UNIVERSITÄT KOBLENZ · LANDAU
Fachbereich 4: Informatik

Fraunhofer Institut Angewandte Informationstechnik
Forschungsbereich Life Sciences Informatik

Entwicklung einer modularen graphischen Evaluationsumgebung zur Merkmalsbewertung und Anwendung am Beispiel der Pollen-Erkennung

Diplomarbeit

zur Erlangung des Grades eines

Diplom-Informatikers
im Studiengang Computervisualistik

vorgelegt von

Matthias Häusler

Betreuer:	Dipl.-Inform. Dietlind Zühlke, Fraunhofer-Institut für Angewandte Informationstechnik FIT, Forschungsbereich Life Sciences Informatik
Erstgutachter:	Prof. Dr.-Ing. Dietrich Paulus, Institut für Computervisualistik, Fachbereich Informatik
Zweitgutachter:	Prof. Dr. Thomas Berlage, Fraunhofer-Institut für Angewandte Informationstechnik FIT, Forschungsbereich Life Sciences Informatik

Koblenz, im März 2009

Erklärung

Ich versichere, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe und dass die Arbeit in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen hat und von dieser als Teil einer Prüfungsleistung angenommen wurde. Alle Ausführungen, die wörtlich oder sinngemäß übernommen wurden, sind als solche gekennzeichnet.

Die Richtlinien der Arbeitsgruppe für Studien- und Diplomarbeiten habe ich gelesen und anerkannt, insbesondere die Regelung des Nutzungsrechts.

Mit der Einstellung dieser Arbeit in die Bibliothek bin ich einverstanden. ja ☐ nein ☐

Der Veröffentlichung dieser Arbeit im Internet stimme ich zu. ja ☐ nein ☐

Koblenz, den ..... Unterschrift

Inhaltsverzeichnis

Kapitel 1

Einleitung

Die Mustererkennung, als Teilgebiet der Informatik, befasst sich mit der Simulation einer perzeptiven Leistung durch den Computer. Verfahren aus der maschinellen Mustererkennung finden bereits in vielen Bereichen Anwendung. In der biologischen und medizinischen Forschung werden durch solche Automatisierungstechniken z. B. die kontinuierliche Durchführung und Auswertung von umfangreichen Experimenten ermöglicht.

Ein Teilproblem der Mustererkennung ist die Klassifikation von Mustern. Bei Verfahren nach dem Prinzip des überwachten Lernens (engl. supervised learning) wird dabei einem Muster genau eine von mehreren erlernten Klassen zugewiesen. Diese Zuweisung findet auf Basis von Modellen oder Prototypen der Klassen statt. Die Parameter dieser Modelle werden in einer Lernphase anhand von klassifizierten Stichproben geschätzt.

Bei allen praktischen Anwendungsszenarien ist das Ziel nicht die Klassifikation bereits bekannter Daten, also der, die zum Training des Klassifikations-Algorithmus verwandt wurden, sondern es sollen von dem System neue, unbekannte Daten korrekt erkannt und klassifiziert werden. Der Erfolg dieses Vorgehens ist bedingt durch den Grad der Generalisierung, der von den gewählten Modellen und deren geschätzten Parametern ausgeht. Das heißt, wurde ein Modell gewählt und während des Lernschrittes so parametrisiert, dass es in seiner Komplexität in der Lage ist, die einzelnen Klassen für die Muster aus der Lernstichprobe hinreichend zu beschreiben und zu unterscheiden, muss dies für Muster aus einer anderen Stichprobe nicht mehr zwingend zutreffen.

Diese mangelnde Fähigkeit zur Generalisierung kann durch verschiedene Faktoren verursacht werden. Ein großes Problem stellt dabei die Schätzung der Parameter anhand einer endlichen Stichprobe dar. Ist die Stichprobe für den Problemkreis nicht ausreichend repräsentativ, ist auch mit keiner guten Generalisierung des Klassifikators zu rechnen. Bei Modellen mit vielen Parametern kann es auch zu der sogenannten Überanpassung (engl. overfitting) kommen. Mit einer entsprechend großen Anzahl an Parametern ist man so prinzipiell fast immer in der Lage, die Grenzen zweier Klassen für eine einzelne Stichprobe zu bestimmen [Nie07]. Durch den damit verbundenen hohen Spezialisierungsgrad muss dies dann aber nicht mehr für Muster außerhalb der Stichprobe gelten. Generell besteht bei Klassifikationssystemen eine Tendenz zu steigenden Fehlerraten bei steigender Komplexität des Modells. Dieses Phänomen der „curse of dimensionality“ wird im Wesentlichen durch entstehende Schätzfehler bei der Parameterbestimmung begründet [DHS00], welchen wieder das Problem der endlichen Größe der Stichprobe und die daraus resultierende Unterabtastung des hochdimensionalen Merkmalsraumes zu Grunde liegt [Bis06]. Allgemein lässt sich daher in der Praxis durch Hinzunahme neuer Parameter, in Form zusätzlicher Merkmale, die Fehlerrate des Klassifikators nicht beliebig verringern, sondern sie kann sogar mit steigender Dimension des Merkmalsraumes zunehmen [DHS00]. Das Problem die Komplexität des Modells – die Anzahl der Merkmale – so zu wählen, dass einerseits die Unterschiede zwischen den Klassen dadurch erfasst werden und andererseits eine gute Generalisierung gewährleistet wird, ist in der statistischen Musterklassifikation Gegenstand vieler Forschungsarbeiten.

Ansätze aus diesem Bereich befassen sich mit der Reduktion der Dimension des Merkmalsraumes unter Maximierung eines Gütekriteriums. Meistens wird davon ausgegangen, dass dieses Kriterium an die Fehlerwahrscheinlichkeit des Klassifikations-Algorithmus gekoppelt ist (vgl. Abschnitt 2.2.1). Einige wählen die Fehlerwahrscheinlichkeit auch direkt als Kriterium. Methoden zur Dimensionsreduktion lassen sich in zwei Kategorien einteilen. Zum einen in Verfahren der linearen und nichtlinearen Einbettung, die unter der Prämisse arbeiten, dass die Daten in einen höherdimensionalen Raum eingebettet sind und so auch in einem Unterraum dargestellt werden können, der ihrer eigentlichen, intrinsischen Dimensionalität entspricht. Durch diese Abbildung entstehen Merkmalsvektoren zu einer neuen Basis. Die Verfahren der zweiten Kategorie bewerten die Merkmale und selektieren

aus der Gesamtheit eine Untermenge. Bei diesem Ansatz bleiben die einzelnen Merkmale für sich bestehen und verlieren nicht durch Kombinationen ihre direkte Interpretierbarkeit.

Alle diese Ansätze führen durch die angestrebte Reduktion der Datendimension auch in der Regel zu einer Verkürzung der Laufzeit während der Lern- und Klassifikationsphase (vgl. Abschnitt 2.2.1). Methoden der Merkmalsselektion bewirken zusätzlich einen Performance-Gewinn während der Merkmalskonstruktion, da nur noch ein Teil der ursprünglichen Merkmale berechnet werden muss.

Diese Diplomarbeit findet im Rahmen des Projektes Pollenmonitor (vgl. Abschnitt 2.1.3) am Fraunhofer-Institut für Angewandte Informationstechnik (FIT) im Forschungsbereich Life Science Informatik statt. Innerhalb dieses Projektes wurde ein Klassifikationssystem zur Bestimmung der Taxa luftgetragener Pollenkörner realisiert. Dieses System stellt eine der Software-Komponenten eines Apparates zur vollautomatischen Erkennung und Auszählung von Pollenkörnern in Luftproben dar. Die durch den Automaten ermittelten Pollenkonzentrationen stehen online zur Weiterverarbeitung bereit. Auf Basis dieser Daten soll in einem festen Zeitfenster eine Pollenflug-Prognose gestellt werden. Um eine zuverlässige Prognose zu ermöglichen muss das Klassifikationssystem robust und in Echtzeit laufen. Robust im Sinne der Klassifikatorleistung heißt, eine geringe Wahrscheinlichkeit einer Fehlklassifikation. Dies setzt eine gute Generalisierung voraus, welche in diesem Kontext eine besondere Herausforderung darstellt, da hier Daten klassifiziert werden, die unter realen, unsteten Bedingungen entstehen und somit große Varianzen innerhalb der einzelnen Klassen nicht zu vermeiden sind. Das Kriterium der Echtzeit fordert eine konstante, möglichst geringe, Laufzeit der Algorithmen. Um diesen Forderungen gerecht zu werden und auch die unumgänglichen Adaptionsprozesse, welche durch Hinzunahme neuer Pollen-Klassen oder durch neue Ausprägungen von bereits erlernten Klassen angestoßen werden, zu unterstützten, bieten sich Verfahren der Dimensionsreduktion und Merkmalsbewertung an.

Ziel dieser Arbeit ist es Kriterien und Gütemaße zur Bewertung von Merkmalen aus der Musterklassfikation zu finden und diese so in eine graphische Evaluationsumgebung zu integrieren, dass der Nutzer dadurch befähigt wird, Erkenntnisse über die Struktur des Merkmalsraumes und die Qualität der einzelnen Merkmale zu erlangen, so dass er anhand dieser zielführend eine möglichst optimale Teilmenge – im Sinne der Klassifika-

tionsgüte und der Anzahl der Merkmale – aus den vorhandenen Merkmalen gewinnen kann. Die entwicklete Applikation soll mittels eines klassifizierten Testdatensatzes aus dem Pollenmonitor-Projekt validiert werden, der extrahierte Textur- und Form-Merkmale luftgetragener Pollenkörner verschiedener Taxa und anderer Schwebeteilchen enthält. Zur Klassifikation wird der hierarchische Multi-Klassen-Klassifikator, der im Rahmen des Projektes entwicklet wurde, genutzt.

Die Arbeit gliedert sich wie folgt. In Kapitel 2 wird zuerst auf die allgemeinen visuellen Merkmale von Pollenkörnern und die manuelle Pollenzählung eingegangen, dann wird eine Übersicht über das Pollenmonitor-Projekt zur automatisierten Pollen-Erkennung gegeben. Der zweite Abschnitt des Kapitels 2 erläutert die Grundlagen der Klassifikation von Mustern und gibt einen Überblick über die Teilschritte Merkmalsextraktion und Klassifikation. Kapitel 3 zeigt Kriterien und Gütemaße zur Merkmalsbewertung und bietet eine Übersicht der Verfahren zur Merkmalsselektion. Speziell wird auf die Familie der Relief-Algorithmen eingegangen, um anschließend die Modifikationen und Erweiterungen, die im Rahmen dieser Arbeit entwickelt und umgesetzt wurden, vorzustellen. Darauf folgt in Kapitel 4 der Entwurf und die Umsetzung der modularen Evaluationsumgebung, welche die graphische Benutzerschnittstelle zu den implementierten Merkmalsselektionsverfahren und dem genutzten Klassifikator darstellt. Experimente zur Bewertung und Selektion einer „möglichst geeigneten“ (siehe Abschnitt 3.1) Untermenge an Merkmalen aus dem Pollenmonitor-Testdatensatz folgen in Kapitel 5. In Kapitel 6 werden schließlich die Ergebnisse dieser Arbeit zusammengefasst und ein Ausblick auf Erweiterungen der Evaluationsumgebung und der implementieren Verfahren gegeben sowie weitere Anwendungsbereiche aufgezeigt.

Kapitel 2

Grundlagen

In diesem Kapitel folgen nun zuerst Grundlagen zu Teilbereichen der Pollenanalyse, das heißt eine kurze Darstellung der manuellen Pollen-Bestimmung und Pollen-Vorhersage sowie eine Beschreibung des Pollenmonitor-Projektes zur automatischen Pollen-Erkennung, in dessen Rahmen diese Arbeit erstellt wurde. In Abschnitt 2.2 wird zu Anfang das Problem der Musterklassifikation formal mathematisch betrachtet, um dann auf die einzelnen Teilbereiche einzugehen, dabei wird in Abschnitt 2.2.1 das Teilproblem Merkmalsextraktion detaillierter betrachtet, um die Verfahren zur Merkmalsbewertung und Merkmalsauswahl im Gesamtzusammenhang darzustellen.

2.1 Pollenanalyse

Das Gebiet der Pollenanlyse (Palynologie) erstreckt sich interdisziplinär über die Bereiche der Geowissenschaften, Biologie und Klimaforschung. Hierbei werden Pollenkörner, Sporen und andere organische Mikropartikel untersucht.

2.1.1 Morphologie des Pollenkorns

Pollenkörner sind Zellen, welche durch eine besonders dicke Zellwand, das Sporoderm, geschützt sind. Diese lässt sich in eine äußere (Exine) und eine innere Schicht (Intine) unterteilen [WOW01]. Die Schichten bilden, durch ihre spezifischen Strukturen, wie Verdickungen, Öffnungen und Beschaffenheit der Oberfläche, die visuellen Hauptmerkmale zur Einteilung der Pollenkörner in verschiedene Taxa (z. B. Pflanzenfamilie, -gattung oder -art). So befinden sich in der Exine-Schicht Aussparungen unterschiedlicher Form, welche als Keimöffnungen dienen. Solche Aperturen sind entweder durch Deckel auf der Exine oder durch Verdickungen der Intine (Onci oder „Zwischenkörper“ [RG88]) geschützt. Sie treten in Form von runden Poraten, als längliche Keimspalten (Colpen) oder in Kombination als colporate Öffnungen auf [WOW01].

Die Größe der Pollenkörner schwankt zwischen den einzelnen Taxa beträchtlich. Es existieren Körner mit einem Durchmesser von $8\,\mu\mathrm{m}$ bis zu $250\,\mu\mathrm{m}$ [WOW01]. Die Form entspricht bei windblütigen (anemophilen) Pflanzen überwiegend einer Kugel oder einem gestreckten Ellipsoiden. Andere Gestalten treten vor allem bei Pollen von insektenbestäubten (entomophilen) Pflanzen sowie bei Körnern der Windblütler auf, die wegen ihrer Größe nur durch zusätzliche Luftsäckchen eine Schwebefähigkeit erlangen. Dazu gehören z. B. die Pollen von Nadelhölzern (Koniferen), welche sich deutlich in Form und Größe visuell von den Pollen anderer Windblütler abgrenzen, deren Umfang normalerweise 15–$40\,\mu\mathrm{m}$ beträgt [WOW01].

Entomophile Pflanzen bilden Pollenkörner, die von Pollenkit umgeben und mit einer rauen Exine ausgestattet sind [WOW01]. Der Kit dient zur Bildung von Pollen-Agglomeraten und sorgt für die Haftung am Insekt. Im Gegensatz dazu ist die Exine der Pollen der Windblütler eher glatt mit weniger prägnanter Textur.

In Bild 2.1 und 2.2 sind einige der genannten Merkmale ersichtlich.

2.1.2 Pollenflug-Vorhersage

Die stetig steigende Zahl der Neuerkrankungen an allergischer Rhinitis (Heuschnupfen) und eine Prävalenz von 15–20 % unter der erwachsenen Bevölkerung innerhalb der Indu-

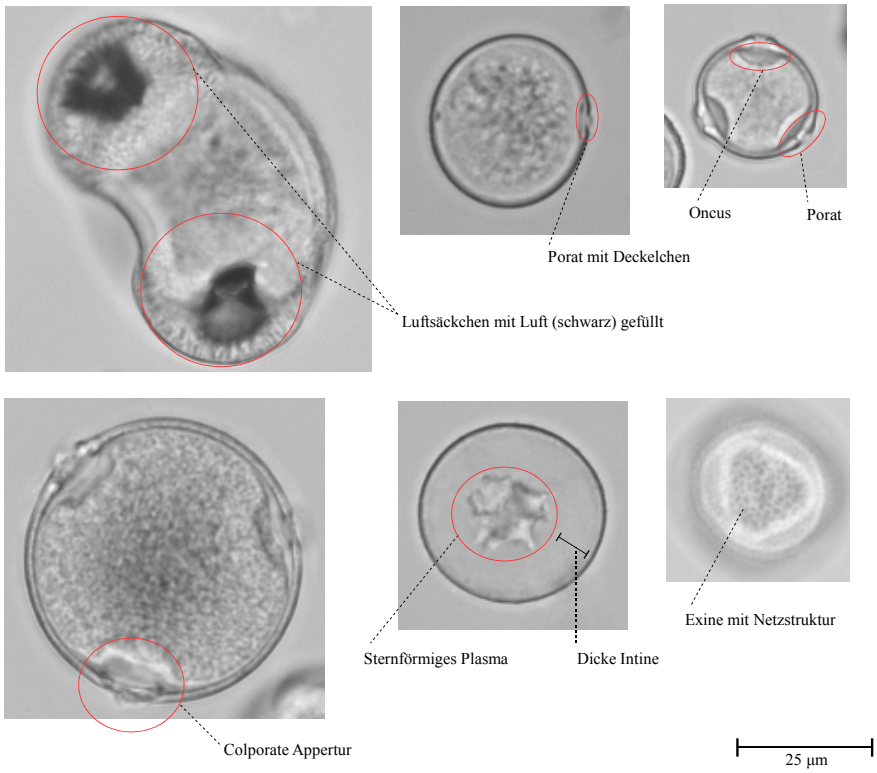
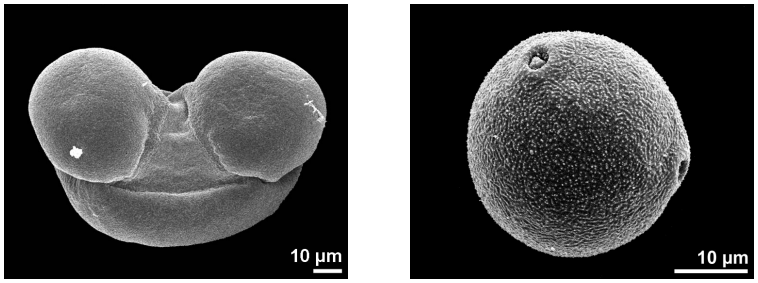


Bild 2.1: Schichtbilder (Durchlichtmikroskopie) von Pollenkörnern mit markierten Merkmalen folgender Taxa (soweit nicht anders vermerkt, Äquatorialschnitt der Pollenkörner in Pollage): Kiefer (oben links), Gras (oben Mitte), Birke (oben rechts), Rotbuche (unten rechts), Wacholder (unten Mitte), Esche im Polschnitt (unten rechts) (Schichtbilder: Helmut Hund GmbH).



(a) Mit Luftsäckchen. (b) Mit deutlichen Aperturen.

Bild 2.2: Rasterelektronenmikroskop-Aufnahmen eines Pollenkorns eines Kieferngewächses (a) (Quelle: [Hala]) und einer Birke (b) (Quelle: [Halb]).

strieländer [WOW01], schaffen einen großen Handlungsbedarf, der durch mehrere Faktoren bedingt ist. Die mannigfaltigen Ausprägungen des Krankheitsbildes und die damit einhergehenden Beeinträchtigungen im Privat- und Arbeitsleben, führen während der Vegetationsperioden zu einer deutlichen Minderung der Lebensqualität der Betroffenen sowie zu hohen Kosten für das Gesundheitssytem und die Wirtschaft. Zusätzliche Komorbiditäten, wie eine pollenassoziierte Nahrungsmittelallergie, erweitern dabei noch das Spektrum der Symptome [WOW01].

So sind weltweit schon in mehreren Ländern Messnetze zur Pollenflug-Vorhersage entstanden, um Pollen-Allergikern, die durch ärztliche Untersuchungen über die auslösenden Allergene informiert sind, Hinweise zur Meidung gewisser Regionen oder zur medikamentösen Vorbeugung zu liefern.

In Deutschland wird die Prognose von der Stiftung Polleninformationsdienst Deutschland (PID) in Kooperation mit dem Deutschen Wetterdienst (DWD) erstellt. Hierfür besitzt der PID ein Messnetz aus über 50 Pollenfallen. Diese „Burkard-Fallen“[1] (siehe Bild 2.3) saugen einen konstanten Luftstrom von $10\,\mathrm{l}$ pro Minute an [WOW01]. Auf diese Weise wird das durchschnittliche Atemvolumen eines Menschen simuliert. Die in der Luft enthaltenen Schwebeteilchen (Aerosolpartikel) werden dabei auf eine präparierte rotierende Trommel abgeschieden. Auf diese Weise können später Aussagen über den Verlauf des Pollenaufkommens während der Probenahme getroffen werden, z. B. Anzahl der Birken-

[1] „Burkhard Pollen and Spore Trap“ der englischen Firma Burkhard Manufactoring Company Limited

Bild 2.3: Burkard-Falle (Quelle: Burkhard Manufactoring Company Limited).

pollen pro m^3 Luft im Tagesdurchschnitt. Um solche Statistiken zu erhalten, müssen zuvor die einzelnen Pollenkonzentrationen auf den Proben bestimmt werden. Das geschieht zur Zeit noch von Hand durch entsprechend ausgebildete Fachkräfte. Die Zuordnung der Pollenkörner zu ihrem korrespondierenden Taxon[2] erfolgt dabei auf Basis spezifischer visueller Merkmale (vgl.Abschnitt 2.1.1). Zur Sichtung dieser Charakteristika werden die Partikel zuerst in ein Einbettungsmedium überführt in welchem sie quellen. Das hierdurch gewonnene Präparat wird dann anhand eines Durchlichtmikroskopes analysiert. Aus den Auswertungen der Proben generiert man für jeden Einzugsbereich einer Pollenfalle einen lokalen Pollenflug-Trend. Aus diesen Trends, phänomenologischen Daten aus Beobachtungen des Blütestandes der Pflanzen und den, durch den DWD berechneten, meteorologischen Faktoren entsteht die regionale Pollenflug-Vorhersage.

Momentan findet die Berechnung und Ausgabe der Prognose jeden dritten Tag (nur während der Vegetationsperiode) statt. [WOW01]. Ein Pollenflug-Kalender, basierend auf über Jahre gemittelten Werten, ist hingegen nicht in der Lage, die wetterabhängigen Varianzen in den Blütezeiten der einzelnen Pflanzen zu modellieren [WOW01] und hiermit kein Ersatz für diese Prognose.

[2]Nach [WOW01] handelt es sich hierbei entweder um die Pflanzenfamile oder -gattung. Eine Bestimmung auf Ebene der Pflanzenart ist über morphologische Kriterien meist nicht möglich.

2.1.3 Das Projekt „Pollenmonitor“

Die in Abschnitt 2.1.2 beschriebene manuelle Analyse der Luftproben durch den Experten ist ein sehr zeitintensiver Vorgang. Pollenkörner verschiedener Taxa unterscheiden sich häufig nur durch kleinste Details, welche mühsam lokalisiert werden müssen. Um aber eine möglichst aussagekräftige Pollenflug-Prognose zu stellen, bedarf es einer zeitnahen Auswertung der Proben und einer Maximierung der Repräsentativität der Messungen. Erstem sind bei der manuellen Auszählung klare Grenzen gesetzt und Zweites ist nur durch Erhöhung der Anzahl der Stichproben, in Form zusätzlicher Pollenfallen, zu erreichen. Beides erfordert einen großen personellen und logistischen Aufwand. Dieser ist bedingt durch die Menge der dafür benötigten Pollenzähler und den Zustand, dass nach bisherigen Methoden (vgl. Abschnitt 2.1.2) die Proben zur Untersuchung in dafür vorgesehene Labore transportiert werden müssen. Eine performante und robuste vollautomatische Pollenanalyse vor Ort kann die Faktoren Zeit und Personalaufwand entscheidend minimieren.

Im Projekt Pollenmonitor wird eine solche automatisierte Pollen-Erkennung realisiert und zur Serienreife gebracht. Das Projekt findet in einer Kooperation zwischen der Helmut Hund GmbH und dem FIT statt. In diesem Rahmen wird auch an einer aktuellen Ausschreibung des DWD, zur Einführung eines engmaschigen autonomen Pollen-Messnetzes in Deutschland teilgenommen. Der hierbei entwickelte BIO-AEROSOL-ANALYSATOR (siehe Bild 2.4) ist ein Automat mit Ansaug-Mechanik, integrierter Optik und Bildanalyse-Software.

Zur Erfassung der Pollenkonzentrationen beprobt dieser kontinuierlich $5\ \mathrm{m}^3$ Luft pro Stunde. Die darin enthaltenen Partikel werden zuerst durch einen virtuellen Impaktor so nach Größe getrennt, dass nur noch Objekte von relevanter Größenordnung verbleiben und z. B. Feinstaub und Pflanzenfasern ausgeschieden werden. Der gefilterte Partikelstrom wird danach auf Probeplättchen abgesondert. Hierauf befindet sich bereits ein Einbettungsmedium (vgl. Abschnitt 2.1.2) in dem die Pollenkörner zur gleichmäßigen Quellung kommen. Eine Mechanik führt dann die Plättchen unter das interne Lichtmikroskop, welches die Proben schichtweise abtastet. Die so erzeugten Bildstapel werden an die Bildanalyse-Software weitergereicht. Dort werden zunächst die 3D-Stapel in einem Vorverarbeitungsschritt auf Einzelbilder reduziert. Danach werden aus diesen durch ein mehrstufiges Segmentierungs-

Bild 2.4: BIO-AEROSOL-ANALYSATOR (BAA500) (Quelle: Helmut Hund GmbH).

verfahren die Schwebeteilchen extrahiert. Auf den resultierenden Bildregionen werden Merkmale berechnet, anhand deren ein hierarchischer Klassifizierungs-Algorithmus die Pollenkörner und die zugehörigen Taxa bestimmt. Die ermittelten Luft-Pollenkonzentrationen des erkannten Taxon und lokale Umweltdaten, wie Luftdruck, Temperatur und Luftfeuchtigkeit werden stündlich an eine zentrale Erfassungsstelle übermittelt. Somit ermöglicht man eine deutlich höhere zeitliche Auflösung der Prognosen als bei bisheriger manueller Vorgehensweise (vgl. Abschnitt 2.1.2) und auch Statusberichte über aktuelle lokale Pollenbelastungen stehen so erstmals zur Verfügung.

Zum Training und zur Validierung des Klassifikators steht ein durch einen Experten annotierter Datensatz (siehe Abschnitt 5.1.1) zur Verfügung. Dieser enthält ca. 20 000 segmentierte luftgetragene Objekte mit zugehörigem Klassen-Label. Bei der Validierung wurden auf den vom Trainingsdatensatz disjunkten Testdaten Erkennungraten zwischen 80 % und 98 % mit einer ebenso hohen Präzision (Anteil der richtig klassifizierten Objekte einer Klasse, die dieser zugewiesen wurden, vgl. Abschnitt 5.1.2) erzielt. Unter den erkannten Taxa sind die vom Deutscher Wetterdienst (DWD) als allergologisch relevant eingestuften Pollenkörner von Hasel, Erle, Birke, Süßgräser ohne Roggen, Beifuß und Traubenkraut (Ambrosia) sowie die nicht allergologisch relevanten Pollen von Ahorn, Eibe, Eiche, Hainbuche, Roggen und Weide. Die Identifikation zusätzlicher Pollen-Taxa oder anderer Aerosolpartikel kann über entsprechende Trainingsdaten ebenfalls erlernt werden.

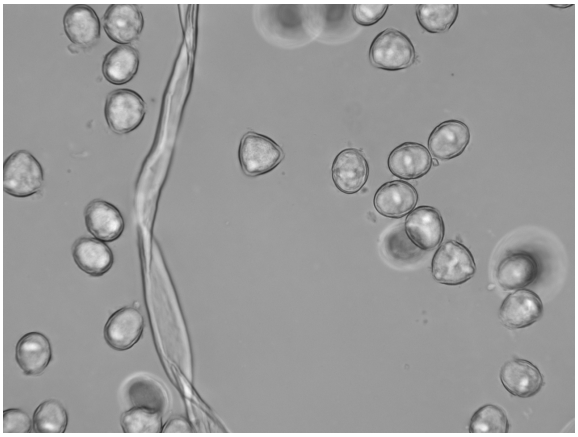


Bild 2.5: Eine Position einer gescannten Probe.

Repräsentative Klassifikations-Ergebnisse des laufenden Gesamtssystem (Hardware- und Softwarekomponenten) unter Zielbedingungen stehen bis dato noch nicht zur Verfügung, da vor Einsetzen der Vegetationsperiode keine Pollenkörner in den Luftproben zu finden sind.[3]

2.2 Klassifikation von Mustern

Im Weiteren wird zuerst die Klassifikation von Mustern als mathematisches Problem beschrieben. Die in diesem Zusammenhang verwandten mathematischen Definitionen stammen, sofern nicht anders angegeben, aus [Nie07]. Darauf folgen Beschreibungen der ermittelten Teilschritte der Musterklassifikation. Im Abschnitt 2.2.1 wird der Teilschritt „Merkmalsextraktion“ ausführlicher betrachtet, um die in dieser Arbeit implementierten Verfahren aus dem Bereich der Merkmalsbewertung und Merkmalsselektion in diesen Kontext einzuordnen. In Abschnitt 2.2.2 wird der Klassifikationsschritt als Problem der numerischen Klassifikation dargestellt. Hierbei werden Grundlagen der statistischen Entscheidungstheorie und mögliche Lösungen des Klassifikationsproblems kurz skizziert. Abschnitt 2.2.3 zeigt Probleme auf, die sich aus dem Verhältnis der Größe der endlichen

[3] Diese Arbeit wurde während der Wintermonate verfasst.

Stichprobe zur Komplexität des verwandten Modells (Klassifikators) ergeben.

Niemann bezieht die Klassifikation von Mustern auf einen Problemkreis Ω, der einen Teil der sensorisch erfassbaren Umwelt U repräsentiert. Ein Problemkreis beinhaltet daher nur Muster ${}^{\varrho}\boldsymbol{f}(\boldsymbol{x})$ eines klar abgegrenzten Bereiches und lässt sich als Menge

$$\Omega = \{{}^{\varrho}\boldsymbol{f}(\boldsymbol{x}) \mid \varrho = 1, 2, \ldots\} \subset U \tag{2.1}$$

darstellen. Ein Muster ist hierbei eine Funktion

$$\boldsymbol{f}(\boldsymbol{x}) = \begin{pmatrix} f_1(x_1, \ldots, x_n) \\ f_2(x_1, \ldots, x_n) \\ \vdots \\ f_m(x_1, \ldots, x_n) \end{pmatrix} \tag{2.2}$$

deren Komponentenanzahl m und n durch die verwandte Sensorik bedingt ist. Wird ein Problemkreis durch mehrere Sensoren $1, \ldots, \sigma, \ldots, s$ erfasst, so ergibt sich daraus die Menge

$$\Omega = \{{}^{\varrho}\boldsymbol{f}_1(\boldsymbol{x}_1), \ldots, {}^{\varrho}\boldsymbol{f}_\sigma(\boldsymbol{x}_\sigma), \ldots, {}^{\varrho}\boldsymbol{f}_s(\boldsymbol{x}_s) \mid \varrho = 1, 2, \ldots\} \subset U \; . \tag{2.3}$$

Im Rahmen dieser Arbeit werden nur Grauwertbilder $f(x, y)$, bzw. Bildstapel $f(x, y, z)$, eines Sensors verarbeitet. Insofern gilt im Weiteren Gleichung 2.1 mit fester Komponentenanzahl für alle Muster ${}^{\varrho}\boldsymbol{f}(\boldsymbol{x})$. Zusätzlich gilt die Beschränkung auf „einfache“ Muster. Niemann unterscheidet diese von „komplexen“ Mustern dadurch, dass ein „einfaches“ Muster durch die Abbildung

$${}^{\varrho}\boldsymbol{f}(x) \rightarrow \Omega_\kappa \tag{2.4}$$

klassifiziert wird, also durch die Zuordnung zu genau einer Klasse Ω_κ von k möglichen – im Falle der Klassifikation von Pollenkörner also einem Taxon, z. B. Kiefern. Diese k Klassen bilden eine Partition von Ω durch

$$\begin{aligned} \Omega_\kappa \neq \emptyset \qquad & \kappa = 1, \ldots, k \; , \\ \Omega_\kappa \cap \Omega_\lambda = \emptyset \qquad & \lambda \neq \kappa \; , \\ und \qquad & \bigcup_{\kappa=1}^{k} \Omega_\kappa = \Omega \; . \end{aligned} \tag{2.5}$$

Die Klassen sind somit disjunkt und enthalten mindestens ein Muster. Einen Sonderfall bildet die Klasse Ω_0. Sie kann zusätzlich als Rückweisungsklasse für nicht eindeutig klassifizierbare Muster eingeführt werden, so dass sich der Problemkreis dann aus

$$\bigcup_{\kappa=0}^{k} \Omega_\kappa = \Omega \tag{2.6}$$

ergibt. Die Aufteilung des Problemkreises in die jeweiligen Klassen wird anhand einer repräsentativen Stichprobe

$$\omega = \{({}^1\boldsymbol{f}(\boldsymbol{x}), y_1), \ldots, ({}^o\boldsymbol{f}(\boldsymbol{x}), y_o), \ldots, ({}^N\boldsymbol{f}(\boldsymbol{x}), y_N)\} \subset \Omega \tag{2.7}$$

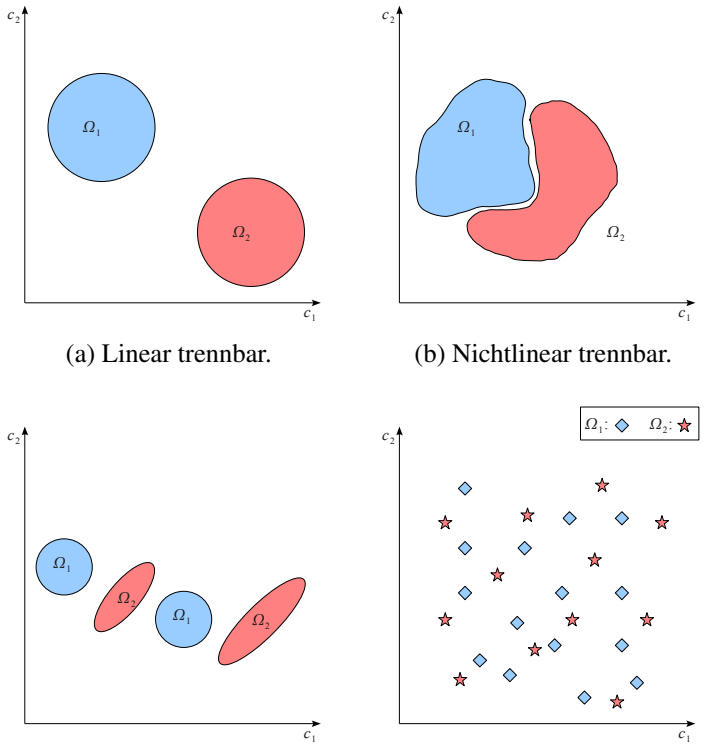
geschätzt. Da in dieser Arbeit die Klassifikation nur im Kontext des überwachten Lernens behandelt wird, ist diese klassifiziert, das heißt jedem Muster aus ω ist seine entsprechende Klasse in Form einer Annotation y_i richtig zugeordnet. Die Gewährleistung der Korrektheit dieser Zuordnung, kann bei „real world“ Daten ein nicht triviales Problem darstellen (vgl. Abschnitt 5.1.1). Neben der Reliabilität der Klassen-Korrespondenzen bedingt die Repräsentativität der Stichprobe die Übertragbarkeit der gewonnenen Erkenntnisse auf unbekannte Muster. Diese Generalisierungsfähigkeit des Klassifikationsystems ist Voraussetzung für die Klassifikation unbekannter Daten, wie sie in dieser Arbeit im Kontext der Pollen-Erkennung behandelt wird. Die Klassifikations-Algorithmen zum überwachten Lernen berechnen also aus einer Stichprobe mit Klassenlabeln y_i eine Funktion (Zielfunktion, engl. target function), mit der die Klassenlabel y_i für nicht in der Stichprobe enthaltene Muster möglichst genau geschätzt werden können.

Um nun Muster einzelnen Klassen zuordnen zu können, müssen sie nach Niemanns Kompaktheitshypothese [Nie07] folgende Kriterien erfüllen:

1. Die Muster verfügen über klassenspezifische Merkmale, welche sie von Mustern anderer Klassen unterscheiden.

2. Diese Merkmale bilden für jede Klasse im Merkmalsraum einen hinreichend kompakten Bereich.

3. Diese Bereiche sind hinreichend voneinander getrennt.



(a) Linear trennbar. (b) Nichtlinear trennbar.

(c) Nichtlinear trennbar. (d) Quasi nicht trennbar.

Bild 2.6: Unterschiedlich kompakte und trennbare Klassen im Merkmalsraum.

Bild 2.6 zeigt Klassen im Merkmalsraum, welche genannte Kriterien unterschiedlich gut erfüllen.

Extrahiert man nun aus den Mustern solche Merkmale, so stellt sich die Klassifikation von Mustern als Abbildung

$$ {}^{\varrho}\boldsymbol{c} \rightarrow \kappa \in \{1, \ldots, k\} \quad \text{oder} \quad {}^{\varrho}\boldsymbol{c} \rightarrow \kappa \in \{0, 1, \ldots, k\} \tag{2.8} $$

dar. ${}^{\varrho}\boldsymbol{c}$ ist hier der aus ${}^{\varrho}\boldsymbol{f}(\boldsymbol{x})$ gewonnene Merkmalsvektor. Die reellwertigen Komponenten c_ν des Vektors repräsentieren die einzelnen Merkmale. Ordnet man nun der Entscheidung für eine Klasse Ω_κ Kosten V zu, so lässt sich die Klassifikation als Optimierungsproblem

$$ \delta^* = \underset{\{\delta\}}{\operatorname{argmin}} V(\delta) \tag{2.9} $$

darstellen. Gesucht wird die Entscheidungsregel (Klassifikator) δ, welche die mittleren Kosten minimiert [Nie07] (vgl. Abschnitt 2.2.2).

Ein Ansatz zur Lösung des komplexen Gesamtproblems der Musterklassifikation besteht darin, es in überschaubare Teilprobleme zu zerlegen und diese für sich sukzessive zu lösen. So existieren bereits viele Verfahren für einzelne Aspekte der Musterklassifikation. Die Kategorisierung dieser Verfahren und die Bezeichnungen der Kategorien fallen in der Literatur nicht einheitlich aus. Eine strikte Unterscheidung ist hier generell schwierig, da die Grenzen zwischen den definierten Bereichen je nach Verfahren fließend verlaufen können. Niemann wählt für die Klassifikation von Mustern ein Modell eines modularen sequentiellen Klassifikationssystems [Nie07]. Einzelne Teilschritte des Klassifikationsprozesses werden hier durch Module repräsentiert. Auf diese Art wird in vielen Veröffentlichungen zur Mustererkennung der Klassifikationsprozess strukturiert, z. B. in [JDM00] und [DHS00] einem Standardwerk der Mustererkennung. Bild 2.7 zeigt ein solches modulares Schema. Hier wurde der Entwurf aus [Nie07] verfeinert und zur zusätzlichen Differenzierung für diese Arbeit neben der Lernphase eine Analysephase eingeführt (vgl. Abschnitt 2.2.1). Die Bezeichnungen der Teilschritte wurden teilweise aus [GGNZ06] übernommen. Optionale Module wurden durch gestrichelte Umrandungen gekennzeichnet. Die fehlenden Erläuterungen zu den verwandten Variablen folgen in den nächsten Abschnitten.

Die Leistung eines solchen modularen Klassifikationssystems lässt sich nun nach verschiedenen Kriterien optimieren. Die Güte des Gesamtsystems – im Sinne dieser Kriterien – ergibt sich hierbei durch das Zusammenspiel der Module, das heißt wie sie gemeinsam auf die Optimierung der gewählten Kriterien hinwirken. Die kompletten Werte aller Zielgrößen stehen allerdings erst am Ende der Prozesskette zur Verfügung, das heißt wenn der Klassifikationsprozess beendet und die Klassen den Mustern zugewiesen wurden. Um jetzt eine optimale Konfiguration aller Module, bezüglich dieser Kriterien zu erhalten, müssten alle möglichen Ausprägungen dieser Konfigurationen durchlaufen werden. Für einzelne Module bestehen bereits Verfahren, die durch entsprechenden Suchstrategien versuchen, möglichst effizient verschiedene Konfigurationen im Kontext des Gesamtsystems zu testen (vgl. Abschnitt 3.1). Ein Verfahren, welches aber nur anhand einer klassifizierten Stichprobe ein vorgegebenes Kriterium für die Leistungsfähigkeit des Klassifikationssystems

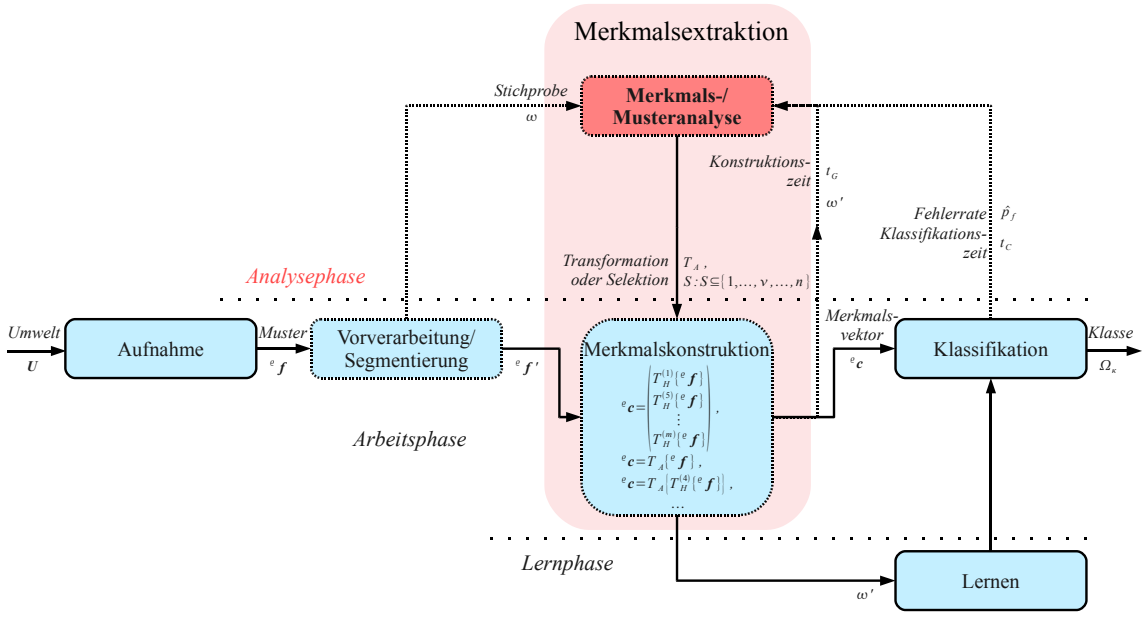


Bild 2.7: System zur Klassifikation von Mustern aus modularer Sichtweise.

optimiert, existiert bis dato nicht [Nie07]. Viele Verfahren optimieren alternativ Gütemaße (Kriterien) auf Modulebene. Sie betrachten so diese Module als eigenständiges Problem unabhängig vom Gesamtsystem. Auch die in dieser Arbeit implementierten Verfahren basieren auf solch einem Prinzip (vgl. Abschnitt 3) wie auch nahezu alle anderen Verfahren aus dem Bereich der Merkmalsextraktion (vgl. Abschnitt 2.2.1). Hierbei gilt es nun die Verfahren zu wählen, deren Gütemaße mit den erwünschten Kriterien für das Gesamtsystem zusammenhängen. Das bedeutet auf Ebene der Merkmalsextraktion z. B. ein Maß für die Kompaktheit der Merkmale zu finden, unter der Annahme, dass durch dessen Maximierung das erwünschte „globale“ Kriterium „niedrige Fehlerrate des Klassifikationssystems“ erreicht wird.

Neben dieser analytischen Methode besteht auch die Möglichkeit nach Erfahrungswerten und Heuristiken eine Vorauswahl der Verfahren und Parameter zu treffen, von denen man vermutet, dass sie im Sinne der erwünschten Kriterien fungieren. Diese Wahl kann dann über ein Feedback durch das Klassifikationssystem validiert werden (siehe Bild 2.7). Methoden aus dem Bereich der Vorverarbeitung/Segementierung funktionieren z. B. nach

diesem Prinzip.

Niemann gibt als Beispiel für Größen zur Optimierung eines Klassifikationssystems unter anderem folgende an:

$$\left[\begin{pmatrix} \text{Klassifikationsfehler} \\ \text{Klassifikationszeit} \\ \text{Lernzeit} \end{pmatrix} \longrightarrow \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \right] , \quad \forall \Omega \, . \tag{2.10}$$

Durch die in Bild 2.7 erstellten Untergliederungen verfeinert und angepasst an die Anforderungen im Rahmen des Pollenmonitor-Projektes ergibt sich daraus:

$$\left[\begin{pmatrix} \textbf{Klassifikationsfehler} \\ \textbf{Arbeitszeit} \\ \text{Analysezeit} \\ \text{Lernzeit} \end{pmatrix} \longrightarrow \begin{pmatrix} \textbf{0} \\ \textbf{0} \\ 0 \\ 0 \end{pmatrix} \right] , \quad \text{für } \Omega_{Pollen} \, , \tag{2.11}$$

Arbeitszeit := Aufnahme + Vorverarbeitung + **Konstruktion** + **Klassifikation** .

(Die priorisierten Kriterien sind hier fett gedruckt.)

Im Folgenden und in den Abschnitten 2.2.1 und 2.2.2 werden nun die Verfahren der einzelnen Module, des in Bild 2.7 aufgezeigten Modells eines Klassifikationssystems, beschrieben.

Aufnahme: Hierunter fällt die Aufnahme des Musters ${}^{\varrho}\boldsymbol{f}(x)$ durch den Sensor sowie die Abtastung (Sampling), Quantisierung, Kodierung und Speicherung. Das auf diese Weise digitalisierte Muster liegt danach in diskreter Form ${}^{\varrho}\boldsymbol{f}$ vor. Im Fall eines digitalen Bildes setzt sich ${}^{\varrho}\boldsymbol{f}$ aus den Abtastwerten ${}^{\varrho}f_{ij}$ zusammen.

In diesem Schritt, als erstes Glied der Prozesskette, wird der Grundstein für die Teilergebnisse aller anderen Module gelegt. Information, welche an dieser Stelle durch zu niedrige Auflösungen nicht erfasst oder z. B. durch verlustbehaftete Komprimierung verlorengeht, kann durch die folgenden Module nicht wiederbeschafft werden. Dieser Sachverhalt gilt generell für alle Module.

Vorverarbeitung/Segmentierung: Verfahren aus diesem Bereich können folgende Ziele verfolgen [Nie07]:

- *Verbesserung des Signal-Rausch-Verhältnisses:* Hierzu existieren unter anderem lineare Operatoren, wie z. B. Faltungen mit verschiedenen Kernen, Fourier-Transformationen, nichtlineare Operatoren, wie Rangordnungsfilter, morphologische Operatoren und binäre Masken sowie verschiedene Schwellwertverfahren.

- *Normierung:* An dieser Stelle sollen Schwankungen der Eingangswerte durch Anpassung an einen Referenzwert oder ein Intervall verkleinert werden, so dass die später extrahierten Merkmale, für die einzelnen Klassen, möglichst kompakte Bereiche im Merkmalsraum bilden und nicht relevante Parameter einen möglichst kleinen Wertebereich oder konstante Werte erhalten. Bei der Normierung ist darauf zu achten, dass für die Klassifikation relevante Unterschiede nicht eliminiert werden, z. B. kann durch eine Größennormierung das vielleicht entscheidende diskriminative Merkmal Größe abhanden kommen.

  Verfahren zur Normierung können zusätzlich auch noch während oder nach der Merkmalsextraktion Anwendung finden (siehe z. B. Abschnitt 3.2).

- *Segmentierung:* Liegen die eingehenden Daten in Form von Bildern vor und befinden sich pro Bild mehrere Muster oder sollen nur bestimmte relevante Bereiche weiterverarbeitet werden, dann kann man diese Gebiete anhand von Bild-Segmentierungs-Algorithmen trennen. Für die weitere Verarbeitung stehen dann segmentierte Regionen zur Verfügungen, die einzeln prozessiert werden können.

  Die separierten Bildregionen der Pollenkörner und anderer Aerosolpartikel, die hier als Grundlage für den in Kapitel 5 verwandten Testdatensatz dienen, wurden z. B. durch ein mehrstufiges Segmentierungsverfahren – basierend auf Hough-Circles und Graph-Cuts – extrahiert.

Nach der Vorverarbeitung liegt das eingegangene Muster ${}^{\varrho}\boldsymbol{f}$ in veränderter Form ${}^{\varrho}\boldsymbol{f}'$ vor.

2.2.1 Merkmalsextraktion

Allen Ansätzen aus diesem Bereich ist gemein, dass hierbei versucht wird aus den eingehenden Daten die trennscharfen Informationen herauszuarbeiten. Trennscharf bedeutet hier, die Informationen, die notwendig sind, um die einzelnen Klassen zu unterscheiden – unter der Voraussetzung, dass diese Informationen überhaupt in den Mustern erfasst wurden. Ein weiterer Gesichtspunkt ist neben dieser Konzentration auf die relevanten Informationen, eine generelle Reduktion der Datendimension und die damit einhergehende Vereinfachung der Datenrepräsentation und des Klassifikators. Ebenso können Effekte der „curse of dimensionality“ (vgl. Abschnitt 2.2.3) durch weniger komplexe Muster gemindert und damit die Generalisierungsfähigkeit des Klassifikationssystems gesteigert und die Klassifikationsfehler reduziert werden [JDM00][Nie07][GGNZ06]. Um nun eine Menge diskriminativer Merkmale zu erhalten, welche die Güte des Gesamtsystems maximieren (vgl. Gleichung 2.10 und 2.11), gibt es verschiedene Vorgehensweisen. Diese lassen sich nach Niemann in Verfahren der Merkmalsgewinnung und Merkmalsauswahl einteilen. Die folgenden Verfahren arbeiten allesamt unter der Prämisse die Faktoren Fehlerwahrscheinlichkeit des Klassifikators und Merkmalsanzahl zu minimieren.

Merkmalsgewinnung: Gesucht wird eine Transformation T_r, welche ein Muster ${}^{\varrho}\boldsymbol{f}$ in einen Vektor ${}^{\varrho}\boldsymbol{c}$ von Merkmalen c_ν überführt, durch

$$ {}^{\varrho}\boldsymbol{c} = ({}^{\varrho}c_1, \ldots, {}^{\varrho}c_n)^{\mathrm{T}} = T_r \left\{ {}^{\varrho}\boldsymbol{f} \right\} \ . \tag{2.12} $$

Diese Transformation kann auf zwei verschiedene Herangehensweisen bestimmt werden:

- *heuristisch:* Bei diesem Ansatz werden Merkmale nach Erfahrungwerten ausgewählt. Eine erste Überprüfung der Auswahl kann durch die Klassifikationsergebnisse erfolgen. Transformationen T_H aus diesem Bereich werden entweder auf das ganze Muster angewandt und erzeugen so „globale“ Merkmale, oder es werden für einzelne Stellen des Musters „lokale“ Merkmale generiert.

  Sind problemspezifische Invarianten erforderlich, können Merkmale verwandt werden, die diese Invarianten (z. B. Rotationsinvarianten oder Skalierungsinvarianzen) aufweisen. Beschränkt man sich auf Muster, die als digitale Bil-

der vorliegen, so kann man die aus ihnen erzeugbaren Merkmale in Kontur- und Textur-Merkmale einteilen. Kontur-Merkmale werden im Gegensatz zu Textur-Merkmalen nur auf der Konturlinie des Musters berechnet.[4]

In Abschnitt 5.1.1 werden entsprechende Merkmale aus dem Testdatensatz aufgelistet, der zur Durchführung der in Kapitel 5 beschriebenen Experimente verwandt wurde.

- *analytisch:* Hier besteht der Unterschied zu den oben genannten heuristischen Methoden darin, dass die Transformationen T_A unter Optimierung einer zuvor gewählten Gütefunktion, auf einer Stichprobe von Mustern ω bestimmt werden. Da diese Transformationen dadurch von der Stichprobe abhängen, bezeichnet Nieman sie als problemabhängig. Ein weiterer Unterschied besteht darin, dass diese Verfahren nicht nur auf die Abtastwerte der Muster ${}^{\varrho}\boldsymbol{f}$ angewandt werden, sondern auch auf bestehende Merkmalsvektoren ${}^{\varrho}\boldsymbol{c}$, um daraus neue Merkmale im oben genannten Sinne zu generieren.

  Um nun die Transformationen T_A zu finden, wird meist nicht die Fehlerwahrscheinlichkeit direkt, sondern ein Kriterium auf Merkmalsebene unabhängig vom Klassifikator optimiert. Der Vorteil dabei ist, dass solche Kriterien einfacher und mit weniger Aufwand zu berechnen sind [Nie07]. Gute „lokale“ Kriterien sind solche, die mit der Fehlerwahrscheinlichkeit des Klassifikators korrelieren. Hierbei gilt: Je stärker der Zusammenhang zwischen dem Kriterium und der Fehlerwahrscheinlichkeit, desto komplexer die Berechnung des Kriteriums. Niemann leitet aus der oben genannten Kompaktheitshypothese für Merkmale vier Gütekriterien ab, die auf dem Quadrat der Euklidschen Metrik basieren:

  1. Mittlerer quadratischer Abstand zwischen allen Merkmalen, definiert durch

$$s_1 = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{j=1}^{N} \left({}^{i}\boldsymbol{c} - {}^{j}\boldsymbol{c}\right)^{\mathrm{T}} \left({}^{i}\boldsymbol{c} - {}^{j}\boldsymbol{c}\right) \ . \tag{2.13}$$

  2. Mittlerer quadratischer Abstand aller Merkmale einer Klasse von den Merkmalen einer anderen Klasse.

[4] Ein ausführlicher Überblick über verschiedene Textur-Merkmale wird in [TJ93] gegeben.

3. Mittlerer quadratischer Abstand der Merkmale einer Klasse.
4. Mittlerer quadratischer Abstand der Klassenzentren.

Das zweite und das dritte Kriterium werden auch als *Interklassenabstand* bzw. *Intraklassenabstand* bezeichnet. Man ist bestrebt die Kriterien 1, 2 und 4 zu maximieren und Kriterium 3 zu minimieren. Es können nun orthogonale lineare[5] Transformationen der Art

$$\boldsymbol{c} = \boldsymbol{\Phi}\boldsymbol{f} \tag{2.14}$$

gefunden werden, die für eine feste Anzahl von Merkmalen n eines oder eine Kombination dieser Kriterien optimieren.

Eine andere Sicht auf diese Problematik besteht darin, die Muster so zu betrachten, als wären sie in einen höherdimensionalen Raum eingebettet [MPH07] [GGNZ06]. Ziel hierbei ist es, die Abbildung zu finden, welche die Merkmale aus diesem Raum in den n-dimensionalen Unterraum überführt, der von den Merkmalen tatsächlich eingenommen wird. Dies wird auch die intrinsische Dimensionalität der Daten genannt [MPH07][JDM00]. Unter diesem Paradigma existieren neben den klassischen linearen Verfahren auch nichtlineare Verfahren, wie z. B. die auf gewichteten Graphen basierenden Isomaps [MPH07] oder die kernbasierte Hauptachsentransformation – die nichtlineare Erweiterung der weit verbreiteten Hauptachsentransformation (engl. principal component analysis, PCA) [Nie07]. Die Hauptachsentransformation ist zudem ein Beispiel für eine orthogonale Abbildung, welche Kriterium 1 (vgl. Gleichung 2.13) maximiert (ein simples Beispiel zur Hauptachsentransformation zeigt Bild 2.8).

Die Verfahren der analytischen Merkmalsgewinnung werden in der Literatur auch unter dem Oberbegriff der Dimensionsreduktion (engl. dimensionalty reduction) geführt, wobei hierunter meist Verfahren fallen, die keinen Gebrauch von Klasseninformationen machen. Das heißt z. B. keine Methoden, die auf den Kriterien 2–4 beruhen.[6] In einigen Veröffentlichungen wird auch die Pro-

[5] Niemann bezieht sich hier nur auf lineare Transformationen, wobei er hierzu auch die Verfahren zählt, die sich einer Kernfunktion [Nie07] bedienen und somit in dem Parameter $\boldsymbol{f}$ nicht linear sind.

[6] Eine Zusammenfassung solcher Verfahren zur linearen und nichtlinearen Dimensionsreduktion gibt [MPH07].

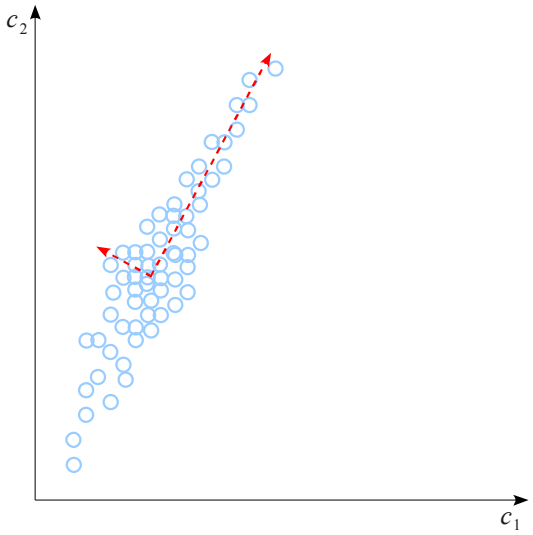


Bild 2.8: Streudiagramm zweier korrelierter Merkmale mit eingezeichneten Hauptachsen. Die intrinsische Dimensionalität der Daten scheint hier 1 zu sein. Das neue Merkmal c' ergibt sich nach Projektion auf die längere Hauptachse $\boldsymbol{\varphi_1}$ durch $c' = \boldsymbol{\varphi_1}(c_1, c_2)^{\mathrm{T}}$.

zedur der analytischen Merkmalsgewinnung direkt als Merkmalsextraktion bezeichnet [JDM00] und mit den Methoden der Merkmalsselektion unter dem Begriff der Dimensionsreduktion geführt. Andere ordnen wiederum die Transformationen der analytische Merkmalsgewinnung dem Schritt „Vorverarbeitung“ zu [GGNZ06].[7]

Merkmalsbewertung und –auswahl: Im Gegensatz zu den Verfahren der Merkmalsgewinnung werden hier keine neuen Merkmale erzeugt, sondern aus einer vorgegebenen Menge von Merkmalen soll hier eine möglichst „gute“ Untermenge ausgewählt werden. Die Merkmale hierzu stammen meist aus der heuristischen Merkmalsgewinnung. Prinzipiell ist es aber auch möglich Merkmale über analytische

[7]Zur Begründung dieser Entscheidung lässt sich z. B. die Hauptachsentransformation nennen. Was die Vorverarbeitungsschritte zur Verbesserung des Signal-Rausch-Verhältnisses auf Ebene der einzelnen Muster leisten, leistet sie, durch Maximierung des Kriteriums 1 (vgl. Gleichung 2.13), im Merkmalsraum.

Methoden zu erzeugen und an dieser Stelle noch eine zusätzliche Datenreduktion zu versuchen (genauso kann auch umgekehrt verfahren werden, das heißt auf eine ausgewählte Teilmenge von Merkmalen können zusätzlich analytische Verfahren der Merkmalsgewinnung angewandt werden, die Kombinationsmöglichkeiten sind in Bild 2.7 durch entsprechende Schleifen dargestellt). Da hier die Merkmale in ihrer ursprünglichen Form erhalten werden und nicht durch eine lineare oder nichtlineare Kombination einzelner Merkmale (bedingt durch eine Transformation T_A) ersetzt werden, bleiben sie interpretierbar und behalten ihre Aussagekraft. Ferner entstehen hier zusätzlich Laufzeit-Vorteile für das Klassifikationssystem während der Arbeits- und Lernphase (siehe Bild 2.7), da durch die Selektion weniger Merkmale im Modul „Merkmalskonstruktion“ berechnet werden müssen. Eine solche Merkmalsauswahl S kann formal als Teilmenge der ursprünglichen Indizes der Merkmalsvektoren betrachtet werden. Somit ergibt sie sich zu:

$$S \subset \{1, \ldots, \nu, \ldots, n\} \ . \tag{2.15}$$

Verfahren, die nun versuchen solche Teilmengen von Merkmalen zu finden, basieren wie die Methoden der analytischen Merkmalsgewinnung auf der Analyse einer klassifizierten Stichprobe und entsprechenden Gütemaßen. Ein Unterschied zu den oben genannten Methoden besteht darin, dass die Klasseninformationen Grundvoraussetzung für die Merkmalsbewertung sind.

Neben der Fehlerwahrscheinlichkeit werden bei der Merkmalsselektion ebenfalls „lokale“ Gütemaße zur Bewertung der Merkmale verwandt, welche unabhängig vom Klassifikator auf der Stichprobe berechnet werden können.

Eine Zusammenfassung der verschieden Bewertungs- und Selektionsverfahren, sowie die Beschreibung der in dieser Arbeit implementierten Verfahren wird in Abschnitt 3.1 gegeben.

In Bild 2.7 wurde der Versuch unternommen den gesamten Klassifikationsprozess anhand eines modularen Klassifikationssystems zu beschreiben. Einer der Unterschiede im Vergleich zu der von Niemann in [Nie07] vorgestellten Strukturierung, ist hier die Einführung einer zusätzlichen Analysephase. Diese dient dazu die Verfahren der analytischen Merk-

malsgewinnung sowie die Verfahren der Merkmalsselektion differenzierter in das Klassifikationssystem einzuordnen. In dieser Phase findet analog zur Lernphase des Klassifikators offline eine Analyse des Daten-/Merkmalsraumes statt. Die hieraus gewonnen Transformationen T_A und/oder Selektion S finden dann während (Transformationen), respektive vor (Selektionen) der Arbeitsphase, Anwendung. Um die Leistung des Klassifikationssystems während des Routinebetriebs (Arbeitsphase) zu verbessern, können so parallel zur Lernphase auch immer wieder neue Analysephasen angestoßen werden.

Viele praktische Anwendungen aus dem Bereich der Mustererkennung sind auf diese Art strukturiert [Nie07]. Auch die im „Pollenmonitor-Projekt“ entwickelte Bildanalyse-Software weist die entsprechenden Phasen und Module auf. Bei einigen Verfahren sind aber solche Modulgrenzen nur schwer oder gar nicht zu ziehen, besonders bei dem in diesem Abschnitt behandelten Teilbereich der Merkmalsextraktion. So kann z. B. bei der Klassifikation mit künstlichen Neuronalen-Netzen in so genannten „hidden units“ eine analytische Merkmalsgewinnung während der Lernphase des Netzes stattfinden [GGNZ06]. Das in Bild 2.7 vorgestellte Modell gilt daher primär zur Orientierung und groben Kategorisierung der einzelnen Verfahren.

2.2.2 Klassifikation

Nachdem das Muster ${}^{\varrho}\boldsymbol{f}(\boldsymbol{x})$ zuerst durch geeignete Sensoren in digitale Abtastwerte ${}^{\varrho}\boldsymbol{f}$ überführt wurde, um dann aus diesen einen Merkmalsvektor ${}^{\varrho}\boldsymbol{c}$ zu extrahieren, besteht der letzte Schritt des Klassifikationsprozesses nun in der Zuweisung des Merkmalsvektors zu seiner korrespondierenden Klasse Ω_κ. Dieser Schritt wird als numerische Klassifikation bezeichnet [Nie07], da hier nur reellwertige (keine symbolischen, nominalen) Merkmalsvektoren zum Einsatz kommen.

Betrachtet man die Klassifikation eines Musters als Problem der statistischen Entscheidungstheorie, so folgt hier auf Basis der Beobachtung eines Musters, respektive einses Merkmalsvektors ${}^{\varrho}\boldsymbol{c}$, eine Entscheidung für eine Klasse. Die Muster sind hierbei Ergebnisse eines Zufallsexperimentes in Form von Realisationen $(\kappa, {}^{\varrho}\boldsymbol{c})$ einer mehrdimensionalen Zufallsvariablen $\boldsymbol{c}$ mit der bedingten Dichte $p(\boldsymbol{c} \,|\, \Omega_\kappa)$ [Nie07]. Geht man von bekannten a priori Wahrscheinlichkeiten der Klassen $p(\Omega_\kappa) = p_\kappa$ und bekannten bedingten Vertei-

lungsdichten der Merkmale aus, so erhält man über das Bayestheorem

$$p(\Omega_\kappa \mid \boldsymbol{c}) = \frac{p_\kappa p(\boldsymbol{c} \mid \Omega_\kappa)}{p(\boldsymbol{c})} \tag{2.16}$$

die a posteriori Wahrscheinlichkeiten der Klassen $p(\Omega_\kappa \mid {}^{\varrho}\boldsymbol{c})$ nach Beobachtung eines Merkmalsvektors ${}^{\varrho}\boldsymbol{c}$. Aus dieser Gleichung geht unmittelbar der Bayes-Klassifikator hervor, der sich für die Klasse Ω_κ mit der größten a posteriori Wahrscheinlichkeit entscheidet. Er ist definiert durch:

$$\kappa = \underset{\kappa \in \{1,\ldots,k\}}{\operatorname{argmax}} p(\Omega_\lambda \mid \boldsymbol{c}) = \underset{\kappa \in \{1,\ldots,k\}}{\operatorname{argmax}} p_\lambda p(\boldsymbol{c} \mid \Omega_\lambda) \ . \tag{2.17}$$

Dieser Klassifikator ist optimal im Sinne der Fehlerwahrscheinlichkeit, das heißt wählt man für die Kosten V in Gleichung 2.9 die Fehlerwahrscheinlichkeit, minimiert der Bayes-Klassifikator die mittleren Kosten.

Würde man die bedingten Verteilungsdichten der Merkmalsvektoren und die a priori Wahrscheinlichkeiten der Klassen kennen, wäre das Klassifikationsproblem nicht nur mathematisch sondern auch praktisch gelöst. Bei „real world“ Daten sind diese Größen aber unbekannt und können nur durch möglichst repräsentative endliche Stichproben geschätzt werden. Klassifikatoren, die auf solchen Schätzungen der klassenbedingten Verteilungsdichten der Merkmalsvektoren beruhen, werden als statistische Klassifikatoren bezeichnet. Die Schätzung der Verteilungsdichten ist hier aber nur unter bestimmten Prämissen möglich. Vor der Schätzung muss das statistische Modell gewählt werden, dessen Parameter man bestimmen möchte. Hier ist die Familie der Normalverteilungen der wichtigste und weit verbreitetste Vertreter [Nie07]. Meistens werden zusätzlich noch Annahmen über die Parameter und deren Beziehungen zueinander gemacht, um den Schätzprozess zu vereinfachen. So kann z. B. eine klassenweisen statistischen Unabhängigkeit der Merkmale angenommen werden, wodurch sich die Verteilungsdichte der Merkmalsvektoren als folgendes Produkt darstellen lässt:

$$p(\boldsymbol{c} \mid \Omega_\kappa) = \prod_{\nu=1}^{n} p(c_\nu \mid \Omega_\kappa) \ . \tag{2.18}$$

Dadurch wird das Problem also auf die Schätzung von univariaten klassenbedingten Verteilungsdichten reduziert.

Das Verfahren der linearen Diskriminanzanalyse, welches in modifizierter, hierarchischer Form dem „Pollen-Monitor“-Projekt als Klassifikator dient, ist ein Beispiel für einen einfachen statistischen Klassifikator. Es steht im engen Zusammenhang mit der Fisherschen Diskriminanzanalyse, die das Fishersche Kriterium

$$J(\boldsymbol{w}) = \frac{(m_1 - m_2)^2}{{s_1}^2 + {s_2}^2} \tag{2.19}$$

optimiert [Bis06]. Gesucht ist ein Projektionsvektor $\boldsymbol{w}$, der die *Intraklassenvarianz* s^2 der Klassen Ω_1 und Ω_2 minimiert und die *Interklassenvarianz* $(m_1 - m_2)^2$ maximiert. Dieses Kriterium kann auch als Quotient der Kriterien 2 und 3 aus Abschnitt 2.2.1 betrachtet werden. Somit handelt es sich hierbei auch um ein Verfahren zur Dimensionsreduktion.

Die Klassifikation findet dann anhand des in der Trainingsphase ermittelten Projektionsvektors $\boldsymbol{w}$ über das Skalarprodukt $y = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{c}$ statt. Durch einen Schwellwert γ_0 wird dann die Entscheidung für eine der beiden Klassen Ω_1 oder Ω_2 getroffen [Bis06]

$$\kappa = \begin{cases} 1 & : y > \gamma_0, \\ 2 & : y \leq \gamma_0 \ . \end{cases} \tag{2.20}$$

Um nun die Leistung eines Klassifikators zu bewerten, wird meist eine Spezialisierung des Kostenbegriffes in Form der Fehlerwahrscheinlichkeit vorgenommen. Für den Bayes-Klassifikator lässt sie sich bei vollständig bekannten Verteilungsdichten durch Integration berechnen. In Bild 2.2.2 ist sie durch den schraffierten Bereich gekennzeichnet. In der Praxis ist nur eine Schätzung möglich. Diese wird anhand des Klassifikators und eines klassifizierten Testdatensatzes durchgeführt und ergibt sich zu:

$$\hat{p}_f = \frac{\text{Zahl der falsch klassifizierten Muster}}{\text{Gesamtzahl der klassifizierten Muster}} \tag{2.21}$$

Die so geschätzte Fehlerwahrscheinlichkeit wird Fehlerrate $\hat{p}_f$ genannt. Die Zuverlässigkeit dieser Schätzung hängt von der Komplexität des Klassifikators und der Größe der Stichprobe ab (siehe Abschnitt 2.2.3).

Neben den statistischen Klassifikatoren existieren viele andere leistungsfähige Ansätze zur Lösung des Problems der Musterklassifikation, die nicht auf statistischen Modellen basieren. Hierzu zählen unter anderem die Support Vektor Maschinen (SVM), neuronale Netze und Polynomklassifikatoren [Nie07].

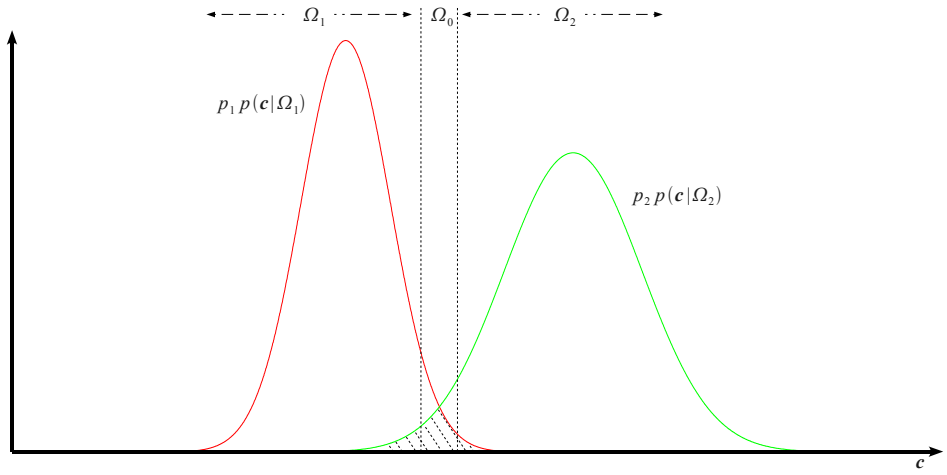


Bild 2.9: Überlappende univariate Normalverteilungen eines Merkmales für zwei Klassen Ω_1 und Ω_2. Der schraffierte Bereich stellt die Fehlerwahrscheinlichkeit dar. Die gestrichelten Linien markieren einen möglichen Bereich für die Rückweisungsklasse Ω_0.

2.2.3 Dimensionierungsprobleme

Betrachtet man das Problem der Klassifikation unter dem Ansatz der statistischen Entscheidungstheorie, so lässt sich theoretisch nachweisen, dass durch Hinzunahme neuer Merkmale die Fehlerwahrscheinlichkeit nur verringert werden kann. Je diskriminativer diese neuen Merkmale sind, umso stärker verringert sich die Fehlerwahrscheinlichkeit. Dies lässt sich z. B. exemplarisch über den Mahalanobis-Abstand (vgl. Gleichung 3.2) zwischen den einzelnen Klassen zeigen [DHS00]. In der Praxis verhält es sich aber oft genau konträr [Nie07] [DHS00]. Durch Hinzunahme neuer Merkmale bei konstanter Stichprobengröße kann sich die Fehlerwahrscheinlichkeit erhöhen. Dieses Verhalten fällt unter das Problem der „curse of dimensionality“[JDM00]. Es lässt sich dadurch erklären, dass die vollständige Kenntnis des statistischen Modells des Problemkreises vorausgesetzt wird, welches aber in der Praxis anhand einer endlichen Stichprobe nur unter Einschränkungen geschätzt werden kann. Diese Schätzfehler werden größer je schlechter das Verhältnis zwischen Stichprobengröße und Dimension des Merkmalsraumes ausfällt. Tastet man z. B. einen n-dimensionalen Merkmalsraum in äquidistanten Schritten in allen Dimensionen ab,

hängt die dafür benötigte Anzahl an Samples exponentiell von der Anzahl der Merkmale ab [Bis06]. Es existiert eine Vorgabe aus Erfahrungswerten [JDM00] für das Verhältnis zwischen Größe der Stichprobe N und Anzahl der Merkmale n zu:

$$\frac{N}{n} > 10 \; . \tag{2.22}$$

Auch die Zuverlässigkeit der Schätzung der Fehlerwahrscheinlichkeit basiert auf diesem Verhältnis [Nie07].

Ein anderes Phänomen, welches man auch unter diesem Aspekt der „curse of dimensionality“ sehen kann, ist das Problem der Überanpassung (overfitting). Häufig wird dies aber im Kontext des maschinellen Lernens als eingenständiges Problem betrachtet [JDM00] [DHS00]. So kann durch einen zu komplexen Klassifikator, bedingt auf Grund zu vieler Parameter (Merkmale), in der Lernphase eine starke Spezialisierung auf die Trainingsdaten stattfinden, die zu einer schlechten Generalisierung auf unbekannten Daten führt [DHS00][JDM00]. Die Gefahr besteht darin, den Klassifikator durch Hinzunahme neuer Parameter so komplex zu gestalten, dass er in seiner Komplexität zwar die Trainingsdaten trennen kann, dies aber für andere Daten nicht mehr leistet.

Kapitel 3

Merkmalsbewertung und Selektion

In dieser Arbeit wird ein Verfahren der Merkmalsselektion zur Reduktion der Dimension des Datenraumes verwandt. Die Nutzung einer Methode aus diesem Bereich wird durch mehrere Faktoren motiviert.

1. Im Rahmen des Pollenmonitor-Projektes wird durch das Klassifikationssystem bereits ein fester Merkmalssatz anhand von Verfahren der heuristischen Merkmalsgewinnung (vgl. Abschnitt 2.2.1) erzeugt. Durch Merkmalsauswahl bleiben hier die Einzel-Merkmale bestehen und werden nicht wie bei Verfahren zur analytischen Merkmalsgewinnung über eine Transformation T_A linear oder nicht-linear zu neuen Merkmalen kombiniert. Auf diese Weise behalten die Merkmale ihre Aussagekraft und Interpretierbarkeit. Durch diesen Umstand können die zuvor gewählten Heuristiken bewertet und Erkenntnisse über die Struktur der zu klassifizierenden Objekte gewonnen werden. Aus diesen Erkenntnissen könnten wiederum neue Heuristiken zur Merkmalserzeugung abgeleitet werden. Im Kontext der Pollen-Klassifikation ist das ein wichtiger Aspekt, da hier ständig mit neuen Pollen-Taxa zu rechnen ist, die jeweils spezifische Strukturen aufweisen (vgl. Abschnitt 2.1.1).

2. Durch die Auswahl einer Merkmalsteilmenge entstehen neben kürzeren Laufzeiten des Klassifikations-Algorithmus auch Geschwindigkeitsvorteile bei der Merkmalskonstruktion. Eine Transformation T_A würde hingegen an dieser Stelle je nach Kom-

plexität einen zusätzlichen Rechenaufwand nach sich ziehen.

3. Der genutzte hierarchische Klassifikator leistet durch Verwendung der linearen Diskriminanzanalyse bereits während des Klassifikationsschrittes eine Dimensionsreduktion im Sinne der analytischen Merkmalsgewinnung (vgl. Abschnitt 2.2.2).

Auch wenn durch lineare und im Speziellen nicht-lineare Transformationen des Merkmalsraumes Merkmalsvektoren gewonnen werden können, die anderes diskriminatives Potential besitzen als Vektoren, die sich aus einer „besten“ Untermenge der vorhandenen Merkmale zusammensetzen, hat sich in der Praxis die für diese Arbeit gewählte Kombination aus Transformationen zur heuristischen Merkmalsgewinnung T_H und Merkmalsselektion ebenfalls bewährt und als sehr leistungsfähig bewiesen [Nie07].

Methoden zur Merkmalsselektion als Teilbereich der Merkmalsextraktion verfahren nach folgendem Prinzip: Sie suchen innerhalb einer vorgegebenen Menge von Merkmalen nach möglichst trennscharfen Merkmalssätzen, das heißt Teilmengen, welche nur Merkmale enthalten, die für das Zielkonzept von Relevanz sind, also den Problemkreis beschreiben. Eine solche Merkmalsselektion S (vgl. Gleichung 2.15) ist nach [KS96] optimal, wenn sie die kleinste Merkmalsteilmenge ist, für deren Merkmalsvektoren $\boldsymbol{c}'$ gilt:

$$P(\Omega_\kappa \,|\, \boldsymbol{c}') = P(\Omega_\kappa \,|\, \boldsymbol{c}), \quad \forall \Omega_\kappa \in \Omega \; . \tag{3.1}$$

Dies bedeutet, die a posteriori Wahrscheinlichkeiten der Klassen bleiben für diese reduzierten Merkmalsvektoren $\boldsymbol{c}'$ unverändert. Es geht also durch die Auswahl keine diskriminative Information über den Problemkreis Ω verloren, sondern es werden nur irrelevante und redundante Merkmale entfernt. Das Problem der Merkmalsauswahl besteht darin unter den vielen möglichen Selektionen S diejenige Selektion S_τ zu finden, die diese optimale Untermenge darstellt. Diese Suche stellt auf Grund der Größe des Suchraumes kein triviales Problem dar. Geht man davon aus, dass die Merkmale nicht statistisch unabhängig sind, was in der Praxis meistens der Fall ist [Nie07], müssen alle möglichen Merkmalsteilmengen auf das Kriterium aus Gleichung 3.1 geprüft werden. Bei einer Merkmalsanzahl n sind das 2^n mögliche Teilmengen, selbst bei Einschränkung auf eine feste Größe der Merkmalsauswahl $|S| = n'$ verbleiben noch $\binom{n}{n'}$ Teilmengen. So würden z. B. allein die 75 ursprünglichen Merkmale, welche in dem Pollenmonitor-Testdatensatz enthalten sind,

bei einer Begrenzung der Auswahlgröße auf 20 Merkmale zu ca. $8 \cdot 10^{17}$ Selektionen führen.

Im Folgenden wird ein Überblick zu Lösungsansätzen des Problems der Merkmalsselektion gegeben. Anschließend wird ein ausgewählter Ansatz, die Familie der Relief-Verfahren, dargestellt. In Abschnitt 3.3 werden die im Rahmen dieser Arbeit vorgenommenen Modifikationen und Beschleunigungen dieses Verfahrens dargelegt sowie die darauf basierende entwickelte Methode zur Redundanzerkennung beschrieben.

3.1 Verfahren

Das in Gleichung 3.1 vorgestellte Kriterium für eine optimale Merkmalsteilmenge ist in der Praxis nicht zu prüfen, da hier das statistische Modell des Problemkreises nicht bekannt ist und nur geschätzt werden kann. So ist lediglich eine optimale Teilmenge in Bezug auf andere Kriterien zu finden, welche im besten Fall möglichst stark mit der optimalen Teilmenge im Sinne von Gleichung 3.1 übereinstimmt. Auf diese Weise erweitert sich das Problem der Merkmalsselektion neben der Bestimmung einer geeigneten Suchstrategie um das Auffinden geeigneter Gütekriterien. Die bis dato entwickelten Kriterien lassen sich nach [DL97] in folgende Kategorien einteilen:

1. Abstandsmaße: Hier werden Abstände zwischen den geschätzten klassenbedingten Verteilungsdichten der Merkmalsvektoren zweier Klassen Ω_κ und Ω_λ bestimmt. Ein bekannter, numerisch leicht zu berechnender Vertreter ist hier der Mahalanobis-Abstand [Nie07]. Er setzt normalverteilte Merkmalsvektoren mit Vektoren der Erwartungswerte zu $\boldsymbol{\mu_\kappa}$ und $\boldsymbol{\mu_\lambda}$ sowie Gleichheit der bedingten Kovarianzmatrizen zu $\boldsymbol{\Sigma_\kappa} = \boldsymbol{\Sigma_\lambda} = \boldsymbol{\Sigma}$ voraus. Der Mahalanobis-Abstand ergibt sich dann aus

$$d(\kappa, \lambda) = (\boldsymbol{\mu_\kappa} - \boldsymbol{\mu_\lambda})\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu_\kappa} - \boldsymbol{\mu_\lambda}) \; . \tag{3.2}$$

Prinzipiell kann auch z. B. der Interklassenabstand (Kriterium 2 aus Abschnitt 2.2.1) hier dazu gezählt werden, obwohl dies ein geometrischer und kein probabilistischer Ansatz ist. Vergleicht man über diese Abstandmaße Merkmalsvektoren, die aus zwei

Merkmalsselektionen S' und S'' zu $\boldsymbol{c}'$ und $\boldsymbol{c}''$ hervorgehen, so ist Selektion S' eine bessere im Sinne dieser Gütemaße, falls für deren Merkmalsvektoren der Abstand $d'(\kappa, \lambda)$ größer ausfällt als für die Merkmalsvektoren der Selektion S'', also $d'(\kappa, \lambda) > d''(\kappa, \lambda)$.

2. Maße aus der Informationstheorie: Diese Maße zeigen auf wieviel Information über eine Klasse Ω_κ in einem Merkmalssatz, respektive einem einzelnen Merkmal, beinhaltet ist. Die Transinformation (engl. mutual information) ist an dieser Stelle ein häufig verwandtes Maß [GGNZ06][PLD05][KC02]. Sie besagt dabei wie stark die Merkmalsvektoren $\boldsymbol{c}$ einer Selektion S z. B. mit einer Klasse Ω_κ zusammenhängen. Das bedeutet, sind diese statistisch unabhängig, ist deren Transinformation null, also ist die Merkmalsselektion die schlecht-möglichste. Die Transinformation ist definiert zu:

$$I = -\sum_{\kappa=1}^{\kappa} \int p(\boldsymbol{c}, \Omega_\kappa) \ln \frac{p(\boldsymbol{c}, \Omega_\kappa)}{p(\boldsymbol{c}, p_\kappa)} \mathrm{d}\boldsymbol{c} \; . \tag{3.3}$$

Die Ermittlung dieser Größe kann in der Praxis z. B. über eine Parzen-Schätzung erfolgen [Nie07][KC02]. Man erhält hier im Gegensatz zu den oben erwähnten Abstandsmaßen auch Werte für mehr als zwei Klassen, kann diese also auch direkt bei Multi-Klassen-Problemen anwenden.

3. Abhängigkeitsmaße: Über die Methode der Korrelation kann hier z. B. durch den Korrelationskoeffizienten (siehe Gleichung 3.6) ein Maß für den linearen Zusammenhang zwischen einem Merkmal c_ν und einer Klasse Ω_κ berechnet werden [Nie07] [GGNZ06]. Diese Methode lässt sich nur auf binäre Klassifikationsprobleme anwenden.

4. Konsistenzmaße: Diese Maße beruhen auf der Vereinbarung, dass ein konsistenter Merkmalssatz keine Merkmalsvektoren enthalten darf, die für die gleichen Merkmalswerte unterschiedliche Klassenlabel besitzen. Unter Zugeständnis einer gewissen Inkonsistenz-Rate werden minimale Merkmalssätze ermittelt, welche diese nicht überschreiten [DL97] [DLM00] [DL03].

5. Fehlerrate des Klassifikators: Dieses Gütemaß unterscheidet sich von den vorangehenden dadurch, dass zu dessen Erstellung der Klassifikationsalgorithmus verwandt

wird. Verfahren, die unter Optimierung dieses Kriteriums den „besten“ Merkmalssatz bestimmen, sind dementsprechend abhängig von dem verwandten Klassifikator, liefern aber so für diesen die bestmöglichen Ergebnisse [DL97], unter der Voraussetzung einer geeigneten Suchstrategie. Methoden, die sich dieses Maßes bedienen, werden *Wrapper*-Methoden genannt und besitzen wegen direkter Verwendung des Klassifikators die höchste Rechenkomplexität [Nie07]. Eine Übersicht dieser Methoden wird in [KJ97]gegeben.

Methoden die unter Verwendung der Maße 1–4 arbeiten werden im Gegensatz zu den Verfahren, die unter Nutzung des Klassifikations-Algorithmus direkt dessen Fehlerrate optimieren, *Filter*-Methoden genannt [DL97]. In [Duc06] findet sich dazu eine ausführliche Zusammenfassung mehrerer Ansätze aus diesem Bereich. Die Berechnung der Kriterien für die Filter-Methoden ist meistens weniger komplex als die Schätzung der Fehlerrate. Die verwandten Kriterien stellen hierbei ein vom Klassifikations-Algorithmus unabhängiges Maß für die Struktur des Merkmalsraumes dar. Eine verbreitete Variante der Filter-Methode sind die Ranking-Verfahren [GGNZ06]. Häufig werden diese Bezeichnungen auch synonym gebraucht. Bei dieser Variation wird über das entsprechende Kriterium ein Relevanz-Index erstellt, der jedem einzelnen Merkmal eine Relevanz zuordnet. Ist das Kriterium „gut“ gewählt, so ist diese Relevanz auch mit der „tatsächlichen“ Relevanz – im Sinne der optimalen Merkmalsmenge (vgl. Gleichung 3.1) – positiv korreliert [GGNZ06]. Ranking-Verfahren können z. B. auf Abstands- oder Korrelationsmaßen basieren. Bei den Abstandsmaßen bestimmt man hier die Abstände nicht für ganze Merkmalssätze, sondern für jedes Merkmal einzeln, um das Ranking zu erhalten [Nie07]. Es existieren aber auch multivariate Ranking-Verfahren, welche die Merkmale nicht unabhängig betrachten, sondern im Kontext, wie z. B. das in dieser Arbeit verwandte Relief-Verfahren (vgl. Abschnitt 3.2).

Für sowohl Filter- als auch Wrapper-Methoden werden Suchstrategien benötigt, um möglichst optimale Merkmalssätze zu finden. Die Ranking-Verfahren stellen hier einen Spezialfall dar, da hier die „optimale“ Merkmals-Teilmenge über Schwellwerte auf Ebene des Relevanz-Kriteriums bestimmt werden kann. Merkmale, deren Gütemaß unter diesem Schwellwert liegt, werden so *ausgefiltert*. Es existiert eine Vielzahl an heuristischer und systematischer Suchverfahren zur Merkmalsselektion. Sie sind aber für diese Arbeit nicht

von Relevanz, da hier durch Verwendung eines Ranking-Verfahrens (vgl. Abschnitt 3.2) aus oben genannten Gründen keine komplexe Suchstrategie benötigt wird. Hier kommt lediglich ein simples Auswahlverfahren zur Anwendung, welches in Abschnitt 5.2 beschrieben wird.

3.2 Relief-Verfahren

Mit der Familie der Relief-Verfahren wurde hier ein klassischer Vertreter der Ranking-Algorithmen gewählt. Die Entscheidung fiel auf ein Ranking-Verfahren, da diese Verfahren anhand des jeweiligen Kriteriums für jedes Merkmal Bewertungen vergeben. So ist hier, im Gegensatz zu anderen Filter- oder Wrapper-Verfahren, welche die Qualität einzelner Merkmalssätze beurteilen, die Qualität jedes einzelnen Merkmales ersichtlich. In Folge dessen können diese miteinander verglichen und Rückschlüsse auf die Struktur der zugrunde liegenden Muster gezogen werden. Ferner sind die Ranking-Methoden als Spezialfall der Filter-Verfahren in der Regel von geringerer Komplexität als die Klassifikations-Algorithmen, die von den Wrapper-Verfahren zur Berechnung des Gütemaßes verwandt werden. Dadurch sind schneller Ergebnisse zu erzielen, was auch im Rahmen des Pollenmonitor-Projektes von Nutzen ist, da hier unter Umständen Experimente in hoher Frequenz durchgeführt werden müssen, um entsprechende Adaptionen des Klassifikationssystems durchzuführen.

Das Relief-Verfahren wurde von Kira et al. bereits 1992 entwickelt [KR92a] [KR92b] und wurde seitdem einigen Modifikationen und Erweiterungen unterzogen [Kon94] [SK97] [RvK03] [SL06] [GBA$^+$03]. Laut Dietterich ist es eines der erfolgreichsten Ranking-Verfahren [Die97]. Es findet nach wie vor in vielen Bereichen des maschinellen Lernens Anwendung, so auch z. B. in der Bioinformatik zum Zweck der Gen-Selektion [YAH$^+$08] [ZDL08]. Das Verfahren weist viele spezielle Eigenschaften auf. So zählt es zu den multivariaten Methoden, die Merkmale werden also im gegenseitigen Kontext bewertet und nicht einzeln für sich [GGNZ06]. Auf diese Weise berücksichtigen die Relief-Methoden Abhängigkeiten zwischen den Merkmalen – was gerade bei „real-world“ Daten häufig der Fall ist [RvK03]. Univariate Ranking-Methoden, wie z. B. Verfahren, welche auf die oben genannten Korrelations-Maße beruhen, leisten dies nicht. Ebenso Verfahren, die zur

Vereinfachung auf Schätzungen univariater Verteilungsdichten beruhen, dies wird z. B. beim Maß der Transinformation oft angewandt [Die97]. Neben dem multivariaten Ansatz werden die Relief-Algorithmen in [GGNZ06] auch als nicht-linear bezeichnet, dies ist der Funktionsweise des Verfahrens geschuldet, welches über Nächste-Nachbar-Suche Abstände zwischen den Verteilungsdichten der Merkmalsvektoren bestimmt und so auch nicht-lineare Grenzflächen zwischen den Klassen erfasst. Im Kontext der Pollenklassifikation ist diese Eigenschaft von Vorteil, da auf dem Pollenmonitor-Datensatz durch Tests bereits starke Nicht-Linearitäten vermutet werden und auch der verwendete Klassifikator durch die Verschachtelung mehrerer linearer Klassifikatoren nicht-lineare Grenzflächen ziehen kann. Die Gesamtheit dieser Faktoren motivierte die Verwendung des Relief-Verfahrens.

3.2.1 Funktionsweise

Für diese Arbeit wurde nicht der ursprüngliche Algorithmus aus [KR92b] gewählt, sondern die Erweiterung von Kononenko [Kon94], welche die meisten Problemstellungen berücksichtigt. Diese Erweiterung ReliefF kann im Gegensatz zum originären Algorithmus auf Multiklassen-Probleme angewandt werden und ist robuster gegenüber Ausreißern durch die Einbeziehung mehrerer Nächster-Nachbarn zur Ermittlung des Merkmals-Rankings.

Dieser Algorithmus arbeitet wie folgt: Er bestimmt zuerst für einen zufällig gewählten klassifizierten Merkmalsvektor ${}^{\varrho}\boldsymbol{c}$ mit Klassenlabel y_ϱ sogenannte *nearest hits* und *nearest misses* [KR92b]. Wobei die *nearest hits* Merkmalsvektoren der gleichen Klasse, die *nearest misses* jeweils Merkmalsvektoren der anderen Klassen sind. Um diese Vektoren zu erhalten, wird eine Nächste-Nachbar-Suche gemäß einer L_1-Metrik (Manhattan-Metrik) durchgeführt und so die k Nächsten-Nachbarn innerhalb der gleichen Klasse (*nearest hits*) und für alle anderen Klassen ebenfalls jeweils die k Vektoren mit kleinstem Abstand zu ${}^{\varrho}\boldsymbol{c}$ (*nearest misses*) ermittelt. Während die Suche noch im gesamten Merkmalsraum (multivariate) stattfindet, wird zur Gewichtung der einzelnen Merkmale eine Projektion auf die jeweilige Merkmalsebene durchgeführt. Hier wird für jedes Merkmal des Vektors ${}^{\varrho}\boldsymbol{c}$ der Abstand zum entsprechenden Merkmal seiner k *nearest hits* und seiner jeweils k *nearest misses* bestimmt. Die Abstände zu den *nearest misses* werden mit den a priori Wahr-

scheinlichkeiten der jeweiligen Klassen (zu ermitteln über die relativen Häufigkeiten) so multipliziert, dass der Beitrag der *nearest misses* zur Gesamtsumme genauso stark ins Gewicht fällt wie der Beitrag der *nearest hits*. Diese Gesamtsumme aus den Werten für die *nearest hits* und *misses* wird so gebildet, dass große Abstände zu den *nearest misses* und kleine Abstände zu den *nearest hits* insgesamt zu großen Werten führen. Diese Merkmalsgewichte w_ν können auf Grund entsprechender Skalierung (vgl. Gleichung 3.4) maximal den Wert 1 annehmen, aber auch negative Werte für sehr schlechte Merkmale im Sinne dieses Kriteriums sind möglich. Dieser Vorgang kann für beliebig viele Merkmalsvektoren iteriert werden. Die Merkmalsgewichte werden hier bei jedem Durchlauf entsprechend aktualisiert. Algorithmus 1 zeigt das gesamte Verfahren zusätzlich in Pseudocode.

Algorithmus 1 : ReliefF

Input : klassifizierte Merkmalsvektoren $\{({}^1\boldsymbol{c}, y_1), \ldots, ({}^N\boldsymbol{c}, y_N)\} = \omega$

Output : Vektor $\boldsymbol{w}$ mit Gewichtungen für jedes Merkmal c_ν

setze $w_\nu = 0\,, \quad \nu = 1, \ldots, n$;
for $\varrho = 1$ **to** M **do**
  wähle beliebigen Merkmalsvektor ${}^\varrho\boldsymbol{c} \in \omega$;
  finde k *nearest hits* Merkmalsvektoren $\boldsymbol{h}^{(i)} \in \Omega_{y_\varrho}$;
  for jede Klasse Ω_κ : ${}^\varrho\boldsymbol{c} \notin \Omega_\kappa$ **do**
    finde k *nearest misses* Merkmalsvektoren ${}_\kappa\boldsymbol{m}^{(i)} \in \Omega_\kappa$;
  end
  for $\nu = 1$ **to** n **do**
    $$w_\nu = w_\nu - \frac{1}{Mk}\sum_{i=1}^{k}\mathrm{diff}({}^\varrho c_\nu, {h_\nu}^{(i)}) + \frac{1}{Mk}\sum_{\kappa\neq y_\varrho}\left[\frac{P(\Omega_\kappa)}{1-P(\Omega_{y_\varrho})}\sum_{i=1}^{k}\mathrm{diff}({}^\varrho c_\nu, {}_\kappa{m_\nu}^{(i)})\right]$$
  end
end

Die verwandte Funktion zur Berechnung des Abstandes zwischen den einzelnen Merkmalen skaliert automatisch deren Wertebereich auf ein Intervall der Größe 1 und sorgt durch diese Normalisierung dafür, dass Merkmale nicht auf Grund ihres Wertebereiches anders

gewichtet werden.[1] Diese Abstands-Funktion wird auch zur Nächsten-Nachbar-Suche verwandt und berechnet sich zu

$$\mathrm{diff}({}^{\varrho}c_\nu, {}^{\tau}c_\nu) = \frac{|({}^{\varrho}c_\nu - {}^{\tau}.c_\nu)|}{\max(c_\nu) - \min(c_\nu)} \tag{3.4}$$

Bild 3.1 zeigt – anhand eines Beispiels im zweidimensionalen Merkmalsraum – für ein Zwei-Klassen-Problem die Funktionsweise des Verfahrens an jeweils einem *nearest miss* ${}_1\boldsymbol{m}^{(1)}$ und einem *nearest hit* $\boldsymbol{h}^{(1)}$.

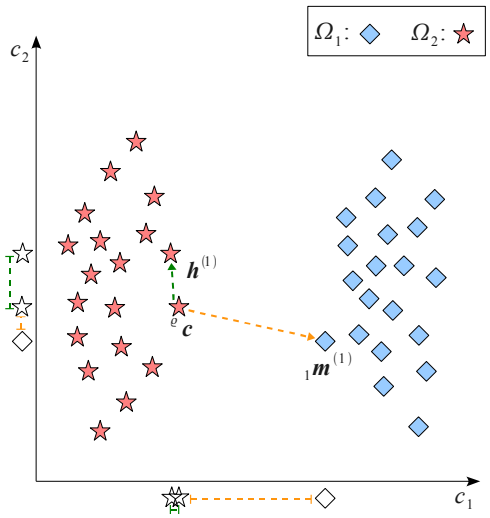


Bild 3.1: Demonstration des Relief-Algorithmus in 2D. Der Intraklassenabstand (grün) ist für Merkmal c_1 kleiner als für c_2, bei dem Interklassenabstand (orange) verhält es sich umgekehrt. Das Merkmal c_1 erhält auf Grund dessen durch den Algorithmus ein höheres Gewicht als c_2.

3.2.2 Analyse

Die Gewichte der Merkmale können als kombiniertes Maß für die in Abschnitt 2.2.1 vorgestellten Interklassen- und Intraklassenabstände betrachtet werden. Hierbei werden Klei-

[1] Diese Normalisierung kann zur Beschleunigung des Algorithmus auch anfänglich für alle Merkmalsvektoren vorgenommen werden, um nicht bei jeder Abstandsberechnung diese Skalierung vorzunehmen. So wurde der Algorithmus auch für diese Arbeit umgesetzt.

ne Intra- und große Interklassenabstände belohnt. An dieser Stelle besteht wiederum eine Analogie zum in Gleichung 2.19 vorgestellten Fishershem Kriterium zur Optimierung der Intraklassen- und Interklassenvarianz. So können letztlich die Merkmalsgewichte als Maß für die Erfüllung der Kompaktheitshypothese von Niemann (vgl. Abschnitt 2.2.2) betrachtet werden.

Eine ausführliche theoretische und empirische Analyse des Verfahrens wurde in [RvK03] vorgenommen. Hier wird die Funktion des Verfahrens informationstheoretisch beschrieben. So kann nachgewiesen werden, dass das erstellte Relevanz-Ranking stark mit dem Gini-Index [RS04] zusammenhängt und so ein Maß für den Informationszugewinn durch das entsprechende Merkmal darstellt, das heißt wieviel jedes Merkmal zur Beschreibung des Problemkreises beträgt. In [RS04] wurden auch Tests durchgeführt um eine Heuristik bezüglich der Anzahl der k Nächsten-Nachbarn und damit der zu erhalten. Hier stellte sich ein Wert von 10 für alle getesteten Szenarien als besonders geeignet heraus. Die Vorgabe wurde auch für diese Arbeit übernommen. Der in [KR92b] vorgeschlagene Schwellwert auf Ebene der Gewichtungen zur Merkmalsauswahl zu

$$0 < \theta \leq \frac{1}{\alpha M} \tag{3.5}$$

wurde für diese Arbeit nicht verwandt, da hier über einen zusätzlichen Wrapper-Ansatz gemäß des Rankings die n besten Merkmale ausgewählt werden, welche die Fehlerrate minimieren (vgl. Abschnitt 5.2). Auf diese Weise wird garantiert, dass die im Kontext des vorgegeben Rankings bestmögliche Fehlerrate erreicht wird. Der Schwellwert ist hierfür nur eine grobe Vorgabe und nicht in der Lage das Extremum der Fehlerrate genau zu bestimmen. M bezeichnet hier die Anzahl der Iterationen (vgl. Algorithmus 1, Zeile 2) und α gibt eine Wahrscheinlichkeit bezüglich der Fehleinschätzung eines irrelevanten Merkmales als relevant an [RS04]. Die Zeitkomplexität des ReliefF-Verfahrens lässt sich zu $\mathrm{O}(N \cdot M \cdot k \cdot n)$ angeben (vgl. Algorithmus 1), also abhängig von der Größe der Stichprobe N, der Anzahl der Iteration M (gewählten Samples ${}^{q}\boldsymbol{c}$), der Anzahl der Nächsten-Nachbarn k und der Anzahl der Merkmale n.

3.3 Modifikationen und Erweiterungen des Relief-Verfahrens

Es wurden einerseits Maßnahmen zur Beschleunigung des ReliefF-Algorithmus unternommen, um die in Abschnitt 3.2.2 aufgeführte Komplexität zu mindern oder trotz dieser kurze Laufzeiten des Algorithmus zu erhalten, andererseits wurde das Verfahren an sich modifiziert, um gegebenenfalls bessere Ergebnisse zu erzielen.

3.3.1 Beschleunigung

Parallelisierung: Das Relief-Verfahren gestattet quasi ohne Modifikationen die parallele Berechnung. So können anstatt nur eines Merkmalsvektors ${}^{\varrho}\boldsymbol{c}$ mehrere parallel prozessiert werden. Dabei erhält jeder Prozess einen eigenen Gewichtungsvektor $\boldsymbol{w}_j$, so dass keine Nebeneffekte durch gleichzeitige Zugriffe entstehen können. Um den finalen Gewichtungsvektor $\boldsymbol{w}$ zu erhalten werden die einzelnen Vektoren gemittelt $\boldsymbol{w} = \frac{1}{m}\sum_{j=0}^{m} \boldsymbol{w}_j$ oder entsprechend der Anzahl der prozessierten Merkmalsvektoren gewichtet. Auf diese Weise wurde auch die Parallelisierung des Relief-Verfahrens für diese Arbeit umgesetzt.

Schnelle Nächste-Nachbar-Suche: In [Rob] wurde bereits mit k-d-Bäumen experimentiert, um die Nächste-Nachbar-Suche zu beschleunigen, deren Berechnungen faktisch die Laufzeit des Relief-Algorithmus bestimmt. Die k-d-Suchbäume, welche den Merkmalsraum partitionieren, um schneller auf Suchanfragen antworten zu können, brachten bei den Tests keine Geschwindigkeitsvorteile im Vergleich zur Verwendung einfacher Vorwarteschlangen (engl. priority queue), welche lediglich die nötigen k Vergleiche mit den bereits gefunden Nächsten-Nachbarn reduzieren. Die k-d-Bäume wurden hier durch die hohen Dimensionen des Merkmalsraumes ineffizient und brachten bereits bei 70 Merkmalen schlechtere Ergebnisse als der Ansatz mit priority queues [Rob]. Auf Grund dessen wurde für diese Arbeit auch ein Verfahren verwandt, das von diesen Gebrauch macht.

In [ATSK08] wurde ein Verfahren zur schnellen k-Nächsten-Nachbar-Suche in hochdimensionalen Datenräumen entwickelt. Es setzt die priority queue anhand einer Heap-Datenstruktur in Form eines binären Baumes um. Die Heap-Invariante ist da-

bei, dass kein Schlüssel im Heap größere Werte besitzen darf als sein Eltern-Knoten. Auf diese Weise befindet sich der Schlüssel mit dem höchsten Wert in der Wurzel des Baumes. Intern wird der Baum als Array repräsentiert und erlaubt dadurch performante Zugriffe auf die Schlüssel. Als Schlüssel fungieren hier die Abstände zu dem Anfragevektor ${}^{\varrho}\boldsymbol{c}$ für den die Nächsten-Nachbarn gesucht werden. Die Werte sind Referenzen auf die korrespondierenden „Nachbar"-Vektoren. Um die Anzahl der benötigten Abstandsberechnungen zu verringern, wird jeweils nach Berechnung des Abstandes für eine Dimension der bis dato akkumulierte Abstand mit dem höchsten Wert in der priority queue (der Wurzel) verglichen. Wird dieser Wert überschritten, muss die Abstandsberechnung nicht vollständig durchgeführt werden und wird daher gestoppt.[2] Auf diese Weise kann die Komplexität des ReliefF-Verfahrens von $\mathrm{O}(N \cdot M \cdot k \cdot n)$ im Prinzip auf $\mathrm{O}(N \cdot M \cdot n)$ reduziert werden. Da nach Füllung des Heap zuerst nur noch Vergleiche mit dem Wurzel-Wert erfolgen. Muss dieser ausgetauscht werden, da ein Vektor mit niedrigerem Abstand gefunden wurde, geschieht dies im worst-case mit einem Aufwand von $\mathrm{O}(\log n)$. Algorithmus 2 zeigt das Verfahren in Pseudocode. Auch diese Form der Beschleunigung wurde in das hier implementierte Relief-Verfahren integriert.

3.3.2 DReliefG

Es wurden zwei wesentliche Veränderungen am bestehendem ReliefF-Verfahren vorgenommen. Statt zufällig ausgewählter Merkmalsvektoren und einer beschränkten Anzahl an Iteration M (vgl. Algorithmus 1), wurde eine deterministische Variante DReliefF erzeugt. Bei dieser wird nicht ein beliebiger Merkmalsvektor ausgesucht, sondern es wird über alle Merkmalsvektoren iteriert. Auf diese Weise erhält man zu jedem Merkmalsvektor ${}^{\varrho}\boldsymbol{c}$ aus der Stichprobe ω seine entsprechenden *nearest hits* und *nearest misses*. Durch dieses deterministische Vorgehen wird ein Vergleich zwischen verschiedenen Testläufen ermöglicht, da keine zufällig gewählten Ausreißer das Ergebnis für einen Testlauf negativ beeinflussen können, im nächsten Test aber ausgelassen werden und so andere Ergebnis-

[2] Da der entsprechende Vektor bereits bis zu dieser Dimension einen größeren Abstand zum Anfragevektor besitzt als der „schlechteste" der bis dato gefundenen „Nachbar"-Vektoren.

Algorithmus 2 : Schnelle NN Suche

Input : N Vektoren $\boldsymbol{x}[\,i\,]$ mit Dimension n, Anzahl nächster Nachbarn k, Suchanfrage Vektor $\boldsymbol{q}$

Output : heap mit k nächsten Nachbarn für $\boldsymbol{q}$

```
// fülle heap mit den ersten k Vektoren.
for i = 1 to k + 1 do
    dist = 0; for j = 1 to n do
        dist = dist + diff(x[i][j], q[j]);
        // dist als Schlüssel
        heap[i] = (dist, x[i]);
    end
    // sortiere heap absteigend.
    // heap[1] enthält den größten Schlüssel.
    sort(heap);
end
for i = k + 1 to N do
    dist = 0;
    for j = 1 to n do
        dist = dist + diff(x[i][j], q[j]);
        if dist > heap[1] then
            break;
        end
    end
    if dist < heap[1] then
        heap[1] = (dist, x[i]);
        // stelle Heap-Invariante wieder her.
        restoreHeap();
    end
end
```

se resultieren. Neben der Reproduzierbarkeit der Ergebnisse bewirkt die Ausnutzung der gesamten Stichprobe die größtmögliche Ausbeute der in ihr enthaltenen Information. Die erhöhte Zeitkomplexität durch Iteration über die gesamte Stichprobe ist auf Grund der vorgenommen Laufzeitoptimierungen (vgl. Abschnitt 3.3.1) vertretbar. Die zweite Modifikation bezieht sich auf die Gewichtung der *nearest hits* und *misses*. Während bei ReliefF die k *nearest misses* für jede Klasse gesucht werden, wurde ein Änderung vorgenommen, die bewirkt, dass insgesamt nur k *nearest misses* berechnet werden. Auf diese Weise wirkt sich der anhand der *nearest hits* geschätzte Intraklassenabstand stärker auf die Bewertung der Merkmale aus. Die *nearest misses* können z. B. nur von einer Klasse belegt werden, die besonders nahe an der Klasse des Anfragevektors ${}^{\varrho}\boldsymbol{c}$ liegt . Dadurch kann der Wert nicht durch Vertreter anderer Klassen, die größere Interklassenabstände aufweisen, positiv verändert werden. Hierdurch entscheidet die Größe des Intraklassenabstandes über die Güte der Merkmale. Die durch diese Modifikationen entstandene Variante DReliefG wird in Algorithmus 3 aufgezeigt. Die Änderungen beziehen sich auf Zeile 2 und Zeile 9 im Pseudocode. Durch die Reduktion auf nur insgesamt k *nearest misses* entstehen auch Laufzeitvorteile für den Algorithmus, wie in Abschnitt 5.2 gezeigt wird.

Algorithmus 3 : DReliefG

Input : klassifizierte Merkmalsvektoren $\{({}^{1}\boldsymbol{c}, y_1), \ldots, ({}^{N}\boldsymbol{c}, y_N)\} = \omega$

Output : Vektor $\boldsymbol{w}$ mit Gewichtungen für jedes Merkmal c_ν

1 setze $w_\nu = 0\,, \quad \nu = 1, \ldots, n$;
2 **for** $\varrho = 1$ **to** N **do**
3 wähle beliebigen Merkmalsvektor ${}^{\varrho}\boldsymbol{c} \in \omega$;
4 finde k *nearest hits* Merkmalsvektoren $\boldsymbol{h}^{(i)} \in \Omega_{y_\varrho}$;
5 **for** alle Klassen $\Omega_\kappa : {}^{\varrho}\boldsymbol{c} \notin \Omega_\kappa$ **do**
6 finde k *nearest misses* Merkmalsvektoren $\boldsymbol{m}^{(i)} \in \Omega \setminus \Omega_{y_\varrho}$;
7 **end**
8 **for** $\nu = 1$ **to** n **do**
9 $w_\nu = w_\nu - \frac{1}{Mk} \sum_{i=1}^{k} \mathrm{diff}({}^{\varrho}c_\nu, {h_\nu}^{(i)}) + \frac{1}{M} \sum_{i=1}^{k} \frac{P(\Omega_\kappa : \boldsymbol{m}^{(i)} \in \Omega_\kappa)}{1 - P(\Omega_{y_\varrho})} \mathrm{diff}({}^{\varrho}c_\nu, {m_\nu}^{(i)})$
10 **end**
11 **end**

3.3.3 Redundanzerkennung mit Relief

Das Relief-Verfahren erkennt keine Redundanzen [KR92a]. So muss hierfür ein zusätzliches Verfahren gewählt werden, um sich möglichst stark an die in Gleichung 3.1 definierte optimale Merkmalsteilmenge anzunähern. Ein gebräuchliches Maß zur Bestimmung der Redundanz zwischen zwei Merkmalen ist die Korrelation [YL04]. Zur Bestimmung dieser Korrelation kann der Korrelationskoeffizient zu

$$\rho_{c_\mu c_\nu} = \frac{\sum_{\varrho=1}^{N} ({}^{\varrho}c_\mu - \bar{c_\mu})({}^{\varrho}c_\nu - \bar{c_\nu})}{\sqrt{\sum_{\varrho=1}^{N} ({}^{\varrho}c_\mu - \bar{c_\mu})^2 \sum_{\varrho=1}^{N} ({}^{\varrho}c_\nu - \bar{c_\nu})^2}} \tag{3.6}$$

verwandt werden. Bei zwei vollständig korrelierten ($\rho = 1$ oder $\rho = -1$) Merkmalen kann von einer Redundanz ausgegangen werden [YL04]. Sind die Merkmale nicht komplett korreliert, muss dies nicht mehr der Fall sein. In Bild 3.2 wird solch ein Szenario aufgezeigt. Bild 3.3.3 zeigt die Auswirkungen von Korrelation anhand zweier bivariater Normalverteilungen. Eine weiteres Verfahren zur Redundanzfindung sind Bayes'sche-

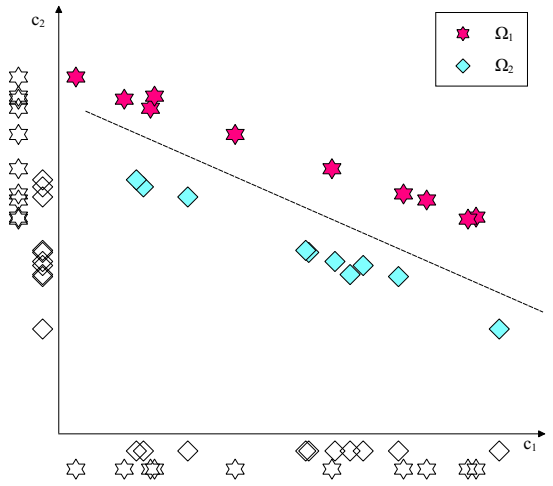
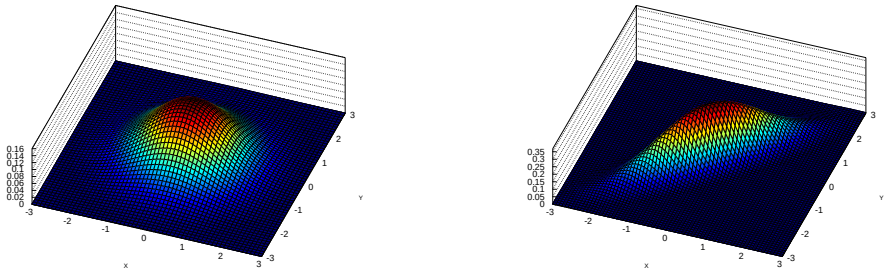


Bild 3.2: Zwei korrelierte Merkmale mit Korrelationskoeffizient $\rho_{c_1 c_2} = -0.75239$. Es werden trotz starker linearer Abhängigkeit beide Merkmale benötigt um die Klassen zu trennen.



(a) $\rho = 0$ (b) $\rho = 0.9$

Bild 3.3: Korrelierte und unkorrelierte bivariate Gaußverteilungen.

Netze [Nie07]. In [YL04] wird ein Verfahren vorgestellt, welches durch eine Redundanz-Definition über Markov-Blankets Redundanzen auffindet. Ein anderer Ansatz ist, die durch ein Filter-Verfahren reduzierte Merkmalsmenge zusätzlich mit einem Wrapper-Ansatz zu bewerten und auf diese Weise etwaige Redundanzen zu eliminieren [DL97].

In dieser Arbeit wurde eine Methode zur Redundanzfindung durch das Relief-Verfahren entwickelt. Diese basiert darauf, wie der Relief-Algorithmus vollständig redundante Merkmale behandelt. In [RvK03] wird anhand eines Merkmalssatzes, welcher Merkmale doppelt enthält, gezeigt, dass solche duplizierten Einträge in Merkmalsvektoren dazu führen, dass diese sich das ursprüngliche Gewicht für das Merkmal teilen. Aus informationstheoretischer Sichtweise teilen sich diese duplizierten Merkmale den Informationszugewinn. Darauf aufbauend wurde folgender Algorithmus entworfen:

1. Entferne jeweils ein Merkmal c_ν aus dem Merkmalssatz und berechne so via Relief n Gewichtungsvektoren $\boldsymbol{w}_\nu$.

2. Suche für jeden dieser Gewichtungsvektoren $\boldsymbol{w}_\nu$ den Gewichtungsvektor mit kleinstem Abstand $\boldsymbol{w}_\mu$.

3. Sortiere die resultierenden Paare nach Größe des Abstands und gebe die korrespondierenden Merkmalspaare (c_ν, c_μ) in Form eines Redundanz-Rankings aus.

Die Gewichtungsvektoren mit kleinstem Abstand stellen hier die größte Redundanz dar, da der ursprünglichen Informationszugewinn durch das ausgelassene Merkmal für diese

auf gleiche Weise zwischen den verbleibenden Merkmalen aufgeteilt wurde. Das heißt die Merkmale leisten einen ähnlichen Informationsbeitrag und weisen daher Redundanzen auf.

Kapitel 4

Entwicklung der Evaluationsumgebung

Um die implementierten Verfahren zur Merkmalsbewertung effektiv und unter Verwendung bereits bestehender Software-Module des Pollenmonitor-Projektes für die Merkmalsanalyse und –auswahl nutzen zu können, wurde eine graphische Umgebung geschaffen, welche die einzelnen Komponenten verbindet und die visuelle Benutzerschnittstelle zu den verwandten Algorithmen und Daten bildet. Aus Sicht des in Bild 2.7 aufgezeigten Klassifikationssystems stellt die Evaluationsumgebung eine Realisierung der Analysephase dar, die dem Nutzer durch Experimente Informationen aus den Modulen „Merkmals-/Musteranalyse“ und „Klassifikation“ zugänglich macht sowie die Möglichkeit bietet diese zu verknüpfen und darauf basierend Entscheidungen auf Merkmalsebene zu treffen.

In diesem Kapitel werden zuerst die Anforderungen an diese graphische Evaluationsumgebung zur Merkmalsbewertung und -auswahl erhoben, um dann im Folgenden aufzuzeigen, wie sie konkret umgesetzt wurden.

4.1 Anforderungen

Im Rahmen des Projektes ergeben sich für die Evaluations-Umgebung, neben allgemeinen Anforderungen an eine zweckdienliche Applikation zur Merkmalsauswahl, projektspezifische Forderungen. Wie in Kapitel 3 aufgeführt, ist man bei der Pollen-Erkennung

unter „real-world“ Bedingungen ständigen Veränderungen des Umfeldes ausgesetzt. Das zu analysierende Datenmaterial wird durch Faktoren, wie Standort des Automaten, Jahreszeit und Wetter bedingt und beeinflusst. Dies führt dazu, dass das Klassifikationssystem an diese neue Begebenheiten angepasst werden muss, z. B. kann durch die Hinzunahme einer Pollen-Klasse die Notwendigkeit neuer Merkmale entstehen, um die Objekte dieser Klasse von denen der vorhanden Klassen abgrenzen zu können. Diese Modifikationen und Erweiterungen des Klassifikationssystems betreffen auch Verfahren der Analysephase als Teil des Gesamtsystems. Daraus ergibt sich, dass auch diese Anwendung so zu gestaltet ist, dass sie möglichst effizient modifiziert und um neue Funktionalitäten erweitert werden kann. Eine zusätzliche projektbedingte Forderung ist die schon oben erwähnte Einbindung der bereits existierenden Softwarekomponenten (Klassifikator, FIT Bildverarbeitungs-Bibliothek) und bestehenden Datenstrukturen. Die Nutzung des Klassifikators aus dem Projekt ermöglicht so, die durch die Merkmalsauswahl erzielten Ergebnisse direkt mit den bisherigen Ergebnissen des Klassifikationssystems zu vergleichen. Desweiteren kann eine vollständige Umgebung zur Merkmalsanalyse, welche alle relevanten Komponenten beinhaltet und somit zentral an einer Stelle zusammenführt, einen reibungsloseren Workflow bieten als mehrere verteilte Applikationen. Diese Gründe bildeten auch das Ausschlusskriterium für die Verwendung einer der bereits bestehenden Applikationen (vgl. Abschnitt 4.1) und motivierten so die Entwicklung einer neuen Evaluationsumgebung. Weitere allgemeine Kriterien ergeben sich aus der Forderung nach einer effektiven Nutzung der zur Verfügung stehenden Algorithmen. Das heißt, diese so zu integrieren, dass dem Nutzer alle wichtigen Information und Parameter zur Verfügung stehen und er so in der Lage ist auf Basis dieser geeignete Merkmalsauswahlen zu treffen.

Aus diesen Vorgaben lassen sich folgende Anforderungen ableiten:

- *Geeignete graphische Benutzerschnittstelle* (GUI)
- *Transparenz und (visuelles) Feedback der eingebunden Algorithmen*
- *Zugriff auf alle relevanten Informationen*
- *Erweiterbarkeit und Modifizierbarkeit*
- *Integration in die bestehende Software-Infrastruktur*

4.2 State of the Art - Andere Evaluationsumgebungen

An dieser Stelle sollen zwei weit verbreitete Evaluationsumgebungen kurz vorgestellt werden. Eine ausführliche Beschreibung findet hier nicht statt, weil dies über den Rahmen dieser Arbeit hinausginge und – wie in Abschnitt 4.1 schon angeführt – die Verwendung einer bereits bestehenden Applikation ausgeschlossen wurde.

WEKA[1] und RAPIDMINER[2] sind bekannte Open Source Anwendungen auf dem Gebiet des Data Mining und der Mustererkennung, die auch entsprechende Funktionalitäten zur Merkmalsanalyse und -auswahl bieten. Beides sind JAVA-Applikationen mit graphischer Oberfläche und universitären Ursprungs. WEKA ist ein Open Source Projekt der Universität Waikato (Neuseeland). Die Entwicklung begann bereits im Jahr 1993. Es bietet unter anderem eine Vielzahl an Verfahren aus den Bereichen Vorverarbeitung, Clustering, Regression, Klassifikation und Merkmalsselektion sowie verschiedene Visualisierungs-Routinen. Eines der ausgeschriebenen Ziele ist es diese Methoden durch WEKA interdisziplinär verfügbar zu machen [HDW94]. Daraus folgt eine graphische Oberfläche, die sich aus Modulen wie „Explorer“, „Experimenter“ oder „Knowledge Flow“ zusammensetzt. Es werden hier zwei verschiedene Paradigmen zur Modellierung der Experimente verfolgt. Zum einen eine stringente Variante über Dialogfenster zu den einzelnen Teiletappen des Experiments, zum anderen eine Darstellung des Experiments als gerichteter Graph, dessen Kanten den Datenfluss und dessen Knoten die Teilschritte des Experiments repräsentieren.

RAPIDMINER entstand an der Universität Dortmund und bietet einen noch weit größeren Umfang als WEKA. Dessen Algorithmen sind in RAPIDMINER als Bibliothek komplett eingebunden. Über 400 Operatoren und eine große Bandbreite an Visualisierungen sind hier verfügbar [MWK+06]. Operatoren bezeichnen hier einzelne Verarbeitungsschritte und werden über eine sogenannten „Operator-Tree“ miteinander verbunden. Bei dem hier verfolgten Ansatz der visuellen Programmierung entsteht eine Baumstruktur. Die Wurzel steht hierbei für das gesamte Experiment, die Knoten für die Operatoren. Ein Unterschied zu WEKA ist hierbei die explizite visuelle Umsetzung einer Prozesskette mit Haltepunk-

[1] http://www.cs.waikato.ac.nz/ml/weka/
[2] http://rapid-i.com/

ten und Bedienelementen zum Starten und Beenden des Gesamtprozesses oder einzelner Teilprozesse.

Screenshots beider Applikation werden in Abschnitt 4.3.4 gezeigt.

4.3 Umsetzung

Die in Abschnitt 4.1 aufgestellten Anforderungen wurden auf folgende Weise realisiert:

Erweiterbarkeit und Modifizierbarkeit: Die Softwarearchitektur wurde gemäß eines modularen Prinzip entworfen, bei dem jeder Verarbeitungsschritt der Merkmalsevaluation durch ein Modul repräsentiert wird (vgl. Abschnitt 4.3.1). Die einzelnen Oberkategorien der Module werden durch entsprechende Basis-Klassen repräsentiert, so dass neue Module auf diesen definierten Schnittstellen aufbauen können. Durch Vererbung wird für jedes Modul ein graphisches Grundgerüst bereitgestellt, welches durch Verfahren der Introspektion (vgl. Abschnitt 4.3.3) automatisch um neue modulspezifische graphische Komponenten erweitert wird (siehe Anhang, Abschnitt B.1). Die verschiedenen Zuständigkeiten wurden gekapselt und so eine Trennung zwischen Programm-Logik, Definition der GUI-Komponenten, Daten-Containern und den Algorithmen des Klassifikationssystems geschaffen. Durch diese Separation können die einzelnen Aspekte weitgehend unabhängig voneinander erweitert und modifiziert werden. Bild 4.3 zeigt diese Struktur in Form eines Klassendiagramms mit den wichtigsten erstellten Klassen.

Geeignete graphische Benutzerschnittstelle: Die Module werden auf graphischer Ebene ebenfalls durch eigenständige Komponenten realisiert. Jedes der Module besitzt seine eigene visuelle Repräsentation. Der Datenfluss zwischen den Modulen wird auf graphischer Ebene durch entsprechende Verbindungen dargestellt. Diese können zwischen den Modulen beliebig gezogen werden und modellieren auf diese Weise intern den Datenfluss zwischen den Algorithmen. Dazu besitzt jede graphische Komponente entsprechende Ein- und Ausgänge, welche die Datenschnittstellen abbilden. Es entsteht auf diese Weise ein gerichteter Graph mit den Modulen als

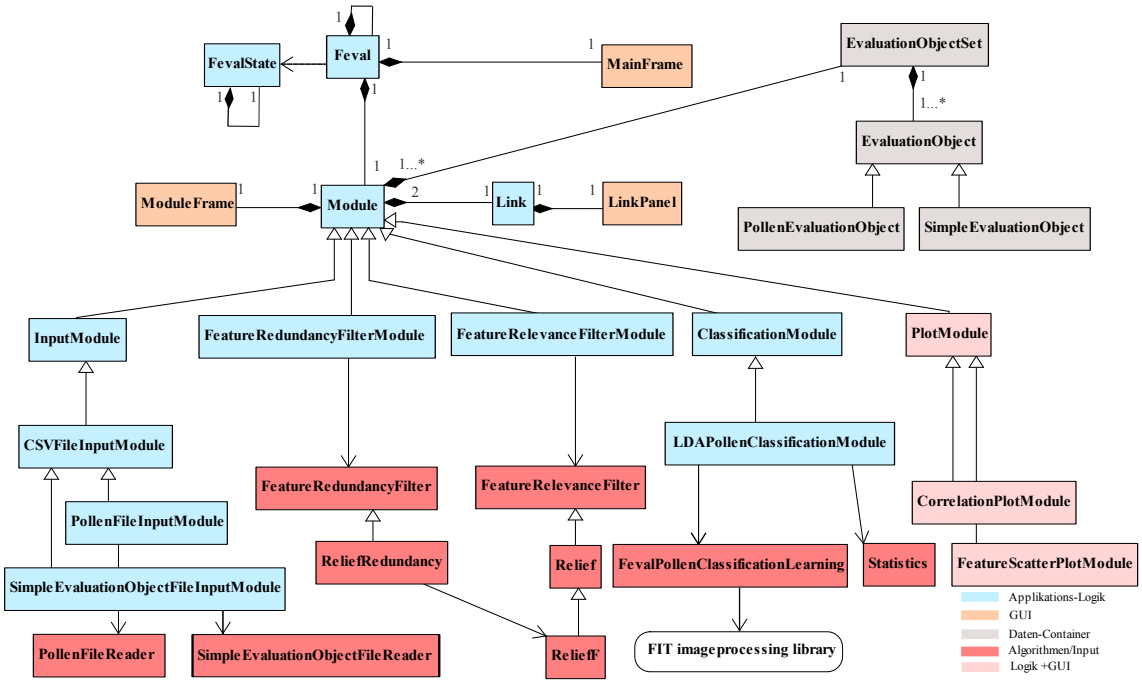


Bild 4.1: Klassendiagramm der Evaluationsumgebung mit den wichtigsten Klassen. Eine Instanz der Klasse `Feval` repräsentiert die Applikation.

Knoten und den Datenverbindungen zwischen den Modulen als Kanten. Zur Konfiguration der Module besitzen diese graphische Eingabemasken und Schaltflächen über die ihre Parameter eingestellt werden können (Abschnitt 4.3.1).

Das Konzept der GUI entspricht dem Paradigma der visuellen Programmierung. Diese soll hierdurch möglichst intuitiv und leicht zu bedienen sein. Das modulare „Baukastenprinzip“ mit entsprechenden „Verdrahtungen“ durch Soft- und Hardware zur digitalen/analogen Klangsynthese inspiriert.

Transparenz und (visuelles) Feedback der eingebunden Algorithmen: Die Algorithmen zur Merkmalsbewertung und der Klassifikator wurden für alle wichtigen Teilergebnisse mit graphischen Outputs auf Modulebene versehen. Zusätzlich geben spezielle Visualisierungs-Module (siehe Abschnitt 4.3.1) graphisches Feedback zur Verteilung der Merkmale im Merkmalsraum und ermöglichen so zusätzliche Überprüfungen der Ergebnisse der Verfahren zur Merkmalsanalyse.

Integration in die bestehende Software-Infrastruktur: Der hierarchische Klassifikator aus dem Pollenmonitor-Projekt wurde über eine Wrapper-Klasse als Modul eingebunden. Für die zu verarbeitenden Objekte wurde ein neues, einheitliches Interface geschaffen und die Algorithmen darauf abgestimmt. Beliebige Datenobjekte können so über Wrapper-Klassen, die dieses Interface implementieren, verarbeitet werden.

Zugriff auf alle relevanten Informationen: Dadurch, dass die Module auf einer einheitlichen Datenstruktur arbeiten, ist für jedes Modul der volle Zugang zu allen in den Objekten gespeicherten Informationen möglich (dieser muss nur über das Interface gewährt werden), z. B. neben Klasse und Merkmalsdaten auch Verweise auf die ursprünglichen Bilddaten aus denen das Objekt extrahiert wurde.

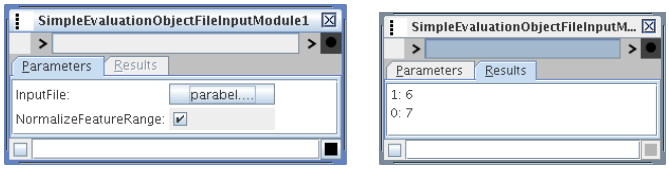
4.3.1 Module

Die graphische Benutzerschnittstelle der Module ist zweigeteilt. So gibt es ein Dialogfeld, das zur Einstellung der Modulparameter dient und ein oder mehre Dialogfelder, welche zur Anzeige der Ergebnisse dienen. Daneben besitzt jedes Modul eine Schaltfläche mit der dessen Verarbeitungsprozess gestartet, beziehungsweise gestoppt werden kann sowie

graphische Komponenten, die als Datenein-/ausgänge fungieren. Ein Modul setzt sich aus seinen graphischen Komponenten, der Anwendungs-Logik, den Parametern sowie dem für die Berechnungen zuständigen Algorithmus zusammen (siehe Bild 4.3).

Hier folgt nun eine Auflistung aller Basismodularten und ihrer konkreten Implementationen.

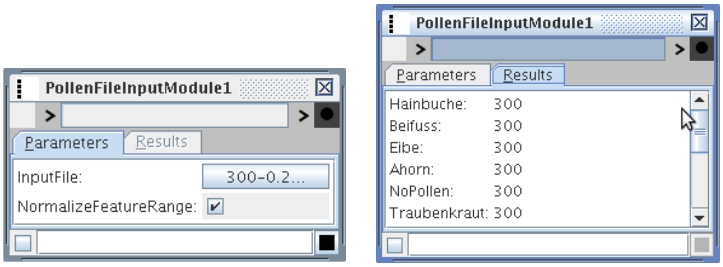
Input-Module: Hier werden die Objekte aus einer Datei geladen und für die Weiterverarbeitung in die entsprechenden Daten-Container überführt. Die Objekte müssen als Mindestanforderung einen Merkmalsvektor mit zugehörigen Merkmalsbezeichnungen sowie ein Klassenlabel enthalten. Es existieren bereits Module zum Einlesen der im Projekt verwandten Dateien mit den darin enthaltenen klassifizierten (Pollen-) Aerosolpartikeln (vgl. Abschnitt 5.1.1) sowie ein Modul zum Laden einfacher Objekte, die nur den oben genannten Mindestanforderungen genügen, um damit z. B. einfache Testdatensätze zu laden. Als Option steht hier die Normalisierung der Merkmale zur Verfügung. Auf diese Weise können sie auf ein Intervall der Länge eins skaliert werden. Der Relief-Algorithmus benötigt z. B. diesen Vorverarbeitungsschritt (Abschnitt 3.2). Bild 4.2 und Bild 4.3 zeigen die GUI-Komponenten dieser Module.



(a) Parameter-Panel (b) Result-Panel

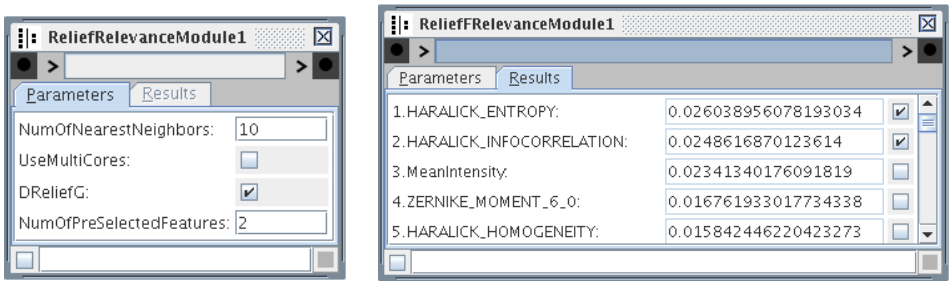
Bild 4.2: `SimpleEvaluationObjectFileInputModule`

Relevanz-Module: Diese Module bilden die Schnittstelle zu den Merkmalsfiltern, die eine Gewichtung der Merkmale nach einem Relevanz-Kriterium vornehmen (vgl. Abschnitt 3.1). Die in Abschnitt 3.3 aufgezeigten Varianten des Relief-Algorithmus zur Merkmalsgewichtung wurden hier in ein entsprechendes Modul integriert. Das Ergebnis-Panel bietet hier Kontrollkästchen um die Merkmale gemäß ihrer Gewichtungen auszuwählen (siehe Bild 4.4).



(a) Parameter-Panel (b) Ergebnis-Panel

Bild 4.3: `PollenFileInputModule`



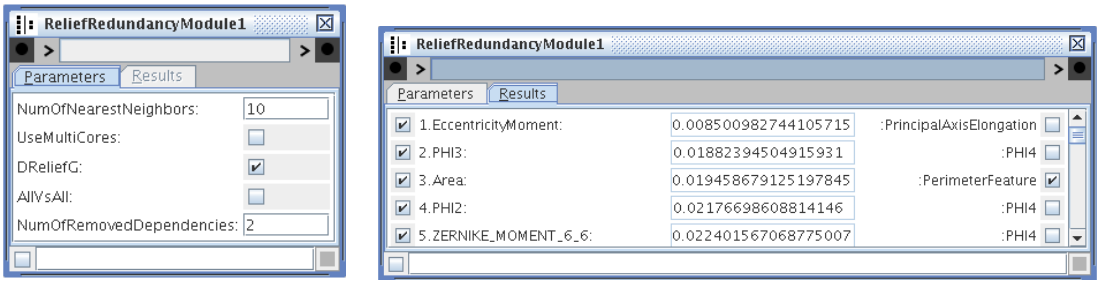
(a) Parameter-Panel (b) Ergebnis-Panel

Bild 4.4: `ReliefRelevanceModule`

Redundanz-Module: Hierunter fallen Verfahren zur Redundanzfindung auf Merkmalsebene. Der in dieser Arbeit entwickelte und implementierte Algorithmus zur Redundanz-Bewertung (vgl. Abschnitt 3.3.3) ist hier als Modul eingebunden worden. Wie bei den Relevanz-Modulen können die Merkmale nach der berechneten Rangfolge ausgewählt werden, wobei hier Merkmalspaare nach Redundanz gewichtet aufgelistet sind (siehe Bild 4.5).

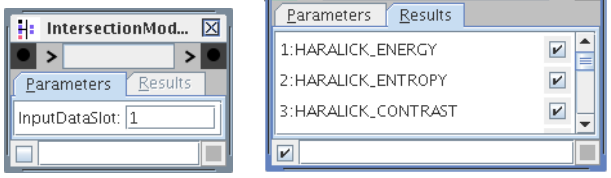
Schnittmengen-Modul: Dieses Modul dient der Kombination unterschiedlicher Merkmalsselektionen. Es wird die Schnittmenge der ausgewählten Merkmale gebildet und dann selektiert. Die Objekte mit der neuen Merkmalsauswahl stehen dann anderen Modulen zur Weiterverarbeitung zur Verfügung. Bild 4.6 zeigt die GUI des Moduls.

Klassifikations-Module: Module, welche sich von diesem Basismodul ableiten, stellen



(a) Parameter-Panel (b) Ergebnis-Panel

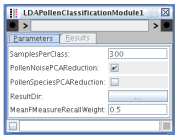
Bild 4.5: `ReliefRedundancyModule`



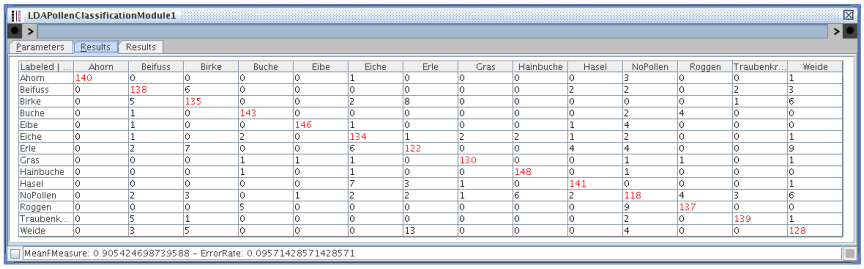
(a) Parameter-Panel (b) Ergebnis-Panel

Bild 4.6: `IntersectionModule`

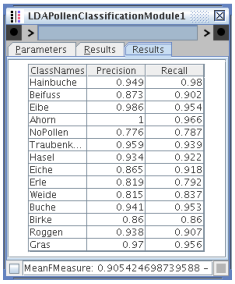
der Evaluationsumgebung Klassifikations-Algorithmen bereit. Die Ergebnisse der Klassifikation werden hier auf mehrere Anzeigen verteilt. Vor der Anzeige werden die Klassifikations-Ergebnisse durch eine spezielle Statistik-Klasse aufbereitet. Hier werden die Fehlerrate, Präzision und Erkennungsrate für jede Klasse (Precision und Recall, Abschnitt 5.1.2) sowie eine Konfusions-Matrix berechnet. Der Klassifikator aus dem Pollenmonitor-Projekt ist hier in ein entsprechendes Modul eingefügt worden. Neben der Größe des Trainingsdatensatzes (Anzahl der Objekte pro Klasse) kann hier noch ein Verfahren der analytischen Merkmalsgewinnung in Form einer PCA (Abschnitt 2.2.1) hinzugenommen werden. Dieses kann hier getrennt für zwei Stufen des hierarchischen Klassifikations-Algorithmus (Abschnitt 5.1.2) gewählt werden. Bild 4.7 zeigt die verschiedenen graphischen Ansichten des Moduls.



(a) Parameter-Panel



(b) Ergebnis-Panel 1



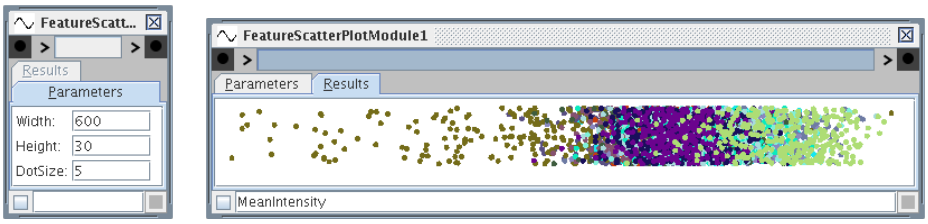
(c) Ergebnis-Panel 2

Bild 4.7: `LDAPollenClassificationModule`

Visualisierungs-Module: Um die Verteilung der Merkmalswerte zu visualisieren wurden zwei Module entwickelt. Ein Modul zum Zeichnen von 1D-Streudiagrammen, welches die Verteilung der Werte eines Merkmals zeigt und ein Modul, welches 2D-Streudiagramme zur Verteilung zweier ausgewählter Merkmale zeichnet. Die Klassenzugehörigkeit wird hierbei farblich gekennzeichnet, um neben Korrelationen eventuelle Clusterbildungen erkennen zu können. In Bild 4.8 und Bild 4.9 werden beide Module mit entsprechenden Plots gezeigt.

4.3.2 Workflow

Die Experimente werden modelliert, indem man die verschiedenen Module miteinander verknüpft und auf diese Weise den Datenfluss zwischen ihnen abbildet. Am Anfang der Prozesskette steht zuerst eine Instanz eines Input-Moduls, um die zu analysierenden Daten bereitzustellen. Darauf können Kombinationen von Relevanz- und Redundanz-Modulen sowie Visualisierungs-Module zur Bewertung der Teilergebnisse folgen. Am En-



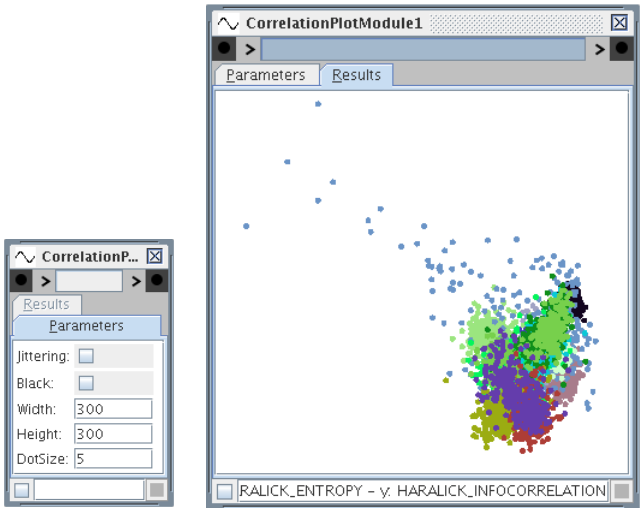
(a) Parameter-Panel

(b) Ergebnis-Panel

Bild 4.8: `FeatureScatterPlotModule`

de der Kette steht zur Validierung der gewählten Merkmals-Teilmenge ein Klassifikations-Modul. Bildet man nach einzelnen Teilschritten Verzweigungen, entsteht ein Baum mit dem Input-Modul als Wurzel und den Klassifikations-Modulen an den Blättern (siehe Bild 4.11). Die Merkmalsauswahl wird hier durch Auswertung der Ergebnisse der Relevanz- und Redundanz-Module unter Feedback des Klassifikations-Moduls getroffen. Die Relevanz- und Redundanz-Module ordnen die Merkmale in Ranglisten. In diesen geordneten Mengen können mit verschiedenen Suchstrategien Teilmengen gesucht werden, die ein gewähltes Kriterium auf Klassifikator-Ebene maximieren. Nach Erreichen dieses Kriteriums (z. B. minimale Fehlerrate oder kleinste Merkmals-Teilmenge über einem Schwellwert) ist das Experiment beendet. Bild 4.10 zeigt diesen Evaluationsprozess. Dieses Vorgehen stellt eine kombinierte Filter-/Wrapper-Methode zur Merkmalsauswahl dar, wie sie in Abschnitt 3.1 aufgezeigt wurde. Statt eines Schwellwertes bezüglich der „lokalen“ Gütemaße der Filter-Routinen wird hier durch den Wrapper-Ansatz der Klassifikator zur Ermittlung des Gütemaßes genutzt.

Die Suche nach der geeigneten Teilmenge innerhalb der Merkmals-Rangliste findet hier manuell durch den Nutzer statt. Dieser kann so z. B. zusätzlich nach jedem Teilschritt die Informationen aus den Visualisierungs-Modulen zur Merkmalsauswahl nutzen (vgl. Bild 4.12). Die Implementierung geeigneter Wrapper-Suchstrategien geht über das Thema dieser Arbeit hinaus, da hier der Fokus auf die Filter-Verfahren zur Bewertung einzelner Merkmale gelegt wurde. Die Möglichkeit einer diesbezüglichen Erweiterung der Evaluationsumgebung ist durch das gewählte Softwaredesign gegeben.



(a) Parameter-Panel

(b) Ergebnis-Panel

Bild 4.9: `CorrelationPlotModule`

Die Umgebung ermöglicht einen parallelen Workflow, das heißt es können mehrere Module gleichzeitig Berechnungen durchführen unter zusätzlicher Beibehaltung der Interaktion durch die GUI. So können auch mehrere Experimente modelliert und parallel berechnet werden. Zur Automatisierung dieser Experimente besteht die Möglichkeit die Module so zu konfigurieren, dass sie eine vorgegebene Anzahl an Merkmalen aus den generierten Ranglisten top-down auswählen. Bild 4.13 zeigt hierzu ein Beispiel.

Schnelle Algorithmen zur Merkmalsanlyse (vgl. Abschnitt 3.3) und deren Feedback über Fortschrittsanzeigen, die intuitive GUI-Metapher und die Parallelisierung sollen insgesamt für einen reibungslosen und schnellen Workflow sorgen.

4.3.3 Implementierung

Implementiert wurde die Software in der objektorientierten, plattformunabhängigen Programmiersprache JAVA unter Verwendung der Version 6 des Sun JAVA Development Kit (JDK). Die Nutzung von JAVA ergab sich dadurch, dass sämtliche schon bestehenden

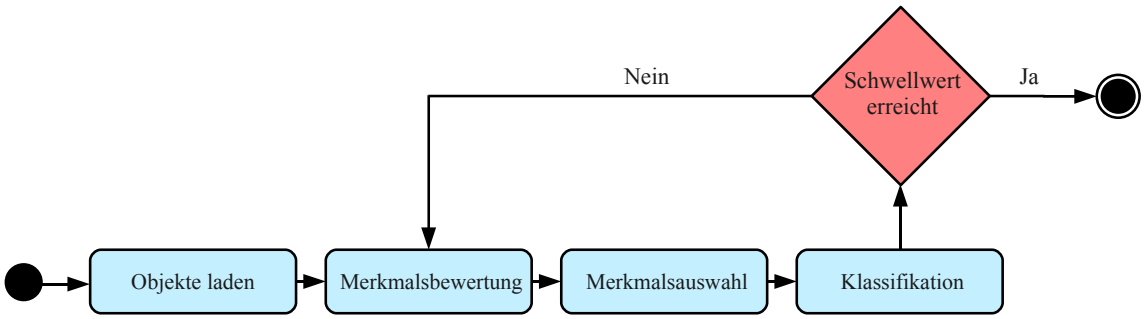


Bild 4.10: Ablauf der Merkmals-Evaluation.

projektspezifischen Software-Module und Bibliotheken bereits in JAVA entwickelt wurden und so deren bestmögliche Integration auf Quellcode-Basis gewährleistet wird. Die GUI wurde über die SWING-Graphikbibliothek realisiert. Diese wurde gewählt, da die Bibliothek in der JAVA-Runtime enthalten ist, sie eine gut dokumentierte API besitzt und Wiederverwendbarkeit des Codes für andere SWING-Applikation aus dem Projekt gegeben ist. Der parallele Workflow wurde durch `SwingWorker`-Threads umgesetzt, welche via Multithreading das Ausführen der Algorithmen zur Merkmalsanalyse bei gleichzeitiger Ausführung der GUI-Routinen ermöglichen. Plausible Werte der Modul-Parameter werden durch SWING-`InputVerifier` für die Parameter-Eingabemasken garantiert. Die Struktur der Modul-Klassen wurde an JAVA-Beans angelehnt. Sie besitzen einen parameterlosen Konstruktor und für die Zugriffe auf die Modul-Parameter spezielle *Getter*- und *Setter*-Methoden. Nach einer Registrierung im Quellcode der Modul-Basisklasse werden über Verfahren der Introspektion, die zur Laufzeit die Struktur der Klassen-Objekte untersuchen, die einzelnen Module anhand der JAVA-Reflexion-API instanziiert und mit graphischen Schnittstellen für die Modul-Parameter versehen. Die von den Basis-Klassen geerbten graphischen Oberflächen werden auf diese Weise automatisch um die entsprechenden Komponenten erweitert. Es werden neben primitiven Daten-Typen auch Modul-Parameter, die Objekte der `File`- oder `Directory`-Klasse darstellen, mit entsprechenden Eingabefeldern ausgestattet. Die Parameter können über vordefinierte JAVA-Annotationen mit Default-Werten sowie eventuellen Ober- und Unterschranken (bei Zahlenwerten) versehen werden. Im Anhang B.1 wird die Implementierung und Einbindung eines

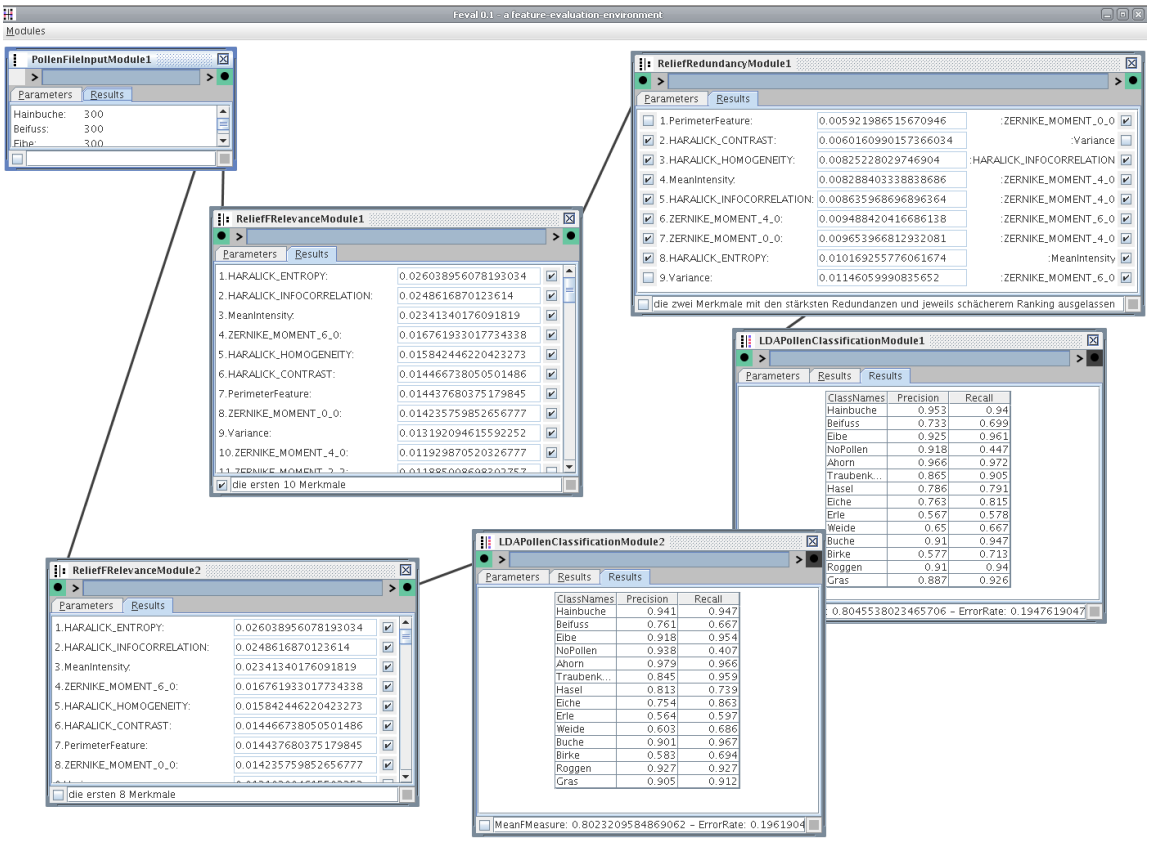


Bild 4.11: Experiment mit zwei verschiedenen Merkmalsauswahlen. Erste über zehn Merkmale via Relevanz-Gewichtung und dann Aufhebung der zwei höchsten Redundanzen durch Entfernung je eines Merkmales pro Redundanz-Paar. Zweite nur durch die ersten acht Merkmale aus dem Relevanz-Ranking.

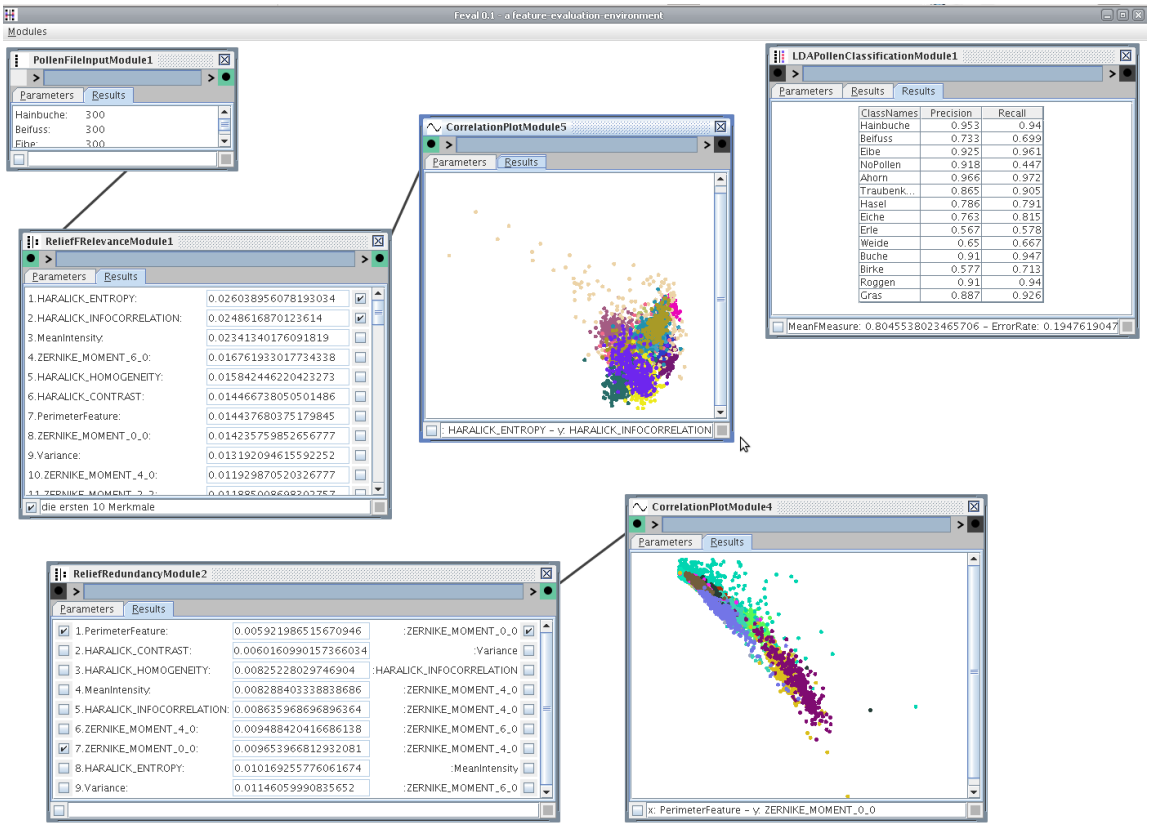


Bild 4.12: Visuelle Überprüfung der beiden Merkmale mit der höchsten Relevanz und des Paares mit der höchsten Redundanz.

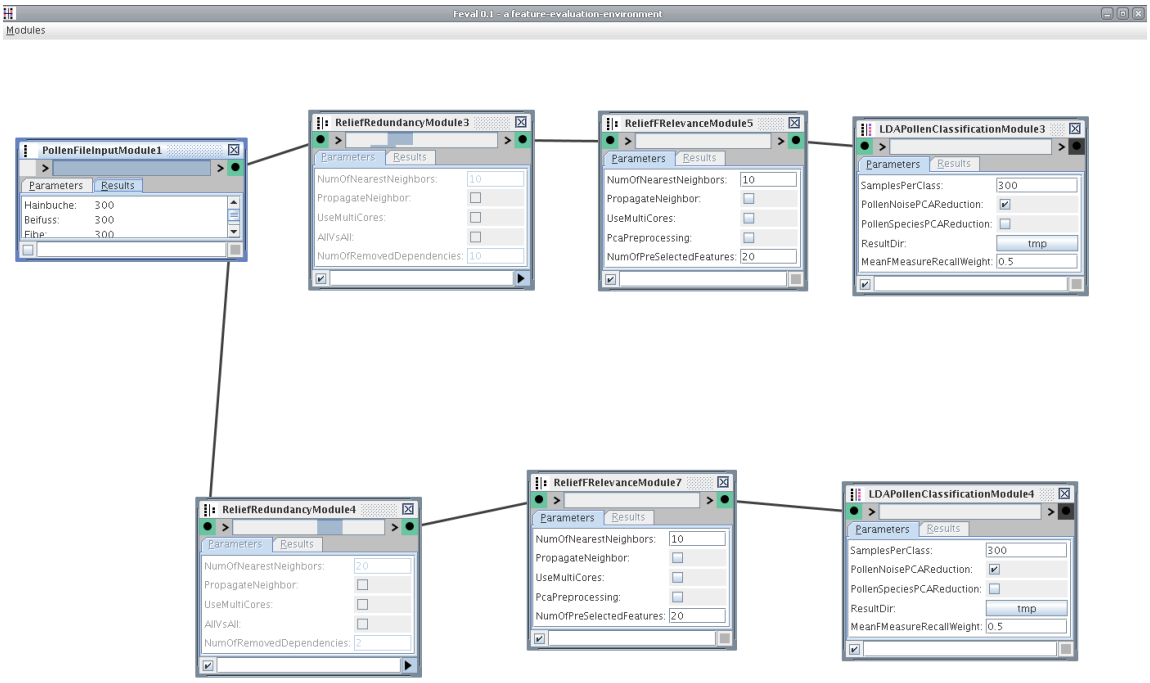


Bild 4.13: Zwei parallele automatisierte Experimente (Module im automatischen Modus, siehe Häkchen). Mit verschiedenen Parametern zur Merkmalsauswahl.

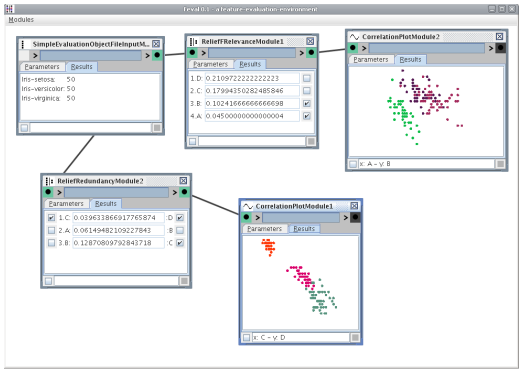
neuen Moduls an einem Beispiel verdeutlicht.[3]

4.3.4 Vergleich

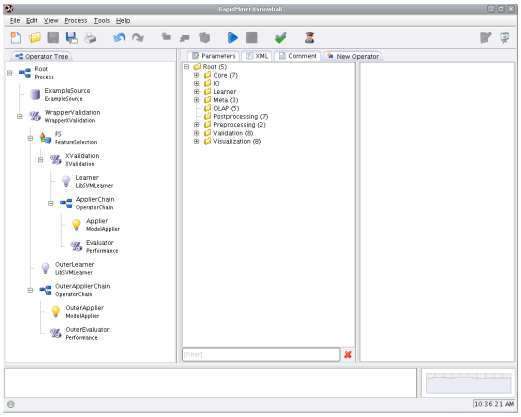
Ein richtiggehender Vergleich der entwickelten Umgebung mit den in Abschnitt 4.2 vorgestellten Applikationen, ist auf Grund deren Umfangs nicht anzustellen. Daher wird nur kurz auf die graphischen Paradigmen zur Modellierung der Experimente eingegangen.

Die hier entwickelte Evaluationsumgebung basiert auf Modulen, die auch visuell entsprechend modular dargestellt werden. Der Datenfluss wird durch virtuelle Verbindungen zwischen diesen Modulen angezeigt. Auf diese Weise entsteht ein gerichteter Graph, dessen Layout und Aufbau via Drag and Drop beliebig geändert werden kann. Dieses Modellierungsprinzip wird auch von WEKA und in abgeänderter Form in RAPIDMINER durch die „Operator-Trees“ (Abschnitt 4.2) unterstützt. Wie RAPIDMINER bietet die Evaluationsumgebung Schaltflächen zum Starten und Stoppen des Experimentes. Eine Ähnlichkeit zu WEKA besteht in der Benutzung eines Multiple Document Interfaces, das heißt die Darstellung mehrerer Unterfenster innerhalb eines großen Basisfensters. Die GUI ist hierbei im Vergleich zu WEKA und RAPIDMINER schlank und einfach gehalten und bietet so einen schnellen Zugang zu den gesuchten Informationen und Parmetern. Bild 4.14 zeigt Screenshots der Anwendungen im Vergleich.

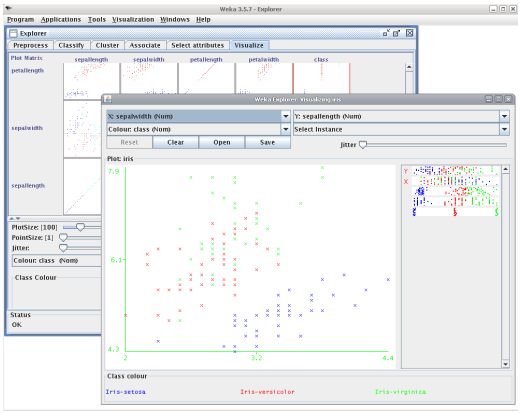
[3]Die in diesem Abschnitt erwähnten JAVA-Konzepte werden z. B. in folgendem Buch erläutert: Guido Krüger, Handbuch der Java-Programmierung, `http:\\www.javabuch.de`.



(a) Neue Evaluationsumgebung (FEVAL) mit Relevanz- und Redundanz-Ranking und Streudiagramm des Paares mit der höchsten Relevanz und Redundanz.



(b) RAPIDMINER mit „Operator-Tree“ zur Merkmalsselektion.



(c) WEKA im „Explorer“-Modul und Streudiagramm zweier Merkmale.

Bild 4.14: Vergleich der graphischen Benutzerschnittstellen. In (a) und (b) wurde der IRIS-Datensatz geladen – einer der Standard-Datensätze aus der Mustererkennung, erstellt von R.A. Fisher, verfügbar unter `http://archive.ics.uci.edu/ml/datasets/Iris`.

Kapitel 5

Experimente und Ergebnisse

In diesem Kapitel folgt eine Dokumentation der durchgeführten Experimente zur Merkmalsselektion. Diese wurden anhand von zwei Testdatensätzen, welche aus dem „Pollenmonitor“-Testdatensatz extrahiert wurden, realisiert. Die Experimente fanden unter der Prämisse statt, die Fehlerrate des Klassifikators bei Verkleinerung des Merkmalssatzes konstant zu halten oder bestenfalls zu verringern. Zu diesem Zweck wurden die implementierten Filter-Verfahren zur Relevanz- und Redundanz-Bewertung mittels der entwickelten Evaluationsumgebung angewandt.

In Abschnitt 5.1 werden der genutzte Datensatz und Klassifikator aus dem „Pollenmonitor“-Projekt vorgestellt.Im Weiteren werden die Kriterien zur Bewertung der Leistung des Klassifikators erläutert. Abschnitt 5.2 beschreibt die Durchführung und die Ergebnisse der Experimente. Hierbei wird zusätzlich der Versuch unternommen die implementierten Verfahren zu validieren.

5.1 Versuchsbedingungen

Neben den Verfahren zur Merkmalsanalyse und der verwandten Evaluationsumgebung bilden folgende zwei Faktoren die Grundlage für die Experimente.

5.1.1 Testdatensätze

Der verwandte Datensatz stammt aus dem Pollenmonitor-Projekt und beinhaltet ursprünglich ca. 20.000 klassifizierte Aerosolpartikel, die durch entsprechende Merkmalsvektoren zu je 75 Merkmalen repräsentiert werden. Aus diesem Datensatz wurden für die folgenden Experimente zweimal zufällig 4.200 Objekte dreizehn unterschiedlicher anemophiler Pollen-Taxa und einer „NoPollen"-Klasse (alle Objekte, die keine Pollen-Körner sind) so ausgewählt, dass in den resultierenden zwei Datensätzen jede Klasse statistisch gleich stark vertreten ist (300 Objekte pro Klasse). Es wurden zwei Datensätze extrahiert, da zu einigen der Klassen weitaus mehr als 300 Objekte vorliegen. Da zwei der für das Projekt wichtigen Taxa (Beifuß und Traubenkraut/Ambrosia) nur durch knapp über 300 klassifizierte Merkmalsvektoren vertreten sind, mussten alle Klassen auf diesen Wert beschränkt werden. Die beiden so gewonnenen Datensätze 1 und 2 sind daher für diese Klassen nahezu deckungsgleich, besitzen aber für die anderen Klassen disjunkte Anteile.

Die enthaltenen Merkmale wurden nach Erfahrungswerten gewählt und entstanden durch Transformation T_H der heuristischen Merkmalsgewinnung (vgl. Abschnitt 2.2.1). In Tabelle 5.1 werden einige der verwandten Merkmale nach Kategorien geordnet aufgeführt. Alle diese Merkmale weisen Rotationsinvarianten auf. Zur Berechnung der Merkmale

Art/Methode	Anzahl	Beispiele
Form	11	Normierte Momente, Rundheit, Kompaktheit
Größe	2	Flächeninhalt, Durchmesser
Statistik	10	Haralick, Varianz
Orthogonale Basis	52	Zernike-Momente, Fourier-Deskriptoren (Konturlinie)

Tabelle 5.1: Merkmale im Pollenmonitor-Testdatensatz.

wurden die ursprünglichen Bildstapel durch ein Verfahren zur Kontrastmaximierung auf Einzelbilder projiziert und aus diesen über Segmentierungsmethoden die einzelnen Objekt-Regionen extrahiert. Diese Vorgehensweise ermöglicht die Verwendung von 2D Textur- und Kontur-Merkmalen. Trotz vieler nicht vollständig kugelsymmetrischer Pollenkörner, z. B. bedingt durch Keimöffnungen und spezielle Formen der Körner, hat sich dieser Ansatz als sehr leistungsstark erwiesen (vgl. Abschnitt 2.1.3). Durch Verwendung dieses Ver-

fahrens werden wieder Methoden zur Merkmalsanalyse motiviert, welche die Merkmale liefern können, die unabhängig von der jeweiligen Projektion sind. Bild 5.1 zeigt ein auf diese Weise entstandenes „synthetisches“ Bild mit segmentierten Objekten.

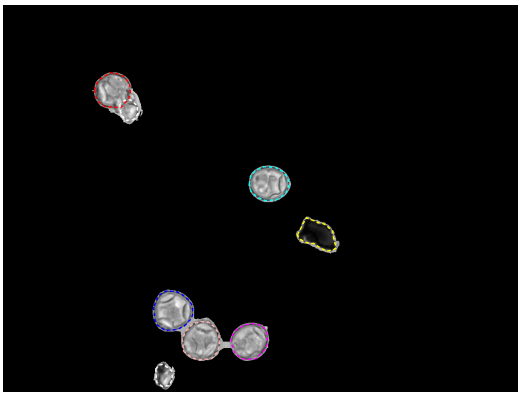


Bild 5.1: Synthetisches Bild aus gescanntem Bildstapel mit segmentierten Hasel-Pollenkörnern und einem „NoPollen“-Objekt (Ursprüngliches Bildmaterial: Helmut Hund GmbH).

Die Klassen-Label der segmentierten Objekte wurden von einem Experten erstellt, wobei hier Fehlzuordnungen nicht ausgeschlossen werden können. Reliabilitätstests dazu wurden in [Ron07] durchgeführt. Die enthaltenen Objekte stammen teils aus dem laufenden Betrieb des Automaten, teils wurden diese von handpräparierten „Schüttelproben“ abgescannt. Auf Grund dieser unterschiedlichen Modalitäten weisen die Klassen mehr oder weniger starke Intraklassenvarianzen auf. Die Pollenkörner, die unter realen Bedingungen durch den Automaten gesammelt wurden, stammen von unterschiedlichen Einzelpflanzen und waren verschiedenen Witterungseinflüssen ausgesetzt. Die Objekte aus den Schüttelproben stammen hingegen teilweise nur von einer Pflanze des entsprechenden Taxon und wurden zu einem festen Zeitpunkt erstellt. Hier besteht die Gefahr einer schlechten Generalisierung des Klassifikators für Pollenkörner der gleichen Klasse, aber von anderen Einzelpflanzen. Dies zeigt die Herausforderung auf Merkmalsebene auf, die darin besteht, Merkmale zu finden, welche sich trotz dieser Heterogenität für einzelne Klassen relativ robust verhalten.

5.1.2 Klassifikator

Für die Tests wird der hierarchische Klassifikator aus dem Pollenmonitor-Projekt verwandt, der im Rahmen dieser Arbeit bereits in die entwickelte Evaluationsumgebung integriert wurde. Er basiert auf Verfahren zur linearen Diskriminanzanalyse (vgl. Abschnitt 2.2.2) und löst das gegebene Multi-Klassen-Problem über mehrere Klassifikationsschritte. Hierzu werden einzelne binäre Klassifikatoren hintereinander geschaltet. Es ergeben sich so drei konsekutive Klassifikations-Phasen:

1. *Pollen-Noise:*
   - Trennung der „NoPollen“-Objekte von den Pollenkörnern.
   - Weitereichen der Pollen-Objekte an nächste Phase.
2. *One-versus-All:*
   - Test jeder einzelnen Klasse gegen alle Klassen.
   - Erstellung eines Ratings für die Zugehörigkeit zu einer Klasse aus den einzelnen Klassifikations-Ergebnissen.
3. *One-versus-One:*
   - Eins-zu-eins Klassifikation zwischen den Klassen mit den besten Ratings.
   - Die Klasse mit dem höchsten Rating hieraus „gewinnt“.

Der Klassifikator liefert immer eine Entscheidung für eine tatsächliche Klasse. Eine Rückweisungsklasse Ω_0 für unsichere Entscheidungen ist in der verwandten Version noch nicht enthalten.

Trotz der vielen benötigten Einzel-Klassifikatoren, lässt sich hier der Klassifikationsprozess durch die Verwendung der wenig zeitkomplexen linearen Diskriminanzanalyse schnell durchführen.

Zur Validierung des Klassifikationssystems werden die verwandten Datensätze 1 und 2 jeweils zu gleichen Teilen in einen Trainings-Datensatz und einen disjunkten Datensatz

zum Testen des Klassifikators aufgeteilt. Die Klassifikations-Ergebnisse werden neben der Schätzung der Fehlerwahrscheinlichkeit durch die Fehlerrate $\hat{p}_f$ (vgl. Abschnitt 2.2.2) anhand von zwei weiteren statistischen Maße und einer Kombination der beiden bewertet. Diese stammen aus dem Bereich des Information Retrival und beziehen sich auf die Klassifikator-Leistung bezüglich einer einzelnen Klasse. Diese Maße Precision (Genauigkeit) und Recall (Erkennungsrate, Trefferquote) sind dabei wie folgt definiert:

$$\text{Precision} = \frac{|\{\text{Objekte aus } \Omega_\kappa\} \cap \{\text{als } \Omega_\kappa \text{ klassifizierte Objekte}\}|}{|\{\text{als } \Omega_\kappa \text{ klassifizierte Objekte}\}|} \tag{5.1}$$

$$\text{Recall} = \frac{|\{\text{Objekte aus } \Omega_\kappa\} \cap \{\text{als } \Omega_\kappa \text{ klassifizierte Objekte}\}|}{|\{\text{Objekte aus } \Omega_\kappa\}|} \,. \tag{5.2}$$

Das kombinierte F-Maß [MKSW99] ergibt sich aus dem gewichteten harmonischen Mittel beider Größen zu:

$$F = \frac{\text{Precison } \cdot \text{Recall}}{(1-\alpha) \cdot \text{Precision } + \alpha \cdot \text{ Recall}} \,. \tag{5.3}$$

α ist hier eine Gewichtung für die normalerweise der Wert 0.5 gewählt wird, so auch im Folgenden. Zur Auswertung der Experimente wurde das F-Maß über alle Klassen gemittelt – als Komplement zur Fehlerrate. Zusätzlich zu diesen Maßen ist noch eine Konfusions-Matrix gegeben, die sämtliche Zuordnungen zwischen den Test-Objekten und ihrer eigentlichen und der durch den Klassifikator zugewiesenen Klasse aufzeigt. Hieraus lassen sich z. B. Werte für falsch positive oder falsch negative Zuordnungen ableiten.

Von der Option, die in den Klassifikator integrierte Hauptachsentransformation – als zusätzlichen Schritt zur Dimensionsreduktion – nach der Merkmalsauswahl durchzuführen, wurde kein Gebrauch gemacht, da diese schon in vorangehenden Tests durchgängig eine Vergrößerung der Fehlerrate nach sich zog.

5.2 Experimente

Neben der Bestimmung einer geeigneten Merkmals-Teilmenge wurden die Experimente unter drei zusätzlichen Gesichtspunkten durchgeführt:

1. Validierung des Relevanz-Verfahrens.

2. Vergleich der beiden Varianten DReliefF und DReliefG.

3. Validierung des Verfahrens zur Redundanzerkennung.

Zur Durchführung der Experimente wurde jeweils nach dem gleichen Schema verfahren. Die über die Filter-Methoden gelieferten Merkmals-Ranglisten dienten hier als Vorgabe zur Merkmalsauswahl. Anstatt über einen Schwellwert auf Ebene des Merkmals-Gütemaßes (vgl. Abschnitt 3.1), wurden in diesem Fall die Teilmengen durch sukzessives Entfernen einzelner Merkmale gemäß ihrer Gewichte gebildet. Es wurde dann die Untermenge gewählt, die das beste Klassifikations-Ergebnis herbeiführte. Dieser Ansatz ist somit eine Mischung aus Filter- und Wrapper-Methoden (vgl. Abschnitt 3.1). Die Suchstrategie zur Findung der Teilmengen besteht dabei in einem einfachen stufenweisen bottom-up Entfernen der Merkmale, bzw. der top-down Auswahl der Merkmale mit den besten Bewertungen. Bei dem Verfahren zur Redundanzerkennung verhält es sich genau umgekehrt, da hier die Merkmalspaare mit einer hohen Redundanz im Ranking oben stehen. Die Merkmale wurden erst in 5er Schritten entfernt, die bei größeren Schwankungen der Fehlerrate verfeinert wurden, um das eventuelle Extremum genau zu lokalisieren.

5.2.1 Validierung des implementierten Relief-Verfahrens

Hier wurde unter Verwendung des Datensatzes 1 und der entwickelten DReliefG-Variante des Relief-Verfahrens – nach oben genannter Vorgehensweise – die Fehlerrate und das über alle Klassen gemittelte F-Maß bestimmt. Die niedrigste Fehlerrate wurde mit den 50 „besten“ Merkmalen zu $7,86\%$ erzielt, bei einer ursprünglichen Fehlerrate von $8,10\%$ unter Verwendung aller 75 Merkmale. Die Fehlerrate bleibt bis zu einer Auswahlgröße von ca. 20 Merkmalen fast konstant auf dem anfänglichen Wert. Nach dem Minimum bei 50 Merkmalen steigt sie nur ganz allmählich an.

Zur weiteren Validierung des Verfahrens wurde eine inverse Selektion vorgenommen, das heißt bottom-up die „schlechtesten“ Merkmale aus der Rangliste beibehalten. Dies führte schon bei 70 Merkmalen zu einer so hohen Fehlerrate wie bei Auswahl der 20 am besten bewerteten Merkmale. In Bild 5.2 ist der Verlauf der Fehlerrate und des F-Maßes für diese beiden Versuche aufgezeigt.

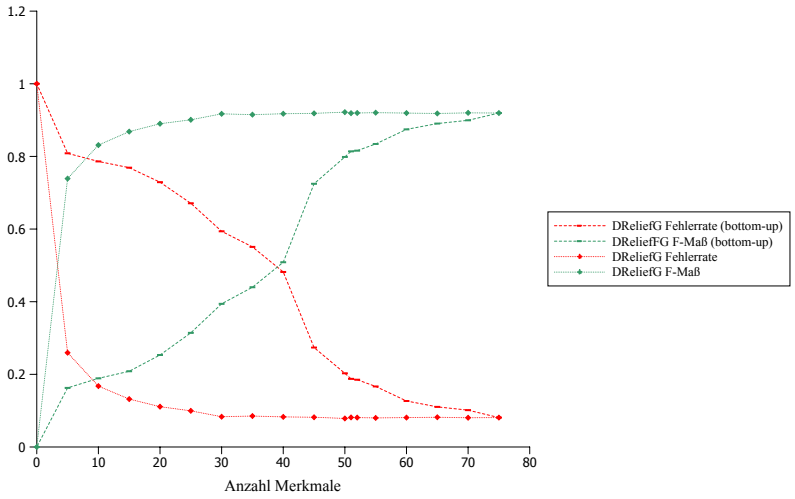


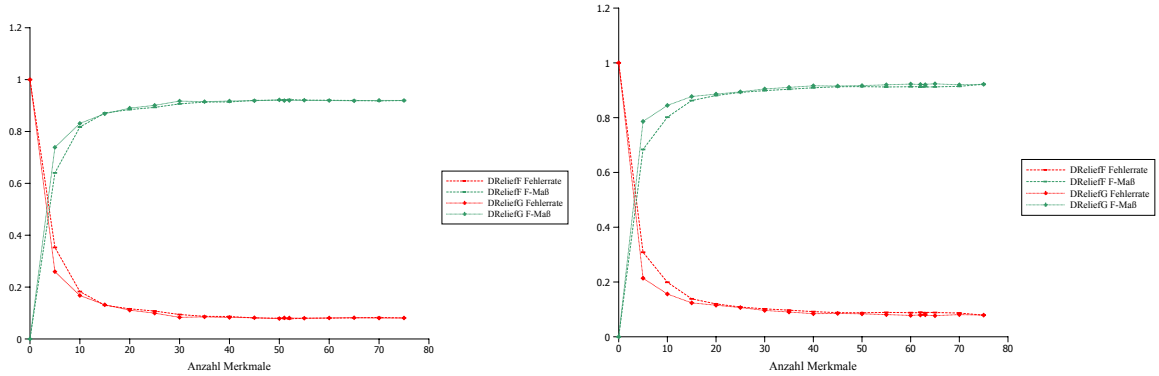
Bild 5.2: Datensatz 1 bei normaler und inverser Selektion (bottom-up).

5.2.2 Vergleich zwischen DReliefG und DReliefF

Zur Bewertung der Modifikation DReliefG des Relief-Verfahrens wurde zusätzlich ein Test mit dem ursprünglichen (bis auf die deterministische Berechnung) DReliefF-Algorithmus ausgeführt.[1] Um die Aussagekraft des Experiments zu erhöhen, wurden beide Verfahren sowohl auf Datensatz 1 als auch auf Datensatz 2 angewandt.

Aus Bild 5.3 gehen die Ergebnisse dieses Experiments hervor. Es ist hier zu erkennen, dass die DReliefG-Modifikation in beiden Fällen (Datensatz 1 und 2) bessere Ergebnisse liefert. Wenn auch der Unterschied zuerst minimal ausfällt, besteht bei einem Merkmalssatz der Größe fünf eine Differenz von ca. 10% zwischen den Fehlerraten der beiden Varianten, sowohl für Datensatz 1 als auch 2. Das Optimum der Fehlerrate ist beim Datensatz 2 über DReliefG nur mit 60 Merkmalen zu erreichen und auch die erreichte Senkung der Fehlerrate von 7.90% auf 7.80% fällt geringer aus. Der Gesamtverlauf der Fehlerrate, beziehungsweise des F-Maßes, verhält sich aber analog zu den auf Datensatz 1 erzielten Ergebnissen.

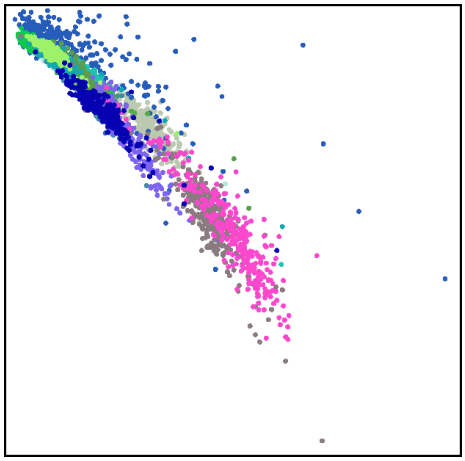
[1] Ein Test der originalen, nicht-deterministischen Variante wurde nicht durchgeführt, da diese auf Grund des randomisierten Samplings keine konstanten Ergebnisse liefert.



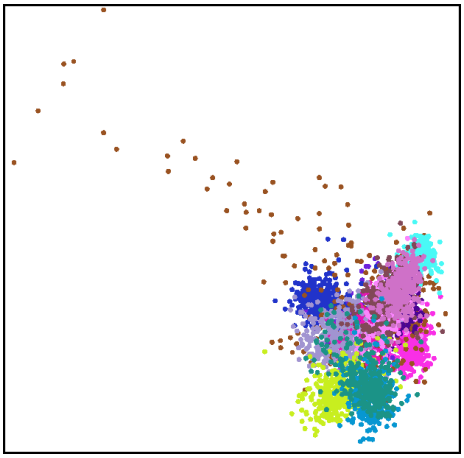
(a) Datensatz 1. (b) Datensatz 2.

Bild 5.3: Vergleich der Verfahren DReliefF und DReliefG.

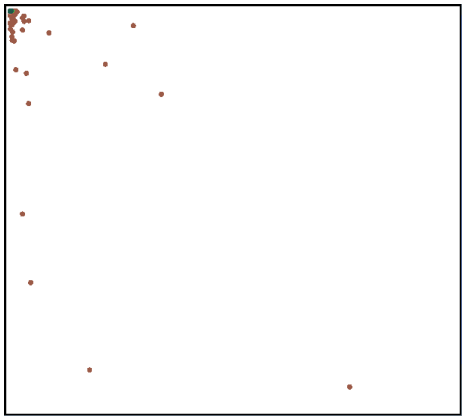
Vergleicht man das Streudiagramm der beiden „besten“ Merkmale nach DReliefF mit dem Streudiagramm der beiden „besten“ Merkmale nach DReliefG, so sieht man, dass bei DReliefG die Merkmale kompaktere Bereiche für die einzelnen Klassen (gleiche Farbe) bilden, im Gegensatz zu der korrelierten Verteilung bei DReliefF. Dies lässt sich durch die höhere Gewichtung des Intraklassenabstandes durch das DReliefG-Verfahren erklären (vgl. Abschnitt 3.3). In beiden getesteten Datensätzen wirkt sich diese Vorgehensweise auch positiv auf die Güte – im Sinne der Fehlerrate – der Rangliste, respektive der dadurch gewählten Merkmalssätze, aus. Bild 5.4 zeigt diese Streudiagramme und noch zusätzliche für die Paare mit der niedrigsten Gewichtung. Auch hier ist zu erkennen, dass DReliefG durch die ungleichmäßige Gewichtung zwischen Intraklassenabstand und Interklassenabstand nicht wie ReliefF das Paar auswählt, welches ein komprimiertes Cluster für alle Klassen bildet und so einen kleinen Interklassenabstand bedingt, sondern eines, welches für alle Klassen stark streut und so einen großen Intraklassenabstand besitzt. Für beide Varianten lässt sich in den Diagrammen erkennen, dass die beiden Merkmale mit der höchste Bewertung – im klaren Gegensatz zu den beiden schlechtesten – kompakte Bereiche für die Klassen im zweidimensionalen Merkmalsraum bilden und somit Niemanns Kompaktheitshypothese (vgl. Abschnitt 2.2.2) entsprechen. Es ist zu beachten, dass die Merkmale mit dem besten Ranking im 2D-Plot nicht zwingend die beste Separation aufweisen müssen, da es sich bei dem Relief-Verfahren um eine multivariate Methode handelt (vgl. Ab-



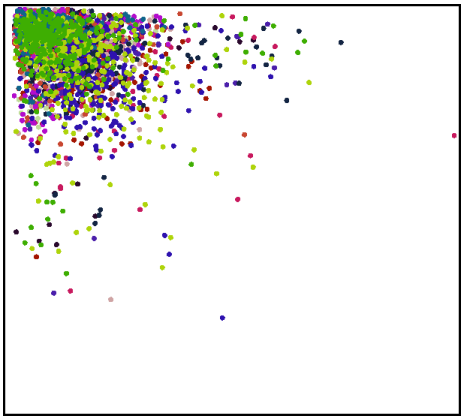
(a) Höchste Gewichtungen (DReliefF).



(b) Höchste Gewichtungen (DReliefG).



(c) Niedrigste Gewichtungen (DReliefF).



(d) Niedrigste Gewichtungen- (DReliefG).

Bild 5.4: Streudiagramme der beiden Merkmale mit höchstem und niedrigstem Gewicht (Datensatz 1).

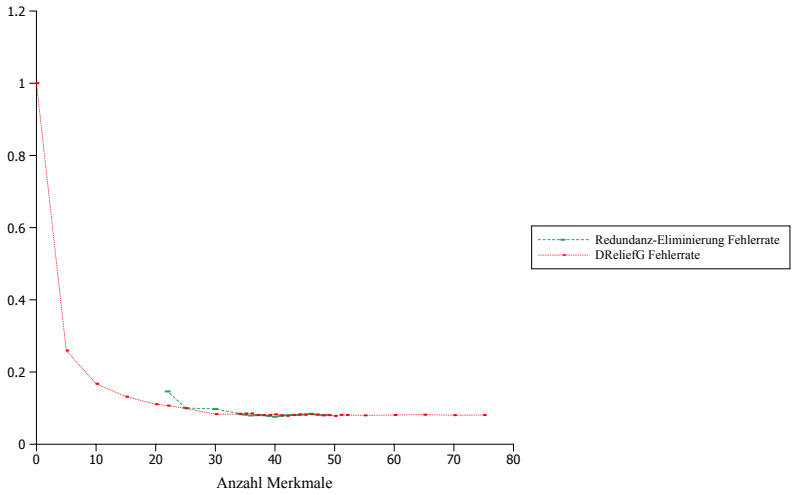


Bild 5.5: Vergleich DReliefG und DReliefG + Redundanz-Eliminierung.

schnitt 3.2), welche die einzelnen Merkmale im Kontext des gesamten Merkmalsraumes bewertet.

5.2.3 Redundanzerkennung

Zur Validierung der Redundanzerkennung wurden für Datensatz 1 – unter Verwendung von DReliefG – die ersten 50 Merkmale bestimmt, um diese dann durch das Verfahren zur Redundanzerkennung zu bewerten und Merkmale entsprechend dieses Redundanz-Rankings zu entfernen. Dieses Ranking bezieht sich nicht wie bei den Verfahren zur Relevanz-Bestimmung auf einzelne Merkmale, sondern auf Merkmalspaare. Um diese Redundanz-Paare aufzulösen, wurde unter Feedback des vorher erstellten Relevanz-Rankings das Merkmal aus dem Paar entfernt, welches in diesem Ranking den niedrigeren Wert erhalten hatte. Nach diesem Prinzip wurden mehrere Merkmale schrittweise entfernt und die entsprechenden Fehlerraten zu den jeweiligen Merkmalssätzen berechnet, um diese Ergebnisse so mit den vorherigen – ohne Redundanz-Eliminierung – vergleichen zu können. Bild 5.5 zeigt diesen Vergleich.

Über diese Herangehensweise, also der Kombination aus Merkmalsauswahl gemäß des

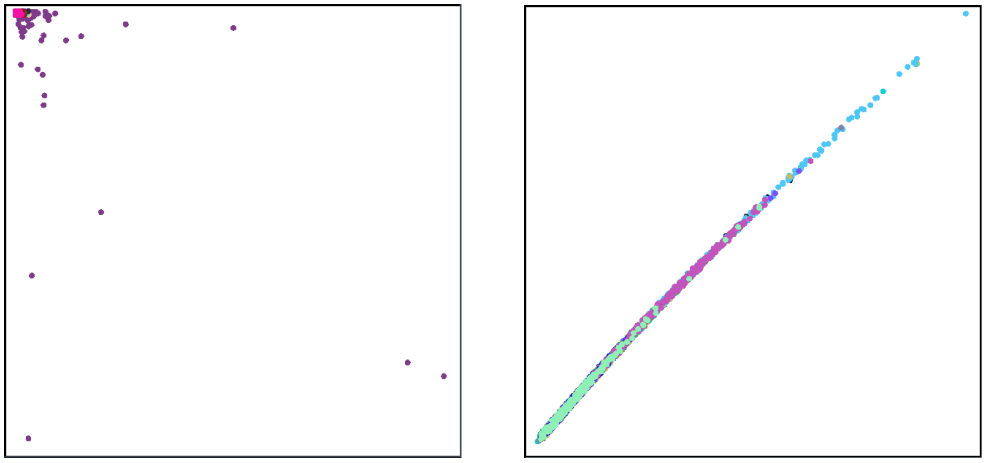
Verfahren	75	50	40
DReliefG	$8,10\%$	$7,86\%$	$8,29\%$
DReliefG + Redundanz	$8,10\%$	$7,86\%$	$7,61\%$

Tabelle 5.2: Fehlerraten für Merkmalsauswahlen unterschiedlicher Größe nach DReliefG und nach DReliefG + Redundanzerkennung.

Relevanz-Rankings und anschließender Redundanzerkennung mit Merkmalsauswahl unter Verwendung des zuvor erstellten Relevanz-Rankings, konnte im Test für Datensatz 1 ein neuer Tiefpunkt der Fehlerrate bei einer Merkmalsauswahl von 40 Merkmalen mit einem Wert von $7,61\%$ erreicht werden. Tabelle 5.2 zeigt diesen Sachverhalt.

Zur visuellen Überprüfung der Ergebnisse werden in Bild 5.6 die ersten beiden Redundanz-Paare im Streudiagramm dargestellt. Die Merkmale aus dem zweiten Redundanz-Paar sind hierbei vollständig korreliert. Es besteht also mit hoher Wahrscheinlichkeit eine Redundanz für den Klassifikationsprozess (vgl. Abschnitt 5.2.3). Das erste Redundanz-Paar weist im 2D-Plot ein konzentriertes Cluster auf. Man kann hier erkennen, dass eine Klasse („NoPollen“) große Ausreißer verursacht auf Grund derer das Schaubild für die anderen Klassen gestaucht ist.[2] Es ist durch das Cluster gegeben, dass die beiden Merkmale hierin gleiche Werte besitzen und damit auch eine Redundanz zwischen ihnen besteht. Ferner sieht man, dass das Paar mit der höchsten Redundanz das gleiche Paar ist, welches durch das DReliefF-Verfahren die geringste Relevanz erhielt (vgl. Bild 5.4). Es ist aus den Ergebnissen bei einigen Merkmalen ersichtlich, dass hohe Relevanz-Rankings mit niedrigen Redundanz-Ranking korrelieren, z. B. Merkmale *Haralick entropy* und *mean intensity* belegen bei Datensatz 1 Platz 1 und 2 im Relevanz-Ranking und den drittletzten und letzten Platz im Redundanz-Ranking. Dies spricht zusätzlich für die Güte der Merkmale im oberen Bereich der Relevanz-Rangliste.

[2]Die großen Schwankungen der Merkmalswerte innerhalb der „NoPollen“-Klasse können sich durch die Normierung der Merkmale auch negativ auf das Relief-Verfahren auswirken. Da hier auf Grund eines Ausreißers der Wertebereich gestaucht werden kann. So sind hier zukünftig Verfahren zur Eliminierung solcher Ausreißer anzudenken.



(a) Erstes Redundanz-Paar (*phi2* und *phi4*).

(b) Zweites Redundanz-Paar (*principal axis elongation* und *eccentricity moment*).

Bild 5.6: Die ersten beiden Paare aus der Redundanzerkennung (Datensatz 1).

5.2.4 Laufzeiten

Die Experimente wurden auf einem Rechner mit Doppelkern-Prozessor und $2.4\ \mathrm{GHz}$ Taktfrequenz durchgeführt. Tabelle 5.3 zeigt die Laufzeiten der implementierten Verfahren unter den Versuchsbedingungen. Es ist hier zu erkennen, dass die Parallelisierung des Relief-

Verfahren	Laufzeit
DReliefF	$4,444\ \mathrm{s}$
DReliefG	$3,057\ \mathrm{s}$
DReliefF (parallel)	$2,861\ \mathrm{s}$
DReliefG (parallel)	$1,659\ \mathrm{s}$
Redundanz (parallel)	$126,783\ \mathrm{s}$

Tabelle 5.3: Laufzeiten der Algorithmen bei 75 Merkmalen 10 NN und 4200 Objekten.

Verfahrens die Laufzeit quasi halbiert und die Redundanzerkennung so, trotz der 75fachen (bei 75 Merkmalen) Ausführung des Relief-Algorithmus, in akzeptabler Zeit Ergebnisse liefert. Durch diese hohe Zahl der Aufrufe wirkt sich der Laufzeitunterschied von ca. 40% zwischen DReliefG und DReliefF hier auch entsprechend stärker aus.

Da der lineare Klassifikator während der Arbeitsphase eine Zeitkomplexität von $\mathrm{O}(n)$ besitzt, wirkt sich die Merkmalsanzahl auf dessen Laufzeit proportional aus. Die Trainingsphase wird auch entsprechend beeinflusst, wobei die Verkürzung der Trainingszeit hier keine Priorität besitzt (vgl. Gleichung 2.11). Zu den heuristischen Verfahren zur Merkmalskonstruktion liegen keine Laufzeiten vor, da die Tests auf den bereits extrahierten Merkmalsvektoren durchgeführt wurden. Es kann aber davon ausgegangen werden, dass eine entsprechende Reduktion des Merkmalssatzes sich im Durchschnitt ebenfalls proportional auf die Laufzeit des Merkmalskonstruktions-Moduls auswirken wird.

5.3 Zusammenfassung der Ergebnisse

Durch die Experimente wurde für Datensatz 1 aus den vorgegebenen 75 Merkmalen eine Untermenge aus 40 Merkmalen gewonnen, welche für die gewählte Suchstrategie die Fehlerrate des genutzten Klassifikators minimiert. Die Strategie besteht in einem kombinierten Verfahren zur Merkmalsfilterung nach Relevanz- und Redundanz-Kriterien – basierend auf der Relief-Methode – und einem Wrapper-Ansatz, der über die Fehlerrate des Klassifikators das Haltekriterium liefert. Auch ohne Redundanzerkennung liefert dieses Verfahren fast gleichwertige Ergebnisse. Die Unterschiede fallen für das getestete Intervall anfänglich im 1% Bereich aus, bei den Merkmalssätzen ab 25 Merkmalen liefert die reine Relevanz-Methode bessere Ergebnisse. Um die generelle Verbesserung durch die zusätzliche Redundanzerkennung zu validieren, könnte erst an dieser Stelle mit der Merkmalsauswahl nach Redundanz-Kriterien begonnen werden, um dann einen Vergleich der Ergebnisse des kombinierten Verfahrens für dieses Intervall mit denen des reinen Relevanz-Rankings zu ermöglichen. Dies zu validieren war aber keines der Ziele dieser Experimente.

Neben Gewinnung der „optimalen“ Merkmals-Untermenge wurde ersichtlich, dass auch ein kleiner Merkmalssatz noch zu sehr guten Klassifikationsergebnissen führen kann. Sowohl für Datensatz 1 als auch für Datensatz 2 wurden mit den 10 „besten“ Merkmalen noch Fehlerraten von nur ca. 10% erzielt, bei einer ursprünglichen Fehlerrate anhand des kompletten Merkmalssatzes von ca. 8% (für Datensatz 1 und 2).

Es wurde zusätzlich ein Vergleich zwischen der im Rahmen dieser Arbeit entstanden Relief-Variante DReliefG und dem original DReliefF-Algorithmus angestellt. Für beide Datensätze lieferte DReliefG bessere Ergebnisse, wobei dies erst bei kleineren Merkmalssätzen richtig deutlich wurde.

Zur Validierung der Verfahren haben neben den erzielten Klassifikationsergebnissen auch visuelle Überprüfungen durch entsprechende Streudiagramme beigetragen, welche die Plausibilität der entstandenen Rankings stützen. Auch die Gewichtungen einzelner Merkmale lassen Rückschlüsse über die Güte der Ranglisten zu. So besetzen die Fourier-Deskriptoren der höchsten Frequenzen in allen Relevanz-Rankings (via DReliefG und DReliefF) die untersten Plätze. Dass die hochfrequenten Anteile der Konturlinie bei primär runden Objekten zur Unterscheidung einzelner Klassen keinen entscheidenden Beitrag leisten, erscheint plausibel. Das gute Abschneiden des Durchmessers als diskriminatives Merkmal erscheint durch die größtenteils nahezu kugelförmigen Pollen-Objekte ebenfalls stimmig, da der Durchmesser in diesem Fall 3D-rotationsinvariant ist.

Bezieht man die Testergebnisse auf das Pollenmonitor-Projekt, zeigen diese hier ein großes Einsparungspotential auf. Der nahezu konstante Verlauf der Fehlerrate bis zu Merkmalssätzen der Größe 20 lässt auf viele redundante, irrelevante und schwach relevante Merkmale im ursprünglichen Merkmalssatz schließen. Neben der Möglichkeit die Fehlerrate minimal zu senken – unter Reduktion auf einen Merkmalssatz der Größe 40–50, besteht so die Option die Anzahl der Merkmale beträchtlich zu verkleinern, unter nur geringer Steigerung der Fehlerrate. Auf diese Weise könnte z. B. durch die Verwendung nur eines Drittels der Merkmale bei kleiner Steigerung der Fehlerrate – in den Tests ca. 2% – die Klassifikationszeit sowie die Zeit für Merkmalskonstruktion um ca. zwei Drittel gesenkt werden. Dies schafft Raum für rechenintensive Erweiterungen des Klassifikationssystems sowie für die Hinzunahme neuer Merkmale, die infolge zusätzlicher Klassen potentiell benötigt werden. Durch die mit der Reduktion des Merkmalssatzes einhergehende Verbesserung des Verhältnisses zwischen Stichprobengröße N und Dimension des Merkmalsraumes n können die Effekte der „curse of dimensionality“ (vgl. Abschnitt 2.2.3) vermindert werden, ohne aufwendig neue klassifizierte Stichproben erstellen zu müssen. Auch kann so die heuristische Vorgabe für dieses Verhältnis von $N/n > 10$ erfüllt und dadurch eine zuverlässigere Schätzung der Fehlerwahrscheinlichkeit des Klassifikators ermöglicht

werden (vgl. Abschnitt 2.2.3).

Kapitel 6

Zusammenfassung

In dieser Arbeit wurde eine Modifikation des Relief-Verfahrens zur Merkmalsbewertung entworfen und umgesetzt. Desweiteren wurde aus den Eigenschaften dieses Verfahrens eine Heuristik zur Erkennung von Redundanzen im Merkmalsraum abgeleitet, um das diesbezügliche Defizit der Relief-Methode zu beheben. Zur effektiven Anwendung dieser Verfahren im Kontext des „Pollenmonitor“-Projektes wurde eine graphische modulare Evaluationsumgebung erstellt, die sich einer intuitiven GUI-Metapher – inspiriert durch Soft- und Hardware zur Klangsynthese – bedient. In diese wurden die Verfahren zur Merkmalsbewertung sowie der hierarchische Pollen-Klassifikator in Form von Modulen eingebunden. Um eine „gute“ Merkmals-Teilmenge zu finden, wurde eine kombinierte Suchstrategie angewandt, die unter der Vorgabe der Merkmalsbewertungen in Form von Relevanz- und Redundanz-Ranglisten sukzessive Merkmale entfernt und dann über Feedback des Klassifikators die Fehlerrate minimiert. Anhand dieses kombinierten Ansatzes bestehend aus Filter- und Wrapper-Methode wurden mit der Applikation mehrere Experimente im Kontext der Pollenklassifikation durchgeführt. Hierbei konnten die ursprünglichen Merkmalssätze unter Reduktion der Fehlerrate des Klassifikators entscheidend verkleinert werden und damit im Sinne der in Gleichung 2.11 gegeben Kriterien die Güte des gesamten Klassifikationssystems gesteigert werden.

Durch die in dieser Arbeit entwickelte Evaluationsumgebung wird dem Nutzer ein Werkzeug zur Hand gegeben, anhand dessen er die Qualität vorgegebener Merkmale überprü-

fen und daraus eventuell Rückschlüsse auf die Struktur der zu klassifizierenden Objekte ziehen kann. Anhand der Relevanz- und Redundanz-Ranglisten können irrelevante und redundante Merkmale eliminiert und die Datendimension und Komplexität des Klassifikators verringert werden. So können Effekte der „curse of dimensionality“ reduziert und die Laufzeit des Klassifikationsystems verkürzt werden.

6.1 Ausblick

Das in dieser Arbeit verwandte Relief-Verfahren zur Merkmalsbewertung wurde bereits in mehreren Publikationen theoretisch analysiert und praktisch getestet. Dies gilt es für die entwickelte Methode zur Redundanz-Erkennung noch zu tun. Auch Vergleiche mit anderen Methoden zur Redundanz-Eliminierung, z. B. auf Korrelations-Maßen basierende Verfahren, sind noch anzustellen. Die ersten Ergebnisse aus den in dieser Arbeit durchgeführten Experimenten sprechen bereits für das Verfahren. Auch ist von Interesse, wie sich andere Kombinationen der Relevanz- und Redundanz-Rankings auf die Qualität des daraus resultierenden Merkmalssätze auswirken. Es ist auch eine umgekehrte Reihenfolge denkbar, das heißt zuerst eine Redundanz-Eliminierung und dann eine Filterung nach Relevanz-Kriterien. Desweiteren wäre es für Tests im größeren Maßstab sinnvoll, die in den Experimenten verwandte Suchstrategie, die hier noch manuell durchgeführt wurde, zu implementieren und über die Evaluationsumgebung verfügbar zu machen. Auf dem Feld der Merkmalsbewertung und –selektion wird noch aktiv geforscht und publiziert (siehe Literaturverzeichnis). So wird die Anwendung dieses Methoden auch weiter durch Probleme aus der Bioinformatik, wie z. B. der Genselektion, motiviert.

Anhang A

Mathematische Notation

Bezeichner	Bedeutung
U	Umwelt
Ω	Problemkreis
ω	klassifizierte Stichprobe
Ω_κ	Klasse $\in \Omega$
Ω_0	Rückweisungsklasse
y_κ	Klassenlabel der Klasse Ω_κ
${}^{\varrho}\boldsymbol{f}(\boldsymbol{x})$	Muster $\in \Omega_\kappa$
${}^{\varrho}\boldsymbol{f}$	Abtastwerte eines Musters ${}^{\varrho}\boldsymbol{f}(\boldsymbol{x})$
${}^{\varrho}\boldsymbol{c}$	Merkmalsvektor
${}^{\varrho}c_\nu$	Komponente eines Merkmalsvektors ${}^{\varrho}\boldsymbol{c}$
S	Merkmalsselektion : $\subset \{1, \ldots, \nu, \ldots, n\}$
T_A	Analytische, probelmabhängige Transformation zur Merkmalsgewinnung
T_H	Heuristische, problemunabhängige Transformation zur Merkmalsgewinnung

Anhang B

Implementationsdetails

B.1 Implementierung der graphischen Komponente eines Modules

1. Wahl der Art des Moduls durch Ableitung von der entsprechenden Basis-Klasse, hier `PlotModule`.

```
1  package de.fraunhofer.fit.pm.evaluation.gui.logic;
2
3  import java.beans.PropertyChangeListener;
4
5  public class ExamplePlotModule extends PlotModule
6  {
7      public void processData(PropertyChangeListener l) throws Exception
8      {
9
10     }
11 }
```

2. Wahl der Modul-Parameter und Implementierung der *Getter* und *Setter* (beide müssen hier als `public` implementiert sein, damit die entsprechenden GUI-Komponenten zur Laufzeit durch die Reflexion-API automatisch erzeugt werden).

```
1  package de.fraunhofer.fit.pm.evaluation.gui.logic;
2
3  import java.beans.PropertyChangeListener;
4
5  public class ExamplePlotModule extends PlotModule
6  {
7
```

```
8       public boolean _paintItRed;
9
10      public boolean isPaintItRed()
11      {
12          return _paintItRed;
13      }
14
15      public void setPaintItRed(boolean paintItRed)
16      {
17          _paintItRed = paintItRed;
18      }
19
20      public void processData(PropertyChangeListener l) throws Exception
21      {
22
23      }
24  }
```

3. Optionales setzen der Standard-Werte, durch JAVA-Annotationen.

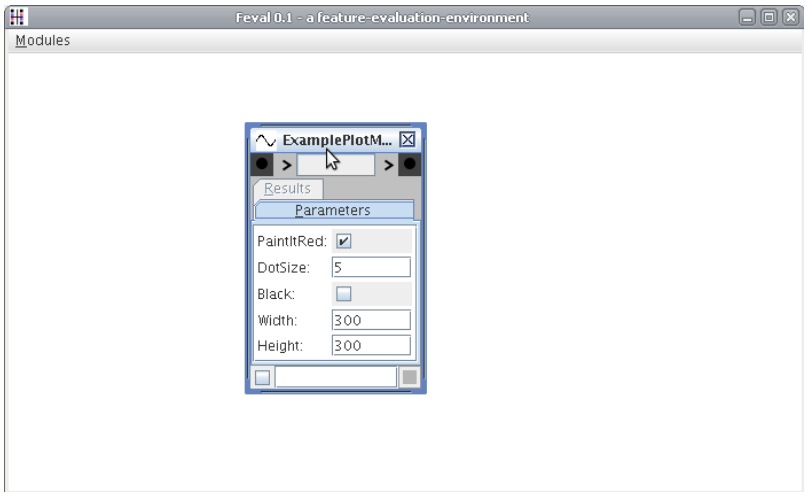
```
1   package de.fraunhofer.fit.pm.evaluation.gui.logic;
2
3   import java.beans.PropertyChangeListener;
4
5   public class ExamplePlotModule extends PlotModule
6   {
7
8       public boolean _paintItRed;
9
10      public boolean isPaintItRed()
11      {
12          return _paintItRed;
13      }
14
15      @DefaultState(defaultValue = true)
16      public void setPaintItRed(boolean paintItRed)
17      {
18          _paintItRed = paintItRed;
19      }
20
21      public void processData(PropertyChangeListener l) throws Exception
22      {
23
24      }
25  }
```

4. Registrierung des neuen Moduls bei der Basis-Klasse `Module`.

```
1   public abstract class Module
2   {
3       public static final Class[] MODULES = {
4               PollenFileInputModule.class,
5               SimpleEvaluationObjectFileInputModule.class,
6               FevalMenuSeparator.class,
7               FeatureScatterPlotModule.class,
8               CorrelationPlotModule.class,
9
10              ExamplePlotModule.class,
11
12              FevalMenuSeparator.class,
13              ReliefFRelevanceModule.class,
```

```
14              FevalMenuSeparator.class,
15              ReliefRedundancyModule.class,
16              FevalMenuSeparator.class,
17              IntersectionModule.class,
18              FevalMenuSeparator.class,
19              LDAPollenClassificationModule.class,
20              };
```

5. Neues `ExamplePlotModule` in der Evaluations-Umgebung verfügbar, mit GUI-Komponete für das Modul-Parameter `paintItRed` und gesetztem Standard-Wert.



6.

Anhang C

CD-ROM

C.1 Inhalt der beiliegenden CD-ROM

Verzeichnis	Inhalt
Ausarbeitung/	Schriftlicher Teil dieser Arbeit in LaTeX-Quellcode und mit beinhalteten Bildern.
Programmcode/	Implementierter JAVA-Quellcode der Evaluationsumgebung, benötigte Klassen-Bibliotheken (ohne Klassifikator aus der FIT Bildverarbeitungs-Bibliothek), ANT-Skript zur Erzeugung der lauffähigen JAR-Datei, BASH-Skript zum Starten der Applikation und JAVADOC-Dokumentation des Quellcodes.
Literatur/	Digitale Versionen der referenzierten Literatur, sofern frei verfügbar.
Testdatensatz/	Benutzter Testadtensatz (Klassennamen wurden aus verwertungsgrechtlichen Gründen anonymisiert).

Literaturverzeichnis

[ATSK08] AJIOKA, Shiro ; TSUGE, Satoru ; SHISHIBORI, Masami ; KITA, Kenji: Fast multidimensional nearest neighbor search algorithm using priority queue. In: *Electrical Engineering in Japan* 164 (2008), Nr. 3, S. 69–77. http://dx.doi.org/10.1002/eej.20502. – DOI 10.1002/eej.20502

[Bis06] BISHOP, Christopher M.: *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, 2006. – ISBN 0387310738

[DHS00] DUDA, Richard O. ; HART, Peter E. ; STORK, David G.: *Pattern Classification (2nd Edition)*. Wiley-Interscience, 2000. – ISBN 0471056693

[Die97] DIETTERICH, Thomas G.: Machine-learning research: four current directions. In: *AI Magazine* 18 (1997), S. 97–136

[DL97] DASH, M. ; LIU, H.: Feature selection for classification. In: *Intelligent Data Analysis* 1 (1997), Nr. 1-4, S. 131–156. http://dx.doi.org/10.1016/S1088-467X(97)00008-5. – DOI 10.1016/S1088–467X(97)00008–5

[DL03] DASH, Manoranjan ; LIU, Huan: Consistency-based search in feature selection. In: *Artif. Intell.* 151 (2003), Nr. 1-2, S. 155–176. http://dx.doi.org/10.1016/S0004-3702(03)00079-1. – DOI 10.1016/S0004–3702(03)00079–1. – ISSN 0004–3702

[DLM00] DASH, Manoranjan ; LIU, Huan ; MOTODA, Hiroshi: Consistency Based Feature Selection. Version: 2000. http://dx.doi.org/10.1007/3-540-45571-X_12. In: *Knowledge Discovery and Data Mining. Current*

Issues and New Applications. 2000. – DOI 10.1007/3–540–45571–X_12, S. 98–109

[Duc06] DUCH, Google: Filter methods. In: *Feature extraction, foundations and applications*, 2006, 89–118

[GBA+03] GUYON, I. ; BITTER, H. M. ; AHMED, Z. ; BROWN, M. ; HELLER, J.: Multivariate nonlinear feature selection with kernel multiplicative updates and gram-schmidt relief. In: *2003 BISC FLINT-CIBI workshop* (2003). http://citeseerx.ist.psu.edu/showciting;jsessionid=C3AC3B6AB79A5ECAA7EC9928D5338F5D?cid=3471819

[GGNZ06] GUYON, Isabelle ; GUNN, Steve ; NIKRAVESH, Masoud ; ZADEH, Lotfi A.: *Feature Extraction: Foundations and Applications (Studies in Fuzziness and Soft Computing)*. Secaucus, NJ, USA : Springer-Verlag New York, Inc., 2006. – ISBN 3540354875

[Hala] HALBRITTER, H.: Abies cephalonica. In: *Buchner R. and Weber M. (200 onwards). Paldat - a palynological database: Descriptions, illustrations, identification and information retrival.* http://www.paldat.org

[Halb] HALBRITTER, H.: Ostrya carpinifolia. In: *Buchner R. and Weber M. (200 onwards). Paldat - a palynological database: Descriptions, illustrations, identification and information retrival.* http://www.paldat.org

[HDW94] HOLMES, G. ; DONKIN, A. ; WITTEN, I. H.: WEKA: a machine learning workbench. In: *Intelligent Information Systems,1994. Proceedings of the 1994 Second Australian and New Zealand Conference on*, 1994, S. 357–361

[JDM00] JAIN, Anil K. ; DUIN, Robert P. W. ; MAO, Jianchang: Statistical Pattern Recognition: A Review. In: *IEEE Trans. Pattern Anal. Mach. Intell.* 22 (2000), January, Nr. 1, S. 4–37. http://dx.doi.org/10.1109/34.824819. – DOI 10.1109/34.824819. – ISSN 0162–8828

[KC02] KWAK, N. ; CHOI, Chong-Ho: Input feature selection for classification problems. In: *Neural Networks, IEEE Transactions on* 13 (2002), Nr.

1, S. 143–159. http://dx.doi.org/10.1109/72.977291. – DOI 10.1109/72.977291

[KJ97] KOHAVI, Ron ; JOHN, George H.: Wrappers for feature subset selection. In: *Artif. Intell.* 97 (1997), Nr. 1-2, S. 273–324. http://dx.doi.org/10.1016/S0004-3702(97)00043-X. – DOI 10.1016/S0004–3702(97)00043–X. – ISSN 0004–3702

[Kon94] KONONENKO, Igor: Estimating Attributes: Analysis and Extensions of RELIEF. In: *European Conference on Machine Learning*, 1994, 171–182

[KR92a] KIRA, Kenji ; RENDELL, Larry A.: The Feature Selection Problem: Traditional Methods and a New Algorithm. In: *AAAI*. Cambridge, MA, USA : AAAI Press and MIT Press, 1992, S. 129–134

[KR92b] KIRA, Kenji ; RENDELL, Larry A.: A practical approach to feature selection. In: *ML92: Proceedings of the ninth international workshop on Machine learning*. San Francisco, CA, USA : Morgan Kaufmann Publishers Inc., 1992. – ISBN 15586247X, 249–256

[KS96] KOLLER, Daphne ; SAHAMI, Mehran: Toward Optimal Feature Selection. In: *International Conference on Machine Learning*, 1996, 284–292

[MKSW99] MAKHOUL, John ; KUBALA, Francis ; SCHWARTZ, Richard ; WEISCHEDEL, Ralph: Performance measures for information extraction. In: *In Proceedings of DARPA Broadcast News Workshop*, 1999, 249–252

[MPH07] MAATEN, L. J. P. d. ; POSTMA, E. O. ; HERIK, H. J. d.: *Dimensionality Reduction: A Comparative Review*. Published online. http://www.cs.unimaas.nl/l.vandermaaten/dr/DR_draft.pdf. Version: 2007

[MWK+06] MIERSWA, I. ; WURST, M. ; KLINKENBERG, R. ; SCHOLZ, M. ; EULER, T.: *YALE: Rapid prototyping for complex data mining tasks*. http://citeseer.ist.psu.edu/mierswa06yale.html. Version: 2006

[Nie07] NIEMANN, Heinrich: *Klassifikation von Mustern, 2. überarbeitete und erweiterte Auflage im Internet*. 2007 http://www5.informatik.uni-erlangen.de/Personen/niemann/klassifikation-von-mustern/m00links.html

[PLD05] PENG, H. ; LONG, F. ; DING, C.: Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy. In: *IEEE Trans Pattern Anal Mach Intell* 27 (2005), August, Nr. 8, S. 1226–1238. http://dx.doi.org/10.1109/TPAMI.2005.159. – DOI 10.1109/TPAMI.2005.159. – ISSN 0162–8828

[RG88] RODRÍGUEZ-GARCÍA, M.: A review of the terminology applied to apertural thickenings of the pollen grain: Zwischenkörper or oncus? In: *Review of Palaeobotany and Palynology* 54 (1988), February, Nr. 1-2, S. 159–163. http://dx.doi.org/10.1016/0034-6667(88)90011-5. – DOI 10.1016/0034–6667(88)90011–5. – ISSN 00346667

[Rob] ROBNIK, Marko: *Speeding up Relief algorithms with k-d trees*. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.43.3453

[Ron07] RONNEBERGER, O.: *3D invariants for automated pollen recognition*, Albert-Ludwigs-Universität Freiburg, Diss., 2007

[RS04] RAILEANU, Laura ; STOFFEL, Kilian: Theoretical Comparison between the Gini Index and Information Gain Criteria. In: *Annals of Mathematics and Artificial Intelligence* 41 (2004), May, Nr. 1, S. 77–93. http://dx.doi.org/10.1023/B:AMAI.0000018580.96245.c6. – DOI 10.1023/B:AMAI.0000018580.96245.c6

[RvK03] ROBNIK-ŠIKONJA, Marko ; KONONENKO, Igor: Theoretical and Empirical Analysis of ReliefF and RReliefF. In: *Machine Learning* 53 (2003), October, Nr. 1, S. 23–69. http://dx.doi.org/10.1023/A:1025667309714. – DOI 10.1023/A:1025667309714

[SK97] SIKONJA, M. ; KONONENKO, I.: *An adaptation of Relief for attribute estimation in regression*. http://citeseer.ist.psu.edu/85013.html. Version: 1997

[SL06] SUN, Yijun ; LI, Jian: Iterative RELIEF for feature weighting. In: *ICML '06: Proceedings of the 23rd international conference on Machine learning*. New York, NY, USA : ACM, 2006. – ISBN 1595933832, S. 913–920

[Tip] TIPLER, Frank J.: *Die Physik der Unsterblichkeit. Moderne Kosmologie, Gott und die Auferstehung der Toten*. 5. A. Piper. – ISBN 3492036112

[TJ93] TUCERYAN, Mihran ; JAIN, Anil K.: Texture analysis. (1993), S. 235–276

[WOW01] WINKLER, Helga ; OSTROWSKI, René ; WILHELM, Margarete: *Pollenbestimmungsbuch der Stiftung Deutscher Polleninformationsdienst*. Stiftung Deutscher Polleninformationsdienst, D-33175 Bad Lippspringe, 2001. – ISBN 3931732096

[YAH+08] YE, Kai ; ANTON ; HERINGA, Jaap ; IJZERMAN, Adriaan P. ; MARCHIORI, Elena: Multi-RELIEF: a method to recognize specificity determining residues from multiple sequence alignments using a Machine-Learning approach for feature weighting. In: *Bioinformatics* 24 (2008), January, Nr. 1, S. 18–25. http://dx.doi.org/10.1093/bioinformatics/btm537. – DOI 10.1093/bioinformatics/btm537

[YL04] YU, Lei ; LIU, Huan: Efficient Feature Selection via Analysis of Relevance and Redundancy. In: *J. Mach. Learn. Res.* 5 (2004), 1205–1224. http://portal.acm.org/citation.cfm?id=1005332.1044700. – ISSN 1533–7928

[ZDL08] ZHANG, Yi ; DING, Chris ; LI, Tao: Gene selection algorithm by combining reliefF and mRMR. In: *BMC Genomics* 9 (2008), Nr. Suppl 2. http://dx.doi.org/10.1186/1471-2164-9-S2-S27. – DOI 10.1186/1471–2164–9–S2–S27. – ISSN 1471–2164