1.57.7612.

[Lod+02] H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini und C. Watkins. "Text Classification using String Kernels". *Journal of Machine Learning Research (JMLR)* 2 (2002), S. 419–444. DOI: 10.1162/153244302760200687.

[Lóp+11] R. López-Sastre, T. Tuytelaars, F. Acevedo-Rodríguez und S. Maldonado-Bascón. "Towards a more discriminative and semantic visual vocabulary". *Computer Vision and Image Understanding* 115.3 (März 2011), S. 415–425. DOI: 10.1016/j.cviu.2010.10.009.

[Low04] D. G. Lowe. "Distinctive Image Features from Scale-Invariant Keypoints". *International Journal of Computer Vision* 60.2 (Nov. 2004), S. 91–110. DOI: 10.1023/B:VISI.0000029664.99615.94.

[LYG15] Y. LeCun, B. Yoshua und H. Geoffrey. "Deep learning". *Nature* 521.7553 (2015), S. 436–444. DOI: 10.1038/nature14539.

[Mai+09a] J. Mairal, F. Bach, J. Ponce und G. Sapiro. "Online dictionary learning for sparse coding". *International Conference on Machine Learning (ICML)*. New York, New York, USA: ACM Press, 2009, S. 1–8. DOI: 10.1145/1553374.1553463.

[Mai+09b] J. Mairal, J. Ponce, G. Sapiro, A. Zisserman und F. R. Bach. "Supervised Dictionary Learning". *Advances in Neural Information Processing Systems (NIPS)*. Hrsg. von D. Koller, D. Schuurmans, Y. Bengio und L. Bottou. Curran Associates, Inc., Juni 2009, S. 1033–1040. DOI: 10.1109/CVPR.2010.5539989.

[Mai+16] G. Maier, M. Bromberger, T. Längle und W. Karl. "High-throughput sensor-based sorting via approximate computing". *Forum Bildverarbeitung*. Karlsruhe, 2016, S. 99–110. DOI: 10.5445/KSP/1000059899.

[Mat75] B. Matthews. "Comparison of the predicted and observed secondary structure of T4 phage lysozyme". *Biochimica et Biophysica Acta (BBA) - Protein Structure* 405.2 (Okt. 1975), S. 442–451. DOI: 10.1016/0005-2795(75)90109-9.

[Maz96] V. Maz'ya. "On approximate approximations using Gaussian kernels". *IMA Journal of Numerical Analysis* 16.1 (Jan. 1996), S. 13–29. DOI: 10.1093/imanum/16.1.13.

[McD82] J. McDermott. "R1: A rule-based configurer of computer systems". *Artificial Intelligence* 19.1 (1982), S. 39–88. DOI: 10.1016/0004-3702(82)90021-2.

[Mer09] J. Mercer. "Functions of Positive and Negative Type, and their Connection with the Theory of Integral Equations". *Philosophical Transactions of the Royal Society of London* 209.1909 (1909), S. 415–446.

[Mit98] M. Mitchell. *An introduction to genetic algorithms*. 1998.

[MKG15] V. Mohammadi, K. Kheiralipour und M. Ghasemi-Varnamkhasti. "Detecting maturity of persimmon fruit based on image processing technique". *Scientia Horticulturae* 184 (März 2015), S. 123–128. DOI: 10.1016/j.scienta.2014.12.037.

[Mor+03] M. Morita, R. Sabourin, F. Bortolozzi und C. Suen. “Unsupervised feature selection using multi-objective genetic algorithms for handwritten word recognition”. *International Conference on Document Analysis and Recognition*. Bd. 1. IEEE Comput. Soc, 2003, S. 666–670. DOI: 10.1109/ICDAR.2003.1227746.

[MS11] I. Mukherjee und R. E. Schapire. “A theory of multiclass boosting”. *Journal of Machine Learning Research (JMLR)* 14.1 (2011), S. 437–497. DOI: citeulike-article-id:8467733.

[NJ01] A. Y. Ng und M. I. Jordan. “On Discriminative vs. Generative classifiers: A comparison of logistic regression and naive Bayes”. *Advances in Neural Information Processing Systems (NIPS)*. 2001, S. 841–848.

[NNY12] S. Nakauchi, K. Nishino und T. Yamashita. “Selection of optimal combinations of band-pass filters for ice detection by hyperspectral imaging”. *Optics Express* 20.2 (Jan. 2012), S. 986. DOI: 10.1364/OE.20.000986.

[NW06] J. Nocedal und S. J. Wright. *Numerical optimization*. 2. Aufl. New York, NY: Springer, 2006.

[OF97] B. A. Olshausen und D. J. Field. “Sparse coding with an overcomplete basis set: A strategy employed by V1?” *Vision Research* 37.23 (Dez. 1997), S. 3311–3325. DOI: 10.1016/S0042-6989(97)00169-7.

[OFL07] M. Ozuysal, P. Fua und V. Lepetit. “Fast Keypoint Recognition in Ten Lines of Code”. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Bd. 32. 3. IEEE, Juni 2007, S. 1–8. DOI: 10.1109/CVPR.2007.383123.

[Oja82] E. Oja. “Simplified neuron model as a principal component analyzer”. *Journal of Mathematical Biology* 15.3 (Nov. 1982), S. 267–273. DOI: 10.1007/BF00275687.

[OPM02] T. Ojala, M. Pietikainen und T. Maenpaa. “Multiresolution gray-scale and rotation invariant texture classification with local binary patterns”. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 24.7 (Juli 2002), S. 971–987. DOI: 10.1109/TPAMI.2002.1017623.

[Oqu+14] M. Oquab, L. Bottou, I. Laptev und J. Sivic. “Learning and Transferring Mid-level Image Representations Using Convolutional Neural Networks”. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, Juni 2014, S. 1717–1724. DOI: 10.1109/CVPR.2014.222.

[Osb+97] S. D. Osborne, R. Künnemeyer, S. D. Osborne und R. B. Jordan. “Method of Wavelength Selection for Partial Least Squares”. *The Analyst* 122.12 (1997), S. 1531–1537. DOI: 10.1039/a703235h.

[Ozu+10] M. Ozuysal, M. Calonder, V. Lepetit und P. Fua. “Fast keypoint recognition using random ferns.” *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 32.3 (März 2010), S. 448–61. DOI: 10.1109/TPAMI.2009.23.

[Pac+02] P. Paclík, R. P. W. Duin, G. M. P. van Kempen und R. Kohlus. “On Feature Selection with Measurement Cost and Grouped Features”. *IAPR Workshop on Structural, Syntactic, and Statistical Pattern Recognition*. Hrsg. von T. Caelli, A. Amin, R. P. W. Duin, D. de Ridder und M. Kamel. Berlin, Heidelberg: Springer, 2002, S. 461–469. DOI: 10.1007/3-540-70659-3_48.

[Pal09] M. Pal. “Margin-based feature selection for hyperspectral data”. *International Journal of Applied Earth Observation and*

Geoinformation 11.3 (Juni 2009), S. 212–220. DOI: 10.1016/j.jag.2009.02.001.

[Pal12] M. Pal. "Multinomial logistic regression-based feature selection for hyperspectral data". *International Journal of Applied Earth Observation and Geoinformation* 14.1 (Feb. 2012), S. 214–220. DOI: 10.1016/j.jag.2011.09.014.

[Par62] E. Parzen. "On estimation of a probability density function and mode". *The Annals of Mathematical Statistics* 33.3 (1962), S. 1065–1076.

[PD07] F. Perronnin und C. Dance. "Fisher Kernels on Visual Vocabularies for Image Categorization". *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, Juni 2007, S. 1–8. DOI: 10.1109/CVPR.2007.383266.

[Ped+11] F. Pedregosa et al. "Scikit-learn: Machine Learning in {P}ython". *Journal of Machine Learning Research (JMLR)* 12 (2011), S. 2825–2830.

[Phi+08] J. Philbin, O. Chum, M. Isard, J. Sivic und A. Zisserman. "Lost in quantization: Improving particular object retrieval in large scale image databases". *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, Juni 2008, S. 1–8. DOI: 10.1109/CVPR.2008.4587635.

[Pir+08] A. Piron, V. Leemans, O. Kleynen, F. Lebeau und M.-F. Destain. "Selection of the most efficient wavelength bands for discriminating weeds from crop". *Computers and Electronics in Agriculture* 62.2 (Juli 2008), S. 141–148. DOI: 10.1016/j.compag.2007.12.007.

[PK09] J. H. Plasberg und W. B. Kleijn. "Feature selection under a complexity constraint". *IEEE Transactions on Multimedia* 11.3 (2009), S. 565–571. DOI: 10.1109/TMM.2009.2012944.

[Pla98] J. C. Platt. “Fast training of support vector machines using sequential minimal optimization”. *Advances in Kernel Methods: Support Vector Learning*. Hrsg. von B. Schölkopf, C. J. C. Burges und A. J. Smola. The MIT Press, 1998, S. 41–65.

[PLD05] H. Peng, F. Long und C. Ding. “Feature Selection Based on Mutual Information: Criteria of Max-Dependency, Max-Relevance, and Min-Redundancy”. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 27.8 (2005), S. 1226–1238.

[PSM10] F. Perronnin, J. Sánchez und T. Mensink. “Improving the Fisher kernel for large-scale image classification”. *European Conference on Computer Vision (ECCV)*. Bd. 6314. Berlin, Heidelberg: Springer, 2010, S. 143–156. DOI: 10.1007/978-3-642-15561-1.

[PY10] S. J. Pan und Q. Yang. “A Survey on Transfer Learning”. *IEEE Transactions on Knowledge and Data Engineering* 22.10 (Okt. 2010), S. 1345–1359. DOI: 10.1109/TKDE.2009.191.

[RB14a] M. Richter und J. Beyerer. “Optical filter selection for automatic visual inspection”. *IEEE Winter Conference on Applications of Computer Vision (WACV)*. 2014, S. 123–128. DOI: 10.1109/WACV.2014.6836110.

[RB14b] M. Richter und J. Beyerer. “Parameter-learning for color sorting of bulk materials using genetic algorithms”. *Forum Bildverarbeitung*. 2014, S. 107–118.

[Ret+16] M.-G. Retzlaff, M. Richter, T. Längle, J. Beyerer und C. Dachsbacher. “Combining synthetic image acquisition and machine learning : Accelerated design and deployment of sorting systems”. *Forum Bildverarbeitung*. January 2017. 2016, S. 49–61.

[RG11] P. Ram und A. G. Gray. “Density estimation trees”. *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. ACM. New York, New York, USA: ACM Press, 2011, S. 627. DOI: 10.1145/2020408.2020507.

[RHE14] M. Richter, Hua Gao und H. K. Ekenel. “Extending explicit shape regression with mixed feature channels and pose priors”. *IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, März 2014, S. 1013–1019. DOI: 10.1109/WACV.2014.6835993.

[RHW86] D. E. Rumelhart, G. E. Hinton und R. J. Williams. “Learning representations by back-propagating errors”. *Nature* 323.6088 (Okt. 1986), S. 533–536. DOI: 10.1038/323533a0.

[Ric+16a] M. Richter, G. Maier, R. Gruna, T. Längle und J. Beyerer. “Feature Selection with a Budget”. *World Congress on Electrical Engineering and Computer Systems and Science (EECSS)*. Juli 2016, S. 104.1–104.8. DOI: 10.11159/mvml16.104.

[Ric+16b] M. Richter, K.-U. Vieth, T. Längle und J. Beyerer. “Bag of visual words—A computer vision method applied to bulk material sorting”. *Sensor-Based Sorting & Control*. 2016, S. 169–177.

[RLB15a] M. Richter, T. Längle und J. Beyerer. “An approach to color-based sorting of bulk materials with automated estimation of system parameters”. *tm – Technisches Messen* 82.3 (Jan. 2015), S. 135–144. DOI: 10.1515/teme-2014-0042.

[RLB15b] M. Richter, T. Längle und J. Beyerer. “Large scale classification of spectral signatures”. *tm – Technisches Messen* 82.12 (Jan. 2015), S. 663–671. DOI: 10.1515/teme-2015-0040.

[RLB15c] M. Richter, T. Längle und J. Beyerer. “Visual words for automated visual inspection of bulk materials”. *IAPR International Conference on Machine Vision Applications (MVA)*. IEEE, Mai 2015, S. 210–213. DOI: 10.1109/MVA.2015.7153169.

[RLB16] M. Richter, T. Längle und J. Beyerer. “Knowing when you don’t: Bag of visual words with reject option for automatic visual inspection of bulk materials”. *International Conference on Pattern Recognition (ICPR)*. IEEE, Dez. 2016, S. 3079–3084. DOI: 10.1109/ICPR.2016.7900107.

[RLB17] M. Richter, T. Längle und J. Beyerer. “Gaussian Mixture Trees for One Class Classification in Automated Visual Inspection”. *International Conference on Image Analysis and Recognition (ICIAR)*. 2017, S. 341–351. DOI: 10.1007/978-3-319-59876-5_38.

[Röd13] J. Röder. “Active Learning: New Approaches, and Industrial Applications”. Diss. Ruprecht-Karls-Universität Heidelberg, 2013.

[Ros61] F. Rosenblatt. *Principles of neurodynamics: Perceptrons and the Theory of Brain Mechanisms*. Techn. Ber. CORNELL AERONAUTICAL LAB INC BUFFALO NY, 1961.

[Roy83] J. P. Royston. “Some Techniques for Assessing Multivarate Normality Based on the Shapiro- Wilk W”. *Applied Statistics* 32.2 (1983), S. 121. DOI: 10.2307/2347291.

[RV13] N. Rasiwasia und N. Vasconcelos. “Latent Dirichlet Allocation Models for Image Classification”. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 35.11 (Nov. 2013), S. 2665–2679. DOI: 10.1109/TPAMI.2013.69.

[Sam59] A. L. Samuel. “Some studies in machine learning using the game of checkers”. *IBM Journal of Research and Development* 3.3 (1959), S. 210–229. DOI: 10.1147/rd.33.0210.

[SB91] M. J. Swain und D. H. Ballard. “Color indexing”. *International Journal of Computer Vision* 7.1 (1991), S. 11–32. DOI: 10.1007/BF00130487.

[SB98] R. S. Sutton und A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

[Sch+01] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola und R. C. Williamson. “Estimating the Support of a High-Dimensional Distribution”. *Neural Computation* 13.7 (Juli 2001), S. 1443–1471. DOI: 10.1162/089976601750264965.

[Sch+15] H. Schulte, G. Brink, R. Gruna, R. Herzog und H. Grüger. “Utilization of Spectral Signatures of Food for Daily Use”. *Optical Characterization of Materials*. 2015, S. 1–18.

[Sch15] J. Schmidhuber. “Deep Learning in neural networks: An overview”. *Neural Networks* 61 (2015), S. 85–117. DOI: 10.1016/j.neunet.2014.09.003.

[Set12] B. Settles. “Active Learning”. *Synthesis Lectures on Artificial Intelligence and Machine Learning* 6.1 (Juni 2012), S. 1–114. DOI: 10.2200/S00429ED1V01Y201207AIM018.

[SGM73] J. W. Schoonahd, J. D. Gould und L. A. Miller. “Studies of Visual Inspection”. *Ergonomics* 16.4 (1973), S. 365–379. DOI: 10.1080/00140137308924528.

[Sha08] A. Shahbahrami. “Comparison Between Color and Texture Features for Image Retrieval”. *International Symposium on Applied Machine Intelligence and Informatics* 27648 (2008), S. 221–224.

[Sho74] E. H. Shortliffe. “A Rule-based Computer Program for Advising Physicians Regarding Antimicrobial Therapy Selection”. *Annual ACM conference*. June. 1974, S. 739–739. DOI: 10.1145/1408800.1408906.

[Sil+16] D. Silver et al. “Mastering the game of Go with deep neural networks and tree search”. *Nature* 529.7587 (Jan. 2016), S. 484–489. DOI: 10.1038/nature16961.

[Sil+17] D. Silver et al. “Mastering the game of Go without human knowledge”. *Nature* 550.7676 (Okt. 2017), S. 354–359. DOI: 10.1038/nature24270.

[Siv06] J. Sivic. “Efficient visual search of images videos”. *Toward Category-Level Object Recognition*. Springer, 2006, S. 199. DOI: 10.1007/11957959_7.

[SJ14] Saroj und Jyoti. “Multi-objective genetic algorithm approach to feature subset optimization”. *IEEE International Advance Computing Conference (IACC)*. IEEE, Feb. 2014, S. 544–548. DOI: 10.1109/IAdCC.2014.6779383.

[SK87] L. Sirovich und M. Kirby. “Low-dimensional procedure for the characterization of human faces”. *Journal of the Optical Society of America* 4.3 (1987), S. 519–524. DOI: 10.1364/JOSAA.4.000519.

[Son+17] L. Song, S. Vempala, J. Wilmes und B. Xie. “On the Complexity of Learning Neural Networks”. *arXiv preprint* (Juli 2017), S. 1–21.

[SS01] B. Schölkopf und A. J. Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2001.

[SS05] J. Schäfer und K. Strimmer. “A Shrinkage Approach to Large-Scale Covariance Matrix Estimation and Implications for Functional Genomics”. *Statistical Applications in Genetics and Molecular Biology* 4.1 (Jan. 2005). DOI: 10.2202/1544-6115.1175.

[SS09] J. Sun und Q. Sun. “A Support Vector Machine Based Online Learning Approach for Automated Visual Inspection”. *Canadian Conference on Computer and Robot Vision*. IEEE, Mai 2009, S. 192–199. DOI: 10.1109/CRV.2009.13.

[SSM98] B. Schölkopf, A. Smola und K.-R. Müller. “Nonlinear Component Analysis as a Kernel Eigenvalue Problem”. *Neural Computation* 10.5 (Juli 1998), S. 1299–1319. DOI: 10.1162/089976698300017467.

[Ste46] S. S. Stevens. “On the theory of scales of measurement”. *Science* 103.2684 (1946), S. 667–680. DOI: 10.1016/0022-2496(81)90045-6.

[SUW08] C. Steger, M. Ulrich und C. Wiedemann. *Machine Vision Algorithms and Applications*. Wiley-VCH Verlag GmbH & Co. KGaA, 2008.

[Tap15] M. Taphanel. “Chromatisch konfokale Triangulation - Hochgeschwindigkeits 3D-Sensorik auf Basis der Wellenlängenschätzung mit optimierten Filtern”. Diss. 2015. DOI: 10.5445/IR/1000048739.

[TB97] Q. M. Tieng und W. W. Boles. “Recognition of 2D object contours using the wavelet transform zero-crossing representation”. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 19.8 (1997), S. 910–916. DOI: 10.1109/34.608294.

[TC88] C.-H. Teh und R. Chin. “On image analysis by the methods of moments”. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 10.4 (Juli 1988), S. 496–513. DOI: 10.1109/34.3913.

[TD99] D. M. Tax und R. P. Duin. “Support vector domain description”. *Pattern Recognition Letters* 20.11-13 (Nov. 1999), S. 1191–1199. DOI: 10.1016/S0167-8655(99)00087-2.

[Tib96] R. Tibshirani. “Regression Shrinkage and Selection via the Lasso”. *Journal of the Royal Statistical Society* 58.1 (1996), S. 267–288.

[Til15] A. M. Tillmann. “On the Computational Intractability of Exact and Approximate Dictionary Learning”. *IEEE Signal Processing Letters* 22.1 (Jan. 2015), S. 45–49. DOI: 10.1109/LSP.2014.2345761.

[TSD09] U. Tekguc, H. Soyel und H. Demirel. “Feature selection for person-independent 3D facial expression recognition using NSGA-II”. *International Symposium on Computer and Information Sciences*. 2009, S. 35–38.

[TSF10] L. Torresani, M. Szummer und A. Fitzgibbon. “Efficient object category recognition using classemes”. *European Conference on Computer Vision (ECCV)*. Springer, Sep. 2010, S. 776–789.

[TTA13] Y. Tanaka, T. Takiguchi und Y. Ariki. “Unknown Object Identification Using Category Visual Words with Rejection Function”. *IAPR International Conference on Machine Vision Applications (MVA)*. 2013, S. 375–378.

[Tur50] A. M. Turing. “Computing machinery and intelligence”. *Mind* 59.236 (1950), S. 433–460.

[TZ17] E. Togban und D. Ziou. “Classification Using Mixture of Discriminative Learners: The Case of Compositional Data”. Hrsg. von F. Karray, A. Campilho und F. Cheriet. Bd. 10317. Lecture Notes in Computer Science. Cham: Springer, 2017, S. 416–425. DOI: 10.1007/978-3-319-59876-5_46.

[Vap95] V. N. Vapnik. *The Nature of Statistical Learning Theory*. New York, NY: Springer, 1995, S. 188. DOI: 10.1007/978-1-4757-2440-0.

[VJ01] P. Viola und M. J. Jones. “Robust Real-time Object Detection”. *Cambridge Research Laboratory - Technical Report Series* (Feb. 2001).

[VL98] N. Vasconcelos und A. Lippman. “Learning mixture hierarchies”. *Advances in Neural Information Processing Systems (NIPS)*. Bd. pp. 1998, S. 606–613.

[WCM05] J. Winn, A. Criminisi und T. Minka. “Object categorization by learned universal visual dictionary”. *IEEE International Conference on Computer Vision (ICCV)*. Bd. II. IEEE, 2005, 1800–1807 Vol. 2. DOI: 10.1109/ICCV.2005.171.

[WH10] H. Wotruba und H. Harbeck. “Sensor-based sorting”. *Ullmann’s Encyclopedia of Industrial Chemistry* (2010). DOI: 10.1002/14356007.b02.

[WHH05] N. A. Weiss, P. T. Holmes und M. Hardy. *A Course in Probability*. Pearson Addison Wesley, 2005.

[Wil99] C. K. I. Williams. “A MCMC Approach to Hierarchical Mixture Modelling.” *Advances in Neural Information Processing Systems (NIPS)*. 1999, S. 680–686.

[Wol96] D. H. Wolpert. “The Lack of A Priori Distinctions Between Learning Algorithms”. *Neural Computation* 8.7 (1996), S. 1341–1390. DOI: 10.1162/neco.1996.8.7.1341.

[Xin99] Xin Yao. “Evolving artificial neural networks”. *Proceedings of the IEEE* 87.9 (1999), S. 1423–1447. DOI: 10.1109/5.784219.

[XZB13] B. Xue, M. J. Zhang und W. N. Browne. “Particle swarm optimization for feature selection in classification: a multi-objective approach”. *IEEE Transactions on Cybernetics* 43.6 (2013), S. 1656–71. DOI: 10.1109/TSMCB.2012.2227469.

[YH98] J. Yang und V. Honavar. “Feature Subset Selection Using a Genetic Algorithm”. *Intelligent Systems and their Applications* 13.2 (1998), S. 44–49. DOI: 10.1109/5254.671091.

[YM99] H. H. Yang und J. Moody. “Feature Selection Based on Joint Mutual Information”. *ICSC Symposium on Advances in Intelligent Data Analysis*. 1999, S. 22–25.

[YW10] M. Yuan und M. Wegkamp. “Classification methods with reject option based on convex risk minimization”. *Journal of Machine Learning Research (JMLR)* 11.1 (2010), S. 111–130.

[ZB17] J.-J. Zhu und J. Bento. “Generative Adversarial Active Learning”. *arXiv preprint* (Feb. 2017), S. 1–11.

[Zei12] M. D. Zeiler. “ADADELTA: An Adaptive Learning Rate Method”. *arXiv preprint* (2012), S. 6.

[ZF13] M. D. Zeiler und R. Fergus. “Visualizing and Understanding Convolutional Networks”. *Lecture Notes in Computer Science* 8689 LNCS.PART 1 (Nov. 2013), S. 818–833. DOI: 10.1007/978-3-319-10590-1_53.

[ZG15] Q. Zhang und B. Goncalves. “Why should I trust you? Explaining the predictions of any classifier”. *arXiv preprint* (2015), S. 4503. DOI: 10.1145/1235.

[Zha+14a] B. Zhang, W. Huang, J. Li, C. Zhao, S. Fan, J. Wu und C. Liu. “Principles, developments and applications of computer vision for external quality inspection of fruits and vegetables: A review”. *Food Research International* 62 (Aug. 2014), S. 326–343. DOI: 10.1016/j.foodres.2014.03.012.

[Zha+14b] D. Zhang, D.-J. Lee, B. J. Tippetts und K. D. Lillywhite. “Date maturity and quality evaluation using color distribution analysis and back projection”. *Journal of Food Engineering* 131 (Juni 2014), S. 161–169. DOI: 10.1016/j.jfoodeng.2014.02.002.

[Zim14] D. Zimmerer. “Supervised Learning of Human-Interpretable Image-Features for Bulk-Good-Sorting”. Bachelorthesis. Karlsruher Institut für Technologie, 2014.

[ZL02] D. Zhang und G. Lu. “Shape-based image retrieval using generic Fourier descriptor”. *Signal Processing: Image Communication* 17.10 (Nov. 2002), S. 825–848. DOI: 10.1016/S0923-5965(02)00084-X.

[ZL04] D. Zhang und G. Lu. “Review of shape representation and description techniques”. *Pattern Recognition* 37.1 (Jan. 2004), S. 1–19. DOI: 10.1016/j.patcog.2003.07.008.

[ZLZ13] F. Zhao, H. zhang Lu und Z. yong Zhang. “Real-time single-pass connected components analysis algorithm”. *EURASIP Journal on Image and Video Processing* 2013.1 (Dez. 2013), S. 21. DOI: 10.1186/1687-5281-2013-21.

[ZW05] F. Zheng und G. Webb. “A Comparative Study of Semi-naive Bayes Methods in Classification Learning”. *Australasian Data Mining Workshop*. 2005, S. 141–156.