Softcomputing:

Wie Maschinen (erkennen) lernen

Selbstlernende Algorithmen werden in der heutigen Zeit immer wichtiger. Denn Daten- und Informationsfluten können vom Menschen selbst kaum mehr betrachtet und ausgewertet werden. Das beginnt bei der Mustererkennung in Fertigungsstraßen, geht über zur Überwachung und Kontrolle von Personen und Objekten und endet in Bereichen der Diagnostik in der Medizin. Immer wieder stellt sich die Aufgabe, Zusammenhänge oder Muster zu erkennen und diese einzuordnen, also zu klassifizieren. In diesem Kontext häufig auftretende Fragen sind zum Beispiel "Ist das unter dem Scanner Musterstück A oder Musterstück B?" oder "Befindet sich auf dem Überwachungsbild eine Person?".

All diese Aufgaben liegen im Bereich des sogenannten Softcomputing. Klassische Einsatzgebiete des Softcomputing sind Probleme, die keinen starren Regeln folgen – was bedeutet, dass die eingesetzten Verfahren tolerant gegenüber Unsicherheiten, Ungenauigkeiten und Rauschen sein müssen. Konventionelle starre Verfahren der Bildverarbeitung sind häufig nicht in der Lage, solche Aufgaben adäquat zu bearbeiten.

Im Bereich des Maschinenlernens existieren allerdings diverse Ansätze: Künstliche neuronale Netze, genetische Algorithmen, RBF-Netzwerke (Radiale Basis Funktions Netzwerke) und Support Vector Machines, um nur einige zu nennen. Wie auch das im Folgenden beschriebene Verfahren sind die meisten Algorithmen dem Bereich des überwachten Lernens zuzuordnen.

Prinzipiell ist überwachtes maschinelles Lernen durch folgendes Vorgehen gekennzeichnet:

- Bei jedem Lernschritt wird dem System eine Eingangs- und die dazu erwartete Ausgangsgröße übergeben.
- Anschließend wird die Leistungsfähigkeit überprüft und das System aufgrund möglicherweise aktuell auftretender Fehler korrigiert.
- Dieser Vorgang wird so lange wiederholt, bis das System wie gewünscht und fehlerfrei reagiert: Damit ist der Lernvorgang abgeschlossen.

Ziel der Lernphase ist es, neue Eingangsdaten einer der erlernten Ausgangsklassen eindeutig zuzuordnen.

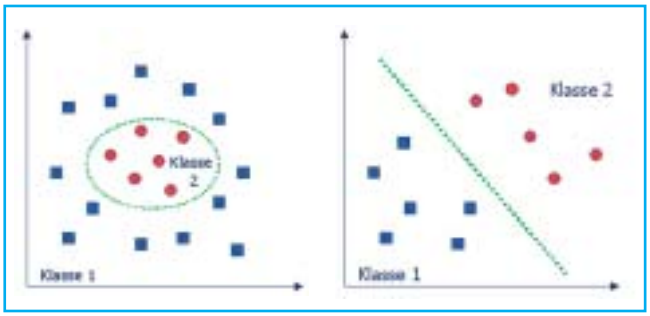


Abb. 1: Nicht linear (links) und linear trennbare Probleme

Algorithmen

Das wohl gängigste und bekannteste Verfahren aus der Gruppe der selbstlernenden Algorithmen sind die künstlichen neuronalen Netze. Sie sind, zumindest in ihrer simpelsten Form, einfach und intuitiv zu erfassen – ganz im Gegensatz zu den Support Vector Machines (SVMs), die ein mathematisch komplexes und wenig intuitives Verfahren darstellen. Die beiden Ansätze differieren nicht in ihrer Repräsentationskapazität, also in dem, was sie abbilden können, dafür aber grundlegend in der Art des Lernens.

Die Support Vector Machines, Kernthema dieses Aufsatzes, sind ein noch recht junges Verfahren, welches seinen Durchbruch erst Mitte der 90er Jahre mit einer Anwendung zur Erkennung handgeschriebener Ziffern und Buchstaben hatte. Sie haben gegenüber künstlichen neuronalen Netzen einige Vorteile, wie beispielsweise Robustheit gegen Overfitting und deutlich schnellere Klassifizierung. Leider bestehen immer noch weit verbreitete Vorbehalte gegenüber SVMs, da sie mathematisch sehr anspruchsvoll und wenig intuitiv sind.

Zur Veranschaulichung der Funktionsweise eines künstlichen neuronalen Netzes sind verschiedene Anwendungen verfügbar (auch frei), so beispielsweise eine zur Erkennung handgeschriebener Ziffern. Ein anschauliches Beispiel zur Darstellung des Lernprozesses einer SVM war bisher aber nicht zu finden. Es sind zwar Programme zugänglich, die das Klassifizierungsvermö-

gen einer angelernten SVM oder die Reduktion genereller Fehler während des Lernprozesses zeigen, jedoch keine Anwendung, die den Lernvorgang des Verfahrens für den Betrachter erfassbar macht. Diese Lücke sollte durch meine Abschlussarbeit "Support Vector Machine zur Gender Recognition mit Visualisierung des Lernprozesses" geschlossen werden.

Lineare Trennung

Wie bereits erwähnt nutzen SVMs das Verfahren des überwachten Lernens: Sie arbeiten mit binärer Klassifizierung, also der Unterscheidung von genau zwei Klassen mit Hilfe linearer Trennung der Daten.

Linear trennbare Probleme haben einen großen Vorteil: Die Funktion, welche die Grenze zwischen den Klassen darstellt, ist mathematisch sehr einfach und deshalb auch mit vergleichsweise wenig Aufwand lösbar. Entscheidungen bei der Klassifizierung können daher sehr schnell getroffen werden. Ein einfaches Beispiel zur Unterscheidung von linear trennbaren und nicht linear trennbaren Problemen zeigt Abb. 1.

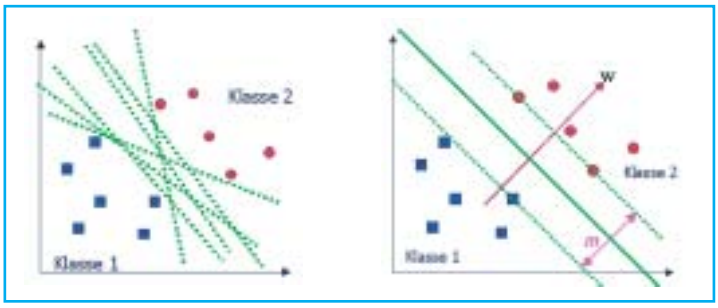
Margin Classifier und Trade-off

Doch selbst bei offensichtlich linear trennbaren Problemen ist es schwierig, die optimale Trennfunktion, welche die beste Generalisierung bietet, zu finden. Gute Generalisierung bedeutet, dass die Klassen so getrennt werden, dass unbekannte Daten bestmöglich zugeordnet werden. Ein sogenannter "Maximum Margin Classifier" – eine Trennfunktion, welche den größtmöglichen Abstand zu den Klassengrenzen hat – löst das Problem (Abb. 2).

Das Prinzip des Maximum Margin Classifiers lässt sich noch verfeinern, indem es um eine sogenannte "Slack Variable" – eine Schlupfvariable (ξ) – erweitert wird, welche Ausreißer dem Rand zuteilt. Diese Sonderform des Maximum Margin Classifiers, welche durch ihren "weichen" Rand eine gewisse Fehlertoleranz erhält, wird "Soft Margin Classifier" genannt. Sie kann auch bei leicht überlappenden, also gerade nicht mehr linear trennbaren Klassen, eingesetzt werden (siehe Abb. 3).

SVMs nutzen den Soft Margin Classifier. Sie trennen also die Klassen linear mit möglichst großem Rand und weisen Ausreißern gegenüber eine gewisse Toleranz auf. Dazu verfügen sie über einen anpassbaren "Trade-off"-Parameter, der ähnlich der Schlupfvariablen den Kompromiss zwischen Fehler und Rand beschreibt. So werden Ausreißer beim Lernprozess dem Rand zugeteilt und fallen nicht so stark ins Gewicht. Dadurch wird auch das "Overfitting", also die zu genaue Anpassung der Trennfunktion an die Trainingsdaten und damit eine schlechtere Generalisierung, verhindert.

Abb. 2: Maximum Margin Classifier

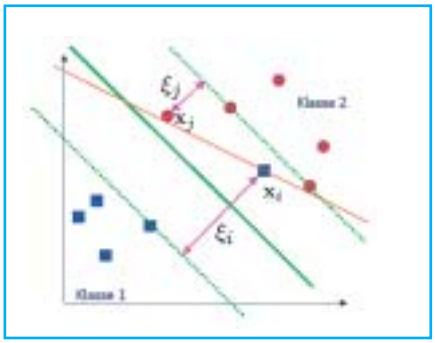


Doch woher weiß die SVM, welche Daten sich an den Klassengrenzen befinden? Die Ermittlung dieser Daten findet über eine geeignete mathematische Transformation in eine andere Repräsentation statt, zu deren Lösung das Lagrange-Verfahren verwendet wird. Aus der entstehenden Schreibweise lassen sich dann die Lagrange-Multiplikatoren, die sogenannten Alphawerte, ablesen. Für alle Daten, die an der Klassengrenze liegen, sind diese Werte ungleich Null. Der Alphawert kann auch als eine Art Strafwert betrachtet werden: Je schwieriger ein Datum einzuordnen ist, desto höher ist dieser Wert. Herleitung und mathematische Grundlagen des Formalismus können den Literaturvorschlägen entnommen werden.

Mapping

In der realen Welt sind aber linear trennbare Probleme nicht zu finden. Um dennoch deren Vorteile nutzen zu können wird ein recht einfacher Trick verwendet, das

Abb. 3: Soft Margin Classifier



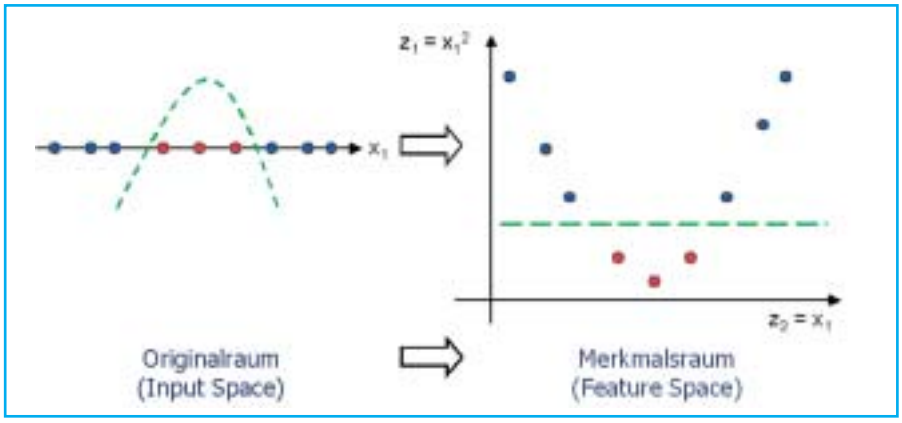


Abb. 4: Überführung des Original- in den Merkmalsraum (Mapping)

sogenannte "Mapping". Dessen Kernaussage lautet: Jedes Problem ist linear trennbar, wenn es sich nur in der richtigen Dimension befindet. Folglich überführt man die nicht linear trennbaren Daten aus dem Eingabe- oder Originalraum ("Input Space") in den sogenannten Merkmalsraum ("Feature Space"). Dabei werden die Daten stets als Vektoren betrachtet. Beispielhaftes Mapping zeigen die Abb. 4 und 5.

Infolge des Mappings wird bei der trennenden Funktion nun von einer Hyperebene gesprochen. Hyperebenen stellen die Erweiterung des anschaulich fassbaren Ebenenbegriffs im 3-dimensionalen Raum auf ein mathematisches Objekt im n-dimensionalen Raum dar. Eine Hyperebene in einem n-dimensionalen Raum ist damit (n-1)-dimensional, eine Hyperebene im 2-dimensionalen Raum also eine Gerade, im 3-dimensionalen eine Ebene.

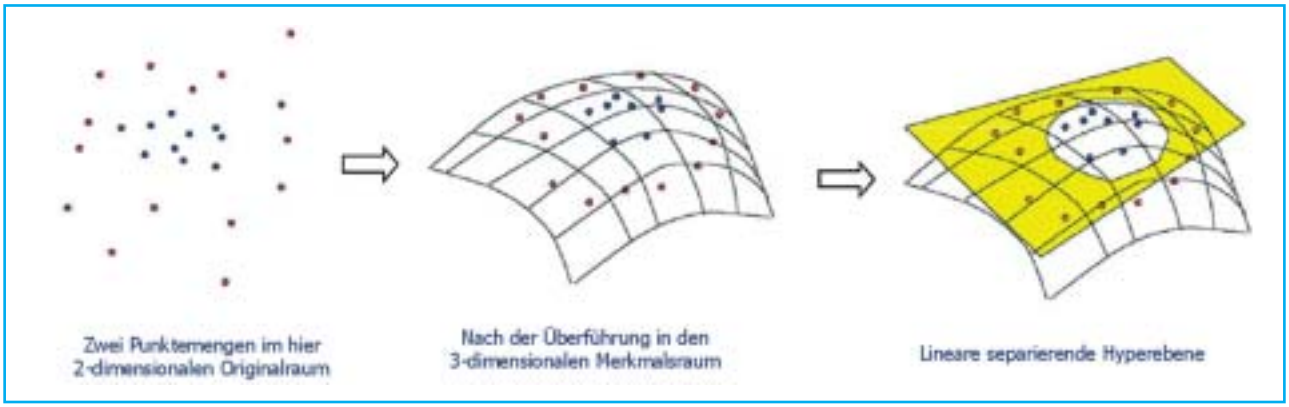
Doch auch das Mapping ist nicht unproblematisch. Zum einen bereitet die Auswahl eines passenden Mappings bei realen Problemen erhebliche Schwierigkeiten, zum anderen sagt das "Curse of Dimensionality"-Phänomen, dass der Rechenaufwand mit der Anzahl der Dimensionen enorm steigt. So führt beispielsweise der Versuch, ein Polynom 4ten oder 5ten Grades in einem 256-dimensionalen Raum zu erzeugen, zur Erzeugung einer Hyperebene im billionen-dimensionalen Raum! Selbst mit der heutigen Technik ist das kaum berechenbar.

Kernel Trick

Doch auch diese Schwierigkeit wurde überwunden: Support Vector Machines arbeiten mit dem sogenannten "Kernel-Trick". Das Mapping und die benötigten Skalarprodukte werden durch eine Kernelfunktion ersetzt. Das Ergebnis der Kernelfunktionen auf den Vektoren im Eingaberaum entspricht dem Skalarprodukt der transformierten Vektoren im Merkmalsraum und macht somit explizites Mapping unnötig.

Neben der enormen Reduktion des Rechenaufwandes bietet der Kernel-Trick weitere Vorteile. Zum einen können so SVMs konstruiert werden, die in einem unendlich-dimensionalen Raum arbeiten, wobei diese Dimension niemals real auftritt. Zum anderen muss das konkrete Mapping nicht bekannt sein und auch nicht berechnet werden, denn es muss ledig-

Abb. 5: Hyperebenen und Mapping



lich die Auswahl des Kernels getroffen werden. Hierbei kann aus zahlreichen erprobten Kernelfunktionen eine für das gegebene Problem passende ermittelt werden.

Aus mathematischer Sicht lässt sich eine SVM daher wie folgt darstellen:

$$h(x) = \operatorname{sign}\left(\sum_{i=1}^{n} \alpha_i y_i K(x, x_i) + b\right)$$

$$L_d(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\left(\sum_{i=1,j=1}^{n} \alpha_i \alpha_j y_i y_j K(x_i, x_j)\right)$$

$$\text{mit} \quad C \geq \alpha_i \geq 0 \quad \text{für alle } i; \quad \sum_{i=0}^{n} \alpha_i y_i = 0$$

Diese Formeln enthalten sowohl den Alphawert (α) und den Trade-off-Parameter (C) als auch die Eingabedaten (x_i), ihre Klassen (y_i), das zu klassifizierende Datum (x), die Transformation durch den Kernel K und einen so genannten Bias-Wert (b), der den linearen Versatz der Funktion beschreibt. Insgesamt ergeben diese die Entscheidungsfunktion h(x). Eine ausführliche Beschreibung der Schreibweise und ihre Bedeutung kann der empfohlenen Fachliteratur oder der eingangs erwähnten Abschlussarbeit entnommen werden.

Gender Recognition

Für die Visualisierung des Lernprozesses wurde die "Gender Recognition", also das Erkennen des Geschlechts einer Person anhand eines Bildes des Gesichts, gewählt.

Gender Recognition bzw. "Gender Classification" ist ein in der Psychologie häufig diskutiertes Thema, für welches es jedoch nur wenige computerbasierte Ansätze gibt. Gender Recognition ist keineswegs trivial und folgt auch keinen starren Regeln, ist also bestens geeignet für das oben beschriebene Softcomputing. In der Praxis spielt die Gender Recognition im Gegensatz zur "Face Recognition" (Wiedererkennung von Gesichtern) und "Face Detection" (Auffinden von Gesichtern in einem Bild) nahezu keine Rolle. Sie ist jedoch ein für Menschen sehr gut nachvollziehbares Klassifizierungsproblem und wegen der erforderlichen ausgesprochen feinen Differenzierung hochinteressant.

Feature oder Template?

Von Menschen instinktiv und aufgrund jahrelanger Erfahrung getroffene Einschätzungen sind durch eine Maschine bekanntlich nur schwer nachzuvollziehen. Für die adressierte Problematik bedeutet das, dass zunächst die für eine Klassifizierung relevanten Gesichtsmerkmale identifiziert werden müssen. Hierzu kann zwei allgemein propagierten Ansätzen gefolgt werden. Das sogenannte "feature-basierte" Verfahren, welches aus einem Bild zunächst die gewünschten Features extrahiert und diese dann verarbeitet, bezieht sich allein auf geometrische Merkmale des Gesichts (Abb. 6).

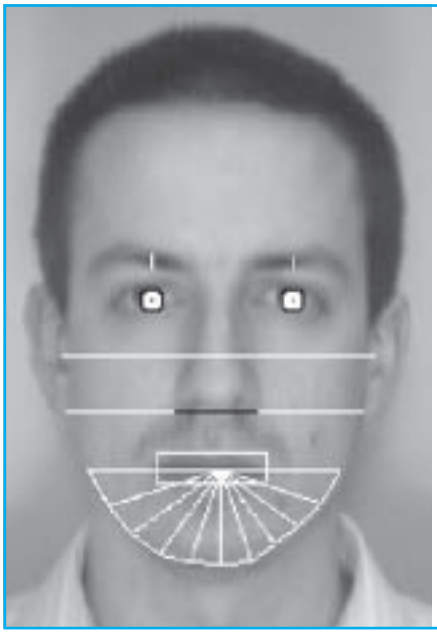


Abb. 6: Geometrische Merkmale des Gesichts: pupil to eyebrow separation; pupil to nose vertical distance; pupil to mouth vertical distance; pupil to chin vertical distance; eyebrow thickness; nose width; mouth width; bizygomatic breadth; bigonial breadth; six chin radii; mouth height

Hingegen kann das sogenannte "template-basierte" Verfahren als ganzheitlicher Ansatz verstanden werden. Es werden hier die "kompletten" Bilder als Eingabedaten genutzt, sie sind lediglich bereits normiert und auf relevante Bereiche beschnitten. Es findet also keine Extraktion von Merkmalen aus den Bildern statt. Dadurch ergibt sich ein deutlich breiteres Informationsspektrum: Nicht nur geometrische, sondern auch topologische und somit holistische Informationen werden berücksichtigt.

Beide Verfahren haben Vor- und Nachteile. So ist beispielsweise die Verarbeitung der Templates weniger aufwendig, dafür benötigen sie mehr Speicherplatz. Insgesamt aber hat sich der template-basierte Ansatz als erfolgreicher herausgestellt, da bei diesem Vorgehen keine Information verloren geht.

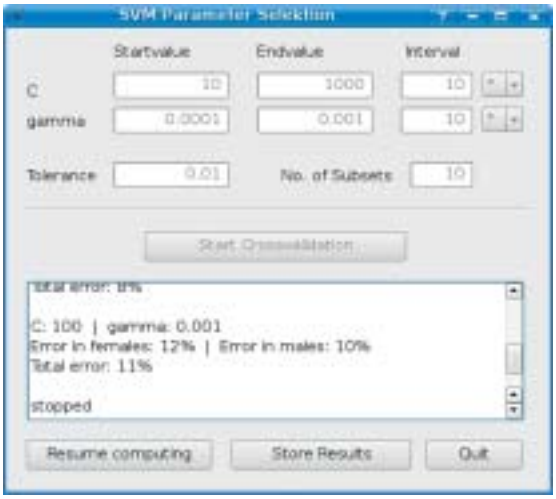


Abb. 7 - 9: Einfache übersichtliche Oberflächen: Parameter Selektion, Trainer, Klassifizierung (im Uhrzeigersinn)

Das template-basierte Verfahren zur Gender Recognition und die Art der Verarbeitung der Daten in einer SVM ergänzen sich sehr gut. Support Vector Machines sind darauf ausgelegt, hochdimensionale Vektoren, welche durch das template-basierte Verfahren entstehen (in Fall dieser Arbeit 10304-dimensionale Vektoren), effektiv zu verarbeiten. Für den Aspekt der Visualisierung ist zudem entscheidend, dass die Ausgangsbilder bei dieser Art der Umsetzung erhalten bleiben (bis auf die erwähnten Anpassungen) und die Klassifizierungsaufgabe somit für den menschlichen Betrachter nachvollziehbar ist.

Lernschritte

Für die Implementierung der SVM wurde das spezielle SMO-Verfahren ("Sequential Minimal Optimization") eingesetzt, welches derzeit am häufigsten verwendet wird. Dieses Verfahren führt die folgenden Lernschritte durch:

1) Optimierung von immer zwei, heuristisch ausgewählten Trainingsbeispielen gegeneinander durch Anpassung der Trennfunktion, so dass sie diese Beispiele optimal trennt. Dieses erfolgt durch Anpassung der Alphawerte der beiden Trainingsbeispiele.
2) Neue Berechnung der Fehler für die restlichen Trainingsbeispiele (der Fehler eines Datums ist die Abweichung vom Klassenoptimum bei Verwendung der aktuellen Trennfunktion).
3) Wiederholung der ersten beiden Schritte, bis das gewünschte Ergebnis erzielt ist, also entweder die angestrebte Toleranz erreicht ist oder alle möglichen Optimierungen ausgeschöpft sind. Als Ergebnis des Lernvorgangs werden die Support Vektoren, die die Hyperebene bestimmen, anhand der Alphawerte ermittelt und alle gelernten Bilder richtig klassifiziert.

Software-Suite

Im Verlauf der beschriebenen Arbeit wurden verschiedene Programme entwickelt. Diese ermöglichen zum einen dem interessierten Benutzer, sich selbstständig mit Support Vector Machines zu befassen. Zum anderen können sie aber auch zu Präsentationszwecken eingesetzt werden, beispielsweise im Rahmen einer Vorlesung.

Ein einfaches Interface (Abb. 7) dient der Parameterbestimmung für die SVM. Hier werden die optimalen Werte für die Kernel- und den Trade-off-Parameter in der aktuellen Konstellation ermittelt. Ein wei-

teres (Abb. 8), welches zugleich die Daten für die spätere Visualisierung aufzeichnet, ist zuständig für das Anlernen der SVM. Ein drittes schließlich (Abb. 9) ermöglicht die Klassifizierung ungesehener Daten mit Hilfe einer beliebigen SVM, die mit einem entsprechenden Programm bereits angelernt wurde. Darüber hinaus wurden zahlreiche Skripte zur Vorverarbeitung der Bilddaten bereitgestellt. Kernstück dieser Programme ist die Software zur Visualisierung des Lernvorgangs (Abb. 10 und 11), welche sich der Daten, die sich während der Lernphase ändern, bedient. Das sind zum einen die Alphawerte, welche die Support Vektoren identifizieren, zum anderen der aktuelle Fehlerwert der einzelnen Trainingsdaten.

Wie bereits beschrieben wählt das SVM-SMO-Verfahren bei einem einzelnen Lernschritt jeweils zwei Erfolg versprechende Eingabedaten aus und optimiert sie gegeneinander, wodurch sich die Alphawerte der jeweiligen Eingabedaten verändern. Anhand dieser Veränderung kann verfolgt werden, welche Eingabedaten sich im Laufe des Verfahrens als die "schwierigsten" herausstellen und wie sich diese Einordnung entwickelt. Nach Abschluss des Lernvorgangs ist ersichtlich, welche der Eingabedaten Support Vektoren sind und sich somit an der Klassengrenze befinden. Diese Eingabedaten beschreiben die Trennfunktion. In der Software werden alle Eingabedaten als Bilder angezeigt und je nach Alphawert mit einem grünen Overlay versehen (Abb. 12).

Zusätzlich kann der Benutzer die Veränderung der Fehlerwerte der beiden jeweils aktuell zu optimierenden Trainingsdaten verfolgen. Dabei ist zu beobachten, dass die Abweichungen von den Klassengrenzen anfangs noch recht groß sind und die Optimierungen in der letzten Phase des Lernvorgangs in sehr kleinen Schritten erfolgen. Zudem entnimmt man der Anzeige, welche beiden Trainingsbeispiele für eine Optimierung den größten Erfolg versprechen (Abb. 13).

Zusammenfassung

Im Rahmen dieser Abschlussarbeit ist nicht nur eine zu Lehr- und Lernzwecken nützliche Software-Suite entstanden, sondern auch ein überzeugend klassifizierendes Verfahren. Als Datenbanken wurden

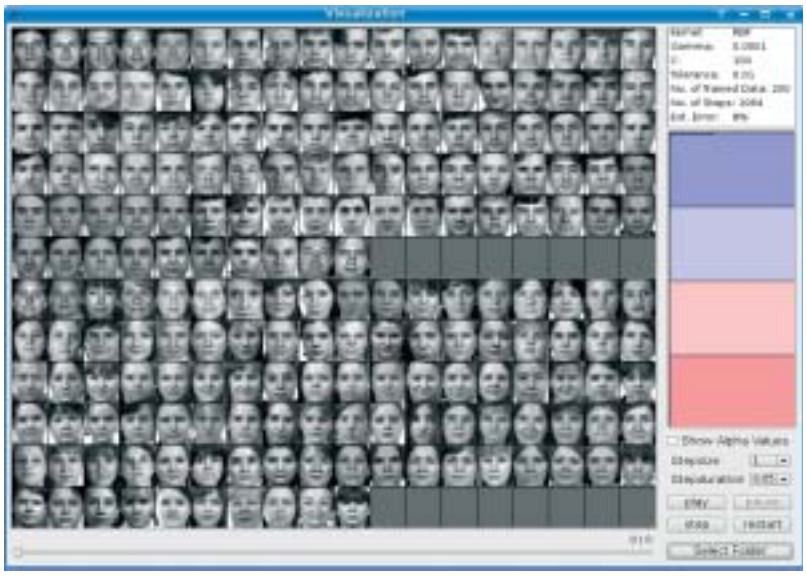


Abb. 10

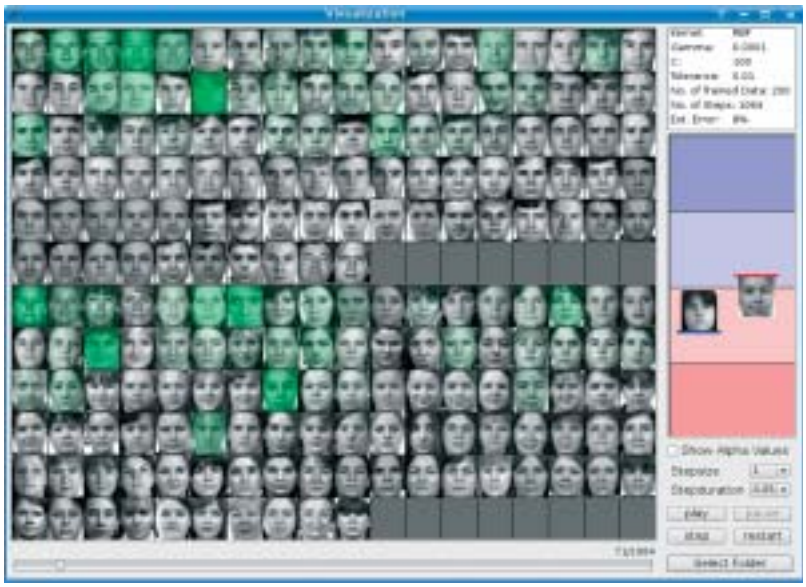


Abb. 11

Visualisierung des Lernverlaufs

Abb. 12

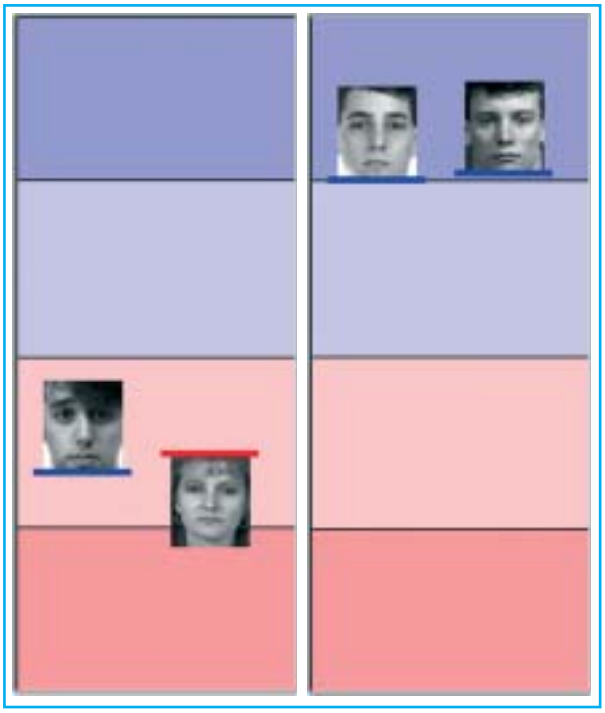


Abb. 13: Erfolg versprechende Trainingsbeispiele

die GenderRecDB, die eigens für diese Arbeit mit Unterstützung Wedeler Studenten erstellt wurde, und die vom Massachusetts Institute of Technology verwaltete FERET-Database eingesetzt.

Die geschätzten (also beim Lernen ermittelten) Fehler bewegen sich zwischen 10.5% Fehlerquote (also 89.5% Klassifizierungsrate als bester Wert, erzielt mit 200 Trainingsbildern aus der FERET-Database) und 20% Fehlerquote (also 80% Klassifizierungsrate als schlechtester Wert, erzielt mit 100 Bildern aus der GenderRecDB).

Die tatsächlichen (also beim Klassifizieren ermittelten) Fehler waren im besten Fall 5.8% Fehlerquote (94.2% richtige Klassifizierung). Dabei wurde die SVM mit 200 Bildern aus der FERET-Database angelernt und mit ungelernten Bildern aus dieser getestet. Beim Testen der Bilder einer Datenbank mit einer SVM, die mit Bildern der anderen Datenbank angelernt wurde, waren allerdings Fehlerquoten von bis zu 40% zu verzeichnen. Dieses Ergebnis zeigt deutlich, wie sehr die Güte eines derartigen Verfahrens von der Gleichartigkeit der Eingabedaten abhängt. Eine SVM, die mit einer gemischten Datenbank angelernt wurde, hatte eine Fehlerquote von 6-11% (89-94% Klassifizierungsrate) auf ungesehenen Bildern aus beiden Datenbanken und somit eine deutlich bessere Generalisierung.

Mit Klassifizierungsraten von über 90% steht das hier verwendete Verfahren den Werten kaum nach, die derzeit in Artikeln zur Gender Recognition mit SVMs publiziert werden. Bei dem Vergleich ist zudem zu beachten, dass die hier verwendeten Trainingsdatenmengen für eine Support Vector Machine recht gering sind. Eine adäquate, sehr viel größere Menge an Trainingsdaten ließe sich allerdings nicht mehr visualisieren.

Ausblick

Die beschriebene Arbeit bietet einen guten Ausgangspunkt für weitere Abschlussarbeiten dieser oder ähnlicher Thematik. So soll beispielsweise die entwickelte Software im Master-Studiengang Informatik der FH Wedel in der Veranstaltung "Learning and Softcomputing" eingesetzt werden. Diese Veranstaltung, die zwecks Verfestigung des Stoffes einen beträchtlichen praktischen Anteil hat, befasst sich mit den verschiedenen Verfahren des maschinellen Lernens.

Acknowledgement

Portions of the research presented in this paper used the FERET-Database of facial images collected under the FERET-program, sponsored by the DOD Counterdrug Technology Development Program Office. We thank the Massachusetts Institute of Technology, Cambridge, USA, for kindly placing the database at our disposal.

Literatur

Bücher

Vojislav Kecman: Learning and Soft Computing; MIT Press, Cambridge, USA 2001; ISBN: 0-262-11255-8
Das Werk ist eines der Referenzbücher zu selbstlernenden Algorithmen, welches einen sehr guten Überblick über die verschiedenen Algorithmen und deren mathematische Grundlagen gibt.

Nello Cristianini and John Shawe-Taylor: An Introduction to Support Vector Machines and other kernel-based learning methods; Cambridge University Press, United Kingdom, 2000;
ISBN: 0-521-78019-5
Mathematische Grundlagen speziell zu Kernel-basierten Verfahren.

Links

http://www.learning-with-kernels.org/
Enthält hauptsächlich Auszüge aus dem Buch "Learning with Kernels" (Bernhard Schölkopf and Alexander J. Smola; MIT Press, Cambridge, USA, 2002; ISBN: 0-262-19475-4).

http://www.kernel-machines.org/
Sammlung mit Workshops und Foren zum Thema maschinelles Lernen.

http://www.svms.org/
Erklärungen und Links zu den Themen, die im Zusammenhang mit Support Vector Machines stehen, und eine lange Liste von Applikationen, die mit SVMs umgesetzt wurden.

http://www.support-vector.ws/
Webseite von Vojislav Kecman, auf der sich unter anderem auch ein Vortrag findet, den er zu Support Vector Machines an der University of Auckland (New Zealand) gehalten hat.

Helga Karafiat; FH Wedel, 3. Sem. Masterstudium Informatik