Bayessches Transferlernen für die Semantische Segmentierung von Luftbildern

Der Fakultät für Elektrotechnik und Informatik

der Gottfried Wilhelm Leibniz Universität Hannover

zur Erlangung des akademischen Grades

Doktor-Ingenieur

genehmigte

Dissertation

von

Karsten Vogt, M.Sc.

geboren am 15. November 1982 in Hannover.

2019

Referent	Prof. Dr.-Ing. Jörn Ostermann
Korreferent	apl. Prof. Dr. techn. Franz Rottensteiner
Tag der Promotion	17.08.2018

Kurzfassung

Diese Arbeit beschäftigt sich mit dem Problem der semantischen Segmentierung von Luftbildern in Landbedeckungsklassen. Maschinelle Lernverfahren bieten dabei sehr robuste Methoden zur Erzeugung von hochgenauen Klassifikatoren, welche die Basisbausteine von Verfahren zur Erzeugung solcher Segmentierungen darstellen. Moderne und praktikable Lernverfahren sind jedoch nahezu ausschließlich den überwachten Methoden zuzuschreiben. Ihr Wissen über das zu lösende Klassifikationsproblem wird aus gelabelten Stichproben extrahiert, welche zuerst in mühevoller manueller Arbeit von qualifizierten Fachkräften aus den Eingangsbildern erzeugt werden müssen. Die Reduktion dieses manuellen Aufwands wurde daher in vielen Arbeiten als Kernproblem in Angriff genommen, woraus sich sehr unterschiedliche und heutzutage essentielle Teilgebiete des maschinellen Lernens entwickelt haben, wie u. a. dem Transferlernen. Im Transferlernen wird das Anlernen des Klassifikators durch andere, bereits vorhandene Datenquellen augmentiert. Insbesondere gilt für die *Domänenadaption*, dass das Eingangsbild ungelabelt ist und Klasseninformation lediglich aus diesen Datenquellen abgeleitet werden kann. Dieses Vorgehen kann jedoch nur funktionieren, falls das Eingangsbild und die Datenquellen große Ähnlichkeiten aufweisen und in einer modellierbaren Relation zueinander stehen. Die ansonsten resultierende Verschlechterung der Klassifikation heißt dann *negativer Transfer*.

In dieser Arbeit wird ein bestehendes theoretisches Modell zur Prädiktion von negativen Transfer für die Domänenadaption in ein Bayessches Entscheidungsmodell umformuliert. Dessen Implementierung durch ein Markov-Chain-Monte-Carlo Verfahren ist mit einer hohen Wahrscheinlichkeit dazu in der Lage, aus einer Vielzahl von potentiellen Quellenkandidaten die beste Quelle für ein gegebenes Eingangsbild auszuwählen. Eine anschließend hergeleitete Approximation dieser Methode besitzt eine lineare Laufzeitkomplexität bezüglich der Stichprobengröße und der Anzahl der Quellenkandidaten und erlaubt somit die Verarbeitung von sehr großen Datensätzen mit mehr als 15.000 Lernbeispielen pro Quelle und mehreren Dutzend Quellenkandidaten in weniger als 10 Sekunden. Weiterhin konnte der Speicherverbrauch ohne messbare Einbußen auf wenige Kilobyte pro Quelle reduziert werden. Auf 7 untersuchten Datensätzen reduzierte die Bayessche Quellenselektion den Medianverlust der Klassifikationsgenauigkeit im Vergleich mit konventionellen Ansätzen im Mittel von 5% auf 3%. Das entwickelte Verfahren besitzt keine kritischen Parameter. Eine aufwändige Parametersuche entfällt somit im direkten Vergleich zu Konkurrenzmethoden.

In einem weiteren Experiment wurden zunächst ungelabelte Quellenkandidaten verwendet. Durch einen neuartigen Clustering Ansatz, basierend auf dem zuvor entwickelten Quellenselektionsmodell, konnten aus diesen die relevanten Quellen zuverlässig ermittelt werden. Ein manuelles labeln von redundanten Quellen entfällt somit. Auf den untersuchten Datensätzen konnte bei gleichbleibender Klassifikationsgenauigkeit eine Reduktion des Arbeitsaufwandes um bis zu 90% erreicht werden.

Schlagwörter: Maschinelles Lernen, Transferlernen, Domänenadaption

Abstract

This thesis deals with the problem of the semantic segmentation of aerial imagery into land cover classes. Machine learning methods provide very robust methods for generating highly-accurate classifiers which are the basic building blocks of methods for generating such segmentations. However, modern and practical learning methods are almost exclusively based on supervised learning. Their required knowledge of the classification problem to be solved is extracted from labeled samples, which must first be produced in laborious manual work by qualified specialists from the input images. The reduction of this manual effort has therefore been tackled in countless publications as a core problem. Many different and nowadays essential areas of machine learning have spawned from these efforts, such as transfer learning. In transfer learning the learning of the classifier is augmented by other existing data sources. In particular, for *domain adaptation* the input image is unlabeled and class information is only derived from these data sources. However, this procedure will only work if the input image and the data sources have great similarities and are in a modelable relation to each other. The otherwise observed deterioration of the classification performance is called *negative transfer*.

In this thesis, an existing theoretical model for the estimation of negative transfer in domain adaptation is reformulated as a Bayesian decision model. This model, implemented via a Markov-chain-monte-carlo method, is able to select the best source from a variety of potential source candidates with a very high probability of success. A subsequent approximation of this method has a linear runtime complexity in terms of sample sizes and number of source candidates, allowing the processing of very large datasets with more than 15,000 examples per source and tens of source candidates in less than 10 seconds. Furthermore, the memory consumption could be reduced to a few kilobytes per source without any measurable losses in quality. Bayesian source selection reduced the median loss of classification accuracy on average from 5% to 3% compared to conventional approaches on 7 datasets. The developed method has no critical parameters. A complex parameter search can thus be avoided in direct comparison to competing methods.

In another experiment the source candidates were initially unlabeled. By introducing a novel clustering approach based on the previously developed source selection model the relevant sources could reliably be determined. Manual labeling of redundant sources could therefore be eliminated. Thus, a reduction of the workload of up to 90% could be achieved on the examined data sets while achieving comparable classification accuracy.

Keywords: Machine Learning, Transfer Learning, Domain Adaptation

INHALTSVERZEICHNIS

ABKÜRZUNGSVERZEICHNIS

AdaBoost	Adaptive Boosting
ADS	Adaptive Direction Sampling
AIC	Akaike Information Criterion
BAGGING	Bootstrap Aggregierung
BIC	Bayesian Information Criterion
BPM	Bayes-Point-Machine
CART	Classification and Regression Trees
DOM	digitales Oberflächenmodell
DGM	digitales Geländemodell
INN	Iterative Nearest Neighbor
KLD	Kullback Leibler Divergence
LBP	Local-Binary-Patterns
LS	Least Squares
MAP	Maximum a-Posteriori
MCMC	Markov-Chain-Monte-Carlo
MH	Metropolis-Hastings
ML	Maximum Likelihood
MMD	Maximum Mean Discrepancy
NDVI	Normalized-Difference-Vegetation-Index
NN	Nächster-Nachbar
oVo	One-vs-One
oVr	One-vs-Rest
PCA	Hauptkomponentenanalyse
RBF	Radial-Basis-Function
SVM	Support-Vector-Machine

TV	Total-Variation
VC	Vapnik-Chervonenkis

SYMBOLVERZEICHNIS

Generelle Symbole

a	Skalare Variable
$\boldsymbol{a}$	Vektorwertige Variable
$\langle \boldsymbol{a}, \boldsymbol{b} \rangle$	Skalarprodukt
$f(a) \propto g(a)$	Die Funktion $f(a)$ ist proportional zu $g(a)$ falls gilt $\exists c(f(a) = c \cdot g(a))$
$\mathbb{I}(a)$	Indikatorfunktion: $\mathbb{I}(a) = 1$ falls a, sonst 0
$\Gamma(a)$	Eulersche Gammafunktion
I	Einheitsmatrix
$d_2(\boldsymbol{x}_1, \boldsymbol{x}_2)$	Euklidische Distanz

Wahrscheinlichkeits- und Entscheidungstheorie

$p(A)$	Wahrscheinlichkeit des Ereignisses A
$p(A\|I)$	Bedingte (bzw. konditionierte) Wahrscheinlichkeit des Ereignisses A nach Beobachtung der Information I
$E[a]$	Der Erwartungswert der Zufallsvariablen a
$E[a\|I]$	Der bedingte (bzw. konditionierte) Erwartungswert der Zufallsvariablen a nach Beobachtung der Information I
μ	Mittelwert einer Wahrscheinlichkeitsverteilung
σ^2	Varianz einer Wahrscheinlichkeitsverteilung
Σ	Kovarianzmatrix einer Wahrscheinlichkeitsverteilung
$\mathcal{N}(\mu, \sigma^2)$	Normalverteilung mit Mittelwert μ und Varianz σ^2

$Unif(a, b)$	Gleichverteilung im Intervall $[a, b]$
$\boldsymbol{\theta}$	Parametervektor
$\bar{\boldsymbol{\theta}}$	Geschätzter Parametervektor bezüglich einer Punktschätzung
$\boldsymbol{\varpi}$	Vektor der Wahrscheinlichkeiten der Elementarereignisse einer diskreten Wahrscheinlichkeitsverteilung
$\boldsymbol{\alpha}$	Parameter einer Dirichlet Verteilung
r	Wahlmöglichkeit
$\mathcal{R}$	Menge der Wahlmöglichkeiten
$u(r)$	*Utility* der Wahlmöglichkeit r
$L(r)$	Kosten, welche durch Wahl der Wahlmöglichkeit r entstehen

Maschinelles Lernen

$\boldsymbol{x}$	Merkmalsvektor
$\mathcal{F}$	Merkmalsraum
D	Dimensionalität des Merkmalsraums
y	Klassenlabel
$\mathcal{K}$	Menge der Klassenlabel
T	Stichprobe
Tr	Lernstichprobe
Te	Teststichprobe
N	Stichprobengröße
$\boldsymbol{x} \sim p(\boldsymbol{x})$	Ziehen eines Werts aus einer Wahrscheinlichkeitsverteilung
$\mathrm{T} \sim p(\boldsymbol{x})$	Ziehen einer statistisch unabhängigen Stichprobe aus einer Wahrscheinlichkeitsverteilung mit Zurücklegen
$y = h(\boldsymbol{x})$	Entscheidungsfunktion
$\boldsymbol{\varphi}$	Parametrisiertes Partitionierungs-/Entscheidungsmodell
$\mathcal{P}$	Menge der Partitionierungs-/Entscheidungsmodelle

P	Anzahl der Parameter in einem Partitionierungs-/Entscheidungsmodell
$(\boldsymbol{\nu}, b)$	Lineare Trennebene in Hessescher Normalform

Kapitel 1

Einleitung

1.1 Motivation und Problemstellung

Ende der Neunzigerjahre haben rapide Fortschritte in der Mustererkennung sowohl in Theorie als auch in der Methodik das Feld für breitere und komplexere Aufgabenstellungen geöffnet. Dies kann neuen Ansätzen, wie dem *Boosting* [45], *Boostrap Aggregation* (z. B. *Random Forest* [14]), Kernel Methoden (z. B. Support-Vector-Machine (SVM) [113]) und den später aufkommenden tiefen Netzarchitekturen für künstliche neuronale Netze [70] zugeschrieben werden. Eines dieser Anwendungsgebiete ist zum Beispiel die Analyse natürlicher Sprache (*Natural Language Processing* [81]). Verbesserungen in diesem Gebiet haben unter anderem zur Proliferation sprachgesteuerter Assistenten geführt (Siri, Alexa, Cortana, etc.). Viele Anwendungen lassen sich jedoch auf die Analyse von visuell interpretierbaren Daten zurückführen. Zu diesen Anwendungen zählen insbesondere Fahrerassistenzsysteme [104], unterstützende medizinische Diagnostik [56, 63], Biometrische Sicherheitssysteme [34, 136], die Inspektion von Rohrleitungssystemen [69] und die Fernerkundung [60, 137]. Beispiele zu einigen dieser Anwendungsgebiete sind in Abbildung 1.1 aufgeführt.

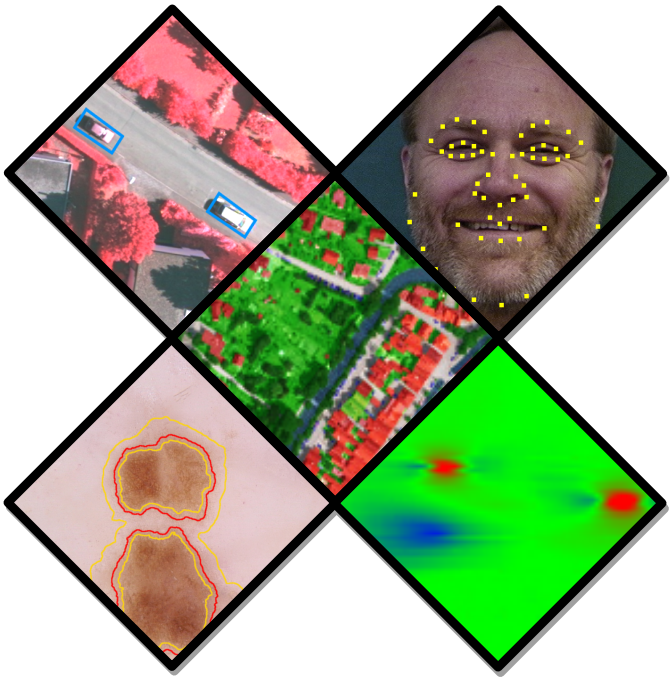


Abbildung 1.1: Anwendungsbeispiele für das maschinelle Lernen. *Mitte*: Klassifikation von Landbedeckungsklassen. *Links-Oben*: Erkennung und Verfolgung von Fahrzeugen. *Rechts-Oben*: Landmarkendetektion in Gesichtern. *Links-Unten*: Segmentierung von Hautkrebs. *Rechts-Unten*: Detektion von Defektstellen in Rohrleitungssystemen.

In der Fernerkundung wurden die Möglichkeiten der computergestützten Bildanalyse und des maschinellen Lernens bereits sehr früh untersucht. Durch Verbesserung in Technik und durch Skaleneffekte eines stetig wachsenden Marktes für Fernerkundungsdaten wird das verfügbare Bildmaterial immer weitreichender, aktueller und kostengünstiger. Fortschritte in der Sensorik ermöglichen heutzutage nicht nur hochauflösende Aufnahmen im visuellen Spektrum und nahem Infrarot, sondern auch von hyperspektralen Daten, Radar und Oberflächenmodellen. Diese ständig wachsende Datenflut birgt viel Potential. Jedoch sind die Daten nahezu unüberschaubar und weitestgehend unstrukturiert und können daher nicht mehr manuell von menschlichen Operateuren verarbeitet werden. Am Institut für Informationsverarbeitung wurden daher zum Teil in Kooperation mit dem Institut für Photogrammetrie und GeoInformation der Leibniz Universität Hannover seit vielen Jahren Methoden für die (semi-)automatische Verarbeitung von hochauflösenden Luft- und Satellitenbildern entwickelt. Es wurden dabei unterschiedliche Anwendungen untersucht, wie die Verifikation von Geobasisdaten [60], Objektdetektion und -verfolgung [84] sowie der Detektion von Subpixelobjekten in Hyperspektralbildern [97]. Eine vitale Komponente in diesen Anwendungen ist die semantische Segmentierung von Luftbildern. Abbildung 1.2 zeigt ein Beispiel für eine solche Segmentierung bei der die Landbedeckungsklassen *Gebäude*, *Baum*, *Fahrzeug*, *niedrige Vegetation*, *versiegelte Oberfläche* und *Sonstiges* anhand einer Merkmalsrepräsentation erkannt werden. Der Algorithmus, welcher diese Zuordnung durchführt wird als Klassifikator bezeichnet. Der Klassifikator hat somit die Aufgabe, jedem Pixel der Szene anhand des zur Verfügung stehenden Bildmaterials die wahrscheinlichste Klasse zuzuweisen. In früheren Arbeiten konnten Klassifikatoren noch manuell erstellt werden und nahmen häufig die Form von einfachen Schwellwertentscheidungen oder kleinen Entscheidungsbäumen an.

Die großen Datenmengen und immer komplexeren Zusammenhänge haben allerdings dazu geführt, dass die Konstruktion von Klassifikatoren heutztage computergestützt durch einen Algorithmus durchgeführt wird. In Abbildung 1.3a wird ein Szenario für einen solchen Vorgang gezeigt. Im überwachten Lernen soll der Klassifikator automatisch aus einer kleinen zuvor manuell erzeugten Segmentierung, welche als Lernstichprobe bezeichnet wird, abgeleitet werden. Typischerweise wird zur Erstellung dieser Lernstichprobe nur ein kleiner Ausschnitt (im Folgenden *Patch* genannt) aus der gesamten Szene manuell gelabelt. Im überwachten Lernen muss daher die Annahme getroffen werden, dass dieser Patch bezüglich aller statistischen Eigenschaften, wie z. B. der relativen Klassenhäufigkeiten, Bebaaungsdichten und der photometrischen Eigenschaften die gesamte Szene repräsentiert. Im Beispiel 1.3a wurde als Lernstichprobe jedoch ein für diese Szene atypisch dicht bebautes Stadtgebiet gewählt. Die Klassifikation des gewerblich geprägten Gebiets in der unteren rechten Ecke kann daher nur mit Abstrichen in der erreichbaren Klassifikationsleistung erfolgen. Das Transferlernen beschäftigt sich mit Ansätzen, solche Unterschiede zwischen der Lernstichprobe und dem Eingangspatch zu erkennen

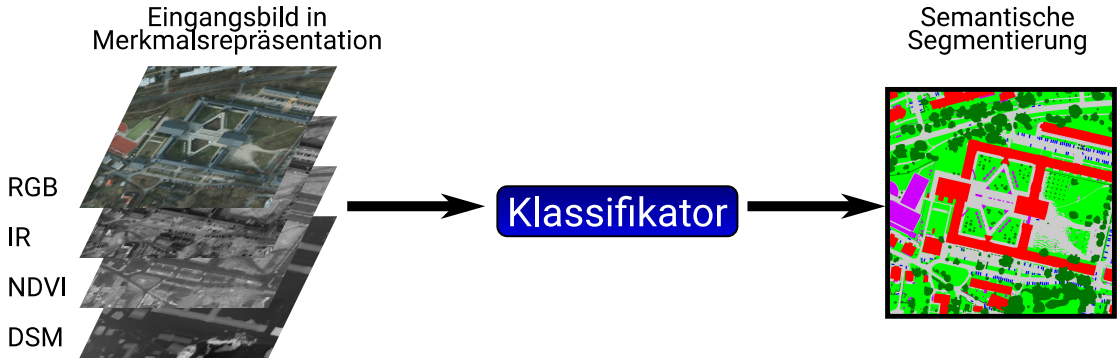
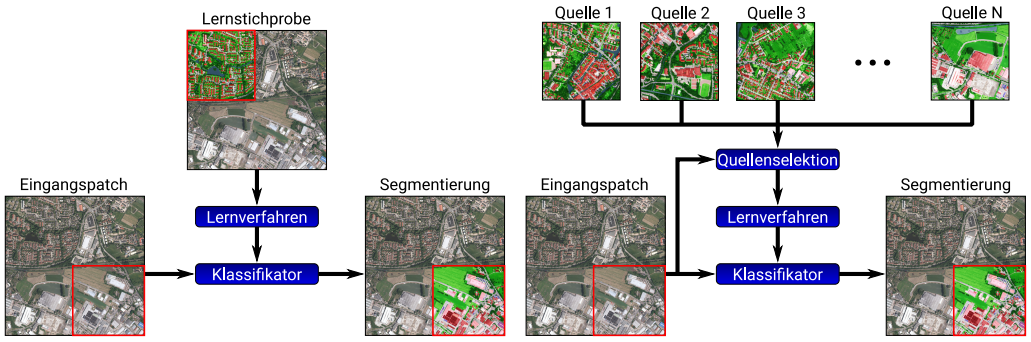


Abbildung 1.2: Semantische Segmentierung in Landbedeckungsklassen. Die Merkmalsrepräsentation besteht hier aus dem visuellen Spektrum (RGB), Infrarot (IR), dem *Normalized Difference Vegetation Index* (NDVI) und einem digitalen Oberflächenmodell (DSM).

und auszugleichen [92]. In Abbildung 1.3b wird ein Beispiel für ein Teilgebiet des Transferlernens gezeigt, der Quellenselektion. In diesem Beispiel existieren bereits N gelabelte Patches, welche nicht unbedingt aus derselben Szene wie der Eingangspatch stammen müssen. Es könnten somit auch Daten aus früheren Aufnahmen wiederverwendet werden, welche in räumlicher Nähe zum Eingangspatch liegen oder zu anderen Zeitpunkten, beziehungsweise unter anderen Aufnahmebedingungen, entstanden sind. Die Quellenselektion vergleicht die statistische Ähnlichkeit dieser Patches mit dem Eingangspatch um somit den besten Kandidaten zum Anlernen des Klassifikators zu bestimmen. Mit wachsender Datenbasis besteht somit die Hoffnung, dass durch diese Wiederverwendung von Trainingsdaten in Zukunft auf ein manuelles Labeln weitestgehend verzichtet werden kann. Obwohl sich diese Arbeit im wesentlichen mit der Quellenselektion beschäftigt, wird im nachfolgenden Abschnitt zunächst ein breiter Überblick den Stand der Forschung im Transferlernen näher bringen. Anschließend werden konkrete Forschungslücken aufgezeigt und die inhaltlichen Ziele dieser Arbeit abgesteckt.

1.2 Stand der Forschung

Das Gebiet des Transferlernens wird breits seit einigen Jahren intensiv erforscht. In früheren Arbeiten wurden zunächst Szenarien betrachtet, welche dem überwachten Lernen noch sehr nahe standen. Beim *Covariate Shift* wird zum Beispiel davon ausgegangen, dass die Lernstichprobe lediglich nichtrepräsentativ erzeugt wurde [125, 150]. Nichtrepräsentativ bedeutet in diesem Kontext, dass einzelne Klassen, bzw. Klassenausprägungen, in der Stichprobe über- oder unterrepräsentiert sind. Spätere Arbeiten haben unterschiedliche Generalisierungen des überwachten Lernens und



(a) Überwachtes Lernen mit einer gegebenen Lernstichprobe

(b) Überwachtes Lernen mit Quellenselektion

Abbildung 1.3: Darstellung der überwachten Lernszenarien. *(a)* Beim überwachten Lernen sind der Eingangspatch (rechts unten) und die Lernstichprobe (links oben) vorgegeben. Es wird davon ausgegangen, dass diese dieselben statistischen Eigenschaften aufweisen. In diesem Beispiel ist dies nicht gegeben, daher muss mit einer verminderten Klassifikationsleistung gerechnet werden. *(b)* Bei der Quellenselektion stehen N Lernstichprobenkandidaten zur Auswahl. Hiervon wird eine ausgewählt, welche dem Eingangspatch statistisch am ähnlichsten ist. Auf dieser Lernstichprobe wird nun überwachtes Lernen durchgeführt.

diverse Arten von Transfermodellen untersucht, welche im Folgenden kurz vorgestellt werden. Diese Arbeit orientiert sich bezüglich der Terminologie und der Klassifikation von Transfermodellen an dem Übersichtsartikel von Pan und Yang [92].

1.2.1 Lernszenarien

Multitask Lernen

In diesem Szenario sollen viele ähnliche Lernaufgaben parallel gelöst werden. Für jede Lernaufgabe wird ein separater Klassifikator erzeugt. Verbesserung gegenüber dem überwachten Lernen werden erreicht, indem zwischen ähnlichen Lernaufgaben Informationen ausgetauscht werden. Meistens geschieht dies durch eine gemeinschaftlich verwendete Teilschicht in der Klassifikatorbeschreibung. In [19] teilen sich die Lernaufgaben in einem neuronalen Netz eine gemeinsame Eingabe- und Zwischenschicht. Xue et al. verwenden einen *Dirichlet-Process-Prior* um die Lernaufgaben implizit zu clustern [147]. Lernaufgaben innerhalb eines Clusters teilen sich auch hier die Modellparameter. In [151] hingegen teilen sich die Lernaufgaben die Modellparameter nicht direkt, aber sie hängen von gemeinsam genutzten versteckten Variablen ab. Lawrence und Platt verwenden Gauß-Prozesse mit gemeinsamen Hyperparametern [71]. Eine weitere beliebte Idee kann bei Klassifikationsverfahren genutzt werden,

welche einen Regularisierungsterm explizit in ihrer Kostenfunktion enthalten. In [74] teilen sich z. B. die einzelnen Lernaufgaben einen Regularisierungsterm und werden somit zu einer gemeinsamen Modellkomplexität gesteuert. Ein anderer Ansatz wird in [3, 4] verfolgt. Hier werden die Lernaufgaben zuerst in einen gemeinsamen niedrigdimensionalen Merkmalsraum projiziert. In dieser Darstellung sind irrelevante Merkmale nicht mehr enthalten, und die Lernaufgaben können somit auch einzeln mit wenigen Lernbeispielen robust gelöst werden.

In Abschnitt 5 wird eine Variation dieses Multitask Szenarios behandelt. Im Gegensatz zum typischen Multitask Lernen ist es hier das Ziel, die Größe der gelabelten Lernstichprobe zu minimieren.

Induktiver Transfer

Im Gegensatz zum Multitask Lernen sind die einzelnen Lernaufgaben in den folgenden Lernszenarien nicht mehr gleichgestellt. Beim induktiven Transfer wird lediglich ein Klassifikator angelernt, dessen Güte auf einer spezifizierten Lernaufgabe evaluiert wird. Diese Lernaufgabe wird als *Ziel*lernaufgabe deklariert. Die anderen Lernaufgaben tragen nur Information bei, um diese Güte zu maximieren. Diese unterstützenden Lernaufgaben werden als *Quellen*lernaufgaben bezeichnet. Es besteht somit ein klare Trennung zwischen Ziel und Quelle. Beim induktiven Transfer muss zu jeder Lernaufgabe eine Lernstichprobe existieren. Es wird davon ausgegangen, dass wie beim überwachten Lernen die Ziellernstichprobe gelabelt ist. Die Quellenlernstichproben, hingegen, dürfen auch ungelabelt sein. Aufgrund der Nähe zum überwachten Lernen wurden viele Entwicklungen als Erweiterungen von bekannten Lernverfahren konzipiert. In [144] wurde zum Beispiel eine induktive Variante der SVM präsentiert, bei der die Güte des Klassifikators auf der Ziel- und Quellenlernstichprobe gemeinsam optimiert wird. Allerdings dürfen Stützvektoren nur aus der Ziellernstichprobe gewählt werden, wodurch die lernbaren Klassifikationsmodelle auf solche beschränkt werden, welche kompatibel mit den Zieldaten sind. Viele spätere Veröffentlichungen legten den Fokus auf *boosting*-basierte Ansätze, da die hier im Modell enthaltenen Gewichte über die Lerninstanzen eine sehr flexible Adaption auf Variationen im Lernszenario erlauben ([30, 38–40, 146, 148]). In [32] und [26] wird hingegen die Beziehung der beiden Lernaufgaben in ihrer Merkmalsrepräsentation untersucht. Die Ziel- und Quellenlernstichproben sollen hier durch eine Formulierung als Regressionsproblem in [32] oder durch eine geometrische Morphingoperation in [26] in eine gemeinsame Darstellung überführt werden. Der induktive Transfer hat nach den initialen Forschungserfolgen viel an Popularität verloren und neuere Veröffentlichungen im Transferlernen beschäftigen sich zumeist mit den anderen Lernszenarien. Dies kann darauf zurückgeführt werden, dass induktive Ansätze die Klassenlabels der Ziellernstichprobe nutzen, und somit bereits weitreichendes Wissen über die Lernaufgabe vorausgesetzt wird. Ein erfolgreicher Transfer kann somit erwartungsgemäß nur noch kleine Gewinne erzielen. Lediglich in der Convolutional

Neural Network (CNN) Literatur wird noch der induktive Transfer in Form von vortrainierten neuronalen Netzen genutzt um die Robustheit des hier verwendeten stochastischen Gradientenabstiegs zu erhöhen und somit die Dauer der Lernphase deutlich zu reduzieren.

Domänenadaption

Diese Arbeit beschäftigt sich schwerpunktmäßig mit der Domänenadaption. Im Unterschied zum induktiven Transfer steht in der Domänenadaptation nur eine ungelabelte Ziellernstichprobe zur Verfügung. Der statistische Zusammenhang zwischen der Merkmalsrepräsentation und den Klassenlabels muss daher aus den gelabelten Quellenlernstichproben erlernt und auf die Ziellernaufgabe übertragen werden. Dieses Lernszenario ist im Allgemeinen schwieriger als der induktive Transfer, da aufgrund der fehlenden Klassenlabels die Beziehung von Ziel- und Quellenlernaufgabe oft nicht explizit gefasst werden kann. Allerdings wird dieses Szenario in realen Anwendungen häufig angetroffen, weshalb sich neuere Publikationen zumeist mit der Domänenadaption beschäftigen. Arbeiten können nach ihren Annahmen über die Beziehung zwischen Ziel und Quelle eingeordnet werden. In [125, 150] wird vorausgesetzt, dass sich die konditionierten Wahrscheinlichkeitsverteilungen, aus denen die Ziel- und Lernstichproben erzeugt wurden, nicht unterscheiden dürfen, wenn diese auf die Merkmalsrepräsentation konditioniert werden (*Covariate Shift*). Neuere Arbeiten versuchen diese strikte Vereinfachung aufzuheben. In [1, 7, 131] wird angenommen, dass Ziel und Quelle ähnliche Clusterstrukturen aufweisen, welche einander z. B. durch ein *Graph Matching* zugeordnet werden können. Andere Arbeiten erwarten einen semantischen Zusammenhang zwischen den Datensätzen. In [16, 17] sollte es sich bei Ziel und Quelle zum Beispiel um multitemporale Daten derselben Region zu unterschiedlichen Aufnahmezeitpunkten handeln. In [53, 82] sowie in Arbeiten für tiefe neuronale Netzarchitekturen [149] werden die Daten in eine Merkmalsrepräsentation überführt, welche die Unterschiede zwischen Ziel und Quelle minimiert. Es wird daher vorausgesetzt, dass eine solche Repräsentation existiert. In anderen Publikationen, u. a. [8, 23, 95], wird zusätzlich der Einfluss der Komplexität des gelernten Klassifikators auf den erreichbaren Transfergewinn berücksichtigt. Im Allgemeinen konnte gezeigt werden, dass einfache Lernprobleme, mit einem größeren *Margin*, einen größeren Spielraum bei der Selektion von geeigneten Quellen zulassen und somit auch ein Transfer aus eher unähnlichen Quellen sinnvoll sein kann [8]. Im Vergleich zu einem induktiven Transfer ist die Aussagekraft von theoretischen Lernmodellen in der Domänenadaption jedoch eher eingeschränkt. Die Domänenadaption hat sich allerdings auch als das vielversprechendste Szenario zur Realisierung von großen Einsparungen im Labelingaufwand herausgestellt und steht deshalb im Fokus dieser Arbeit.

Unüberwachter Transfer

Unüberwachten Lernverfahren stehen nur ungelabelte Lernstichproben zur Verfügung. Verfahren, welche in diesem Szenario arbeiten, können einem von zwei Anwendungszielen zugeordnet werden. Zum einen sind dies Verfahren, welche eine Clusteranalyse auf der Ziellernaufgabe durchführen. Durch Einbeziehung der Quellenlernstichproben sollen die korrekten Clustergrenzen robuster ermittelt werden [31]. In anderen Arbeiten werden die ungelabelten Daten genutzt um die Lernstichproben in eine gemeinsame Merkmalsrepräsentation zu überführen. In [131] wird zum Beispiel eine Cluster-Korrespondenz-Analyse zwischen den Clustern auf der Ziellernstichprobe und der Quellenlernstichprobe durchgeführt um eine nichtlineare Abbildung zwischen den Merkmalsräumen zu bestimmen. In [18] wird für die Klassifikation von Hyperspektraldaten ein Ähnlichkeitsmaß zwischen spektralen Signaturen induziert, welches die Grundlage für einen Nächster-Nachbar-Klassifikator bildet. Daran ist auch zu sehen, dass ein reiner unüberwachter Transfer aufgrund der fehlenden Datengrundlage nur für wenige Anwendungen geeignet ist. Vielmehr werden solche Verfahren als ein Baustein von vielen in Verbindung mit einem induktiven Lernverfahren oder einer Domänenadaption, wie hier entwickelt, angewendet.

1.2.2 Klassifikation von Transfermodellen

Neben den Lernszenarien können Transfermodelle danach kategorisiert werden, welcher Art das transferierte Wissen aus den Quellenlernaufgaben ist. In Abbildung 1.4 wird eine abstraktes Modell eines Transferlernverfahrens für den Transfer aus N Quellen dargestellt. Nahezu alle Verfahren des Transferlernens, welche sich im induktiven Transfer, der Domänenadaption oder im unüberwachten Lernen befinden, können nach diesem Schema drei Familien von Transfermodellen zugeordnet werden. Gemäß dem *No Free Lunch* Theorem [143] ist keines dieser Transfermodelle kategorisch den anderen überlegen. Die Wahl muss daher immer auf einen gegebenen Anwendungsfall zugeschnitten werden.

Transfer von Parametern: Die Ziel- und Quellenmodelle teilen sich entweder gemeinsame Hyperparameter in einem hierarchischen Modell oder es werden gemeinsame latente Variablen eingeführt, wie in [147].

Transfer von Merkmalsräumen: Es wird angenommen, dass die Ziel- und Quellenlernstichproben bezüglich einer Teilmenge der vorhandenen Merkmale eine ähnliche Verteilung aufweisen. Die Merkmale, welche dem Wissenstransfer schaden, können somit durch Merkmalsselektion [5] oder durch Projektion in einen geeigneten Untermerkmalsraum [32] herausgefiltert werden.

Transfer von Instanzen: Zum Anlernen des Klassifikators werden Lernbeispiele aus den Quellenlernstichproben verwendet. In den boosting-basierten Verfahren geschieht dies direkt [146]. In anderen Ansätzen werden sie zunächst transformiert,

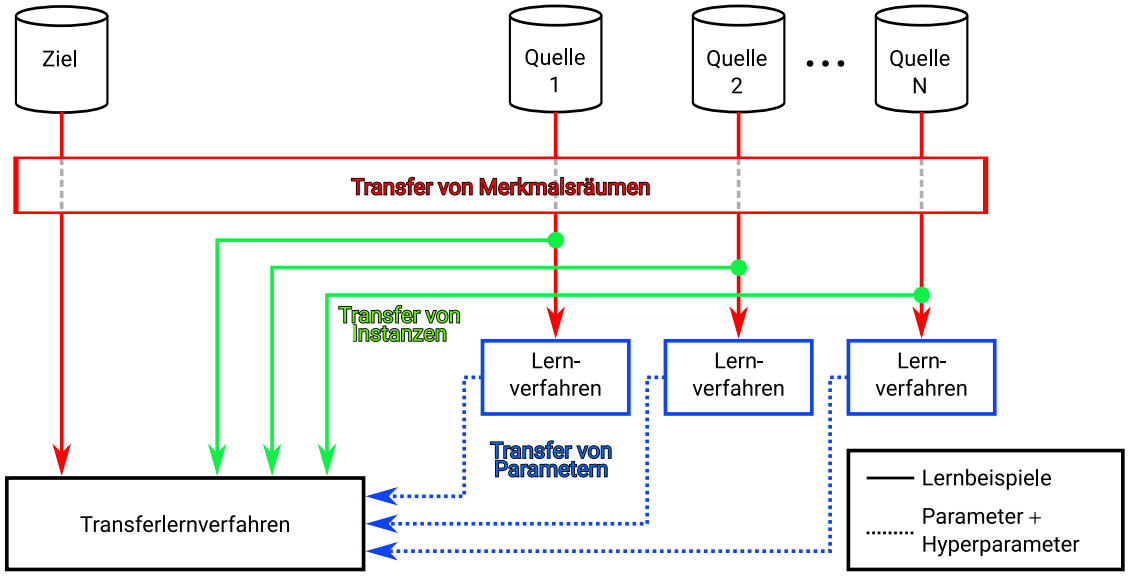


Abbildung 1.4: Klassifikation von Transfermodellen.

um die statistische Verteilung der Ziellernstichprobe besser zu approximieren [26], oder lediglich zur Induktion von sogenannten Pseudolabels in den ansonsten ungelabelten Daten des Ziels verwendet [95]. Der Transfer von Instanzen ist von besonderem Interesse, da dieser dem klassischen überwachten Lernen sehr nahe steht und daher der intuitivere der drei vorgestellten Transfermodelle ist. Allerdings ist er auch mit besonderen Herausforderungen verbunden, welche im Folgenden vorgestellt werden und mit welchen sich diese Arbeit befasst.

1.2.3 Vermeidung negativen Transfers

Im Gegensatz zum überwachten Lernen dürfen beim Transferlernen die Lernstichproben von Ziel und Quelle aus unterschiedlichen statistischen Verteilungen entstanden sein. Die Verfahren unterscheiden sich insbesondere darin, welche Annahmen über diese Unterschiede getroffen werden. Wenn diese Annahmen verletzt werden, ist das Verhalten eines Transferlernverfahrens undefiniert und führt in der Regel zum Einbrechen der Klassifikationsleistung. Falls daher die Klassifikationsleistung eines überwacht gelernten Klassifikators höher ist als unter Verwendung von Transferlernen spricht man von einem *negativen Transfer*. In vielen Arbeiten werden diese Annahmen jedoch nicht explizit gefasst oder können nicht einfach für einen gegeben Datensatz überprüft werden. Die Verantwortung liegt dann beim Anwender, dem Transferlernverfahren nur relevante Trainingsdaten zu übergeben. Für den induktiven Transfer gestaltet sich die Detektion von negativem Transfer als unproblematisch. Falls sowohl die Ziellernstichprobe als auch die Quellenlernstichprobe gelabelte Daten enthalten,

kann die Klassifikationsleistung auf dem Ziel direkt durch Standardmethoden evaluiert werden. Eaton et al. haben hierfür zum Beispiel eine Kreuzvalidierung verwendet [39, 40]. Die Differenz in der Klassifikationsleistung zwischen einem Klassifikator, welcher nur auf der Ziellernstichprobe trainiert wurde, und dem Transfermodell wurde hier mit dem Begriff *Transferability* bezeichnet. Weitergehend wurde der Fall untersucht, dass in der Ziellernstichprobe nur sehr wenige gelabelte Daten vorhanden sind [13, 152]. In der Domänenadaption besteht das Problem des negativen Transfers jedoch weiterhin. Die erfolgversprechendsten Arbeiten erarbeiteten anhand der bekannten Lerntheorie für das überwachte Lernen von Vapnik et al. [133] eine obere Schranke für den Klassifikationsfehler [8, 23]. Die zwei wichtigsten Erkenntnisse dieser Arbeiten bestehen darin, dass ein positiver Transfer nur dann möglich ist, wenn sowohl die ungelabelten Ziel- und Quellenstichproben nahezu dieselbe Verteilung besitzen und alle Klassen mit einem niedrigen Klassifikationsfehler unterscheidbar sind. Weiterhin können diese Voraussetzungen im Kontext der Domänenadaption nicht zuverlässig überprüft werden. Dennoch stellen diese Arbeiten die derzeit einzige theoretische Abhandlung dieses schwierigen Lernszenarios parat, aus denen sich konkrete Prädiktionen mit Lerngarantien ableiten lassen. Während Ben-David u. a. [8] das theoretische Modell hergeleitet hatten, beschäftigten sich Chattopadhyay u. a. [23] mit einer Implementierung für die Multiquellenselektion auf medizinischen Datensätzen. Diese Arbeit befindet sich jedoch nicht in der reinen Domänenadaption. Das vollständige dort entwickelte Verfahren verwendet immer geringe Mengen an gelabelten Zieldaten. Es handelt sich somit eher um einen induktiven Transfer. Die Methoden dieser Arbeit können hingegen wahlweise im Kontext einer Domänenadaption oder für einen unüberwachten Transfer verwendet werden. Weiterhin wurde die Multiquellenselektion aus [23] auf ein *Quadratic Programming* Problem zurückgeführt. Die Algorithmen zur Lösung solcher Optimierungsprobleme besitzen eine größere asymptotische Laufzeitkomplexität als das in dieser Arbeit präsentierte Verfahren.

Bruzzone und Marconcini [16] haben hingegen auf eine Vorhersage von negativem Transfer verzichtet. Der durch das Transferlernverfahren angelernte Klassifikator wird hier genutzt um Pseudo Klassenlabel in der Ziellernstichprobe zu generieren. Anschließend werden die Rollen von Ziel und Quelle getauscht und das Transferlernverfahren auf diesen synthetisch erweiterten Daten erneut angewendet. Der Transfer gilt als erfolgreich, falls die Klassifikationsleistung auf der Quelle nicht weit unterhalb eines Klassifikators liegt, welcher direkt auf der gelabelten Quellenlernstichprobe antrainiert wurde. Dieses Vorgehen wird in der Arbeit als zirkuläre Validierungsstrategie bezeichnet. Die zirkuläre Validierungsstrategie hatte sich zwar in Experimenten als wirksam herausgestellt, ist allerdings weniger flexibel als das theoretische Modell aus [8]. Somit bietet dieser Ansatz weniger Spielraum für Verbesserungen sowie Erweiterungen zur Behandlung von komplexeren Lernszenarien.

1.3 Ziel der Arbeit

Das Ziel dieser Arbeit ist die Entwicklung eines Transferlernverfahrens für die semantische Segmentierung von hochaufgelösten Luft- und Satellitenbildern in Landbedeckungsklassen. Einer der wichtigsten Aspekte des Transferlernens in Bezug auf die praktische Anwendbarkeit ist die Erkennung und Vermeidung negativen Transfers. Derzeit ist dieser Aspekt jedoch nur für den induktiven Transfer zufriedenstellend gelöst. Diese Arbeit wird daher den Fokus auf die Vermeidung von negativen Transfer in der Domänenadaption und dem unüberwachten Transfer durch eine Quellenselektion legen. Diese Arbeit stützt sich dabei insbesondere auf ein theoretisches Transfermodell von Ben-David et al. [8]. Die wesentlichen Beiträge dieser Arbeit stellen die Erweiterung dieses Modells dar sowie die Entwicklung einer praktikablen Lösungsmethode für die Quellenselektion auf sehr großen Datensätzen. Diese Beiträge lassen sich detaillierter wie folgt aufzählen:

- Eine Bayessche Formulierung des Transfermodells aus [8] für die Quellenselektion
- Vermeidung von kritischen Parametern; alle Parameter werden entweder direkt aus den Daten abgeleitet oder sind über einen großen Wertebereich approximativ optimal
- Eine stochastische Lösungsmethode basierend auf Markov-Chain-Monte-Carlo
- Erweiterung des Modells auf einen gleichzeitigen Transfer aus mehreren Quellen
- Anwendbarkeit auf sehr großen Datensätzen durch State-of-the-Art Laufzeitkomplexität und Speicherverbrauch
- Übertragung der Ergebnisse auf ein unüberwachtes Transferszenario

1.4 Aufbau der Arbeit

In Kapitel 2 werden zunächst Grundlagen eingeführt, welche zum Verständnis der Forschungsergebnisse erforderlich sind. Hierzu gehören insbesondere Terminologie, Notation und Einführungen in die Bayessche Entscheidungstheorie, das maschinellen Lernen, insbesondere dem Transferlernen, und Methoden der statistischen Erzeugung von Stichproben durch Markov Ketten. Das erste Hauptergebnis dieser Arbeit wird anschließend in Kapitel 3 hergeleitet. Mit dem dort entwickelten theoretischen Modell zur Domänenadaptation wird ein Distanzmaß zwischen Domänen definiert. Diese Distanz bildet die Basis für ein Bayessches Modell für die Einzelquellenselektion. In Kapitel 4 wird dieses Modell auf eine Multiquellenselektion erweitert und für den

Einsatz auf sehr großen Datensätzen bezüglich des Laufzeitverhaltens und des Speicherverbrauchs optimiert. Das zweite Hauptergebnis aus Kapitel 5 ist die Anpassung des entwickelten Domänenadaptionsmodells für den unüberwachten Transfer. Die Evaluation aller Methoden erfolgt in Kapitel 6, wonach diese Arbeit in Kapitel 7 mit einer Zusammenfassung abschließt.

Kapitel 2

Grundlagen

State-of-the-Art Methoden für den Wissenstransfer mit Anwendung im maschinellen Lernen bauen auf mehreren Jahrzehnten Grundlagenforschung sowie auf diversen Aspekten der statistischen Entscheidungstheorie auf. In diesem Kapitel sollen zunächst das Konzept der Inferenz und die Bayessche Entscheidungstheorie eingeführt werden. Anschließend werden die Grundlagen der wichtigsten Bereiche des überwachten sowie des unüberwachten maschinellen Lernens erläutert. Nachfolgend werden diese Grundlagen um Konzepte aus dem Transferlernen erweitert. Die in den Kapiteln 3 bis 5 vorgestellten Modelle und Verfahren basieren auf der Bayesschen Statistik. Daher werden hier abschließend Simulationsverfahren vorgestellt, welche innerhalb der Bayesschen Statistik häufig für die approximative Inferenz Anwendung finden.

2.1 Statistik

Die Statistik stellt eine Erweiterung der Logik um unvollständige oder unsichere Information dar. Sie wird genutzt um zufällige Ereignisse zu modellieren und Entscheidungen bei unvollständigem Wissen zu rechtfertigen. Über die letzten Jahrhunderte haben sich diverse Schulen der Statistik entwickelt, welche zwar ähnliche Algebren und Methoden hervorgebracht haben, aber dem Wahrscheinlichkeitsbegriff einen unterschiedlichen Stellenwert einräumen. Zu diesen Schulen zählen die frequentistische Statistik [42], die Evidenztheorie von Dempster und Shafer [116] sowie die Bayessche Statistik [106]. In dieser Arbeit beziehen sich alle Begriffe und Methoden auf die Bayessche Statistik. Soweit angebracht werden jedoch Ähnlichkeiten zu Methoden der anderen Schulen aufgeführt. In diesem Abschnitt werden zunächst die grundlegenden Axiome und Werkzeuge der Bayesschen Statistik definiert. Anwendungen, welche auf statistischen Modellen der Realität aufsetzen, lassen sich zumeist als Entscheidungsprobleme, wie der Parameterschätzung oder Modellselektion, formulieren. Daher wird anschließend die Entscheidungstheorie aus Sicht der Bayesschen Statistik erläutert und um Methoden der Parameterschätzung ergänzt.

2.1.1 Bayessche Statistik

Die Einführung in die Bayessche Statistik [106] soll zuerst mit der Definition des Bayesschen Wahrscheinlichkeitsbegriff anfangen.

Definition 1 (Wahrscheinlichkeit)**.** *Die Wahrscheinlichkeit beschreibt ein Maß für die persönliche Sicherheit über die Einschätzung einer Behauptung oder eines Sachverhalts A. Sie wird als $p(A)$ notiert. Dabei impliziert eine Wahrscheinlichkeit von $p(A) = 0$ die vollständige Sicherheit über dessen Unwahrheit, und $p(A) = 1$ über dessen Wahrheit.*

In der Bayesschen Statistik ist die Wahrscheinlichkeit also etwas sehr subjektives. Ein essentielles Konzept ist die Aktualisierung der persönlichen Sicherheit nach

Beobachtung von neuen Tatsachen. Dieser Vorgang wird als Inferenz bezeichnet.

Definition 2 (Inferenz)**.** *Als Inferenz wird die Aktualisierung der persönlichen Sicherheit über die Einschätzung einer Behauptung oder eines Sachverhalts A nach Beobachtung einer Information I bezeichnet. Eine Inferenz wird durch die bedingte Wahrscheinlichkeit $p(A|I)$ notiert.*

Im Unterschied zu einem allgemeineren Konfidenzbegriff erlaubt es uns die Wahrscheinlichkeit, Informationen zu neuen Erkenntnissen zu verknüpfen. Hierfür wurden in der Literatur diverse axiomatische Systeme vorgeschlagen, aus denen ein Framework aus Rechenoperationen und Methoden ableitbar ist. An dieser Stelle sollen die Axiome der Inferenz von Cox [105] vorgestellt werden. In Verbindung mit der Boolschen Algebra kann gezeigt werden, dass sie äquivalente Aussagen zu anderen axiomatischen Systemen der Wahrscheinlichkeit liefern, wie z. B. von Kolmogoroff [117] oder von de Finetti [33].

Axiom 1. *Die Wahrscheinlichkeit einer Inferenz A gegeben einer beobachteten, bzw. sicheren, Information I bestimmt eindeutig die Wahrscheinlichkeit der komplementären Inferenz A^C gegeben derselben Information.*

$$p\left(A^C \middle| I\right) = F[p(A|I)] \tag{2.1}$$

Axiom 2. *Die Wahrscheinlichkeit, dass zwei Inferenzen A und B gegeben einer beobachteten, bzw. sicheren, Information I wahr sind, wird eindeutig bestimmt durch die Wahrscheinlichkeit von Inferenz A gegeben I und der Wahrscheinlichkeit von Inferenz B gegeben I unter der zusätzlichen Annahme, dass A wahr ist.*

$$p(A, B|I) = G[p(A|I)\,, p(B|A, I)] \tag{2.2}$$

In [105] wurde anschließend gezeigt, dass die Funktionen $F(.)$ und $G(.)$ eindeutig aus den Definitionen 1, 2 sowie den Axiomen 1 und 2 hergeleitet werden können. Es gilt:

$$p\left(A^C \middle| I\right) = F[p(A|I)] = 1 - p(A|I) \tag{2.3}$$

$$p(A, B|I) = G[p(A|I)\,, p(B|A, I)] = p(B|A, I) \cdot p(A|I)\ . \tag{2.4}$$

Abgeleitet aus diesem axiomatischen System wird die Bayessche Aktualisierung der persönlichen Sicherheit durch den Satz von Bayes verkörpert [106]:

$$p(A|I) = \frac{p(A, I)}{p(I)} = \frac{p(I|A)\,p(A)}{p(I)}\,. \tag{2.5}$$

Die Variable I beschreibt eine Beobachtung, bzw. eine sichere Information, und wird als unabhängige Variable bezeichnet. Durch Ausnutzen von statistischen Abhängigkeiten zwischen I und A können wir nun Informationen über den Zustand der

uns unbekannten abhängigen Variable A gewinnen. In diesem Kontext wird $p(I|A)$ als Likelihood bezeichnet und $p(A)$ als a-Priori Wahrscheinlichkeit. Die Likelihood beschreibt immer einen datenerzeugenden Prozess. In diesem Fall wird durch sie für jede potentielle Beobachtung aus A eine Wahrscheinlichkeitsverteilung der Daten beschrieben. Die a-Priori Wahrscheinlichkeit spiegelt die Sicherheit über das Eintreten eines Ereignisses wider, bevor die Daten beobachtet wurden. Der Term $p(I)$ wird als Evidenz bezeichnet und kann in Entscheidungsmodellen als Normierungskonstante behandelt werden. Die Bayessche Aktualisierung verknüpft demnach unser a-Priori Wissen mit den beobachteten Daten. Dieses Vorgehen kann iteriert werden, indem die so gewonnene a-Posteriori Wahrscheinlichkeit nach Beobachtung von weiteren Daten als neue a-Priori Wahrscheinlichkeit interpretiert wird. Mit jeder Beobachtung kann so die persönliche Sicherheit einer objektiven Einschätzung angenähert werden. Die besondere Stärke, und gleichzeitig auch der größte Kritikpunkt der Bayesschen Statistik, besteht darin, Vorwissen und Modellannahmen in Form der a-Priori Wahrscheinlichkeit beschreibbar zu machen. Durch falsche Wahl der a-Priori Verteilung können jedoch beliebige Inferenzen erzwungen werden. Viele Arbeiten haben sich daher mit der Frage beschäftigt, wie der unbeabsichtigte Einfluss der a-Priori Verteilung auf die Inferenz vermieden werden kann. Hieraus hat sich das Konzept der uninformativen a-Priori Verteilungen entwickelt. Die beiden wichtigsten Klassen von uninformativen a-Priori Verteilungen sind die Jeffrey's Prior [106] und die Referenz Prior von Bernado [9]. Die a-Priori Verteilung wird für den Jeffrey's Prior z. B. so gewählt, dass sie der Wurzel der Fisher-Information zwischen den Beobachtungen und den messbaren Daten entspricht. Bereiche, welche über den gesamten Raum der Beobachtungen eine niedrige mittlere log-Likelihood aufweisen, besitzen demnach nur wenig Information über die Daten. Dieser Umstand sollte folglich zu einer ebenfalls niedrigen a-Priori Wahrscheinlichkeit führen.

Die Abbildung 2.1 zeigt die Bayesche Inferenz durch Aktualisierung der a-Posteriori Verteilung am Beispiel eines Münzwurfexperiments. Die unbekannten Seitenwahrscheinlichkeiten einer Münze werden durch eine Betaverteilung modelliert und sollen durch Beobachtung von wiederholten Münzwürfen geschätzt werden. Wir besitzen a-Priori keinerlei Information darüber, ob die Münze fair ist. Daher bietet sich hier die Verwendung des Jeffrey's Prior an, welcher in diesem Experiment die Form einer Betaverteilung mit den Parametern $\alpha = \beta = 0.5$ annimmt. Nach jeweils 10 Münzwürfen wird die Likelihood empirisch aus den beobachteten relativen Seitenhäufigkeiten als Binomialverteilung geschätzt und zur Aktualisierung des statistischen Modells genutzt. Bereits nach wenigen Aktualisierungen kommen wir mit einer hohen Sicherheit zu den Schluss, dass die Münze die Seitenwahrscheinlichkeiten $p(\text{Zahl}) \approx 0.7$ und $p(\text{Kopf}) = 1 - p(\text{Zahl}) \approx 0.3$ besitzen muss. Es handelt sich somit nicht um eine faire Münze. In einem zweiten Experiment soll dieser Versuchsaufbau mit einer ungünstig gewählten a-Priori Verteilung wiederholt werden (Abbildung 2.2). Wir nehmen nun mit einer hohen Sicherheit an, dass die Münze unfair ist und eine Seitenwahrscheinlichkeit von $p(\text{Kopf}) \approx 0.9$ besitzt. Durch Beobachtung von

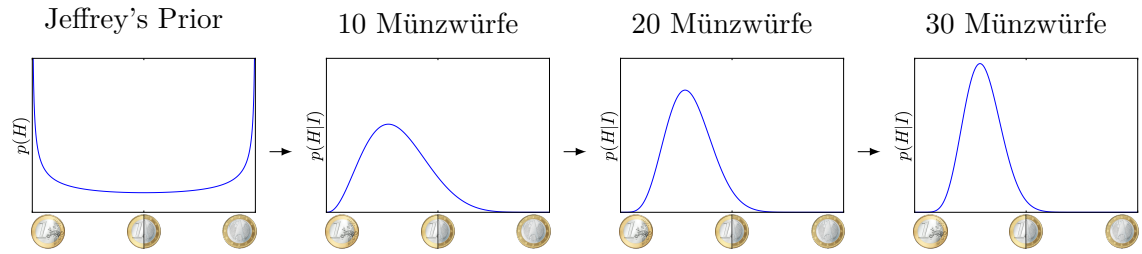


Abbildung 2.1: Bayessche Aktualisierung am Beispiel eines Münzwurfexperiments. Es wird eine unfaire Münze mit den Seitenwahrscheinlichkeiten $p(\text{Zahl}) = 0.7$ und $p(\text{Kopf}) = 0.3$ verwendet. Die Wahrscheinlichkeiten können ohne weitere a-Priori Annahmen bereits nach wenigen Münzwürfen zuverlässig geschätzt werden.

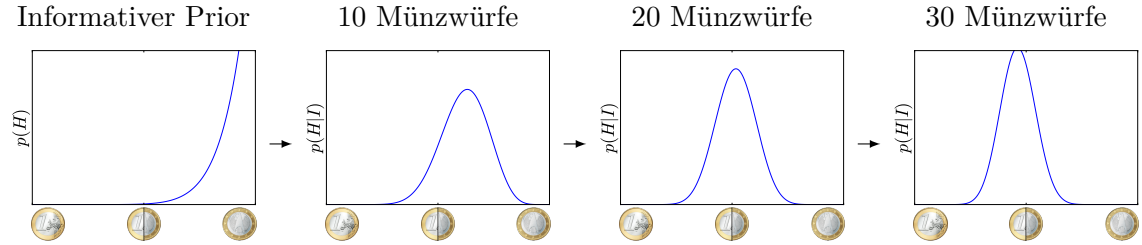


Abbildung 2.2: Bayessche Aktualisierung am Beispiel eines Münzwurfexperiments. Es wird eine unfaire Münze mit den Seitenwahrscheinlichkeiten $p(\text{Zahl}) = 0.7$ und $p(\text{Kopf}) = 0.3$ verwendet. Als a-Priori Wissen wird eine unfaire Münze mit den falschen Seitenwahrscheinlichkeiten $p(\text{Zahl}) \approx 0.1$ und $p(\text{Kopf}) \approx 0.9$ angenommen. Die Schätzung konvergiert erst nach sehr vielen Münzwürfen zu den korrekten Wahrscheinlichkeiten.

genügend Experimenten nähert sich die a-Posteriori Verteilung langsam der Realität an. Jedoch werden selbst nach 30 Münzwürfen die Seitenwahrscheinlichkeiten mit $p(\text{Zahl}) \approx 0.54$ und $p(\text{Kopf}) \approx 0.46$ falsch geschätzt.

Bereits für einfache Modelle sind uninformative a-Priori Verteilungen entweder schwer zu berechnen oder führen zu einer unzulässigen Verteilung. Daher werden häufig sogenannte schwach informative a-Priori Verteilungen verwendet. Da z. B. der Jeffrey's Prior von Skalenparametern, wie der Standardabweichung einer Normalverteilung, die Form $p(\sigma) \propto \sigma^{-1}$ besitzt und somit nicht alle Eigenschaften von gültigen Wahrscheinlichkeitsverteilung erfüllt, werden in der Literatur in der Regel die inverse Gammaverteilung oder die Halb-Cauchy-Verteilung verwendet [100]. In der Literatur existiert zwar keine einheitliche Definition von solchen a-Priori Verteilungen, allerdings besitzen sie immer die Eigenschaft, dass sie über fast den gesamten

Träger der Verteilung durch die Likelihood dominiert werden. Diese Eigenschaft kann durch Vergleich der partiellen Ableitungen der log-a-Priori-Verteilung und der log-Likelihood getestet werden. Eine weitere Möglichkeit, den Einfluss von a-Priori Verteilungen bei fehlendem Vorwissen zu reduzieren, stellen die hierarchischen Modelle dar. Angenommen, unser Modell enthalte einen Skalenparameter σ, dessen a-Priori Verteilung $p(\sigma)$ z. B. als Exponentialverteilung mit dem Parameter λ angenommen werden kann. Die Wahl des Parameters λ kann bei kleinen Datenmengen einen großen Einfluss auf eine Inferenz haben. Durch Hinzufügen einer a-Priori Verteilung auf λ beschreibt das resultierende hierarchische Modell $p(\sigma|\lambda)\,p(\lambda)$ unsere Annahme, dass wir zwar die Form der a-Priori Verteilung von σ kennen, jedoch nicht dessen Mittelwert. In hierarchischen Modellen werden Parameter von a-Priori Verteilungen als *Hyperparameter* bezeichnet.

2.1.2 Entscheidungstheorie

Die Entscheidungstheorie beschreibt Systeme, welche es uns erlauben, Entscheidungen auf Basis von unsicheren oder unvollständigen Informationen zu treffen. Die Ursprünge der Entscheidungstheorie liegen in der Ökonomie und Spieltheorie [139]. Aufgrund von scheinbar unlösbaren Paradoxen [10] wurde die theoretische Beschreibbarkeit von allgemeinen Entscheidungssystemen lange angezweifelt. In den 1940er und 1950er Jahren wurden von Wald [140] sowie von Neumann und Morgenstern [139] erstmals die tiefgreifende Beziehung zwischen der Statistik und der Entscheidungstheorie hergeleitet und ein axiomatisches System für rationale Entscheidungen definiert.

Axiom 3 (Vollständigkeit)**.** *Gegeben sei ein rationaler Agent welcher zwischen zwei Wahlmöglichkeiten r_1 und r_2 ($r \in \mathcal{R}$) entscheiden muss. Es gilt, dass der Agent entweder eine der beiden Wahlmöglichkeiten bevorzugt oder sie als gleich akzeptabel einschätzt.*

$$\underbrace{r_1 \prec r_2}_{\textit{Bevorzugt } r_2} \vee \underbrace{r_1 \succ r_2}_{\textit{Bevorzugt } r_1} \vee \underbrace{r_1 \sim r_2}_{\textit{Kein Vorzug}} \tag{2.6}$$

Axiom 4 (Transitivität)**.** *Falls der Agent eine Wahlmöglichkeit r_2 der Wahlmöglichkeit r_1 bevorzugt und gleichzeitig eine Wahlmöglichkeit r_3 der Wahlmöglichkeit r_2 bevorzugt, dann muss er auch r_3 vor r_1 bevorzugen.*

$$r_1 \preceq r_2 \wedge r_2 \preceq r_3 \implies r_1 \preceq r_3 \tag{2.7}$$

Axiom 5 (Kontinuität)**.** *Gegeben seien die drei Wahlmöglichkeiten r_1, r_2 und r_3 in der Reihenfolge ihrer Akzeptabilität für den Agenten. Falls sich die beiden Wahlmöglichkeiten r_1 und r_3 in ihrer Akzeptabilität einander annähern sollten, dann gibt es einen Punkt, an dem sie als gleich akzeptabel zur Wahlmöglichkeit r_2 eingestuft werden.*

$$\forall r_1, r_2, r_3,\ r_1 \preceq r_2 \preceq r_3 \exists p \in [0{,}1] \implies p \cdot r_1 + (1-p) \cdot r_3 \sim r_2 \tag{2.8}$$

Axiom 6 (Unabhängigkeit)**.** *Fall der Agent eine Wahlmöglichkeit r_2 der Wahlmöglichkeit r_1 bevorzugt, dann darf sich diese Präferenz nicht ändern, falls eine dritte Wahlmöglichkeit r angeboten wird.*

$$\forall r \forall p \in [0,1] \forall r_1, r_2, r_1 \prec r_2 \implies p \cdot r_1 + (1-p) \cdot r \preceq p \cdot r_2 + (1-p) \cdot r \tag{2.9}$$

Diese vier Axiome führen zur Hypothese der erwarteten Nützlichkeit:

Hypothese 1 (Hypothese der erwarteten Nützlichkeit)**.** *Ein Agent, welcher nach den vier Axiomen der rationalen Entscheidung agiert, versucht bei Entscheidungen mit Unsicherheit oder unvollständigem Wissen immer die erwartete Nützlichkeit seiner Entscheidung zu maximieren:*

$$r_1 \prec r_2 \Leftrightarrow E[u(r_1)] < E[u(r_2)] \; . \tag{2.10}$$

Die Nützlichkeit wird durch die Utility *Funktion $u : \mathcal{R} \to \mathbb{R}$ beschrieben. Für sichere Entscheidungen mit vollständigem Wissen gilt für die Utility Funktion:*

$$r_1 \prec r_2 \Leftrightarrow u(r_1) < u(r_2) \, . \tag{2.11}$$

Daraus folgt, dass beliebige Entscheidungsprobleme, unter der Annahme von rationalen Agenten, als Inferenzen in der Bayesschen Statistik formuliert werden können. Statt der *Utility* Funktion $u(r)$ wird häufig äquivalent eine Kostenfunktion $L(r)$ verwendet. Im allgemeinen gilt $L(r) = -u(r)$. Für den Spezialfall der Parameterschätzung werden im nächsten Abschnitt einige weit verbreitete Kostenfunktionen vorgestellt.

2.1.3 Parameterschätzung

Die a-Posteriori Verteilung beschreibt unser Vorwissen sowie die getätigten Beobachtungen bezüglich eines Sachverhalts. Das Bayessche Aktualisierungsschema legt nun die Intuition nahe, dass für ein vollständiges Verständnis des modellierten Sachverhalts immer die gesamte a-Posteriori Verteilung betrachtet werden muss. Immerhin könnten zukünftige Messungen die subjektive Einschätzung im Grenzfall beliebig stark zu anderen Regionen im Parameterraum hin verschieben [75]. Aus praktischen Gründen wird allerdings die Verteilung entweder durch einen Bereichsschätzer auf ein Werteintervall oder durch einen Punktschätzer auf einen konkreten Parameterwert reduziert. Für den letzteren Fall liefert uns die Hypothese der erwarteten Nützlichkeit eine Methode für die Parameterschätzung. Es gilt für den optimalen Parameter $\bar{\theta}$ unter Beobachtung der Information I und der gegebenen Kostenfunktion $L(\theta, x)$:

$$\bar{\theta} = \arg\min_{\theta \in \Theta} \mathrm{E}[L(\theta, x)|I] \; . \tag{2.12}$$

In dieser Formulierung wird der Erwartungswert über die Zufallsvariable x gebildet, welche der a-Posteriori Verteilung von θ über den gesamten Parameterraum folgt. Der Schätzwert $\bar{\theta}$ repräsentiert somit einen Parameterwert, welcher eine hohe mittlere *Ähnlichkeit* zu einem zufällig gewählten $x \sim p(\theta|I)$ aufweisen sollte, wobei das Maß der Ähnlichkeit durch die Kostenfunktion beschrieben wird. Die Kostenfunktion sollte deshalb immer problemspezifisch in Abhängigkeit von der Form der a-Posteriori Verteilung ausgewählt werden. Aus der Literatur sind diverse Funktionen bekannt, welche zu Schätzern mit unterschiedlichem Eigenschaften führen:

Maximum a-Posteriori Schätzer Als Maximum a-Posteriori (MAP) Schätzer wird der Modalwert der a-Posteriori Verteilung bezeichnet. Es handelt sich somit um den Parameter, welcher die a-Posteriori Verteilung maximiert:

$$\bar{\theta}_{\text{MAP}} = \arg\max_{\theta \in \Theta} p(\theta|I)\,. \tag{2.13}$$

In der Bayesschen Statistik besitzen sie nach dem *Bernstein-von-Mises* Theorem [72] die einzigartige Eigenschaft, dass der Einfluss des a-Priori Wissens bei einer steigenden Anzahl von Beobachtungen gänzlich verschwindet. Für andere Schätzer gilt dies nicht für beliebige a-Priori Verteilungen. Die Kostenfunktion eines MAP Schätzers hat die Form

$$L_{\text{MAP}}(\theta, x) = \begin{cases} 0 & \text{, falls } |\theta - x| \leq \epsilon \\ 1 & \text{, sonst} \end{cases}\,. \tag{2.14}$$

für $\epsilon \to 0$.

Maximum Likelihood Schätzer Als Maximum Likelihood (ML) Schätzer wird der Modalwert der Likelihood Verteilung bezeichnet:

$$\bar{\theta}_{\text{ML}} = \arg\max_{\theta \in \Theta} p(I|\theta)\,. \tag{2.15}$$

Man beachte, dass für diese Schätzung die a-Priori Verteilung ignoriert wird. Dies ist äquivalent mit der Verwendung einer Gleichverteilung als a-Priori Wissen. In der frequentistischen Statistik werden häufig ML Schätzer verwendet, da ihnen gute asymptotische Eigenschaften nachgewiesen werden können [73]. Bei wenigen Beobachtungen tendieren sie allerdings zu wenig robusten Schätzungen, welche von sporadischen Effekten und Rauschen beeinflusst werden.

Erwartungswertschätzer In enger Verwandtschaft zu Least Squares (LS) Optimierungsverfahren werden bei unimodalen a-Posteriori Verteilungen häufig quadratische Schätzkosten angenommen. Die Kostenfunktion lautet:

$$L_{\text{MSE}}(\theta, x) = (\theta - x)^2\,. \tag{2.16}$$

Damit ergibt sich für den Schätzer eine besonders einfache Form. Es gilt:

$$\bar{\theta}_{\text{MSE}} = \text{E}[\theta|I] = \text{mean}(\theta \mid I)\,. \tag{2.17}$$

Für diesen Schätzer gilt, dass er wenig robust gegenüber Ausreißern ist. Für endlastige a-Posteriori Verteilungen empfiehlt sich daher die Verwendung der absoluten Kostenfunktion.

Medianschätzer Um starke Schätzfehler weniger stark zu gewichten und somit eine größere Robustheit zu erreichen werden alternativ zu den quadratischen Kosten manchmal absolute Kosten verwendet:

$$L_{\text{MAE}}(\theta, x) = |\theta - x|\,. \tag{2.18}$$

Auch für die absoluten Kosten nimmt der Schätzer eine einfache Form an:

$$\bar{\theta}_{\text{MAE}} = \text{median}(\theta \mid I)\,. \tag{2.19}$$

2.2 Klassifikation

Bei der Klassifikation spricht man von einem Teilgebiet der Entscheidungstheorie, bei der einem Merkmalsvektor $\boldsymbol{x} \in \mathcal{F}$ aus einem Merkmalsraum $\mathcal{F}$ ein Klassenlabel $y \in \mathcal{K}$ zugewiesen wird. Die hierfür zuständige Funktion $y = h(\boldsymbol{x})$ wird auch Entscheidungsfunktion oder Klassifikator genannt. Als vereinfachende, aber wie in Abschnitt 2.2.4 gezeigt wird keinesfalls einschränkende, Annahme soll $\mathcal{F} \subseteq \mathbb{R}^D$ eine Teilmenge des kartesischen Koordinatensystems der Dimension D sein. Die Menge $\mathcal{K}$ hingegen skaliert nominal. D. h. die möglichen Klassenlabels sind unterscheidbar, weisen aber keine natürliche Rangfolge auf. Dies unterscheidet die Klassifikation von der Regression, bei der die Zielvariable y reellwertig und typischerweise intervallskaliert ist.

In Anwendungsszenarien kann $h(\boldsymbol{x})$ aufgrund seiner Komplexität und dem komplexen Zusammenspiel von interagierenden Merkmalen oft nicht manuell angegeben werden. Stattdessen wird die Entscheidungsfunktion automatisch aus einer Lernstichprobe Tr abgeleitet. Je nachdem, welche Informationen uns über Tr zur Verfügung stehen, wird zwischen überwachten und unüberwachten Verfahren unterschieden. Überwachte Verfahren verwenden die Lernstichprobe $\text{Tr} = \{\boldsymbol{x}_i, y_i\}_{i=1}^N$, bei der zu jedem Lernbeispiel $\boldsymbol{x}_i$ das dazugehörige korrekte Klassenlabel y_i vorhanden ist. Unüberwachte Verfahren hingegen haben nur Kenntnis von den Lernbeispielen selbst. Die Lernstichprobe ist daher als $Tr = \{\boldsymbol{x}_i\}_{i=1}^N$ definiert. Diese Verfahren werden in den Abschnitten 2.2.2 bzw. 2.2.3 genauer erläutert.

Die konkrete Wahl von geeigneten Merkmalen ist höchstgradigst anwendungsabhängig und kann einen signifikanten Einfluss auf die Klassifikationsleistung haben.

Bei vielen Klassifikationsverfahren besteht hierin auch die einzige Möglichkeit, domänenspezifisches Expertenwissen einfließen zu lassen. Dennoch soll den Merkmalen in dieser Arbeit eine sekundäre Rolle zugewiesen werden. Vielmehr liegt der Fokus in den statistischen Modellen und in robusten, gut generalisierenden Lernverfahren. Die folgenden Abschnitte sollen daher die wichtigsten Konzepte erläutern, welche zum Verständnis von modernen Verfahren zum Lernen von Entscheidungsfunktionen unabdingbar sind.

2.2.1 Generative und diskriminative Entscheidungsmodelle

Die Entscheidungsfunktion $h(\boldsymbol{x})$ kann für die meisten Klassifikationsverfahren als MAP Schätzung des Klassenlabels interpretiert werden, d. h.

$$h(\boldsymbol{x}) = \arg\max_{y \in \mathcal{K}} p(y|\boldsymbol{x})\,. \tag{2.20}$$

In generativen Entscheidungsmodellen wird die a-Posteriori Wahrscheinlichkeit wie in Formel (2.5) beschrieben durch Modellierung der Verbundwahrscheinlichkeit $p(\boldsymbol{x}, y)$ gebildet. Um diese zuverlässig aus der Lernstichprobe ableiten zu können werden häufig Annahmen bezüglich der in Frage kommenden Verteilungsfamilien getroffen. Im einfachsten Fall wird die Verbundwahrscheinlichkeit durch einfache Verteilungen mit wenigen Parametern, wie z. B. der Normalverteilung, approximiert. Bei komplexeren Ansätzen kommen sogenannte parameterfreie Verteilungen zum Einsatz [49]. Parameterfreie Modelle sind entgegen ihrer Bezeichnung zwar parametrisiert, jedoch kann die Anzahl ihrer Parameter, und somit die potentiell darstellbare Modellkomplexität, frei auf die Eingangsdaten zugeschnitten werden. Beispiele für solche Modelle sind Histogramme, Mischverteilungen und die Parzen-Fenster Methode [12]. In Abbildung 2.3 wird ein Beispiel für eine Klassifikation durch ein generatives Entscheidungsmodell gezeigt. Es ist im Allgemeinen schwierig sicherzustellen, dass das gewählte Modell die zugrundeliegende Datenverteilung ausreichend gut approximiert. Weiterhin stehen diese Approximationsfehler in keinem einfach handhabbaren Verhältnis zum Klassifikationsfehler des daraus abgeleiteten Entscheidungsmodells. Dies hat zur Folge, dass jegliche fehlerhaften Modellannahmen einen undefinierbaren und unkorrigierbaren Einfluss auf die Klassifikation haben können. Aus diesem Grund kommen in modernen Klassifikationsverfahren meistens diskriminative Entscheidungsmodelle zum Einsatz. Bei diesen Modellen wird die a-Posteriori Verteilung direkt modelliert. Auch hier müssen Annahmen über mögliche Verteilungsfamilien getroffen werden. Allerdings ist es für diskriminative Entscheidungsmodelle möglich, fehlerhafte Modellannahmen während der Lernphase so auszugleichen, dass der resultierende Klassifikationsfehler dennoch minimiert wird. Aufgrund der genannten Vorteile wird sich diese Arbeit im Folgenden exklusiv mit diskriminativen Entscheidungsmodellen beschäftigen. Die Trennebene für ein diskriminatives Entscheidungsmodell ist ebenfalls in Abbildung 2.3 gegeben.

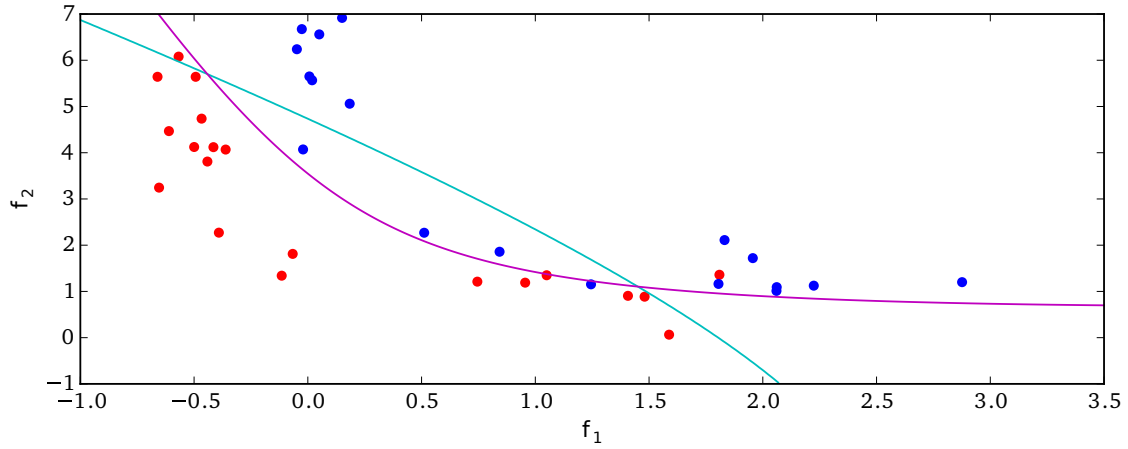


Abbildung 2.3: Vergleich eines generativen und diskriminativen Modells für ein Zweiklassen Entscheidungsproblem unter Annahme der Normalität. Die roten und blauen Merkmalsvektoren wurden tatsächlich aus Mischverteilungen mit jeweils zwei Normalverteilten Komponenten erzeugt. Das generative Entscheidungsmodell (cyan) kann die Verletzung der Annahme nicht korrigieren und erzeugt einen empirischen Fehler von 15%. Das diskriminative Modell (magenta) schneidet mit einem Fehler von 5% deutlich besser ab.

2.2.2 Überwachtes Lernen

Für das überwachte Erlernen von Entscheidungsmodellen aus einer Lernstichprobe wurden über die vergangenen Jahrzehnte diverse theoretische Grundgerüste, Modelle und Verfahren entwickelt. In diesem Abschnitt werden zunächst die grundlegenden Klassifikationsverfahren genannt. Anschließend folgt eine Diskussion über die diversen Qualitätsanforderungen, welche an moderne Verfahren gestellt werden und welche Kriterien zum Vergleich von Entscheidungsmodellen genutzt werden sollten. Zum Abschluss werden weiterführende, aber dennoch essentielle, Themengebiete vorgestellt. Hierzu gehören Methoden, mit denen Multiklassenprobleme mit mehr als zwei Klassenlabels behandelt werden können. Abschließend folgt die Präsentation von Verfahren zur Abschätzung der Entscheidungssicherheit von Klassifikationen.

Verfahren

Aus der Vielzahl der existierenden überwachten Klassifikationsverfahren und ihrer Varianten können hier aus Platzgründen nur einige historisch besonders relevante Kandidaten genannt werden. Viele der nicht genannten Methoden basieren jedoch auf ähnlichen Konzepten. Um die wichtigen Entwurfsentscheidungen besser verstehen zu können, welche bei der Entwicklung dieser Algorithmen getroffen wurden, werden

hier die drei wichtigen Kernkomponenten von Klassifikationsverfahren gesondert behandelt.

Partitionierunsgmodelle Bei der Entscheidungsfunktion $y = h(\boldsymbol{x})$ handelt es sich um eine Abbildung der Form $h : \mathcal{F} \to \mathcal{K}$. Solch eine Abbildung kann auch als Partitionierung des Merkmalsraums in $|\mathcal{K}|$, nicht notwendigerweise zusammenhängende, Partitionen interpretiert werden. Um $h(\boldsymbol{x})$ für Computeralgorithmen greifbar zu machen benötigen wir als erstes ein parametrisiertes Partitionierungsmodell, welches sich durch seine Parameter $\boldsymbol{\varphi} = \{\varphi_i\}_{i=1}^{P} \in \mathcal{P}$ beschreiben lässt. Wir nehmen außerdem an, dass P endlich ist. Die a-Posteriori Wahrscheinlichkeit gegeben einem Partitionierungsmodell wird nun zu $p(y|\boldsymbol{x}, \boldsymbol{\varphi})$ umformuliert.

Bei der Nächster-Nachbar (NN) Entscheidungsregel [12] handelt es sich um eine der frühesten diskriminativen parameterfreien Entscheidungsmodelle. Sie kann so formuliert werden, dass jedem Merkmalsvektor $\boldsymbol{x}$ das Klassenlabel y_i des räumlich nächstgelegensten Lernbeispiels $(\boldsymbol{x}_i, y_i)$ aus der Lernstichprobe zugeordnet wird. Die hierfür benötigten Distanzvergleiche können im Prinzip über beliebige Metriken oder semi-Metriken im Merkmalsraum erfolgen. Typischerweise wird die euklidische Distanz $d_2(\boldsymbol{x}, \boldsymbol{x}_i) = \|\boldsymbol{x} - \boldsymbol{x}_i\|$ verwendet. Die Partitionen ergeben sich dann direkt aus dem Voronoi Diagramm der Lernstichprobe.

Die einfachste Klasse von parametrisierten Entscheidungsregeln stellen die linearen Trennebenen dar [12, 59, 64]. Für eindimensionale Merkmalsräume ergibt sich der Sonderfall einer Schwellwertentscheidung. Für den allgemeinen Fall hängt die Entscheidungsfunktion von einer Hyperebene in Hessescher Normalform mit den Parametern $(\boldsymbol{\nu}, b)$ ab. Die Entscheidungsfunktion für Zweiklassen Probleme ergibt sich zu

$$h_{\text{linear}}(\boldsymbol{x}) = \begin{cases} +1 & \text{, falls } \langle \boldsymbol{x}, \boldsymbol{\nu} \rangle > b \\ -1 & \text{, sonst .} \end{cases} \tag{2.21}$$

Die Klassenlabels werden in Zweiklassenproblemen zur Vereinfachung einiger Definitionen und Rechnungen im Allgemeinen zu $y \in \{-1, +1\}$ definiert. Die Entscheidungsfunktion lässt sich dann so verstehen, dass der Merkmalsvektor zuerst in einen durch die Normale aufgespannten eindimensionalen Unterraum projiziert wird. Dieser Unterraum wird so gewählt, dass sich in ihm die beiden Klassen möglichst effektiv durch eine einfache Schwellwertentscheidung mit Schwellwert b unterscheiden lassen. Durch $d(\boldsymbol{x}) = \langle \boldsymbol{x}, \boldsymbol{\nu} \rangle - b$ wird eine vorzeichenbehaftete Distanz eines Merkmalsvektors zur Entscheidungsbene definiert. Das Vorzeichen von $d(\boldsymbol{x})$ gibt hierbei die Klassenentscheidung wieder, während ihr Betrag als Konfidenz dieser Entscheidung betrachtet werden kann. Als

Vorteil dieses Partitionierungsmodells sei zu nennen, dass es sich durch wenige Parameter beschreiben lässt und dennoch Korrelationen zwischen den einzelnen Merkmalen effektiv erfassen kann. Weiterhin können lineare Trennebenen als Basis in Verbindung mit dem Kernel Trick aus Abschnitt 2.2.4 genutzt werden, um komplexere Entscheidungsmodelle zu generieren.

Durch lineare Trennebenen lassen sich nur einfache Partitionierungen mit zusammenhängenden Partitionen beschreiben. Bei Problemen, in denen die Merkmalsvektoren der einzelnen Klassenlabels in separate Cluster zerfallen, bieten sich Entscheidungsbäume an [14, 102]. Die Partitionierung des Merkmalsraums geschieht hierbei hierarchisch in einer Baumstruktur. Der Wurzelknoten repräsentiert den gesamten Merkmalsraum. Jeder Knoten im Baum kann nun entweder als Blattknoten oder als innerer Knoten markiert werden. Ein Blattknoten beendet die hierarchische Partitionierung und weist der aktuellen Partition, welche eindeutig durch den Pfad vom Wurzelknoten zum Blattknoten beschrieben wird, ein Klassenlabel zu. Innere Knoten hingegen werden mit einer Entscheidungsfunktion versehen, welche die aktuelle Partition in M Subpartitionen zerlegt. Für jede Subpartition wird ein neuer Kindknoten angelegt. Für den einfachsten Fall, dass als Entscheidungsfunktionen Schwellwertentscheidungen oder lineare Trennebenen verwendet werden, wird somit ein binärer Entscheidungsbaum ($M = 2$) aufgebaut.

Partitionierungsmodelle aus Entscheidungsbäumen sind aufgrund der einfachen Entscheidungsfunktionen jedes Knotens stückweise linear. Künstliche neuronale Netze [12, 66, 111] sind hingegen in der Lage, lokale Nichtlinearitäten zu modellieren und erlauben somit das Erlernen von nahezu beliebigen funktionalen Zusammenhängen [29]. Die Kernkomponente eines künstlichen neuronalen Netzes ist das Neuron. Ein einzelnes Neuron wird durch die Eingänge $\boldsymbol{I} = \{I_j\}_{j=1}^{N}$ angesteuert und errechnet aus ihnen den Ausgang O. Jedem der Eingänge wird eine Gewicht $\boldsymbol{w} = \{w_j\}_{j=1}^{N}$ zugeordnet. Durch gewichtete Addition der Eingänge wird nun zunächst der Zwischenwert $d(\boldsymbol{w}) = \langle \boldsymbol{I}, \boldsymbol{w} \rangle - b$ bestimmt. Jedes Neuron definiert also auch eine lineare Trennebene und somit eine Projektion in einem eindimensionalen Unterraum. Komplexere Entscheidungsfunktionen können gebildet werden, indem die Ausgänge von solchen Neuronen mit den Eingängen von weiteren Neuronen verbunden werden. Allerdings wurde gezeigt, dass ein so definiertes künstliches neuronales Netz aufgrund der Linearität der Berechnungsvorschrift von $d(\boldsymbol{w})$ keine komplexeren Partitionierungen erzeugen kann als eine einfache lineare Trennebene [29]. Daher wird der Ausgang eines Neurons immer durch Auswertung einer nichtlinearen Funktion auf $d(\boldsymbol{w})$ berechnet. Es gilt $O = f(d(\boldsymbol{w}))$. Die Funktion $f(.)$ wird in diesem Kontext als Aktivierungsfunktion bezeichnet. Typische Implementierungen hierfür sind Schwellwertentscheidungen, Sigmoidfunktionen oder radiale Basisfunktionen. Obwohl jedoch solche neuronalen Netze in der Theorie beliebige Funktionen

approximieren können, setzt dies eine korrekte Wahl der Netzstruktur und der Aktivierungsfunktionen voraus. Die Gewichte hingegen können danach durch gradientenbasierte Optimierungsverfahren, wie dem *Backpropagation* Algorithmus, aus einer Lernstichprobe ermittelt werden.

Kostenfunktionen Die vorgestellten Partitionierungsmodelle stellen erst einmal nur Strukturen zur Modellierung der Entscheidungsfunktion $h(\boldsymbol{x})$ zur Verfügung. Zur Konstruktion eines Klassifikators müssen nun deren Parameter unter Berücksichtigung der Lernstichprobe eingestellt werden. Bei einigen Verfahren geschieht dies auf Basis von Heuristiken (siehe NN Entscheidungsregel). Die meisten modernen Verfahren definieren jedoch eine Kostenfunktion, bzw. ein statistisches Modell, welches es zu optimieren gilt. Die zu ermittelnde a-Posteriori Wahrscheinlichkeit kann in Abhängigkeit von der Lernstichprobe über die Kostenfunktion $L(\boldsymbol{x}, y, \boldsymbol{\varphi})$ ohne Beschränkung der Allgemeinheit als Boltzmann Verteilung umformuliert werden:

$$p(y|\boldsymbol{x}, \boldsymbol{\varphi}) \propto e^{-L(\boldsymbol{x},y,\boldsymbol{\varphi})} \,. \tag{2.22}$$

Beide Darstellungen sind somit äquivalent. Das theoretische Grundgerüst der empirischen Risikominimierung besagt, dass das Ziel des überwachten Lernens die Minimierung des mittleren Entscheidungsrisikos durch Minimierung der erwarteten Kosten ist [133], also

$$\arg\min_{\boldsymbol{\varphi}\in\mathcal{P}} E[L(\boldsymbol{x}, y, \boldsymbol{\varphi})] = \int\int p(\boldsymbol{x}, y) \cdot L(\boldsymbol{x}, y, \boldsymbol{\varphi})\, d\boldsymbol{x}\, dy \,. \tag{2.23}$$

Zur Vereinfachung des Problems wird die Annahme getroffen, dass die Lernstichprobe Tr eine repräsentative Stichprobe aus der Verteilung $p(\boldsymbol{x}, y)$ ist. In diesem Fall wird das erwartete Risiko durch das empirische Risiko approximiert:

$$E[L(\boldsymbol{x}, y, \boldsymbol{\varphi})] \approx \frac{1}{N} \sum_{i=1}^{N} L(\boldsymbol{x}_i, y_i, \boldsymbol{\varphi}) \,. \tag{2.24}$$

Im Folgenden werden einige verbreitete Kostenfunktionen vorgestellt, welche als Grundlage in allen gängigen überwachten Lernverfahren Anwendung finden.

Ein einfaches Kostenmodell stellen die 0-1 Kosten dar:

$$L_{0-1}(\boldsymbol{x}, y, \boldsymbol{\varphi}) = \begin{cases} 1 & \text{, falls } h(\boldsymbol{x}) \neq y \\ 0 & \text{, sonst} \,. \end{cases} \tag{2.25}$$

Jedes falsch klassifizierte Beispiel der Lernstichprobe erzeugt konstante Kosten, unabhängig von seiner sonstigen Relation zur durch $\boldsymbol{\varphi}$ beschriebenen Trennebene. Die Gesamtkosten sind somit direkt durch den empirischen Fehler auf der Lernstichprobe gegeben. Dies macht die 0-1 Kosten insbesondere sehr robust

gegenüber Ausreißern. Allerdings kann auch gezeigt werden, dass selbst für einfache Partitionierungsmodelle, wie der linearen Trennebene, die Optimierung der 0-1 Kosten bei nicht-separierbaren Lernstichproben NP-hart ist [88]. Aus diesem Grund wurden diverse andere Kostenfunktionen entwickelt, welche eine konvexe Relaxierung der 0-1 Kosten zum Ziel haben. Für Zweiklassen Probleme ($y \in \{-1, +1\}$) sind drei solcher Kostenfunktionen die Hinge-Kosten, die modifizierten Huber-Kosten und die logistischen Kosten [108]:

$$L_{\text{hinge}}(\boldsymbol{x}, y, \boldsymbol{\varphi}) = \max(0, 1 - y \cdot d(\boldsymbol{x})) \tag{2.26}$$

$$L_{\text{huber}}(\boldsymbol{x}, y, \boldsymbol{\varphi}) = \begin{cases} \max(0, 1 - y \cdot d(\boldsymbol{x}))^2 & \text{, falls } y \cdot d(\boldsymbol{x}) \geq -1 \\ -4 \cdot y \cdot d(\boldsymbol{x}) & \text{, sonst} \end{cases} \tag{2.27}$$

$$L_{\text{logistic}}(\boldsymbol{x}, y, \boldsymbol{\varphi}) = \log_2(1 + e^{-y \cdot d(\boldsymbol{x})}) \,. \tag{2.28}$$

Diese Funktionen stellen die Kosten eines Beispiels in Relation zu seiner Distanz zur Trennebene. Falsch klassifizierte Beispiele erzeugen somit deutlich höhere Kosten, falls sie sich außerdem weit weg von der Trennebene befinden. Diese Logik ist damit begründet, dass $d(\boldsymbol{x})$ auch als Konfidenz der Entscheidung interpretiert werden kann. Fehlklassifikationen sollten somit weniger stark gewichtet werden, falls das entsprechende Entscheidungsmodell $\boldsymbol{\varphi}$ dieser Entscheidung nur eine geringe Aussagekraft zuschreibt. Alle vier vorgestellten Kostenmodelle werden in Abbildung 2.4 gegenüber gestellt.

Die bereits erwähnte Konvexität des Hinge-Loss, Huber-Loss und Logistic-Loss erleichtert die Optimierung dieser Kostenfunktionen durch gradientenbasierte Optimierungsverfahren. Die Verwendung der Distanz als Gewichtungsfaktor sorgt allerdings auch dafür, dass Ausreißer einen übermäßig großen Einfluss auf das gelernte Entscheidungsmodell haben können. Weiterhin existiert aufgrund der im Abschnitt 2.2.2 näher ausgeführten Überlegungen eine Präferenz für einfache Entscheidungsfunktionen. Diese beiden Umstände können durch Einführung einer geeigneten a-Priori Verteilung $p(\boldsymbol{\varphi})$ in die a-Posteriori Verteilung aus Formel (2.22) berücksichtigt werden:

$$p(y|\boldsymbol{x}, \boldsymbol{\varphi}) \propto p(\boldsymbol{x}|y, \boldsymbol{\varphi}) \cdot p(\boldsymbol{\varphi}) \propto e^{-L(\boldsymbol{x}, y, \boldsymbol{\varphi}) + R(\boldsymbol{\varphi})} \,. \tag{2.29}$$

Die Funktion $R(\boldsymbol{\varphi})$ wird in dieser Darstellung auch als Regularisierungsterm bezeichnet.

Optimierungsverfahren Je nach Kombination von Partitionierungsmodell und Kostenfunktion bieten sich unterschiedliche Verfahren zur Optimierung einer Entscheidungsfunktion an. In frühester Vergangenheit wurde für einfache künstliche neuronale Netze die Perzeptron Lernregel zum Erlernen einer linearen Trennebene bei gegebenen 0-1 Kosten eingeführt [85]. Dieser Ansatz beruht auf einem Gradientenabstiegsverfahren, bei dem die Suchrichtungen direkt aus den

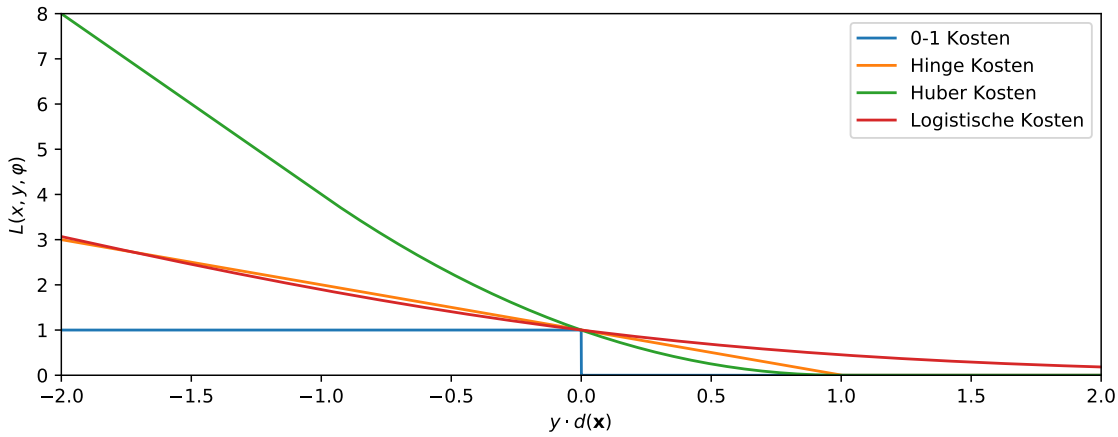


Abbildung 2.4: Vergleich von vier Kostenfunktionen für diskriminative Entscheidungsmodelle. Die Abszisse beschreibt die vorzeichenbehaftete Entfernung zur Trennebene. Auf der Ordinate stehen die Kosten für ein Lernbeispiel.

Merkmalsvektoren der Lernstichprobe abgeleitet werden. Dieser Algorithmus kann allerdings die Konvergenz nur für linear separierbare Datensätze garantieren. Um diesen Nachteil zu beheben wurden viele Varianten entwickelt, aus denen später die Perzeptron Lernregel mit maximaler Stabilität entstanden ist [110]. Dieses Verfahren ist ein direkter Vorgänger der heutzutage immer noch konkurrenzfähigen SVM [59]. Neben dem Gradientenabstiegsverfahren wurden auch Lösungen auf Basis von linearen Gleichungssystemen [127], linearer Optimierung [153] und quadratischer Optimierung [59] verfolgt. Der Classification and Regression Trees (CART) Algorithmus zum Erlernen von binären Entscheidungsbäumen [77] sowie Adaptive Boosting (AdaBoost) [45] verwenden iterative Greedy Algorithmen, für die gezeigt werden kann, dass sie durch sequentielle lokale Entscheidungen im Grenzfall eine gegebene globale Kostenfunktion optimieren. Abschließend können aufgrund der äquivalenten Formulierung von Kostenfunktionen durch Wahrscheinlichkeitsverteilungen auch statistische Verfahren für die Parameterschätzung oder Modellselektion angewendet werden. Hier sind insbesondere Verfahren zu nennen, welche auf einer Stichprobenwiederholung der Lernstichprobe basieren, wie z. B. dem Random Forest Algorithmus [14]. Weiterhin können Optimierungsprobleme in dieser Darstellung als ML, MAP oder als Erwartungswertschätzer formuliert werden, welche entweder approximativ durch Variationsansätze [52] oder durch Simulationsverfahren [87] gelöst werden können. Der *Backpropagation* Algorithmus, welcher zum Erlernen von künstlichen neuronalen Netzen verwendet wird, gehört zu den Gradientenabstiegsverfahren. Aufgrund der starken Nichtlinearitäten innerhalb solcher Netzwerke müssen Implementierungen dieses Verfahrens

jedoch zusätzlich Probleme von verschwindenen oder explodierenden Gradienten berücksichtigen sowie spezielle problemangepasste Regularisierungsansätze zur Vermeidung von Überanpassung (siehe folgender Abschnitt) bereitstellen.

Generalisierung auf ungesehene Testdaten

Die Entscheidungsfunktion wird einzig durch Berücksichtigung einer Lernstichprobe, gezogen aus einer ansonsten unbekannten Grundgesamtheit, angelernt. In der Anwendung soll aber mit einer ausreichend hohen Konfidenz garantiert werden, dass gleichzeitig auch die Klassifikationsleistung auf der Grundgesamtheit maximiert wird. Hierfür werden von der empirischen Risikominimierung [133] zwei wesentliche Annahmen gemacht:

1. Die Lernstichprobe Tr muss eine repräsentative Stichprobe aus $p(\boldsymbol{x}, y)$ sein, d. h. $\mathrm{Tr} \sim p(\boldsymbol{x}, y)$.
2. Der empirische Schätzer (Formel 2.24) ist eine gute Approximation des erwarteten Risikos.

Punkt 1 ist in der Regel für ein Klassifikationsproblem nicht garantierbar. Die Lernstichprobe wird meistens manuell durch menschliche Bearbeiter mit einem unbekannten Bias erstellt. Dieser Bias kann z. B. in der Form einer Über- oder Unterrepräsentation einzelner Klassen oder einer Über- oder Unterrepräsentation einzelner Regionen im Merkmalsraum zum Vorschein treten. Es existieren Ansätze, diese Verzerrung der Lernstichprobe durch Umgewichtung anhand einer repräsentativen ungelabelten Stichprobe aus $p(\boldsymbol{x})$ zu kompensieren (*Covariate Shift* [126]). Allgemeiner werden jedoch diskriminative Lernverfahren gewählt, welche robuster gegenüber Verletzungen der ersten Annahme sind [11]. Für repräsentative Stichproben hängt der Approximationsfehler des empirischen Risikos in Punkt 2 im Wesentlichen von der Stichprobengröße in Relation zur Komplexität des Klassifikationsproblems ab. Das *Berry-Esseen Theorem* garantiert allerdings, dass die Konvergenzrate der Approximation mindestens von der Ordnung $1/\sqrt{N}$ ist [119]. Ein Vervierfachung der Lernstichprobengröße reduziert den Approximationsfehler somit im Mittel auf die Hälfte. Selbst schwierige Probleme sollten daher mit realistischen Stichprobengrößen lernbar sein.

Da die Einhaltung der fundamentalen Annahmen der empirischen Risikominimierung unter praktischen Bedingungen nicht garantiert werden kann, sollte die gelernte Entscheidungsfunktion in einem nachfolgenden Schritt zusätzlich validiert werden. Der Klassifikator wird hierfür auf die Grundgesamtheit angewendet und die resultierende Klassifikationsleistung durch eine geeignete Kenngröße ermittelt [68]. Wie bereits angemerkt wurde, ist häufig nur wenig Information über die Grundgesamtheit vorhanden. Die Überprüfung findet daher auf einer Teststichprobe Te statt, welche statistisch unabhängig von der Lernstichprobe sein muss. Der empirische Fehler auf Te wird dann als Generalisierungsfehler bezeichnet. Der Trainingsfehler sowie der

Generalisierungsfehler haben eine fundamentale Bedeutung in vielen Lernprinzipien. Dennoch konnte gezeigt werden, dass sich die Güte eines Klassifikators nicht objektiv durch eine einzelne Kenngröße bewerten lässt [43]. In der Literatur wurden daher diverse Maße vorgeschlagen:

Trainingsfehler Als Trainingsfehler wird der relative Klassifikationsfehler auf einer Lernstichprobe Tr bezeichnet. Es gilt

$$\epsilon(h, \mathrm{Tr}) = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}(h(\boldsymbol{x}_i) \neq y_i) \,. \tag{2.30}$$

Der Trainingsfehler wird synonym zum Begriff empirischer Fehler verwendet.

Generalisierungsfehler Als Generalisierungsfehler wird der relative Klassifikationsfehler auf einer Teststichprobe Te bezeichnet. Es gilt

$$\epsilon(h, \mathrm{Te}) = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}(h(\boldsymbol{x}_i) \neq y_i) \,. \tag{2.31}$$

Genauigkeit Als Genauigkeit (engl. *Accuracy*) wird die relative Anzahl von korrekt klassifizierten Beispielen auf einer Stichprobe T (im Allgemeinen die Teststichprobe) bezeichnet. Es gilt

$$acc(h, \mathrm{T}) = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}(h(\boldsymbol{x}_i) = y_i) = 1 - \epsilon(h, \mathrm{T}) \,. \tag{2.32}$$

Precision und Recall Die Genauigkeit kann bei unbalancierten Problemen, bei denen die einzelnen Klassen durch eine unterschiedliche Anzahl von Lernbeispielen vertreten werden, zu unintuitiven Ergebnissen führen. Insbesondere werden Entscheidungsmodelle, welche sich immer für die dominierende Klasse entscheiden, zu positiv bewertet. In Zweiklassenproblemen wird die erste Klasse als Referenzklasse betrachtet. Dann können solche Situationen durch separate Beobachtung der Fehler erster und zweiter Art erkannt werden. Häufig werden für die Messung der Fehlerarten die Statistiken *Precision* und *Recall* verwendet:

$$precision(h, T) = p(y = +1 | h(\boldsymbol{x}) = +1) \tag{2.33}$$

$$recall(h, T) = p(h(\boldsymbol{x}) = +1 | y = +1) \,. \tag{2.34}$$

Um dennoch die Leistung eines Klassifikators durch einen einzelnen Wert zu beschreiben, können diese beiden Kenngrößen durch Berechnung ihres harmonischen Mittels zum sogenannten *F1-Score* zusammengefasst werden:

$$F_1(h, T) = 2 \frac{precision(h, T) \cdot recall(h, T)}{precision(h, T) + recall(h, T)} \,. \tag{2.35}$$

Cohen's Kappa Für unbalancierte Multiklassenprobleme wurde von Jacob Cohen die Kappa Statistik vorgeschlagen [27]. Gegeben seien zwei Entscheidungsfunktionen h_1 und h_2. Die Kappa Statistik misst dann den Grad der Übereinstimmung zwischen den Funktionen auf einer gegebenen Stichprobe T. Es gilt

$$\begin{aligned}\kappa(h_1, h_2, \mathrm{T}) &= \frac{p_0(h_1, h_2, \mathrm{T}) - p_e(h_1, h_2, \mathrm{T})}{1 - p_e(h_1, h_2, \mathrm{T})} \\ p_0(h_1, h_2, \mathrm{T}) &= \frac{|\{\boldsymbol{x} \mid \boldsymbol{x} \in \mathrm{T} \wedge h_1(\boldsymbol{x}) = h_2(\boldsymbol{x})\}|}{|\mathrm{T}|} \\ p_e(h_1, h_2, \mathrm{T}) &= \sum_{y \in \mathcal{K}} \frac{|\{\boldsymbol{x} \mid \boldsymbol{x} \in \mathrm{T} \wedge h_1(\boldsymbol{x}) = y\}|}{|\mathrm{T}|} \cdot \frac{|\{\boldsymbol{x} \mid \boldsymbol{x} \in \mathrm{T} \wedge h_2(\boldsymbol{x}) = y\}|}{|\mathrm{T}|} .\end{aligned} \tag{2.36}$$

Der Wert p_0 beschreibt die relative Übereinstimmung zwischen beiden Funktionen und p_e die Wahrscheinlichkeit der zufälligen Übereinstimmung. Somit gibt κ den Grad der Übereinstimmung an, welcher über den puren Zufall hinausgeht. Um diese Statistik als Bewertungsmaß für Klassifikatoren anzuwenden kann h_2 als optimale Entscheidungsfunktion angenommen werden, d. h. $h_2(\boldsymbol{x}_i) = y_i$ für alle $(\boldsymbol{x}_i, y_i) \in \mathrm{T}$.

Die Annahme 2 der empirischen Risikominimierung kann im Allgemeinen nur im Grenzfall $N \to \infty$, also für unendlich große Lernstichproben, garantiert werden. Bei endlichen Stichprobengrößen muss dass Problem der Überanpassung berücksichtigt werden. Gegeben sei eine Lernstichprobe für ein beliebiges Klassifikationsproblem. Der Trainingsfehler kann für jedes N unter eine beliebige Schranke gebracht werden, indem die Modellkomplexität der Entscheidungsfunktion genügend hoch gesetzt wird. Dies führt dazu, dass statt der zugrundeliegenden Verteilung nur die stichprobenspezifischen Eigenheiten gelernt werden. In Abbildung 2.5 wird ein Extrembeispiel für ein solche Überanpassung gezeigt. Es kann beobachtet werden, dass mit zunehmender Überanpassung gleichzeitig der Generalisierungsfehler zunimmt. Abbildung 2.6 zeigt das generelle Verhältnis zwischen diesen beiden Größen. Die strukturelle Risikominimierung wurde von Vapnik und Chervonenkis als Erweiterung der empirischen Risikominimierung entwickelt, welche dieses Verhalten für endliche Stichprobengrößen explizit berücksichtigt [133]. Um die Überanpassung zu vermeiden wird ein Bestrafungsterm für die Modellkomplexität eingeführt. Aus der Modellselektion war dieses Konzept unter den Begriffen Akaike Information Criterion (AIC) [2] und Bayesian Information Criterion (BIC) [115] bereits bekannt. Hier wurde allerdings die Modellkomplexität lediglich als Funktion der Anzahl an Modellparametern definiert. Für die strukturelle Risikominimierung wurde die Vapnik-Chervonenkis (VC) Dimension als Komplexitätsmaß für Klassifikationsmodelle vorgeschlagen. Die VC Dimension eines Modells entspricht der größten Anzahl N an Lernbeispielen, welche in beliebiger Lage und mit einer beliebigen Zuweisung von Klassenlabels durch dieses Modell mit einem Trainingsfehler von 0 gelernt werden können. Man sagt dann, dass das Modell

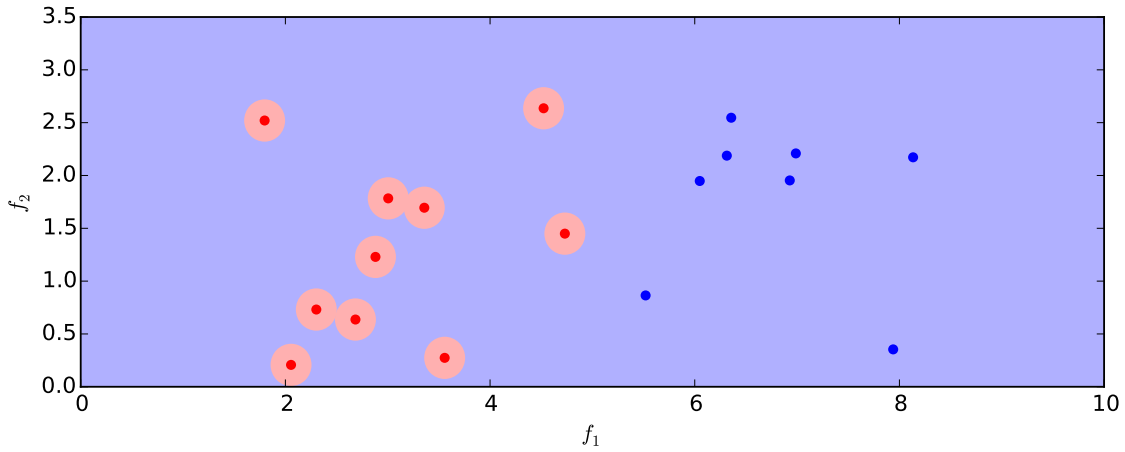


Abbildung 2.5: Beispiel für ein Entscheidungsmodell mit starker Überanpassung. Das Modell entscheidet sich nahezu für alle Merkmalsvektoren für die Klasse 2 (blau). Nur in direkter Nähe zu Beispielen aus der Lernstichprobe kann die Entscheidung auf die Klasse 1 (rot) fallen.

die Stichprobe zerschmettert (engl. *Shattering*). Falls die Klassenkonzepte in dem gegebenen Merkmalsraum trennbar sind und Ausreißer in den Daten ausgeschlossen werden können, dann sollte das Modell mit der geringsten VC Dimension gewählt werden, welches die Lernstichprobe zerschmettern kann. Im Allgemeinen sind die Klassenkonzepte jedoch nicht perfekt trennbar, daher sollte die Modellkomplexität stärker bestraft werden. In der Praxis geschieht dies durch einen zusätzlichen Regularisierungsterm, welcher durch Validierung auf einer zurückgehaltenen Stichprobe eingestellt werden muss.

Lernen von Multiklassenproblemen

Viele Entscheidungsmodelle unterstützen das Erlernen von Multiklassenproblemen, für die $|\mathcal{K}| > 2$ gilt, bereits nativ. Zu diesen gehören alle generativen Modelle [12], die NN Entscheidungsregel [12] und Entscheidungsbäume [102] [77]. Für einige wichtige und erfolgreiche diskriminative Modelle ist dies jedoch nicht der Fall. Das Hauptproblem besteht darin, dass die Zielvariable y nominalskaliert. Für die betroffenen Verfahren gilt, dass diese als vereinfachende Annahme die Zielvariable als ordinalskalierend betrachten. D. h. es wird angenommen, dass Klassen mit ähnlichen Klassenlabels auch ähnliche Konzepte repräsentieren. Z. B. würde sich eine so gelernte Entscheidungsfunktion, welche Anzeichen für die Klassen $y = 1$ und $y = 3$ beobachtet, letztendlich für die Klasse $y = 2$ entscheiden. Für Zweiklassenprobleme stellt diese Annahme keine Beschränkung dar, führt aber bei steigender Anzahl von Klassenlabels zu verzerrten Entscheidungsmodellen. Um dieses Problem zu korrigieren muss unterschieden werden, ob dessen Ursache im Partitionierungsmodell

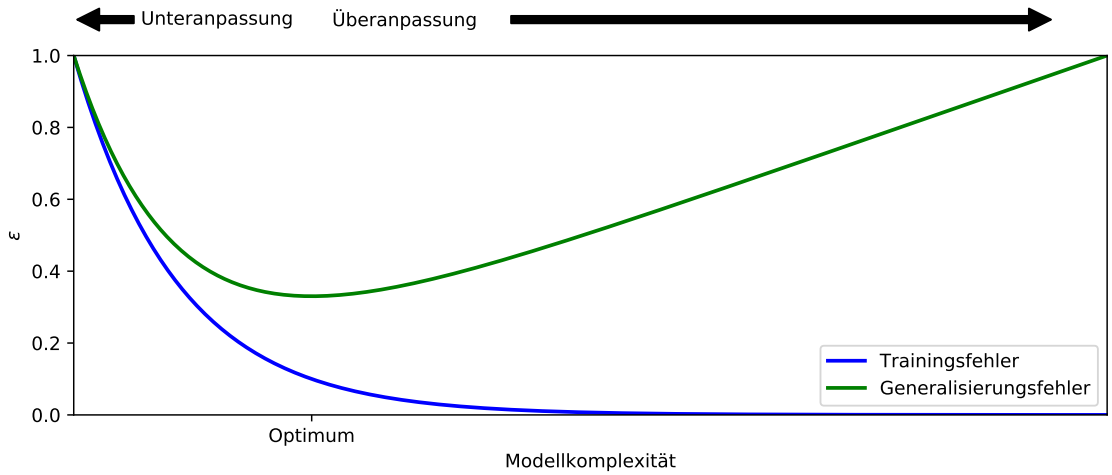


Abbildung 2.6: Im Allgemeinen kann für jede beliebige Lernstichprobe ein komplexes Entscheidungsmodell generiert werden, welches den Trainingsfehler ϵ auf 0 reduziert. Übersteigt die Komplexität des Modells allerdings die Komplexität des Problems, dann steigt der Generalisierungsfehler. Dieser Fall wird als Überanpassung bezeichnet. Der gegenläufige Fall, wenn das Entscheidungsmodell simpler als das Problem ist, heißt Unteranpassung.

oder in der Kostenfunktion zu finden ist. Für den ersten Fall kann die Zielvariable y durch die Zielvariablen $y^j, j \in \mathcal{K}$ ersetzt werden. Hier beschreibt $y^j \in \{-1, +1\}$ die binäre Entscheidung, ob ein Merkmalsvektor zur Klasse j gehört oder nicht und $d^j(\boldsymbol{x})$ eine entsprechende vorzeichenbehaftete Konfidenzfunktion zu dieser Entscheidung. Die Konfidenzen können durch Anwendung der *SoftMax* Funktion [12] zu einer gemeinsamen Entscheidung kombiniert werden:

$$p(y = j|\boldsymbol{x}, \boldsymbol{\varphi}) = \frac{e^{d^j(\boldsymbol{x})}}{\sum_{k \in \mathcal{K}} e^{d^k(\boldsymbol{x})}} \,. \tag{2.37}$$

Obwohl das Gesamtproblem in binäre Teilprobleme zerlegt wird, müssen alle Teilprobleme durch gemeinsame Optimierung der Gesamtkostenfunktion gelernt werden. Dieser Ansatz wird z. B. bei künstlichen neuronalen Netzen und in der logistischen Regression verfolgt. Falls die Kostenfunktion selbst nicht auf Multiklassenprobleme erweiterbar ist, muss ein anderes Vorgehen gewählt werden. Die zwei verbreitetsten Strategien sind die One-vs-Rest (OVR) und One-vs-One (OVO) Strategie [46]. Auch sie zerlegen das Gesamtproblem in binäre Entscheidungsprobleme, allerdings werden die Teilprobleme als separate Zweiklassenprobleme optimiert. Die Strategien lassen sich im Detail wie folgt beschreiben:

One-vs-Rest Für jedes Klassenlabel $j \in \mathcal{K}$ wird ein Zweiklassenproblem $p^j(y|\boldsymbol{x})$ mit den Klassen $\mathcal{K}^j = \{+1 = j, -1 = \mathcal{K} \setminus j\}$ gelöst. Die finale Entscheidung

fällt auf das Klassenlabel, dessen Teilproblem die höchste Wahrscheinlichkeit prädiziert, also

$$y = \arg\max_{j \in \mathcal{K}} p^j(y = +1|\boldsymbol{x}) \,.$$

One-vs-One Für jedes unterscheidbare Paar von Klassenlabels $(i, j), i,j \in \mathcal{K}, i \neq j$ wird ein Zweiklassenproblem mit den Klassen $\mathcal{K}^{i,j} = \{i, j\}$ gelöst. Die finale Entscheidung fällt durch eine Mehrheitsentscheidung, also

$$y = \arg\max_{j \in \mathcal{K}} \sum_{(i,j) \in \mathcal{K}, i \neq j} \mathbb{I}(j = h^{i,j}(\boldsymbol{x})) \,.$$

In Untersuchungen konnte gezeigt werden, dass von diesen beiden Ansätzen die OVO Strategie häufig die besseren Resultate erzielt [65].

Probabilistische Entscheidungen

Im Kontext dieser Arbeit müssen Wahrscheinlichkeitsverteilungen und Konfidenzfunktionen klar voneinander abgegrenzt werden. Generative Entscheidungsmodelle erzeugen immer echte a-Posteriori Wahrscheinlichkeiten. Bei diskriminativen Modellen ist allerdings häufig nur eine Konfidenzfunktion, wie die Distanz zur Trennebene, gegeben. Als eine der wenigen Ausnahmen sei hier die logistische Regression zu nennen. In vielen Anwendungen ist der Unterschied nur von geringer Bedeutung. In der Bayesschen Statistik werden allerdings Wahrscheinlichkeiten mit Vorwissen in Form von a-Priori Verteilungen kombiniert um so ein besseres Modell der zur Verfügung stehenden Informationen zu generieren. Konfidenzfunktionen wären hierfür ungeeignet, da diese in der Regel unkalibriert sind und nur bis auf eine Skalierung und Verschiebung bestimmbar sind. Das kombinierte Modell würde daher von diesen undefinierten Parametern abhängen. Dennoch gilt, dass jede Konfidenzfunktion durch eine monotone Abbildung in eine gültige Wahrscheinlichkeitsverteilung umgerechnet werden kann. Platt hat diesen Umstand ausgenutzt um eine Kalibrierung der Konfidenzfunktion durch Einpassung einer Sigmoidfunktion in aus der Lernstichprobe abgeleiteten a-Posteriori Verteilung zu erreichen [98]. Dieses Vorgehen entspricht in etwa dem Anlernen eines logistischen Regressionsmodells, wobei die Konfidenzwerte die Rolle der Merkmale einnehmen. Die Gültigkeit dieser Methode konnte empirisch für viele überwachte Lernverfahren, wie z. B. SVMs und AdaBoost gezeigt werden [89].

Weiterhin existiert die Notwendigkeit, a-Posteriori Wahrscheinlichkeiten für Multiklassenprobleme zu erzeugen. Durch die vorgestellten Methoden ist es zunächst möglich, Wahrscheinlichkeitsverteilungen für die binären Zweiklassenprobleme zu

schätzen. Die Wahrscheinlichkeiten der Klassenlabels können dann durch Kombination dieser Verteilungen approximiert werden. Für die oVr Strategie gilt:

$$p(y = i|\boldsymbol{x}) = \frac{p^i(y = +1|\boldsymbol{x})}{\sum_{k\in\mathcal{K}} p^k(y = +1|\boldsymbol{x})}\,. \tag{2.38}$$

Für die oVo Strategie wurden diverse Methoden vorgeschlagen. Beispielhaft soll hier das Verfahren von Price et al. [101] genannt werden. Hier wird durch die Nebenbedingung, dass die Wahrscheinlichkeiten der Klassenlabels zu 1 summieren müssen, eine Invariante zwischen der a-Posteriori Verteilung und der paarweisen Wahrscheinlichkeitsverteilungen hergestellt. Durch Umformung dieser Invariante entsteht folgende Beziehung:

$$p(y = i|\boldsymbol{x}) = \left(\sum_{j\in\mathcal{K},i\neq j} \frac{1}{p^{i,j}(y = i|\boldsymbol{x})} - (|\mathcal{K}| - 2)\right)^{-1}\,. \tag{2.39}$$

In Experimenten hat sich diese einfache Methode als sehr konkurrenzfähig erwiesen [145].

2.2.3 Unüberwachtes Lernen

Falls keine, bzw. nur unzuverlässige, Klassenlabels verfügbar sind, kommen unüberwachte Lernverfahren zum Einsatz. Die Eingabe für diese Methoden ist eine ungelabelte Lernstichprobe $\mathrm{Tr} = \{\boldsymbol{x}_i\}_{i=1}^N$. Die darin enthaltenen Informationen sind für einige Anwendungen, wie der Merkmalsselektion [55], Ausreißerdetektion [155], Hauptkomponentenanalyse (engl. *Principal Component Analysis*, pca) [12] oder Dimensionsreduktion [44], bereits ausreichend. Für das Erlernen von Entscheidungsfunktionen muss allerdings zusätzlich starkes a-Priori Wissen über die Verteilung der Lernbeispiele angenommen werden. Die meisten Ansätze basieren auf Annahmen, welche den Klassen eine Kompaktheit zuschreiben. Diese Annahmen lassen sich in der Literatur unter diversen Begriffen wiederfinden [21]:

Clusterannahme Die Klassen sind kompakt und gut separiert. Daraus folgt, dass die mittlere Distanz von Beispielen derselben Klasse geringer ist als die Distanz von Beispielen unterschiedlicher Klassen. Weiterhin gilt, dass die Trennebene der Separierung von Klassen in Bereichen geringer Dichte liegt.

Glattheitsannahme Beispiele einer Klasse sollten in der Nähe von anderen Beispielen derselben Klasse liegen. Die Klassen sind nicht notwendigerweise kompakt, werden aber von Bereichen geringer Dichte separiert.

Mannigfaltigkeitsannahme Beispiele derselben Klasse liegen auf einer Mannigfaltigkeit mit deutlich geringerer Dimensionalität als der Merkmalsraum. Diese Annahme wird insbesondere bei hochdimensionalen Merkmalsräumen zur Bekämpfung des „Fluchs der Dimensionalität“ [67] eingeführt.

Unüberwachte Verfahren zum Erlernen von Entscheidungsfunktionen werden auch als Clusteringverfahren bezeichnet. Die gebräuchlichen Ansätze sind hier *K-Means Clustering* [80], Hierarchisches Clustering [107], *Spectral Clustering* [138] und *Subspace Clustering* [94].

2.2.4 Nichtlineare Entscheidungsmodelle durch den Kernel Trick

Durch parameterfreie Modelle können Entscheidungsmodelle beliebiger Komplexität dargestellt werden. Allerdings besitzen Methoden basierend auf parametrisierten Modellen in der Regel ausgereiftere Implementierungen, und ihnen können aus der strukturellen Risikominimierung bessere Generalisierungseigenschaften zugeschrieben werden [133]. Der *Kernel Trick* ist ein Werkzeug um einige Klassen von parametrisierten Modellen in parameterfreie Modelle zu transformieren um somit die Eigenschaften beider Ansätze zu kombinieren. Dieses Vorgehen soll hier am Beispiel von linearen Trennebenen näher erläutert werden. Die allgemeine parametrisierte Form für eine lineare Trennebene lautet:

$$d(\boldsymbol{x}) = \langle \boldsymbol{x}, \boldsymbol{\nu} \rangle - b\,. \tag{2.40}$$

Weiterhin soll gelten, dass sich die Lernbeispiele aus Tr in allgemeiner Lage befinden und $N \geq D$. Zu jeder Dimension des Merkmalsraums existiert somit mindestens ein Lernbeispiel, welches sich nicht als Linearkombination von anderen Lernbeispielen darstellen lässt. Da die Lernsbeispiele somit den gesamten Merkmalsraum aufspannen, lässt sich die Trennebene nach dem *Representer Theorem* [112] äquivalent als Linearkombination der Lernstichprobe darstellen:

$$d(\boldsymbol{x}) = \sum_{i=1}^{N} w_i \cdot \left(\sum_{j=1}^{D} \boldsymbol{x}_j \cdot \boldsymbol{x}_{i,j} \right) - b = \sum_{i=1}^{N} w_i \cdot \langle \boldsymbol{x}, \boldsymbol{x}_i \rangle - b\,. \tag{2.41}$$

Die Parameter $\boldsymbol{w} = (w_i)_{i=1}^{N}$ beschreiben die Trennebene als parameterfreies Modell. Diese Reparametrisierung hat aufgrund der genannten Äquivalenzbeziehung erst einmal keinen Einfluss auf die darstellbare Modellkomplexität. Als nächstes wird daher die Abbildung $\phi : \mathcal{F} \to \mathcal{H}$ eingeführt, welche einen Merkmalsvektor aus dem ursprünglichen Merkmalsraum $\mathcal{F}$ in einen Merkmalsraum $\mathcal{H}$ überführt. Die Abbildung 2.7 zeigt, wie durch solch eine typischerweise nichtlineare Transformation komplexe Entscheidungsprobleme lösbar werden. Bei $\mathcal{H}$ handelt es sich um einen Hilbertraum, d. h. die Norm wird in diesem Raum eindeutig durch sein Skalarprodukt induziert. Diese Eigenschaft kann ausgenutzt werden um explizite Berechnungen von $\phi(\boldsymbol{x})$ zu vermeiden. Hierfür muss die Gleichung (2.41) durch Einführung der Kernelfunktion $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \rangle$ umgeschrieben werden:

$$d(\boldsymbol{x}) = \sum_{i=1}^{N} w_i \cdot k(\boldsymbol{x}, \boldsymbol{x}_i) - b\,. \tag{2.42}$$

Der Satz von Mercer [113] definiert eine hinreichende Bedingung für gültige Kernelfunktionen. Somit ist es möglich Kernelfunktionen zu deklarieren, deren assoziierte Transformation des Merkmalsraums hochkomplex oder sogar unbekannt ist. Zu den populärsten Kernelfunktionen zählen:

$$
\begin{aligned}
k_{\text{linear}}(\boldsymbol{x}_i, \boldsymbol{x}_j) &= \langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle && \text{(Linearer Kernel)} \\
k_{\text{poly}}(\boldsymbol{x}_i, \boldsymbol{x}_j) &= (\langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle + \alpha)^{\beta} && \text{(Polynomieller Kernel)} \\
k_{\text{stump}}(\boldsymbol{x}_i, \boldsymbol{x}_j) &= -\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_1 && \text{(Entscheidungsstumpf Kernel)} \\
k_{\text{RBF}}(\boldsymbol{x}_i, \boldsymbol{x}_j) &= e^{-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{2\sigma^2}} . && \text{(RBF Kernel)}
\end{aligned}
$$

Insbesondere der Radial-Basis-Function-Kernel (RBF-Kernel) ist von spezieller Bedeutung, da er es bei geschickter Wahl des Hyperparameters σ ermöglicht, Entscheidungmodelle beliebiger Komplexität zu modellieren. In der Literatur wurden weiterhin diverse Kernelfunktionen vorgeschlagen, welche es ermöglichen, strukturierte Merkmalsräume (Graphen) [47] oder Merkmalsräume dynamischer Größe (Strings) [76] zu behandeln.

Der Kernel Trick ist auf viele lineare Modelle anwendbar, wie z. B. der logistischen Regression [154], linearen Regression [120], PCA [114] und kanonischen Korrelationsanalyse [57]. Durch die Beziehung

$$
d_2^2(\boldsymbol{x}_i, \boldsymbol{x}_j) = \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2 = \langle \boldsymbol{x}_i, \boldsymbol{x}_i \rangle + \langle \boldsymbol{x}_j, \boldsymbol{x}_j \rangle - 2\langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle \tag{2.43}
$$

lässt sich der Trick weiterhin auf einige distanzbasierte nichtlineare Modelle anwenden. Hierzu zählen u. a. *K-Means Clustering* [37] und der Support Vektor Domänen Deskriptor [128].

2.3 Transferlernen

Eine sehr fundamentale Annahme, welche von traditionellen maschinellen Lernverfahren getroffen wird, ist, dass die Lernstichprobe Tr aus derselben Verteilung gezogen wurde wie die zukünftigen Merkmalsvektoren, welche es zu klassifizieren gilt. Bereits vor Benennung und Formalisierung der Problematik des Transferlernens hatten sich viele Veröffentlichungen mit dem Teilproblem der Stichprobenverzerrung, und insbesondere mit dem *Covariate Shift* [126], befasst. In diesem Szenario wird angenommen, dass sich die Verteilungen von Tr und Te im Term $p(\boldsymbol{x})$ unterscheiden. Dies würde bedeuten, dass in der Lernstichprobe bestimmte Regionen im Merkmalsraum über- oder unterrepräsentiert sind. Beim Transferlernen wird diese Annahme weiter aufgelockert, z. B. indem sich auch die Verteilung $p(y|\boldsymbol{x})$ verändern darf. Neben dem zu lösenden Entscheidungsproblem der Klassifikation, welches nach Abschnitt 2.2 definiert ist, sollen noch weitere Daten vorhanden sein, welche eventuell aus einer beliebigen anderen Verteilung gezogen wurden oder sogar einen anderen

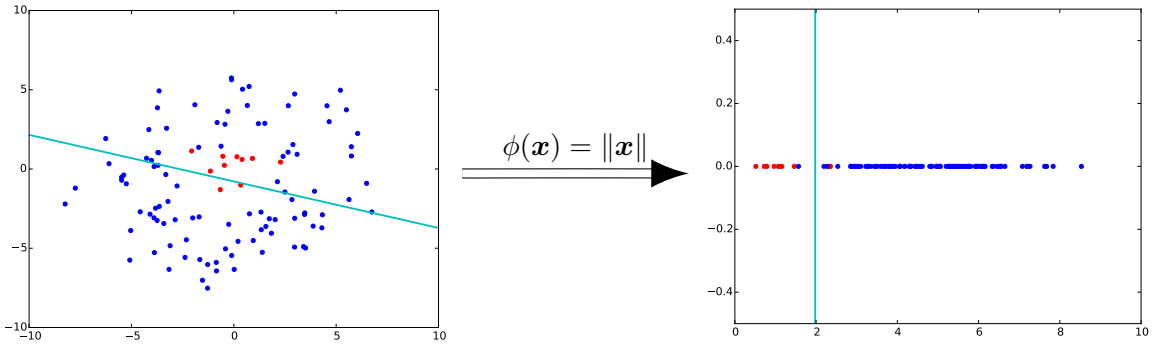


Abbildung 2.7: Vergleich eines linearen Entscheidungsmodells im originalen Merkmalsraum $\mathcal{F}$ und nach einer nichtlinearen Abbildung $\phi(\boldsymbol{x})$ in einen diskriminativeren Merkmalsraum $\mathcal{H}$. Der Trainingsfehler kann so von nahezu 50% auf beinahe 0% reduziert werden. Durch Anwendung des Kernel Tricks muss diese Abbildung nicht explizit durchgeführt werden.

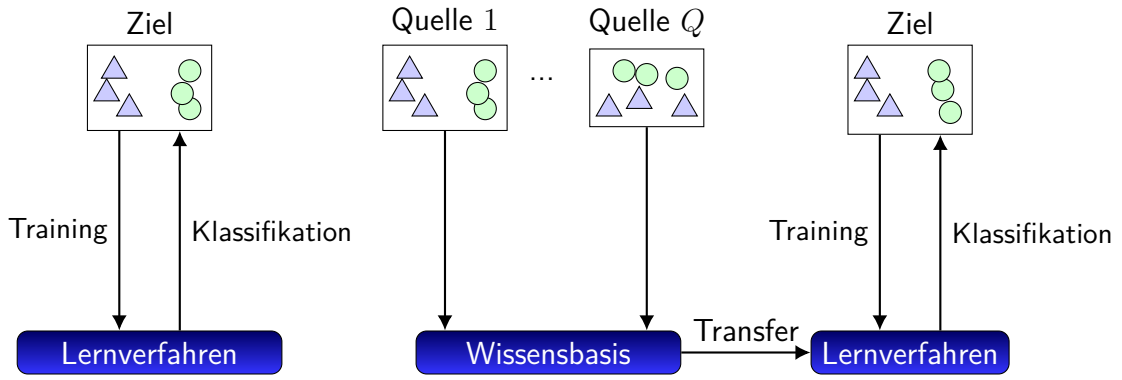
Merkmalsraum aufweisen. Für die genaue Definition der neuen Problemstellung des Transferlernens sollen zunächst einige Begriffe und Symbole nach Pan und Yang [92] eingeführt werden.

Definition 3 (Klassifikationsproblem)**.** *Ein Klassifikationsproblem besteht aus einem Merkmalsraum $\mathcal{F} \subseteq \mathbb{R}^D$ der Dimension D, der Menge der Klassenlabels $\mathcal{K}$ und der Entscheidungsfunktion $y = h(\boldsymbol{x})$. Weiterhin existiert entweder eine gelabelte Lernstichprobe* $\mathrm{Tr} = \{\boldsymbol{x}_i, y_i\}_{i=1}^N$*, welche aus der Verteilung $p(\boldsymbol{x}, y)$ gezogen wurde, oder eine ungelabelte Lernstichprobe* $\mathrm{Tr} = \{\boldsymbol{x}_i\}_{i=1}^N$*, welche aus $p(\boldsymbol{x})$ erzeugt wurde.*

Definition 4 (Transferlernen)**.** *Das Transferlernen hat die Aufgabe, den Generalisierungsfehler für ein Klassifikationsproblem T zu minimieren. Dabei sollen Informationen aus einem ähnlichen, aber typischerweise unterschiedlichen, Klassifikationsproblem S verwendet werden. Im Folgenden wird T als Ziel und S als Quelle bezeichnet. Wird Wissen aus mehreren Quellen $\{S_1, \cdots, S_Q\}$ transferiert, so spricht man von einem Multiquellentransfer.*

Die Beziehung zwischen dem Transferlernen und klassischen überwachten Lernverfahren wird in Abbildung 2.8 schematisch dargestellt. Sowohl bei dem Ziel als auch bei der Quelle handelt es sich um vollwertige Klassifikationssprobleme, welche unterschiedlich sein können, sich aber aus denselben Komponenten zusammensetzen. Beiden Problemen wird allerdings eine unterschiedliche Bedeutung beigemessen, wodurch eine Asymmetrie entsteht.

Im Transferlernen können die Unterschiede zwischen Ziel und Quelle entweder in den Daten selbst bestehen oder in der Definition des Entscheidungsproblems. Diese Konzepte werden unter den Begriffen der Domäne und der Aufgabe zusammengefasst.



(a) Überwachtes Lernen (b) Transferlernen mit Q Quellen

Abbildung 2.8: Schematische Darstellung des (a) überwachten Lernens und einem (b) verallgemeinerten Multiquellentransfer.

Definition 5 (Domäne)**.** *Die Domäne $\mathcal{D}$ eines Klassifikationsproblems setzt sich aus seinem Merkmalsraum und der Verteilungsfunktion der Daten zusammen. Es gilt $\mathcal{D} = (\mathcal{F}, p(\boldsymbol{x}))$.*

Definition 6 (Aufgabe)**.** *Die Aufgabe $\mathcal{T}$ eines Klassifikationsproblems setzt sich aus seinen Klassenlabels und der a-Posteriori Verteilung der Klassenlabels zusammen. Es gilt $\mathcal{T} = (\mathcal{K}, p(y|\boldsymbol{x}))$.*

Letztendlich wird der Wissenstransfer in unterschiedliche Transferszenarien eingeteilt, die sich in der Qualität der Lernstichproben unterscheiden, welche uns jeweils für das Ziel und für die Quellen zu Verfügung stehen. Die nachfolgende Kategorisierung stellt eine Erweiterung des Konzepts des überwachten und unüberwachten Lernens auf das Transferlernen dar.

Definition 7 (Unüberwachter Transfer)**.** *Für die Lernstichproben Tr_T und Tr_S sind jeweils die korrekten Klassenlabels unbekannt. Es gilt $\mathrm{Tr}_T = \{\boldsymbol{x}_{T,i}\}_{i=1}^{N_T}$ und $\mathrm{Tr}_S = \{\boldsymbol{x}_{S,i}\}_{i=1}^{N_S}$.*

Definition 8 (Induktiver Transfer)**.** *Für den Fall des induktiven Transfers können für die Quelle S die korrekten Klassenlabels optional vorhanden sein, d. h. es gilt entweder $\mathrm{Tr}_S = \{\boldsymbol{x}_{S,i}, y_{S,i}\}_{i=1}^{N_S}$ oder $\mathrm{Tr}_S = \{\boldsymbol{x}_{S,i}\}_{i=1}^{N_S}$. Für das Ziel T hingegen muss eine Lernstichprobe mit bekannten Klassenlabels vorhanden sein. Hier gilt $\mathrm{Tr}_T = \{\boldsymbol{x}_{T,i}, y_{T,i}\}_{i=1}^{N_T}$.*

Definition 9 (Domänenadaption)**.** *Für den Fall der Domänenadaptation muss für die Quelle S eine Lernstichprobe mit bekannten Klassenlabels vorliegen. Es gilt $\mathrm{Tr}_S = \{\boldsymbol{x}_{S,i}, y_{S,i}\}_{i=1}^{N_S}$. Für das Ziel T liegen keine Klassenlabels vor, d. d. es gilt $\mathrm{Tr}_T = \{\boldsymbol{x}_{T,i}\}_{i=1}^{N_T}$. Falls sich T und S nur in ihren Verteilungen der Merkmalsvektoren unterscheiden, d. h. $p_T(\boldsymbol{x}) \neq p_S(\boldsymbol{x})$, so spricht man von einer Domänenadaption.*

Von diesen Möglichkeiten wird sich diese Arbeit insbesondere mit der Domänenadaption mit ungelabelten Ziellernstichproben beschäftigen. Teile der entwickelten Verfahren werden jedoch auch einen unüberwachten Transfer einsetzen.

2.4 Simulationsverfahren

Die aus der Bayesschen Statistik abgeleiteten Methoden führen sehr häufig zu Modellen mit komplizierten Erwartungswerten und der Berechnung von hochdimensionalen Integralen. In der Bayesschen Entscheidungstheorie müssen z. B. mittlere Kosten der Form

$$f(\theta) = \mathrm{E}[L(\theta, x)|I] = \int L(\theta, x)\, p(x|I)\, dx \tag{2.44}$$

minimiert werden. Solche Probleme sind selten analytisch lösbar, was die Praktikabilität von Bayesschen Methoden in der Vergangenheit stark eingedämmt hat. Mit steigender Rechenkapazität werden aber derzeit selbst hochkomplexe Probleme mittels numerischer oder stochastischer Approximationsverfahren lösbar. Numerische Quadraturmethoden, wie z. B. die Trapezregel oder die Gaußsche Quadratur [106], sind besonders gut dafür geeignet, um Integrale von eindimensionalen Funktionen zu lösen. Anstatt das Integral der Zielfunktion $f(\theta)$ direkt zu lösen, wird durch Auswertung des Integranden an wenigen Stützstellen eine approximierende Funktion $\hat{f}(\theta)$ erzeugt, auf der eine analytische Lösung möglich ist. Die Genauigkeit der Lösung hängt im Wesentlichen von der Anzahl der Stützstellen N und der Erfüllung einiger Glattheitsannahmen ab. Die Lösung von hochdimensionalen Integralen erfordert jedoch die Auswertung der Funktion an N^D Stützstellen [106]. Stochastische Integrationsverfahren bieten hier deutlich bessere Konvergenzeigenschaften, typischerweise in der Größenordnung $1/\sqrt{N}$ unabhängig von der Dimensionalität des Problems [106, 119]. Bei den sogenannten Monte-Carlo Verfahren wird das kontinuierliche Integral durch Auswertung von gleichverteilten zufälligen Punkten θ_i durch die Summe

$$\hat{f}(\theta) \approx \frac{V}{N} \sum_{i=1}^{N} f(\theta_i) \tag{2.45}$$

$$V = \int 1 d\theta \tag{2.46}$$

approximiert, wobei V dem Volumen des Trägers der Zielfunktion entspricht. Die Konvergenz kann häufig beschleunigt werden, indem die Stützpunkte statt aus einer Gleichverteilung aus einer problemangepassten Vorschlagsverteilung $h(\theta)$ gezogen werden. Das Ziehen einer Stichprobe aus einer Verteilung wird in diesem Kontext auch als Simulation bezeichnet. Durch Umgewichtung der Summanden ergibt sich die Lösung:

$$\hat{f}(\theta, h) \approx \frac{V}{N} \sum_{i=1}^{N} \frac{f(\theta_i)}{h(\theta_i)} \quad , \theta_i \sim h(\theta)\,. \tag{2.47}$$

Für eine korrekte Lösung muss die Vorschlagsverteilung den gleichen oder einen größeren Träger wie die Zielfunktion aufweisen. Für die Berechnung von Erwartungswerten bietet es sich weiterhin an, eine Vorschlagsverteilung zu wählen, welche möglichst viel Information über das Problem enthält und eine einfache Simulation ermöglicht. Kandidaten für solche Vorschlagsverteilungen wären die a-Priori Verteilung oder die Likelihood.

Mithilfe des Gesetzes der großen Zahlen kann gezeigt werden, dass die Summen in (2.45) und (2.47) für $N \to \infty$ gegen den gesuchten Wert konvergieren. Weiterhin besagt der zentrale Grenzwertsatz, dass der Schätzfehler normalverteilt ist [106].

Die Monte-Carlo Simulationsmethode erfordert statistisch unabhängige Stichproben aus der Vorschlagsverteilung $h(\theta)$. In komplexen Modellen kann diese Anforderung nur unter sehr großem Aufwand für sinnvolle Vorschlagsverteilungen sichergestellt werden. Eine einfachere Möglichkeit, diese Bedingung aufzulockern und somit die Verwendung von informativeren Vorschlagsverteilungen zu ermöglichen, stellen die Markov-Chain-Monte-Carlo (MCMC) Verfahren dar, welche im nächsten Abschnitt vorgestellt werden.

2.4.1 Markov-Chain-Monte-Carlo Verfahren

Die Anwendung von Bayesschen Methoden erfordert die Simulation der a-Posteriori Verteilung. Eine direkte Simulation von ihr ist in praktischen Problemstellungen allerdings oft nicht erreichbar. In MCMC Verfahren wird diese direkte Simulation daher durch eine Markov-Kette ersetzt, welche nur im Grenzfall $N \to \infty$ zur gewünschten Verteilung konvergiert [106]. Der Startwert θ_0 der Kette kann im Prinzip beliebig gewählt werden. In jedem Simulationsschritt i wird aus einer Übergangsverteilung $p(\theta_{i+1}|\theta_i)$ ein neuer Messwert erzeugt. Eine solche Markov-Kette wird in Abbildung 2.9 dargestellt. Im Vergleich zur Vorschlagsverteilung aus Abschnitt 2.4 wird hier eine statistische Abhängigkeit zum aktuellen Messwert erlaubt, was der Kette auch ihre Markov Eigenschaft zuschreibt. Weil die Übergangsverteilungen so adaptiv Informationen aus der a-Posteriori Verteilung berücksichtigen, können sie typischerweise deutlich einfacher sein als vergleichbare Vorschlagsverteilungen. Das resultierende Verhalten der Kette beschreibt einen *Random Walk*, welcher den Träger der Zielverteilung durchwandert und im Mittel länger in den wahrscheinlicheren Gebieten der a-Posteriori Verteilung verweilt. Es sei hier anzumerken, dass die garantierte Konvergenz zur Zielverteilung streng genommen nur im Grenzfall gilt. Die Simulation kann aber vorzeitig abgebrochen werden, sobald der Einfluss des Startwerts auf den aktuellen Zustand nahezu verschwunden ist [15]. In Abschnitt 2.4.2 werden zwei Methoden zur Diagnose der approximativen Konvergenz vorgestellt.

Eine von zwei hinreichenden Bedingungen um die Konvergenz der Markov-Kette zur Zielverteilung zu garantieren stellt die Eigenschaft des detaillierten Gleichgewichts dar [106]. Sie besagt, dass die Markov-Kette keine zeitliche Vorzugsrichtung haben darf. Die Folge der simulierten Messwerte muss also sowohl vorwärts als auch rückwärts

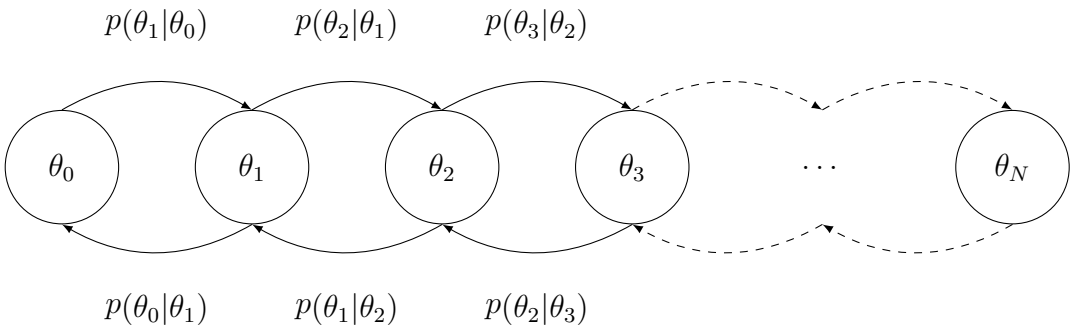


Abbildung 2.9: Eine eindimensionale Markov Kette mit den Zuständen θ_i und den Übergangswahrscheinlichkeiten $p(\theta_i|\theta_{i-1})$ und $p(\theta_i|\theta_{i+1})$.

angereiht zu denselben Zustandswahrscheinlichkeiten führen. Diese Überlegung führt zu der Bedingung:

$$p(\theta_i, \theta_{i+1}) = p(\theta_{i+1}, \theta_i) \Leftrightarrow p(\theta_{i+1}|\theta_i)\, p(\theta_i) = p(\theta_i|\theta_{i+1})\, p(\theta_{i+1}) \ . \tag{2.48}$$

Weiterhin muss für den durch die Markov-Kette beschriebenen Prozess Ergodizität gelten. Für die Kette folgt daraus, dass sie irreduzibel, aperiodisch und positiv-rekurrent sein muss.

Aus der Literatur sind viele MCMC Verfahren bekannt, welche diese Eigenschaften erfüllen. Die Unterschiede der Verfahren sind in der Definition der verwendeten Übergangsverteilungen zu suchen. Einige Verfahren, wie das Metropolis-Hastings (MH) Sampling, können als Black-Box Verfahren für beliebige Simulationsprobleme verwendet werden. Wissen über die Parametrisierung des Problems und der Struktur der a-Posteriori Verteilung können jedoch von spezielleren Simulationsverfahren genutzt werden um die Konvergenzrate deutlich zu beschleunigen. Im Folgenden sollen daher einige gängige Techniken vorgestellt werden, welche einzeln oder in Kombination zur Lösung von komplexen Simulationsproblemen genutzt werden können.

Der Metropolis-Hastings Algorithmus

Der Metropolis Algorithmus wurde 1953 zur Lösung von thermodynamischen Systemen veröffentlicht [83]. In den 1970ern wurde dass Verfahren von W.K. Hastings auf allgemeine statistische Fragestellungen erweitert [58]. Der modifizierte Algorithmus wird heutzutage Metropolis-Hastings Algorithmus genannt. Die Grundidee des Verfahrens lässt sich auf Basis des detaillierten Gleichgewichts aus Formel (2.48) leicht herleiten. Wir definieren zunächst eine beliebige Übergangsverteilung $p(\theta_{i+1}|\theta_i)$. Die

Eigenschaft des detaillierten Gleichgewichts gibt uns die Bedingung

$$p(\theta_{i+1}|\theta_i)\,p(\theta_i) = p(\theta_i|\theta_{i+1})\,p(\theta_{i+1}) \Leftrightarrow \frac{p(\theta_{i+1}|\theta_i)\,p(\theta_i)}{p(\theta_i|\theta_{i+1})\,p(\theta_{i+1})} = 1\,. \tag{2.49}$$

Diese Gleichung kann für allgemeine Übergangsverteilungen offensichtlich nicht erfüllt werden. Vielmehr gilt

$$\frac{p(\theta_{i+1}|\theta_i)\,p(\theta_i)}{p(\theta_i|\theta_{i+1})\,p(\theta_{i+1})} = \kappa\,. \tag{2.50}$$

Um dennoch die Eigenschaft des detaillierten Gleichgewichts einzuhalten, beschreibt der MH Algorithmus folgendes Vorgehen:

Algorithmus 1 Metropolis-Hastings

Eingabe: Startkonfiguration θ_0
Ausgabe: Markov Kette $\{\theta_1, \cdots, \theta_N\}$

for $i \in [0,N-1]$ **do**
  $\bar{\theta}_{i+1} \sim p(\theta|\theta_i)$
  $\kappa \leftarrow \left[p\left(\bar{\theta}_{i+1}\middle|\theta_i\right) p(\theta_i)\right] / \left[p\left(\theta_i\middle|\bar{\theta}_{i+1}\right) p\left(\bar{\theta}_{i+1}\right)\right]$
  $\alpha \leftarrow min(1,\kappa)$

  if $u \leq \alpha$ for $u \sim Unif(0,1)$ **then**
    $\theta_{i+1} \leftarrow \bar{\theta}_{i+1}$
  else
    $\theta_{i+1} \leftarrow \theta_i$
  end if
end for

Hastings konnte zeigen, dass die Eigenschaft des detaillierten Gleichgewichts eingehalten werden kann, wenn die Akzeptanzwahrscheinlichkeit auf $\alpha = min(1,\kappa)$ gesetzt wird.

Für die Übergangsverteilungen werden häufig symmetrische Verteilungen bevorzugt, welche ihren Modalwert in θ_i annehmen, wie z. B. eine Normalverteilung. In diesem Fall vereinfacht sich Gleichung (2.50), da sich die Übergangswahrscheinlichkeiten herauskürzen. Die Varianz der Übergangsverteilungen stellt allerdings eine wichtige einzustellende Größe des Verfahrens dar. Kleine Varianzen führen zu einer langsamen Konvergenz, da der resultierende *Random Walk* nur kleine Sprünge enthält. Große Varianzen hingegen führen zu sehr kleinen Akzeptanzwahrscheinlichkeiten. Trotz dieses Problems des Finetunings wird der MH Algorithmus gerne als Referenzverfahren genutzt, da er eine Black-Box Lösung darstellt. Das Verfahren erfordert lediglich die Auswertbarkeit der a-Posteriori Verteilung an einer gegebenen Position θ.

Gibbs Sampling

Mit dem Gibbs Sampling Verfahren [48] wird versucht, das Finetuning Problem des MH Algorithmus durch Ausnutzung von Wissen über die Struktur der Zielverteilung zu umgehen. Hierbei wird angenommen, dass die Zielverteilung $p\left(\theta^1, \cdots, \theta^K\right)$ des multivariaten Modells mit K Parametern zu komplex ist um darauf direkt zu simulieren. Die bedingten Verteilungen der Form $p\left(\theta^j \middle| \theta^1, \cdots, \theta^{j-1}, \theta^{j+1}, \cdots, \theta^K\right)$ hingegen sollen eine relativ einfache Form haben. Eine gültige Markov-Kette, welche zur Zielverteilung konvergiert, lässt sich dann mit dem folgenden Verfahren erzeugen:

Algorithmus 2 Gibbs Sampling

Eingabe: Startkonfiguration $\boldsymbol{\theta}_0$
Ausgabe: Markov Kette $\{\boldsymbol{\theta_1}, \cdots, \boldsymbol{\theta}_N\}$

for $i \in [0, N-1]$ **do**
$\theta^1_{i+1} \leftarrow p\left(\theta^1 \middle| \theta^2_i, \cdots, \theta^K_i\right)$
$\theta^2_{i+1} \leftarrow p\left(\theta^2 \middle| \theta^1_{i+1}, \theta^3_i, \cdots, \theta^K_i\right)$
$\vdots$
$\theta^K_{i+1} \leftarrow p\left(\theta^K \middle| \theta^1_{i+1}, \cdots, \theta^{K-1}_{i+1}\right)$
end for

Das initiale komplexe Problem wurde in K einfachere Simulationsprobleme zerlegt. Eine Folge dieser Zerlegung ist, dass der hierbei entstehende *Random Walk* immer achsenorientiert bezüglich des gewählten Parameterraums ist. Ein beispielhafter Simulationspfad wird in Abbildung 2.10 gezeigt. Falls die Parameter statistisch unabhängig sind, dann erzeugt Gibbs Sampling eine statistisch unabhängige Stichprobe. Ansonsten hängt die Konvergenzrate im Wesentlichen von der Korrelationsstruktur zwischen den Parametern ab. Dennoch wird die Eigenschaft des detaillierten Gleichgewichts, und somit eine Konvergenz zur Zielverteilung, garantiert, solange die Ergodizität für den Markov Prozess gilt. Beim Gibbs Sampling reicht es aus zu zeigen, dass der Träger der Verbundverteilung identisch ist mit dem kartesischen Produkt der Träger der bedingten Verteilungen [106]. Dies bedeutet, dass der Träger entweder den gesamten Parameterraum umfasst oder die Form eines achsenorientierten Hyperwürfels besitzt. Für die Zerlegung der multivariaten Zielverteilung können auch andere Schemata verwendet werden. So ist Gibbs Sampling insbesondere für die Simulation von allgemeinen hierarchischen Modellen, wie z. B. von graphischen Modellen [99], geeignet. Die Zerlegung erfolgt optimalerweise entlang der Korrelationsstruktur des Modells, so dass hochkorrelierte Parameter zusammengefasst und unkorellierte Parameter voneinander getrennt werden. Eine solche Zerlegung wird auch als *Blocking* bezeichnet.

Hit-and-Run Sampling

Bei starken Korrelationen zwischen Parametern ist Gibbs Sampling häufig ineffizient. Der resultierende *Random Walk* enthält nur kurze Sprünge und benötigt somit viele Iterationen um den gesamten Träger zu besuchen. In Abbildung 2.10 wird gezeigt, wie dieses Problem aus den achsenorientierten Übergangsverteilungen von Gibbs Sampling resultiert. *Hit-and-Run* Sampling [24] kann als eine Erweiterung von Gibbs Sampling aufgefasst werden, bei der die Beschränkung der Achsenorientierung aufgehoben wird. Der Algorithmus gestaltet sich wie folgt:

Algorithmus 3 Hit-and-Run Sampling

Eingabe: Startkonfiguration $\boldsymbol{\theta}_0$
Ausgabe: Markov Kette $\{\boldsymbol{\theta_1}, \cdots, \boldsymbol{\theta}_N\}$

for $i \in [0, N-1]$ **do**
  $\boldsymbol{u} \sim \mathcal{N}(0, I)$ ▷ Erzeuge zufälligen Richtungsvektor durch
  $\boldsymbol{d} \leftarrow \boldsymbol{u}/\|\boldsymbol{u}\|$ ▷ Normierung einer normalverteilten Zufallsvariable

  $\mathcal{L} \leftarrow \{\boldsymbol{\theta} \mid \boldsymbol{\theta} = \boldsymbol{\theta}_i + s \cdot \boldsymbol{d}, s \in \mathbb{R}\}$
  $\boldsymbol{\theta}_{i+1} \sim p(\boldsymbol{\theta} | \boldsymbol{\theta} \in \mathcal{L})$
end for

In jedem Simulationsschritt wird eine zufällige Teilmenge des Parameterraums in Betracht gezogen. Die Beschränkung der Verteilung auf die Gerade $\boldsymbol{\theta} + s \cdot \boldsymbol{d}$ entspricht hierbei der Idee, in jeder Iteration nur eine zufällig gewichtete Kombination von Parametern zu simulieren. Sollten die so erzeugten Gewichte in etwa die Korrelationsstruktur der betrachteten Parameter widerspiegeln, so kann die Markov-Kette mit einer hohen Wahrscheinlichkeit große Sprünge zurücklegen und somit schneller konvergieren. Für konvexe Träger und gleichverteilte sowie exponentialverteilte Zielverteilungen konnte sogar bewiesen werden, dass Hit-and-Run Sampling optimale Konvergenzeigenschaften besitzt [78, 79].

2.4.2 Konvergenzkriterien

Die garantierte Konvergenz von MCMC Verfahren gilt nur im Grenzfall $N \to \infty$. Dennoch lässt sich empirisch beobachten, dass gute Approximationen häufig bereits für kleine N erreicht werden können. Die konkrete Wahl dieses Parameters ist jedoch in erster Linie von problembezogenen Größen, wie z. B. der Varianz der a-Posteriori Verteilung, der Korrelationsstruktur der Modellparameter und dem gewählten Startpunkt der Markov-Kette abhängig. In der Literatur wurden daher diverse Konvergenzkriterien vorgeschlagen, aus denen direkt Abbruchbedingungen

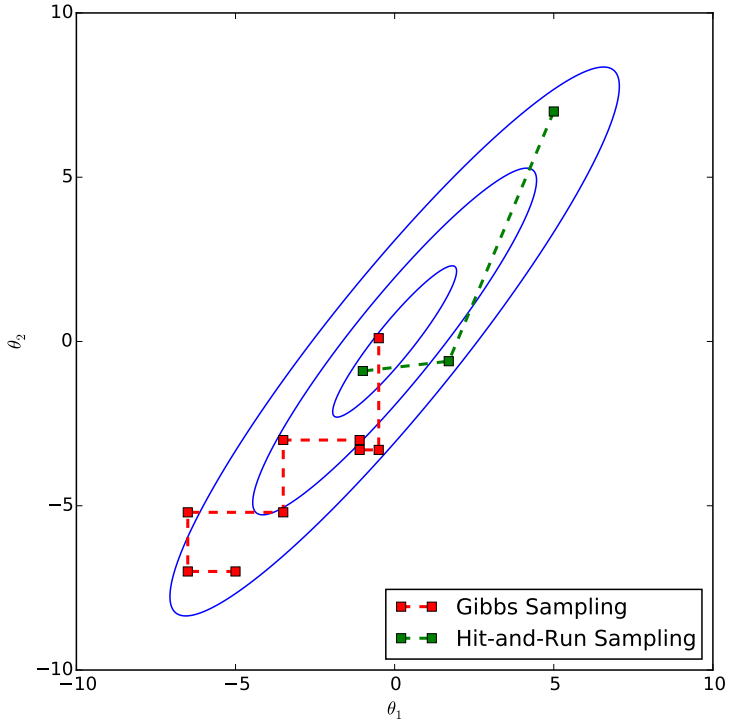


Abbildung 2.10: Beispielhafter Verlauf von simulierten Markov-Ketten auf einer zweidimensionalen Normalverteilung mit den korrelierten Parametern θ_1 und θ_2. Gibbs Sampling benötigt aufgrund seines achsenorientierten *Random Walk* Verhaltens viele Iterationen um Regionen hoher Wahrscheinlichkeit zu erreichen. Hit-and-Run Sampling benötigt hierfür bei günstig gewählten Suchrichtungen nur wenige Iterationen.

für die Simulationen abgeleitet werden können. Im Folgenden sollen zwei gängige Kriterien vorgestellt werden, welche unterschiedliche Aspekte der Konvergenz beleuchten.

Effektive Stichprobengröße

Aus der Normalapproximation lässt sich die Varianz des Schätzfehlers von Monte-Carlo Methoden in Gleichung (2.45) in direkter Abhängigkeit zur Stichprobengröße angeben. Die Normalapproximation gilt allerdings nur für identisch verteilte und statistisch unabhängige Stichproben. Durch Verletzung dieser Annahmen wird in MCMC Simulationen der tatsächliche Schätzfehler eine deutlich höhere Varianz aufweisen. Beide Annahmen sollten unter Berücksichtigung der Eigenschaften des zu untersuchenden Markov Prozesses separat untersucht werden. Die erste Bedingung wird für MCMC Verfahren für $N \to \infty$ garantiert erfüllt, da die Stichprobe in diesem Fall entsprechend der a-Posteriori Verteilung verteilt ist. Aus der Eigenschaft des

detaillierten Gleichgewichts kann aber geschlussfolgert werden, dass die Bedingung auch für endliche N beliebig genau erfüllt wird, sobald die Markov-Kette ihren Startpunkt vergessen hat [15]. Dieser Zustand kann z. B. mit dem Verfahren von Gelman & Rubin aus dem nachfolgenden Abschnitt detektiert werden. Die zweite Bedingung fordert die statistische Unabhängigkeit der Stichprobe. Diese Bedingung wird von MCMC Verfahren aufgrund ihres *Random Walk* Verhaltens typischerweise nicht erfüllt. In erster Näherung können die Abhängigkeiten des Markov Prozesses durch dessen normierte Autokorrelationsfunktion

$$R(\tau) = \mathrm{E}\left[\frac{(\theta_t - \mu)(\theta_{t-\tau} - \mu)}{\sigma^2}\right] \tag{2.51}$$

beschrieben werden. Hieraus wird nun die effektive Stichprobengröße $\tilde{N}$ abgeleitet [35]. Der Wert $\tilde{N}$ beschreibt die Stichprobengröße eines virtuellen Prozesses mit unkorrelierten Messwerten, welcher dieselbe Schätzfehlervarianz produziert wie der zugrundeliegende Originalprozess. Für die Berechnung von $\tilde{N}$ gilt:

$$\tilde{N} = N \cdot \left(\sum_{\tau=-\infty}^{\infty} R(\tau)\right)^{-1} \tag{2.52}$$

Die Berechnung dieser Kenngröße kann insbesondere bei Existenz von negativen Keulen in der Autokorrelationsfunktion sehr instabil werden. Eine Heuristik versucht dieses Problem in den Griff zu bekommen, indem die Summierung der Autokorrelationsfunktion beim Erreichen des ersten negativen Anteils abgebrochen wird. Dieser Ansatz erzeugt jedoch nicht immer verlässliche Schätzungen der effektiven Stichprobengröße. Deshalb wird in dieser Arbeit die Konvergenzdiagnose nach Gelman & Rubin bevorzugt verwendet.

Konvergenzdiagnose nach Gelman & Rubin

Ein wichtiges Konvergenzkriterium besagt, dass der Markov-Prozess Stationarität erreicht haben muss. Daraus folgt weiterhin, dass der Einfluss des Startpunkts auf den Prozess verschwunden ist [15]. Bei der Betrachtung von M unabhängigen Realisierungen des Prozesses müssen daher dessen statistische Eigenschaften nach der Konvergenz ununterscheidbar sein. Gelman & Rubin haben für die Analyse dieses Konvergenzkriteriums den potentiellen Skalenreduktionsfaktor $\hat{R}$ vorgeschlagen. Zur Vereinfachung nehmen wir an, dass sich die Zielverteilung des stationären Markov Prozesses durch eine Normalverteilung approximieren lässt. Falls also alle Realisierungen des Prozesses gegen dieselbe Verteilung konvergiert sind, so sollten deren Mittelwerte und Varianzen identisch sein. Zur Überprüfung dieser Behauptung werden zwei Schätzer für die Prozessvarianz definiert. Der erste Schätzer W arbeitet zu optimistisch und berechnet die Prozessvarianz durch Mittelung der Varianzen aller Realisierungen. Vor der Konvergenz hatten die Prozessrealisierungen nicht genug Zeit den gesamten

Träger zu besuchen, wodurch die Varianz immer zu niedrig geschätzt wird. Der zweite Schätzer $\hat{\sigma}^2$ arbeitet pessimistisch und berechnet die Varianz der Konkatenation aller Realisierungen. Vor der Konvergenz sollten die Mittelwerte der Realisierungen aufgrund der unterschiedlichen Startpunkte weit auseinanderliegen, wodurch die Varianz zu groß geschätzt wird. Im Falle der Konvergenz werden beide Schätzer gegen dieselbe unverzerrte Prozessvarianz konvergieren. Die Berechnungsvorschriften der Schätzer und des potentiellen Skalenreduktionsfaktors lauten [15]:

$$W = \frac{1}{M} \sum_{i=1}^{M} \frac{1}{N-1} \sum_{j=1}^{N} \left(\theta_{ij} - \bar{\theta}_i\right)^2 \tag{2.53}$$

$$B = \frac{N}{M-1} \sum_{i=1}^{M} \left(\bar{\theta}_i - \frac{1}{M} \sum_{j=1}^{M} \bar{\theta}_j\right)^2 \tag{2.54}$$

$$\hat{\sigma}^2 = \frac{N-1}{N} W + \frac{1}{N} B \tag{2.55}$$

$$\hat{R} = \frac{\hat{\sigma}^2}{W} \,. \tag{2.56}$$

Die Parameter θ_{ij} beschreiben den j-ten Messwert der i-ten Realisierung. Durch die Parameter $\bar{\theta}_i$ sind die Prozessmittelwerte der i-ten Realisierung gegeben. Schlussendlich beschreibt der Faktor $\hat{R}$ das Verhältnis der beiden Schätzer. Er nimmt bei vollständiger Konvergenz den Wert 1.0 an. Aus praktischen Gesichtspunkten kann die Simulation allerdings bereits abgebrochen werden, sobald $\hat{R}$ einen Schwellwert etwas größer als 1.0 unterschreitet. In der Literatur wurden unterschiedliche Schwellwerte zwischen 1.05 und 1.2 vorgeschlagen. Der Wert kann so interpretiert werden, dass die Varianz der simulierten Zielverteilung durch eine unendliche Fortführung der Markov-Kette höchstens um den Faktor $\hat{R}$ reduziert werden kann.

Kapitel 3

Statistisches Quellenselektionsmodell

In diesem und dem folgenden Kapitel wird eine Methode zur Domänenadaption für den Transfer von Instanzen auf Grundlage eines Bayesschen Lernmodells entwickelt. Das Verfahren soll in zwei Stufen vorgestellt werden. In diesem Kapitel wird zunächst ein theoretisches Modell der Domänenadaptation hergeleitet. Das Anlernen eines Transferlernverfahrens geschieht zunächst durch eine stochastische Approximation auf Basis einer MCMC Simulation. Im nächsten Kapitel wird dieses Modell dann für den Einsatz auf sehr großen Datensätzen optimiert und erweitert. Dieses Kapitel startet zunächst in Abschnitt 3.1 mit der Formulierung der Domänenadaption als Problem der Quellenselektion. Hierfür wird in Abschnitt 3.1 ein statistisches Distanzmaß zwischen Domänen entwickelt. Diese Distanz bildet die Grundlage für ein Bayessches Transferlernverfahren, dessen Parameter durch ein in Abschnitt 3.2 entwickeltes MCMC Verfahren zunächst für die Einzelquellenselektion geschätzt werden. Die Evaluation der beschriebenen Verfahren erfolgt in Abschnitt 6.2.

3.1 Quellenselektion

Für konkrete Anwendungen der Fernerkundung, in denen gelernte Entscheidungsmodelle zum Einsatz kommen, existieren häufig bereits eine Vielzahl an Daten, zu denen die korrekten Klassenlabels bereits bekannt sind. Durch Aufteilung dieser Daten, z. B. anhand ihrer Sensoreigenschaften, Geokoordinaten und dem Zeitpunkt der Aufnahme, entsteht eine Menge von verfügbaren Quellen $(S_i)_{i=1}^{Q} = S \subset \mathcal{S}$. Die Bilddaten, bzw. daraus abgeleitete Merkmalsvektoren, und die dazugehörigen Klassenlabels können als Stichproben aus den Verbundverteilungen $p_{S_i}(\boldsymbol{x}, y)$ verstanden werden. Doch nicht alle potentiellen Verbundverteilungen repräsentieren eine plausible Quelle. Zum Beispiel dürfte zu einer konstruierten Quelle, in der Bäume eine geringere Höhe als Fahrzeuge besitzen, kein entsprechender realer Datensatz zu finden sein. Jede Quelle besitzt somit eine Auftretenswahrscheinlichkeit $p(s)$, wobei zufällig oder künstlich erzeugte Quellen in der Regel nicht plausibel sind für solche $p(s) = 0$ gilt. Die Existenz einer solchen a-Priori Wahrscheinlichkeit legt weiterhin die Vorstellung nahe, dass jede Quelle s, sowie auch das Ziel T, in einem abstrakten Raum $\mathcal{S}$ aller modellierbaren Quellen liegt. Dieser Raum ist im Allgemeinen unendlichdimensional und nichteuklidisch. Mit steigendem Q sollten jedoch die verfügbaren Quellenkandidaten ein zunehmend genaueres Bild über den Raum $\mathcal{S}$ sowie $p(s)$ ermöglichen. Im Folgenden wird die Menge der Quellenkandidaten S als statistisch unabhängige Stichprobe aus $p(s)$ interpretiert. Zu jeder Quelle S_i ist weiterhin eine Entscheidungsfunktion $h_{S_i}(\boldsymbol{x})$ vorhanden, welche durch ein überwachtes Lernverfahren aus Tr_{S_i} gelernt wurde. Transferlernen soll in dieser Arbeit daher als statistisches Regressionsproblem interpretiert werden, bei dem aus den Quellenkandidaten eine Abbildung $f : \mathcal{S} \times \mathcal{S}^Q \to \mathcal{P}$ gelernt wird. Es gilt:

$$h_T(\boldsymbol{x}) = f(T, S_1, \ldots, S_Q)\,. \tag{3.1}$$

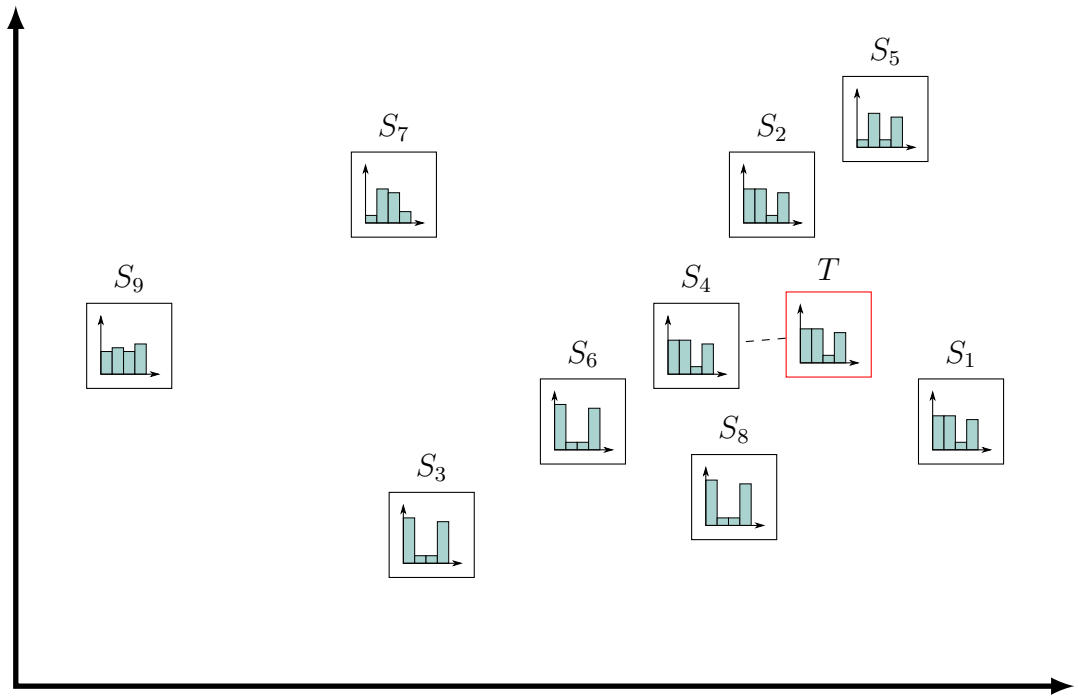


Abbildung 3.1: Nächster Nachbar Regression im Raum aller Quellen für ein Quellenselektionsproblem mit den Kandidaten $S_1, \ldots, S_9$. Jeder Kasten repräsentiert eine Verbundverteilung in dem unendlichdimensionalen Raum. Anhand eines Distanzmaßes kann jedoch mit S_4 die der Zielverteilung T ähnlichste Quelle bestimmt werden.

In diesem Kapitel wird dieses Regressionsmodell zunächst durch eine einfache Nächster-Nachbar (NN) Regression implementiert. Dies bedeutet, zu einem gegebenem Ziel T suchen wir bezüglich einer geschickt gewählten Distanzfunktion $d(.,.)$ die nächste Quelle S_{NN}:

$$S_{NN} = \arg\min_{s \in S} d(p_T(\boldsymbol{x}, y), p_s(\boldsymbol{x}, y)) \,. \tag{3.2}$$

In Abbildung 3.1 wird dieses Prinzip dargestellt.

Im weiteren Verlauf muss die für unser Regressionsmodell benötigte Distanzfunktion hergeleitet werden. Eine natürliche Distanzfunktion ergibt sich aus der Überlegung, dass eine Entscheidungsfunktion einer Quelle auf ähnlichen Klassifikationsproblemen möglichst hohe Klassifikationsgenauigkeiten erzielen muss. Für den Fall des induktiven Transfers ergibt sich so eine Definition der Quellenselektion:

Definition 10 (Induktive Quellenselektion)**.** *Gegeben seien das Ziel* $T = (\mathcal{D}_T, \mathcal{T}_T, \mathrm{T}_T)$ *sowie die Quellen* $S = \{(\mathcal{D}_{S_i}, \mathcal{T}_{S_i}, \mathrm{T}_{S_i})\}_{i=1}^{Q}$*. Als Quellenselektion wird eine Methode*

bezeichnet, welche das Optimierungsproblem

$$\hat{S} = \arg\min_{s \in S} \epsilon(h_s, \mathrm{T}_T) \tag{3.3}$$

löst. Es wird somit eine Quelle $\hat{S}$ ermittelt, deren Entscheidungsfunktion $h_{\hat{S}}(\boldsymbol{x})$ auf dem Ziel den geringsten Generalisierungsfehler aufweist. Die Funktion $\epsilon(.,.)$ nimmt hier somit die Rolle einer Distanzfunktion an.

Die Berechnung einer induktiven Quellenselektion erfordert Kenntnis über die korrekten Klassenlabels der Zielstichprobe, welche jedoch für das Lernszenario, mit dem sich diese Arbeit beschäftigt, nicht vorhanden sind. Für den Fall der Domänenadaption wird daher eine Approximation der Formel (3.3) benötigt, welche im Rahmen einer Domänenadaption aus der verfügbaren Informationen berechnet werden kann. In dieser Arbeit wird der Ansatz verfolgt, den Generalisierungsfehler $\epsilon(h_s, \mathrm{T}_T)$ durch eine obere Abschätzung zu ersetzen. Somit wird eine Quelle gesucht, deren *maximal möglicher Generalisierungsfehler* auf dem Ziel minimiert wird. Ben-David et al. konnten in [8] eine solche Abschätzung herleiten:

$$\epsilon(h(\boldsymbol{x}), \mathrm{T}_T) \leq \epsilon(h(\boldsymbol{x}), \mathrm{T}_S) + 2d_{\mathrm{TV}}(p_T(\boldsymbol{x}), p_S(\boldsymbol{x})) + \gamma \tag{3.4}$$

Für γ muss gelten $\min_{h^*}[\epsilon(h^*(\boldsymbol{x}), \mathrm{T}_T) + \epsilon(h^*(\boldsymbol{x}), \mathrm{T}_S)] \leq \gamma$, d. h. es muss eine Entscheidungsfunktion $h^*(.)$ existieren, welche sowohl auf dem Ziel als auch auf der Quelle einen niedrigen Generalisierungsfehler erzeugt. Die Variable γ liefert somit ein Maß dafür, ob ein Transfer aus der Quelle, bezogen auf eine gewählte Familie von Klassifikationsmodellen, überhaupt sinnvoll möglich ist. Allerdings ist dieser Wert nur unter Kenntnis der Klassenlabels der Zielstichprobe berechenbar. Im Folgenden muss daher angenommen werden, dass γ für alle behandelten Probleme hinreichend kleine Werte annimmt. Bei der Funktion $d_{\mathrm{TV}}(p_T(\boldsymbol{x}), p_S(\boldsymbol{x}))$ handelt es sich um ein statistisches Maß der Distanz zwischen den Wahrscheinlichkeitsverteilungen $p_T(\boldsymbol{x})$ und $p_S(\boldsymbol{x})$. Diese sogenannte Total-Variation (TV) Distanz ist durch folgende Formel definiert:

$$d_{\mathrm{TV}}(p_T(\boldsymbol{x}), p_S(\boldsymbol{x})) = \frac{1}{2} \int_{\boldsymbol{x} \in \mathcal{F}} |p_T(\boldsymbol{x}) - p_S(\boldsymbol{x})| d\boldsymbol{x} \,. \tag{3.5}$$

Diese Formel gilt allerdings nur bei bekannten Wahrscheinlichkeitsdichten. Diese sind in der Praxis üblicherweise unbekannt, und es stehen jeweils nur kleine Stichproben aus den Verteilungen zur Verfügung. In [123] wurde gezeigt, dass ein einfacher empirischer Schätzer der TV Distanz immer zum trivialen Resultat $d_{\mathrm{TV}}(\mathrm{T}_T, \mathrm{T}_S) = 1$ führt. Eine sinnvolle Approximation der empirischen TV Distanz ergibt sich allerdings aus einer anderen Interpretation der Formel (3.5). Abbildung 3.2 zeigt ein Beispiel der Berechnung der TV Distanz zwischen zwei Normalverteilungen. Die Distanz entspricht hier der Hälfte der grauen Fläche zwischen den Kurven. Hier ist einfach zu erkennen, dass die TV Distanz äquivalent aus der roten Fläche berechnet werden

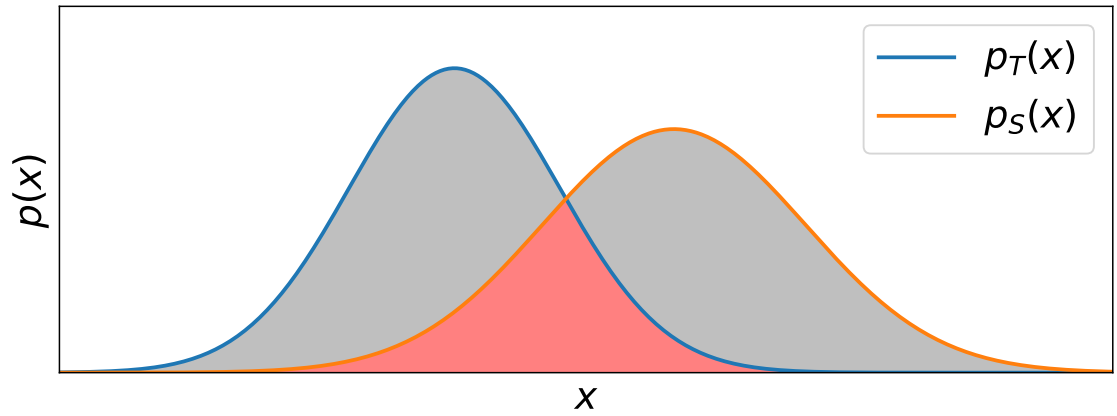


Abbildung 3.2: Beziehung zwischen der Total-Variation Distanz und dem Bayes-Klassifikator. Die rote Fläche entspricht dem erwarteten Klassifikationsfehler eines Bayes-Klassifikators. Die graue Fläche entspricht der Total-Variation Distanz.

kann. Diese Fläche entspricht dem Generalisierungsfehler eines optimalen Bayes-Klassifikators unter 0-1 Kosten, welcher das Entscheidungsproblem $T \perp S$ löst. Die Notation $T \perp S$ beschreibt hier ein Klassifikationsproblem, bei dem ein Klassifikator Merkmalsvektoren, welche zufällig aus einer der beiden Domänen T oder S erzeugt wurden, der korrekten Ursprungsdomäne zuordnen soll. Hierzu werden keinerlei Klassenlabels benötigt. Bei ununterscheidbaren Domänen ist selbst der optimale Klassifikator nicht in der Lage diese Zuordnung durchzuführen. Deshalb tendiert der Generalisierungsfehler für ähnliche Domänen zu 0,5. Die TV Distanz ergibt sich deshalb aus der Beziehung

$$d_{\mathrm{TV}}(p_T(\boldsymbol{x}), p_S(\boldsymbol{x})) = 1 - 2 \min_{h \in \mathcal{P}} \epsilon(h(\boldsymbol{x}), \mathrm{T}_{T \perp S}) \,. \tag{3.6}$$

Der Generalisierungsfehler $\epsilon(h(\boldsymbol{x}), \mathrm{T}_{T \perp S})$ muss allerdings nicht notwendigerweise durch einen Bayes-Klassifikator ermittelt werden. Gute Approximationen der TV Distanz ergeben sich durch Ersetzen des Bayes-Klassifikators durch robustere Alternativen. Zwei solcher Alternativen werden im weiteren Verlauf dieses Kapitels sowie im folgenden Kapitel hergeleitet.

Aus den vorangegangenen Überlegungen ergibt sich somit eine Erweiterung der induktiven Quellenselektion auf die Domänenadaption.

Definition 11 (Domänenadaptive Quellenselektion)**.** *Gegeben seien das Ziel* $T = (\mathcal{D}_T, \mathcal{T}, \mathrm{T}_T)$ *sowie die Quellen* $S = \{(\mathcal{D}_{S_i}, \mathcal{T}, \mathrm{T}_{S_i})\}_{i=1}^{Q}$*. Bei* T_T *handelt es sich um eine ungelabelte Stichprobe, während für alle* T_{S_i} *die vollständigen Labelinformationen vorhanden sind. Weiterhin gilt für alle* $i \in [1..Q]$ $\mathcal{F}_T = \mathcal{F}_{S_i}$ *und* $\mathcal{K}_T = \mathcal{K}_{S_i}$*. Als domänenadaptive Quellenselektion wird eine Methode bezeichnet, welche das*

Optimierungsproblem

$$\hat{S} = \arg\min_{s \in S} d(\mathrm{T}_T, \mathrm{T}_s)\,. \tag{3.7}$$

mithilfe der sogenannten Domänendistanz

$$d(\mathrm{T}_T, \mathrm{T}_s) = \epsilon(h(\boldsymbol{x}), \mathrm{T}_s) + 2d_{TV}(\mathrm{T}_T, \mathrm{T}_s) \tag{3.8}$$

löst.

Zur Berechnung der Domänendistanz (3.8) wird im nächsten Abschnitt ein hierarchisches Bayessches Modell zur robusten Schätzung von Generalisierungsfehlern entwickelt. Die hierarchische Modellierung erlaubt eine automatische Anpassung der Modellkomplexität an die Daten und vermeidet somit gleichzeitig Probleme der Unter- sowie Überanpassung.

3.2 Bayessche Domänendistanz

In diesem Abschnitt wird eine Bayessche Formulierung des Problems der Quellenselektion

$$\hat{S} = \arg\min_{s \in S} d(\mathrm{T}_T, \mathrm{T}_s) \qquad \text{(siehe 3.7)}$$

mit der Domänendistanz

$$d(\mathrm{T}_T, \mathrm{T}_S) = \epsilon(h(\boldsymbol{x}), \mathrm{T}_S) + 2d_{\mathrm{TV}}(\mathrm{T}_T, \mathrm{T}_S) \qquad \text{(siehe 3.8)}$$

vorgestellt. Die Gleichung (3.8) ist dabei direkt aus der theoretischen Schranke von Ben-David et al. [8] abgeleitet worden. Bei Verwendung einer quadratischen Kostenfunktion ergibt sich ein Schätzer der Domänendistanz durch einen Bayesschen Erwartungswertschätzer:

$$\begin{aligned}
\hat{S} &= \arg\min_{s \in S} \mathrm{E}[\epsilon(h_s(\boldsymbol{x}), \mathrm{T}_s) + 2d_{\mathrm{TV}}(\mathrm{T}_T, \mathrm{T}_s)|\mathrm{T}_T, \mathrm{T}_s] \\
&= \arg\min_{s \in S} \mathrm{E}[\epsilon(h_s(\boldsymbol{x}), \mathrm{T}_s) + 2(1 - 2\epsilon(h_{T\perp s}(\boldsymbol{x}), \mathrm{T}_{T\perp s}))|\mathrm{T}_T, \mathrm{T}s] \\
\Rightarrow \hat{S} &= \arg\min_{s \in S} \mathrm{E}[\epsilon(h_s(\boldsymbol{x}), \mathrm{T}_s)|\mathrm{T}_s] - 4\,\mathrm{E}[\epsilon(h_{T\perp s}(\boldsymbol{x}), \mathrm{T}_{T\perp s})|\mathrm{T}_T, \mathrm{T}_s]\,.
\end{aligned} \tag{3.9}$$

Zur Minimierung dieser Zielfunktion reicht es aus, für ein gegebenes s die beiden Erwartungswerte separat zu optimieren. Zur Berechnung der Erwartungswerte wird in den folgenden Abschnitten ein MCMC Verfahren hergeleitet. Dieses setzt sich zum Einen aus einer a-Posteriori Verteilung $p(\boldsymbol{\varphi}|\mathrm{Tr})$ über Klassifikationsmodelle $\boldsymbol{\varphi}$ konditioniert auf den gegeben Lernstichproben Tr sowie einer Methode zur Simulation aus dieser Verteilung zusammen. Anhand von Teststichproben Te können somit aus den erzeugten Markov Ketten die mittleren Generalisierungsfehler berechnet werden.

3.2.1 Likelihood

Laut dem Satz von Bayes kann die benötigte a-Posteriori Verteilung folgendermaßen formuliert werden:

$$p(\boldsymbol{\varphi}|\mathrm{Tr}) \propto p(\mathrm{Tr}|\boldsymbol{\varphi})\, p(\boldsymbol{\varphi}) \ . \tag{3.10}$$

Die Variable $\boldsymbol{\varphi} \in \mathcal{P}$ beschreibt hier ein parametrisiertes Klassifikationsmodell, welches aus der Lernstichprobe Tr angelernt wird. Nach den Erkenntnissen aus Abschnitt 2.1.1 wird die Form der a-Posteriori Verteilung für große Lernstichproben durch die Likelihood $p(\mathrm{Tr}|\boldsymbol{\varphi})$ dominiert. Daher soll deren Herleitung und Definition im Fokus dieses Abschnitts stehen. Allgemein wird die Likelihood der Lernstichprobe für Klassifikationsprobleme basierend auf der Kreuzentropie zu

$$p(\mathrm{Tr}|\boldsymbol{\varphi}) = e^{-\sum_{i=1}^{N} L(\boldsymbol{x}_i, y_i, \boldsymbol{\varphi})} = \prod_{i=1}^{N} p(y_i|\boldsymbol{x}_i, \boldsymbol{\varphi}) \tag{3.11}$$

formuliert. Die Klassenwahrscheinlichkeiten $p(y_i|\boldsymbol{x}_i, \boldsymbol{\varphi})$ werden in diesem Modell von der Wahl der Kostenfunktion $L(.)$ bestimmt. Für den Fall der logistischen Kosten ergibt sich z.B. die logistische Regression [12]. Aufgrund der engen Beziehung der TV Distanz zu einem optimalen Bayesschen Klassifikator bietet sich allerdings die Verwendung der 0-1 Kosten an. Dann gilt:

$$p(y_i|\boldsymbol{x}_i, \boldsymbol{\varphi}) = \varpi_{y_i, h(\boldsymbol{x}_i)} \ , \tag{3.12}$$

wobei $\varpi_{.,.}$ konstante Wahrscheinlichkeiten sind, welche frei einstellbar sind und daher optimiert werden müssen. Ein möglicher Ansatz wird von Bayes-Point-Machines (BPMs) verfolgt [61]. Hier gilt:

$$\varpi_{c,p} = \begin{cases} 1 & \text{, falls } c = p \\ 0 & \text{, sonst} \, . \end{cases} \tag{3.13}$$

Als unerwünschter Nebeneffekt setzt diese Formulierung die Likelihood auf 0, sobald auch nur ein Lernbeispiel der Lernstichprobe falsch klassifiziert wird. Somit können sich überlagernde Klassenverteilungen, welche einen positiven minimalen Trainingsfehler besitzen, von einer BPM grundsätzlich nicht modelliert werden. Im Folgenden wird daher ein neuer alternativer Ansatz vorgestellt um diese Parameter automatisch aus der Lernstichprobe zu schätzen.

Nehmen wir zunächst an, die Parameter $\boldsymbol{\varpi}$ seien konstant, jedoch mit unbekannten Werten. In der Bayesschen Methodik müssen für unbekannte Variablen immer alle potentiell annehmbaren Werte betrachtet werden. Durch Integration über diesen Wertebereich werden die Unbekannten dann *herausmarginalisiert*. Dies bedeutet, dass, obwohl der Effekt dieser Variablen berücksichtigt wird, sie im Resultat nicht

mehr direkt vorkommen. Da nicht immer alle potentiellen Werte gleich plausibel sind, sollte zusätzlich eine a-Priori Verteilung $p(\boldsymbol{\varpi}|\boldsymbol{\alpha})$ über $\boldsymbol{\varpi}$ definiert werden. Dann gilt:

$$p(\mathrm{Tr}|\boldsymbol{\varphi},\boldsymbol{\alpha}) = \int p(\mathrm{Tr}|\boldsymbol{\varphi},\boldsymbol{\varpi})\, p(\boldsymbol{\varpi}|\boldsymbol{\alpha})\, d\boldsymbol{\varpi}\,. \tag{3.14}$$

Die einzigen Unbekannten in diesem hierarchischen Modell sind nun die funktionelle Form der a-Priori Verteilung sowie die Werte der Hyperparameter $\boldsymbol{\alpha}$. Zum ersten Punkt existiert kein objektives Argument, welches herangezogen werden kann, um Verteilungen der einen oder der anderen Art zu bevorzugen. Die Wahl einer Dirichlet Verteilung [106] hat jedoch den Vorteil zu einer besonders einfachen Likelihood zu führen:

$$p(\mathrm{Tr}|\boldsymbol{\varphi},\boldsymbol{\alpha}) \propto \prod_{c=1}^{|\mathcal{K}|} \frac{\prod_{p=1}^{|\mathcal{K}|} \Gamma(C_{c,p}(\mathrm{Tr},\boldsymbol{\varphi}) + \alpha_{c,p})}{\Gamma(\sum_{p=1}^{|\mathcal{K}|} C_{c,p}(\mathrm{Tr},\boldsymbol{\varphi}) + \alpha_{c,p})}\,. \tag{3.15}$$

Eine solche Verteilung bildet eine Sonderform einer *Pólya Verteilung*, bzw. einer *multinomiellen Dirichlet Verteilung* [86]. Die Funktion $\Gamma(.)$ bezeichnet hierbei die Eulersche Gammafunktion. Die Funktion $C_{c,p}(.,.)$ zählt, wie viele Lernbeispiele mit dem Klassenlabel c vom Klassifikationsmodell der Klasse p zugeordnet wurden, und erzeugt somit die Konfusionsmatrix des Klassifikationsproblems. Somit können die Hyperparameter $\boldsymbol{\alpha}$ in einer *Pólya Verteilung* einfach interpretiert werden. Ihr Effekt ist äquivalent mit dem Hinzufügen von synthetischen Lernbeispielen, deren tatsächliche Klassenlabel und Partitionszuordnungen durch den Klassifikator bereits festgelegt sind. Im Vergleich zu den Parametern $\boldsymbol{\varpi}$ können die Hyperparameter nun deutlich einfacher eingestellt werden. Hierzu treffen wir zwei wesentliche Annahmen. Erstens seien alle Klassen $c \in \mathcal{K}$ gleich wichtig und durch gleich viele Lernbeispiele repräsentiert[1]. Somit entsteht eine starke Symmetrie bei den Hyperparametern, denn dann gilt $\alpha_{c,c} = \alpha_1$ und $\alpha_{c,p} = \alpha_2$ für $c \neq p$. Zweitens soll die a-Priori Verteilung der Hyperparameter möglichst uninformativ für eine gegebene Lernstichprobe sein. Die Dirichlet Verteilung ergibt für $\boldsymbol{\alpha} = 1$ eine Gleichverteilung innerhalb des gültigen Definitionsbereichs und wäre somit maximal uninformativ. Allerdings zeigt die Abbildung 3.3, dass die resultierende Likelihood symmetrisch bezüglich des Trainingsfehlers wäre. Somit wären Klassifikationsmodelle invariant bezüglich der Permutation der Klassenlabels. Für eine solche Likelihood gilt z. B. im Zweiklassenfall $p(\epsilon|\boldsymbol{\varphi},\boldsymbol{\alpha}) = p(1-\epsilon|\boldsymbol{\varphi},\boldsymbol{\alpha})$, d. h. Modelle mit hohen Trainingsfehlern werden ebenso hohe Likelihoods zugesprochen wie Modellen mit geringen Trainingsfehlern. Wir stellen die korrekte Korrespondenz zwischen einem Klassenlabel und seiner zugehörigen Partition im Klassifikationsmodell sicher, indem synthetische Lernbeispiele für jede korrekte Zuordnung von Klasse zu seiner Partition hinzugefügt werden. Dies entspricht der Addition von Konstanten $\alpha_{c,c}$ auf der Haupdiagonalen der Konfusionsmatrix $C(\mathrm{Tr},\boldsymbol{\varphi})$. Damit der Dirichlet Prior durch diesen Eingriff nicht

[1] Dies kann durch Stichprobenwiederholung oder Umgewichtung der Klassen sichergestellt werden

zu informativ wird, sollten die $\alpha_{c,c}$ nur gerade hoch genug gesetzt werden, dass eine Reduktion des Trainingsfehlers immer zu einer Steigerung der Likelihood führt. Daher gilt in unserem schwach informativen Modell

$$\alpha_{c,p} = \begin{cases} N_c + 1 & \text{, falls } c = p \\ 1 & \text{, sonst} \,, \end{cases} \tag{3.16}$$

wobei N_c der Anzahl von Beispielen in der Lernstichprobe entspricht, welche der Klasse c angehören.

Beweis. Zunächst zeigt ein genauerer Blick auf die Formel (3.15), dass die Summe im Nenner über alle Partitionen p summiert. Somit ist der Wert des Nenners nicht von $\boldsymbol{\varphi}$ abhängig. Daher soll im Folgenden $G_c = [\Gamma(\sum_{p=1}^{|\mathcal{K}|} C_{c,p}(\mathrm{Tr}, \boldsymbol{\varphi}) + \alpha_{c,p})]^{-1}$ gelten. Um zu vermeiden, dass fehlklassifizierte Lernbeispiele zu einer Verbesserung der Likelihood führen muss für alle $c \in \mathcal{K}$ und $\boldsymbol{\varphi} \in \mathcal{P}$ weiterhin die Bedingung

$$G_c \cdot \Gamma(C_{c,c}(\mathrm{Tr}, \boldsymbol{\varphi}) + \alpha_{c,c})) \geq G_c \cdot \prod_{p \neq c} \Gamma(C_{c,p}(\mathrm{Tr}, \boldsymbol{\varphi}) + \alpha_{c,p}) \tag{3.17}$$

gelten. Da die Eulersche Gammafunktion $\Gamma(x)$ für alle $x \geq 1$ streng monoton wachsend ist und superlinear wächst, tritt der *worst-case* genau dann ein, wenn alle Fehlklassifikationen derselben Klasse zugewiesen werden. Diese Klasse heißt im Folgenden $\neg c$. Somit vereinfacht sich die Rechnung zu

$$G_c \cdot \Gamma(C_{c,c}(\mathrm{Tr}, \boldsymbol{\varphi}) + \alpha_{c,c})) \geq G_c \cdot \Gamma(C_{c,\neg c}(\mathrm{Tr}, \boldsymbol{\varphi}) + \alpha_{c,\neg c}) \,. \tag{3.18}$$

Durch weitere Umformungen und Vereinfachungen ergibt sich

$$\Rightarrow \Gamma(C_{c,c}(\mathrm{Tr}, \boldsymbol{\varphi}) + \alpha_{c,c})) \geq \Gamma(C_{c,\neg c}(\mathrm{Tr}, \boldsymbol{\varphi}) + \alpha_{c,\neg c}) \tag{3.19}$$

$$\Rightarrow C_{c,c}(\mathrm{Tr}, \boldsymbol{\varphi}) + \alpha_{c,c} \geq C_{c,\neg c}(\mathrm{Tr}, \boldsymbol{\varphi}) + \alpha_{c,\neg c} \tag{3.20}$$

$$\Leftrightarrow \alpha_{c,c} \geq C_{c,\neg c}(\mathrm{Tr}, \boldsymbol{\varphi}) - C_{c,c}(\mathrm{Tr}, \boldsymbol{\varphi}) + \alpha_{c,\neg c} \tag{3.21}$$

$$\Rightarrow \alpha_{c,c} \geq N_c + 1 \,. \tag{3.22}$$

Der letzte Schritt ergibt sich daraus, dass es sich sowohl bei $\alpha_{c,c}$ als auch bei $\alpha_{c,\neg c}$ um freie Parameter handelt. Um die geringste Informativität des Dirichlet Priors zu erhalten, wird $\alpha_{c,\neg c}$ auf den kleinsten zulässigen Wert gesetzt. Daraus ergibt sich $\alpha_{c,\neg c} = 1$. □

3.2.2 Parametrisierung und a-Priori Verteilung

Zur vollständigen Beschreibung der a-Posteriori Verteilung müssen neben der Likelihood noch der Parameterraum $\mathcal{P}$ sowie die a-Priori Verteilung $p(\boldsymbol{\varphi})$ definiert

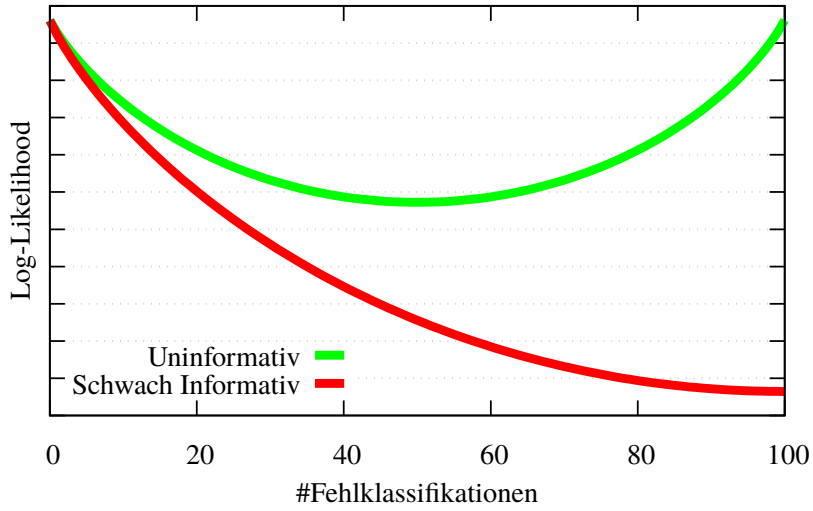


Abbildung 3.3: Darstellung der log-Likelihood konditioniert auf den Trainingsfehler für ein Zweiklassenproblem mit jeweils 50 Lernbeispielen pro Klasse. Für den uninformativen Hyperprior gilt, dass für $\epsilon > 0.5$ eine Erhöhung des Trainingsfehlers zu einer Verbesserung der Likelihood führt. Dieser unerwünschte Effekt wird mit dem schwach informativen Hyperprior vermieden.

werden. Da diese in Bayesschen Modellen in einer sehr engen Beziehung zueinander stehen, erfolgt deren Behandlung in diesem Abschnitt gemeinsam.

Die Parameter $\boldsymbol{\varphi} \in \mathcal{P}$ beschreiben eine Partitionierungsfunktion $h(\boldsymbol{x})$, welche jedem Merkmalsvektor $\boldsymbol{x} \in \mathcal{F}$ eine Klasse $y \in \mathcal{K}$ zuweist. In Anlehnung an Generalisierte Lineare Modelle, wie z. B. der logistischen Regression, erfolgt hier diese Zuordnung äquivalent zu einer *SoftMax* Formulierung durch ein System von linearen Antwortfunktionen. Es gilt:

$$h(\boldsymbol{x}) = \arg\max_{c \in \mathcal{K}} \boldsymbol{\varphi}_c^T \boldsymbol{x} + \varphi_{c,0} \,. \tag{3.23}$$

Dem Vektor wird somit das Klassenlabel zugewiesen, dessen korrespondierende Antwortfunktion maximiert wird. Die Parameter $\boldsymbol{\varphi}_c$ beschreiben den Grad der linearen Abhängigkeit der Merkmale zu den einzelnen Klassen. Die Parameter $\varphi_{c,0}$ werden als Bias bezeichnet. Ein solches Modell besteht aus $|\mathcal{K}| \cdot (D+1)$ Parametern, enthält allerdings noch Redundanz. Die beschriebene Partitionierungsfunktion ändert sich z. B. nicht durch Modifikation des Bias der Form $\varphi'_{c,0} \rightarrow \varphi_{c,0} + \gamma$ für $c \in \mathcal{K}$ und beliebige $\gamma \in \mathbb{R}$. Wird nun die Klasse cr als Referenzklasse gewählt und $\gamma = -(\boldsymbol{\varphi}_{cr}^T \boldsymbol{x} + \varphi_{cr,0})$ eingesetzt, so ergibt die lineare Antwortfunktion für diese

Referenzklasse 0 für alle $\boldsymbol{x}$. Eine modifizierte Parametrisierung mit

$$h(\boldsymbol{x}) = \arg\max_{c\in\mathcal{K}} \begin{cases} 0 & \text{, falls } c = cr \\ \boldsymbol{\varphi'}_c^T \boldsymbol{x} + \varphi'_{c,0} & \text{, sonst} \end{cases} \tag{3.24}$$

ist somit äquivalent und besitzt lediglich $(|\mathcal{K}| - 1) \cdot (D + 1)$ Parameter. Die Parametrisierung in Formel (3.24) enthält exakt einen weiteren Grad der Redundanz, denn es gilt

$$\boldsymbol{\varphi} \equiv s \cdot \boldsymbol{\varphi}, \quad s > 0\,. \tag{3.25}$$

Eine a-Posteriori Verteilung basierend auf der vorgestellten Likelihood und Parametrisierung wäre somit unzulässig, da dessen Integral nicht nach oben begrenzt werden kann. Die a-Priori Verteilung $p(\boldsymbol{\varphi})$ hat daher in erster Linie die Aufgabe, einen Bias innerhalb der Klassen von äquivalenten Parametervektoren zu induzieren. Zumeist geschieht dies, indem Vektoren mit einer geringeren Norm bevorzugt werden. Je nach Wahl der Norm führt dies zu unterschiedlichen a-Priori Verteilungen. Für die L_2 und L_1 Normen gilt jeweils

$$p(\boldsymbol{\varphi}) \propto \exp\!\left(-\frac{1}{2\sigma^2}\|\boldsymbol{\varphi}\|_2^2\right) \qquad \text{(Gauß-Verteilung)}$$

$$p(\boldsymbol{\varphi}) \propto \exp\!\left(-\frac{1}{\sigma}\|\boldsymbol{\varphi}\|_1\right) \qquad \text{(Laplace-Verteilung)}$$

Die Gauß-Verteilung bevorzugt dabei *dichte* Modell, d. h. im Allgemeinen gilt $\varphi_i \neq 0$. Ein solches Entscheidungsmodell verwendet somit alle vorhandenen Merkmale für die Klassifikation, selbst solche, welche nur gering diskriminierend sind. Die Laplace-Verteilung hingegen erzeugt *dünn besetzte* Parametervektoren, für die der Einfluss solcher Merkmale eliminiert wird. Im Allgemeinen erfordern dünn besetzte Modelle deutlich komplexere Lernalgorithmen, führen aber auch zu kompakteren Modellen welche einen geringen Speicherverbrauch und ein besseres Laufzeitverhalten aufweisen. Um die Komplexität der MCMC Methode im nachfolgenden Abschnitt gering zu halten, wird im Folgenden eine a-Priori Verteilung verwendet, welche zu dichten Parametervektoren führt. Dieser Prior schränkt den Raum der modellierbaren Parametervektoren auf solche mit der Norm 1 ein:

$$p(\boldsymbol{\varphi}) = \frac{\Gamma(\frac{n}{2})}{2\pi^{n/2}}\, \mathbb{I}(\|\boldsymbol{\varphi}\|_2 = 1)\,. \tag{3.26}$$

Es werden somit nur Parametervektoren zugelassen die auf der Oberfläche der P-dimensionalen Einheits-Hyperkugel liegen, wobei P der Größe des Parameterraums entspricht. Aufgrund der Beziehung (3.25) und der radialen Symmetrie des Gauß-Priors führen beide a-Priori Verteilungen zu äquivalenten, jedoch nicht zu identischen, Resultaten. Die a-Priori Verteilung aus Formel 3.26 vermeidet Simulationsschritte entlang der redundanten Suchrichtung gemäß Formel 3.25 und soll somit die Konvergenz der Simulation geringfügig beschleunigen.

3.2.3 MCMC Simulationsmethode

Die MCMC Simulation hat zur Aufgabe, eine repräsentative Stichprobe aus einer gegebenen Wahrscheinlichkeitsverteilung zu ziehen. Dies geschieht in der Regel auf Basis der zufälligen Bewegung eines Partikels $\boldsymbol{\varphi}$ durch den Parameterraum $\mathcal{P}$. Die Zufallsbewegung dieses Partikels geschieht dabei nicht willkürlich, sondern wird anhand der Verteilung $p(\boldsymbol{\varphi}|\mathrm{Tr})$ gesteuert. Daher müssen zunächst die wesentlichen Eigenschaften dieser a-Posteriori Verteilung untersucht werden, bevor eine konkrete Simulationsmethode entwickelt werden kann. Zunächst ergibt sich aus der Definition der Likelihood (3.15), dass die Wahrscheinlichkeitsdichte auf die Zählvariablen $C_{c,p}(.,.)$ zurückgeführt werden kann. Die Likelihood wird somit für einen gegebenen Parametervektor $\boldsymbol{\varphi}$ ausgewertet, indem gezählt wird, wie viele Beispiele einer Klasse in eine gegebene Partition fallen. Parametervektoren, welche zu denselben Werten der Zählvariablen führen, sind somit äquivalent. Aufgrund der Stetigkeit der linearen Antwortfunktion in Gleichung (3.24) ergeben weiterhin alle äquivalenten Parametervektoren zusammenhängende Regionen. In Abbildung 3.4 wird ein Beispiel für den vereinfachten Fall eines Zweiklassenproblems mit normalverteilten Klassen in einem 1-dimensionalen Merkmalsraum gezeigt. Für Lernbeispiele in allgemeiner Lage und $D \geq |\mathrm{Tr}|$, was bei bei Anwendung des Kernel Tricks trivialerweise gilt, wird der Parameterraum so in $2^{|\mathrm{Tr}|}$ stückweise konstante Regionen zerlegt. Eine nicht-approximierende Methode zur Ziehung von statistisch unabhängigen Parametervektoren aus dieser Verteilung müsste alle Äquivalenzregionen aufzählen und erlaubt somit keine effiziente Implementierung. Wir können allerdings beobachten, dass in diesem Parameterraum eine beliebige Linie der Form

$$\boldsymbol{\varphi} + t \cdot \boldsymbol{d}, \quad t \in \mathbb{R} \tag{3.27}$$

maximal $|\mathrm{Tr}| + 1$ Äquivalenzregionen besucht. Ein effizienter MCMC Algorithmus lässt sich somit durch folgenden Pseudo-Algorithmus angeben:

Algorithmus 4 MCMC Line Sampling

Eingabe: Startkonfiguration $\boldsymbol{\varphi}_0$
Ausgabe: Markov Kette $\{\boldsymbol{\varphi}_1, \cdots, \boldsymbol{\varphi}_N\}$

for $i \in [0, N-1]$ **do**
  (1) Erzeuge zufällige Suchrichtung $\boldsymbol{d}_{i+1}$
  (2) Ziehe eine zufällige Position t auf dem Suchstrahl
  $\boldsymbol{\varphi}_{i+1} \leftarrow \boldsymbol{\varphi}_i + t \cdot \boldsymbol{d}_{i+1}$ ▷ Berechne neuen Parametervektor
end for

Statt direkt aus der P-dimensionalen Verteilung zu ziehen wird die a-Posteriori Verteilung auf zufällig gezogene Suchrichtungen $\boldsymbol{d}_i$ konditioniert. Diese konditionierten

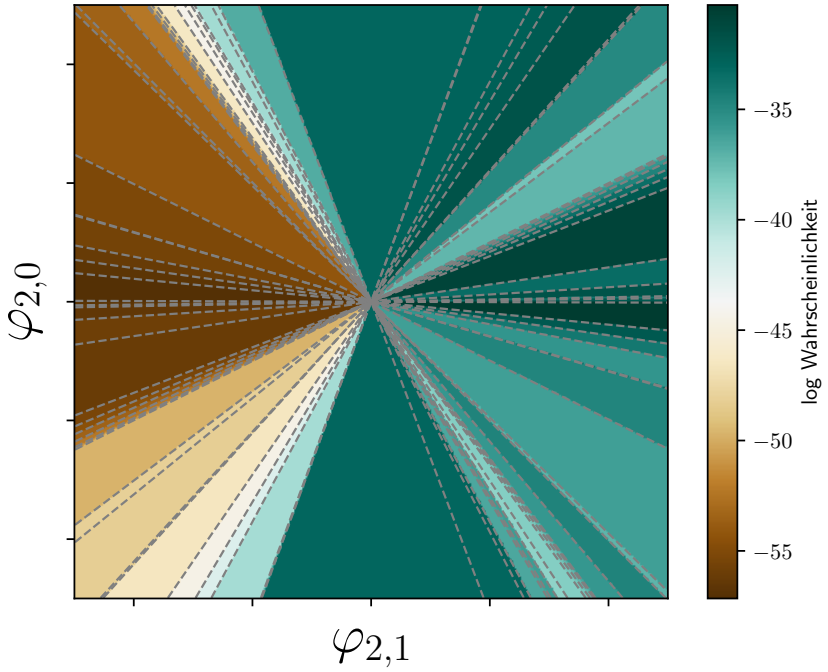


Abbildung 3.4: Heatmap der log-Likelihood für ein eindimensionales Klassifikationsproblem mit zwei normalverteilten Klassen ($\log(p(y = +1|\boldsymbol{x}, \boldsymbol{\varphi}))$). Die grau gestrichelten Linien repräsentieren die Sprungstellen zwischen Regionen konstanter Likelihood.

Verteilungen sind nun deutlich simpler und ermöglichen ein effizientes direktes Ziehen. Bei geschickter Wahl der Suchrichtungen gilt die Eigenschaft des detaillierten Gleichgewichts und somit die Konvergenz der Markov Kette. In den folgenden Abschnitten werden Implementierungen für die beiden Hauptkomponenten (1) und (2) dieses Pseudo-Algorithmus vorgestellt.

Erzeugung von Suchstrahlen

In Abschnitt 2.4 wurden bereits zwei MCMC Verfahren vorgestellt, welche das Problem der Generation von Stichproben aus mehrdimensionalen Verteilungen auf eine Folge von eindimensionalen Teilproblemen reduzierten. Beim *Gibbs Sampling* werden zum Beispiel nur Suchrichtungen verwendet, welche entlang der Einheitsvektoren des Parameterraums ausgerichtet sind. Beim *Hit-and-Run Sampling* hingegen werden die Suchrichtungen gleichverteilt auf der Oberfläche einer Hyperellipsoiden gezogen. Ein Generator für solche Suchrichtungen kann durch Normierung von Zufallsvektoren

aus einer multivariaten Normalverteilung implementiert werden:

$$\boldsymbol{d}' \sim \mathcal{N}(0, \Sigma) \tag{3.28}$$

$$\boldsymbol{d} = \frac{\boldsymbol{d}'}{\|\boldsymbol{d}'\|} \,. \tag{3.29}$$

Durch die Kovarianzmatrix Σ können allerdings bestimmte Suchrichtungen priorisiert werden. Falls sich die a-Posteriori Verteilung gut durch eine solche Normalverteilung approximieren lässt, wäre das *Hit-and-Run Sampling* bereits eine gute Wahl. Die hier verwendete a-Posteriori Verteilung ist jedoch multimodal und unstetig. Daher wird stattdessen eine adaptive Variante des *Hit-and-Run Sampling* verwendet, welche statt einer globalen Korrelationsstruktur die lokalen Korrelationen an Position $\boldsymbol{\varphi}_i$ aus der bereits erzeugten Markov Kette schätzt. Das von Gilks et al. [51] veröffentlichte Verfahren trägt die Bezeichnung Adaptive Direction Sampling (ADS). Unter dem Titel ADS wird tatsächlich eine Familie von adaptiven Sampling Verfahren zusammengefasst, bei der die Eigenschaft des detaillierten Gleichgewichts durch eine Korrektur der eindimensionalen konditionierten a-Posteriori Verteilung wiederhergestellt wird. Für den folgenden in [51] hergeleiteten Sonderfall entspricht diese Korrektur einer Identitätsabbildung und entfällt somit. Zunächst werden $M \geq 3$ unabhängige Markov Ketten $\{\boldsymbol{\varphi}_0^1, \cdots, \boldsymbol{\varphi}_0^M\}$ initialisiert. In jeder Iteration wird eine dieser Ketten zufällig gewählt und fortgesetzt. Die Suchrichtung ergibt sich aus der Differenz der aktuellen Zustände von zwei anderen Markov Ketten, welche unabhängig und ohne Zurücklegen selektiert wurden (Algorithmus 5). Die initialen Zustände sollten möglichst breit über den gesamten Parameterraum gestreut werden. Somit approximieren die ersten paar Suchrichtungen, welche durch ADS erzeugt werden, im Mittel die globale Korrelationsstruktur. Mit der Zeit clustern sich die Partikel mit einer hohen Wahrscheinlichkeit in der Nähe der Modalwerte der a-Posteriori Verteilung. Die Suchrichtungen adaptieren sich somit an die lokalen Korrelationen in diesen Umgebungen. Für den Fall $M < \mathcal{P}$ spannen die Differenzvektoren der Markov Ketten jedoch nur eine Untermenge des Parameterraumes auf. Die erzeugte Suchrichtungen können somit ebenfalls nur in demselben Unterraum liegen. Um dennoch den gesamten Parameterraum zu besuchen, wurde in [51] daher vorgeschlagen, in jeder Iteration nur mit einer Wahrscheinlichkeit p_{ADS} ADS zu verwenden und ansonsten auf *Hit-and-Run Sampling* mit einer Einheits Kovarianzmatrix auszuweichen. Die Variable p_{ADS} ist dabei ein unkritischer Parameter. In unseren Experimenten haben sich Werte im Intervall 80% bis 95% als sinnvoll erwiesen. Das gesamte Verfahren ist in Algorithmus 5 zusammengefasst. Im Vergleich zum Listing 4 wurde hier die Komponente (1) durch ADS ersetzt und die hierfür benötigten parallelen Markov Ketten eingeführt. Da die aktive Kette in jeder Iteration zufällig selektiert wird, werden die einzelnen Markov Ketten im Verlauf des Algorithmus unterschiedliche Längen aufweisen. Die Notation $\boldsymbol{\varphi}_{\triangleleft}^i$ bezeichnet daher die zuletzt gezogene Konfiguration der i-ten Kette. Im folgenden Abschnitt wird abschließend eine Implementierung für die Hauptkomponente (2) der präsentierten Simulationsmethode hergeleitet.

Algorithmus 5 Adaptive Direction Sampling

Eingabe: Startkonfigurationen $\{\boldsymbol{\varphi}_0^1, \cdots, \boldsymbol{\varphi}_0^M\}$
Ausgabe: Markov Ketten $\{\{\boldsymbol{\varphi}_1^m, \cdots, \boldsymbol{\varphi}_{N_m}^m\}\}_{m=1}^M$

for $n \in [1, N]$ **do**
  $m \sim [1, M]$ ▷ Selektiere aktive Markov Kette
  if $p_{\text{ADS}} < u, \quad u \sim Unif(0,1)$ **then** ▷ Generiere Suchrichtung
    $i \sim [1, M] \setminus m$
    $j \sim [1, M] \setminus \{m, i\}$
    $\boldsymbol{d}_n \leftarrow \boldsymbol{\varphi}_\triangleleft^i - \boldsymbol{\varphi}_\triangleleft^j$
  else
    $\boldsymbol{d}_n \sim \mathcal{N}(0, I)$
  end if
  $\boldsymbol{d}_n \leftarrow \boldsymbol{d}_n / \|\boldsymbol{d}_n\|$ ▷ Normalisiere Suchrichtung

  (2) Ziehe eine zufällige Position t auf dem Suchstrahl
  $\boldsymbol{\varphi}_{\triangleleft+1}^m \leftarrow \boldsymbol{\varphi}_\triangleleft^m + t \cdot \boldsymbol{d}_n$ ▷ Berechne neuen Parametervektor
end for

Effiziente Ziehung aus eindimensionalen Unterparameterräumen

Das ADS Verfahren erfordert das Ziehen von Konfigurationen aus der konditionierten a-Posteriori Verteilung $p(t|\boldsymbol{\varphi}_\triangleleft^m, \boldsymbol{d}, \text{Tr})$. Da es sich um eine eindimensionale Verteilung handelt, kann sie durch die skalare Variable t parametrisiert werden. Die Variable t stellt somit die Position entlang der Suchlinie dar, wobei der aktuelle Zustand $\boldsymbol{\varphi}_\triangleleft^m$ im Ursprung dieses lokalen Koordinatensystems liegt. In Abbildung 3.5 wird dies anhand eines Beispielproblems dargestellt. Die Heatmap repräsentiert die stückweise konstante Likelihood (Formel (3.15)) und rote Kreisbögen die a-Priori Verteilung (Formel (3.26)). Durch ADS könnte zum Beispiel die rote vertikale Suchlinie gewählt worden sein. In diesem Abschnitt wird eine effiziente Methode vorgestellt, um eine neue Konfiguration anhand der Likelihood auf diesem Suchstrahl zu erzeugen. Die a-Priori Verteilung wird durch Projektion dieser Konfiguration auf die Einheits-Hyperkugel berücksichtigt. Diese Projektion erfordert jedoch Suchrichtungen, welche tangential zur Einheits-Hyperkugel verlaufen. Diese Bedingung wird vor dem Ziehen durch Projektion der Suchrichtung auf die Tangentialebene der Hyperkugel und anschließender Renormalisierung sichergestellt (Algorithmus 6).

Zunächst kann beobachtet werden, dass der Suchstrahl maximal $|\text{Tr}| \cdot (|\mathcal{K}| - 1)$ Unstetigkeitsstellen der Likelihood schneidet. Jeder Merkmalsvektor der Lernstichprobe sorgt dabei für bis zu $|\mathcal{K}| - 1$ dieser Übergänge. Für jede Instanz $(\boldsymbol{x}_i, y_i) \in \text{Tr}$ ergeben sich zunächst aus Formel (3.23) durch Berechnung aller paarweisen Schnittpunkte t_{c_1,c_2} zwischen den linearen Antwortfunktion der Klassen c_1 und c_2 potentielle

Positionen für die Unstetigkeitsstellen:

$$(\boldsymbol{\varphi}_{c_1} + t \cdot \boldsymbol{d}_{c_1})^T \boldsymbol{x}_i + \varphi_{c_1,0} + t \cdot d_{c_1,0} = (\boldsymbol{\varphi}_{c_2} + t \cdot \boldsymbol{d}_{c_2})^T \boldsymbol{x}_i + \varphi_{c_2,0} + t \cdot d_{c_2,0} \tag{3.30}$$

$$\Rightarrow t_{c_1,c_2} = \frac{(\boldsymbol{\varphi}_{c_2}^T \boldsymbol{x}_i + \varphi_{c_2,0}) - (\boldsymbol{\varphi}_{c_1}^T \boldsymbol{x}_i + \varphi_{c_1,0})}{(\boldsymbol{d}_{c_1}^T \boldsymbol{x}_i + d_{c_1,0}) - (\boldsymbol{d}_{c_2}^T \boldsymbol{x}_i + d_{c_2,0})}\,. \tag{3.31}$$

Die gültigen Schnittpunkte, welche den tatsächlichen Unstetigkeitsstellen entsprechen, müssen jedoch auch die Bedingung

$$\begin{aligned}(\boldsymbol{\varphi}_{c_1} + t_{c_1,c_2} \cdot \boldsymbol{d}_{c_1})^T \boldsymbol{x}_i + &\varphi_{c_1,0} + t_{c_1,c_2} \cdot d_{c_1,0} \\ &= \arg\max_{c \in \mathcal{K}} (\boldsymbol{\varphi}_c + t_{c_1,c_2} \cdot \boldsymbol{d}_c)^T \boldsymbol{x} + \varphi_{c,0} + t_{c_1,c_2} \cdot d_{c,0}\end{aligned} \tag{3.32}$$

erfüllen. D. h. sie müssen auf der oberen Einhüllenden aller Antwortfunktionen liegen. Diese kann in linearer Laufzeitkomplexität, z. B. durch Dynamische Programmierung berechnet werden. Der hier verwendete Algorithmus wurde aus [96] entnommen. Abbildung 3.6 zeigt ein Beispiel einer solchen oberen Einhüllenden für drei Klassen. Neben den gesuchten Unstetetigkeitsstellen liefert der Algorithmus Intervalle über t in denen die durch $\boldsymbol{\varphi} + t \cdot \boldsymbol{d}$ parametrisierte Entscheidungsfunktion den Merkmalsvektor $\boldsymbol{x}_i$ jeweils den Klassen aus $\mathcal{K}$ zuordnet. Nicht immer lassen sich auf dem gegebenen Suchstrahl Konfigurationen für jede der möglichen Klassen finden. Im Beispiel 3.6 kann sich keiner der parametrisierbaren Funktionen für die Klasse 1 entscheiden. Die so über die gesamte Lernstichprobe berechneten Übergänge werden nun nach ihrer Position t auf dem Suchstrahl sortiert. Diese Sortierung hat eine Laufzeitkomplexität von $\mathcal{O}(|\mathrm{Tr}| \cdot |\mathcal{K}| \cdot \log(|\mathrm{Tr}| \cdot |\mathcal{K}|))$ und beschleunigt die nachfolgenden Berechnungsschritte. Die Unstetigkeitsstellen wurden bisher auf dem linearen Suchstrahl lokalisiert. Das Ziehen der nächsten Konfiguration $\boldsymbol{\varphi}_{i+1}$ erfolgt jedoch auf der Oberfläche der Einheits-Hyperkugel. Die berechneten und sortierten Positionen werden daher anschließend auf die Oberfläche dieser Hyperkugel projiziert (Abbildung 3.5). Die Positionen t_i liegen somit auf einem zufällig ausgerichteten Kreisbogen dessen Ursprung bei $t = 0$ liegt und welcher immer jeweils eine Hyperhalbkugel umspannt. Die lokalen Winkelkoordinaten auf dieser Halbkugel werden durch $t' = \tan^{-1}(t), t' \in (-\pi/2, \pi/2)$ bestimmt. Die Intervalle zwischen zwei aufeinanderfolgenden Unstetigkeitsstellen besitzen eine stückweise konstante Likelihood $p(C|\boldsymbol{\alpha})$. Im Vergleich zur Formel (3.15) müssen die Einträge der Konfusionsmatrix $C_{c,p}$ nicht explizit aus der Lernstichprobe berechnet werden, da die Konfusionsmatrix stetig aktuell gehalten wird. Die Intervalle $(t'_i, t_{i'+1})$ bilden zusammen mit den dazugehörigen Intervallwahrscheinlichkeiten $(t'_{i+1} - t'_i) \cdot p(C|\boldsymbol{\alpha})$ eine diskrete Wahrscheinlichkeitsverteilung. Das Ziehen einer neuen Konfiguration auf dem Kreisbogensegment erfolgt nun in zwei Schritten. Zunächst wird ein neues Intervall aus dieser Verteilung gezogen. Durch den sogenannten *Gumbel-Max Trick* [135] konnten hier leichte Effizienzsteigerungen erreicht werden. Anschließend wird innerhalb dieses Intervalls die neue Konfiguration $\boldsymbol{\varphi}_{i+1}$ aus einer Gleichverteilung gezogen. Das vollständige Verfahren ist in Algorithmus 6

wiedergegeben. Die Laufzeitkomplexität ergibt sich pro Iteration bei Verwendung des Kernel Tricks zu $\mathcal{O}(|\mathrm{Tr}|^2|\mathcal{K}| + |\mathrm{Tr}||\mathcal{K}|^2)$.

Algorithmus 6 Ziehen aus 1D Verteilung

Eingabe: Konfiguration $\boldsymbol{\varphi}_i$, Richtung $\boldsymbol{d}_i$
Ausgabe: Konfiguration $\boldsymbol{\varphi}_{i+1}$

Erzeuge zufällige Suchrichtung:
$\boldsymbol{d}_i \leftarrow \boldsymbol{d}_i - \boldsymbol{\varphi}_i \cdot \langle \boldsymbol{\varphi}_i, \boldsymbol{d}_i \rangle$ ▷ Richtung auf Tangentialebene projizieren
$\boldsymbol{d}_i \leftarrow \boldsymbol{d}_i / \|\boldsymbol{d}_i\|$ ▷ Normalisiere Suchrichtung

Erzeuge Liste aller Unstetigkeitsstellen entlang der Suchlinie. Die Variable c_i entspricht der tatsächlichen Klasse, p'_i sowie p_i jeweils dem Klassifikationsergebnis vor und nach der Unstetigkeitsstelle.
$[(t_i, c_i, p'_i, p_i)] \leftarrow$ gemäß (3.31), (3.32)
$[(t'_i, c_i, p'_i, p_i)] \leftarrow [(\tan^{-1}(t_i), c_i, p'_i, p_i)]$ ▷ Projiziere Schnittpunkte auf Einheitshyperkugel

$C \leftarrow C(t' = -\pi/2)$ ▷ Initialisiere die Konfusionsmatrix für die Position $t' = -\pi/2$

Besuche Schnittpunkte t'_i in aufsteigender Reihenfolge und ziehe Intervall $[t'_i, t'_{i+1}]$:
$p_{\max} \leftarrow 0$

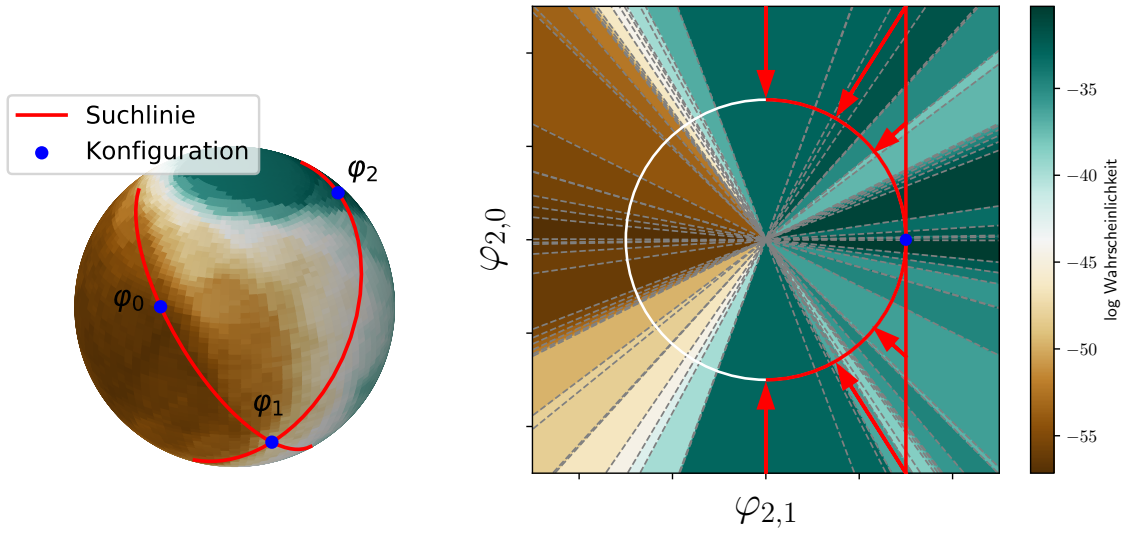
for $(t'_j, c_j, p'_j, p_j) \in sorted([(t'_i, c_i, p'_i, p_i)])$ **do**
  Aktualisiere Konfusionsmatrix:
  $C_{c_j, p'_j} \leftarrow C_{c_j, p'_j} - 1$
  $C_{c_j, p_j} \leftarrow C_{c_j, p_j} + 1$

  Berechne Likelihood nach Formel (3.15):
  $L \leftarrow p(C(\mathrm{Tr}, \boldsymbol{\varphi})|\boldsymbol{\alpha}) = p(\mathrm{Tr}|\boldsymbol{\varphi}, \boldsymbol{\alpha})$

  Ziehe zufällige Perturbierung für den Gumbel-Max Trick:
  $u \sim Unif(0, 1)$
  $G \leftarrow -log(-log(u))$

  Merke den Start des gezogenen Intervalls:
  if $log(L) + G > p_{\max}$ **then**
    $p_{\max} \leftarrow log(L) + G$
    $t'_{\max} \leftarrow t'_j$
  end if
end for

$t'_{i+1} \sim Unif(t'_{\max}, t'_{\max+1})$ ▷ Ziehe gleichverteilte Position aus dem Intervall
$t_{i+1} \leftarrow \tan(t'_{i+1})$ ▷ t_{i+1} von Hyperkugel auf Suchlinie zurück projizieren
$\boldsymbol{\varphi}_{i+1} \leftarrow \boldsymbol{\varphi}_i + t_{i+1} \cdot \boldsymbol{d}_i$ ▷ Erzeuge neue Konfiguration
$\boldsymbol{\varphi}_{i+1} \leftarrow \boldsymbol{\varphi}_{i+1} / \|\boldsymbol{\varphi}_{i+1}\|$ ▷ Normalisiere Konfiguration



(a) Die Suchlinien spannen Kreisbögen auf, welche jeweils die halbe Hyperkugel umspannen. Somit können mit wenigen Iterationen große Strecken zurückgelegt werden. Hier konnte aus der Startkonfiguration φ_0 bereits nach zwei Iterationen die deutlich bessere Konfiguration φ_2 erreicht werden.

(b) Aufgrund der a-Priori Verteilung dürfen gültige Lösungen nur auf dem weißen Einheitskreis liegen. Die Ziehung von neuen Konfigurationen geschieht jedoch zunächst antlang der roten Suchlinie (vertikal) und wird anschließend auf den gültigen Lösungsraum (rotes Kreissegment) projiziert. Aufgrund dieser Projektion liegen alle Konfigurationen auf Kreisbögen, welche lokal von $-\pi/2$ bis $\pi/2$ parametrisiert werden können.

Abbildung 3.5: Heatmap einer log-Likelihood für ein Beispielproblem. Gültige Lösungen liegen auf der Oberfläche der Einheits-Hyperkugel. (a) Neue Konfigurationen werden durch Sampling auf der Kugeloberfläche erzeugt. (b) In jedem Schritt werden die Schnitte der Suchlinie mit den Unstetigkeitsstellen der Likelihood entlang der Tangentialebene, welche um die aktuelle Konfiguration aufgespannt wird, berechnet. Anschließend werden die berechneten Schnittpunkte auf die Hyperkugel projiziert.

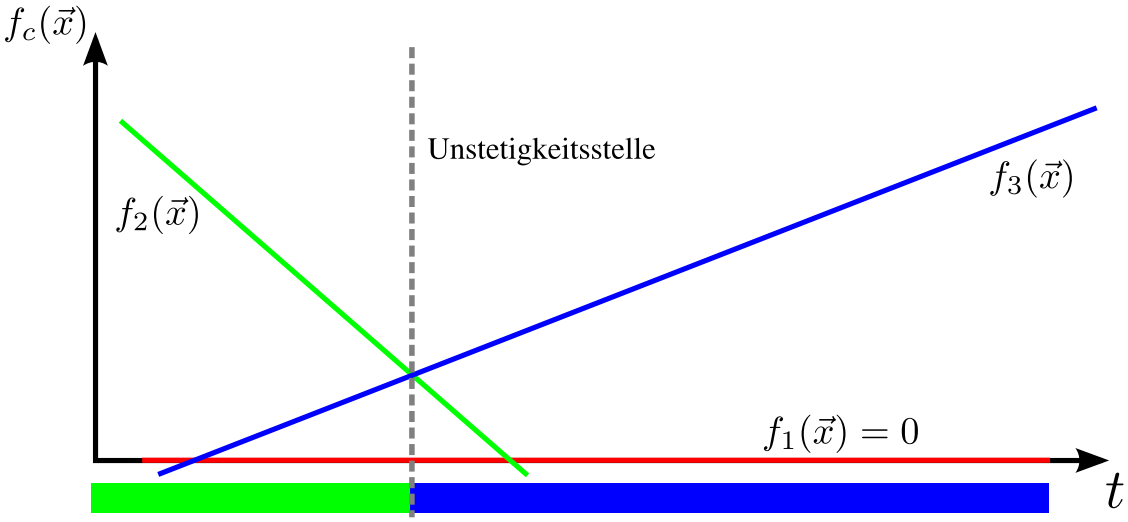


Abbildung 3.6: Lineare Antwortfunktionen für ein konkretes Lernbeispiel $\boldsymbol{x}$ für ein Dreiklassenproblem. Jede Position t der Suchlinie beschreibt eine parametrisierbare Entscheidungsfunktion. Das Lernbeispiel wird von diesem Klassifikator immer der Klasse zugeordnet, deren Antwortfunktion zur oberen Einhüllenden gehört. Übergänge zwischen Äquivalenzklassen von Entscheidungsfunktionen entstehen somit nur an Schnittpunkten zwischen den Antwortfunktionen. Für die Antwortfunktion der Referenzklasse (rot) gilt immer $f_1(\boldsymbol{x}) = 0$. Sie bildet somit eine untere Grenze der oberen Einhüllenden.

Kapitel 4

Approximatives Quellenselektionsmodell für große Datensätze

Die Laufzeitkomplexität des im vorherigen Kapitel vorgestellten MCMC basierten Schätzverfahrens erlaubt nur eine sinnvolle Anwendung auf kleinen Beispieldatensätzen. In diesem Kapitel wird daher eine unimodale Approximation des Transfermodells entwickelt. Eine Erweiterung dieses Modells erlaubt weiterhin den gleichzeitigen Transfer aus mehreren Quellen. Alle Parameter und Hyperparameter werden durch ein hocheffizientes *Boosting* basiertes Gradientenabstiegsverfahren geschätzt, wodurch die Verarbeitung von sehr großen Datensätzen erst möglich wird.

4.1 Konvexe Relaxation

Das Bayessche Quellenselektionsmodell wurde in Abschnitt 3.2 durch die Formel

$$\hat{S} = \arg\min_{s \in S} \mathrm{E}[\epsilon(h_s(\boldsymbol{x}), \mathrm{T}_s) + 2d_{\mathrm{TV}}(\mathrm{T}_T, \mathrm{T}_s) | \mathrm{T}_T, \mathrm{T}_s] \qquad \text{(siehe 3.9)}$$

beschrieben. Die a-Posteriori Verteilung dieses Modells war aufgrund der unstetigen 0-1 Kosten multimodal und hochgradig nicht-konvex. Im Allgemeinen sind jedoch konvexe Verteilungen wesentlich einfacher zu handhaben. Dies rührt aus einer Menge von Eigenschaften, welche zusammengenommen garantieren, dass jedes lokale Optimum einer Funktion gleichzeitig ihr globales Optimum ist. Somit kann der Modalwert der Funktion durch einfache und mathematisch gut verstandene Optimierungsverfahren gefunden werden. Die notwendigen Bedingungen für die Konvexität eines Optimierungsproblems lauten:

Konvexität der Zielfunktion: Eine Funktion $f(\boldsymbol{a})$ ist konvex falls für alle $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{D}$ gilt $f(\lambda \cdot \boldsymbol{a} + (1-\lambda) \cdot \boldsymbol{b}) \leq \lambda \cdot f(\boldsymbol{a}) + (1-\lambda) \cdot f(\boldsymbol{b})$ für $\lambda \in [0, 1]$.

Konvexität des Definitionsgebiets: Ein Definitionsgebiet $\mathbb{D}$ ist konvex falls für alle $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{D}$ gilt $\lambda \cdot \boldsymbol{a} + (1-\lambda) \cdot \boldsymbol{b} \in \mathbb{D}$ für $\lambda \in [0, 1]$.

Diese Bedingungen beziehen sich auf Minimierungsprobleme. Maximierungsprobleme werden verallgemeinert auch als konvex bezeichnet falls die genannten Bedingungen auf die modifizierte Zielfunktion $f^{\circ}(\boldsymbol{a}) = -f(\boldsymbol{a})$) zutreffen. Die *konvexe Relaxation* beschreibt ein Vorgehen, bei dem die Zielfunktion eines Optimierungsproblems durch eine konvexe Zielfunktion ersetzt wird ohne dabei die Position des globalen Optimums wesentlich zu verändern. Wie in Abschnitt 2.2.2 geschildert wurde, werden Entscheidungsmodelle in der Regel durch Austausch der Kostenfunktion relaxiert. Einige Alternativen zum 0-1 Kosten wurden dort bereits genannt. Im Folgenden wird eine konvexe Relaxation des Transfermodells aus Kapitel 3 entwickelt, bei der die 0-1 Kosten durch die *linearen Kosten*

$$L_{\mathrm{linear}}(\boldsymbol{x}, y, \boldsymbol{\varphi}) = y \cdot d(\boldsymbol{x}) \qquad (4.1)$$

ersetzt werden. Unter dieser Änderung und der Verwendung des Kernel Tricks ergibt sich eine Approximation der TV Distanz $d_{\mathrm{TV}}(.,.)$ zu

$$\begin{aligned} d^2_{\mathrm{MMD}}(\mathrm{T}_T, \mathrm{T}_S) &= \|\mu_T - \mu_S\|^2_{\mathcal{H}} \\ &= \langle \mu_T, \mu_T \rangle_{\mathcal{H}} + \langle \mu_S, \mu_S \rangle_{\mathcal{H}} - 2\langle \mu_T, \mu_S \rangle_{\mathcal{H}} \\ &= \mathrm{E}[\langle \phi(\boldsymbol{x}_T), \phi(\boldsymbol{x}'_T) \rangle] + \mathrm{E}[\langle \phi(\boldsymbol{x}_S), \phi(\boldsymbol{x}'_S) \rangle] - 2\,\mathrm{E}[\langle \phi(\boldsymbol{x}_T), \phi(\boldsymbol{x}_S) \rangle] \qquad (4.2) \\ &= \mathrm{E}[k(\boldsymbol{x}_T, \boldsymbol{x}'_T)] + \mathrm{E}[k(\boldsymbol{x}_S, \boldsymbol{x}'_S)] - 2\,\mathrm{E}[k(\boldsymbol{x}_T, \boldsymbol{x}_S)]\ . \qquad (4.3) \end{aligned}$$

Der Beweis für den Zusammenhang zwischen dem statistischen Klassifikationsmodell aus Kapitel 3 und dieser Approximation wird im Appendix A.1 gegeben. Diese Distanz zwischen Stichproben wird in der Literatur als Maximum Mean Discrepancy (MMD) bezeichnet [54]. Die Symbole μ_T und μ_S bezeichnen die Kernel Einbettungen der Verteilungsmittelwerte in einem gegebenen Hilbertraum $\mathcal{H}$. Wird die Einbettung durch Wahl einer Kernelfunktion $k(.,.)$ geschickt gewählt, dann sind diese Mittelwerte dazu in der Lage, eine Vielzahl der Eigenheiten der Verteilungen im Hilbertraum abzubilden. In dieser Formel ist $\boldsymbol{x}'$ als Merkmalsvektor aus einer Stichprobe T' zu verstehen, bei der es sich um eine zweite unabhängige Stichprobe aus derselben Verteilung handelt, aus der bereits T gezogen wurde. Aus dieser vorausgesetzten statistischen Unabhängigkeit der Stichproben ergibt sich weiterhin die Gültigkeit der Umformung zur Gleichung (4.2). Für positive beschränkte Kernel konnte in [123] gezeigt werden, dass folgende Beziehung zwischen der MMD und der TV Distanz gilt:

$$d_{\mathrm{MMD}}(\mathrm{T}_T, \mathrm{T}_S) \leq d_{\mathrm{TV}}(\mathrm{T}_T, \mathrm{T}_S)\,. \qquad (4.4)$$

Tatsächlich existieren eine Vielzahl von Parallelen zwischen diesen auf den ersten Blick sehr unterschiedlichen Distanzmaßen. Insbesondere kann die MMD, ähnlich zu Formel (3.6), als Statistik eines Klassifikationsmodells interpretiert werden. Daher ist die MMD in vielerlei Hinsicht als Approximation der TV Distanz in Formel (3.4) zu empfehlen. Der empirische Fehler der Quelle kann laut Formel (A.7) durch Ersetzen der 0-1 Kosten durch die linearen Kosten ebenfalls aus der MMD geschätzt werden. Das modifizierte Transfermodell lautet somit

$$\hat{S} = \operatorname*{arg\,min}_{s \in S} \epsilon_{\mathrm{MMD}}(\mathrm{T}_s) + 2 d_{\mathrm{MMD}}(\mathrm{T}_T, \mathrm{T}_s)\,. \qquad (4.5)$$

Die Laufzeitkomplexität der empirischen Schätzung von d^2_{MMD} spielt eine wesentliche Rolle für die effiziente Auswertung des Transfermodells auf sehr großen

Datensätzen. Der unverzerrte Schätzer mit minimaler Varianz hat die Form

$$\begin{aligned} d^2_{\mathrm{MMD}_u}(\mathrm{T}_T, \mathrm{T}_S) = &\frac{1}{|\mathrm{T}_T||\mathrm{T}'_T|} \sum_{i=1}^{|\mathrm{T}_T|} \sum_{j=1}^{|\mathrm{T}'_T|} k(\boldsymbol{x}_{T,i}, \boldsymbol{x}'_{T,j}) \\ &+ \frac{1}{|\mathrm{T}_S||\mathrm{T}'_S|} \sum_{i=1}^{|\mathrm{T}_S|} \sum_{j=1}^{|\mathrm{T}'_S|} k(\boldsymbol{x}_{S,i}, \boldsymbol{x}'_{S,j}) \\ &- \frac{2}{|\mathrm{T}_T||\mathrm{T}_S|} \sum_{i=1}^{|\mathrm{T}_T|} \sum_{j=1}^{|\mathrm{T}_S|} k(\boldsymbol{x}_{T,i}, \boldsymbol{x}_{S,j}) \,. \end{aligned} \tag{4.6}$$

und weist eine quadratische Laufzeitkomplexität auf. Für große Datensätze wurde von Gretton et al. ein Schätzer mit einer linearen Laufzeitkomplexität vorgestellt [54]. Unter der Bedingung $|\mathrm{T}_T| = |\mathrm{T}_S| = M$ lautet der Schätzer

$$d^2_{\mathrm{MMD}_1}(\mathrm{T}_T, \mathrm{T}_S) = \frac{2}{M} \left[\sum_{i=1}^{M/2} k(\boldsymbol{x}_{T,2i}, \boldsymbol{x}_{T,2i-1}) + \sum_{i=1}^{M/2} k(\boldsymbol{x}_{S,2i}, \boldsymbol{x}_{S,2i-1}) - \sum_{i=1}^{M} k(\boldsymbol{x}_{T,i}, \boldsymbol{x}_{S,i}) \right] . \tag{4.7}$$

Hierfür muss jedoch gelten, dass T_T und T_S unabhängigen gleichverteilten Stichproben entsprechen. Aufeinanderfolgende Merkmalsvektoren sind somit statistisch unabhängig. Der Schätzer $d^2_{\mathrm{MMD}_1}$ besitzt für eine gegebene Stichprobengröße eine geringere Effizienz als $d^2_{\mathrm{MMD}_u}$ und somit auch einen größeren Schätzfehler. Jedoch konnte empirisch gezeigt werden, dass sich dieser Umstand bei Fixierung der Laufzeit umkehrt, da $d^2_{\mathrm{MMD}_1}$ bei gegebener Rechenzeit deutlich mehr unterschiedliche Merkmalsvektoren in die Schätzung einfließen lässt.

4.2 Multiquellenselektion

Das bisher vorgestellte Transfermodell erlaubt nur den Transfer aus einer einzelnen Quelle. Dieses Vorgehen wurde in Formel (3.2) durch einen NN Regressor modelliert, welcher annimmt, dass die zur Verfügung stehenden Quellen den Raum aller Verteilungen repräsentativ und dicht abdecken. Falls jedoch Q sehr klein ist, bzw. die Variabilität in den Untersuchungsgebieten besonders hoch ist, kann diese Annahme nicht erfüllt sein. In diesem Abschnitt wird daher das bisherige Regressionsmodell um eine Multiquellenselektion erweitert. Für dieses erweiterte Transfermodell müssen wir zusätzlich annehmen, dass alle Verteilungen, welche gültigen Lernproblemen zugeordnet sind, in einem Untervektorraum von $\mathcal{S}$ liegen. Somit repräsentieren alle Linearkombinationen in S neue gültige Quellen. Der Regressor hat dann die Form

$$p_{\bar{S}}(\boldsymbol{x}, y) = \sum_{i=1}^{Q} \pi_i \cdot p_{S_i}(\boldsymbol{x}, y) \tag{4.8}$$

$$h_T(\boldsymbol{x}) = f(S, \boldsymbol{\pi}) = \arg\max_{y \in \mathcal{K}} p_{\bar{S}}(y|\boldsymbol{x}) \tag{4.9}$$

und das erweiterte Multiquellen-Transfermodell ergibt sich zu

$$\hat{S}_{\boldsymbol{\pi}} = \arg\min_{\boldsymbol{\pi}\in\mathbb{R}^Q} \epsilon_{\mathrm{MMD}}(\mathrm{T}_{S_{\boldsymbol{\pi}}}) + 2d_{\mathrm{MMD}}(\mathrm{T}_T, \mathrm{T}_{S_{\boldsymbol{\pi}}})\,. \tag{4.10}$$

Das Schema dieses Transfer Verfahrens ist in Abbildung 4.1 dargestellt. Wir bezeichnen im Folgenden die Quelle $\bar{S}_{\boldsymbol{\pi}}$ als synthetisierte Quelle. Damit jedoch $p_{\bar{S}_{\boldsymbol{\pi}}}(\boldsymbol{x}, y)$ eine gültige Wahrscheinlichkeitsverteilung ergibt, müssen zusätzliche Nebenbedingungen an den Gewichtsvektor $\boldsymbol{\pi}$ gestellt werden. Für alle Wahrscheinlichkeitsverteilungen $p(x)$ muss gelten $p(x) \geq 0$ sowie $\int_{x\in X} p(x)\,dx = 1$. Aus diesen Eigenschaften folgen trivialerweise die äquivalenten Bedingungen $\pi_i \geq 0$ und $\|\boldsymbol{\pi}\|_1 = 1$. Bei der Linearkombination in Formel (4.8) handelt es sich daher tatsächlich um eine Konvexkombination und die Gewichte $\boldsymbol{\pi}$ müssen als baryzentrische Koordinaten verstanden werden. Weiterhin entspricht die Definition von $p_{\bar{S}_{\boldsymbol{\pi}}}(\boldsymbol{x}, y)$ genau der einer Mischverteilung. Somit entspricht die Vereinigung der gewichteten Quellenstichproben T_{S_i} einer repräsentativen Stichprobe aus $\bar{S}_{\boldsymbol{\pi}}$:

$$\mathrm{T}_{\bar{S}_{\boldsymbol{\pi}}} = \bigcup_{i=1}^{Q} \{\boldsymbol{x}_{S_i,r}, y_{S_i,r}, \pi_i\}_{r=1}^{N_{S_i}}\,. \tag{4.11}$$

Letztendlich ist es das Ziel einen Klassifikator aus $\mathrm{T}_{\bar{S}_{\boldsymbol{\pi}}}$ anzulernen. Für einige Verfahren, wie z. B. der logistischen Regression, Entscheidungsbäumen oder der SVM existieren bereits Implementierungen, welche die Instanzengewichte π_i korrekt verarbeiten können. Ansonsten kann $\mathrm{T}_{\bar{S}_{\boldsymbol{\pi}}}$ durch ein *Resampling* durch eine neue Stichprobe ersetzt werden, welche diese auch ohne Instanzengewichte gut approximiert.

Wie in Abbildung 4.2 zu sehen ist, kann durch die Interpolation zwischen den vorhandenen Quellen eine beliebige Zielverteilung potentiell deutlich genauer approximiert werden. Doch hierfür muss der korrekte Parametervektor $\boldsymbol{\pi}$ zunächst ermittelt werden. Im nächsten Abschnitt wird daher ein Verfahren zur Lösung dieses Parameterschätzproblems vorgestellt. Weiterhin muss die Formel zur Berechnung der MMD angepasst werden um die Berechnung der Distanz zwischen einer gegebenen Zielverteilung und einer synthetisierten Quelle zu ermöglichen. Die modifizierte Distanz mit gewichteten Quellenstichproben wird in dieser Arbeit definiert zu

$$\begin{aligned} d^2_{\mathrm{MMD}_1}(\mathrm{T}_T, \mathrm{T}_{\bar{S}}) = \frac{2}{M}\Big[&\sum_{i=1}^{M/2} k(\boldsymbol{x}_{T,2i}, \boldsymbol{x}_{T,2i-1}) \\ &+ \sum_{j=1}^{Q} \pi_j^2 \sum_{i=1}^{M/2} k(\boldsymbol{x}_{S_j,2i}, \boldsymbol{x}_{S_j,2i-1}) + \sum_{j=2}^{Q}\sum_{k=1}^{j-1} \pi_j\pi_k \sum_{i=1}^{M} k(\boldsymbol{x}_{S_j,i}, \boldsymbol{x}_{S_k,i}) \\ &- \sum_{j=1}^{Q} \pi_j \sum_{i=1}^{M} k_\sigma(\boldsymbol{x}_{T,i}, \boldsymbol{x}_{S_j,i})\Big]\,. \end{aligned} \tag{4.12}$$

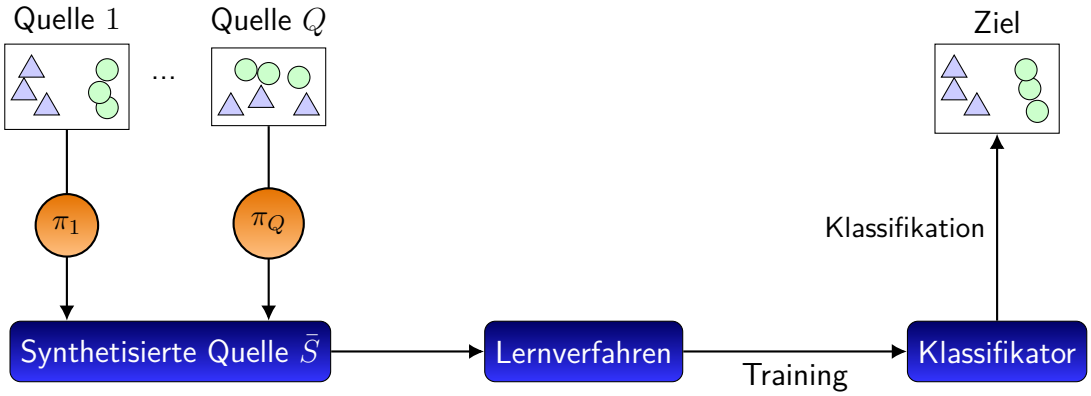


Abbildung 4.1: Diagramm eines Systems zur Multiquellenselektion. Die synthetisierte Quelle $\bar{S}$ entsteht durch Vereinigung der mit π_i gewichteten Quellenkandidaten. Der Klassifikator wird somit direkt aus $\bar{S}$ angelernt.

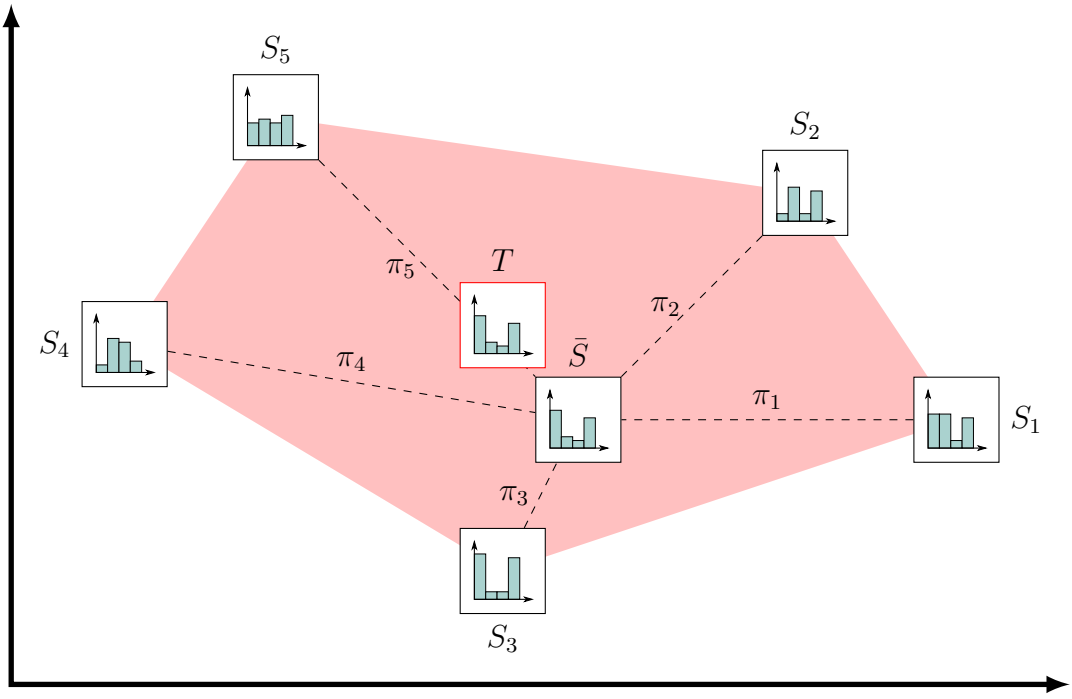


Abbildung 4.2: Synthetisierung von Quellenverteilungen durch Kombination der Quellen $S_1 \ldots S_5$. Es wird angenommen, dass der Raum aller gültigen Verteilungen durch die konvexe Hülle von S_i (in rot) aufgespannt wird. Die Quellengewichte π_i können somit als baryzentrische Koordinaten verstanden werden.

4.3 Konvexe Optimierung durch Boosting

Die soweit vorgestellte Multiquellenselektion lässt sich durch das Optimierungsproblem

$$\hat{S_{\boldsymbol{\pi}}} = \underset{\boldsymbol{\pi}\in\mathbb{R}^Q}{\arg\min}\, \epsilon_{\mathrm{MMD}}(\mathrm{T}_{S_{\boldsymbol{\pi}}}) + 2d_{\mathrm{MMD}_1}(\mathrm{T}_T, \mathrm{T}_{S_{\boldsymbol{\pi}}})$$
$$\text{mit den Nebenbedingungen} \quad \boldsymbol{\pi} \geq 0 \quad \text{und} \quad |\boldsymbol{\pi}|_1 = 1 \tag{4.13}$$

beschreiben. Somit kommen zu dessen Lösung eine ganze Reihe von Verfahren der konvexen Optimierung in Frage. Für konvexe Optimierungsprobleme ohne Nebenbedingungen gelten iterative Methoden, wie der Gradientenabstieg oder die Newton Methode, als ausreichend effizient. Verfahren dieser Familien nutzen Information des Gradienten sowie der Hessematrix um mit jeder Iteration i einen neuen Parametersatz $\boldsymbol{\beta}_{i+1}$ zu bestimmen, welcher bezüglich der Zielfunktion niedrigere Kosten erreicht als $\boldsymbol{\beta}_i$. Aufgrund der Konvexität der Zielfunktion wird das Optimum somit aus jedem gültigen Startpunkt $\boldsymbol{\beta}_0$ erreicht. Die Verfahren unterscheiden sich daher im Wesentlichen darin, wie in jeder Iteration der Sprungvektor $\Delta\boldsymbol{\beta}_i = \boldsymbol{\beta}_{i+1} - \boldsymbol{\beta}_i$ bestimmt wird. Die Richtung und Länge von $\Delta\boldsymbol{\beta}$ können als separate Komponenten betrachtet werden und tragen in Gradientenabstiegsverfahren die Bezeichnungen Suchrichtung und Schrittweite. Für das gegebene Problem aus Formel (4.3) werden diese beiden Komponenten im Folgenden separat betrachtet.

Die optimale Suchrichtung ergibt sich aus dem Gradienten und der Hessematrix der Zielfunktion. Die konkrete Berechnung dieser Größen ist jedoch aufwändig. Insbesondere muss auch darauf geachtet werden, dass die Hessematrix einen quadratischen Speicherverbrauch bezüglich der Anzahl der Parameter aufweist. Ein optimaler Gradientenabstieg ist daher bei vielen Kandidatenquellen nicht effizient umsetzbar. Aus den Abbildungen 3.1 und 4.2 wird jedoch ersichtlich, dass sich für sehr große Q jede Zielverteilung durch eine gewichteten Kombination von sehr wenigen ähnlichen Quellenkandidaten approximieren lässt. Der gesuchte Parametervektor $\boldsymbol{\beta} := \boldsymbol{\pi}$ wäre somit auf realen Daten dünn besetzt, d. h. es gilt $\pi_q \approx 0$ für fast alle $q \in [1{,}Q]$. Aus dieser Beobachtung folgt, dass der Gradient im Parameterraum nahezu achsenparallel ausgerichtet ist. Eine Beschränkung der Suchrichtungen auf die Q Koordinatenachsen reduziert die Komplexität der Optimierung und führt zu der Klasse der Koordinatenabstiegsverfahren. Die beste Suchrichtung wird in jeder Iteration durch Ausprobieren ermittelt.

Wurde eine Suchrichtung gewählt, folgt anschließend die Bestimmung der Schrittweite. Auf die explizite Berechnung des Gradienten und der Hessematrix soll weiterhin verzichtet werden. Die Schrittweite könnte durch eine benutzergesteuerte Konstante festgelegt werden. Diese einfache Strategie hat jedoch zwei entscheidende Nachteile. Wird die Schrittweite zu gering gewählt, dann können in jeder Iteration nur sehr kleine Sprünge erreicht werden. Somit werden sehr viele Iterationen zur Konvergenz

benötigt. Wird die Schrittweite allerdings zu groß eingestellt, kann die Suche über das Minimum der Zielfunktion hinausspringen. Der Parametervektor umkreist somit das Optimum statt zu ihm zu konvergieren. Um diese Probleme zu umgehen wird die Schrittweite oft adaptiert. Diese Adaption kann direkt an die Eigenheiten der Zielfunktion erfolgen. In einfacheren Ansätzen wird die Schrittweite jedoch durch eine Lernrate geregelt. Somit können zu Beginn der Optimierung große Sprünge zurückgelegt werden um schnell in die Nähe des Optimums zu gelangen. In späteren Iterationen wird die Schrittweite langsam reduziert um eine garantierte Konvergenz zu erreichen. Die Zuordnung von Iterationsnummer zu Schrittweite wird als Plan bezeichnet. Im Allgemeinen stellt die Lernrate einen kritischen Parameter von solchen Optimierungsverfahren dar und muss daher problemabhängig eingestellt werden. Für das Optimierungsproblem in (4.3) kann jedoch das Vorwissen genutzt werden, dass es sich bei $\boldsymbol{\pi}$ um baryzentrische Koordinaten handelt. Sinnvolle Schrittweiten können somit analog zum Iterative Nearest Neighbor (INN) Verfahren [129] ermittelt werden.

Der INN Algorithmus approximiert die synthetisierte Quellenverteilung als gewichtete Kombination der Kandidatenquellen. Timofte et al. entwickelten dieses Verfahren zur Lösung von Problemen im *Sparse Dictionary Learning* [129]. Hier sollen allgemeine Vektoren in euklidischen Räumen als Kombination von wenigen Referenzvektoren eines sogenannten Wörterbuchs kodiert werden. In dieser Arbeit stellen diese Vektoren jedoch Wahrscheinlichkeitsverteilungen in einem abstrahierten hochdimensionalen Raum dar. Die hierin definierte Domänendistanz ist nichteuklidisch. Im Folgenden wird daher eine Variante des INN Algorithmus vorgestellt, welche unter diesen geänderten Bedingungen operieren kann. In jedem Iterationsschritt i aus maximal L Runden wird eine der Quellenkandidaten selektiert und das Rundengewicht w^i zugewiesen. Daher gilt

$$p_S^L(\boldsymbol{x}) = \sum_{i=1}^{L} w^i p_{S_i}(\boldsymbol{x})\,. \tag{4.14}$$

Mit jeder neuen Runde wird die bisherige Summe um einen Term erweitert und die resultierende synthetische Quelle konvergiert so, bezogen auf die verwendete Domänendistanz, zur Zielverteilung $p_T(\boldsymbol{x})$. Die Rundengewichte hängen nicht von den Daten ab und können daher direkt aus i abgeleitet werden:

$$w^i = \frac{\lambda}{(1+\lambda)^i}\,. \tag{4.15}$$

Der Parameter λ ist benutzergesteuert und ermöglicht einen Kompromiss zwischen der Konvergenzgeschwindigkeit und der Dichte des Lösungsvektors $\boldsymbol{\pi}$. Da in jeder Runde nur eine Quelle selektiert werden kann und die Rundengewichte w^i immer positiv sind, muss sich die aktuelle Lösung somit in Richtung dieser Quelle bewegen. In der ursprünglichen Formulierung wurde zur Vereinfachung der Rechnung die Differenz $p_S^L(\boldsymbol{x}) - p_T(\boldsymbol{x})$ betrachtet, welche es zu minimieren galt. Dieser Differenzvektor

kann jedoch negative Wahrscheinlichkeiten enthalten und ist nicht normiert. Er repräsentiert somit keine physikalische Wahrscheinlichkeitsverteilung. Die in dieser Arbeit verwendete Domänendistanz liefert auf solchen Verteilungen allerdings undefinierte Aussagen. Durch Umformulierung der Iterationsvorschrift aus [129] können problematische Zwischenergebnisse allerdings vermieden werden. Die modifizierte äquivalente Vorschrift lautet:

$$S_{L+1} = \min_{S \in \mathbb{S}} d\Big(\mathrm{Tr}_T, \{\boldsymbol{x}_{S,r}, y_{S,r}, \frac{w^{L+1}}{\sum_{j=1}^{L+1} w^j}\}_{r=1}^{N_S} + \bigcup_{i=1}^{L} \{\boldsymbol{x}_{S_i,r}, y_{S_i,r}, \frac{w^i}{\sum_{j=1}^{L+1} w^j}\}_{r=1}^{N_{S_i}} \Big) . \tag{4.16}$$

Es gilt zu beachten, dass die Rundengewichte nicht den gesuchten Quellengewichten entsprechen. Derselbe Quellenkandidat kann mehrfach in unterschiedlichen Runden selektiert werden. Die Quellengewichte ergeben sich daher zu

$$\pi_{S^s} = \sum_{i=1}^{L} w^i \cdot \mathbb{I}(S_i = S^s) . \tag{4.17}$$

Das Verfahren ist in Algorithmus 7 als Pseudocode dargelegt.

Der INN Algorithmus besitzt lediglich einen einstellbaren Parameter $\lambda \in (0,1]$ und konvergiert schnell und zuverlässig unabhängig vom Datensatz. Bei größeren Werten konvergiert das Verfahren sehr schnell zu dünnbesetzten Approximationen der optimalen Lösung. Dieses Verhalten ist bei den betrachteten Datensätzen wünschenswert. Die maximale Anzahl an Iterationen L sollte an der gewünschten Approximationsgüte ausgerichtet sein. Es gilt $\lim_{L\to\infty} \sum_{i=1}^{L} w^i = 1$. Es kann vorzeitig abgebrochen werden, sobald die Summe der Gewichte eine festgelegte Schranke überschreitet und somit der Einfluss der folgenden Iterationen begrenzt werden kann. Wird diese Schranke zu $\beta \leq \sum_{i=1}^{L} w^i$ definiert, dann folgt

$$L = \left\lceil -\frac{\log(1-\beta)}{\log(1+\lambda)} \right\rceil . \tag{4.18}$$

Somit wird bei einer typischen Parametrisierung von $\beta = 0.9$ und $\lambda = 0.5$ eine Konvergenz in nur 6 Iterationsschritten erreicht. Die asymptotische Laufzeit ergibt sich unter Verwendung des MMD Schätzers mit linearer Laufzeit zu $\mathcal{O}(L^3QM)$. Das INN Verfahren zur Lösung des Multiquellenselektionsproblems schlägt somit für kleine Werte von L State-of-the-Art Methoden [23] welche auf *Quadratic Programming* (QP) basieren und daher eine polynomielle Laufzeitkomplexität von $\approx \mathcal{O}(Q^kM)$ besitzen [90]. Der Exponent k der effizienteren QP Verfahren ist im Allgemeinen sehr datenabhängig, liegt aber in der Regel im Bereich $1 \ll k < 3$.

4.4 Tuning der Hyperparameter

Die Wahl der Kernelfunktion $k(.,.)$ stellt einen kritischen Parameter für die Diskriminativität der MMD dar. Aufgrund der Beziehung (4.4) ist es sinnvoll die Auswahl

Algorithmus 7 Konvexe Multiquellenselektion durch Iterative Nearest Neighbors

Eingabe: λ, β, Tr_T, Tr_{S_1}, …, Tr_{S_Q}
Ausgabe: $\boldsymbol{\pi}$

$L \leftarrow \left\lceil -\frac{\log(1-\beta)}{\log(1+\lambda)} \right\rceil$ ▷ Formel (4.18)
$\boldsymbol{\pi} \leftarrow 0$

for $l \in [1, L]$ **do**
  $w^l \leftarrow \frac{\lambda}{(1+\lambda)^l}$ ▷ Formel (4.15)

  for $s \in [1, Q]$ **do**
    $\mathrm{Tr}_{l,s} \leftarrow \{\boldsymbol{x}_{S_s,r}, y_{S_s,r}, \frac{w^l}{\sum_{j=1}^{l} w^j}\}_{r=1}^{N_{S_s}}$
      $+ \bigcup_{i=1}^{l} \{\boldsymbol{x}_{S_i,r}, y_{S_i,r}, \frac{w^i}{\sum_{j=1}^{l} w^j}\}_{r=1}^{N_{S_i}}$ ▷ Formel (4.16)
    $d_{l,s} \leftarrow d(\mathrm{Tr}_T, \mathrm{Tr}_{l,s})$
  end for

  $i \leftarrow \arg\min_{s \in [1,Q]} d_{l,s}$
  $\pi_{S_i} \leftarrow \pi_{S_i} + w^l$
end for

$\boldsymbol{\pi} \leftarrow \frac{\boldsymbol{\pi}}{\|\boldsymbol{\pi}\|}$

auf positive beschränkte Kernel einzugrenzen. Für multivariate Merkmalsräume hat sich der RBF-Kernel durchsetzen können, da er der einzige separierbare radialsymmetrische Kernel ist und starke Glättungseigenschaften besitzt [121]. Die korrekte Einstellung der Kernelbandbreite σ ist jedoch essentiell für die Anwendung von Kernelmethoden. Für die MMD im Speziellen sind hierfür aus der Literatur zwei Methoden bekannt. In der Median-Heuristik wird σ aus dem Median der paarweisen euklidischen Distanzen aller Merkmalsvektoren berechnet [103]. In dem Fall, dass sich die Verwendung der MMD auf einen Hypothesentest reduzieren lässt, der die Aussage „Die Wahrscheinlichkeitsverteilungen $p_T(\boldsymbol{x})$ und $p_S(\boldsymbol{x})$ sind identisch“ widerlegen soll, existiert allerdings ein optimales Kriterium zur Einstellung von σ. In [122] wurde gezeigt, dass die Diskriminativität der MMD durch Wahl des Kernels optimiert wird, welcher die Distanz zwischen den Stichproben maximiert:

$$d^2_{\mathrm{MMD}_{\mathrm{opt}}}(\mathrm{T}_T, \mathrm{T}_S) = \max_{k(.,.)} d^2_{\mathrm{MMD}}(\mathrm{T}_T, \mathrm{T}_S)\,. \tag{4.19}$$

Die Einbettung von $d^2_{\mathrm{MMD}_{\mathrm{opt}}}$ in die Multiquellenselektion aus Formel (4.10) ergibt

$$\hat{S}_{\boldsymbol{\pi}} = \underset{\boldsymbol{\pi}\in\mathbb{R}^Q}{\arg\min} \left[\max_{\sigma_1\in\mathbb{R}^+} \epsilon_{\mathrm{MMD}}(\mathrm{T}_{S_{\boldsymbol{\pi}}}) + \max_{\sigma_2\in\mathbb{R}^+} 2d_{\mathrm{MMD}}(\mathrm{T}_T, \mathrm{T}_{S_{\boldsymbol{\pi}}}) \right] . \tag{4.20}$$

Solche Probleme werden in der Literatur als *MiniMax* Probleme bezeichnet und es existieren bereits einige spezialisierte Optimierungsverfahren [22, 50]. Angelehnt an einer der bekannten Lösungsstrategien betrachten wir zunächst die inneren Optimierungsprobleme gemäß Formel (4.19) als separate Probleme. Im Verlauf dieses Abschnitts wird eine Intervallmethode zur effizienten Suche des gesuchten Maximums für jedes Teilproblem konstruiert. Der *Max* Part des *MiniMax* Problems kann somit als Nebenbedingung interpretiert werden. Zu jedem gegebenen $\boldsymbol{\pi} \in \mathbb{R}^Q$ wird diese Nebenbedingung durch Anwendung der Intervallmethode auf die gültige Lösungsmenge projiziert. Auf diese Weise kann die optimierte MMD aus Formel (4.19) direkt mit dem INN Algorithmus aus Abschnitt 4.3 verwendet werden. Das kombinierte Verfahren entspricht somit einem *Projected Gradient Descent* Ansatz.

Zunächst wird die allgemeine Form des Funktionsverlaufs von (4.19) in Abbildung 4.3 dargestellt. Die generellen Eigenschaften dieses Problems werden im Detail in Appendix A.2 hergeleitet. Hierfür muss vor der Optimierung eine Substitution $x = \sigma^{-2}$ erfolgen. Es wird somit zunächst das Maximum für x gesucht. Es gilt insbesondere, dass die Zielfunktion zwar nicht konvex ist, jedoch nur maximal ein lokales Maximum bei $x_{\max}$ sowie ein lokales Minimum bei $x_{\min}$ besitzen kann. Weiterhin ist die Zielfunktion eindimensional. Zur Lösung bieten sich daher einfache Maximumsuchverfahren basierend auf einer Intervallunterteilung an. Analog zur binären Suche wird bei der ternären Suche ein Startintervall (L, R) durch Auswertung der Zielfunktion an wenigen Stützstellen sukzessive verkleinert. Die Wahl der Stützstellen hat dabei einen Einfluss auf die resultierenden Intervalllängen und somit auf Qualität der Lösung. Es gilt, dass das Intervall möglichst kurz sein sollte und zu jeder Iteration das gesuchte Maximum enthalten muss. Die Methode des goldenen Schnittes [93] gilt hierbei als *worst-case* optimal und wurde daher für diese Arbeit gewählt. Das vollständige Verfahren ist in Algorithmus 8 gegeben. Im Vergleich zu einer Standardimplementierung wird zusätzlich der Fall „$f_B < 0$“ behandelt um die Konvergenz der Suche bei Existenz eines lokalen Minimums zu garantieren [1]. Bei der Anwendung von Intervallunterteilungsverfahren gilt noch zu beachten, dass das Startintervall (L, R) finit ist während der Wertebereich des Kernelparameters $x \in \mathbb{R}^+$ die gesamten nichtnegativen reellen Zahlen umfasst. In Algorithmus 8 wird das Intervall $\mathbb{R}^+ = (0, \infty)$ daher zunächst auf $(0, \pi/2)$ abgebildet. Es gilt

$$x' = tan^{-1}(x) . \tag{4.21}$$

Diese Operation ist bijektiv. Somit kann das Ergebnis nach der Optimierung wieder auf den ursprünglichen Wertebereich zurückgerechnet werden. In der Zwischendarstellung gilt, dass sich die Länge des Ergebnisintervalls nach i Iterationen durch

[1] Das Ausreichen dieser Modifikation folgt aus den Ergebnissen aus Appendix A.2

$|R - L| \leq \frac{\pi}{2}\varphi^{-i}$ begrenzen lässt, wobei $\varphi \approx 1.618$ der Goldenen Zahl entspricht und dem Algorithmus seinen Namen verleiht. Es werden somit bereits nach wenigen Iterationen hohe Genauigkeiten erreicht. Nach der Optimierung ergibt sich σ durch Umkehrung der Substitution:

$$\sigma = \sqrt{x_{\max}^{-1}}. \tag{4.22}$$

Algorithmus 8 Automatische Schätzung der Kernelbandbreite

$\varphi \leftarrow 1.61803398875$
$(L,R) \leftarrow (0,\pi/2)$
$(A,B) \leftarrow (R - (R-L)/\varphi, L + (R-L)/\varphi)$

for $i = 1..\text{MaxIter}$ **do**
  $f_A \leftarrow d^2_{\text{MMD}}(\text{Tr}_T, \text{Tr}_S)$ für $x = \tan(A)$
  $f_B \leftarrow d^2_{\text{MMD}}(\text{Tr}_T, \text{Tr}_S)$ für $x = \tan(B)$

  if $f_B < 0$ **then**
    $L \leftarrow B$
  else if $f_B \leq f_A$ **then**
    $R \leftarrow B$
  else
    $L \leftarrow A$
  end if

  $(A,B) \leftarrow (R - (R-L)/\varphi, L + (R-L)/\varphi)$
end for

return $x_{\max} = \tan((L+R)/2)$

4.5 Erweiterungen des Transfermodells

Die folgenden zwei Ansätze haben zum Ziel, die Robustheit und den Speicherverbrauch des Transfermodells zu verbessern. In der Evaluation in Kapitel 6 konnten diese Methoden nicht in allen Experimenten signifikante Verbesserungen erzielen. Deshalb werden sie in dieser Arbeit nicht als Kern des Transfermodells präsentiert, sondern als optionale Erweiterungen.

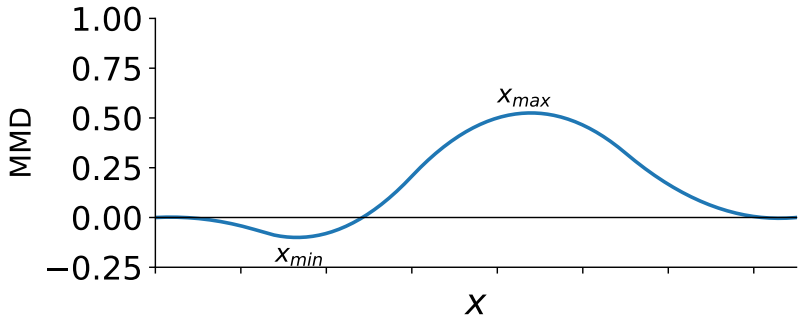


Abbildung 4.3: Allgemeine Form des Maximierungsproblems zur Einstellung des Kernelparameters σ mit der Substitution $x = \sigma^{-2}$. Das gesuchte Maximum liegt bei $x_{\max}$. Weiterhin kann die Funktion ein Minimum bei $x_{\min}$ besitzen.

4.5.1 Robustere Schätzung durch Booststrap Sampling

Als Ergebnis liefert das untersuchte Transfermodell zu den gegebenen Ziel- und Quellenstichproben einen Gewichtungsvektor $\boldsymbol{\pi}$ über die Quellen. Die Berechnung erfolgt durch einen MAP Schätzer und unterliegt somit auch diversen statistischen Einflüssen. Über die geschätzten Quellengewichte liegt somit noch eine Schätzfehlerverteilung mit unbekanntem Mittelwert und Varianz. Die Schätzfehler entstehen in diesem Modell zum Einen aufgrund des Effekts der *Selektionsverzerrung* von Stichproben endlicher Größe. Für kleine Stichprobengrößen steigt u. a. die Wahrscheinlichkeit einer Überrepräsentation von Ausreißern. Zum Anderen kommen bei kleinen Stichproben immer stärker die regularisierenden Glattheitsannahmen zum Tragen, welche vom RBF Kernel impliziert werden und zu einer Überglättung des Modells führen können. Letztendlich wurde zur Reduktion der Berechnungskomplexität mit dem empirischen Schätzer $d^2_{\mathrm{MMD}_1}$ ein Schätzer mit suboptimaler Schätzfehlervarianz gewählt. Das *Bootstrap Sampling* bietet einen modellfreien Ansatz zur Schätzung und Reduktion der Schätzfehlervarianz [41, 62]. Grundlegend gilt, dass die Lernstichproben Tr_T und Tr_S aus einer ansonsten unbekannten Population erzeugt wurden. Falls diese Stichproben bezüglich dieser Population repräsentativ sind, dann gibt es Verfahren der Stichprobenwiederholung (*resampling*) um aus diesen Stichproben neue, ebenfalls repräsentative, Stichproben $\hat{\mathrm{Tr}}_T$ und $\hat{\mathrm{Tr}}_S$ zu erzeugen. Eine Analyse dieser neu erzeugten Stichproben erlaubt einen Rückschluss auf die statistischen Eigenschaften eines Schätzers bezogen auf die Population. Für das *Bootstrap Sampling* werden solche Bootstrap Stichproben $\hat{\mathrm{T}}$ im Allgemeinen durch unabhängiges Ziehen mit Wiederholung aus einer Eingangsstichprobe T erzeugt. Wird das Transfermodell statt auf den ursprünglichen Eingangsdaten nun auf diesen Bootstrap Stichproben angewendet, so müssen die so erzeugten Quellengewichte aus der gesuchten Schätzfehlerverteilung entstammen. Eine wiederholte Anwendung des Transfermodells auf unabhängige Bootstrap Stichproben $\hat{\mathrm{Tr}}_{Ti}$ und $\hat{\mathrm{Tr}}_{Si}$, $i \in [1, B]$, erzeugt somit die Quellengewichte

$\hat{\boldsymbol{\pi}}_i$. In der Bootstrap Aggregierung (BAGGING) wird durch einen Mittelwertschätzer über die $\hat{\boldsymbol{\pi}}_i$ eine Art Metaschätzer mit geringerer Schätzfehlervarianz erzeugt. Es gilt:

$$\hat{\boldsymbol{\pi}} = \frac{1}{B} \sum_{i=1}^{B} \hat{\boldsymbol{\pi}}_i \,. \tag{4.23}$$

4.5.2 Asymmetrische Maximum Mean Discrepancy

Die MMD beschreibt eine symmetrische Domänendistanz für die $d_{\mathrm{MMD}}(\mathrm{T}_T, \mathrm{T}_S) = d_{\mathrm{MMD}}(\mathrm{T}_S, \mathrm{T}_T)$ gilt. Die Rollen der Ziel- und Quellenaufgaben könnten somit vertauscht werden ohne den Wert der berechneten Distanz zu beeinflussen. Aus Abbildung 4.4 ist allerdings zu ersehen, dass diese Beziehung eine fundamental asymmetrische Charakteristik aufweist. Es muss daher nicht zwangsläufig folgen, dass ein positiver Wissenstransfer $S \to T$ auch auf $T \to S$ zutrifft. In diesem Beispiel ist die Ziellernstichprobe eine strikte Teilmenge der Quellenlernstichprobe. Jeder Merkmalsvektor aus Te_T wird von der Quelle durch mindestens ein Lernbeispiel repräsentiert. Somit wird der Generalisierungsfehler in diese Richtung gering ausfallen. Ein auf Tr_T angelernter Klassifikator kann jedoch zu Merkmalsvektoren aus $S \setminus T$ keine zuverlässige Aussagen treffen. Der Generalisierungsfehler ist in diesem theoretischen Beispiel daher sehr viel höher. Durch Ausnutzung dieser asymmetrischen Beziehung soll die MMD in zwei Aspekten verbessert werden. In diesem Abschnitt wird eine asymmetrische MMD (AMMD) entwickelt, welche zum Einen einen verbesserten Transfer in Situationen erreicht, in denen es sich bei einer der Quellen um eine Supermenge der Ziellernstichprobe handelt. Zum anderen wird gezeigt, dass die neu entwickelte AMMD auf einer reduzierten Quellenlernstichproben angewendet werden kann. Gegenüber der symmetrischen MMD wird somit der Speicherverbrauch signifikant reduziert.

Die MMD besteht gemäß Formel (4.3) aus vier Termen, welche in eine Zielkomponente und eine Quellenkomponente zerlegt werden kann:

$$d^2_{\mathrm{MMD}}(\mathrm{T}_T, \mathrm{T}_S) = \underbrace{\mathrm{E}[k(\boldsymbol{x}_T, \boldsymbol{x}'_T)] - \mathrm{E}[k(\boldsymbol{x}_T, \boldsymbol{x}_S)]}_{\text{Ziel}} + \underbrace{\mathrm{E}[k(\boldsymbol{x}_S, \boldsymbol{x}'_S)] - \mathrm{E}[k(\boldsymbol{x}_S, \boldsymbol{x}_T)]}_{\text{Quelle}} \,. \tag{4.24}$$

Aus dem Beispiel aus Abbildung 4.4 wird ersichtlich, dass die Kompaktheit der Quelle, welche durch den dritten Term repräsentiert wird, keinen Einfluss auf die Transferierbarkeit haben sollte. Im Folgenden wird daher nur die Zielkomponente weiter betrachtet. Die Erwartungswerte können in dieser Formel für endliche Stichprobengrößen weiter aufgeteilt werden:

$$\begin{aligned} \tilde{d}^2_{\mathrm{MMD}}(\mathrm{T}_T, \mathrm{T}_S) &= \mathrm{E}[k(\boldsymbol{x}_T, \boldsymbol{x}'_T)] - \mathrm{E}[k(\boldsymbol{x}_T, \boldsymbol{x}_S)] \\ &= \frac{1}{|\mathrm{T}_T|^2} \sum_{i=1}^{|\mathrm{T}_T|} \sum_{j=1}^{|\mathrm{T}_T|} k(\boldsymbol{x}_{T,i}, \boldsymbol{x}'_{T,j}) - \frac{1}{|\mathrm{T}_T||\mathrm{T}_S|} \sum_{i=1}^{|\mathrm{T}_T|} \sum_{j=1}^{|\mathrm{T}_S|} k(\boldsymbol{x}_{T,i}, \boldsymbol{x}_{S,j}) \,. \end{aligned} \tag{4.25}$$

Diese Domänendistanz bestimmt für jeden Merkmalsvektor aus T_T jeweils die mittlere Ähnlichkeit zu Merkmalsvektoren der Ziel- und Quellenverteilung. Wenn die Lernstichproben als Punktwolken interpretiert werden, so entspricht dieses Vorgehen der *Average Linkage* Funktion welche unter anderem in hierarchischen Clustering Algorithmen zur Anwendung kommen [107]. Für das Anlernen eines Klassifikators auf Basis von diskriminativen Entscheidungsmodellen reicht es allerdings aus, wenn zu jedem Merkmalsvektor aus T_T zumindest ein Merkmalsvektor in T_S existiert, welcher in räumlicher Nähe zu diesem liegt. Diese Bedingung ist im Allgemeinen einfacher zu gewährleisten und entspricht einer *Single Linkage* Funktion. Die *Single Linkage* wird in der Kernelformulierung durch die Maximumbildung implementiert. Daher folgt:

$$d^2_{\mathrm{AMMD}}(\mathrm{T}_T, \mathrm{T}_S) = \frac{1}{|\mathrm{T}_T|} \sum_{i=1}^{|\mathrm{T}_T|} \max_{j \in [1,|\mathrm{T}_T|]} k(\boldsymbol{x}_{T,i}, \boldsymbol{x}'_{T,j}) - \frac{1}{|\mathrm{T}_T|} \sum_{i=1}^{|\mathrm{T}_T|} \max_{j \in [1,|\mathrm{T}_S|]} k(\boldsymbol{x}_{T,i}, \boldsymbol{x}_{S,j})\,. \tag{4.26}$$

Eine naive Implementierung der AMMD resultiert in einer Laufzeitkomplexität von $\mathcal{O}(|\mathrm{T}_T|^2 + |\mathrm{T}_T| \cdot |\mathrm{T}_S|)$ und in einem Speicherverbrauch von $\mathcal{O}(|\mathrm{T}_T| + |\mathrm{T}_S|)$. Allerdings ist es im Allgemeinen unnötig, die exakten Maxima zu bestimmen. Die Berechnung kann durch Aufweichung der Suche auf eines der k ähnlichsten Lernbeispiele approximiert werden. Das heißt, wir können die Berechnung von $\max_{j \in [1,|\mathrm{T}_S|]} k(\boldsymbol{x}_{T,i}, \boldsymbol{x}_{S,j})$ vorzeitig abbrechen, sobald ein j gefunden wurde, für das die ausgewertete Kernelfunktion $k(.,.)$ im oberen q-Quantil liegt. Dieses Abbruchkriterium ist allerdings nicht effizient bestimmbar. Wird die Stichprobe T_S jedoch durch eine Bootstrap Stichprobe $\hat{\mathrm{T}}_S$ der Größe $N_{\max}$ ersetzt, dann enthält $\hat{\mathrm{T}}_S$ mit einer Wahrscheinlichkeit $1 - p$ mindestens ein Lernbeispiel im oberen q-Quantil der Verteilung, aus der die ursprüngliche Stichprobe erzeugt wurde. Die Parameter p und q können an die geforderte Approximationsgüte angepasst werden. Daraus ergibt sich die Größe der Bootstrap Stichprobe zu $N_{\max} \geq \log_{1-q}(p)$. Zum Beispiel müssen für $q = 0.05$ und $p = 0.01$ so nur $N_{\max} = 90$ Lernbeispiele betrachtet werden. Diese Aussage wird in Appendix A.3 bewiesen und gilt unabhängig von der ursprünglichen Stichprobengröße oder deren Wahrscheinlichkeitsverteilung. Daraus resultiert für die Bootstrap Approximation der AMMD ein Laufzeitverhalten von $\mathcal{O}(|\mathrm{T}_T| \cdot N_{\max})$ und ein Speicherverbrauch von $\mathcal{O}(|\mathrm{T}_T| + N_{\max})$.

Die Integration der AMMD in das Transfermodell erfolgt durch Ersetzen der Domänendistanz $d^2_{\mathrm{MMD}}(.,.)$ in Formel (4.5) durch die asymmetrische Variante in (4.26). Für die Multiquellenselektion aus Abschnitt 4.2 folgt abschließend eine Definition

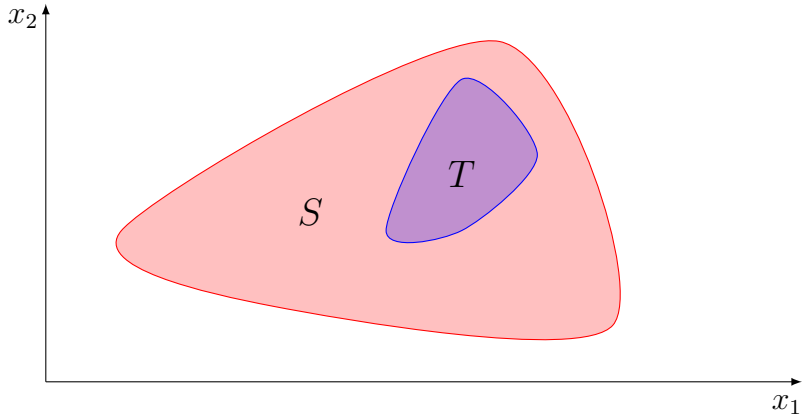


Abbildung 4.4: Darstellung der Merkmalsraumabdeckung für den Fall, dass die Zielstichprobe eine Untermenge der Quellenstichprobe ist. Die symmetrische MMD weist diesem Fall eine hohe Distanz zu, obwohl mit dem auf S gelernten Klassifikator durchaus ein niedriger Generalisierungsfehler auf T zu erwarten ist.

der AMMD für Konvexkombinationen mehrerer Quellen:

$$d^2_{\mathrm{AMMD}}(\mathrm{T}_T, \mathrm{T}_{S_{\boldsymbol{\pi}}}) = \frac{1}{|\mathrm{T}_T|} \sum_{i=1}^{|\mathrm{T}_T|} \max_{j \in [1, |\mathrm{T}_T|]} k(\boldsymbol{x}_{T,i}, \boldsymbol{x}'_{T,j}) - \frac{1}{|\mathrm{T}_T|} \sum_{u=1}^{Q} \pi_{S^u} \sum_{i=1}^{|\mathrm{T}_T|} \max_{j \in [1, |\mathrm{T}_{S^u}|]} k(\boldsymbol{x}_{T,i}, \boldsymbol{x}_{S^u,j}) \,. \tag{4.27}$$

Für die hier vorgestellte asymmetrische Variante gilt indes weder die Konvexität noch kann die theoretische Optimalität der Tuningstrategie des Kernelparameters übertragen werden. Allerdings konnte in empirischen Untersuchungen ein vergleichbares Verhalten zwischen der MMD under AMMD auf realen Datensätzen gezeigt werden.

Kapitel 5

Effiziente Erstellung von Trainingsdaten durch Quellenranking

Die beiden vorangegangen Kapitel beschäftigten sich als Lernszenario mit der Domänenadaption aus einer Menge von Q vollständig gelabelten Quellenkandidaten. Bisher wurde angenommen, dass diese Quellen bereits vorhanden waren und somit kein manueller Aufwand zur Bereitstellung der Lernstichproben erforderlich war. In diesem Kapitel soll ein realitätsnäheres Lernszenario untersucht werden, welches eher mit dem *Multi-Task Lernen* vergleichbar ist. Zunächst soll gelten, dass zu Beginn lediglich E ungelabelte Zielaufgaben vorhanden sind. Die Zielsetzung besteht in diesem Szenario darin, die mittlere Klassifikationsgenauigkeit über alle Zielaufgaben zu maximieren. Weiterhin wurde ein festes Budget eingeplant, welches eine manuelle Segmentierung von exakt Q dieser Zieldomänen ermöglicht. Offensichtlich gilt $Q < E$. Daher müssen die Lernstichproben der $E - Q$ übriggebliebenen Zielpatches durch Anwendung der Quellenselektion aus Kapitel 4 erzeugt werden. Das gesamte Lernszenario wird schematisch in Abbildung 5.1 dargestellt.

Da das Budget limitiert ist, muss anschließend die Frage geklärt werden, welche Teilmenge der Zielpatches als potentiell beste Quellenkandidaten ausgewählt werden sollten. In Kapitel 3 wurde das Konzept des abstrakten Raums $\mathcal{S}$ über alle Domänen eingeführt. Es wurde bereits betont, dass die Domänen in diesem Raum keineswegs gleichverteilt sind und sich somit in Clustern unterschiedlicher Größe und Dichte sammeln. Gute Kandidaten für die Auswahl als Quelle wären somit Bildpatches, welche Nahe dem Zentrum von großen Clustern liegen. Jedoch reicht dies nicht als einziges Kriterium aus, da ansonsten häufig redundante Quellen aus dem größten Cluster ausgewählt werden. Vielmehr müssen die Quellen *informativ* sein. Die Informativität einer Quelle setzt sich abstrakt aus zwei Eigenschaften zusammen:

Zentralität: Eine zentrale Quelle liegt in der Nähe von vielen Quellen bezüglich einer gewählten Domänendistanz.

Ungewöhnlichkeit: Eine ungewöhnliche Quelle besitzt eine große Distanz zu anderen bereits gewählten Quellen.

Viele unüberwachte Clustering Algorithmen arbeiten bereits implizit nach ähnlichen Prinzipien. Für den K-Means Algorithmus konnte sogar direkt gezeigt werden, dass dessen Zielfunktion auf Basis der Zentralität und Ungewöhnlichkeit stochastisch approximiert werden kann [6]. Für diese Arbeit ist jedoch das sogenannte *Kernel Herding* von Chen et al. [25] von besonderem Interesse. Das *Kernel Herding* bietet eine direkte Implementierung des Prinzips der Informativität. Statt einer Distanz kann jedoch jede beliebige gültige Kernelfunktion verwendet werden. Das Verfahren ist also vergleichsweise unkompliziert und kann daher in wenigen Zeilen Pseudo-Code zusammengefasst werden (Algorithmus 9).

Die Kernelfunktion $k(.,.)$ muss für die gegebene Aufgabenstellung eine paarweise Ähnlichkeit zwischen Domänen beschreiben. Weiterhin dürfen nur die ungelabelten Lernstichproben verwendet werden. Die bisher verwendete Domänendistanz kann somit keine Basis zur Definition des Kernels sein. Aus der Formel (4.5) wird daher

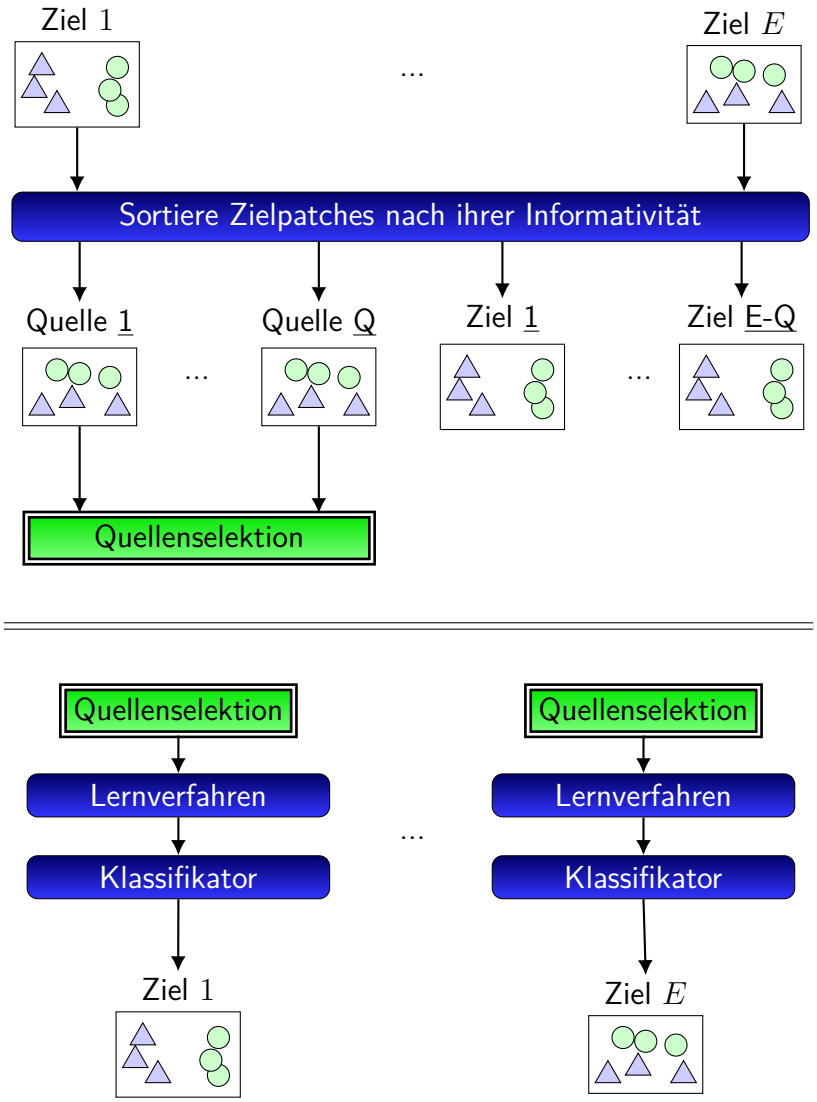


Abbildung 5.1: Lernszenario zur Klassifikation von E Zielpatches bei gleichzeitiger Minimierung der zu labelnden Lernstichproben. Die Bildpatches werden zunächst durch einen Clustering Ansatz anhand ihrer Informativität sortiert. Die Q informativsten Patches gelten dabei als gute Quellenkandidaten. Nachdem diese selektierten Quellen manuell segmentiert wurden, können sie als Quellenkandidaten für eine domänenadaptive Quellenselektion zur Klassifikation der übrigen Zielpatches verwendet werden.

Algorithmus 9 Quellenranking durch Kernel Herding

Eingabe: $Q, k(.,.), T_1, \ldots, T_E$
Ausgabe: $S_1, \ldots, S_Q$

for $i \in [1, Q]$ **do**

$$S_i \leftarrow \underset{S \in \{T_1,\ldots,T_E\} \setminus \{S_1,\ldots,S_{i-1}\}}{\arg\max} \frac{1}{E} \sum_{T \in \{T_1,\ldots,T_E\}} k(S,T) - \frac{1}{i+1} \sum_{S' \in \{S_1,\ldots,S_{i-1}\}} k(S,S')$$

end for

zunächst eine unüberwachte Domänendistanz abgeleitet. Durch Wegstreichen aller Terme, zu deren Berechnung Klassenlabels benötigt werden, folgt:

$$d_{\text{unüberwacht}}(\mathrm{T}_{T_1}, \mathrm{T}_{T_2}) = d_{\text{MMD}}(\mathrm{T}_{T_1}, \mathrm{T}_{T_2})\,. \tag{5.1}$$

Sowohl das statistische Quellenselektionsmodell aus Kapitel 3 als auch das approximative Modell aus Kapitel 4 können mit wenig Anpassungen mit der unüberwachten Domänendistanz (5.1) betrieben werden. Wir definieren den Kernel nun wie folgt:

$$\begin{aligned} k(T_i, T_i) &= 1 \\ k(T_i, T_j) &= \pi_i^j \quad \text{f.a. } i \neq j\,. \end{aligned} \tag{5.2}$$

Die Werte $\boldsymbol{\pi}^j = \{\pi_i^j\}_{\substack{i=1 \\ i \neq j}}^{E}$ bezeichnen die Quellengewichte des Zielpatches T_j, falls als Quellenkandidaten alle anderen Patches $\{T_1, \ldots, T_E\} \setminus T_j$ verwendet wurden. Konkret beschreibt π_i^j dann dass Gewicht der Quelle T_i. Im Prinzip können zur Schätzung der Quellengewichte alle in dieser Arbeit vorgestellten Quellenselektionsverfahren verwendet werden, solange sie mit der unüberwachten Domänendistanz betrieben werden.

Die Quellengewichte bieten eine intuitive Definition eines Ähnlichkeitsmaßes zwischen Domänen. Eine hohe Ähnlichkeit zwischen Domänen sollte erwartungsgemäß dazu führen, dass diese untereinander mit einer hohen Wahrscheinlichkeit als Quellen selektiert werden. Weiterhin wurde die Kernelfunktion (5.2) so definiert, dass solche Domänen als besonders informativ bewertet werden, welche von sehr vielen anderen Domänen als Quelle selektiert wurden. Dadurch entsteht eine Asymmetrie, wodurch $k(.,.)$ streng genommen den *Satz von Mercer* [112] verletzt und somit keinen gültigen Hilbertraum mehr repräsentiert. Viele Kernelmethoden sind jedoch robust gegenüber leichten Verletzungen dieser Voraussetzung. Die Evaluation des hier entwickelten Verfahrens zur Sortierung der Quellen anhand ihrer Informativität erfolgt in Abschnitt 6.4. In den Experimenten konnte empirisch gezeigt werden, dass die *Kernel Herding* Methode ebenfalls zu diesen robusten Verfahren gehört.

Kapitel 6

Evaluation

6.1 Datengrundlage

6.1.1 Bildmaterial

Um eine Untersuchung der entwickelten Transferlernverfahren in einem realistischen Kontext zu ermöglichen werden mehrere kompatible Datensätze benötigt, zwischen denen ein Wissenstransfer plausibel erwartet werden kann. Für die folgenden Untersuchungen wurden fünf Datensätze aus deutschen Städten gewählt, welche typische urbane Szenen enthalten.

Die Datensätze *Vaihingen* und *Potsdam* sind im Rahmen der *2D Semantic Labelling Challenge* der ISPRS WG II/4 entstanden [141]. Der *Vaihingen* Datensatz besteht hierbei aus 33 separaten Bildpatches, von denen allerdings nur für eine Untermenge von 16 Patches die Grundwahrheit zur Verfügung gestellt wurde. Für den *Potsdam* Datensatz gilt, dass von 23 aus 38 Patches die Grundwahrheit bekannt ist. Patches, von denen die Grundwahrheit nicht zur Verfügung stand, wurden aus allen folgenden Experimenten ausgeschlossen. Zu allen Patches steht weiterhin ein digitales Oberflächenmodell (DOM) zur Verfügung.

Die Datensätze *Buxtehude*, *Hannover* und *Nienburg* basieren auf Luftbildern, welche vom Landesamt für Geoinformation und Landesvermessung Niedersachsen (LGLN) bezogen wurden [1]. Die Grundwahrheiten wurden hierbei im Rahmen dieser Arbeit manuell erstellt. Jeder Datensatz besteht aus einem Orthofoto, welches jeweils ein $2 \times 2\,\mathrm{km}^2$ großes Areal abbildet. Diese Luftbilder wurden weiterhin in 3×3 Patches gleicher Größe unterteilt. Auch für diese Datensätze wurde bereits ein DOM zur Verfügung gestellt.

Die Eigenschaften aller Datensätze sind in Tabelle 6.1 zusammengefasst.

In den folgenden Untersuchungen wird außerdem ein synthetischer Datensatz betrachtet, welcher im Folgenden als *Combined* bezeichnet wird. Dieser Datensatz enthält alle Patches der fünf separaten Datensätze, was einer Gesamtgröße von insgesamt 66 Patches entspricht. Dieser zusätzliche Datensatz soll den realistischen Fall simulieren, in dem innerhalb der verfügbaren Quellenkandidaten nur sehr wenige einen sinnvollen Transfer ermöglichen. Weiterhin wird zu diesem *Combined* Datensatz eine reduzierte Variante betrachtet, der *Small* Datensatz. Der *Small* Datensatz enthält aus jedem separaten Datensatz lediglich drei willkürlich gewählte Patches, somit also insgesamt 15 Stück. Dieser Datensatz enthält deutlich weniger Daten und soll somit eine Evaluation des rechenaufwändigen statistischen Quellenselektionsmodells ermöglichen.

[1] © 2013 LGLN

Tabelle 6.1: Eigenschaften der verwendeten Datensätze.

Datensatz	Farbkanäle	DOM	Bodenauflösung	Patches	# Objektklassen
Vaihingen	R-G-IR	✓	9 cm	16	6
Potsdam	R-G-B-IR	✓	8 cm	23	6
Buxtehude	R-G-B-IR	✓	20 cm	9	10
Hannover	R-G-B-IR	✓	20 cm	9	10
Nienburg	R-G-B-IR	✓	20 cm	9	10

(a) Vaihingen (b) Potsdam (c) Buxtehude (d) Hannover (e) Nienburg

Abbildung 6.1: Ausschnitte aus den verwendeten Datensätzen.

6.1.2 Objektartenkatalog

Die Grundwahrheiten der fünf vorgestellten Datensätze wurden unter Berücksichtigung ihrer jeweiligen Objektartenkataloge erstellt und enthalten somit zunächst unterschiedliche Objektklassen. Diese Differenzen resultieren im Wesentlichen durch unterschiedlich tiefe Klassenhierarchien. So wird die Klasse *Niedrige Vegetation* in manchen Datensätzen nach ihrer Landnutzung weiter aufgegliedert. Eine wichtige Annahme der Domänenadaption besagt jedoch, dass die Menge der Klassenlabels $\mathcal{K}$ in allen Lernaufgaben identisch sein muss. Für alle folgenden Experimente werden daher die Klassenlabels der einzelnen Datensätze zunächst auf eine gemeinsame Obermenge abgebildet. Eine sinnvolle Abbildung sollte allerdings einige wichtige Eigenschaften aufweisen:

Vollständig: Jedem Pixel eines Patches wird genau eine Klasse zugeordnet.

Detailliert: Es sollen ausreichen viele Klassen unterschieden werden um eine semantische Interpretation der Szene zu erlauben.

Lernbar: Die gewählten Klassen müssen in den Bilddaten, oder in einer daraus abgeleiteten Merkmalsrepräsentation, unterscheidbar sein.

Ubiquitär: In den meisten Patches sollen von jeder Klasse ausreichend viele Lernbeispiele für eine statistische Evaluation vorhanden sein. Somit sollten Klassen vermieden werden, welche nur in wenigen Patches vorkommen.

Nach diesen Kriterien wurden die vier Klassen *Gebäude*, *Baum*, *Niedrige Vegetation* und *Versiegelte Oberfläche + Sonstige* selektiert. Die Teilklasse *Sonstige* beinhaltet unter anderem Fahrzeuge sowie kleine alleinstehende Objekte und wurde der Klasse *Versiegelte Oberfläche* aufgrund ihrer ähnlichen radiometrischen Charakteristik zugeordnet.

6.1.3 Merkmalsraum

Die in dieser Arbeit entwickelten Methoden sind nichtparametrisch und setzen keine besonderen Voraussetzungen an den gewählten Merkmalsraum. Diese allgemeine Anwendbarkeit soll anhand von Experimenten auf unterschiedlichen Merkmalsräumen belegt werden. Im allgemeinen existieren drei Möglichkeiten zur Konstruktion von Merkmalsräumen. Zum einen können Merkmale durch einen Experten an die jeweilige Lernaufgabe angepasst, selektiert und optimiert werden. Zum anderen können auch *generische* Merkmale, welche bereits in anderen Lernaufgaben gute Resultate zeigten, gewählt werden. Bei steigender Anzahl an generischen Merkmalen steigt die Wahrscheinlichkeit, dass zumindest einige von ihnen bezüglich der zu trennenden Klassen disriminativ sind. Jedoch müssen die irrelevanten Merkmale durch das Lernverfahren zuverlässig erkannt werden, was zu einem steigenden Bedarf an gelabelten Daten führt. Die dritte Möglichkeit besteht darin, einen an die Lernstichprobe optimierten Merkmalsraum zu lernen. Dieser Weg wird insbesondere von tiefen neuronalen Netzwerken verfolgt [111]. Von diesen Möglichkeiten sollen lediglich die ersten beiden untersucht werden, da das Gebiet der Selektion und Synthese von Merkmalen sehr komplex und weitreichend ist und somit den Rahmen dieser Arbeit sprengen würde.

Der expertenoptimierte Merkmalsraum (EXP) richtet sich an die zu trennenden Objektklassen. Hier wurde die Strategie verfolgt, die vier Klassen durch geschickt gewählte Merkmale in jeweils zwei Teilmengen zu zerlegen. Dieses Vorgehen wird in Abbildung 6.2 dargestellt. Die Klassen *Gebäude* und *Baum* können von den anderen beiden Klassen anhand ihrer relativen Höhe zur Erdoberfläche unterschieden werden. Daher bietet sich die Nutzung des digitalen Oberflächenmodells an. Allerdings sind im DOM nur die absoluten Höhenwerte verzeichnet und beinhalten somit den Verlauf des Terrains, welcher für eine Klassifikation der untersuchten Objektklassen keine Rolle spielt (ein Gebäude auf einer Anhöhe sollte nicht anders klassifiziert werden als ein Gebäude in einem Tal). In dieser Arbeit wird das zumeist tieffrequente digitale Geländemodell (DGM) aus dem DOM durch Anwendung eines robusten Tiefpassfilters aus [134] herausgerechnet. Der Glättungsfaktor λ dieses Tiefpassfilters muss in Abhängigkeit von der Bodenauflösung eingestellt werden. Für diese Arbeit gilt $\lambda_{\text{vai}} = 10^{-4}$ und $\lambda_{\text{bux}} = \lambda_{\text{han}} = \lambda_{\text{nin}} = 10^{-6}$. Das DOM wird anschließend durch Subtraktion des so geschätzten DGM normiert. Das DOM des Potsdam Datensatzes wurde bereits um das DGM bereinigt, weshalb hier dieser Verarbeitungsschritt entfallen kann.

Die Klassen *Baum* und *Niedrige Vegetation* können von den anderen Klassen anhand der vorhandenen Chlorophyllkonzentration, welches eine wesentliche Aufgabe in

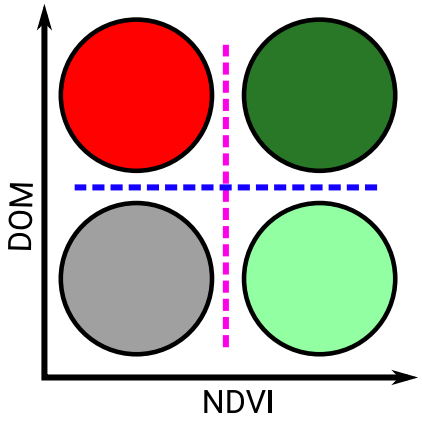


Abbildung 6.2: Aufbau des durch Expertenwissen konstruierten Merkmalsraums. Die Klassen *Gebäude* (rot) / *Versiegelte Oberfläche* (grau) können anhand des NDVI von den Klassen *Baum* (dunkelgrün) / *Niedrige Vegetation* (hellgrün) unterschieden werden (violette Linie). Die Klassen *Gebäude* / *Baum* und *Versiegelte Oberfläche* / *Niedrige Vegetation* sind durch ihre Höhenwerte im DOM unterscheidbar (blaue Linie).

der Photosynthese von Pflanzen erfüllt, unterschieden werden. Chlorophyll absorbiert insbesondere rote und blaue Wellenlängen, weshalb gesunde Pflanzen in Luftbildern einen hohen Grünanteil besitzen. Weiterhin sorgt die spezielle Zellstruktur gesunder Pflanzen für eine erhöhte Reflektion von Strahlung im infraroten Bereich. Ein hoher IR Anteil kann allerdings im nahen Infrarot auch durch Diffusionseffekte von roten Objekten verursacht werden. Um dennoch Pflanzenaktivität zuverlässig erkennen zu können wurden in der Fernerkundung daher unterschiedliche Vegetationsindices untersucht. Der bekannteste hiervon ist der Normalized-Difference-Vegetation-Index (NDVI) [109]:

$$NDVI = \frac{IR - R}{IR + R}\,. \tag{6.1}$$

Ein NDVI nahe bei 1 steht hierbei für eine hohe Pflanzenaktivität, während Werte um die 0 auf keine oder geringe Pflanzenaktivität hinweist. Negative Werte weise u. a. auf Wasserflächen hin. Durch die Kombination der beiden Merkmale DOM und NDVI können somit alle vier Klassen zuverlässig voneinander getrennt werden. Dieser Merkmalsraum wird im Folgenden durch den Bezeichner EXP betitelt.

Als generische Merkmale werden in dieser Arbeit sowohl Farb- als auch Texturmerkmale zur Anwendung kommen. Als Farbmerkmale (COL) gelten hier die pixelweisen Intensitätswerte im visuellen Spektrum sowie dem nahen Infrarot. Da im Vaihingen Datensatz der Blauanteil nicht verfügbar ist, werden alle Versuche lediglich auf die Kanäle Rot, Grün und Nahes Infrarot zugreifen. Um den Einfluss von Rauschen zu

Tabelle 6.2: Klassifikationsgenauigkeiten (OA) für alle untersuchten Datensätze und Merkmalsräume. Die Lern- und Teststichproben wurden durch Bootstrap Sampling erzeugt. Die Messungen wurden über 10 Runden gemittelt. Als Klassifikator kam eine logistische Regression zum Einsatz.

	EXP	COL	COL + LBP	ALL
Small	72.8 %	59.5 %	62.2 %	76.2 %
Vaihingen	77.5 %	70.8 %	74.7 %	81.8 %
Potsdam	75.8 %	61.0 %	64.5 %	78.1 %
Buxtehude	79.5 %	65.9 %	69.3 %	81.1 %
Hannover	70.0 %	59.8 %	65.9 %	78.6 %
Nienburg	76.0 %	62.4 %	65.0 %	80.2 %
Combined	76.1 %	64.1 %	67.8 %	79.8 %

unterdrücken, wurden die Bilder für die Farbmerkmale mit einem Boxfilter der Größe 11×11 geglättet. In der Literatur sind eine Vielzahl von Merkmalen bekannt, welche eine statistische Beschreibung der Textur in einer gefensterten lokalen Umgebung zum Ziel haben. Stellvertretend für diese Texturmerkmale wird hier eine rotationsinvariante Variation der Local-Binary-Patterns (LBP) untersucht [91]. In LBP wird die lokale Umgebung jedes Pixel nach einer lokal-adaptiven Binarisierung durch einen binären Hashwert fester Länge beschrieben. Der Deskriptor ergibt sich für ein gegebenes Bildfenster durch Berechnung des Histogramms über diese Hashwerte über die gesamte Region. In Ermangelung des Blauanteils kann der Texturdeskriptor nicht, wie sonst üblich, auf dem Graufstufenbild berechnet werden. Die LBPs werden in dieser Arbeit daher alternativ auf dem Grünkanal berechnet.

In Tabelle 6.2 sind die erreichbaren Klassifikationsgenauigkeiten (engl. *Overall Accuracy*, OA) nach Formel (2.32) für jeden Datensatz für die untersuchten Merkmalsräume aufgeführt. Der Merkmalsraum ALL ergibt sich durch die gemeinsame Verwendung der drei Merkmale EXP, COL und LBP.

6.2 Statistisches Modell

Zunächst soll hier das statistische Modell aus Kapitel 3 untersucht werden. Der experimentelle Aufbau sowie die verwendeten Metriken werden im folgenden Abschnitt eingeführt. Anschließend folgt eine Betrachtung der Konvergenzeigenschaften des MCMC Simulationsverfahrens sowie eine Diskussion der erzielten Ergebnisse bezüglich einer Bayesschen Einzelquellenselektion.

6.2.1 Experimenteller Aufbau

Die Evaluation des statistischen Modells erfolgt ausschließlich auf dem *Small* Datensatz. Hierfür müssen aus den Bildpatches jedoch noch geeignete Lern- und Teststichproben extrahiert werden. Zu jedem Patch werden zunächst die vier Merkmalsrepräsentationen EXP, COL, COL + LBP und ALL in voller Bildauflösung berechnet. Anschließend werden die Patches in jeder Richtung um den Faktor 25 unterabgetastet. Dieses Vorgehen garantiert zwischen den Lernbeispielen einen räumlichen Mindestabstand und reduziert somit unerwünschte Korrelationen in den Daten. Solche Korrelationen könnten ansonsten aufgrund von nicht vermeidbaren Stichprobenverzerrungen zu einer allzu optimistischen Einschätzung der erreichten Resultate führen. Die Ergebnisse werden immer über 10 Runden gemittelt. In jeder Runde werden aus den unterabgetasteten Daten unabhängige Lern- und Teststichproben durch ein Bootstrap Sampling erzeugt. Hierbei werden 100 zufällige Lernbeispiele pro Bildpatch und Klasse mit Zurücklegen gezogen. Die Größe der Stichproben musste hier aufgrund der quadratischen Laufzeitkomplexität des ADS Samplers aus Abschnitt 3.2.3 sehr klein gewählt werden.

Die Quellenselektion wird für jeden Bildpatch des Datensatzes für alle untersuchten Verfahren zunächst separat durchgeführt. Die übrigen Patches bilden hierbei die Quellenkandidaten. Die Quellenkandidaten gelten in diesem Experiment als vollständig gelabelt, während aus dem Zielpatch nur die ungelabelten Daten verwendet werden. Zur Beurteilung der Verfahren werden unterschiedliche Metriken verglichen. Da das theoretische Modell auf den 0-1 Kosten beruht, bietet sich hier aufgrund dessen Beziehung zum Trainingsfehler insbesondere die Genauigkeit (*acc*) aus Formel (2.32) als Grundlage an. Die im Folgenden benötigten Genauigkeiten wurden für diese Experimente durch eine logistische Regression berechnet. Es gilt zunächst:

$$\text{Ziel OA} = \frac{1}{N_T} \sum_{i=1}^{N_T} \mathbb{I}(h_T(\boldsymbol{x}_{T,i}) = y_{T,i}) \tag{6.2}$$

$$\text{Einzelquellen OA}_s = \frac{1}{N_T} \sum_{i=1}^{N_T} \mathbb{I}(h_s(\boldsymbol{x}_{T,i}) = y_{T,i}) \tag{6.3}$$

$$\text{Quellen OA} = \max_{s \in S} \text{Einzelquellen OA}_s \tag{6.4}$$

$$\text{Selektions OA} = \frac{1}{N_T} \sum_{i=1}^{N_T} \mathbb{I}(h_{\hat{S}}(\boldsymbol{x}_{T,i}) = y_{T,i})\,. \tag{6.5}$$

Hier wird jeweils die Genauigkeit auf der Teststichprobe des Zielpatches berechnet. Die vier Formeln unterscheiden sich darin, auf welchen Daten der Klassifikator angelernt wurde. Der Reihe nach gelten die Formeln für einen Klassifikator, welcher respektive auf der Lernstichprobe des Zielpatches, dem Quellenkandidaten s, dem optimalen Quellenpatch oder auf dem selektierten Quellenpatch $\bar{S}$ trainiert wurde. Der Wert Selektions OA ist aus diesen Vieren von wesentlichem Interesse. Da jedoch

laut Tabelle 6.2 die einzelnen Merkmalsräume unterschiedliche Klassifikationsgenauigkeiten aufweisen, ließen sich die erzielten Resultate nicht über alle Experimente, Datensätze und Merkmalsräume direkt vergleichen. Daher werden im Folgenden erweiterte Metriken eingeführt:

$$\text{Ziel } \Delta\text{OA} = \text{Ziel OA} - \text{Selektions OA} \tag{6.6}$$

$$\text{Quellen } \Delta\text{OA} = \text{Quellen OA} - \text{Selektions OA} \tag{6.7}$$

$$\text{Perzentilrang} = \frac{1}{Q} \sum_{s \in S} \mathbb{I}(\text{Einzelquellen OA}_s < \text{Selektions OA}) \,. \tag{6.8}$$

Die Metrik Ziel ΔOA berechnet den Verlust in Genauigkeit einer Quellenselektion im Vergleich zu einem klassischen überwachten Lernverfahren mit einer vollständig gelabelten Ziellernstichprobe. Dieser Verlust sollte so gering wie möglich ausfallen, kann jedoch im Allgemeinen nicht unter 0 fallen. Die Metrik Quellen ΔOA berechnet den Verlust in Genauigkeit einer Quellenselektion im Vergleich zu einer optimalen Einzelquellenselektion. Diese Metrik ist insbesondere dafür geeignet um zu zeigen, wie viel Verlust durch das Fehlen der Klassenlabels im Zielpatch notwendigerweise in Kauf genommen werden muss. Der Vergleich mit einer optimalen Multiquellenselektion konnte aus Laufzeitgründen nicht herangezogen werden. Als drittes zeigt der Perzentilrang, wie viele Quellen unter den Quellenkandidaten eine vergleichbare oder schlechtere Genauigkeit als die selektierte Quelle erreichen. Ein Perzentilrang von 50% entspricht der Leistung einer zufälligen Einzelquellenselektion.

Das statistische Quellenselektionsmodell wurde bezüglich einer Einzelquellenselektion (*Single*) gemäß Formel (3.9) evaluiert:

$$\hat{S}_{\text{SDS}} = \arg\min_{s \in S} \mathrm{E}[\epsilon(h_s(\boldsymbol{x}), \mathrm{T}_s)|\mathrm{T}_s] - 4\,\mathrm{E}[\epsilon(h_{T \perp s}(\boldsymbol{x}), \mathrm{T}_{T \perp s})|\mathrm{T}_T, \mathrm{T}_s]$$

Das vollständige Modell besteht aus zwei Erwartungswerten und erfordert somit die Simulation von zwei Markov-Ketten pro Quellenkandidaten. Da dieses Modell die Klassenlabels der Quellenpatches benötigt wird dieses im Folgenden als *überwachte Domänenselektion* (engl. *Supervised Domain Selection*, SDS) bezeichnet. Um zu untersuchen, welchen Einfluss die benötigten Klassenlabels auf die Quellenselektion haben, wird zusätzlich ein reduziertes Modell evaluiert, welches nur den zweiten Term enthält. Somit gilt:

$$\hat{S}_{\text{UDS}} = \arg\min_{s \in S} - \mathrm{E}[\epsilon(h_{T \perp s}(\boldsymbol{x}), \mathrm{T}_{T \perp s})|\mathrm{T}_T, \mathrm{T}_s] \tag{6.9}$$

Dieses Modell benötigt keinerlei Klassenlabels und wird daher im Kontrast zum vorherigen Modell als *unüberwachte Domänenselektion* (engl. *Unsupervised Domain Selection*, UDS) bezeichnet. Um die Resultate besser einstufen zu können werden zwei Vergleichsverfahren aufgeführt, welche den Vorgehen in klassischen Klassifikationsframeworks ohne Nutzung von Transferlernen entsprechen. Die Methode *Zufällige Quelle*

selektiert aus den Q Quellenkandidaten eine zufällige Quelle gemäß einer gleichförmigen Wahrscheinlichkeitsverteilung. Die Methode *Alle Quellen* hingegen nutzt alle Quellenkandidaten indem die Vereinigung der Lernstichproben aller Quellenpatches gebildet wird.

Das statistische Klassifikationsmodell aus Kapitel 3 wurde für ein lineares Partitionierungsmodell entwickelt. Die Anwendung des Kernel Tricks 2.2.4 erlaubt jedoch auch die Behandlung von komplexeren Entscheidungsmodellen, was oft für eine Vermeidung von Unteranpassung notwendig ist. Hierbei werden alle Merkmalsvektoren $\boldsymbol{x}$ gemäß dem *Representer-Theorem* durch neue Vektoren $\boldsymbol{x}_{\mathcal{H}}$ ersetzt, welche sich aus den Resultaten der Kernelfunktion mit den Merkmalsvektoren der Lernstichprobe Tr zusammensetzt. Die transformierten Merkmalsvektoren besitzen somit die Länge $|\mathrm{Tr}|$. Der MCMC Algorithmus aus Kapitel 2.4 kann ansonsten unverändert auf einer so transformierten Stichprobe angewendet werden. Für diese Experimente wurde ein RBF Kernel gewählt. Der hierin enthaltene Kernelparameter σ wurde durch eine Kreuzvalidierung auf einer vergleichbaren SVM eingestellt [20]. Die SVM und das statistische Klassifikationsmodell aus Kapitel 3 unterscheiden sich zwar in ihren Kostenfunktionen und dem Regularisierungsansatz, die Kernelbandbreite σ hängt jedoch im wesentlichen von den Eigenschaften des Merkmalsraums ab. Methoden zur Kerneloptimierung sind daher zwischen kernelisierten Klassifikationsmodellen übertragbar.

6.2.2 Konvergenzdiagnose

Bevor die Ergebnisse der Einzelquellenselektion untersucht werden, muss zunächst die Konvergenz des statistischen Klassifikationsmodells sichergestellt werden. Die Konvergenzdiagnose erfolgt nach der Methode von Gelman & Rubin aus Abschnitt 2.4.2. Eine Markov-Kette gilt als konvergiert, sobald die daraus abgeleitete Metrik $\hat{R}$ die Schranke 1.05 unterschritten hat. Diese Methode ist zunächst nur für eindimensionale Parameterräume gültig. In der Literatur werden die Parameter in multivariaten MCMC Simulationen daher häufig separat betrachtet. Die Markov-Kette gilt dann als konvergiert, sobald alle einzelnen Parameter nach Gelman & Rubin konvergieren. Daraus folgt die Konvergenzmetrik

$$\mathrm{Max}\ \hat{R} = \max_{\varphi \in \mathcal{P}} \hat{R}_{\varphi}\,. \tag{6.10}$$

In Abbildung 6.3 sind die Ergebnisse der Konvergenzdiagnose für alle Markov-Ketten der überwachten Einzelquellenselektion (SDS *Single*) für alle Merkmalsräume aufgetragen. Die Metrik Max $\hat{R}$ wird hier über die Länge der simulierten Markov-Kette aufgetragen. Die blaue Linie folgt dem Median über alle Ketten während der hell-blaue Bereich den Interquartilsabstand, also den Bereich zwischen dem 25% und dem 75% Perzentil, aufträgt. Die Hälfte aller Simulationen konvergieren somit bereits nach ca. 150 Iterationen und in über 75% aller Fälle konvergiert die

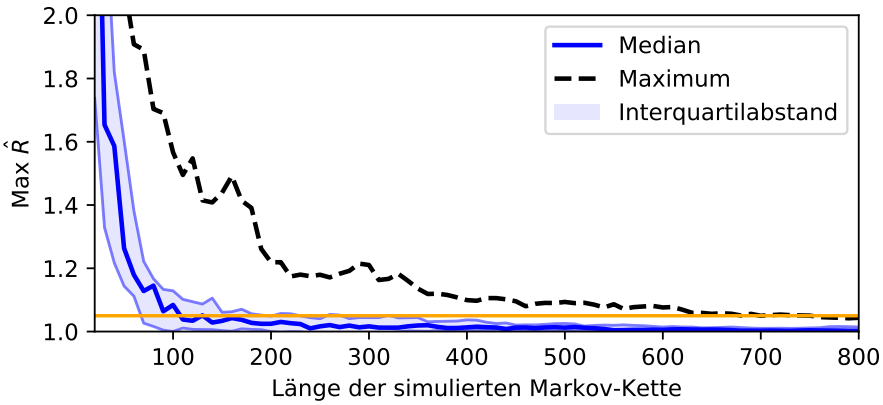


Abbildung 6.3: Konvergenzdiagnose des ADS Verfahrens für das statistische Quellenselektionsverfahren nach Gelman & Rubin. Der Konvergenzschwellwert wird durch die gelbe Linie markiert.

Markov-Kette nach weniger als 200 Iterationen. Diese Konvergenzgeschwindigkeiten sind für ein MCMC Verfahren vergleichsweise kurz; mehrere tausend oder sogar millionen Iterationen sind in anderen Problemstellungen nicht unüblich [15, 124]. Einige Ausreißer, gekennzeichnet durch die schwarze gestrichelte Linie, benötigen dennoch bis zum Vierfachen dieser Kettenlängen zur vollständigen Konvergenz. In Einzelfällen konnte beobachtet werden, dass solche Ausreißer sehr lange in einem lokalen Optimum hängen blieben. Nach dem Verlassen dieses Optimums konnten diese Simulationen jedoch rasant konvergieren. Obwohl dem hier präsentierten ADS basierten MCMC Verfahren im Mittel sehr gute Eigenschaften nachgewiesen werden konnten, muss trotzdem in Einzelfällen zur Vorsicht geraten werden.

6.2.3 Diskussion

In Tabelle 6.3 sind die Ergebnisse der Einzelquellenselektion für das statistische Quellenselektionsmodells auf dem *Small* Datensatz zusammengefasst. Die Resultate werden in dieser Darstellung jeweils über 10 Versuchswiederholungen und über alle Zielpatches des Datensatzes mit dem arithmetischen Mittel gemittelt. Die Bayessche Einzelquellenselektion ist den Referenzverfahren in allen untersuchten Merkmalsräumen überlegen und kann den mittleren Ziel ΔOA konsistent auf $5-6\%$ begrenzen. Dies würde für den Merkmalsraum ALL z. B. eine Reduktion der mittleren Klassifikationsgenauigkeit von 76,2% auf 71,4% bedeuten. Die Betrachtung des Prozentilrangs zeigt weiterhin, dass das statistische Modell in diesem Fall Quellen selektiert, welche im Mittel im oberen 4% Perzentil der Quellenkandidaten liegen. Für ca. 50% der Zielpatches konnte sogar die optimale Quelle gefunden werden. Die hier entwickelte Bayessche Methodik zur Quellenselektion ist somit sicher in der Lage, sinnvolle Lern-

stichproben nur anhand der Domänendistanz zu selektieren. Die Ergebnisse auf dem *Small* Datensatz konnten die Referenzverfahren deutlich übertreffen. Die Ergebnisse der Methoden SDS und SDU scheinen hingegen vergleichbar, wobei die unüberwachte Variante einen kleinen Vorsprung zu haben scheint. Eine diffizilere Analyse kann aufgrund der hier untersuchten Lernstichprobengrößen nicht erfolgen. Im nächsten Abschnitt folgt daher eine Evaluation des approximativen Quellenselektionsmodells, welches eine Untersuchung von größeren Datensätzen sowie eine Multiquellenselektion ermöglicht. Die Untersuchung der statistischen Signifikanz der Unterschiede zwischen den einzelnen Verfahren soll daher auf die nächsten Experimente verschoben werden.

Tabelle 6.3: Statistisches Modell: Ergebnisse für den *Small* Datensatz.

	EXP			COL			COL + LBP			ALL		
	∅ Ziel Δ_{OA} [%]	∅ Quellen Δ_{OA} [%]	∅ Prozentrang [%]	∅ Ziel Δ_{OA} [%]	∅ Quellen Δ_{OA} [%]	∅ Prozentrang [%]	∅ Ziel Δ_{OA} [%]	∅ Quellen Δ_{OA} [%]	∅ Prozentrang [%]	∅ Ziel Δ_{OA} [%]	∅ Quellen Δ_{OA} [%]	∅ Prozentrang [%]
Zufällige Quelle	19.9	17.8	54.7	15.5	13.3	56.9	17.1	14.0	56.3	26.0	22.0	52.4
Alle Quellen	26.4	24.3	39.1	11.2	9.1	66.7	11.1	8.0	78.1	22.2	18.3	57.0
SDS Single	5.6	3.6	85.8	5.8	3.7	83.5	6.0	2.9	89.5	4.9	0.9	96.6
UDS Single	5.6	3.5	86.8	5.0	2.9	86.6	5.8	2.7	89.9	4.8	0.8	96.7

6.3 Approximatives Modell

In diesem Abschnitt wird das approximative Quellenselektionsmodell aus Kapitel 4 evaluiert. Insbesondere soll gezeigt werden, dass die dort eingeführten Approximationen eine sinnvolle Vereinfachung des statistischen Modells erlauben. Durch die resultierende lineare Laufzeitkomplexität wird die Verarbeitung von sehr großen Datensätzen und einer Multiquellenselektion erst ermöglicht. Zunächst soll der erweiterte experimentelle Aufbau im Vergleich zu den bisherigen Experimenten erläutert werden. Eine Diskussion der Ergebnisse findet im Anschluss statt.

6.3.1 Experimenteller Aufbau

Die Aufbereitung der Datensätze erfolgt zu großen Teilen identisch zum Abschnitt 6.2. Für den *Small* Datensatz werden zwecks besserer Vergleichbarkeit dieselben Lern- und Teststichproben verwendet. Für alle anderen Datensätze werden diese pro Runde durch ein Bootstrap Sampling der unterabgetasteten Daten erzeugt. Die Lern- und Teststichproben enthalten jedoch pro Patch 15000 Lernbeispiele und werden

ohne Berücksichtigung der Klassenlabel gezogen. Die relativen Klassenhäufigkeiten der Stichproben entsprechen somit den echten Klassenwahrscheinlichkeiten.

Neben den beiden Referenzmethoden werden die untersuchten Methoden für die folgenden Experimente in vier Versuchsblöcke aufgeteilt:

Einzelquellenselektion: Für die Einzelquellenselektion werden auch hier die Domänendistanzen SDS *Single* und SDU *Single* untersucht. Die TV Distanz wird im approximativen Modell durch die MMD ersetzt. Der Kernelparameter σ ist für die MMD ein kritischer Parameter und muss daher sorgsam eingestellt werden. Zum Vergleich werden die beiden Strategien aus Abschnitt 4.4 gegenübergestellt. Die Methoden *Heuristik* und *Kernel Tuning* entsprechen hierbei der Median Heuristik und dem *Golden Section Search* basierten Optimierungsverfahren. Für die *Golden Section Search* wird die Anzahl der Iterationen auf 10 festgeschrieben.

Multiquellenselektion: Die Berechnung der Quellengewichte gemäß dem Gradientenabstiegsverfahren aus Abschnitt 4.3 erfordert die Einstellung von zwei Parametern, dem Parameter λ, welcher die Dichte des Lösungsvektors regelt, und dem Abbruchkriterium β. Die optimale Parametrisierung soll durch eine erschöpfende Gittersuche ermittelt werden. Der Parameter λ wird mit den Werten 0.5 und 0.25 konfiguriert. Der Parameter β wird so gesteuert, dass jeweils maximal 2, 4, 6 und 8 Quellen selektiert werden. Daher gilt für den Suchraum der Gittersuche $(\lambda, L) \in \{0.5, 0.25\} \times \{2, 4, 6, 8\}$. Für diesen und allen folgenden Versuchsblöcken wird σ mit der Methode *Kernel Tuning* optimiert.

Bootstrap Aggregierung: Die Bootstrap Aggegierung aus Abschnitt 4.5.1 mittelt die geschätzten Quellengewichte über mehrere Lernstichproben um Effekte der Stichprobenverzerrung und einer Überanpassung zu reduzieren. In diesen Experimenten wird die Effektivität dieses Vorgehens für die Einzelquellenselektion und für die Multiquellenselektion ermittelt. Für die Multiquellenselektion wurden die Parameter gemäß den Ergebnissen aus dem vorherigen Versuchsblock auf $\lambda = 0.5$ und $L = 4$ eingestellt.

Asymmetrische Domänendistanz: In diesem Versuchsblock wird die asymmetrische Variante der MMD aus Abschnitt 4.5.2 untersucht. Die überwachte Domänendistanz, in welcher die MMD durch diese AMMD ersetzt wurde wird für diese Versuche als A-SDS bezeichnet, die unüberwachte Domänendistanz analog als A-UDS. Der Parameter $N_{\max}$ der AMMD ergibt sich nach Abschnitt 4.5.2 mit $q = 0.05$ und $p = 0.05$ zu $N_{\max} = 59$. Zum Abschluss wird der Einfluss der Bootstrap Aggegierung auf die asymmetrische Domänendistanz untersucht.

Die Ergebnisse zu den einzelnen Datensätzen sind in den Tabellen 6.4 bis 6.10 zusammengefasst. Diese Tabellen enthalten die gemittelten Resultate bezüglich eines

arithmetischen Mittelwerts, wobei die Resultate der besten Methode in jeder Spalte fett markiert wurden. Um die Streuung der Methoden beurteilen zu können werden weiterhin in den Abbildungen 6.4 bis 6.10 *Box-Whisker-Plots* der erzielten Ziel Δ OA präsentiert. Die Darstellung orientiert sich an Tukeys Arbeiten [132]. Jede Box beschreibt hier den 25% bis 75% Perzentilbereich der Messwerte. Die orangene Markierung kennzeichnet dabei den Median. Die Länge der Fühler entspricht dem 1,5 fachen des Interquartilabstandes, beschnitten auf den extremsten Messwert, welcher innerhalb dieses Bereichs liegt. Datenpunkte, welche außerhalb der Fühler liegen, werden als Ausreißer markiert.

6.3.2 Diskussion

Einzelquellenselektion: Ein direkter Vergleich beider Strategien zur Einstellung des Kernelparameters zeigt zunächst, dass keine Methode über alle Experimente dominieren kann. Die Differenz in Δ OA liegt dabei oft über 0.5% und kann daher nicht vernachlässigt werden. Ein Vergleich der Median Ziel Δ OA durch einen Wilcoxon Rangsummentest [142] zeigt allerdings über alle Datensätze und Merkmalsräume gemittelt, dass kein statistisch signifikanter Unterschied zwischen den Methoden auf einem 95% Signifikanzniveau nachweisbar ist. Bei dem Wilcoxon Rangsummentest handelt es sich um einen verteilungsfreien Test, welcher die Hypothese prüft, dass die Mediane von zwei gepaarten Stichproben unterschiedlich sind. Die Methode *Kernel Tuning* optimiert ein theoretisch optimales Kriterium nach Sriperumbudur u. a. [122]. Dieses Kriterium wurde jedoch dort für einen Hypothesentest zur Unterscheidung von Stichproben hergeleitet. Für die Anwendung in der Quellenselektion müssen außerdem Generalisierungseffekte aus Abschnitt 2.2.2 berücksichtigt werden. Der MMD liegt ein Klassifikationsmodell zugrunde, welches durch eine direkte Anwendung der Methode *Kernel Tuning* die maximale Separierbarkeit der *Lernstichproben* zum Ziel hat und somit eine Tendenz zur Überanpassung haben sollte. In ersten weiterführenden Untersuchungen wurde daher der Kernelparameter durch eine einfache *Shrinkage* Operation [28] regularisiert. Generell gilt $1/\sigma \to 0$, d. h. die Bandbreite des RBF Kernels sollte etwas größer eingestellt werden um einen glatteren Kernel zu erhalten. Insgesamt konnten so die Ergebnisse der Methode *Kernel Tuning* tendenziell etwas verbessert werden. Die Herleitung und korrekte Einstellung eines Regularisierungskriteriums muss jedoch an dieser Stelle entfallen. In allen folgenden Experimenten wird daher das unregularisierte *Kernel Tuning* zur Einstellung des Kernelparameters eingesetzt.

Multiquellenselektion: In den meisten Experimenten werden durch die Multiquellenselektion leichte Verbesserungen gegenüber einer Einzelquellenselektion erzielt. In den Experimenten *Small*/EXP, *Vaihingen*/COL, *Buxtehude*/COL+LBP, *Combined*/COL, *Combined*/COL+LBP sowie *Combined*/ALL kann jedoch das

Gegenteil beobachtet werden. Eine genaue Beobachtung zeigt, dass die Multiquellenselektion konsistent mit jeder zusätzlichen Quelle die Domänendistanz der synthetisierten Quelle zur Zieldomäne verringern oder zumindest erhalten kann. Jedoch können nach der ersten Quelle meist nur noch geringe Verbesserungen der Domänendistanz erzielt werden. Der Gradient, welcher im Gradientenabstiegsverfahren aus Abschnitt 4.3 konstruiert wird, ist somit sehr schwach ausgeprägt und daher anfälliger für Effekte der Stichprobenverzerrung sowie anderer Ursachen von Bias. Gleichzeitig bedeutet dies, dass in diesen Experimenten die Voraussetzungen für eine erfolgreiche Multiquellenselektion, insbesondere, dass sich gute Quellen als Konvexkombination der verfügbaren Lernstichproben darstellen lassen, nicht gegeben sind.

Aus den untersuchten Parametersätzen sollen im Folgenden Empfehlungen für die Einstellung der Parameter des Gradientenabstiegsverfahrens abgeleitet werden. Zunächst werden alle Methoden dieses Versuchsblocks durch einen Friedman Test verglichen. Der Friedman Test vergleicht mehrere gepaarte Stichproben auf Unterschiede in deren Medianwerten [36]. Der Test kann somit als Ersatz des Wilcoxon Tests für Vergleiche von mehr als zwei Stichproben verstanden werden. Die Nullhypothese, dass alle Methoden identisch sind, kann mit einem 95% Signifikanzniveau abgelehnt werden. Mit dem Post-hoc Nemenyi Test [36] dürfen daher die Methoden anschließend paarweise verglichen werden. Dieser zeigt, dass der Parameter λ keinen statistisch signifikanten Einfluss auf die Quellenselektion hat. Weiterhin gilt, dass sich die Parametrisierungen $L = 6$ und $L = 8$ vergleichbar verhalten und diese sich signifikant von $L = 2$ unterscheiden. In den meisten Experimenten gilt, dass für mehr als 4 Quellen die Ziel Δ OA entweder nicht verbessert werden kann oder sich sogar wieder verschlechtert. In allen folgenden Versuchen wird unter der Bezeichnung *Multi* daher eine Multiquellenselektion verstanden, welche mit den Parametern $\lambda = 0.5$ und $L = 4$ angesteuert wurde.

Bootstrap Aggregierung: Durch eine Bootstrap Aggregierung der Quellengewichte konnten die Ziel Δ OA in allen Experimenten teilweise um mehrere Prozentpunkte verbessert werden. Ein Friedman Test mit einer Post-hoc Nemenyi Analyse zeigt, dass diese Verbesserungen auf einen 95% Niveau statistisch signifikant sind. Obwohl die Multiquellenselektion mit einer Bootstrap Aggregierung der Einzelquellenselektion in vielen Experimenten überlegen ist, konnten keine statistisch signifikanten Unterschiede festgestellt werden.

Asymmetrische Domänendistanz: Die asymmetrischen Domänendistanzen nach Abschnitt 4.5.2 können aus deutlich reduzierten Quellenstichproben berechnet werden und besitzen somit einen reduzierten Speicherverbrauch im Vergleich zur klassischen symmetrischen MMD. Durch einen Friedman Test mit einer Post-hoc Nemenyi Analyse wird gezeigt, dass die Ergebnisse bezüglich der Ziel

Δ OA vergleichbar sind. Dies gilt ebenso bei gleichzeitiger Verwendung der Bootstrap Aggregierung. Somit muss die Verbesserung des Speicherverbrauchs nicht durch Einbußen in der Klassifikationsgenauigkeit erkauft werden. Anhand der experimentellen Ergebnisse wird weiterhin empirisch gezeigt, dass sowohl das Gradientenabstiegsverfahren aus Abschnitt 4.3 sowie der Algorithmus zur Einstellung des Kernelparameters aus Abschnitt 4.4 sinnvoll auf asymmetrische Domänendistanzen angewendet werden können.

Anhand der Resultate auf dem *Small* Datensatz können das statistische Quellenselektionsmodell mit dem approximativen Modell direkt verglichen werden. Ein Blick auf die Tabellen 6.3 und 6.4 zeigt, dass das statistische Modell zunächst deutlich bessere Ziel Δ OA bezüglich einer Einzelquellenselektion erzeugt. Die Unterschiede der beiden Modelle lassen sich im wesentlichen durch eine vereinfachte Kostenfunktion und der Umstellung von einem Bayesschen Mittelwertschätzer auf einen MAP Schätzer beschreiben. Eine erneute Analyse der erzeugten Markov-Ketten zeigt, dass die simulierte a-Posteriori Verteilung eine geringe Varianz besitzt. Der Mittelwertschätzer erzeugt in diesen Experimenten somit zum MAP Schätzer vergleichbare Schätzungen. Diskrepanzen zwischen den beiden Modellen müssen daher aus der Kostenfunktion abzuleiten sein. Die deutlich schnellere Laufzeit des approximativen Modells, welche aus der linearen Laufzeitkomplexität folgt, erlaubt jedoch die Verarbeitung von deutlich größeren Stichproben sowie die Anwendung einer Bootstrap Aggregierung. Aus der Tabelle 6.4 ist ersichtlich, dass insbesondere die Bootstrap Aggregierung diese Verluste, welcher aus der vereinfachten Kostenfunktion resultieren, wieder ausgleichen kann. Letztendlich ist das approximative Modell dem statistischen Modell in dieser Konfiguration sowohl in Klassifikationsgenauigkeit als auch in der Laufzeit überlegen.

Ein Vergleich der überwachten Domänendistanzen (SDS) und der unüberwachten Varianten (UDS) durch einen Wilcoxon Rangsummentest kann keine statistisch signifikanten Unterschiede der Medianleistung auf einem 95% Signifikanzniveau aufzeigen. Nahezu die vollständige Diskriminanz der Quellenselektion kann somit durch die Distanz zwischen den ungelabelten Ziel- und Quellenstichproben erklärt werden. Der zusätzliche Term $\epsilon_S(h)$, welcher den Generalisierungsfehler der Quelle S beschreibt, kann als Regularisierungsterm interpretiert werden. Letztendlich bevorzugt dieser Term bei der Wahl zwischen zwei bezüglich der Marginalverteilungen vergleichbaren Quellen diejenige Quelle , deren bedingte Klassenverteilungen die geringere Überlappung aufweisen. Die dazugehörige Entscheidungsfunktion sollte in der Regel somit eine geringe Komplexität besitzen. In den hier untersuchten Datensätzen beträgt der Unterschied der Generalisierungsfehler zwischen allen Bildpatches nur wenige Prozentpunkte und ist daher nahezu vernachlässigbar bezüglich der Abschätzung aus Formel (3.8).

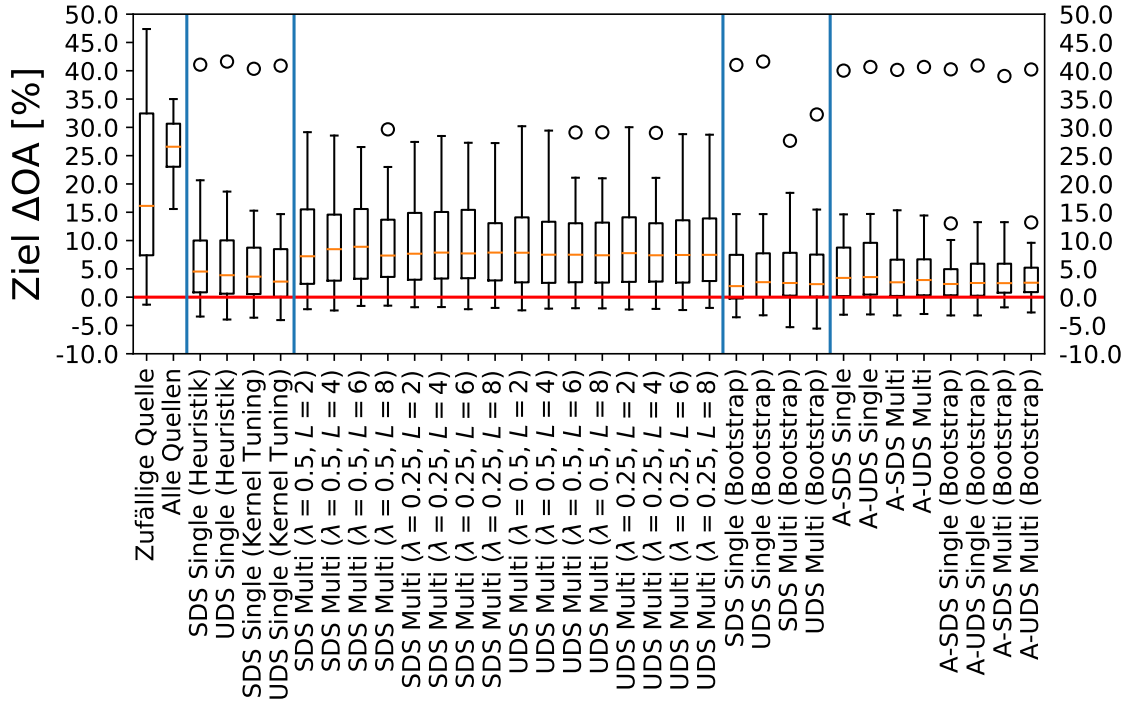
Der Vergleich aller Methoden hat ergeben, dass zwischen den Verfahren, welche Bootstrap Aggregierung verwenden, kein signifikanter Unterschied besteht. Dennoch kann aus der vorangegangenen Auswertung eine Methodenwahl begründet werden.

Zum einen sollte die unüberwachte Domänendistanz bevorzugt werden, da für diese nur der MMD basierte Distanzterm berechnet werden muss. Weiterhin hat die asymmetrische MMD einen geringeren Speicherverbrauch als die symmetrische Variante. Abschließend ergeben sich trotz der fehlenden Signifikanz geringe Unterschiede in der Median Ziel Δ OA zwischen der Einzel- und Multiquellenselektion. Für die Methode A-UDS *Multi (Bootstrap)* ergibt sich gemittelt über alle Datensätze und Merkmalsräume ein Verlust in Genauigkeit von 3.9%, verglichen mit 4.0% für die *Single* Variante. Die Resultate für die gewählte Methode sind noch einmal in Abbildung 6.11 zusammengefasst. Der direkte Vergleich der Mittelwerte und Mediane zeigt deutlich, dass den Ergebnissen eine schiefe Verteilung zugrunde liegt. Ein genauerer Blick auf die Ergebnisse zeigt, dass die mittleren Genauigkeiten durch wenige starke Ausreißer verschoben werden. Für ca. 80% aller Bildpatches liegt die Ziel Δ OA bei unter 5%. Bei einzelnen Bildpatches trifft jedoch die Annahme, dass sich unter den Quellenkandidaten eine sinnvolle Quelle befindet, nicht zu. Abbildung 6.12 zeigt ein Beispiel des *Hannover* Datensatz. Hier wird einer der Bildpatches nahezu vollständig von der Klasse *Baum* bedeckt, während die anderen Patches typische urbane Siedlungsstrukturen zeigen. Ein Verlust von nahezu 20% ist somit unvermeidbar. Statistische Korrelationsanalysen konnten bisher kein Kriterium nachweisen, welches es erlauben würde, diese Situationen ohne Kenntnis von Klassenlabels im Zielpatch zu erkennen.

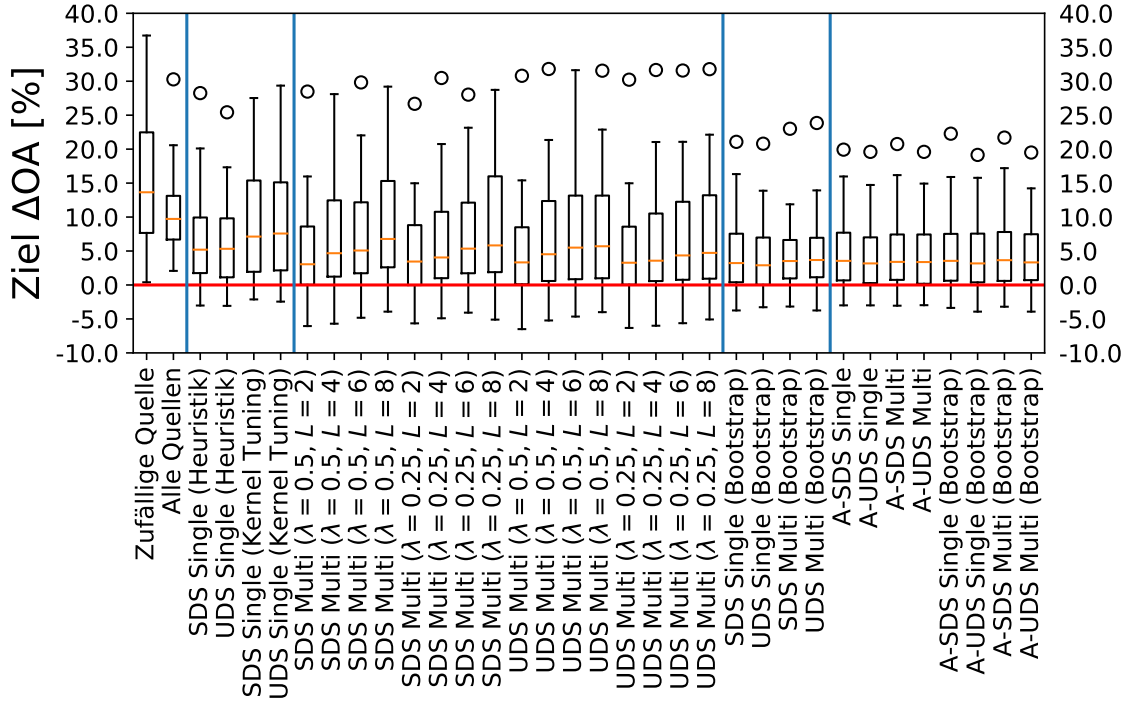
Die unüberwachte Einzelquellenselektion auf Basis der UDS oder A-UDS Domänendistanzen ist hochgradig parallelisierbar und zählt zur Klasse der hochparallen Algorithmen (engl. *Embarassingly Parallel*). Daher wurden für diese Methoden GPU basierte Implementierungen entwickelt und evaluiert. Die Laufzeit lag auf einem handelsüblichen Personalcomputer (CPU: Intel Core i5, GPU: NDVIDIA GTX 1060) je nach Datensatz pro Zielpatch zwischen 1 und 10 Sekunden. Die Integration der entwickelten Verfahren in interaktive Arbeitsabläufe, welche z. B. im aktiven Lernen [130] üblich sind, ist daher gegeben. Diese Messungen beinhalten bereits die automatische Einstellung der Kernelbandbreite sowie eine *Bootstrap Aggregierung*. Die Resultate entsprechen in den Ergebnistabellen somit den Methoden UDS *Single (Bootstrap)* und A-UDS *Single (Bootstrap)*. Weiterhin konnte für die symmetrische Variante ein Speicherverbrauch von ca. 1.7 MB pro Quelle berechnet werden. Für die asymmetrische MMD ergab sich ein Speicherverbrauch von 69 KB pro Quelle und somit eine Reduktion des Speicherverbrauchs nahezu um den Faktor 25. Die in dieser Arbeit entwickelte AMMD ermöglicht es somit deutlich größere Datensätze unter Berücksichtigung von sehr vielen Kandidatenquellen ohne Verwendung von Hochleistungsrechnern zu Verarbeiten.

Tabelle 6.4: Ergebnisse für das approximative Modell auf dem *Small* Datensatz.

	EXP			COL			COL + LBP			ALL		
	∅ Ziel Δ_{OA} [%]	∅ Quellen Δ_{OA} [%]	∅ Prozentrang [%]	∅ Ziel Δ_{OA} [%]	∅ Quellen Δ_{OA} [%]	∅ Prozentrang [%]	∅ Ziel Δ_{OA} [%]	∅ Quellen Δ_{OA} [%]	∅ Prozentrang [%]	∅ Ziel Δ_{OA} [%]	∅ Quellen Δ_{OA} [%]	∅ Prozentrang [%]
Zufällige Quelle	19.9	17.8	54.7	15.5	13.3	56.9	17.1	14.0	56.3	26.0	22.0	52.4
Alle Quellen	26.4	24.3	39.1	11.2	9.1	66.7	11.1	8.0	78.1	22.2	18.3	57.0
SDS Single (Heuristik)	7.7	5.7	79.1	7.2	5.1	80.7	7.9	4.8	83.6	8.8	4.9	89.6
UDS Single (Heuristik)	7.3	5.3	80.2	6.7	4.6	83.7	7.4	4.3	84.8	8.1	4.1	90.7
SDS Single (Kernel Tuning)	6.6	4.6	84.0	9.3	7.2	73.6	10.2	7.1	76.6	9.8	5.8	87.2
UDS Single (Kernel Tuning)	6.2	4.2	85.8	9.5	7.4	71.8	10.2	7.1	75.9	10.2	6.3	85.8
SDS Multi ($\lambda = 0.5$, $L = 2$)	9.5	7.4	72.7	5.4	3.3	86.8	10.5	7.4	76.0	9.7	5.7	86.8
SDS Multi ($\lambda = 0.5$, $L = 4$)	9.6	7.5	71.7	7.7	5.6	81.0	10.5	7.4	75.8	10.4	6.5	83.7
SDS Multi ($\lambda = 0.5$, $L = 6$)	9.9	7.8	70.1	8.0	5.9	78.9	11.4	8.3	71.5	10.0	6.0	84.3
SDS Multi ($\lambda = 0.5$, $L = 8$)	9.6	7.5	70.2	9.5	7.4	73.7	10.8	7.7	73.9	10.0	6.0	83.7
SDS Multi ($\lambda = 0.25$, $L = 2$)	9.7	7.5	71.9	5.4	3.3	86.6	10.8	7.7	75.6	9.9	5.9	86.5
SDS Multi ($\lambda = 0.25$, $L = 4$)	9.9	7.8	69.8	7.0	4.9	84.2	10.9	7.9	74.4	10.0	6.0	85.0
SDS Multi ($\lambda = 0.25$, $L = 6$)	9.8	7.7	70.0	7.9	5.9	79.3	11.5	8.4	71.9	9.8	5.8	85.4
SDS Multi ($\lambda = 0.25$, $L = 8$)	9.1	7.0	70.8	8.8	6.7	77.3	10.2	7.1	77.4	10.1	6.1	83.4
UDS Multi ($\lambda = 0.5$, $L = 2$)	9.1	7.0	72.5	5.5	3.4	86.7	10.1	7.0	76.1	10.3	6.4	84.7
UDS Multi ($\lambda = 0.5$, $L = 4$)	8.8	6.7	73.5	7.3	5.2	82.0	10.1	7.0	75.3	10.7	6.7	83.1
UDS Multi ($\lambda = 0.5$, $L = 6$)	8.8	6.7	73.0	8.1	6.0	78.9	10.2	7.1	74.8	10.9	6.9	82.7
UDS Multi ($\lambda = 0.5$, $L = 8$)	8.9	6.7	72.6	8.1	6.0	78.5	10.2	7.1	74.9	11.0	7.0	82.3
UDS Multi ($\lambda = 0.25$, $L = 2$)	9.0	6.9	72.8	5.4	3.3	86.9	10.0	6.9	75.9	10.3	6.3	84.8
UDS Multi ($\lambda = 0.25$, $L = 4$)	8.9	6.8	72.3	6.7	4.6	83.9	10.2	7.1	74.3	10.7	6.7	83.2
UDS Multi ($\lambda = 0.25$, $L = 6$)	9.0	6.9	72.0	7.4	5.3	80.8	10.3	7.2	74.5	11.0	7.0	82.8
UDS Multi ($\lambda = 0.25$, $L = 8$)	9.2	7.1	71.0	8.0	6.0	78.9	10.4	7.3	74.0	11.2	7.2	81.8
SDS Single (Bootstrap)	5.8	3.7	86.3	5.0	2.9	84.9	5.1	2.0	91.5	5.5	1.5	95.1
UDS Single (Bootstrap)	6.1	4.0	85.3	**4.5**	**2.3**	**88.0**	5.2	2.1	91.1	5.4	1.4	95.7
SDS Multi (Bootstrap)	5.3	3.2	**86.6**	4.8	2.6	85.6	4.8	1.7	90.7	4.8	0.9	94.6
UDS Multi (Bootstrap)	**5.1**	**3.0**	85.6	5.0	2.9	86.0	**4.7**	**1.6**	**91.9**	5.1	1.1	94.0
A-SDS Single	6.5	4.4	83.6	5.1	3.0	85.4	6.0	2.9	89.0	5.4	1.4	95.1
A-UDS Single	6.8	4.7	82.3	4.7	2.6	87.5	5.9	2.8	88.7	5.2	1.3	**95.8**
A-SDS Multi	5.8	3.8	84.1	5.0	2.9	85.4	5.8	2.7	88.5	5.1	1.1	94.9
A-UDS Multi	5.9	3.9	83.6	4.8	2.7	86.5	6.0	2.9	87.9	5.2	1.2	95.5
A-SDS Single (Bootstrap)	5.3	3.3	84.8	5.3	3.2	83.2	5.4	2.3	90.0	4.8	0.9	95.3
A-UDS Single (Bootstrap)	5.6	3.5	83.8	4.8	2.6	85.1	4.9	1.8	90.1	4.8	0.8	94.7
A-SDS Multi (Bootstrap)	5.8	3.7	83.4	5.5	3.4	81.7	5.3	2.2	89.2	**4.6**	**0.6**	95.2
A-UDS Multi (Bootstrap)	5.5	3.5	84.0	4.9	2.8	84.2	4.9	1.8	90.7	4.7	0.8	95.1

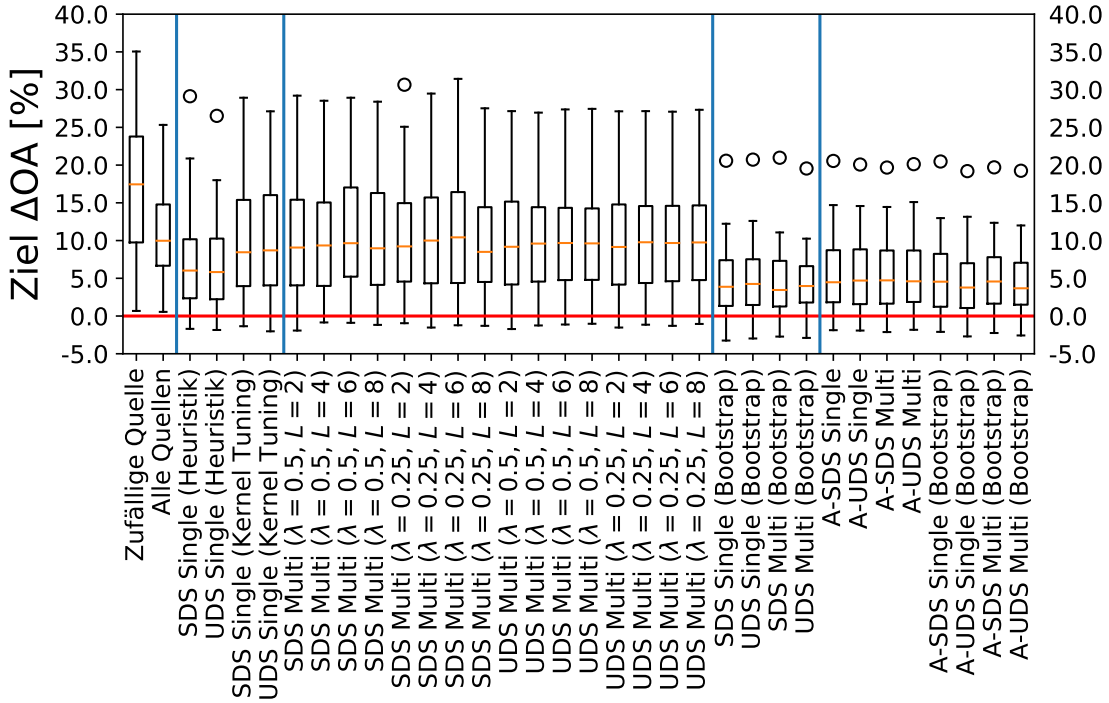


(a) EXP

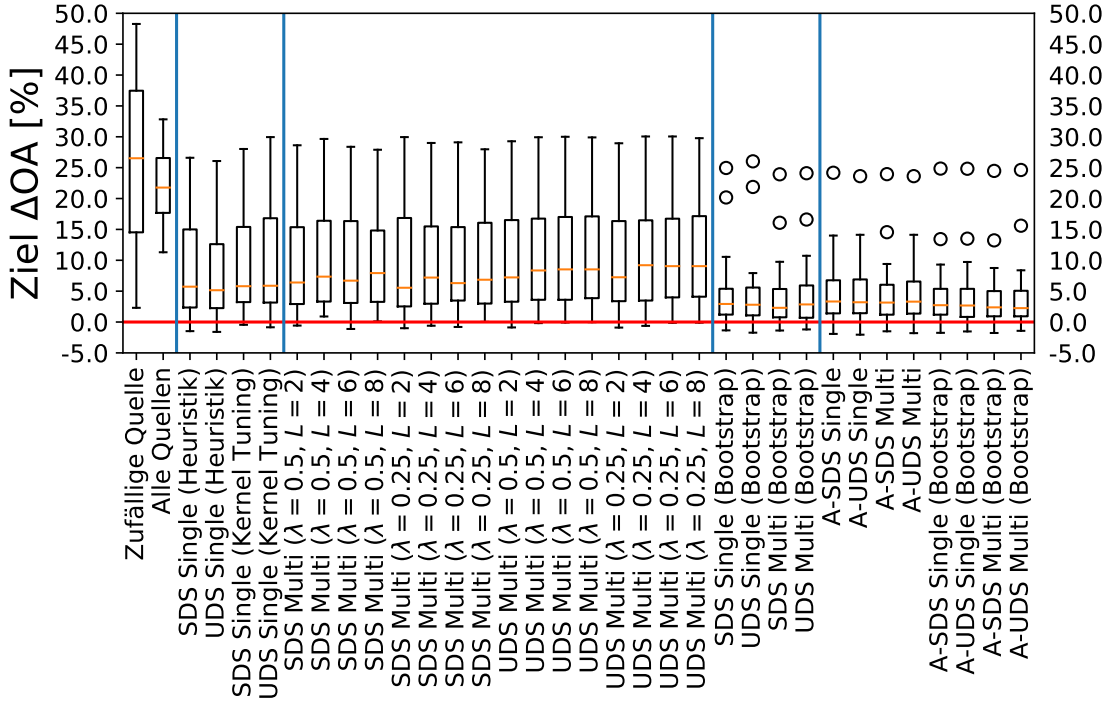


(b) COL

Abbildung 6.4: Box-Whisker-Plots für den *Small* Datensatz.



(c) COL + LBP

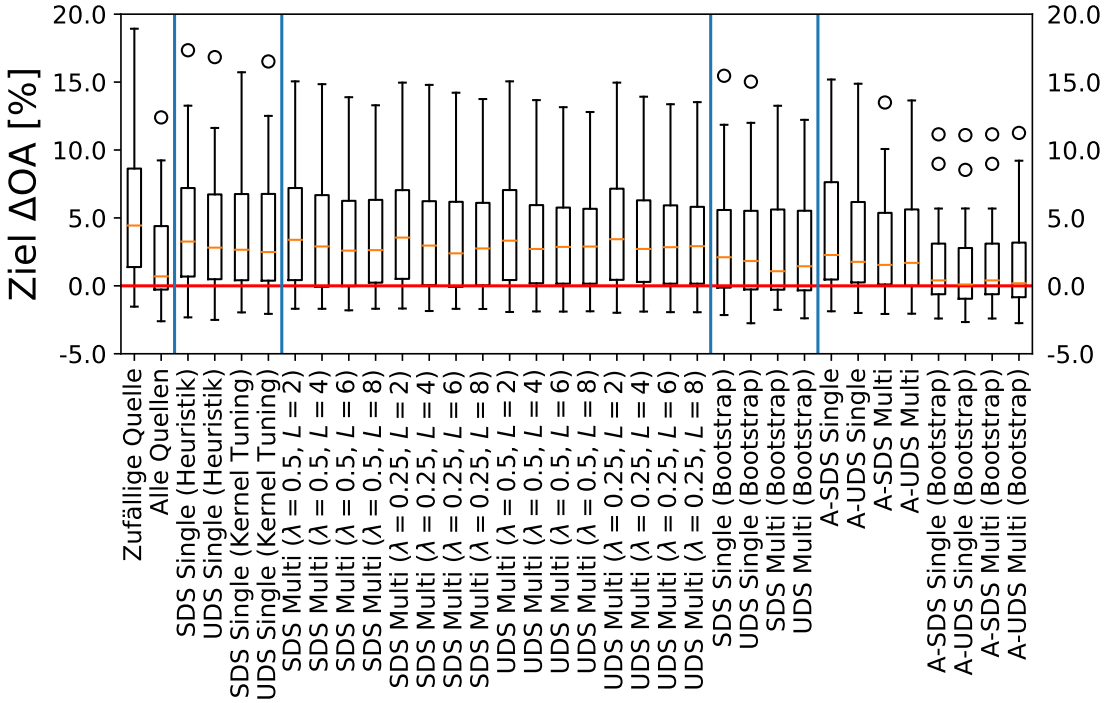


(d) ALL

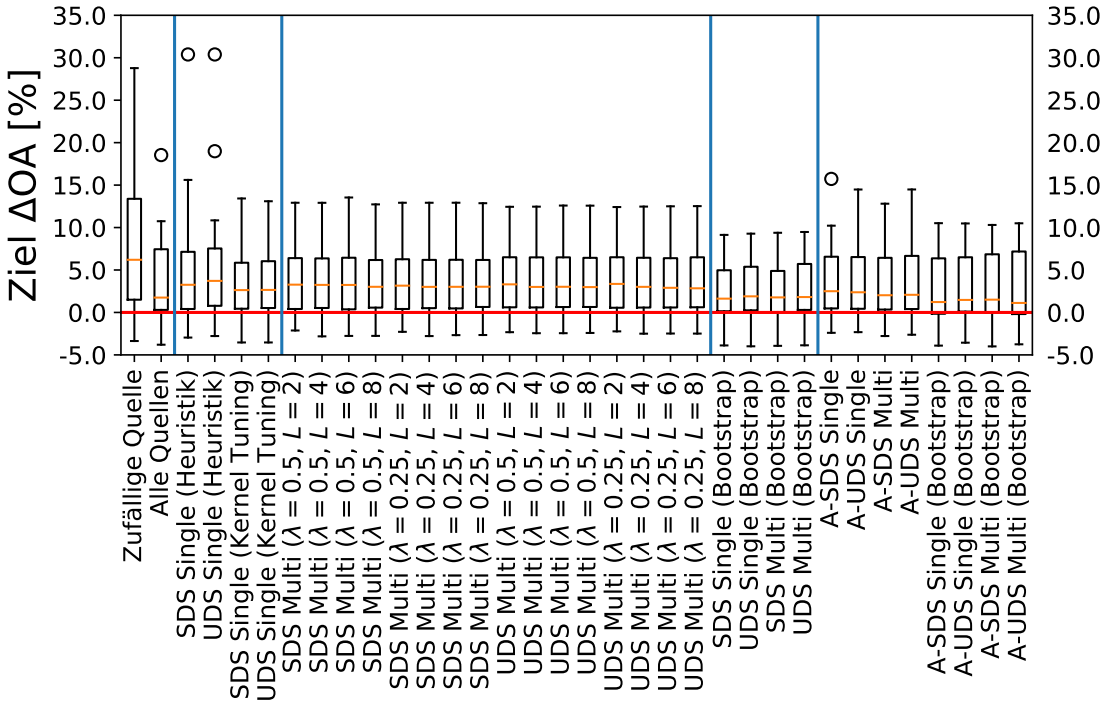
Abbildung 6.4: Box-Whisker-Plots für den *Small* Datensatz.

Tabelle 6.5: Ergebnisse für das approximative Modell auf dem *Vaihingen* Datensatz.

	EXP			COL			COL + LBP			ALL		
	∅ Ziel ΔOA [%]	∅ Quellen ΔOA [%]	∅ Prozentrang [%]	∅ Ziel ΔOA [%]	∅ Quellen ΔOA [%]	∅ Prozentrang [%]	∅ Ziel ΔOA [%]	∅ Quellen ΔOA [%]	∅ Prozentrang [%]	∅ Ziel ΔOA [%]	∅ Quellen ΔOA [%]	∅ Prozentrang [%]
Zufällige Quelle	5.7	6.1	49.9	8.4	9.1	53.4	8.7	7.1	53.0	7.2	5.4	51.8
Alle Quellen	2.3	2.8	76.5	3.9	4.6	72.5	4.5	2.9	83.2	3.8	2.0	82.8
SDS Single (Heuristik)	4.7	5.2	54.9	5.3	6.0	64.8	6.2	4.6	68.5	6.8	5.0	63.6
UDS Single (Heuristik)	4.3	4.7	57.5	5.9	6.6	62.1	6.1	4.6	69.2	5.9	4.1	70.0
SDS Single (Kernel Tuning)	4.3	4.8	59.7	3.4	4.1	71.4	4.7	3.1	75.3	4.6	2.8	71.8
UDS Single (Kernel Tuning)	4.3	4.8	60.9	3.5	4.2	71.2	4.7	3.1	74.9	4.5	2.6	72.8
SDS Multi ($\lambda = 0.5$, $L = 2$)	4.5	5.1	55.9	3.8	4.4	69.5	4.4	2.6	77.4	4.9	3.1	68.5
SDS Multi ($\lambda = 0.5$, $L = 4$)	4.2	4.8	58.4	3.7	4.4	70.5	4.2	2.4	79.1	5.0	3.2	67.3
SDS Multi ($\lambda = 0.5$, $L = 6$)	3.8	4.3	61.0	3.8	4.4	71.0	4.2	2.4	78.9	4.7	2.9	69.7
SDS Multi ($\lambda = 0.5$, $L = 8$)	3.9	4.4	59.9	3.6	4.3	70.8	4.1	2.3	80.3	4.7	2.9	69.7
SDS Multi ($\lambda = 0.25$, $L = 2$)	4.5	5.1	56.0	3.7	4.3	69.8	4.4	2.6	77.9	4.9	3.1	67.9
SDS Multi ($\lambda = 0.25$, $L = 4$)	4.0	4.6	59.2	3.7	4.3	70.3	4.2	2.4	79.2	4.9	3.1	69.3
SDS Multi ($\lambda = 0.25$, $L = 6$)	3.7	4.3	61.4	3.7	4.3	70.1	4.2	2.4	79.4	4.7	2.9	69.5
SDS Multi ($\lambda = 0.25$, $L = 8$)	3.8	4.3	60.4	3.7	4.4	69.8	4.2	2.4	79.2	4.8	3.0	69.4
UDS Multi ($\lambda = 0.5$, $L = 2$)	4.5	5.1	56.4	3.8	4.5	69.1	4.7	2.9	75.9	4.6	2.8	70.9
UDS Multi ($\lambda = 0.5$, $L = 4$)	3.9	4.5	59.6	3.7	4.4	69.4	4.5	2.7	76.8	4.5	2.7	71.6
UDS Multi ($\lambda = 0.5$, $L = 6$)	3.8	4.3	60.1	3.8	4.4	69.5	4.5	2.7	76.9	4.4	2.6	71.6
UDS Multi ($\lambda = 0.5$, $L = 8$)	3.7	4.2	60.1	3.8	4.4	69.3	4.5	2.7	77.0	4.4	2.6	71.8
UDS Multi ($\lambda = 0.25$, $L = 2$)	4.5	5.1	57.1	3.9	4.5	68.6	4.7	2.9	75.4	4.6	2.9	70.5
UDS Multi ($\lambda = 0.25$, $L = 4$)	4.0	4.6	59.5	3.6	4.3	69.9	4.6	2.8	76.5	4.5	2.7	71.2
UDS Multi ($\lambda = 0.25$, $L = 6$)	3.8	4.3	60.1	3.7	4.3	69.8	4.5	2.7	76.9	4.4	2.6	71.9
UDS Multi ($\lambda = 0.25$, $L = 8$)	3.8	4.3	59.9	3.7	4.3	69.8	4.5	2.7	77.0	4.4	2.6	71.8
SDS Single (Bootstrap)	3.6	4.1	64.3	**2.3**	**3.0**	78.2	3.4	1.6	84.9	3.5	1.7	82.8
UDS Single (Bootstrap)	3.4	3.9	67.2	2.6	3.2	76.8	3.7	2.0	83.0	3.4	1.6	83.3
SDS Multi (Bootstrap)	3.1	3.6	67.5	**2.3**	**3.0**	**78.7**	**3.1**	**1.4**	**88.1**	3.5	1.7	82.4
UDS Multi (Bootstrap)	3.0	3.5	68.1	2.6	3.2	77.7	3.7	1.9	84.9	3.3	1.5	86.7
A-SDS Single	4.3	4.8	60.1	3.8	4.5	71.1	4.9	3.2	72.9	4.5	2.7	72.9
A-UDS Single	3.8	4.3	64.1	3.7	4.3	72.1	4.9	3.2	73.4	4.4	2.6	74.2
A-SDS Multi	3.1	3.6	65.4	3.3	4.0	72.3	4.6	2.9	75.1	4.3	2.5	74.0
A-UDS Multi	3.3	3.8	65.6	3.6	4.3	71.2	4.6	2.8	76.6	4.1	2.3	75.6
A-SDS Single (Bootstrap)	1.8	2.3	77.0	2.7	3.3	74.8	3.4	1.7	82.9	3.0	1.2	88.2
A-UDS Single (Bootstrap)	**1.6**	**2.0**	**80.5**	2.7	3.4	74.3	3.8	2.0	79.9	3.0	1.2	88.2
A-SDS Multi (Bootstrap)	1.8	2.2	79.0	2.7	3.4	73.8	3.5	1.7	82.7	**2.9**	**1.1**	88.7
A-UDS Multi (Bootstrap)	1.8	2.3	80.1	2.7	3.3	74.4	3.7	1.9	81.8	**2.9**	**1.1**	**89.9**

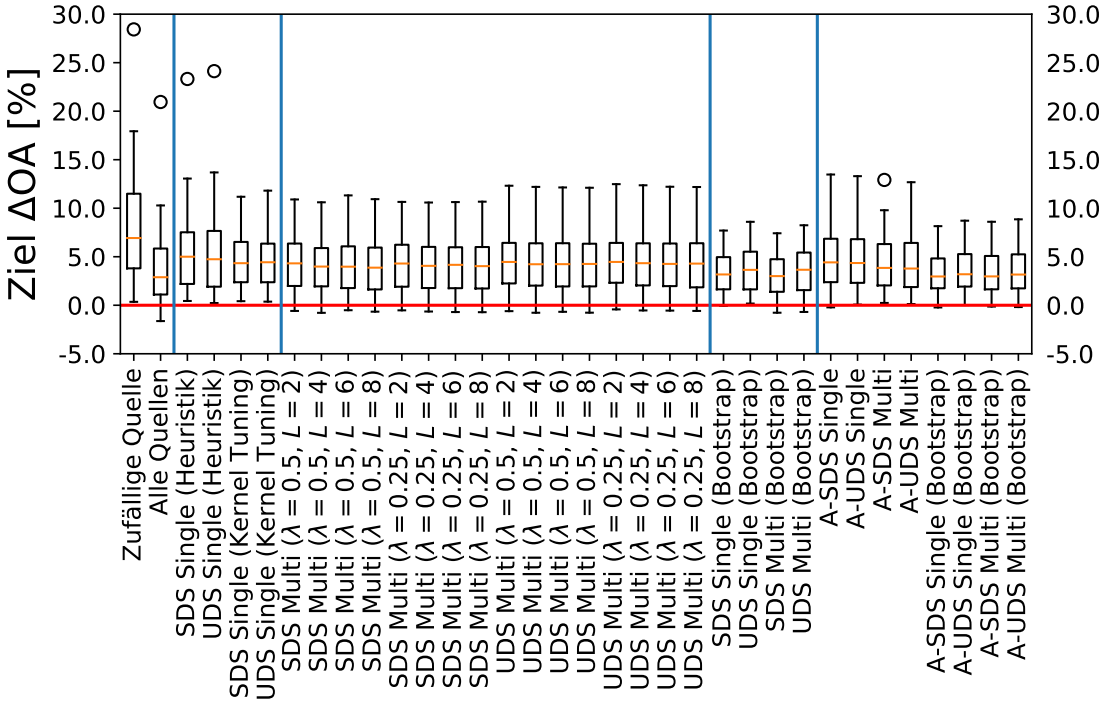


(a) EXP

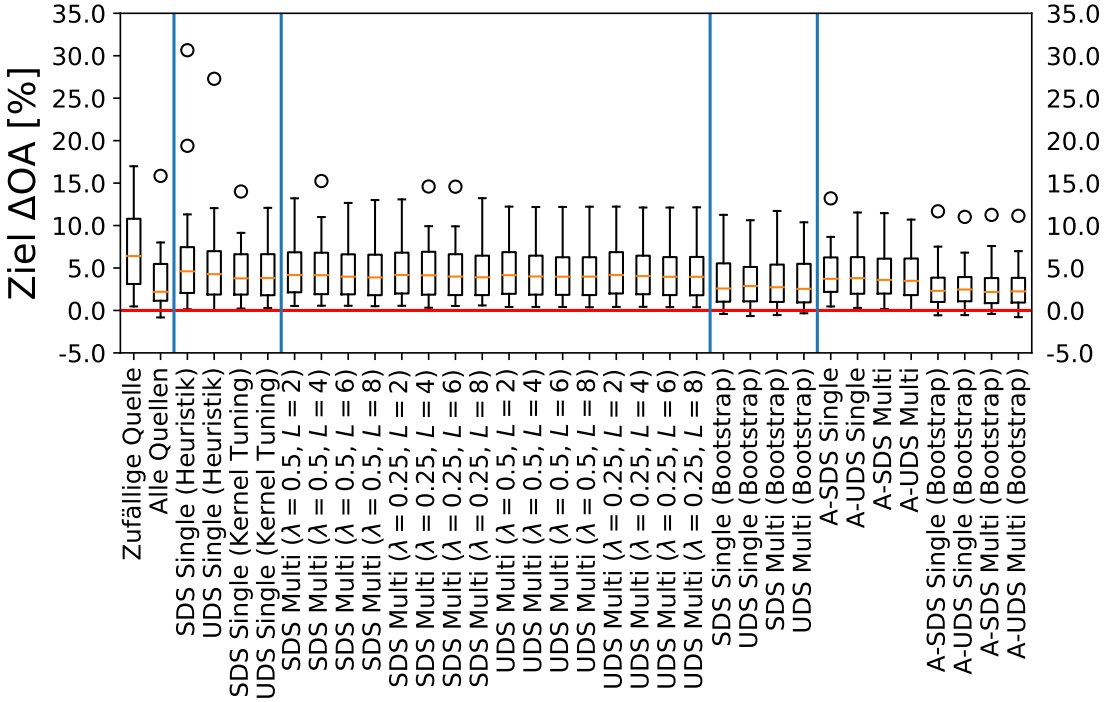


(b) COL

Abbildung 6.5: Box-Whisker-Plots für den *Vaihingen* Datensatz.



(c) COL + LBP

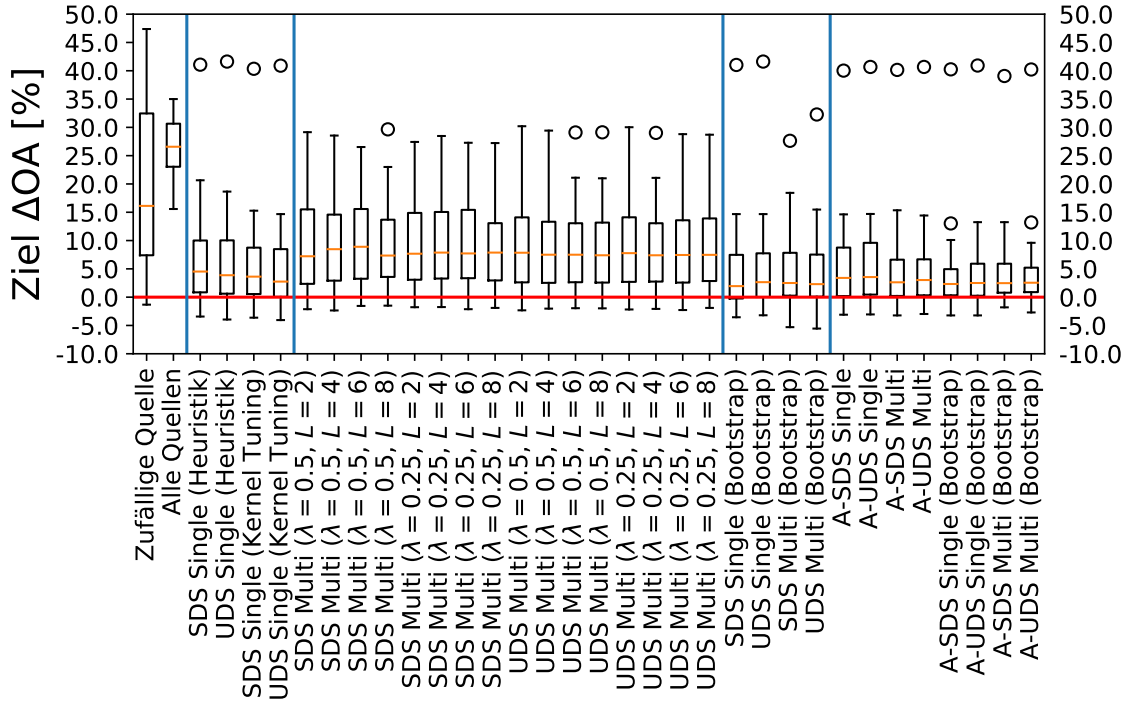


(d) ALL

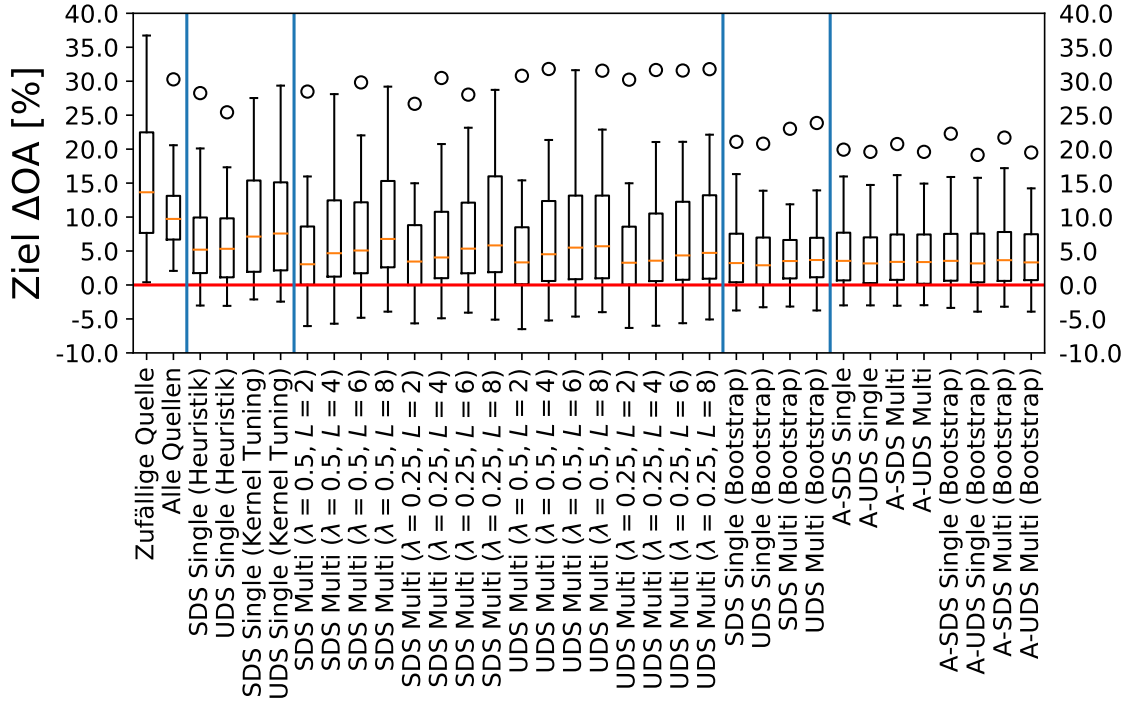
Abbildung 6.5: Box-Whisker-Plots für den *Vaihingen* Datensatz.

Tabelle 6.6: Ergebnisse für das approximative Modell auf dem *Potsdam* Datensatz.

	EXP			COL			COL + LBP			ALL		
	∅ Ziel Δ_{OA} [%]	∅ Quellen Δ_{OA} [%]	∅ Prozentrang [%]	∅ Ziel Δ_{OA} [%]	∅ Quellen Δ_{OA} [%]	∅ Prozentrang [%]	∅ Ziel Δ_{OA} [%]	∅ Quellen Δ_{OA} [%]	∅ Prozentrang [%]	∅ Ziel Δ_{OA} [%]	∅ Quellen Δ_{OA} [%]	∅ Prozentrang [%]
Zufällige Quelle	16.9	16.8	50.9	13.4	12.7	52.0	14.8	11.3	50.2	18.7	16.1	50.2
Alle Quellen	11.1	11.0	56.6	5.7	4.9	75.3	8.0	4.5	76.5	11.8	9.2	58.8
SDS Single (Heuristik)	**2.9**	**2.9**	**83.4**	7.0	6.3	70.8	8.0	4.5	77.1	6.5	3.9	79.3
UDS Single (Heuristik)	3.0	**2.9**	82.7	6.4	5.6	73.8	7.8	4.3	77.7	7.2	4.6	79.9
SDS Single (Kernel Tuning)	3.9	3.8	76.4	8.3	7.7	66.6	8.3	4.9	74.6	6.0	3.4	82.1
UDS Single (Kernel Tuning)	4.3	4.2	74.2	7.9	7.3	67.2	8.4	4.9	74.6	6.4	3.8	82.0
SDS Multi ($\lambda = 0.5$, $L = 2$)	3.6	3.6	77.4	7.0	6.3	70.7	7.8	4.4	75.8	5.7	3.1	81.5
SDS Multi ($\lambda = 0.5$, $L = 4$)	3.4	3.4	76.3	6.8	6.2	70.4	7.4	3.9	77.0	5.6	3.0	79.7
SDS Multi ($\lambda = 0.5$, $L = 6$)	3.6	3.5	75.9	6.5	5.8	72.5	7.4	4.0	77.0	6.0	3.4	79.3
SDS Multi ($\lambda = 0.5$, $L = 8$)	3.4	3.4	76.1	7.0	6.3	70.0	7.3	3.9	76.3	5.7	3.1	79.4
SDS Multi ($\lambda = 0.25$, $L = 2$)	3.3	3.3	78.3	7.2	6.5	69.7	7.6	4.1	76.0	5.7	3.1	81.9
SDS Multi ($\lambda = 0.25$, $L = 4$)	3.4	3.4	77.0	7.1	6.5	69.6	7.5	4.1	75.5	5.9	3.2	80.2
SDS Multi ($\lambda = 0.25$, $L = 6$)	3.5	3.5	76.5	6.5	5.9	71.8	7.6	4.1	75.5	5.7	3.1	79.8
SDS Multi ($\lambda = 0.25$, $L = 8$)	3.4	3.4	76.1	6.9	6.2	70.0	7.1	3.6	77.5	5.7	3.1	78.6
UDS Multi ($\lambda = 0.5$, $L = 2$)	3.9	3.8	76.1	6.5	5.8	72.0	7.3	3.9	77.9	6.5	3.8	79.3
UDS Multi ($\lambda = 0.5$, $L = 4$)	3.9	3.8	73.9	5.9	5.3	73.6	7.2	3.8	78.2	7.0	4.4	74.9
UDS Multi ($\lambda = 0.5$, $L = 6$)	3.8	3.8	74.1	5.8	5.1	74.2	7.2	3.7	78.5	7.2	4.6	73.3
UDS Multi ($\lambda = 0.5$, $L = 8$)	3.8	3.8	73.7	5.8	5.1	74.2	7.1	3.7	78.5	7.2	4.6	72.9
UDS Multi ($\lambda = 0.25$, $L = 2$)	3.8	3.8	76.2	6.6	5.9	71.7	7.4	3.9	77.9	6.4	3.7	79.7
UDS Multi ($\lambda = 0.25$, $L = 4$)	3.8	3.7	74.7	6.1	5.4	73.2	7.2	3.8	78.4	6.9	4.3	75.5
UDS Multi ($\lambda = 0.25$, $L = 6$)	3.8	3.8	74.7	6.0	5.3	73.5	7.2	3.7	78.7	7.1	4.5	74.0
UDS Multi ($\lambda = 0.25$, $L = 8$)	3.7	3.7	74.6	5.9	5.3	73.7	7.2	3.7	78.8	7.2	4.6	73.3
SDS Single (Bootstrap)	4.2	4.1	74.5	4.8	4.1	79.1	5.2	1.7	89.9	**3.9**	**1.3**	88.7
UDS Single (Bootstrap)	4.3	4.2	74.0	3.9	3.3	83.4	5.2	1.8	**90.2**	**3.9**	**1.3**	**89.8**
SDS Multi (Bootstrap)	3.1	3.0	77.8	4.3	3.7	81.6	**5.1**	**1.6**	89.9	4.4	1.8	85.3
UDS Multi (Bootstrap)	3.5	3.4	75.8	**3.8**	**3.2**	**84.0**	5.3	1.8	88.7	5.7	3.1	75.9
A-SDS Single	4.9	4.8	78.2	7.4	6.8	69.0	7.6	4.2	77.2	5.1	2.5	83.4
A-UDS Single	4.9	4.8	77.9	6.6	6.0	72.2	7.4	3.9	78.8	5.2	2.5	83.7
A-SDS Multi	4.1	4.0	78.3	6.7	6.0	70.6	7.1	3.7	79.3	4.9	2.3	84.3
A-UDS Multi	4.4	4.4	76.5	6.6	6.0	71.3	7.2	3.7	79.3	4.9	2.3	84.2
A-SDS Single (Bootstrap)	3.0	**2.9**	83.0	4.6	4.0	78.5	5.6	2.1	88.1	4.0	1.4	88.3
A-UDS Single (Bootstrap)	3.2	3.1	82.1	4.3	3.6	81.3	5.2	1.7	**90.2**	4.0	**1.3**	88.8
A-SDS Multi (Bootstrap)	3.0	**2.9**	80.3	4.6	4.0	78.4	5.4	1.9	88.1	4.0	**1.3**	87.9
A-UDS Multi (Bootstrap)	3.4	3.3	77.7	4.2	3.5	81.1	5.2	1.8	89.8	**3.9**	**1.3**	89.0

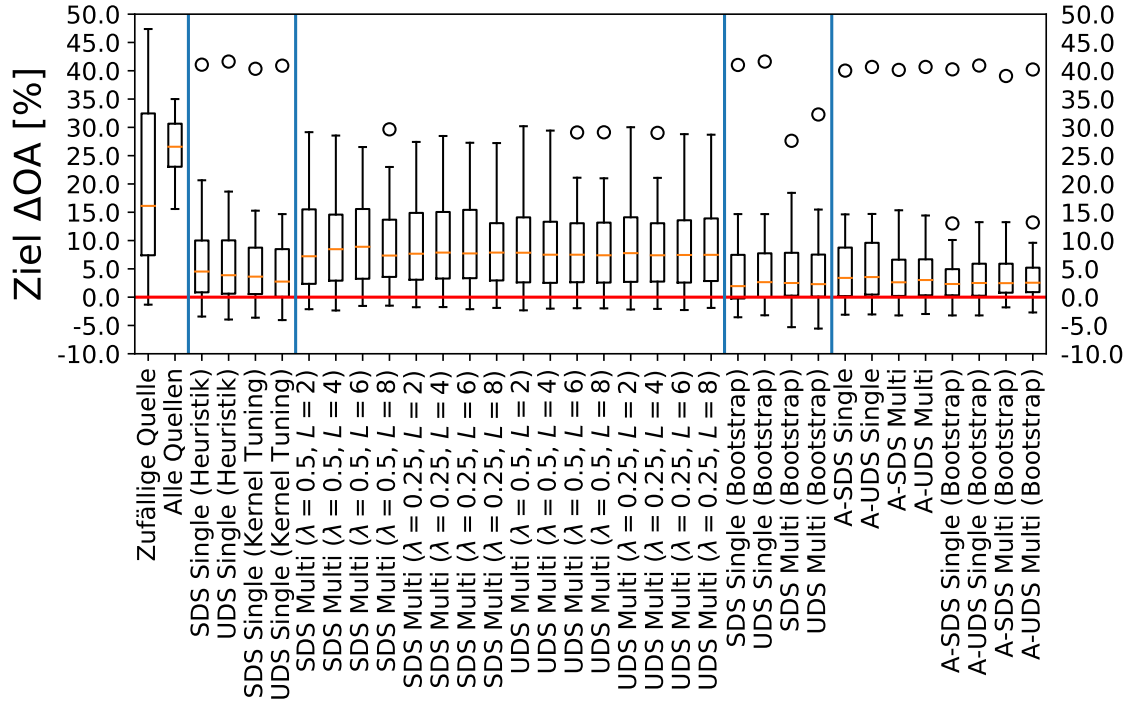


(a) EXP

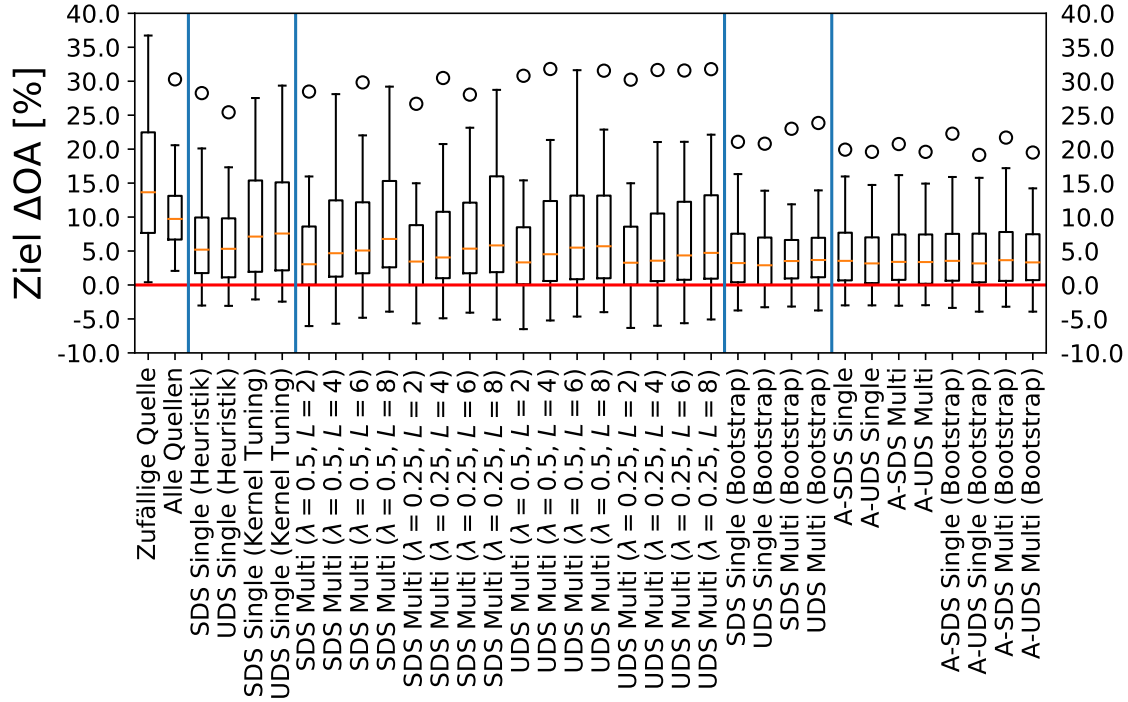


(b) COL

Abbildung 6.6: Box-Whisker-Plots für den *Potsdam* Datensatz.



(c) COL + LBP

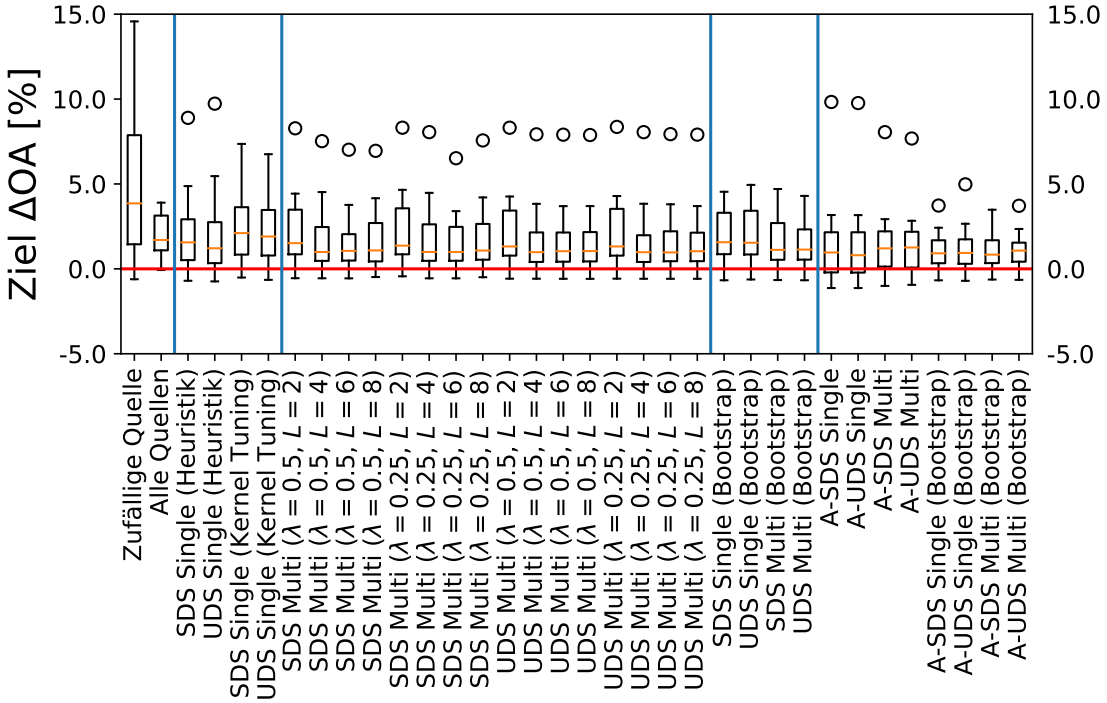


(d) ALL

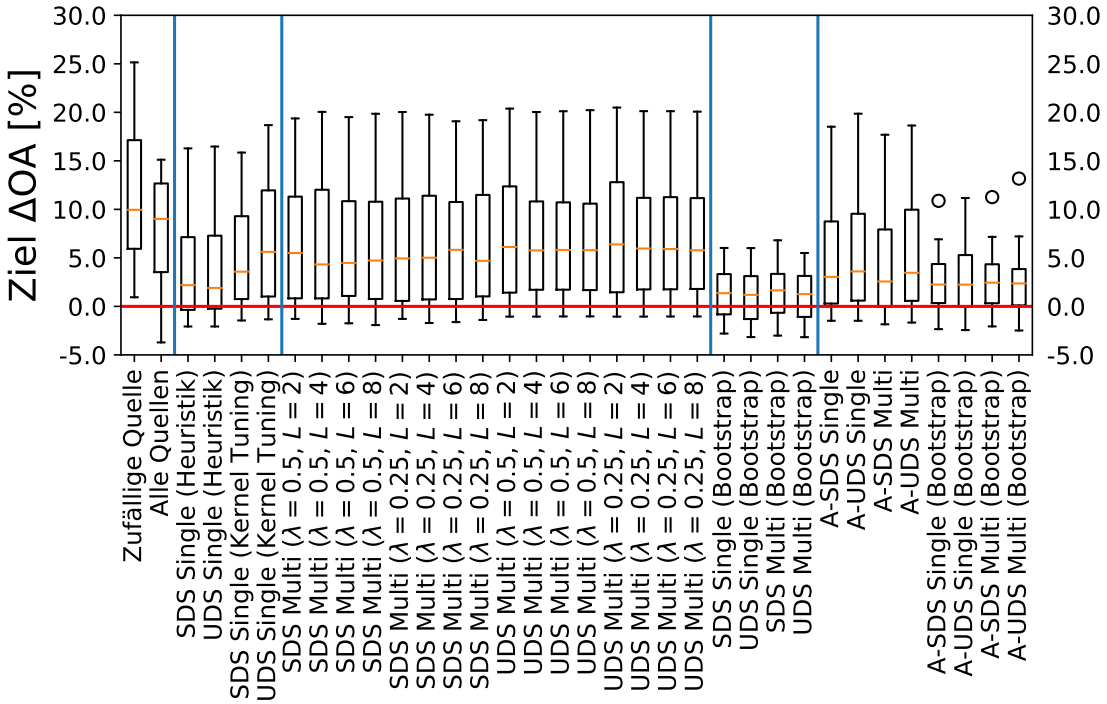
Abbildung 6.6: Box-Whisker-Plots für den *Potsdam* Datensatz.

Tabelle 6.7: Ergebnisse für das approximative Modell auf dem *Buxtehude* Datensatz.

	EXP			COL			COL + LBP			ALL		
	∅ Ziel ΔOA [%]	∅ Quellen ΔOA [%]	∅ Prozentrang [%]	∅ Ziel ΔOA [%]	∅ Quellen ΔOA [%]	∅ Prozentrang [%]	∅ Ziel ΔOA [%]	∅ Quellen ΔOA [%]	∅ Prozentrang [%]	∅ Ziel ΔOA [%]	∅ Quellen ΔOA [%]	∅ Prozentrang [%]
Zufällige Quelle	5.0	5.2	50.2	11.4	10.8	53.3	12.4	10.5	60.8	8.1	6.7	50.8
Alle Quellen	1.9	2.1	52.6	8.1	7.5	68.6	6.4	4.5	78.6	3.1	1.7	72.4
SDS Single (Heuristik)	2.4	2.5	61.7	4.3	3.7	78.8	4.8	2.9	83.2	2.7	1.3	83.8
UDS Single (Heuristik)	2.4	2.5	64.7	4.3	3.7	78.5	4.9	3.0	82.1	2.7	1.4	80.4
SDS Single (Kernel Tuning)	2.5	2.8	59.2	5.5	4.8	74.3	5.4	3.6	79.7	2.9	1.5	80.7
UDS Single (Kernel Tuning)	2.4	2.6	62.2	6.9	6.2	69.7	5.9	4.2	76.4	2.8	1.4	80.0
SDS Multi ($\lambda = 0.5$, $L = 2$)	2.4	2.7	61.3	6.8	6.3	65.8	7.4	5.7	74.2	3.1	1.8	79.3
SDS Multi ($\lambda = 0.5$, $L = 4$)	2.0	2.2	67.6	6.8	6.3	66.0	6.5	4.8	76.8	3.3	1.9	75.8
SDS Multi ($\lambda = 0.5$, $L = 6$)	1.8	2.1	67.2	6.4	5.9	66.0	6.5	4.7	75.8	3.1	1.7	77.2
SDS Multi ($\lambda = 0.5$, $L = 8$)	1.9	2.2	66.1	6.3	5.8	66.8	6.6	4.9	76.1	3.1	1.8	76.0
SDS Multi ($\lambda = 0.25$, $L = 2$)	2.5	2.7	61.2	6.8	6.3	66.5	7.1	5.4	75.0	3.3	1.9	78.3
SDS Multi ($\lambda = 0.25$, $L = 4$)	2.0	2.3	67.1	6.8	6.2	66.2	6.1	4.3	76.7	3.0	1.6	79.3
SDS Multi ($\lambda = 0.25$, $L = 6$)	1.7	2.0	68.1	6.5	6.0	66.9	6.8	5.0	75.8	3.2	1.8	77.1
SDS Multi ($\lambda = 0.25$, $L = 8$)	2.0	2.3	65.1	6.6	6.1	63.5	6.7	5.0	76.1	3.2	1.9	76.4
UDS Multi ($\lambda = 0.5$, $L = 2$)	2.3	2.6	63.2	7.4	6.9	64.0	7.6	5.8	73.3	3.0	1.6	79.2
UDS Multi ($\lambda = 0.5$, $L = 4$)	1.9	2.1	68.2	7.0	6.5	64.4	7.1	5.3	74.0	2.9	1.6	78.1
UDS Multi ($\lambda = 0.5$, $L = 6$)	1.9	2.1	67.4	6.9	6.4	64.4	7.0	5.2	74.4	2.9	1.5	78.6
UDS Multi ($\lambda = 0.5$, $L = 8$)	1.9	2.1	66.9	6.9	6.4	64.3	7.0	5.2	74.0	2.9	1.6	78.5
UDS Multi ($\lambda = 0.25$, $L = 2$)	2.4	2.6	63.1	7.5	7.0	63.1	7.6	5.9	72.8	2.9	1.6	79.7
UDS Multi ($\lambda = 0.25$, $L = 4$)	1.9	2.1	68.5	7.1	6.6	64.0	7.1	5.4	73.8	2.9	1.6	78.5
UDS Multi ($\lambda = 0.25$, $L = 6$)	1.9	2.1	68.1	7.1	6.6	64.2	7.1	5.3	73.1	2.9	1.6	78.2
UDS Multi ($\lambda = 0.25$, $L = 8$)	1.9	2.1	66.8	7.0	6.5	64.0	7.0	5.2	73.6	2.9	1.6	78.3
SDS Single (Bootstrap)	2.0	2.3	58.2	1.4	1.0	87.9	**2.6**	**0.8**	**90.3**	1.9	0.6	**86.8**
UDS Single (Bootstrap)	2.0	2.3	57.4	**1.0**	**0.6**	**90.0**	2.8	1.0	88.5	1.9	0.6	85.4
SDS Multi (Bootstrap)	1.6	1.9	62.2	1.6	1.2	86.8	3.2	1.4	88.3	2.1	0.7	84.6
UDS Multi (Bootstrap)	1.5	1.8	63.5	1.1	0.7	89.3	2.8	1.0	89.3	**1.8**	**0.5**	86.7
A-SDS Single	1.8	2.1	74.5	5.4	5.0	75.3	7.6	5.8	75.6	3.1	1.7	79.1
A-UDS Single	1.7	2.0	**75.5**	6.0	5.5	73.3	7.4	5.6	75.7	3.2	1.8	78.1
A-SDS Multi	1.7	2.0	68.4	4.7	4.3	76.8	6.9	5.1	77.8	3.1	1.8	74.8
A-UDS Multi	1.7	2.0	68.4	5.6	5.2	73.5	6.9	5.1	76.8	3.0	1.7	76.7
A-SDS Single (Bootstrap)	**1.1**	**1.4**	68.3	2.8	2.4	79.7	5.4	3.6	81.2	2.4	1.1	79.6
A-UDS Single (Bootstrap)	1.3	1.6	67.5	2.9	2.5	81.2	4.5	2.7	83.9	2.3	1.0	79.3
A-SDS Multi (Bootstrap)	**1.1**	**1.4**	67.1	3.0	2.6	77.1	5.4	3.6	80.7	2.5	1.2	77.4
A-UDS Multi (Bootstrap)	**1.1**	**1.4**	68.5	3.0	2.6	78.9	4.1	2.3	85.6	2.3	0.9	81.7

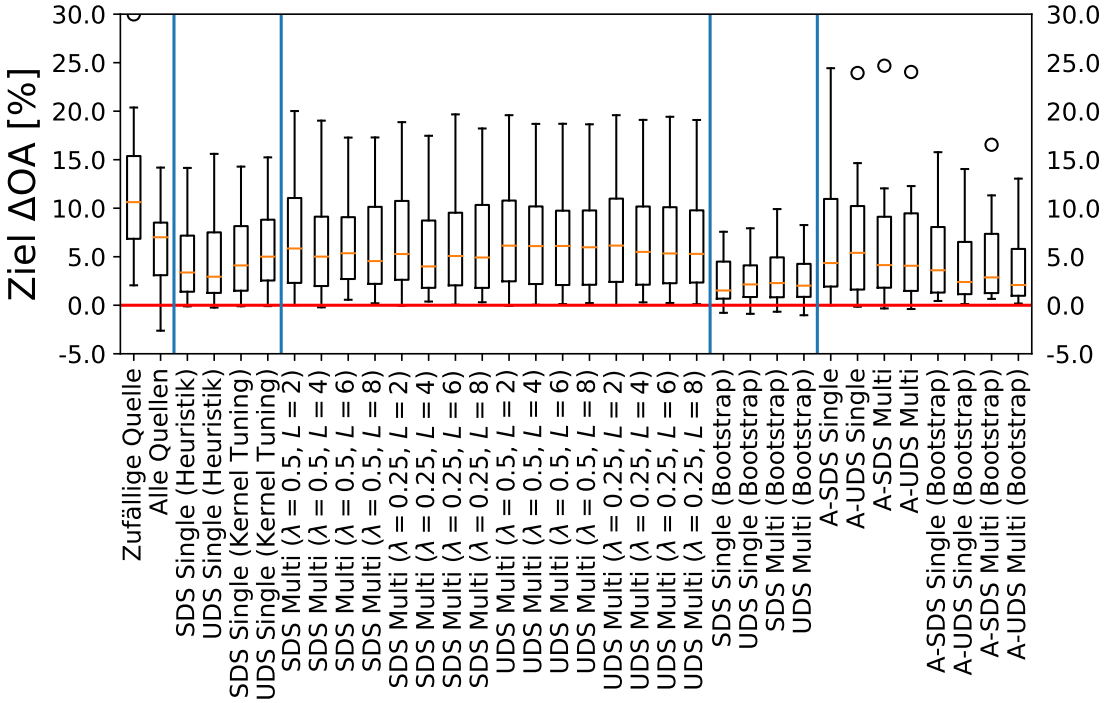


(a) EXP

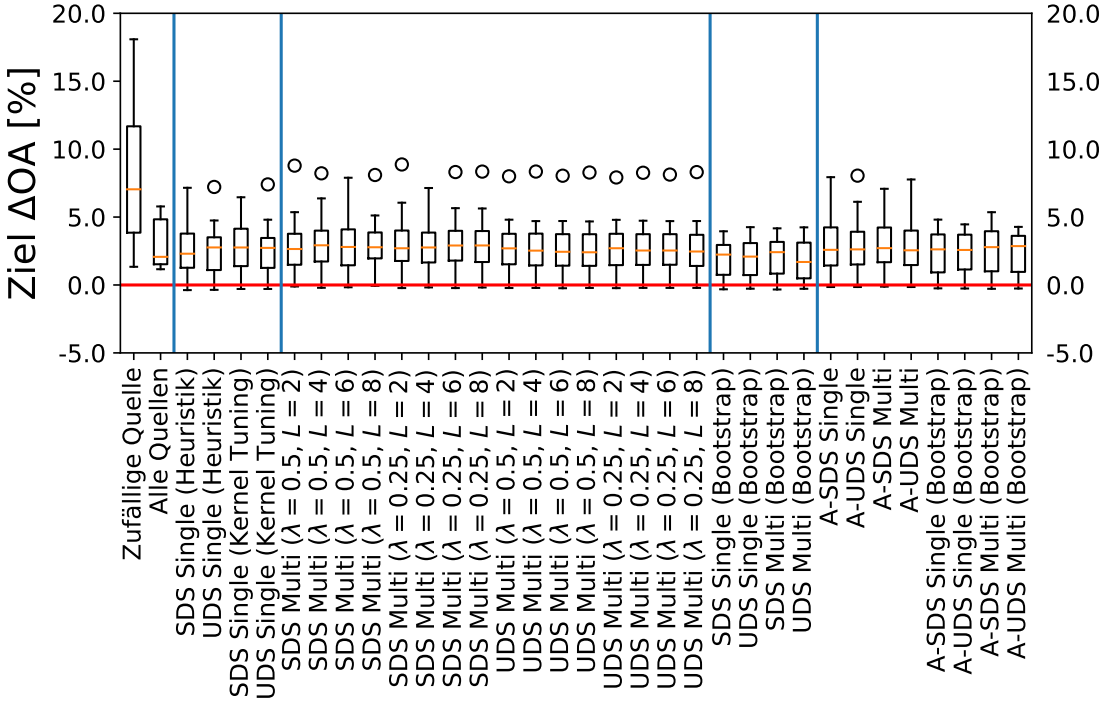


(b) COL

Abbildung 6.7: Box-Whisker-Plots für den *Buxtehude* Datensatz.



(c) COL + LBP

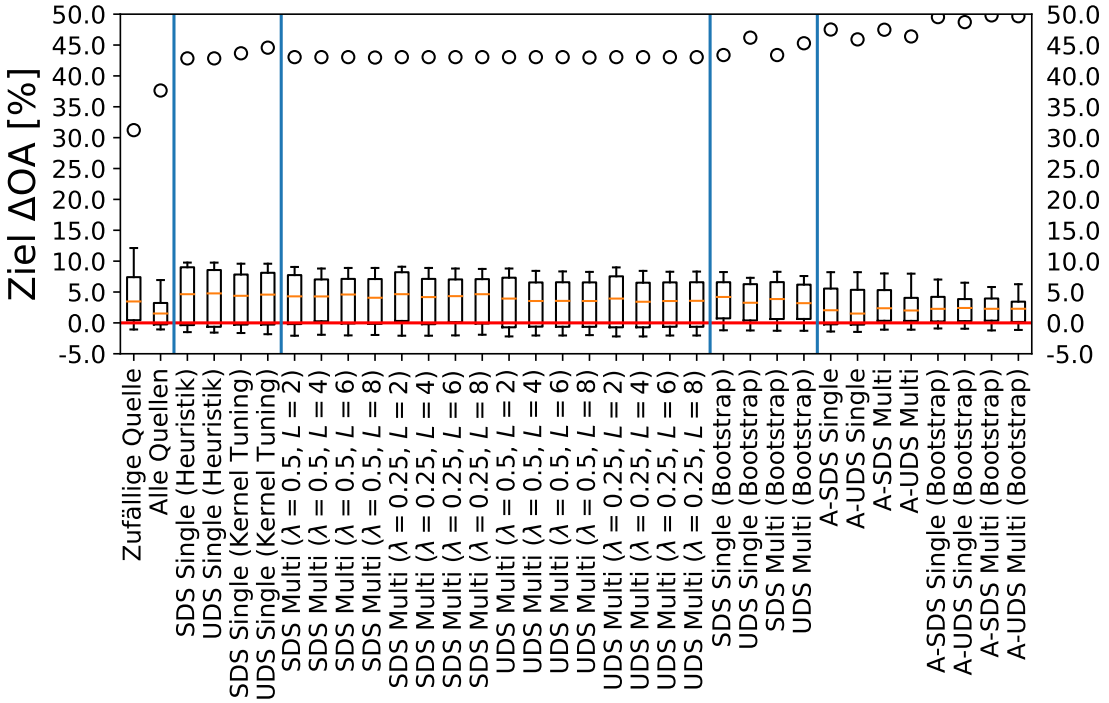


(d) ALL

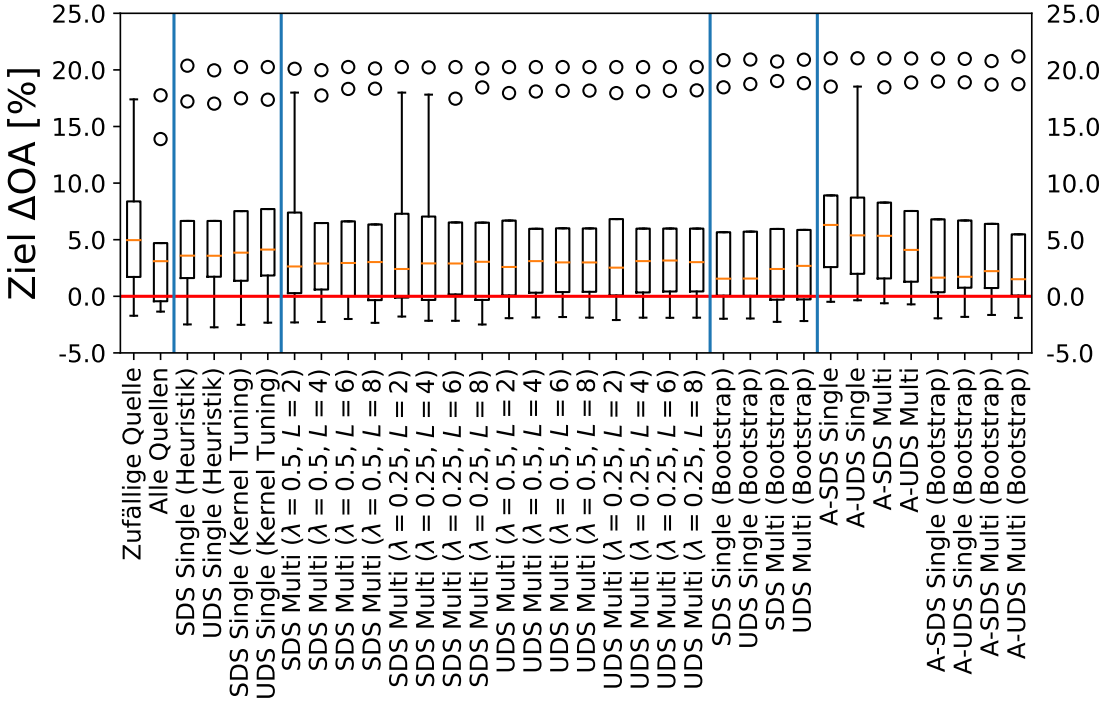
Abbildung 6.7: Box-Whisker-Plots für den *Buxtehude* Datensatz.

Tabelle 6.8: Ergebnisse für das approximative Modell auf dem *Hannover* Datensatz.

	EXP			COL			COL + LBP			ALL		
	∅ Ziel Δ_{OA} [%]	∅ Quellen Δ_{OA} [%]	∅ Prozentrang [%]	∅ Ziel Δ_{OA} [%]	∅ Quellen Δ_{OA} [%]	∅ Prozentrang [%]	∅ Ziel Δ_{OA} [%]	∅ Quellen Δ_{OA} [%]	∅ Prozentrang [%]	∅ Ziel Δ_{OA} [%]	∅ Quellen Δ_{OA} [%]	∅ Prozentrang [%]
Zufällige Quelle	6.6	4.8	55.0	6.0	5.3	49.3	8.4	5.4	48.5	6.1	2.7	57.3
Alle Quellen	**5.7**	**3.9**	**64.6**	**4.8**	**4.1**	**57.2**	**5.5**	**2.6**	**72.2**	5.0	1.7	71.9
SDS Single (Heuristik)	8.0	6.1	55.1	6.1	5.4	50.8	7.5	4.6	55.6	5.1	1.8	69.2
UDS Single (Heuristik)	7.9	6.0	55.7	6.0	5.3	52.4	7.1	4.2	62.4	5.0	1.7	71.3
SDS Single (Kernel Tuning)	7.8	6.1	52.6	6.1	5.7	52.6	7.4	4.4	55.9	5.4	2.1	67.1
UDS Single (Kernel Tuning)	8.0	6.2	53.0	6.4	6.0	49.9	7.4	4.4	57.3	5.5	2.3	64.3
SDS Multi ($\lambda = 0.5$, $L = 2$)	7.6	6.1	54.8	5.7	5.3	55.4	7.5	4.8	52.5	5.6	2.3	64.0
SDS Multi ($\lambda = 0.5$, $L = 4$)	7.5	6.0	53.3	5.6	5.2	54.3	7.6	4.8	50.5	5.6	2.3	62.8
SDS Multi ($\lambda = 0.5$, $L = 6$)	7.6	6.1	51.9	5.6	5.2	53.1	7.2	4.5	53.8	5.6	2.3	62.1
SDS Multi ($\lambda = 0.5$, $L = 8$)	7.5	5.9	53.8	5.5	5.1	54.2	7.3	4.6	51.8	5.5	2.2	61.8
SDS Multi ($\lambda = 0.25$, $L = 2$)	7.8	6.3	52.8	5.7	5.3	54.9	7.5	4.8	53.5	5.4	2.1	66.0
SDS Multi ($\lambda = 0.25$, $L = 4$)	7.6	6.0	52.7	5.6	5.1	54.2	7.4	4.6	53.5	5.6	2.3	63.5
SDS Multi ($\lambda = 0.25$, $L = 6$)	7.5	6.0	53.1	5.6	5.1	53.1	7.3	4.6	52.6	5.5	2.2	64.6
SDS Multi ($\lambda = 0.25$, $L = 8$)	7.7	6.1	51.9	5.5	5.1	54.0	7.3	4.6	51.8	5.5	2.2	61.9
UDS Multi ($\lambda = 0.5$, $L = 2$)	7.3	5.8	56.7	5.6	5.2	53.8	7.0	4.3	58.3	5.4	2.1	65.1
UDS Multi ($\lambda = 0.5$, $L = 4$)	7.1	5.6	55.6	5.7	5.3	52.5	6.9	4.2	58.4	5.4	2.1	65.7
UDS Multi ($\lambda = 0.5$, $L = 6$)	7.1	5.6	56.3	5.8	5.3	50.9	6.9	4.1	58.9	5.3	2.1	65.3
UDS Multi ($\lambda = 0.5$, $L = 8$)	7.1	5.6	56.2	5.8	5.3	50.9	6.9	4.1	58.8	5.3	2.1	65.7
UDS Multi ($\lambda = 0.25$, $L = 2$)	7.4	5.9	56.6	5.7	5.2	53.5	7.0	4.3	58.5	5.4	2.1	66.0
UDS Multi ($\lambda = 0.25$, $L = 4$)	7.1	5.5	57.0	5.7	5.3	52.8	7.0	4.2	59.0	5.4	2.1	66.1
UDS Multi ($\lambda = 0.25$, $L = 6$)	7.1	5.6	56.7	5.8	5.3	51.9	6.9	4.2	57.8	5.3	2.0	65.8
UDS Multi ($\lambda = 0.25$, $L = 8$)	7.1	5.6	56.5	5.8	5.3	51.3	6.9	4.1	58.8	5.4	2.1	65.4
SDS Single (Bootstrap)	7.6	6.0	48.6	5.2	4.6	54.0	6.6	3.7	60.7	**4.7**	**1.4**	**75.1**
UDS Single (Bootstrap)	7.5	5.9	49.9	5.3	4.7	54.6	6.5	3.7	62.2	**4.7**	1.5	72.8
SDS Multi (Bootstrap)	7.5	5.8	48.3	5.4	4.8	48.3	6.7	3.8	58.3	4.8	1.5	72.5
UDS Multi (Bootstrap)	7.4	5.8	49.2	5.5	4.9	49.3	6.6	3.7	60.3	4.8	1.5	72.6
A-SDS Single	7.3	5.7	55.9	7.8	7.2	34.7	8.8	6.0	38.3	5.6	2.3	66.0
A-UDS Single	7.0	5.3	58.4	7.5	6.9	36.9	8.7	5.9	40.0	5.5	2.2	66.9
A-SDS Multi	7.4	5.7	52.7	7.2	6.6	39.2	8.5	5.7	40.7	5.5	2.2	64.7
A-UDS Multi	7.0	5.4	55.8	6.7	6.1	43.9	7.8	5.0	47.2	5.3	2.0	67.6
A-SDS Single (Bootstrap)	7.4	5.7	52.1	5.6	5.0	50.7	7.6	4.8	48.9	4.8	1.5	70.8
A-UDS Single (Bootstrap)	7.2	5.5	53.9	5.6	5.0	50.1	7.0	4.1	54.4	4.8	1.6	71.7
A-SDS Multi (Bootstrap)	7.2	5.5	56.8	5.6	5.0	49.6	7.6	4.8	48.2	4.8	1.5	70.9
A-UDS Multi (Bootstrap)	7.2	5.5	56.8	5.3	4.7	52.8	6.8	3.9	55.7	4.8	1.5	71.7

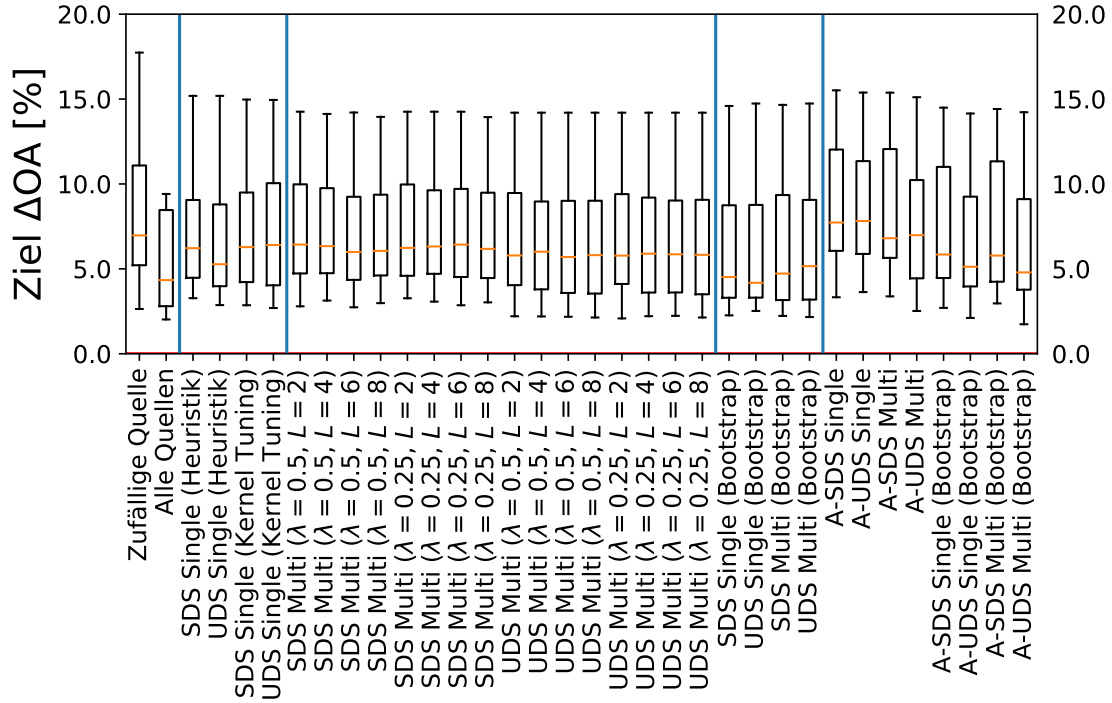


(a) EXP

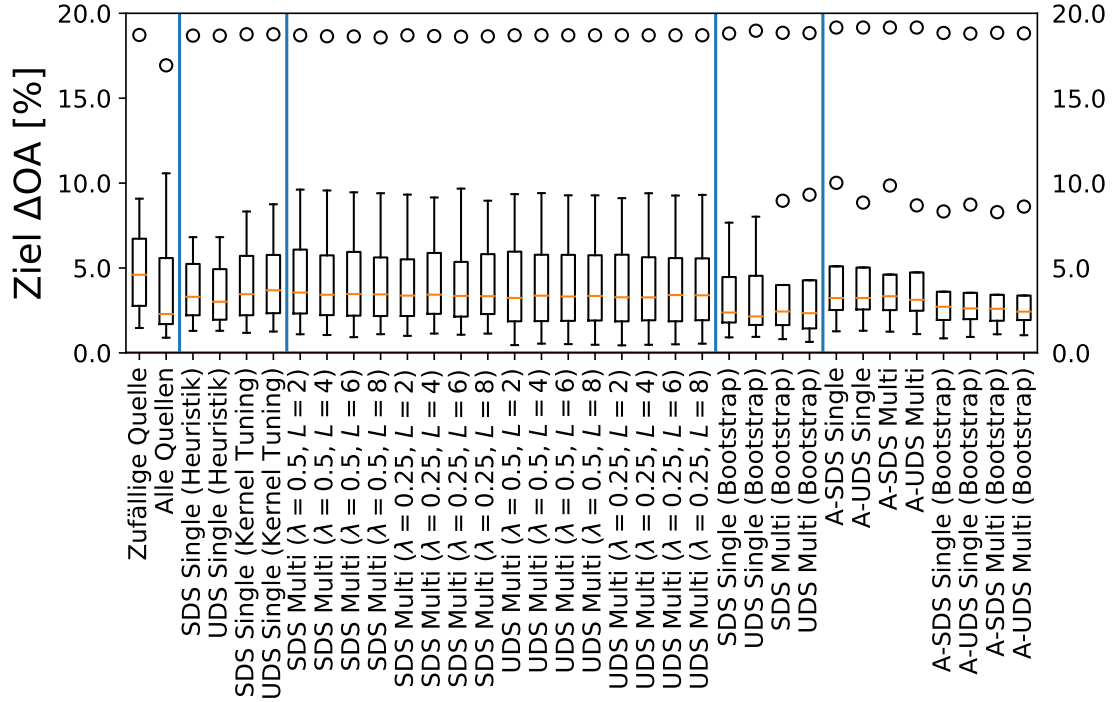


(b) COL

Abbildung 6.8: Box-Whisker-Plots für den *Hannover* Datensatz.



(c) COL + LBP

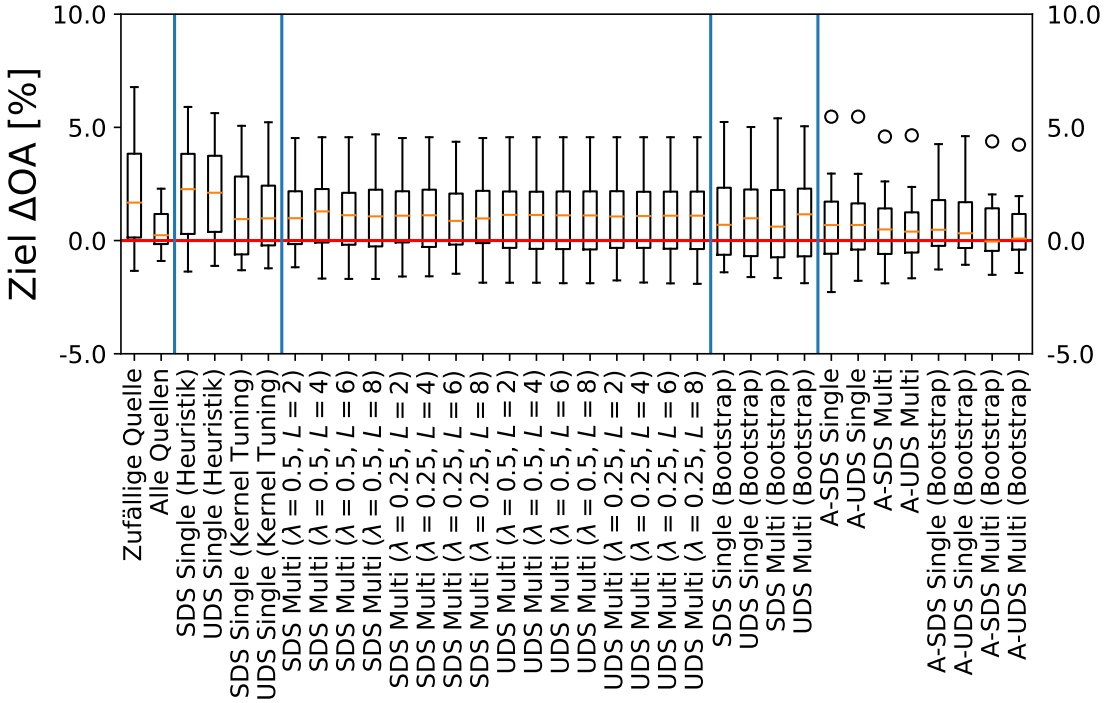


(d) ALL

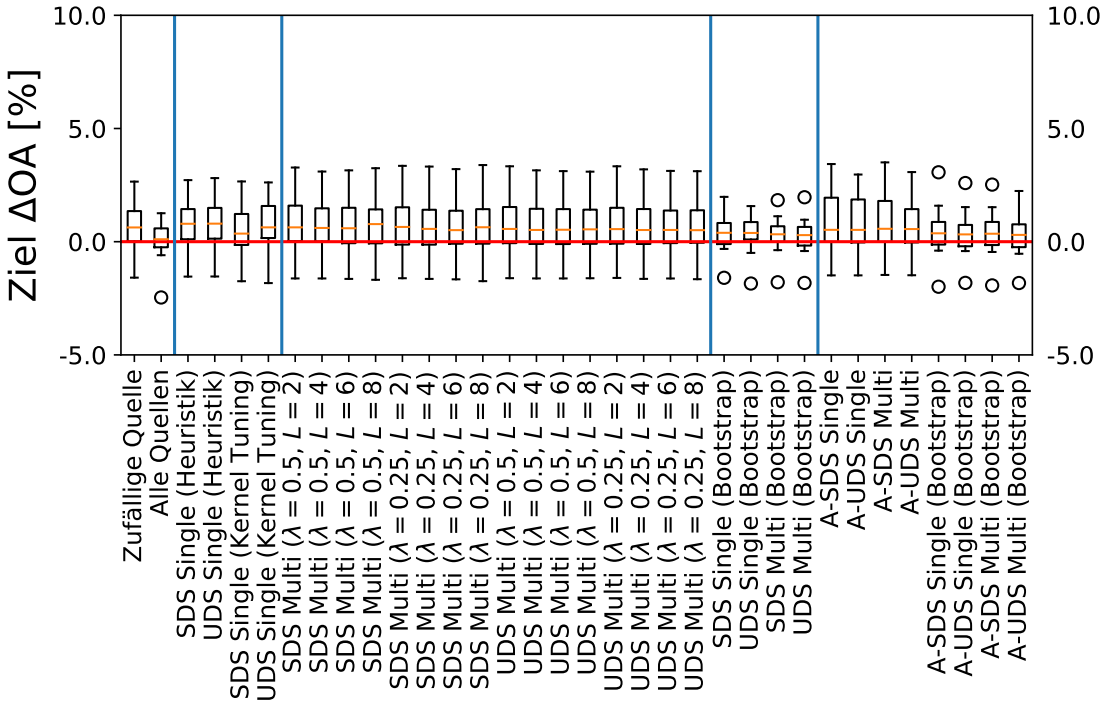
Abbildung 6.8: Box-Whisker-Plots für den *Hannover* Datensatz.

Tabelle 6.9: Ergebnisse für das approximative Modell auf dem *Nienburg* Datensatz.

	EXP			COL			COL + LBP			ALL		
	∅ Ziel ΔOA [%]	∅ Quellen ΔOA [%]	∅ Prozentrang [%]	∅ Ziel ΔOA [%]	∅ Quellen ΔOA [%]	∅ Prozentrang [%]	∅ Ziel ΔOA [%]	∅ Quellen ΔOA [%]	∅ Prozentrang [%]	∅ Ziel ΔOA [%]	∅ Quellen ΔOA [%]	∅ Prozentrang [%]
Zufällige Quelle	2.2	3.0	54.5	0.6	1.1	60.3	5.0	3.4	50.8	1.7	1.4	59.4
Alle Quellen	**0.5**	**1.3**	67.2	**0.0**	**0.5**	**77.5**	**1.5**	**−0.1**	**92.9**	1.4	1.1	59.0
SDS Single (Heuristik)	2.1	2.9	53.6	0.8	1.3	57.5	3.2	1.7	64.4	1.7	1.4	61.6
UDS Single (Heuristik)	2.1	2.9	52.9	0.8	1.3	56.8	3.3	1.7	63.3	1.8	1.5	57.3
SDS Single (Kernel Tuning)	1.3	2.2	64.3	0.5	1.1	67.6	3.5	1.9	60.1	1.7	1.3	64.4
UDS Single (Kernel Tuning)	1.3	2.1	64.1	0.7	1.2	61.8	3.9	2.3	55.6	1.8	1.5	60.0
SDS Multi ($\lambda = 0.5$, $L = 2$)	1.2	2.2	64.7	0.8	1.4	56.7	3.4	1.8	60.7	1.6	1.3	59.0
SDS Multi ($\lambda = 0.5$, $L = 4$)	1.2	2.2	64.7	0.7	1.3	56.9	3.2	1.6	62.9	1.5	1.2	60.4
SDS Multi ($\lambda = 0.5$, $L = 6$)	1.2	2.2	64.7	0.7	1.3	56.8	3.1	1.6	63.1	1.5	1.2	59.9
SDS Multi ($\lambda = 0.5$, $L = 8$)	1.2	2.2	64.2	0.8	1.3	55.6	3.2	1.7	59.3	1.6	1.3	57.5
SDS Multi ($\lambda = 0.25$, $L = 2$)	1.2	2.2	65.4	0.8	1.3	58.4	3.2	1.7	62.8	1.5	1.2	60.3
SDS Multi ($\lambda = 0.25$, $L = 4$)	1.2	2.2	64.9	0.7	1.3	58.3	3.2	1.6	60.8	1.4	1.1	61.7
SDS Multi ($\lambda = 0.25$, $L = 6$)	1.1	2.1	65.3	0.7	1.2	58.0	3.0	1.5	64.6	1.5	1.2	62.2
SDS Multi ($\lambda = 0.25$, $L = 8$)	1.2	2.1	65.1	0.7	1.3	57.9	3.1	1.5	62.0	1.5	1.2	58.9
UDS Multi ($\lambda = 0.5$, $L = 2$)	1.1	2.1	67.1	0.8	1.3	57.2	4.2	2.6	56.7	1.7	1.4	56.9
UDS Multi ($\lambda = 0.5$, $L = 4$)	1.1	2.1	66.7	0.7	1.3	57.9	4.1	2.6	58.2	1.7	1.4	57.1
UDS Multi ($\lambda = 0.5$, $L = 6$)	1.1	2.1	66.1	0.7	1.3	57.4	4.1	2.6	57.5	1.7	1.4	57.4
UDS Multi ($\lambda = 0.5$, $L = 8$)	1.1	2.1	66.0	0.7	1.3	57.2	4.1	2.6	57.8	1.7	1.4	56.5
UDS Multi ($\lambda = 0.25$, $L = 2$)	1.1	2.1	66.9	0.8	1.3	57.6	4.1	2.6	56.7	1.7	1.4	57.1
UDS Multi ($\lambda = 0.25$, $L = 4$)	1.1	2.1	67.1	0.7	1.3	58.0	4.1	2.6	58.2	1.6	1.3	58.3
UDS Multi ($\lambda = 0.25$, $L = 6$)	1.1	2.1	66.7	0.7	1.3	58.8	4.1	2.6	57.4	1.6	1.3	58.3
UDS Multi ($\lambda = 0.25$, $L = 8$)	1.1	2.1	66.4	0.7	1.3	58.3	4.1	2.6	57.5	1.7	1.3	57.5
SDS Single (Bootstrap)	1.1	2.1	62.1	0.3	0.9	64.6	2.2	0.7	74.9	1.3	1.0	63.5
UDS Single (Bootstrap)	1.1	2.1	63.6	0.3	0.9	65.4	2.2	0.7	78.2	1.4	1.2	59.7
SDS Multi (Bootstrap)	1.1	2.0	64.4	0.3	0.9	66.2	2.2	0.7	76.5	1.3	1.1	61.0
UDS Multi (Bootstrap)	1.1	2.1	64.3	0.2	0.8	69.6	2.1	0.5	79.7	1.5	1.2	58.6
A-SDS Single	0.9	1.8	**70.8**	0.9	1.5	57.8	3.8	2.2	50.8	1.5	1.2	66.3
A-UDS Single	1.0	1.9	68.5	0.8	1.4	58.8	4.1	2.6	48.9	1.6	1.3	62.7
A-SDS Multi	0.7	1.6	69.2	0.9	1.5	55.6	3.7	2.1	51.0	1.6	1.3	65.6
A-UDS Multi	0.7	1.6	70.7	0.7	1.3	58.3	3.9	2.3	52.6	1.6	1.3	62.6
A-SDS Single (Bootstrap)	0.9	1.8	62.6	0.5	1.1	63.2	2.5	0.9	71.8	1.3	1.0	65.0
A-UDS Single (Bootstrap)	0.8	1.8	64.0	0.4	1.0	64.3	2.5	0.9	71.5	1.4	1.1	63.6
A-SDS Multi (Bootstrap)	0.6	1.5	68.3	0.4	1.0	65.4	2.4	0.9	72.6	**1.2**	**0.9**	**67.9**
A-UDS Multi (Bootstrap)	0.6	1.5	70.0	0.3	0.9	68.5	2.4	0.8	74.4	1.3	1.0	64.2

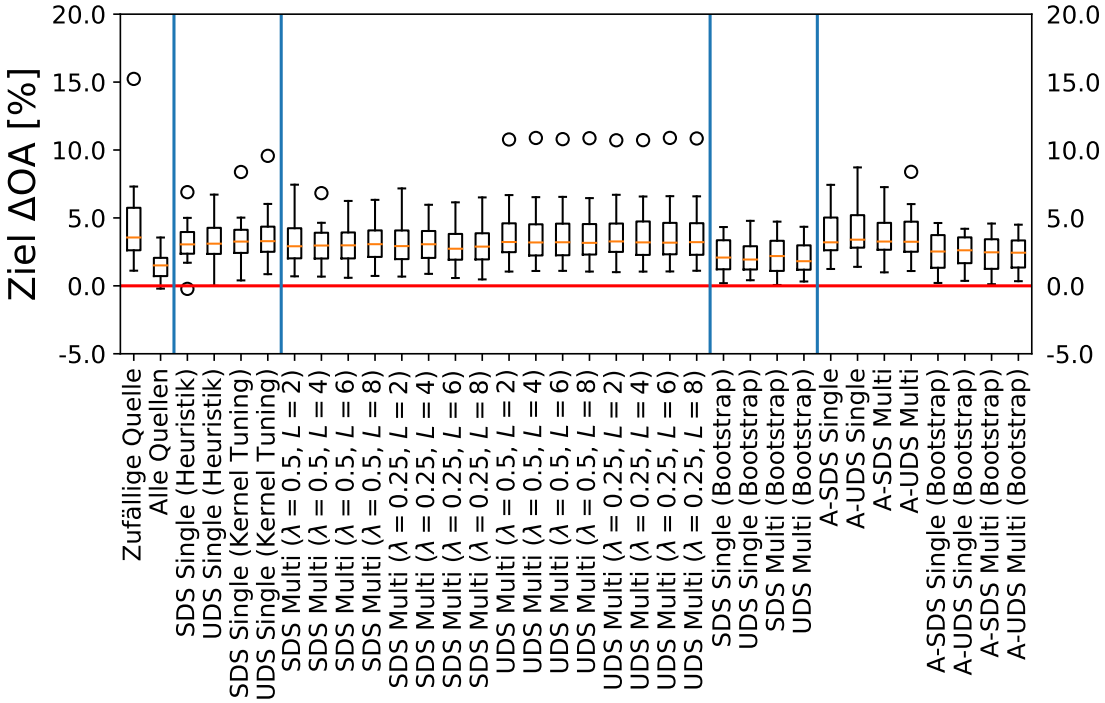


(a) EXP

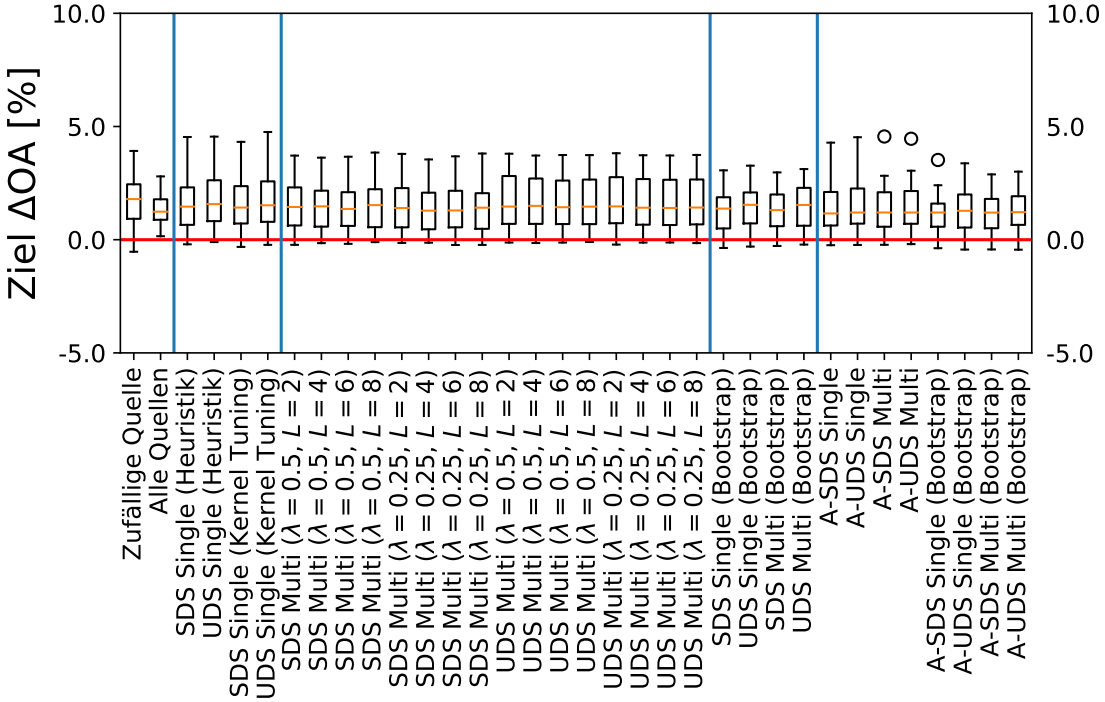


(b) COL

Abbildung 6.9: Box-Whisker-Plots für den *Nienburg* Datensatz.



(c) COL + LBP

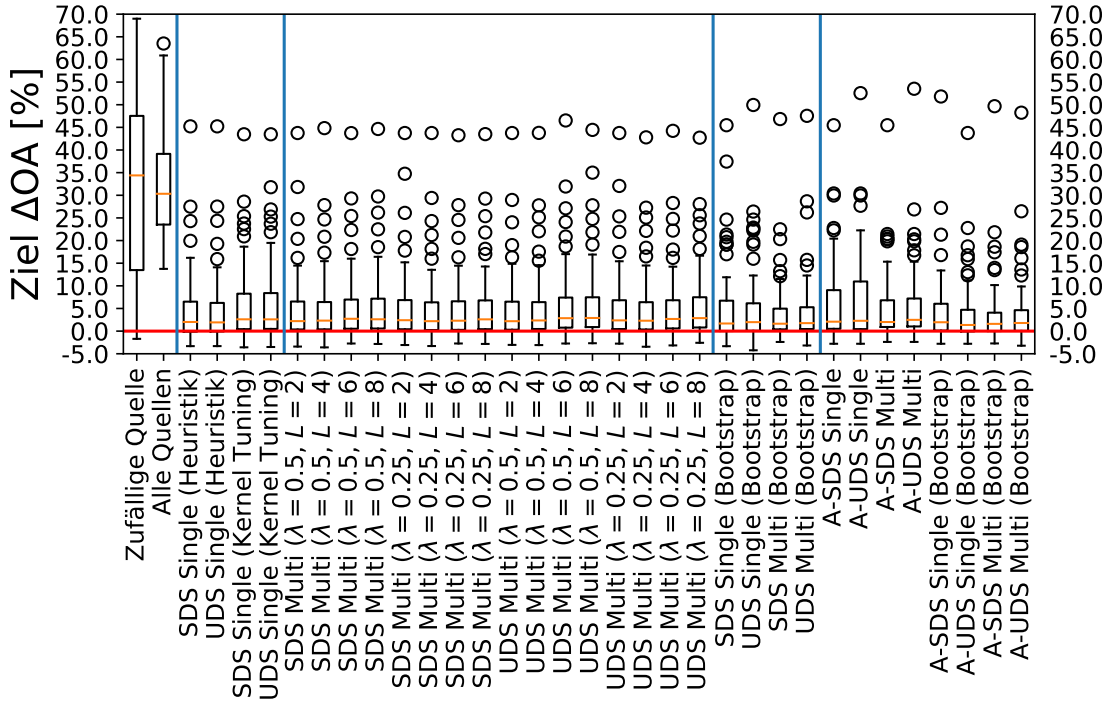


(d) ALL

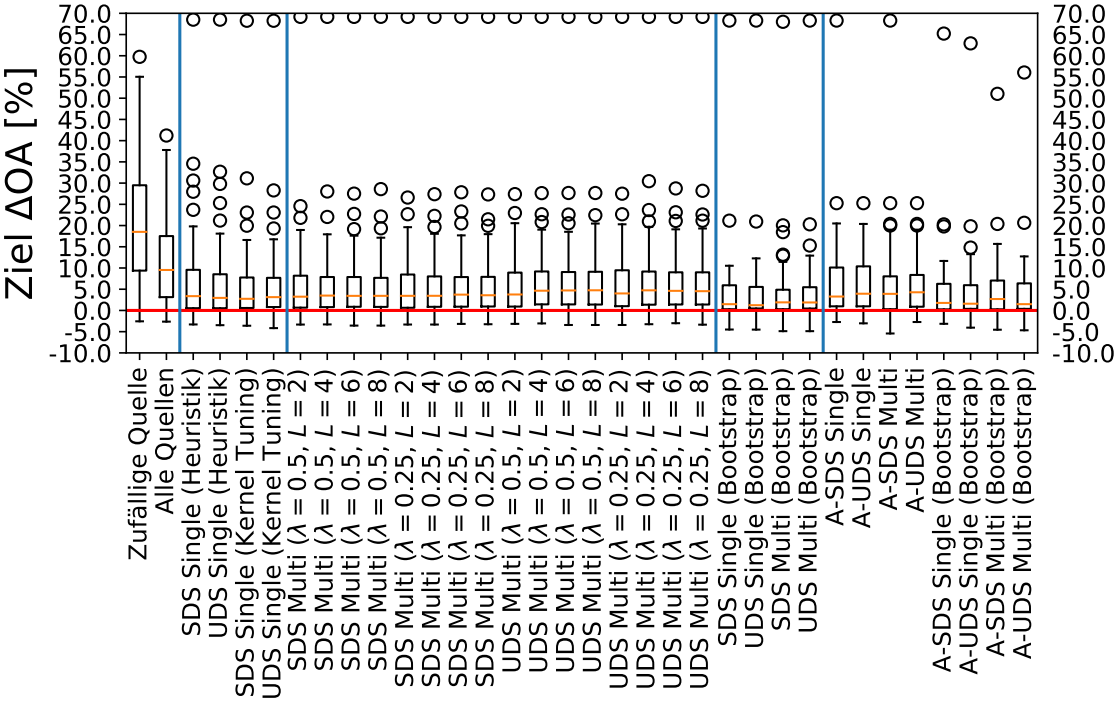
Abbildung 6.9: Box-Whisker-Plots für den *Nienburg* Datensatz.

Tabelle 6.10: Ergebnisse für das approximative Modell auf dem *Combined* Datensatz.

	EXP			COL			COL + LBP			ALL		
	∅ Ziel Δ_{OA} [%]	∅ Quellen Δ_{OA} [%]	∅ Prozentrang [%]	∅ Ziel Δ_{OA} [%]	∅ Quellen Δ_{OA} [%]	∅ Prozentrang [%]	∅ Ziel Δ_{OA} [%]	∅ Quellen Δ_{OA} [%]	∅ Prozentrang [%]	∅ Ziel Δ_{OA} [%]	∅ Quellen Δ_{OA} [%]	∅ Prozentrang [%]
Zufällige Quelle	31.2	31.4	51.2	20.6	20.7	52.6	22.7	20.5	50.6	33.0	31.0	50.3
Alle Quellen	32.2	32.4	55.0	11.8	11.9	74.8	14.8	12.6	72.7	26.7	24.7	64.0
SDS Single (Heuristik)	4.7	4.9	88.9	7.0	7.1	83.1	8.4	6.2	86.7	6.8	4.7	90.8
UDS Single (Heuristik)	4.5	4.7	89.2	6.4	6.4	85.2	7.9	5.7	87.4	6.5	4.5	91.4
SDS Single (Kernel Tuning)	5.9	6.1	86.9	5.9	6.0	85.3	7.2	5.1	89.0	5.4	3.4	93.1
UDS Single (Kernel Tuning)	6.0	6.2	86.9	5.8	5.9	85.6	7.4	5.2	88.6	5.7	3.7	92.9
SDS Multi ($\lambda = 0.5$, $L = 2$)	5.0	5.2	87.9	6.0	6.1	85.0	8.2	6.1	87.5	6.1	4.1	91.8
SDS Multi ($\lambda = 0.5$, $L = 4$)	4.9	5.1	87.5	5.9	6.1	84.9	8.3	6.1	86.9	7.5	5.5	89.3
SDS Multi ($\lambda = 0.5$, $L = 6$)	5.3	5.5	86.9	6.1	6.2	84.5	8.6	6.5	85.2	8.1	6.1	88.3
SDS Multi ($\lambda = 0.5$, $L = 8$)	5.5	5.7	86.4	6.0	6.2	84.7	8.5	6.3	86.0	8.5	6.5	87.6
SDS Multi ($\lambda = 0.25$, $L = 2$)	5.3	5.5	87.4	6.2	6.3	84.4	8.4	6.2	86.2	6.1	4.1	92.0
SDS Multi ($\lambda = 0.25$, $L = 4$)	4.8	5.0	88.0	6.1	6.2	84.5	8.5	6.3	86.0	7.1	5.1	90.1
SDS Multi ($\lambda = 0.25$, $L = 6$)	4.9	5.0	87.6	6.2	6.3	84.7	8.4	6.2	86.5	8.1	6.1	88.5
SDS Multi ($\lambda = 0.25$, $L = 8$)	5.2	5.4	87.0	6.2	6.3	84.5	8.6	6.4	85.2	8.1	6.1	88.2
UDS Multi ($\lambda = 0.5$, $L = 2$)	4.9	5.1	88.1	6.5	6.7	84.1	8.5	6.4	85.6	6.7	4.7	91.0
UDS Multi ($\lambda = 0.5$, $L = 4$)	5.0	5.1	87.6	7.0	7.1	83.2	9.2	7.0	83.2	7.6	5.6	89.4
UDS Multi ($\lambda = 0.5$, $L = 6$)	5.9	6.1	85.8	6.9	7.0	83.5	9.2	7.0	83.3	8.2	6.1	88.4
UDS Multi ($\lambda = 0.5$, $L = 8$)	6.1	6.3	85.3	6.9	7.0	83.5	9.2	7.0	83.1	8.5	6.4	87.9
UDS Multi ($\lambda = 0.25$, $L = 2$)	5.2	5.4	87.6	6.8	6.9	83.6	8.6	6.5	85.5	6.5	4.5	91.4
UDS Multi ($\lambda = 0.25$, $L = 4$)	4.9	5.1	87.7	7.1	7.3	82.7	9.1	6.9	83.4	7.4	5.3	89.8
UDS Multi ($\lambda = 0.25$, $L = 6$)	5.3	5.4	87.0	7.0	7.1	82.9	9.1	7.0	83.2	7.8	5.8	89.1
UDS Multi ($\lambda = 0.25$, $L = 8$)	5.8	6.0	85.9	6.9	7.1	83.1	9.2	7.0	83.0	8.2	6.2	88.5
SDS Single (Bootstrap)	5.4	5.6	87.5	3.9	4.0	89.9	5.2	3.0	**94.9**	4.3	2.3	94.3
UDS Single (Bootstrap)	5.4	5.6	87.6	**3.6**	**3.7**	**91.7**	5.2	3.1	94.4	3.9	1.9	95.7
SDS Multi (Bootstrap)	4.3	4.5	88.7	4.0	4.2	89.5	6.1	3.9	92.9	6.4	4.4	89.9
UDS Multi (Bootstrap)	4.4	4.6	88.7	4.0	4.1	91.2	6.2	4.1	92.5	6.5	4.5	90.9
A-SDS Single	6.7	6.9	86.0	6.7	6.8	83.3	6.9	4.7	90.0	4.3	2.3	94.5
A-UDS Single	7.0	7.2	86.0	5.8	6.0	84.7	5.7	3.5	91.7	4.6	2.6	94.0
A-SDS Multi	5.2	5.4	87.8	6.0	6.1	85.5	6.6	4.4	90.3	4.3	2.3	94.3
A-UDS Multi	5.7	5.9	87.0	5.3	5.5	85.9	5.6	3.4	91.6	4.4	2.4	94.2
A-SDS Single (Bootstrap)	4.3	4.5	88.7	4.3	4.4	89.3	5.6	3.4	93.9	3.7	1.7	95.4
A-UDS Single (Bootstrap)	**3.8**	**3.9**	**90.1**	4.2	4.3	89.9	5.4	3.2	94.3	3.3	1.3	96.4
A-SDS Multi (Bootstrap)	**3.8**	4.0	89.9	4.1	4.3	88.4	5.6	3.4	94.1	3.5	1.5	95.9
A-UDS Multi (Bootstrap)	4.0	4.2	89.2	3.9	4.0	89.3	**5.1**	**2.9**	94.7	**3.2**	**1.2**	**96.8**

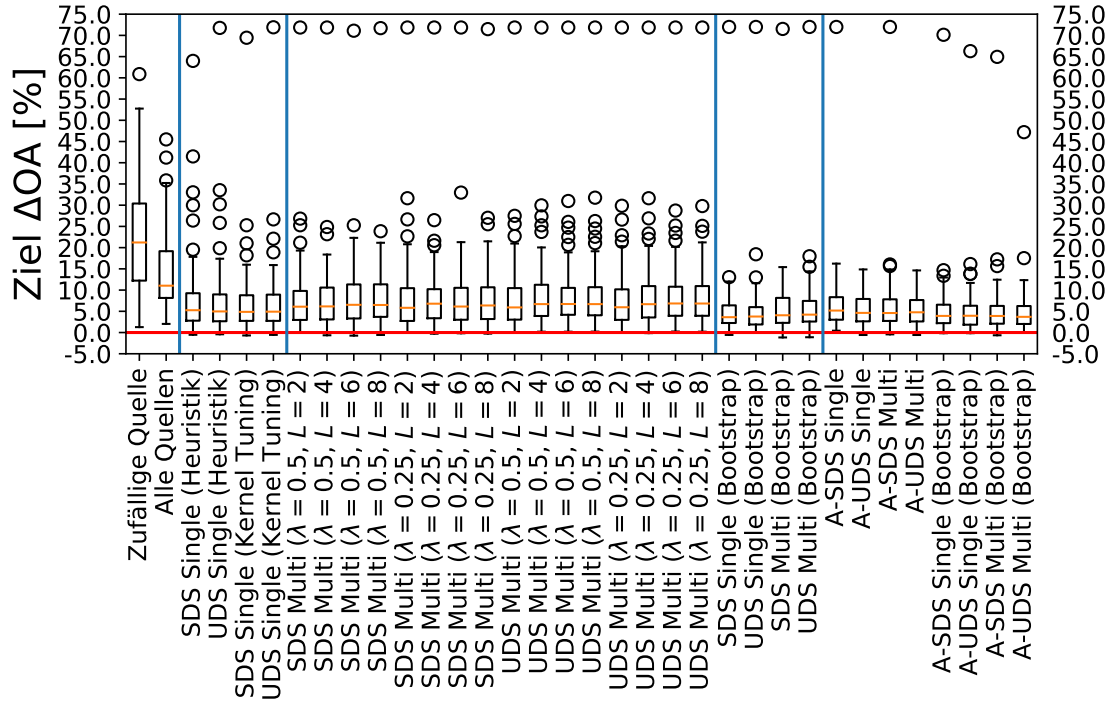


(a) EXP

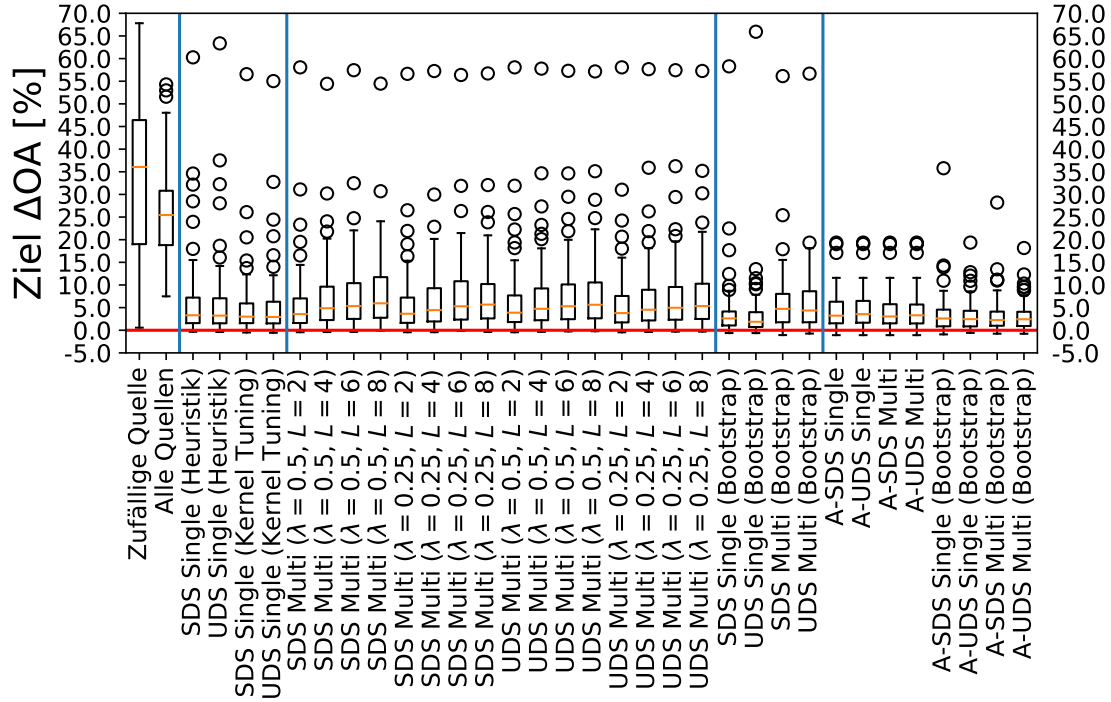


(b) COL

Abbildung 6.10: Box-Whisker-Plots für den *Combined* Datensatz.



(c) COL + LBP



(d) ALL

Abbildung 6.10: Box-Whisker-Plots für den *Combined* Datensatz.

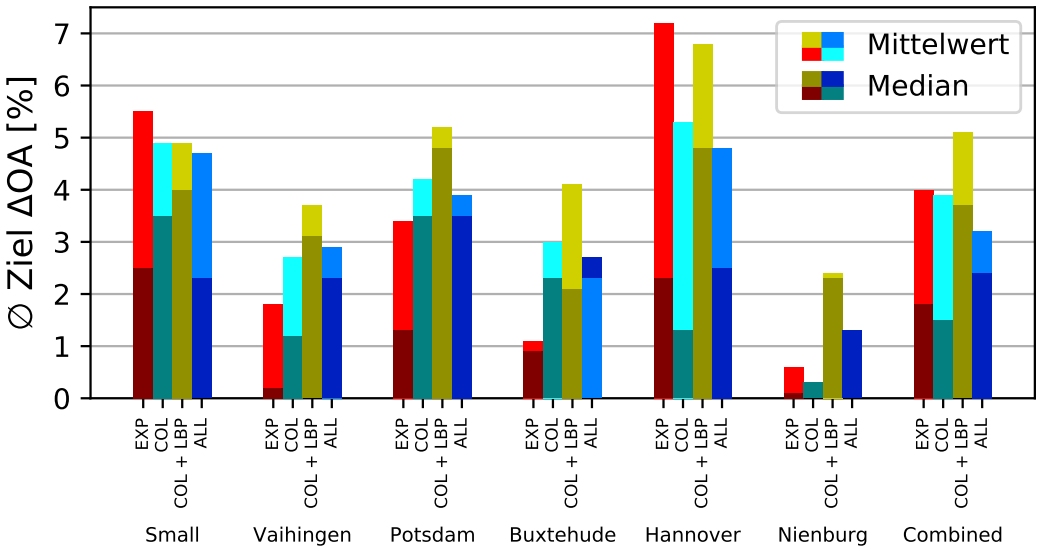


Abbildung 6.11: Arithmethische Mittelwerte und Mediane der Ziel Δ OA der Methode UDS *Multi (Bootstrap) für alle Kombinationen der untersuchten Datensätze und Merkmalsräume.*

6.4 Quellenranking

In diesem Abschnitt wird das Quellenrankung aus Kapitel 5 evaluiert. Es soll gezeigt werden, dass die bisher entwickelte Methode zur Quellenselektion nicht nur bei Vorhandensein einer großen und repräsentativen Menge von gelabelten Quellenkandidaten anwendbar ist. Die Evaluation wird zeigen, dass bereits eine kleine Teilmenge von Quellenkandidaten ausreichend ist und diese durch die Methode des *Kernel Herdings* aus ungelabelten Lernstichproben bestimmt werden kann. Somit kann der manuelle Arbeitsaufwand zur Erstellung der Labelinformation signifikant reduziert werden.

6.4.1 Experimenteller Aufbau

Im Kapitel 5 wird ein Szenario untersucht, nach dem ein bisher ungelabelter Datensatz bestehend aus N Bildpatches segmentiert werden soll. Um die für das Anlernen der Klassifikatoren benötigten gelabelten Lernstichproben zu erzeugen wurde bereits ein festes Budget eingeplant, welches nur ein manuelles Labeln von bis zu N_l Bildpatches vorsieht. Diese Bildpatches können anschließend als Quellenkandidaten einer Quellenselektion verwendet werden, um synthetisierte Quellen für die übrigen Patches zu erzeugen. Das Ziel in diesem Szenario ist somit die Reduktion des Arbeitsaufwandes ($\rightarrow$ kleines N_l) bei gleichzeitiger Maximierung der mittleren

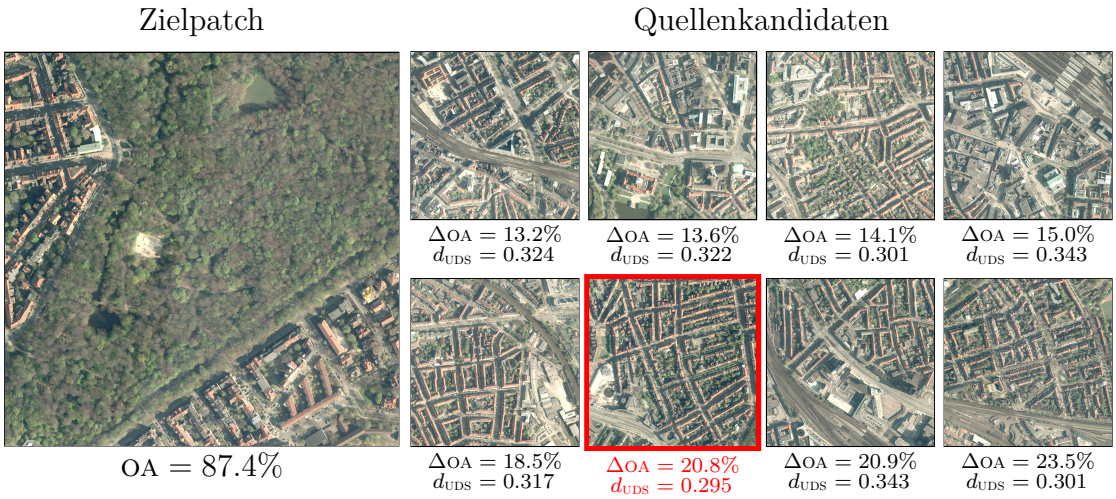
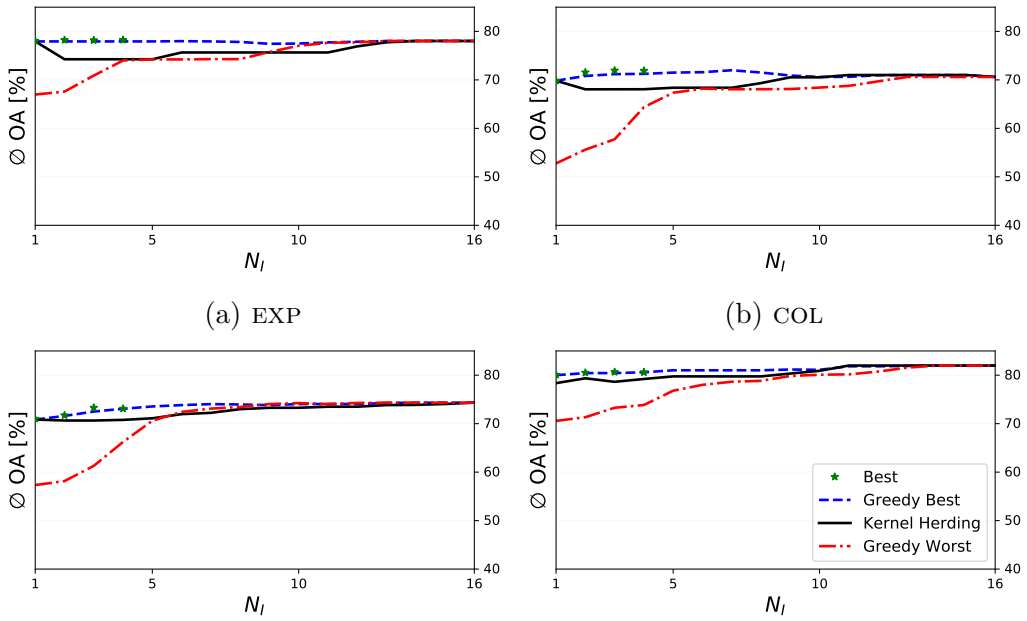


Abbildung 6.12: Beispiel einer fehlgeschlagenen Quellenselektion für einen Bildpatch des *Hannover* Datensatzes. Die Quellenkandidaten wurden anhand ihrer erzielten Klassifikationsgenauigkeit auf dem Zielpatch sortiert. Der rot umrahmte Bildpatch wurde von der Methode UDS *Single* ausgewählt. Die Domänendistanzen aller Bildpatches liegen innerhalb eines sehr schmalen Wertebereichs und zeigen keine erkennbare Korrelation zur ΔOA. Somit scheint die Domänendistanz nicht immer geeignet zu sein um solche Situationen zuverlässig erkennen zu können.

Klassifikationsgenauigkeit über den gesamten Datensatz (→ große Median OA).

Die Lernstichproben werden für alle Datensätze entsprechend dem Vorgehen in Abschnitt 6.3.1 erzeugt. Wir betrachten alle Bildpatches zunächst als ungelabelt. Durch das *Kernel Herding* Verfahren aus Kapitel 5 können diese Patches nun entsprechend ihrer Informativität sortiert werden. Um das Budget einzuhalten, werden daher gemäß dieser Sortierung die ersten N_l Bildpatches gelabelt und im Folgenden als Quellenkandidaten für **alle** Patches des Datensatzes verwendet. Zur Quellenselektion und somit zur Berechnung der Kernelmatrix des *Kernel Herding* Verfahrens wird die Methode UDS *Single (Bootstrap)* verwendet. Nach Ausführung der Quellenselektion können nun die erreichten Klassifikationsgenauigkeiten (OA) für jeden Bildpatch berechnet werden. In den Abbildungen 6.13 bis 6.18 sind mediangemittelten Resultate für alle Datensätze für variierende N_l dargestellt.

Um zu überprüfen, wie dicht die erzielten ∅ OA am Optimum liegen, werden zwei Vergleichskriterien herangezogen. Zum einen wird betrachtet, welche Werte bei vollständig gelabelten Stichproben erzielbar sind. Dies entspricht in den Diagrammen dem Resultat für $N_l = N$. Zum anderen soll verglichen werden, welche ∅ OA für eine optimale Teilmenge der Größe N_l erzielt werden kann. Für $N_l \leq 4$ (bzw. $N_l \leq 3$ auf dem *Combined* Datensatz) konnten diese Vergleichswerte durch eine erschöpfende



(a) EXP (b) COL

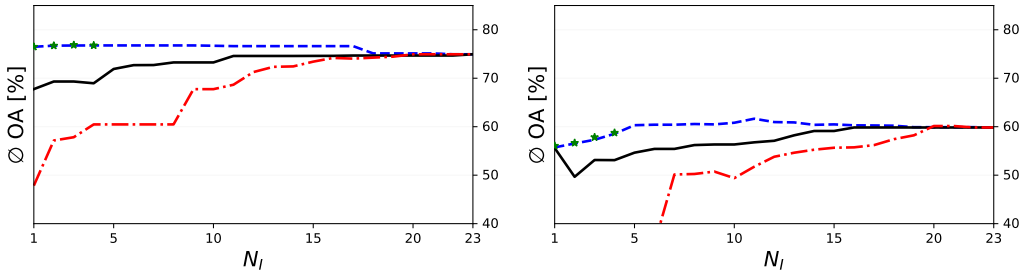
(c) COL + LBP (d) ALL

Abbildung 6.13: Ergebnisse des Quellenrankings für den *Vaihingen* Datensatz.

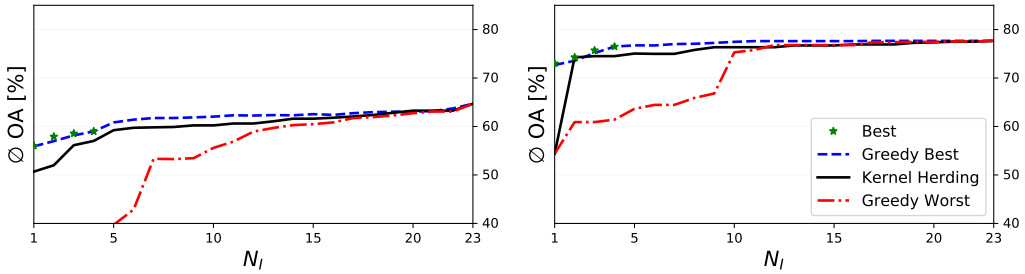
Suche ermittelt werden (*Best*). Aufgrund der kombinatorischen Explosion kann dieses Optimum jedoch nicht für alle N_l effizient berechnet werden. Daher werden die optimalen Teilmengen zusätzlich durch eine Greedy Optimierung (*Greedy Best*) bestimmt. Für $N_l = 1$ kann das globale Optimum immer effizient berechnet werden. Für $N_l > 1$ wird in einem Greedy Verfahren die Teilmenge der Größe $N_l - 1$ aus der vorherigen Iteration übernommen. Somit müssen in jeder Iteration nur $N - N_l + 1$ potentielle Quellenkandidaten untersucht werden. Aus den Diagrammen ergibt sich, dass das Greedy Verfahren für $N_l \leq 4$ ausreichend dicht am globalen Optimum liegt um als Substitutionsverfahren für eine erschöpfende Suche herzuhalten. In den Diagrammen werden weiterhin die Ergebnisse des Verfahrens *Greedy Worst* abgebildet. Hierbei wird durch eine Greedy Optimierung in jedem Iterationsschritt die Quelle zur Teilmenge der Quellenkandidaten hinzugefügt, welche den Wert ∅ OA minimiert.

6.4.2 Diskussion

Durch die erschöpfende Suche kann bestätigt werden, dass die Greedy Selektion als Referenzmethode geeignet ist. Bis auf wenige Ausnahmen wurden in den meisten Experimenten die selben $1 - 4$ Quellen gewählt. Dies bestätigt weiterhin, dass eine Approximation des optimalen Teilmengenproblems der Quellenselektion durch ein Quellenranking sinnvoll ist. Die Diagramme zeigen, dass die Methode *Kernel Herding*

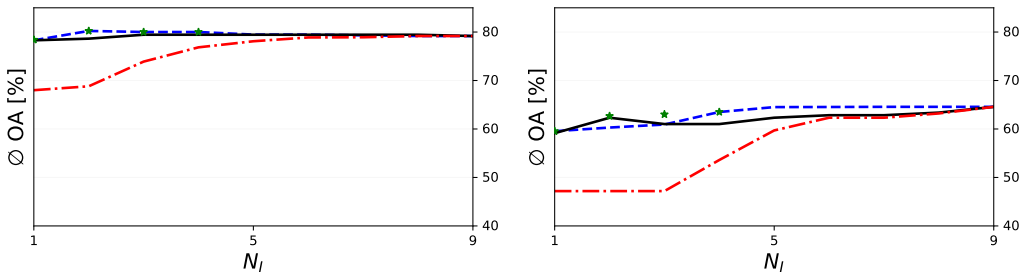


(a) EXP (b) COL

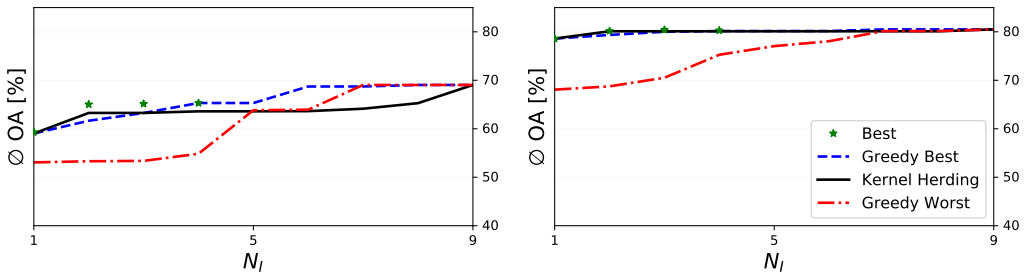


(c) COL + LBP (d) ALL

Abbildung 6.14: Ergebnisse des Quellenrankings für den *Potsdam* Datensatz.

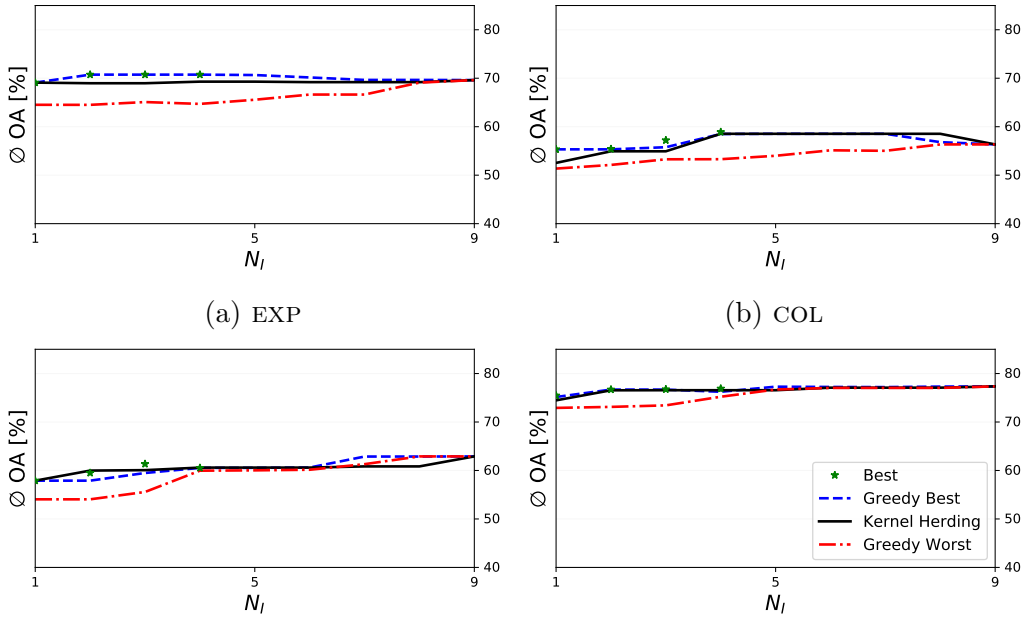


(a) EXP (b) COL



(c) COL + LBP (d) ALL

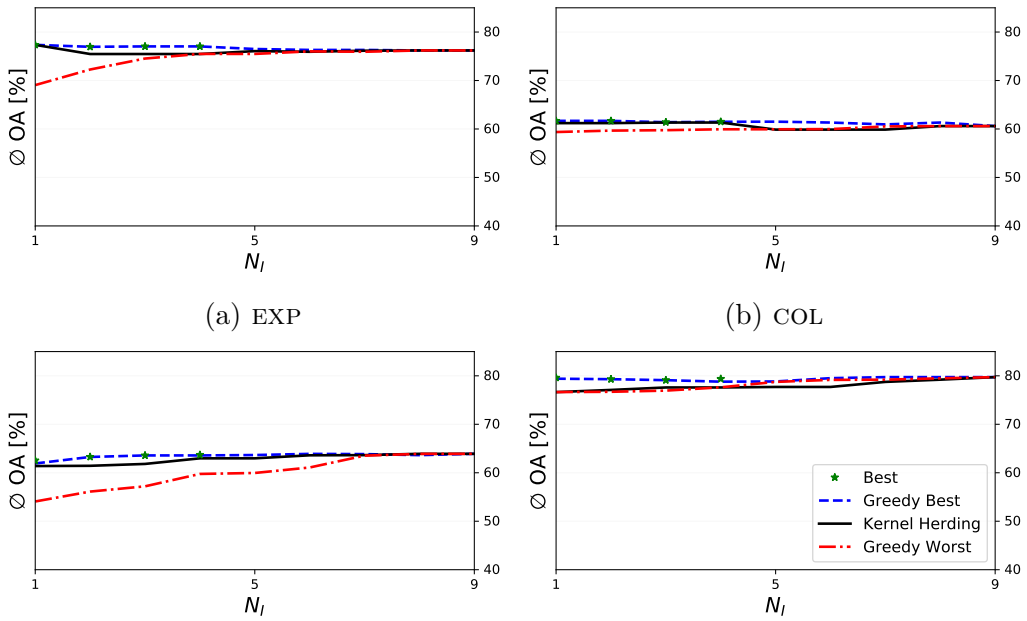
Abbildung 6.15: Ergebnisse des Quellenrankings für den *Buxtehude* Datensatz.



(a) EXP (b) COL

(c) COL + LBP (d) ALL

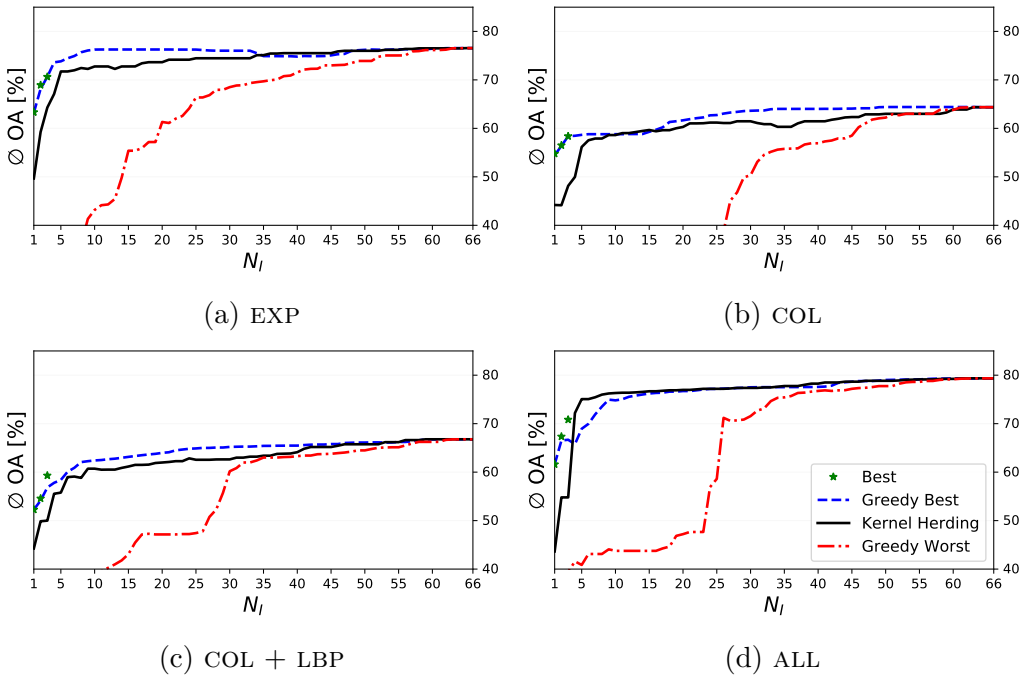
Abbildung 6.16: Ergebnisse des Quellenrankings für den *Hannover* Datensatz.



(a) EXP (b) COL

(c) COL + LBP (d) ALL

Abbildung 6.17: Ergebnisse des Quellenrankings für den *Nienburg* Datensatz.



(a) EXP (b) COL

(c) COL + LBP (d) ALL

Abbildung 6.18: Ergebnisse des Quellenrankings für den *Combined* Datensatz.

bis auf die Fälle *Vaihingen*/EXP, *Vaihingen*/COL und *Nienburg*/ALL dem Verlauf der Referenzmethode *Greedy Best* für $N_l \leq 5$ folgt. Für größere Teilmengen läuft die ∅ OA jedoch in eine Sättigung. Der zusätzliche Aufwand für das manuelle Labeln dieser Quellen schlägt sich nicht in einem messbaren Gewinn bei der Klassifikation nieder. Gleichzeitig kann beobachtet werden, dass die Kostenfunktion des *Kernel Herdings* auch in eine Sättigung läuft. Die unüberwachte Domänendistanz, welche dem *Kernel Herding* zugrunde liegt, kann in diesem Sättigungsbereich somit keine verlässliche Schätzung von subtilen Unterschieden von im Allgemeinen sehr ähnlichen Domänen liefern. Es bleibt zu klären, ob eine rein unüberwachte Domänendistanz überhaupt in der Lage sein kann, diese subtilen Unterschiede zu modellieren, oder ob zur Verbesserung der Ergebnisse notwendigerweise Information über Klassenlabels benötigt wird.

Da im Merkmalsraum ALL die besten Klassifikationsgenauigkeiten erzielt werden, beziehen sich alle folgenden Beobachtungen auf diesen. Um den maximalen Medianverlust auf 5% OA gegenüber einem vollständig gelabelten Datensatz zu beschränken, müssen durch Anwendung des *Kernel Herdings* davon nur respektive 1 (*Vaihingen*), 2 (*Potsdam*), 1 (*Buxtehude*), 1 (*Hannover*), 1 (*Nienburg*) und 5 (*Combined*) Patches gelabelt werden. Dies ergibt eine potentielle Reduktion des Arbeitsaufwandes um $89 - 94\%$. Bei einem maximalen Medianverlust von 3% wären dies 4 (*Vaihingen*), 5 (*Potsdam*), 1 (*Buxtehude*), 1 (*Hannover*), 2 (*Nienburg*) und 11 (*Combined*) Patches, was einer Arbeitsreduktion von $75 - 89\%$ entspräche. Bei mittleren Verlusten von

$3-5\%$, welche für die hier untersuchte Aufgabenstellung der semantischen Segmentierung von Landbedeckungsklassen durchaus vertretbar sind, können somit durch die vorgestellte Methode des *Kernel Herding* mit der hier entwickelten unüberwachten Domänendistanz (UDS) konsistente und signifikante Arbeitseinsparungen von über 80% erzielt werden. Falls jedoch für eine Aufgabenstellung nur geringere Verluste vertretbar sind, und dementsprechend ein größeres Budget zur manuellen Erstellung von Lernstichproben einkalkuliert wurde, arbeitet diese Methode jedoch häufig in einem suboptimalen Sättigungsbereich ($N_l > 5$). Folgeuntersuchungen müssten für solche Aufgabenstellungen klären, ob der Sättigungsbereich des *Kernel Herdings* durch geschickte Augmentierung der ungelabelten Lernstichproben, z. B. durch Methoden des *aktiven Lernens* [130], reduziert werden kann.

Kapitel 7

Zusammenfassung

Diese Arbeit beschäftigte sich mit dem Problem der semantischen Segmentierung Luftbildern in Landbedeckungsklassen. In der semantischen Segmentierung wird jedem Pixel des Eingangsbildes eine der bekannten Klassen zugewiesen. Diese Zuweisung geschieht durch einen mathematischen Operator, welcher im maschinellen Lernen als Klassifikator bezeichnet wird. Der Klassifikator muss jedoch zunächst in einem Vorgang, welcher als überwachtes Lernen bezeichnet wird, aus einer gelabelten Lernstichprobe erzeugt werden. Die allgemeinen theoretischen Resultate der *empirischen Risikominimierung* garantieren dabei unter bestimmten Voraussetzungen, dass der so gelernte Klassifikator imstande ist, auch auf ungesehenen Daten mit einer hohen Wahrscheinlichkeit die korrekten Klassenlabels zu prädizieren. Die wichtigste Voraussetzung besagt allerdings, dass die Lern- und Teststichproben aus derselben Wahrscheinlichkeitsverteilung erzeugt wurden. Diese Annahme kann jedoch nur garantiert werden, falls die Lernstichprobe tatsächlich aus dem Eingangsbild erzeugt wurde. Jedes zu segmentierende Luftbild muss somit zumindest in Teilen manuell gelabelt werden, was mit einem hohen Aufwand verbunden ist.

In dieser Arbeit wurde ein Ansatz entwickelt und evaluiert, mit dem sich diese Einschränkung auflockern lässt. Das Ziel ist eine deutliche Reduktion des manuellen Aufwands, welcher dem überwachten Lernen inhärent ist. Hierzu wird neben dem Eingangsbild (=Zielbild) die Existenz von mehreren unterstützenden Datenquellen vorausgesetzt. Diese Datenquellen liegen in Form von bereits segmentierten Luftbildern vor, welche allerdings nicht notwendigerweise der Verteilung des Zielbildes folgen müssen. Solche Luftbilder könnten z. B. als Resultat von vorangegangenen Auswertungen entstanden sein. Der in dieser Arbeit entwickelte Ansatz der Quellenselektion selektiert aus diesen Quellenkandidaten denjenigen Datensatz, welcher dem Zielbild am *ähnlichsten* ist. Aus dem selektierten Datensatz kann somit ohne Mehraufwand eine Lernstichprobe und somit ein Klassifikator, erzeugt werden, welcher den Anforderungen der *empirischen Risikominimierung* genügt. Jedoch werden die Wahrscheinlichkeitsverteilungen, welche dem Ziel- und Quellenbild zugrunde liegen, im allgemeinen nicht exakt identisch sein. Daher müssen bei diesem sogenannten Transferlernen immer Verluste in Form einer reduzierten Klassifikationsgenauigkeit in Kauf genommen werden. Die Ähnlichkeit zwischen den Datensätzen wurde in Kapitel 3 daher über ein Distanzmaß definiert, welches explizit eine obere Abschätzung dieser Verluste minimiert. Auf Basis dieser Domänendistanz wurde anschließend ein Bayessches Modell der Quellenselektion aufgestellt. In Kapitel 4 wurde zu diesem Modell eine konvexe Approximation entwickelt, welches erst eine effiziente Anwendung der Quellenselektion auf großen Datensätzen und unter Auswahl von vielen Quellenkandidaten ermöglicht. Diese gesteigerte Effizienz schaffte die Grundlage für einige Erweiterungen des Modells, u. a. der Multiquellenselektion. Bei der Multiquellenselektion wird statt einer einzelnen Quelle eine gewichtete Kombination mehrerer Quellenkandidaten verwendet. Die Gewichte werden dabei so eingestellt, dass die Domänendistanz zwischen der Zieldomäne und der interpolierten Quellendomäne minimiert wird. Durch diese Interpolation wird der Raum der beschreibbaren Quellen

auf den Subraum erweitert, welcher zwischen den Quellenkandidaten aufgespannt wird und bietet der Quellenselektion somit mehr Freiheiten eine passende Quelle zu finden. In Kapitel 5 wurde anschließend ein Vorgehen entwickelt, welches eine Anwendung der Quellenselektion trotz Ermangelung von gelabelten Quellenkandidaten ermöglicht. Eine unüberwachte Variante der Domänendistanz berechnet hierzu die paarweisen Distanzen zwischen ungelabelten Datensätzen. Aus diesen Werten wurde eine Sortierung der Datensätze hergeleitet, welche diese nach ihrer Informativität ordnet. Somit müssen nur wenige der informativsten Datensätze manuell gelabelt werden um dennoch eine hohe mittlere Klassifikationsgenauigkeit über alle Datensätze zu erreichen. Die wesentlichen Beiträge dieser Arbeiten lassen sich wie folgt zusammenfassen:

Bayessches Quellenselektionsmodell: Konstruktion eines Bayesschen Quellenselektionsmodells durch Umformung der Ungleichung aus Formel (3.4) als ein Klassifikationsproblem unter 0-1 Kosten. Die Schätzung durch einen Mittelwertschätzer erfolgt durch Simulation einer hierfür konstruierten a-Posteriori Verteilung durch ein Markov-Chain-Monte-Carlo Verfahren.

Adaptiver MCMC Sampler: Entwicklung eines adaptiven Simulationsverfahren zur direkten Lösung des Bayesschen Quellenselektionsmodells für die Einzelquellenselektion. Diesem Verfahren konnte eine sehr schnelle Konvergenz, in der Regel nach $200 - 800$ Iterationen, nachgewiesen werden.

Approximatives Quellenselektionsmodell: Herleitung einer Approximation mit asymptotischer linearer Laufzeitkomplexität für die Einzel- und Multiquellenselektion. Die Approximation erfolgt durch Substitution der Total-Variation Distanz in der Domänendistanz durch die Maximum Mean Discrepancy, welche in geschlossener Form berechnet werden kann.

Boosting basierte Multiquellenselektion: Erweiterung eines effizienten Verfahrens zur Berechnung von baryzentrischen Koordinaten in euklidischen Räumen auf allgemeine statistische Domänendistanzen. Diese Methode kann als ein nicht-adaptives Boostingverfahren interpretiert werden.

Keine kritischen Parameter: Die Parameter des approximativen Quellenselektionsmodells sind unkritisch und müssen nicht an den jeweiligen Datensatz angepasst werden. Die Ausnahme hierzu stellt die Kernelbandbreite σ dar, welche einen wesentlichen Einfluss auf die Diskriminativität der MMD hat. Jedoch konnte ein zuverlässiges Verfahren zur automatischen Optimierung dieses Parameters vorgestellt werden.

Optimierter Speicherverbrauch: Durch Ausnutzung von Asymmetrien im Quellenselektionsproblem konnte der Speicherverbrauch erheblich reduziert werden.

Die neue Methode besitzt einen konstanten Speicherverbrauch bezüglich der Größe der Quellenlernstichproben.

Selektion von Quellenkandidaten: Die Quellenselektion setzt das Vorhandensein von gelabelten Quellenkandidaten voraus. Durch Anwendung einer modifizierten unüberwachten Domänendistanz kann aus ungelabelten Datensätzen automatisch eine nahezu optimale Teilmenge von Quellenkandidaten selektiert werden.

Das entwickelte Quellenselektionsverfahren wurde auf fünf separaten Datensätzen getestet, welche über unterschiedlichen deutschen Städten aufgenommen wurden. Ein weiterer Datensatz (*Combined*) wurde aus deren Vereinigung erzeugt. Aufgrund der größeren Variabilität der Bildpatches enthalten die Quellenkandidaten immer einen signifikanten Anteil von Patches, aus denen ein stark negativer Transfer erwartet werden muss. Hierdurch sollte sichergestellt werden, dass Schwächen im Quellenselektionsmodell spätestens hier zum Vorschein treten. Weiterhin wurden alle Experimente in vier unterschiedlichen Merkmalsräumen durchgeführt. Die Evaluation der Quellenselektion startete mit dem Bayesschen Modell zur Einzelquellenselektion. Auf einer reduzierten Version des *Combined* Datensatzes konnte dieses Modell in ca. 50% aller Fälle die optimale Quelle finden. Der mittlere Verlust in Klassifikationsgenauigkeit, verglichen mit einem überwachten Lernverfahren, welchem die vollständigen Labelinformationen des Zielbildes zur Verfügung stehen, konnte mit dieser Methode in allen Experimenten auf $5 - 6\,\%$ eingeschränkt werden. Das approximative Modell wurde daraufhin ebenfalls für die Einzelquellenselektion getestet, schnitt allerdings deutlich schlechter ab. Die zusätzliche Anwendung einer Bootstrap Aggregierung konnte diese Ergebnisse jedoch deutlich verbessern, so dass für diese Variante gezeigt werden konnte, dass sie dem Bayesschen Modell überlegen ist. Das approximative Modell ist aufgrund seiner linearen asymptotischen Laufzeitkomplexität in der Lage, sehr große Datensätze zu verarbeiten. In den Experimenten konnte so durch eine GPU basierte Implementierung auf Lernstichproben mit mehr als 15000 Lernbeispielen pro Quelle und 65 Quellenkandidaten die Quellenselektion pro Zielpatch in weniger als 10 Sekunden auf einem handelsüblichen Personalcomputer (CPU: Intel Core i5, GPU: NDVIDIA GTX 1060) durchgeführt werden.

Weiterhin wurde die boosting-basierte Multiquellenselektion untersucht. Diese war zuverlässig in der Lage, Quellen zu generieren, welche eine geringere Domänendistanz zur Zieldomäne aufweisen als die Einzelquellenselektion. Dennoch konnten in Verbindung mit der Bootstrap Aggregierung nur minimale Verbesserungen bezüglich der Verluste in Klassifikationsgenauigkeit erzielt werden. Für diese Verbesserungen konnte keine statistische Signifikanz gezeigt werden.

Zum Abschluss wurde eine neue Domänendistanz auf Basis einer asymmetrischen Variante der MMD evaluiert. Diese asymmetrische Domänendistanz wird durch eine stochastische Approximation berechnet und besitzt die Eigenschaft, lediglich eine konstante Anzahl an Lernbeispielen einer Quelle zu betrachten. Der Wert dieser

Konstante kann direkt aus der gewünschten Approximationsgüte hergeleitet werden. In den Experimenten konnte den Speicherverbrauch pro Quelle somit von 1.7 MB auf 69 KB reduziert werden, was in realen Anwendungen die Verarbeitung von deutlich mehr Quellenkandidaten ermöglicht. Die asymmetrische Domänendistanz schneidet weiterhin bezüglich des Verlusts in Klassifikationsgenauigkeit vergleichbar mit der symmetrischen Variante ab.

Als finale Methode wurde daher eine Multiquellenselektion auf Basis der AMMD mit aktivierter Bootstrap Aggregierung untersucht. Diese Methode konnte den Verlust in Klassifikationsgenauigkeit über alle Datensätze und Merkmalsräume gemittelt mit 3.9% in Grenzen halten. Die erzielten Ergebnisse variieren zwar über die untersuchten Datensätze und Merkmalsräume, jedoch konnte immer ein Medianverlust von unter 5% eingehalten werden. Der größte untersuchte Merkmalsraum erzielte auf allen Datensätzen mit ca. 80% die beste Klassifikationsgenauigkeit und konnte gleichzeitig den geringsten Medianverlust, typischerweise unter 3%, erzielen. Auf den meisten Bildpatches konnten sehr gute Resultate erzielt werden. Große Verluste beschränkten sich pro Datensatz auf sehr wenige Bildpatches. Eine Analyse der Ausreißer ergab, dass hiervon ca. $8 - 10\%$ der Bildpatches betroffen waren. Diese Ausreißer konnten jedoch bisher ohne zusätzliche Labelinformation nicht während der Quellenselektion erkannt werden.

Als zweites Szenario wurde das Quellenranking zur effizienten Erzeugung von gelabelten Lernstichproben evaluiert. Es konnte gezeigt werden, dass in den untersuchten Datensätzen nur sehr kleine Teilmengen der Bildpatches manuell gelabelt werden müssen um durch Anwendung der Quellenselektion sehr gute mittlere Klassifikationsgenauigkeiten über den gesamten Datensatz zu erreichen. Die hier vorgestellte Methode des Quellenrankings durch das *Kernel Herding* Verfahren kommt auf diesen Datensätzen sehr nahe an die theoretisch erreichbaren Resultate heran. Bei Vorgabe eines maximalen Medianverlusts von 3% in Klassifikationsgenauigkeit können somit Arbeitseinsparungen von $75 - 89\%$ gegenüber einer vollständigen manuellen Segmentierung des gesamten Datensatzes erzielt werden. Diese Methode ermöglicht daher die Erzeugung von hochqualitativen Lernstichproben mit einem deutlich reduzierten Arbeitsaufwand.

In dieser Arbeit wurde ein Verfahren vorgestellt, welches Transferlernen durch die Methodik der Quellenselektion implementiert. Es konnte experimentell gezeigt werden, dass eine solche Quellenselektion im Mittel hervorragende Ergebnisse über unterschiedliche Datensätze und Merkmalsräume erzielen kann. Die Wahl des Merkmalsraums hat sich jedoch als kritischer Parameter erwiesen. In Zukunft könnte daher eine Erweiterung des präsentierten Verfahrens das Ziel verfolgen, während der Quellenselektion automatisch einen optimierten Merkmalsraum zu erzeugen. Zum Anlernen von optimierten Merkmalsräumen haben sich jüngst faltende neuronale Netzwerke besonders hervorgetan. Daher müsste überprüft werden, ob sich diese unterschiedlichen Ansätze vereinen lassen. Weiterhin hatte sich gezeigt, dass die mittleren Resultate teilweise durch vereinzelte Ausreißer bedeutsam verschlechtert

werden. Diese Ausreißer ließen sich auf zwei Faktoren zurückführen. Zum Einen muss unter den Quellenkandidaten keine Quelle existieren, von welcher ein sinnvoller Transfer auf die Zieldomäne möglich ist. Andererseits beinhaltet die Abschätzung, auf welcher die verwendete Domänendistanz aus Gleichung (3.4) beruht, eine inhärente Unschärfe in Form des Parameters γ. Für den ersten Fall müsste sich in zukünftigen Untersuchungen zeigen lassen, ob diese Ausreißer nur auf Basis von ungelabelten Stichproben detektiert werden können. Für den zweiten Fall folgt aus der Theorie, dass diese Ausreißer nur unter Verwendung von zusätzlichen gelabelten Daten aus der Zielverteilung erkennbar sein dürften. Somit könnten hier Methoden des aktiven Lernens eventuell hilfreich sein, diese Fälle mit einem geringen manuellen Mehraufwand zu erkennen. Abschließend konnten mit der neu entwickelten Methode des Quellenrankings durchaus Parallelen zu aktiven Lernverfahren geschaffen werden. Aktives Lernen verfolgt ebenfalls das Ziel eine deutlich kompaktere Lernstichprobe frei von Redundanz zu erstellen um somit den manuellen Arbeitsaufwand zu reduzieren. Es könnte somit berechtigtes Interesse darin bestehen, die Vorteile beider Methoden zu kombinieren.

Appendix A

Beweise

A.1 Beziehung zwischen den linearen Kosten und der MMD

Im Folgenden soll eine Brücke zwischen dem auf der TV Distanz basierten Quellenselektionsmodell aus Kapitel 3 sowie dem MMD basierten Modell aus Kapitel 4 geschlagen werden. Zunächst gilt, dass der TV Distanz ein Bayessches Klassifikationsmodell unter 0-1 Kosten zugrunde liegt. Die Resultate der nächsten Abschnitte werden zeigen, dass die Kosten einer optimalen linearen Trennebene unter Verwendung von linearen Kosten äquivalent zur Definition $d_{\mathrm{MMD}}(T,S) = \|\mu_T - \mu_S\|$ der MMD ist. Dies bedeutet, dass sich sowohl die TV Distanz als auch die MMD als Bayessche Entscheidungsmodelle interpretieren lassen.

In den ersten beiden Abschnitten sollen zunächst einige Vereinfachungen erarbeitet werden.Daraufhin folgt das Hauptresultat dieses Abschnitts. Die Beweise der Äquivalenz für Zweiklassen- und Multiklassenprobleme werden in zwei aufeinanderfolgenden Sektionen separat geführt.

A.1.1 Bias Terme

Für lineare Klassifikationsmodelle mit einer linearen Kostenfunktion gilt:

$$\begin{aligned} L &= \sum_{i=1}^{N} l(\boldsymbol{x}_i, y_i, \boldsymbol{\varphi}, \varphi_0) \\ &= \sum_{i=1}^{N} y_i \cdot (\langle \boldsymbol{x}_i, \boldsymbol{\varphi} \rangle + \varphi_0) \\ &= \varphi_0 \cdot (N_+ - N_-) + \sum_{i=1}^{N} y_i \cdot \langle \boldsymbol{x}_i, \boldsymbol{\varphi} \rangle \\ &= B + \sum_{i=1}^{N} y_i \cdot \langle \boldsymbol{x}_i, \boldsymbol{\varphi} \rangle \,. \end{aligned} \tag{A.1}$$

Bei Verwendung der linearen Kostenfunktion lassen sich daher die Bias Terme zu einer einzigen Konstante B zusammenfassen. Der Wert von B hängt dabei lediglich von den Stichprobengrößen N_+ und N_- ab. Diese Größen geben an, wie viele Lernbeispiele der Klasse $y = +1$, bzw. der Klasse $y = -1$ angehören. Für balancierte Lernstichproben mit $N_+ = N_-$ gilt daher immer $B = 0$. Die Bias Terme besitzen daher in linearen Kostenmodellen keinen Einfluss auf die Güte des gelernten Klassifikators. Zur Vereinfachung aller folgenden Rechnungen können daher die Bias Terme oBdA auf 0 gesetzt werden.

A.1.2 Unbalancierte Klassifikationsprobleme

Die Kosten des Klassifikationsmodells $\boldsymbol{\varphi}$ auf einer Lernstichprobe können bei Verwendung von linearen Kosten auf die Schwerpunkte beider Klassen im Merkmalsraum vereinfacht werden. Es gilt:

$$\begin{aligned} L &= \sum_{i=1}^{N} l(\boldsymbol{x}_i, y_i, \boldsymbol{\varphi}) \\ &= \sum_{i=1}^{N} y_i \cdot \langle \boldsymbol{x}_i, \boldsymbol{\varphi} \rangle \\ &= \sum_{i=1}^{N_+} \langle \boldsymbol{x}_i^+, \boldsymbol{\varphi} \rangle - \sum_{i=1}^{N_-} \langle \boldsymbol{x}_i^-, \boldsymbol{\varphi} \rangle \\ &= N_+ \cdot \langle \boldsymbol{\mu}^+, \boldsymbol{\varphi} \rangle - N_- \cdot \langle \boldsymbol{\mu}^-, \boldsymbol{\varphi} \rangle \\ &= N_+ \left[\langle \boldsymbol{\mu}^+, \boldsymbol{\varphi} \rangle - \frac{N_-}{N_+} \cdot \langle \boldsymbol{\mu}^-, \boldsymbol{\varphi} \rangle \right] . \end{aligned} \tag{A.2}$$

Weiterhin gilt oBdA $N_+ \geq N_-$. Für balancierte Klassifikationsprobleme ($N_+ = N_-$) ergibt sich somit eine geometrische Interpretation der Klassifikationskosten. Aus den Lernstichproben ergeben sich direkt die Klassenschwerpunkte $\boldsymbol{\mu}^+$ und $\boldsymbol{\mu}^-$. Die Gesamtkosten ergeben sich nach Umformung zu $N_+ \left[\langle \boldsymbol{\mu}^+, \boldsymbol{\varphi} \rangle - \langle \boldsymbol{\mu}^-, \boldsymbol{\varphi} \rangle \right] = N_+ \langle \boldsymbol{\mu}^+ - \boldsymbol{\mu}^-, \boldsymbol{\varphi} \rangle$. Die Kosten sind somit proportional zur Länge des auf $\boldsymbol{\varphi}$ projizierten Differenzvektors zwischen den Klassenschwerpunkten. Bei Verwendung der a-Priori Verteilung (3.26) gilt weiterhin $\|\boldsymbol{\varphi}\| = 1$. Die Klassifikationskosten hängen somit für balancierte Probleme nur von der Distanz der beiden Klassen sowie der Entscheidungsfunktion ab. In unbalancierten Problemen wird jedoch der Schwerpunkt $\boldsymbol{\mu}^-$ um den Faktor $\frac{N_-}{N_+}$ skaliert. Die Kosten hängen somit nicht nur von der relativen Lage der beiden Klassen untereinander ab, sondern zusätzlich von der Position des Koordinatenursprungs. Da dieser willkürlich gewählt ist, können die Klassifikationskosten für unbalancierte Probleme zu beliebigen Werten hin manipuliert werden. Daraus ergibt sich, dass für Klassifikationsmodelle mit einem linearen Kostenmodell nur balancierte Probleme sinnvoll modelliert werden können.

A.1.3 Äquivalenz für Zweiklassenprobleme

Die Ergebnisse des vorherigen Abschnitts können direkt aufgegriffen werden. Die Klassifikationskosten für balancierte Zweiklassenprobleme ergeben sich zu

$$L = N_+ \langle \boldsymbol{\mu}^+ - \boldsymbol{\mu}^-, \boldsymbol{\varphi} \rangle . \tag{A.3}$$

Der optimale Klassifikator $\boldsymbol{\varphi}$ kann in dieser Darstellung leicht aus geometrischen Überlegungen hergeleitet werden. Ziel ist die Minimierung der Kosten L. Dies wird

erreicht, indem $\boldsymbol{\varphi}$ in Gegenrichtung zu $\boldsymbol{\mu}^+ - \boldsymbol{\mu}^-$ positioniert wird. Allerdings sorgt dessen a-Priori Verteilung dafür, dass $\boldsymbol{\varphi}$ wieder auf Einheitslänge skaliert wird. Für die Kosten gilt somit

$$\begin{aligned} L &= N_+ \frac{\langle \boldsymbol{\mu}^+ - \boldsymbol{\mu}^-, \boldsymbol{\mu}^- - \boldsymbol{\mu}^+ \rangle}{\|\boldsymbol{\mu}^- - \boldsymbol{\mu}^+\|} \\ &= -N_+ \frac{\|\boldsymbol{\mu}^+ - \boldsymbol{\mu}^-\|^2}{\|\boldsymbol{\mu}^- - \boldsymbol{\mu}^+\|} \\ &= -N_+ \left\|\boldsymbol{\mu}^+ - \boldsymbol{\mu}^-\right\|. \end{aligned} \tag{A.4}$$

Die Gesamtkosten eines optimalen Klassifikators, basierend auf einer linearen Kostenfunktion, entsprechen somit bis auf einen konstanten Faktor $-N_+$ der MMD.

A.1.4 Erweiterung auf Multiklassenprobleme

Aus der Literatur sind derzeit keine Multiklassen Erweiterungen der MMD bekannt. Es konnte jedoch im vorangegangenen Abschnitt gezeigt werden, dass der MMD ein Klassifikationsmodell zugrunde liegt. Im Abschnitt 2.2.2 wurden für allgemeine Klassifikationsverfahren mehrere Ansätze präsentiert, um Multiklassenprobleme in eine Folge von Zweiklassenprobleme zu zerlegen. Diese Ansätze sollen nun wieder aufgegriffen werden um die Definition einer Multiklassen MMD zu ermöglichen.

One-vs-Rest

Für den OVR Ansatz gilt, dass das Gesamtproblem in $|\mathcal{K}|$ Teilprobleme mit den Klassenaufteilungen $\mathcal{K}^j = \{+1 = j, -1 = \mathcal{K} \setminus j\}$ zerlegt wurde. Unter dem Klassenlabel -1 werden somit $|\mathcal{K}| - 1$ Klassen zusammengefasst. Dieses Vorgehen sorgt notwendigerweise für unbalancierte Teilprobleme. Die Verwendung der OVR Strategie ist in Verbindung mit Klassifikationsmodellen mit linearen Kosten daher aufgrund der Ergebnisse aus Abschnitt A.1.2 nicht anzuraten.

One-vs-One

Im OVO Ansatz werden zunächst alle Klassenpaare separat verglichen. Die ursprüngliche Methode kombinierte die Ergebnisse der Teilprobleme durch eine Mehrheitsentscheidung. Um die Diskriminativität der MMD zu erhalten, soll hier ein etwas anderer Ansatz verfolgt werden. Die OVO Multiklassen MMD wird für diese Arbeit zu

$$\sum_{c_i \in \mathcal{K}} \max_{\substack{c_j \in \mathcal{K} \\ c_j \neq c_i}} d_{\mathrm{MMD}}(\mathrm{Tr}^i, \mathrm{Tr}^j) \tag{A.5}$$

definiert. Für jede Klasse fließt somit nur die Unterscheidbarkeit von der ähnlichsten anderen Klasse in das Ergebnis ein. Die dazugehörigen Kosten der Lernstichprobe

werden definiert durch

$$L = -\sum_{c_i \in \mathcal{K}} N_i \cdot \max_{\substack{c_j \in \mathcal{K} \\ c_j \neq c_i}} d_{\mathrm{MMD}}(\mathrm{Tr}^i, \mathrm{Tr}^j) \tag{A.6}$$

und die erwarteten Kosten pro Lernbeispiel dementsprechend zu

$$\epsilon_{\mathrm{MMD}}(\mathrm{T}) = -\frac{1}{|\mathcal{K}|} \sum_{c_i \in \mathcal{K}} \max_{\substack{c_j \in \mathcal{K} \\ c_j \neq c_i}} d_{\mathrm{MMD}}(\mathrm{Tr}^i, \mathrm{Tr}^j)\,. \tag{A.7}$$

A.2 Eigenschaften des Kernelselektionsproblems

Dieser Abschnitt beleuchtet das Optimierungsproblem zur Einstellung des Kernelparameters σ noch einmal genauer. Die optimale Kernelfunktion ergibt sich für den RBF-Kernel laut [122] zu

$$d^2_{\mathrm{MMD}_{\mathrm{opt}}}(\mathrm{T}_T, \mathrm{T}_S) = \max_{\sigma \in \mathbb{R}^+} d^2_{\mathrm{MMD}}(\mathrm{T}_T, \mathrm{T}_S)\,. \tag{siehe 4.19}$$

Wird dieses Kriterium direkt in die Formel (4.7) zur Berechnung der MMD mit linearer Laufzeitkomplexität eingesetzt, so ergibt sich ein Optimierungsproblem

$$\begin{aligned} d^2_{\mathrm{MMD}_{1,\mathrm{opt}}}(\mathrm{T}_T, \mathrm{T}_S) = \max_{\sigma \in \mathbb{R}^+} \frac{2}{M} \Big[& \sum_{i=1}^{M/2} k(\boldsymbol{x}_{T,2i}, \boldsymbol{x}_{T,2i-1}) \\ & + \sum_{i=1}^{M/2} k(\boldsymbol{x}_{S,2i}, \boldsymbol{x}_{S,2i-1}) - \sum_{i=1}^{M} k(\boldsymbol{x}_{T,i}, \boldsymbol{x}_{S,i}) \Big]\,. \end{aligned} \tag{A.8}$$

Zur Lösung dieses Problems wurde in Abschnitt 4.4 eine modifizierte Methode auf Basis des *Golden-Section-Search* Algorithmus vorgestellt. Der unmodifizierte GSS Algorithmus setzt jedoch eine streng unimodale Zielfunktion voraus. Unimodalität bedeutet in diesem Sinne, dass die Zielfunktion links ihres Maximums streng monoton wachsend ist und rechts davon streng monoton fallend. Diese Eigenschaften werden im Folgenden für das Optimierungsproblem (A.8) auf einem Teilintervall von σ gezeigt. Anschließend wird eine Modifikation des GSS Algorithmus motiviert, welche ein gültiges Optimum über dem gesamten positiven Intervall $\sigma \in \mathbb{R}^+$ ermöglicht.

Aus der Arbeit von Shestopaloff [118] ergeben sich wichtige Aussagen für allgemeine Funktionen der Form

$$S(x) = \sum_{i=1}^{N} C_j a_j^x \tag{A.9}$$

mit $x, C_j, a_j \in \mathbb{R}$. Zunächst wird gezeigt, dass die Zielfunktion aus Formel (A.8) in einer solchen Darstellung vorliegt. Es gilt für die Zielfunktion $f(\sigma)$:

$$f(\sigma) = \frac{2}{M}\left[\sum_{i=1}^{M/2} k(\boldsymbol{x}_{T,2i}, \boldsymbol{x}_{T,2i-1}) + \sum_{i=1}^{M/2} k(\boldsymbol{x}_{S,2i}, \boldsymbol{x}_{S,2i-1}) - \sum_{i=1}^{M} k(\boldsymbol{x}_{T,i}, \boldsymbol{x}_{S,i})\right] \tag{A.10}$$

$$= \frac{2}{M}\left[\sum_{i=1}^{M/2} e^{-\frac{1}{2}\frac{\left\|\boldsymbol{x}_{T,2i}-\boldsymbol{x}_{T,2i-1}\right\|^2}{\sigma^2}} + \sum_{i=1}^{M/2} e^{-\frac{1}{2}\frac{\left\|\boldsymbol{x}_{S,2i}-\boldsymbol{x}_{S,2i-1}\right\|^2}{\sigma^2}} - \sum_{i=1}^{M} e^{-\frac{1}{2}\frac{\left\|\boldsymbol{x}_{T,i}-\boldsymbol{x}_{S,i}\right\|^2}{\sigma^2}}\right] \tag{A.11}$$

$$= \sum_{j=1}^{2M} \frac{2s_j}{M} e^{-\frac{1}{2}\frac{\xi_j}{\sigma^2}} = \sum_{j=1}^{2M} \frac{2s_j}{M}\left(e^{-\frac{1}{2}\xi_j}\right)^{\sigma^{-2}} \Rightarrow f(x) = \sum_{j=1}^{N} C_j a_j^x \tag{A.12}$$

mit den Substitutionen

$$N = 2M \tag{A.13}$$

$$s_j = \begin{cases} +1 & \text{falls } j \leq M \\ -1 & \text{sonst} \end{cases} \tag{A.14}$$

$$C_j = \frac{2s_j}{M} \tag{A.15}$$

$$\xi_j = \begin{cases} \left\|\boldsymbol{x}_{T,2j} - \boldsymbol{x}_{T,2j-1}\right\|^2 & \text{falls } j \leq {}^{M}/_{2} \\ \left\|\boldsymbol{x}_{S,2(j-M/2)} - \boldsymbol{x}_{S,2(j-M/2)-1}\right\|^2 & \text{falls } {}^{M}/_{2} < j \leq M \\ \left\|\boldsymbol{x}_{T,j-M} - \boldsymbol{x}_{S,j-M}\right\|^2 & \text{sonst} \end{cases} \tag{A.16}$$

$$a_j = e^{-\frac{1}{2}\xi_j} \tag{A.17}$$

$$x = \sigma^{-2}\,. \tag{A.18}$$

Aus [118] ergeben sich daher für die Zielfunktion $f(x)$ folgende Eigenschaften: (1) $f(x)$ besitzt bis zu zwei Extremalstellen and den Positionen $x_{\max}$ und $x_{\min}$, (2) $f(x)$ besitzt bis zu zwei Nulldurchgänge und (3) $f(x)$ besitzt bis zu zwei Wendepunkte. Diese Funktion ist somit im Allgemeinen nicht unimodal. Allerdings folgen aus der geometrischen Interpretierbarkeit der MMD im Hilbertraum einige weitere Eigenschaften, welche hier genutzt werden sollen. Zunächst kann für beliebige ξ_j im Grenzfall $\lim_{x\to 0}(e^{-\frac{1}{2}\xi})^x = 1$ beobachtet werden. Weiterhin ist s_j für exakt die Hälfte der Terme negativ und für die andere Hälfte positiv. Somit gilt die Eigenschaft (4) $\lim_{x\to 0} f(x) = 0$. Da angenommen wird, dass sich die Lernbeispiele in allgemeiner Lage befinden, gilt insbesondere $\xi_j > 0$. Daraus folgt $\lim_{x\to\infty}(e^{-\frac{1}{2}\xi})^x = 0$ und somit die Eigenschaft (5) $\lim_{x\to\infty} f(x) = 0$. Aus den Eigenschaften (1), (4) und (5) folgt nun trivialerweise (6) $f(x_{\max}) > 0$ sowie (7) $f(x_{\min}) < 0$. Die bisher gezeigten Eigenschaften gelten für beliebige Lernstichproben. Eine zusätzlich benötigte Eigenschaft kann nur heuristisch angegeben werden. Für fast alle Datensätze soll gelten, dass die,

laut euklidischer Distanz, nächsten Lernbeispiele innerhalb der Ziel- oder Quellendomäne (Intradomänendistanz) dichter beieinander liegen als die nächsten Lernbeispiele zwischen ihnen (Interdomänendistanz). Dies bedeutet $\min_{1\leq i\leq M} \xi_i < \min_{M<j\leq 2M} \xi_j$. Dann wird die Zielfunktion für $x \to \infty$ von diesem einzigen Term mit dem Index i_{nearest} dominiert, d. h. es gilt

$$\frac{\left(e^{-\frac{1}{2}\xi_{i_{\text{nearest}}}}\right)^x}{\left(e^{-\frac{1}{2}\xi_j}\right)^x} = \left(e^{-\frac{1}{2}(\xi_{i_{\text{nearest}}}-\xi_j)}\right)^x \to \infty \text{ für } j \neq i_{\text{nearest}}, \xi_{i_{\text{nearest}}} < \xi_j, x \to \infty\,. \tag{A.19}$$

Somit muss die Zielfunktion für $x \to \infty$ positiv sein. Demnach gilt, dass falls $f(x)$ ein Minimum besitzt, so folgt $x_{\max} > x_{\min}$. Aus allen Aussagen lässt sich zusammen herleiten, dass die Zielfunktion im Intervall $x \in [x_{\min}, \infty]$ somit unimodal ist. Um den regulären GSS Algorithmus anwenden zu können, muss der Algorithmus 8 daher zu Beginn das Suchintervall $[L, R]$ linksseitig sukzessive solange verkleinern, bis $f(x_l) \geq 0$ gilt. Auf dem resultierenden Intervall wird anschließend mit dem regulären GSS Algorithmus fortgeführt.

A.3 Quellenstichprobengröße für die asymmetrische MMD

Die Kernoperation zur Berechnung der asymmetrischen MMD aus Abschnitt 4.5.2 erfordert die effiziente Suche des Maximums aus einer unsortierten Liste der Länge N. Zur Vereinfachung wurde eine stochastische Approximation vorgeschlagen, bei der statt des Maximums lediglich ein beliebiger Wert innerhalb des oberen q-Quantils der Wahrscheinlichkeitsverteilung, aus der diese Liste entstanden ist, ermittelt werden muss. Weiterhin muss diese Bedingung nur mit einer Wahrscheinlichkeit $1 - p$ erfüllt werden. D. h. mit einer Wahrscheinlichkeit p darf der gefundene Wert die geforderte Approximationsgüte verletzen und somit im unteren $(1 - q)$-Quantil der Liste liegen. Daraus folgt, dass zur Berechnung dieser Approximation nur $N_{\max} = \lceil \log_{1-q}(p) \rceil$ zufällige Elemente der Liste betrachtet werden müssen. Dies gilt unabhängig von der Länge der Liste sowie von der Verteilung aus der die Elemente der Liste erzeugt wurden. Diese Aussage wird im Folgenden bewiesen.

Beweis. Nehmen wir zunächst an, dass die Liste die Werte $X = (x_i)_{i=1}^N$ enthält und das die Variablen x_i statistisch unabhängig sind und aus einer unbekannten Verteilungsfunktion $p(x < s)$ gezogen wurden. Es sei $q = p(x \geq s)$ die Wahrscheinlichkeit, dass x größer oder gleich dem Wert s ist. Wir nehmen an, dass wir den Wert s so eingestellt haben, dass q dem gewünschten Quantil entspricht. Weiterhin sei $p = 1 - p(max_{x\in X}\, x \geq s)$ die Wahrscheinlichkeit, dass das größte Element von X

kleiner als s ist. D. h. es gilt:

$$q = p(x \geq s) \tag{A.20}$$

$$p = 1 - p(\max_{x \in X} x \geq s) = p(\forall x \in X \quad x < s) = (1-q)^N . \tag{A.21}$$

Dies bedeutet, dass die Liste X lediglich ein einziges Element enthalten muss welches oberen q-Quantil liegt. Für die Gegenwahrscheinlichkeit gilt dementsprechend, dass jedes einzelne der unabhängig gezogenen Elemente kleiner als s sein muss. Diese Formel liefert uns ein Kriterium zur Einstellung der Listengröße N. Die Variable N kann jedoch nur ganzzahlige Werte annehmen. Daher führen wir zusätzlich $p' \leq p$ ein um zu beschreiben, dass die resultierende Wahrscheinlichkeit der Verletzung unseres Approximationskriteriums auch kleiner als p sein darf. Somit gilt:

$$1 - p(\max_{x \in X} x \geq s) \geq p' \Leftrightarrow (1-q)^N \geq p' \Leftrightarrow N \geq \log_{1-q}(p') . \tag{A.22}$$

Der kleinste ganzzahlige Wert für N welcher diese Bedingung erfüllt ist somit $N_{\max} = \lceil \log_{1-q}(p) \rceil$. □

LITERATUR

[1] Ayan Acharya, Eduardo R Hruschka, Joydeep Ghosh und Sreangsu Acharyya. „Transfer learning with cluster ensembles“. In: *Proceedings of ICML Workshop on Unsupervised and Transfer Learning*. 2012, S. 123–132.

[2] Hirotugu Akaike. „A new look at the statistical model identification“. In: *IEEE transactions on automatic control* 19.6 (1974), S. 716–723.

[3] Rie Kubota Ando und Tong Zhang. „A framework for learning predictive structures from multiple tasks and unlabeled data“. In: *Journal of Machine Learning Research* 6.Nov (2005), S. 1817–1853.

[4] Andreas Argyriou, Theodoros Evgeniou und Massimiliano Pontil. „Multi-task feature learning“. In: *Advances in neural information processing systems*. 2007, S. 41–48.

[5] Andreas Argyriou, Theodoros Evgeniou und Massimiliano Pontil. „Multi-task feature learning“. In: *Advances in neural information processing systems*. Bd. 19. 2007, S. 41–48.

[6] David Arthur und Sergei Vassilvitskii. „k-means++: The advantages of careful seeding“. In: *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial und Applied Mathematics. 2007, S. 1027–1035.

[7] Biplab Banerjee, Francesca Bovolo, Avik Bhattacharya, Lorenzo Bruzzone, Subhasis Chaudhuri und Krishna Mohan Buddhiraju. „A novel graph-matching-based approach for domain adaptation in classification of remote sensing image pair“. In: *IEEE Transactions on Geoscience and Remote Sensing* 53.7 (2015), S. 4045–4062.

[8] Shai Ben-David, John Blitzer, Koby Crammer, Fernando Pereira u. a. „Analysis of representations for domain adaptation“. In: *Advances in neural information processing systems* 19 (2007), S. 137–144.

[9] Jose M Bernardo. „Reference posterior distributions for Bayesian inference“. In: *Journal of the Royal Statistical Society. Series B (Methodological)* (1979), S. 113–147.

[10] N Bernoulli und Richard Pulskamp. „Correspondence of Nicolas Bernoulli concerning the St. Petersburg Game“. In: *Letter to Pierre Raymond de Montmart* 1713 (2005).

[11] Steffen Bickel, Michael Brückner und Tobias Scheffer. „Discriminative learning for differing training and test distributions“. In: *Proceedings of the 24th international conference on Machine learning*. ACM. 2007, S. 81–88.

[12] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006. ISBN: 0387310738.

[13] John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira und Jennifer Wortman. „Learning bounds for domain adaptation“. In: *Advances in neural information processing systems*. 2008, S. 129–136.

[14] Leo Breiman. „Random forests“. In: *Machine learning* 45.1 (2001), S. 5–32.

[15] Stephen P Brooks und Andrew Gelman. „General methods for monitoring convergence of iterative simulations“. In: *Journal of computational and graphical statistics* 7.4 (1998), S. 434–455.

[16] Lorenzo Bruzzone und Mattia Marconcini. „Domain adaptation problems: A DASVM classification technique and a circular validation strategy“. In: *IEEE transactions on pattern analysis and machine intelligence* 32.5 (2010), S. 770–787.

[17] Lorenzo Bruzzone und Mattia Marconcini. „Toward the automatic updating of land-cover maps by a domain-adaptation SVM classifier and a circular validation strategy“. In: *IEEE Transactions on Geoscience and Remote Sensing* 47.4 (2009), S. 1108–1122.

[18] Brian D Bue, Erzsébet Merényi und Beáta Csathó. „An evaluation of class knowledge transfer from synthetic to real hyperspectral imagery“. In: *Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS), 2011 3rd Workshop on*. IEEE. 2011, S. 1–4.

[19] Rich Caruana. „Multitask learning“. In: *Learning to learn*. Springer, 1998, S. 95–133.

[20] Chih-Chung Chang und Chih-Jen Lin. „LIBSVM: a library for support vector machines“. In: *ACM transactions on intelligent systems and technology (TIST)* 2.3 (2011), S. 27.

[21] Olivier Chapelle, Bernhard Schölkopf und Alexander Zien. „Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]“. In: *IEEE Transactions on Neural Networks* 20.3 (2009), S. 542–542.

[22] C. Charalambous und A. Conn. „An Efficient Method to Solve the Minimax Problem Directly“. In: *SIAM Journal on Numerical Analysis* 15.1 (1978), S. 162–187. DOI: 10.1137/0715011. eprint: https://doi.org/10.1137/0715011. URL: https://doi.org/10.1137/0715011.

[23] Rita Chattopadhyay, Qian Sun, Wei Fan, Ian Davidson, Sethuraman Panchanathan und Jieping Ye. „Multisource Domain Adaptation and Its Application to Early Detection of Fatigue“. In: *ACM Trans. Knowl. Discov. Data* 6.4 (Dez. 2012), 18:1–18:26. ISSN: 1556-4681. DOI: 10.1145/2382577.2382582. URL: http://doi.acm.org/10.1145/2382577.2382582.

[24] Ming-Hui Chen und Bruce W Schmeiser. „General hit-and-run Monte Carlo sampling for evaluating multidimensional integrals“. In: *Operations Research Letters* 19.4 (1996), S. 161–169.

[25] Yutian Chen, Max Welling und Alex Smola. „Super-samples from kernel herding“. In: *arXiv preprint arXiv:1203.3472* (2012).

[26] Li Cheng und Sinno Jialin Pan. „Semi-supervised domain adaptation on manifolds“. In: *IEEE transactions on neural networks and learning systems* 25.12 (2014), S. 2240–2249.

[27] Jacob Cohen. „Weighted kappa: Nominal scale agreement provision for scaled disagreement or partial credit.“ In: *Psychological bulletin* 70.4 (1968), S. 213.

[28] John B Copas. „Regression, prediction and shrinkage“. In: *Journal of the Royal Statistical Society. Series B (Methodological)* (1983), S. 311–354.

[29] George Cybenko. „Approximation by superpositions of a sigmoidal function“. In: *Mathematics of control, signals and systems* 2.4 (1989), S. 303–314.

[30] Wenyuan Dai, Qiang Yang, Gui-Rong Xue und Yong Yu. „Boosting for transfer learning“. In: *Proceedings of the 24th international conference on Machine learning*. ACM. 2007, S. 193–200.

[31] Wenyuan Dai, Qiang Yang, Gui-Rong Xue und Yong Yu. „Self-taught clustering“. In: *Proceedings of the 25th international conference on Machine learning*. ACM. 2008, S. 200–207.

[32] Hal Daume. „Frustratingly Easy Domain Adaptation“. In: *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*. Prague, Czech Republic: Association for Computational Linguistics, Juni 2007, S. 256–263. URL: http://www.aclweb.org/anthology-new/P/P07/P07-1033.bib.

[33] Bruno De Finetti. „La prévision: ses lois logiques, ses sources subjectives“. In: *Annales de l'institut Henri Poincaré*. Bd. 7. 1. 1937, S. 1–68.

[34] Kresimir Delac und Mislav Grgic. „A survey of biometric recognition methods“. In: *Electronics in Marine, 2004. Proceedings Elmar 2004. 46th International Symposium*. IEEE. 2004, S. 184–193.

[35] Petros Dellaportas und Gareth O Roberts. „An introduction to MCMC“. In: *Spatial statistics and computational methods*. Springer, 2003, S. 1–41.

[36] Janez Demšar. „Statistical comparisons of classifiers over multiple data sets“. In: *Journal of Machine learning research* 7.Jan (2006), S. 1–30.

[37] Inderjit S Dhillon, Yuqiang Guan und Brian Kulis. „Kernel k-means: spectral clustering and normalized cuts“. In: *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2004, S. 551–556.

[38] Wei Ding, Tomasz F Stepinski, Yang Mu, Lourenco Bandeira, Ricardo Ricardo, Youxi Wu, Zhenyu Lu, Tianyu Cao und Xindong Wu. „Subkilometer crater discovery with boosting and transfer learning“. In: *ACM Transactions on Intelligent Systems and Technology (TIST)* 2.4 (2011), S. 1–22.

[39] Eric Eaton u. a. „Set-based boosting for instance-level transfer“. In: *2009 IEEE International Conference on Data Mining Workshops*. IEEE. 2009, S. 422–428.

[40] Eric Eaton, Terran Lane u. a. „Modeling transfer relationships between learning tasks for improved inductive transfer“. In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer. 2008, S. 317–332.

[41] Bradley Efron. „Bootstrap methods: another look at the jackknife“. In: *Breakthroughs in Statistics*. Springer, 1992, S. 569–593.

[42] Willliam Feller. *An introduction to probability theory and its applications*. Bd. 2. John Wiley & Sons, 2008.

[43] César Ferri, José Hernández-Orallo und R Modroiu. „An experimental comparison of performance measures for classification“. In: *Pattern Recognition Letters* 30.1 (2009), S. 27–38.

[44] Imola K Fodor. *A survey of dimension reduction techniques*. 2002.

[45] Yoav Freund, Robert Schapire und N Abe. „A short introduction to boosting“. In: *Journal-Japanese Society For Artificial Intelligence* 14.771-780 (1999), S. 1612.

[46] Mikel Galar, Alberto Fernández, Edurne Barrenechea, Humberto Bustince und Francisco Herrera. „An overview of ensemble methods for binary classifiers in multi-class problems: Experimental study on one-vs-one and one-vs-all schemes“. In: *Pattern Recognition* 44.8 (2011), S. 1761–1776. ISSN: 0031-3203. DOI: http://dx.doi.org/10.1016/j.patcog.2011.01.017. URL: http://www.sciencedirect.com/science/article/pii/S0031320311000458.

[47] Thomas Gärtner, Peter Flach und Stefan Wrobel. „On graph kernels: Hardness results and efficient alternatives“. In: *Learning Theory and Kernel Machines*. Springer, 2003, S. 129–143.

[48] Stuart Geman und Donald Geman. „Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images“. In: *IEEE Transactions on pattern analysis and machine intelligence* 6 (1984), S. 721–741.

[49] Samuel J Gershman und David M Blei. „A tutorial on Bayesian nonparametric models“. In: *Journal of Mathematical Psychology* 56.1 (2012), S. 1–12.

[50] Gauthier Gidel, Tony Jebara und Simon Lacoste-Julien. „Frank-wolfe algorithms for saddle point problems“. In: *arXiv preprint arXiv:1610.07797* (2016).

[51] Walter R Gilks, Gareth O Roberts und Edward I George. „Adaptive direction sampling“. In: *The statistician* (1994), S. 179–189.

[52] Mark Girolami und Simon Rogers. „Variational Bayesian multinomial probit regression with Gaussian process priors“. In: *Neural Computation* 18.8 (2006), S. 1790–1817.

[53] Raghuraman Gopalan, Ruonan Li und Rama Chellappa. „Domain adaptation for object recognition: An unsupervised approach“. In: *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE. 2011, S. 999–1006.

[54] Arthur Gretton, Dino Sejdinovic, Heiko Strathmann, Sivaraman Balakrishnan, Massimiliano Pontil, Kenji Fukumizu und Bharath K Sriperumbudur. „Optimal kernel choice for large-scale two-sample tests“. In: *Advances in neural information processing systems*. 2012, S. 1205–1213.

[55] Isabelle Guyon und André Elisseeff. „An introduction to variable and feature selection“. In: *Journal of machine learning research* 3.Mar (2003), S. 1157–1182.

[56] HA Haenssle u. a. „Automated Skin Cancer-Screening with contact free Remote-Dermatoscopy“. In: *JOURNAL DER DEUTSCHEN DERMATOLOGISCHEN GESELLSCHAFT* 13 (Apr. 2015), S. 164–165.

[57] David R Hardoon, Sandor Szedmak und John Shawe-Taylor. „Canonical correlation analysis: An overview with application to learning methods“. In: *Neural computation* 16.12 (2004), S. 2639–2664.

[58] W Keith Hastings. „Monte Carlo sampling methods using Markov chains and their applications“. In: *Biometrika* 57.1 (1970), S. 97–109.

[59] Marti A. Hearst, Susan T Dumais, Edgar Osman, John Platt und Bernhard Scholkopf. „Support vector machines“. In: *IEEE Intelligent Systems and their Applications* 13.4 (1998), S. 18–28.

[60] Petra Helmholz u. a. „Semi-automatic Quality Control of Topographic Data Sets“. In: *Photogrammetric Engineering & Remote Sensing* 78.9 (Sep. 2012), S. 959–972.

[61] Ralf Herbrich, Thore Graepel und Colin Campbell. „Bayes point machines“. In: *The Journal of Machine Learning Research* 1 (2001), S. 245–279.

[62] Tim Hesterberg, David S Moore, Shaun Monaghan, Ashley Clipson und Rachel Epstein. *The Practice of Business Statistics Companion Chapter 18: Bootstrap Methods and Permutation Tests*. New York (NY), USA: WH Freeman & Co., 2003.

[63] Rob High. „The era of cognitive systems: An inside look at ibm watson and how it works“. In: *IBM Corporation, Redbooks* (2012).

[64] David W Hosmer und Stanley Lemeshow. „Introduction to the logistic regression model“. In: *Applied Logistic Regression, Second Edition* (2000), S. 1–30.

[65] Chih-Wei Hsu und Chih-Jen Lin. „A comparison of methods for multiclass support vector machines“. In: *IEEE transactions on Neural Networks* 13.2 (2002), S. 415–425.

[66] Guang-Bin Huang, Qin-Yu Zhu und Chee-Kheong Siew. „Extreme learning machine: a new learning scheme of feedforward neural networks“. In: *Neural Networks, 2004. Proceedings. 2004 IEEE International Joint Conference on*. Bd. 2. IEEE. 2004, S. 985–990.

[67] Gordon Hughes. „On the mean accuracy of statistical pattern recognizers“. In: *IEEE transactions on information theory* 14.1 (1968), S. 55–63.

[68] Nathalie Japkowicz und Mohak Shah. *Evaluating learning algorithms: a classification perspective*. Cambridge University Press, 2011.

[69] Ahmad Khodayari-Rostamabad, James P Reilly, Natalia K Nikolova, James R Hare und Sabir Pasha. „Machine learning techniques for the analysis of magnetic flux leakage images in pipeline inspection“. In: *IEEE Transactions on magnetics* 45.8 (2009), S. 3073–3084.

[70] Alex Krizhevsky, Ilya Sutskever und Geoffrey E Hinton. „ImageNet Classification with Deep Convolutional Neural Networks“. In: *Advances in Neural Information Processing Systems 25*. Hrsg. von F. Pereira, C. J. C. Burges, L. Bottou und K. Q. Weinberger. Curran Associates, Inc., 2012, S. 1097–1105. URL: `http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf`.

[71] Neil D. Lawrence und John C. Platt. „Learning to Learn with the Informative Vector Machine“. In: *Proceedings of the Twenty-first International Conference on Machine Learning*. ICML '04. Banff, Alberta, Canada: ACM, 2004, S. 65–. ISBN: 1-58113-838-5. DOI: 10.1145/1015330.1015382. URL: http://doi.acm.org/10.1145/1015330.1015382.

[72] Lucien Le Cam. *Asymptotic methods in statistical decision theory*. Springer Science & Business Media, 2012.

[73] Erich Leo Lehmann und George Casella. *Theory of point estimation*. Springer Science & Business Media, 2006.

[74] Jose M Leiva-Murillo, Luis Gómez-Chova und Gustavo Camps-Valls. „Multitask remote sensing data classification“. In: *IEEE transactions on geoscience and remote sensing* 51.1 (2013), S. 151–161.

[75] Dennis V Lindley. *The Bayesian Approach to Statistics*. Techn. Ber. DTIC Document, 1980.

[76] Huma Lodhi, Craig Saunders, John Shawe-Taylor, Nello Cristianini und Chris Watkins. „Text classification using string kernels“. In: *Journal of Machine Learning Research* 2.Feb (2002), S. 419–444.

[77] Wei-Yin Loh. „Classification and regression trees“. In: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 1.1 (2011), S. 14–23.

[78] László Lovász. „Hit-and-run mixes fast“. In: *Mathematical Programming* 86.3 (1999), S. 443–461.

[79] László Lovász und Santosh Vempala. „Hit-and-run from a corner“. In: *SIAM Journal on Computing* 35.4 (2006), S. 985–1005.

[80] J. MacQueen. „Some methods for classification and analysis of multivariate observations“. In: *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*. Berkeley, Calif.: University of California Press, 1967, S. 281–297. URL: http://projecteuclid.org/euclid.bsmsp/1200512992.

[81] Christopher D Manning, Hinrich Schütze u. a. *Foundations of statistical natural language processing*. Bd. 999. MIT Press, 1999.

[82] Giona Matasci, Michele Volpi, Mikhail Kanevski, Lorenzo Bruzzone und Devis Tuia. „Semisupervised transfer component analysis for domain adaptation in remote sensing image classification“. In: *IEEE Transactions on Geoscience and Remote Sensing* 53.7 (2015), S. 3550–3564.

[83] Nicholas Metropolis, Arianna W Rosenbluth, Marshall N Rosenbluth, Augusta H Teller und Edward Teller. „Equation of state calculations by fast computing machines“. In: *The journal of chemical physics* 21.6 (1953), S. 1087–1092.

[84] Holger Meuel, Luis Angerstein, Roberto Henschel, Bodo Rosenhahn und Jörn Ostermann. „Moving Object Tracking for Aerial Video Coding using Linear Motion Prediction and Block Matching“. In: *Proceedings of 32nd Picture Coding Symposium (PCS)*. 2016, S. 1–5. DOI: 10.1109/PCS.2016.7906333.

[85] Marvin Minsky und Seymour Papert. *Perceptrons*. MIT press, 1988.

[86] James E Mosimann. „On the compound multinomial distribution, the multivariate β-distribution, and correlations among proportions“. In: *Biometrika* 49.1/2 (1962), S. 65–82.

[87] Radford M Neal. *Bayesian learning for neural networks*. Bd. 118. Springer Science & Business Media, 2012.

[88] Tan Nguyen und Scott Sanner. „Algorithms for Direct 0-1 Loss Optimization in Binary Classification.“ In: *ICML (3)*. 2013, S. 1085–1093.

[89] Alexandru Niculescu-mizil und Rich Caruana. „Obtaining Calibrated Probabilities from Boosting“. In: *In: Proc. 21st Conference on Uncertainty in Artificial Intelligence (UAI '05), AUAI Press*. AUAI Press, 2005, S. 413–420.

[90] Jorge Nocedal und Stephen J Wright. *Sequential quadratic programming*. Springer, 2006.

[91] Timo Ojala, Matti Pietikainen und Topi Maenpaa. „Multiresolution gray-scale and rotation invariant texture classification with local binary patterns“. In: *IEEE Transactions on pattern analysis and machine intelligence* 24.7 (2002), S. 971–987.

[92] Sinno Jialin Pan und Qiang Yang. „A survey on transfer learning“. In: *IEEE Transactions on knowledge and data engineering* 22.10 (2010), S. 1345–1359.

[93] M. Papageorgiou, M. Leibold und M. Buss. *Optimierung: Statische, dynamische, stochastische Verfahren für die Anwendung*. Springer Berlin Heidelberg, 2015. ISBN: 9783662469361. URL: https://books.google.de/books?id=FcvjCgAAQBAJ.

[94] Lance Parsons, Ehtesham Haque und Huan Liu. „Subspace clustering for high dimensional data: a review“. In: *ACM SIGKDD Explorations Newsletter* 6.1 (2004), S. 90–105.

[95] Andreas Paul, Franz Rottensteiner und Christian Heipke. „Iterative reweighted instance transfer for domain adaptation“. In: *XXIII ISPRS Congress, Commission III 3 (2016), Nr. 3*. Bd. 3. 3. Göttingen: Copernicus GmbH. 2016, S. 339–346.

[96] PEGWiki. *Convex hull trick*. [Online; accessed 21-April-2016]. 2016. URL: http://wcipeg.com/wiki/index.php?title=Convex_hull_trick&oldid=1900.

[97] Ulrike Pestel-Schiller, Karsten Vogt, Jörn Ostermann und Wolfgang Groß. „Impact of Hyperspectral Image Coding on Subpixel Detection“. In: *Proceedings of 32nd Picture Coding Symposium*. 2016.

[98] John Platt u. a. „Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods“. In: *Advances in large margin classifiers* 10.3 (1999), S. 61–74.

[99] Martyn Plummer u. a. „JAGS: A program for analysis of Bayesian graphical models using Gibbs sampling“. In: *Proceedings of the 3rd international workshop on distributed statistical computing*. Bd. 124. Vienna. 2003, S. 125.

[100] Nicholas G Polson, James G Scott u. a. „On the half-Cauchy prior for a global scale parameter“. In: *Bayesian Analysis* 7.4 (2012), S. 887–902.

[101] David Price, Stefan Knerr, Léon Personnaz, Gérard Dreyfus u. a. „Pairwise neural network classifiers with probabilistic outputs“. In: *Neural Information Processing Systems*. Bd. 7. 1994, S. 1109–1116.

[102] J Ross Quinlan. *C4. 5: programs for machine learning*. Elsevier, 2014.

[103] Sashank J Reddi, Aaditya Ramdas, Barnabás Póczos, Aarti Singh und Larry Wasserman. „Kernel MMD, the median heuristic and distance correlation in high dimensions“. In: *stat* 1050 (2014), S. 9.

[104] Mahdi Rezaei und Reinhard Klette. *Computer Vision for Driver Assistance: Simultaneous Traffic and Driver Monitoring*. Bd. 45. Springer, 2017.

[105] Cox T Richard. *The Algebra of Probable Inference*. 1961.

[106] Christian Robert. *The Bayesian choice: from decision-theoretic foundations to computational implementation*. Springer Science & Business Media, 2007.

[107] Lior Rokach und Oded Maimon. „Clustering methods“. In: *Data mining and knowledge discovery handbook*. Springer, 2005, S. 321–352.

[108] Lorenzo Rosasco, Ernesto De Vito, Andrea Caponnetto, Michele Piana und Alessandro Verri. „Are loss functions all the same?“ In: *Neural Computation* 16.5 (2004), S. 1063–1076.

[109] J. W. Rouse Jr., R. H. Haas, J. A. Schell und D. W. Deering. „Monitoring Vegetation Systems in the Great Plains with Erts“. In: *NASA Special Publication* 351 (1974), S. 309.

[110] Pál Ruján. „A fast method for calculating the perceptron with maximal stability“. In: *Journal de Physique I* 3.2 (1993), S. 277–290.

[111] Jürgen Schmidhuber. „Deep learning in neural networks: An overview“. In: *Neural Networks* 61 (2015), S. 85–117.

[112] Bernhard Schölkopf, Ralf Herbrich und Alex J Smola. „A generalized representer theorem“. In: *International Conference on Computational Learning Theory*. Springer. 2001, S. 416–426.

[113] Bernhard Schölkopf und Alexander J Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2002.

[114] Bernhard Schölkopf, Alexander Smola und Klaus-Robert Müller. „Kernel principal component analysis“. In: *International Conference on Artificial Neural Networks*. Springer. 1997, S. 583–588.

[115] Gideon Schwarz u. a. „Estimating the dimension of a model“. In: *The annals of statistics* 6.2 (1978), S. 461–464.

[116] Glenn Shafer u. a. *A mathematical theory of evidence*. Bd. 1. Princeton university press Princeton, 1976.

[117] Glenn Shafer und Vladimir Vovk. *The origins and legacy of Kolmogorov's Grundbegriffe*. 2005. URL: `http://www.probabilityandfinance.com/articles/04.pdf` (besucht am 22. 05. 2014).

[118] Yuri K Shestopaloff. *Sums of exponential functions and their new fundamental properties, with applications to natural phenomena*. Akvy Press, 2008.

[119] Irina Shevtsova. „On the absolute constants in the Berry-Esseen type inequalities for identically distributed summands“. In: *arXiv preprint arXiv:1111.6554* (2011).

[120] Alex J Smola und Bernhard Schölkopf. „A tutorial on support vector regression“. In: *Statistics and computing* 14.3 (2004), S. 199–222.

[121] Alex J Smola, Bernhard Schölkopf und Klaus-Robert Müller. „The connection between regularization operators and support vector kernels“. In: *Neural networks* 11.4 (1998), S. 637–649.

[122] Bharath K Sriperumbudur, Kenji Fukumizu, Arthur Gretton, Gert RG Lanckriet und Bernhard Schölkopf. „Kernel Choice and Classifiability for RKHS Embeddings of Probability Distributions.“ In: *NIPS*. 2009, S. 1750–1758.

[123] Bharath K Sriperumbudur, Kenji Fukumizu, Arthur Gretton, Bernhard Schölkopf, Gert RG Lanckriet u. a. „On the empirical estimation of integral probability metrics“. In: *Electronic Journal of Statistics* 6 (2012), S. 1550–1599.

[124] Radu Stoica, Xavier Descombes und Josiane Zerubia. „A Gibbs point process for road extraction from remotely sensed images“. In: *International Journal of Computer Vision* 57.2 (2004), S. 121–136.

[125] Masashi Sugiyama, Matthias Krauledat und Klaus-Robert MÃžller. „Covariate shift adaptation by importance weighted cross validation“. In: *Journal of Machine Learning Research* 8.May (2007), S. 985–1005.

[126] Masashi Sugiyama, Shinichi Nakajima, Hisashi Kashima, Paul V Buenau und Motoaki Kawanabe. „Direct importance estimation with model selection and its application to covariate shift adaptation“. In: *Advances in neural information processing systems*. 2008, S. 1433–1440.

[127] Johan AK Suykens und Joos Vandewalle. „Least squares support vector machine classifiers“. In: *Neural processing letters* 9.3 (1999), S. 293–300.

[128] David MJ Tax und Robert PW Duin. „Support vector domain description“. In: *Pattern recognition letters* 20.11 (1999), S. 1191–1199.

[129] Radu Timofte und Luc Van Gool. „Iterative nearest neighbors for classification and dimensionality reduction“. In: 2012, S. 2456–2463.

[130] D. Tuia, M. Volpi, L. Copa, M. Kanevski und J. Munoz-Mari. „A Survey of Active Learning Algorithms for Supervised Remote Sensing Image Classification“. In: *IEEE Journal of Selected Topics in Signal Processing* 5.3 (Juni 2011), S. 606–617. ISSN: 1932-4553. DOI: 10.1109/JSTSP.2011.2139193.

[131] Devis Tuia, Jordi Munoz-Mari, Luis Gomez-Chova und Jesus Malo. „Graph matching for adaptation in remote sensing“. In: *IEEE Transactions on Geoscience and Remote Sensing* 51.1 (2013), S. 329–341.

[132] John W Tukey. *Exploratory data analysis*. Bd. 2. Reading, Mass., 1977.

[133] Vladimir Naumovich Vapnik und Vlamimir Vapnik. *Statistical learning theory*. Bd. 1. Wiley New York, 1998.

[134] Luminita A Vese und Stanley J Osher. „Modeling textures with total variation minimization and oscillating patterns in image processing“. In: *Journal of scientific computing* 19.1-3 (2003), S. 553–572.

[135] Tim Vieira. *Gumbel-max trick*. 2014. URL: http://timvieira.github.io/blog/post/2014/07/31/gumbel-max-trick/.

[136] Karsten Vogt, Oliver Müller und Jörn Ostermann. „Facial Landmark Localization using Robust Relationship Priors and Approximative Gibbs Sampling“. In: *Advances in Visual Computing*. Bd. 9475. 2015, S. 365–376. ISBN: 978-3-319-27857-5. DOI: 10.1007/978-3-319-27857-5.

[137] Karsten Vogt, Björn Scheuermann, Christian Becker, Torsten Büschenfeld, Bodo Rosenhahn und Jörn Ostermann. „Automated Extraction of Plantations from Ikonos Satellite Imagery using a Level Set Based Segmentation Method“. In: *ISPRS Technical Commission VII Symposium*. Bd. 38. 7. 2010, S. 275–280. ISBN: 1682-1777.

[138] Ulrike Von Luxburg. „A tutorial on spectral clustering“. In: *Statistics and computing* 17.4 (2007), S. 395–416.

[139] John Von Neumann und Oskar Morgenstern. *Theory of games and economic behavior*. Princeton university press, 2007.

[140] Abraham Wald. „Contributions to the theory of statistical estimation and testing hypotheses“. In: *The Annals of Mathematical Statistics* 10.4 (1939), S. 299–326.

[141] Jan D. Wegner, Franz Rottensteiner, Markus Gerke und Gunho Sohn. *The ISPRS 2D Labelling Challenge*. http://www2.isprs.org/commissions/comm3/wg4/semantic-labeling.html. Accessed 11/03/2016. 2016.

[142] Frank Wilcoxon. „Individual comparisons by ranking methods“. In: *Biometrics bulletin* 1.6 (1945), S. 80–83.

[143] David H Wolpert und William G Macready. „No free lunch theorems for optimization“. In: *IEEE transactions on evolutionary computation* 1.1 (1997), S. 67–82.

[144] Pengcheng Wu und Thomas G. Dietterich. „Improving SVM Accuracy by Training on Auxiliary Data Sources“. In: *Proceedings of the Twenty-first International Conference on Machine Learning*. ICML '04. Banff, Alberta, Canada: ACM, 2004, S. 110–. ISBN: 1-58113-838-5. DOI: 10.1145/1015330.1015436. URL: http://doi.acm.org/10.1145/1015330.1015436.

[145] Ting-Fan Wu, Chih-Jen Lin und Ruby C Weng. „Probability estimates for multi-class classification by pairwise coupling“. In: *Journal of Machine Learning Research* 5.Aug (2004), S. 975–1005.

[146] Zhijie Xu und Shiliang Sun. „Multi-source transfer learning with multi-view adaboost“. In: *International Conference on Neural Information Processing*. Springer. 2012, S. 332–339.

[147] Ya Xue, Xuejun Liao, Lawrence Carin und Balaji Krishnapuram. „Multi-task learning for classification with Dirichlet process priors“. In: *Journal of Machine Learning Research* 8.Jan (2007), S. 35–63.

[148] Yi Yao und Gianfranco Doretto. „Boosting for transfer learning with multiple sources“. In: *Computer vision and pattern recognition (CVPR), 2010 IEEE conference on*. IEEE. 2010, S. 1855–1862.

[149] Jason Yosinski, Jeff Clune, Yoshua Bengio und Hod Lipson. „How transferable are features in deep neural networks?“ In: *Advances in neural information processing systems*. 2014, S. 3320–3328.

[150] Bianca Zadrozny. „Learning and Evaluating Classifiers Under Sample Selection Bias“. In: *Proceedings of the Twenty-first International Conference on Machine Learning*. ICML '04. Banff, Alberta, Canada: ACM, 2004, S. 114–. ISBN: 1-58113-838-5. DOI: 10.1145/1015330.1015425. URL: http://doi.acm.org/10.1145/1015330.1015425.

[151] Jian Zhang, Zoubin Ghahramani und Yiming Yang. „Learning multiple related tasks using latent independent component analysis“. In: *Advances in neural information processing systems*. 2006, S. 1585–1592.

[152] J. Zhou, J. Liu und X. Luo. „Rademacher complexity bound for domain adaptation regression“. In: *2015 Conference on Technologies and Applications of Artificial Intelligence (TAAI)*. Nov. 2015, S. 273–280. DOI: 10.1109/TAAI.2015.7407123.

[153] Weida Zhou, Li Zhang und Licheng Jiao. „Linear programming support vector machines“. In: *Pattern recognition* 35.12 (2002), S. 2927–2936.

[154] Ji Zhu und Trevor Hastie. „Kernel Logistic Regression and the Import Vector Machine“. In: *Journal of Computational and Graphical Statistics* 14.1 (2005), S. 185–205. DOI: 10.1198/106186005X25619. eprint: http://dx.doi.org/10.1198/106186005X25619. URL: http://dx.doi.org/10.1198/106186005X25619.

[155] Arthur Zimek, Erich Schubert und Hans-Peter Kriegel. „A survey on unsupervised outlier detection in high-dimensional numerical data“. In: *Statistical Analysis and Data Mining* 5.5 (2012), S. 363–387.

LEBENSLAUF

Karsten Vogt
Geboren am 15.11.1982
in Hannover

Beruf

01/2010 – 10/2018	Wissenschaftlicher Mitarbeiter am Institut für Informationsverarbeitung an der Leibniz Universität Hannover
02/2006 – 10/2008, 11/2009 – 12/2009	Studentische Hilfskraft am Institut für Informationsverarbeitung an der Leibniz Universität Hannover
07/2003	Werksstudent bei Volkswagen

Studium

10/2006 – 09/2009	Studiengang M. Sc. in Informatik an der Leibniz Universität Hannover
10/2003 – 12/2006	Studiengang B. Sc. in Informatik an der Leibniz Universität Hannover

Schulbildung

1993 – 06/2002	IGS Garbsen Oberstufe mit Abitur
1989 – 1993	Grundschule Saturnring

Wehrdienst

07/2002 – 03/2003	Wehrdienst bei der Bundeswehr

VERÖFFENTLICHUNGEN

[I] Vogt, K., Paul, A., Rottensteiner, F., Ostermann, J., Heipke, C. „Unsupervised Source Selection for Domain Adaptation“. In: *Photogrammetric Engineering & Remote Sensing* 84.5, 2018, S. 249-261.

[II] Vogt, K., Paul, A., Rottensteiner, F., Ostermann, J., Heipke, C. „Boosted Unsupervised Multi-Source Selection for Domain Adaptation“. In: *ISPRS Annals of Photogrammetry, Remote Sensing & Spatial Information Sciences* IV-1/W1, 2017, S. 229-236.

[III] Vogt, K., Ostermann, J. „Soft Margin Bayes-Point-Machine Classification via Adaptive Direction Sampling“. In: *Scandinavian Conference on Image Analysis*, 2017, S. 313-324.

[IV] Pestel-Schiller, U., Vogt, K., Ostermann, J., Groß, W. „Impact of Hyperspectral Image Coding on Subpixel Detection“. In: *Proceedings of 32nd Picture Coding Symposium*, 2016, S. 1-5.

[V] Baumann, F., Vogt, K., Ehlers, A., Rosenhahn, B. „Probabilistic Nodes for Modelling Classification Uncertainty for Random Forest“. In: *14th IAPR International Conference on Machine Vision Applications (MVA)*, 2015, S. 510-513.

[VI] Baumann, F., Chen, J., Vogt, K., Rosenhahn, B. „Improved Threshold Selection by using Calibrated Probabilities for Random Forest Classifiers“. In: *12th Conference on Computer and Robot Vision (CRV)*, 2015, S. 155-160.

[VII] Vogt, K., Müller, O., Ostermann, J. „Facial Landmark Localization using Robust Relationship Priors and Approximative Gibbs Sampling“. In: *Advances in Visual Computing* 9475, 2015, S. 365-376.

[VIII] Haenssle, H.A., Hofmann, L., Schoen, M., Werfel, T., Guenther, A., Basu, C., Roth, B., Niemann, K.-H., Schmerling, O. Scharenberg, S., Luellau, F., Vogt,

K., Rosenhahn, B., Meinhardt-Wollenweber, M., Emmert, S. „Automated Skin Cancer-Screening with contact free Remote-Dermatoscopy“. In: *Journal der Deutschen Dermatologischen Gesellschaft* 13, 2015, S.164-165.

[IX] Baumann, F., Ehlers, A., Vogt, K., Rosenhahn, B. „Cascaded Random Forest for Fast Object Detection“. In: *18th Scandinavian Conference on Image Analysis (SCIA)*, 2013, S. 131-142.

[X] Helmholz, P., Becker, C., Breitkopf, U., Büschenfeld, T., Busch, A., Braun, C., Grünreich, D., Müller, S., Ostermann, J., Pahl, M., Rottensteiner, F., Vogt, K., Ziems, M., Heipke, C. „Semi-automatic Quality Control of Topographic Data Sets“. In: *Photogrammetric Engineering & Remote Sensing* 78.9, 2012, S. 959-972.

[XI] Vogt, K., Scheuermann, B., Becker, C., Büschenfeld, T., Rosenhahn, B., Ostermann, J. „Automated Extraction of Plantations from Ikonos Satellite Imagery using a Level Set Based Segmentation Method“. In: *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 38.7, 2010, S. 275-280.

[XII] Helmholz, P., Becker, C., Breitkopf, U., Büschenfeld, T., Busch, A., Grünreich, D., Heipke, C., Müller, S., Ostermann, J., Pahl, M., Vogt, K., Ziems, M. „Semiautomatic Quality Control of Topographic Reference Datasets“. In: *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 38.4, 2010.