Praktikumsbericht

Verfahren zur Merkmalsauswahl bei Mehrklassifikatorsystemen

Frank Förster Cornelia Beck Matthias Raab

14. August 2003

Betruer:

Christian Dietrich, dietrich@neuro.informatik.uni-ulm.de
Dr. Friedhelm Schwenker, schwenker@neuro.informatik.uni-ulm.de

Inhaltsverzeichnis

1 Einführung

Grillengesänge können mit Hilfe von Klassifikationssystemen zur Bestandsaufnahme und Überwachung sowohl von einzelnen Arten als auch von ganzen Gemeinschaften verwendet werden. Dabei werden anhand akustischer Parameter wie z.B. der Trägerfrequenz und Pulsraten erkennbare taxonomische Einheiten definiert, deren Ausprägung spezifisch für die einzelnen Arten ist. Viele Grillenarten stellen sensible Umweltindikatoren für die Qualität des Lebensraums in gemäßigten wie auch in tropischen Breiten dar. Deshalb kann durch die Klassifikation der an einem Ort ansässigen Arten eine Beurteilung des momentanen ökologischen Zustands erfolgen. Ebenso können hierdurch Maßnahmen zur Verbesserung der Umweltsituation verifiziert und evaluiert werden [5].

Wünscheswert ist für diese Aufgabe natürlich ein System, das aus den Audioaufnahmen automatisch die Gattung der entsprechenden Grille ermittelt. Mit einem solchen Klassifikationssystem von Grillengesängen beschäftigt sich Christian Dietrich in seiner Disseratation. Eine nicht unwesentliche Rolle spielt hierbei die Merkmalsselektion, mit der wir uns im Rahmen unseres Praktikums beschäftigt haben.

Das Design von Systemen zur Mustererkennung kann allgemein in zwei Phasen unterteilt werden:

- Die Merkmalsselektion (*Feature Selection*)
- Die Klassifikation der ausgewählten Merkmalsvektoren

Das Hauptziel der Merkmalsselektion ist dabei die Auswahl einer Teilmenge von d Merkmalen aus einer gegebenen Menge von D Messungen, wobei $d < D$ gilt. Außerdem sollten die gewählten Merkmale sinnvollerweise so bestimmt werden, dass sie zu einer möglichst fehlerfreien Klassifikation führen.

Dazu wird eine Bewertungsfunktion benötigt, welche die Effektivität einer gewählten Merkmalsmenge beurteilt. Eine solche Funktion vorausgesetzt ist eine Merkmalsselektion nichts anderes als ein Suchproblem mit dem Ziel, eine möglichst optimale Teilmenge an Merkmalen zu finden. Dies kann allerdings bei hochdimensionalen Daten ein sehr komplexes Problem sein, da die Optimierung in einem kombinatorischen Suchraum stattfindet und dieser noch zusätzlichen Einschränkungen unterliegt. Eine begrenzte Anzahl an Trainingsdatensätzen wäre z.B. eine solche Einschränkung.

Zum Auffinden der optimalen Lösung wäre eine erschöpfende Suche notwendig. Eine solche würde jedoch alle $2^n - 1$ möglichen Teilmengen an Merkmalen testen und ist auf Grund der Tatsache, dass die Merkmalsmengen

häufig relativ groß sind, nicht praktikabel. Deshalb ist es notwendig, Heuristiken zu entwickeln, die akzeptable Ergebnisse liefern, ohne den gesamten Suchraum zu durchsuchen. Man arbeitet also hauptsächlich mit suboptimalen Suchverfahren. Ein bekanntes effektives Suchverfahren ist das "plus-l-take-away-r"-Verfahren, welches auch als (l,r)-Suchverfahren bezeichnet wird.

Heuristische Basis der meisten sequentiellen Suchverfahren ist die Annahme, dass die Bewertungsfunktion monoton ist. Wird unter dieser Voraussetzung zur aktuellen Merkmalsmenge ein Merkmal hinzugefügt, so darf sich der Wert der Bewertungsfunktion nicht verschlechtern.

Aus diesem Grund werden bei der Merkmalsselektion mehrheitlich Klassentrennbarkeitsmaße (*class separability measures*) benutzt, welche gerade die Eigenschaft der Monotonie unter Hinzunahme von Elementen zu einer gegebenen Menge erfüllen. Allerdings bringen stochastische Trennbarkeitsmessungen über eine zufällige Menge von Merkmalen eine andere Ordnung der Merkmale hervor, als dies beim Vergleichen der Klassifikationsfehler entstehen würde. Grund für diese Differenz ist die Abhängigkeit der Fehlklassifikationsraten von den folgenden zwei Aspekten:

- stochastische Klassentrennbarkeit
- von der Art des Klassifikators abhängige strukturelle Fehler

Da die auf den stochastischen Trennbarkeitsmessungen beruhenden Methoden zur Merkmalsselektion den letzten Aspekt nicht berücksichtigen, ist die Klassifikationsleistung der so gewählten Merkmale u.U. niedrig. Aus diesem Grund geht der vielversprechendste Weg zur Bewertung und Auswahl von Merkmalen über die Fehlerrate des zu konstruierenden Klassifikators (bzw. der zu konstruierenden Klassifikatoren, falls es mehrere sind). Abgesehen von den Seiteneffekten kleiner Testmengen führt der erste für die Klassifkationsfehlleistungen verantwortliche Aspekt wie oben angedeutet zu einer monotonen Bewertungsfunktion. Der zweite Aspekt allerdings, also strukturelle Fehler, können diese Monotonieeigenschaft zerstören und damit die Heuristik hinter den sequentiellen Suchverfahren zunichte machen.

Zur Überwindung dieses Problems wurde versucht, näherungsweise erreichte Monotonie mit anderen Methoden wie Genetischen Algorithmen und Monte-Carlo-Ansätzen zu kombinieren. Das Ziel war dabei die Entwicklung von Algorithmen, die mit der erwähnten Nicht-Monotonie zurechtkommen. Das funktioniert für Probleme der Merkmalsselektion moderater Größe, bei Problemen größeren Ausmaßes versagen sie allerdings.

Im folgenden wird deshalb in Anlehung an die Floating-Search-Methoden von Pudil wieder auf die sequentiellen Suchverfahren gesetzt. Pudil entwickelte eine Familie von Suchverfahren, die er "floating search methods" nennt.

Diese sind zum einen bei Problemen sehr hoher Dimensionalität sehr effizient und kommen zum anderen mit nicht-monotonen Bewertungsfunktionen zurecht. Die Floating-Search-Methoden sind dabei mit dem "plus-l-take-away-r"- Algorithmus verwandt, bei dem die Anzahl der Vorwärts- und Rückwärtsschritte allerdings festgelegt ist [1]. Im Gegensatz dazu erfolgt die Hinzu- bzw. Wegnahme von Merkmalen bei den Floating-Search-Methoden dynamisch in Abhängigkeit vom Erfolg eines solchen Schritts. Genau genommen wird, abhängig vom jeweiligen Verfahren, die Erweiterung in eine Richtung erzwungen und die andere Richtung ist optional.

Ein relativ einfacher Ansatz ohne Floating-Search, *Forward Search*, besteht darin, jeweils das Merkmal in die Menge der bereits selektierten Merkmale aufzunehmen, mit dem die Klassifikation das beste Ergebnis erzielt. Dies führt jedoch häufig zum sogenannten "Nesting"-Effekt. Dies bedeutet, dass ein einmal erreichtes suboptimales Plateau im Suchraum nicht mehr verlassen wird. Grund dafür ist das Beibehalten jener Merkmale, die der Merkmalsmenge in vorangegangenen Schritten hinzugefügt wurden. Weitestgehend verhindert wird dieser Effekt beim schon erwähnten "plus-l-take-away-r"-Algorithmus, indem systematisch Rückwärtsschritte vorgenommen werden, was in diesem Zusammenhang das Entfernen von Merkmalen aus der für die Klassifikation vorgesehene Merkmalsmenge bedeutet. Dadurch können lokale Optima, aus der die normale Vorwärtssuche nicht mehr herauskommen würde, wieder verlassen werden.

Wir haben in unserem Fall zunächst *Sequential Floating Forward Search (SFFS)* als Nachfolger des *plus-l-take-away-r*-Algorithmus bzw. als Erweiterung der Vorwärtssuche verwendet, da wir es mit einem relativ hochdimensionalen Problem zu tun haben. Unsere Aufgabe bestand darin, den existierenden Algorithmus (Vorwärtssuche) um einen eventuellen Rückwärtsschritt zu erweitern (*SFFS*). Dies führt ähnlich wie beim *plus-l-take-away-r*-Algorithmus zu einem erheblich reduzierten Risiko, was das Auftreten des Nesting-Effekts anbelangt. Die Kernidee einer von uns entwickelten Erweiterung des Algorithmus beruht auf der Selektion von Merkmalskomponenten für Mehrklassifikatorsysteme, was eine weitere Verbesserung der Ergebnisse nach sich zieht.

Die Algorithmen wurden in Form von *Bash*-Skripten und *C++* Programmen unter *Linux* implementiert und mit reellen Daten evaluiert.

[1] Die Anzahl der Vorwärts- und Rückwärtsschritte wird durch die im Namen auftretenden Parameter l und r bestimmt.

2 Klassifikatoren

2.1 Definition

Randbedingung: überwachtes Lernproblem.
geg.: Trainingsmenge der Form $\{(x_1, y_1),...,(x_m, y_m)\}$ für eine unbekannte Funktion $y = f(x)$. Ein Klassifikator ist dann die von einem Lernalgorithmus ausgegebene Hypothese über die wahre Funktion f. Übergibt man ihm einen neuen, also nicht in der Trainingsmenge enthaltenen, Vektor x, so macht er eine Voraussage über die y-Werte, also die Zugehörigkeit des Eingabevektors zu den verschiedenen Klassen [2].

2.2 Fuzzy-k-Nearest-Neighbour Klassifikator

In unserem Fall wurde ein sehr einfacher und auch effizienter Ansatz zur Klassifikation verwendet, nämlich die k-Nearest-Neighbour Methode. Hier werden um einen Merkmalvektor $x \in \mathbb{R}^d$ zu klassifizieren seine k nächsten Nachbarn zu einer Menge aus m Prototypen bestimmt. Als Distanz wird die L_p-Norm ($p \in [1, \infty)$) verwendet:

$$d_j^p(x, x^j) = \|x - x^j\|_p = (\sum_{i=1}^{d} |x_i - x_i^j|^p)^{\frac{1}{p}}$$

Für $p = 1$ ist dies die Manhattan-, für $p = 2$ die Euklidische Distanz. Die k nächsten Nachbarn ($k < m$) von x sind definiert als

$$\mathcal{N}_k(x) =_{def} \{x^{\tau_1}, \ldots, x^{\tau_k}\}$$

wobei $(\tau_i)_{i=1}^m \subset \{1, \ldots, m\}$ eine Folge von Indices mit der Eigenschaft $d_{\tau_1}^p \leq \cdots \leq d_{\tau_k}^p$ ist. Sind nun $c_i = c(x_{\tau_i}), i = 1, \ldots, k$ die Klassen der k nächsten Nachbarn, so ist eine Teilmenge der nächsten Nachbarn, nämlich die deren Klasse j ist, folgendermaßen definiert:

$$\mathcal{N}_k^j(x) = \{y \in \mathcal{N}_k(x) | c(y) = j\}$$

Die Klassifikation eines Eingabevektors x wird nun über die Mächtigkeit dieser Mengen bestimmt. Die Klasse $j^* \in \{1, \ldots, l\}$ ist die, in der am meisten der k nächsten Nachbarn liegen, wobei l die Gesamtzahl der Klassen ist.

$$j^* = \underset{j=1,..,l}{\operatorname{argmax}} |\mathcal{N}_k^j(x)|$$

Um eine Klassenzugehörigkeit für alle l Klassen zu ermitteln, wird der sogenannte Fuzzy-k-Nearest-Neighbour Klassifikator verwendet. Hier wird

Klassenzugehörigkeit durch eine Abbildung $\Delta : \mathbb{R}^d \mapsto [0,1]^l$ dargestellt; für einen Eingabevektor $x \in \mathbb{R}^d$ bezeichnet $\Delta(x) = (\Delta_1(x), \ldots, \Delta_l(x))$ seine Zugehörigkeit zu den l einzelnen Klassen. Sei

$$\delta_j(x) = \frac{1}{\sum_{x^i \in \mathcal{N}_k^j(x)} \|x - x^i\|_p + \alpha}$$

der Wert für die Unterstüzung der Hypothese, dass j^* das richtige Klassenlabel für x ist. Für $\mathcal{N}_k^j(x) = \emptyset$ wird $\delta_j(x) = 0$ gesetzt. Der Parameter $\alpha > 0$ wird zur Begrenzung von $\delta_j(x)$ bei sehr kleinen Werten von $\|x - x^i\|_p$ verwendet.

Durch die Normalisierung

$$\Delta_j(x) = \frac{\delta_j(x)}{\sum_{i=1}^{l} \delta_i(x)}$$

erhalten wir nun für einen Eingabevektor x Klassifikatorausgaben $\Delta_j(x)$ mit den Eigenschaften $\Delta_j(x) \in [0,1]$ und $\sum_{j=1}^{l} \Delta_j(x) = 1$. Diese Ausgaben werden auch *soft labels* genannt.

3 SFFS

Sequential Floating Forward Search (SFFS) ist eine Veränderung des *plus-l-take-away-r*-Algorithmus, welcher jedoch auch von *Forward Search* abstammt. Bei beiden Algorithmen besteht der Ansatz darin, nach erfolgreicher Hinzunahme von Merkmalen eventuell wieder solche zu entfernen.

Dahinter steckt die Idee, die Vorwärtssuche derart zu modifizieren, dass auch Merkmalskombinationen, die im dortigen Verfahren nicht erreicht werden, auftreten können. Dies ist vorteilhaft, um das sogenannte 'Nesting' zu vermeiden. Dabei verschlechtern Merkmale das Klassifikationsergebnis, obwohl sie zunächst die optimalen Merkmale in einem früheren Erweiterungsschritt waren. Doch bei den später folgenden Erweiterungen drücken diese das Ergebnis nach unten, erweisen sich also in der Kombination nicht als eine geeignete Auswahl. Dies führt bei gewöhnlichen Verfahren zur Vorwärtssuche dazu, dass der Klassifikationsfehler auf einem suboptimalem Niveau stagniert. Eine einmal getroffene ungeeignete Merkmalshinzunahme kann in einem solchen Verfahren nicht mehr rückgängig gemacht werden.

Bei *Sequential Floating Forward Search (SFFS)* und dem *plus-l-take-away-r*-Algorithmus dagegen soll dies verhindert werden. Dementsprechend muss es möglich sein, aus der Merkmalsmenge störende Merkmale wieder herauszunehmen. Dafür benötigt man Rückwärtsschritte, die dafür verantwor-

lich sind, dass Merkmale entfernt werden, welche die Ergebnisse verschlechtern. *SFFS* kann folgendermaßen beschrieben werden: Nach jeder Merkmalshinzunahme wird auf mögliche Rückwärtsschritte getestet. Dabei wird das Resultat des ersten Rückwärtsschritts mit dem Resultat der gerade bestimmten erweiterten Merkmalsmenge verglichen. Wenn das Ergebnis beim Rückwärtsschritt besser ist, wird dieser durchgeführt und dementsprechend ein Merkmal aus der Merkmalsmenge entfernt. In diesem Fall wird erneut ein Rückwärtsschritt versucht. Diesmal müssen die Ergebnisse mit dem des soeben durchgeführten Rückwärtsschrittes verglichen werden. Solange sich die Ergebnisse verbessern lassen, werden immer weiter Merkmale entfernt, ansonsten wird wieder versucht, ein neues Merkmal hinzuzunehmen. Keinen Rückwärtsschritt gibt es genau dann, wenn die Merkmalsmenge nach der Hinzunahme des Merkmals bereits optimal ist, d.h. wenn jegliche Entfernung eines Merkmals zu einem schlechteren Ergebnis führen würde.

Aufgrund der nicht statisch festgelegten Anzahl von Rückwärtsschritten zählt dieser Algorithmus zu den *Floating Search*-Verfahren. Beim *plus-l-take-away-r*-Algorithmus dagegen ist die Anzahl der Rückwärtsschritte stets fest. Dies bedeutet, dass nach einer Hinzunahme von l Merkmalen $r < l$ Merkmale wieder aus der Merkmalsmenge entfernt werden. Die Anzahl der jeweiligen Schritte ist also von Anfang an festgelegt. Dies ist problematisch, da für r und l nicht ohne weiteres ein optimaler Wert gefunden werden kann. Auch müssen bei *SFFS* keine Parameter am Anfang initialisiert werden. Der Algorithmus erledigt die komplette Feature Selection automatisch. Durch die dynamische Kontrolle zur Laufzeit kann zusätzlich die Leistung verbessert werden. Bei Experimenten hat sich herausgestellt, dass die Ergebnisse bei *SFFS* meist besser sind als die des *plus-l-take-away-r*-Algorithmus [4].

Bei der Implemtierung von *SFFS* müssen mehrere Dinge berücksichtigt werden: Die größte Gefahr bei diesem Algorithmus besteht darin, sich in Endlosschleifen zu verlieren. Dies kann insbesondere dann passieren, wenn das Merkmal wieder entfernt wird, dass soeben erst hinzugefügt worden ist. Allgemein dürfen keine Zustände der Selektion erreicht werden, in denen man sich zuvor schon befunden hat. Vermeiden läßt sich dies, indem man jede erreichte Merkmalsmenge speichert und bei jedem weiteren Bilden einer Merkmalsmenge vergleicht, ob eine solche schon erstellt worden ist. Wenn dies der Fall ist, wird nach einer anderen Merkmalsmenge gesucht, die noch nicht vorhanden ist.

3.1 Algorithmus

Seien im folgenden

- F - Menge, die alle Merkmale enthält
- F^s - Menge der Merkmale, die bereits ausgewählt wurden
- $n_{\max}$ - maximale Anzahl der Merkmale, die für die Klassifikation verwendet werden dürfen
- T - Testfunktion, die für gegebene Merkmalsmenge eine Fehlklassifikationsrate ermittelt [2]
- *States* - Menge aller in den vorherigen Schritten selektierten Merkmalsmengen

Der Algorithmus von *SFFS* kann nun folgendermaßen formuliert werden.

Algorithmus SFFS

Setze: $F^s = \emptyset;\ States = \emptyset;\ n = 0$
WHILE $n < n_{\max}$

Wahl des Merkmals:
$f = \underset{m \in (F \setminus F^s)}{\text{argmin}} (\mathsf{T}(F^s \cup \{m\}))$
$F^s = F^s \cup \{f\}$
$States = States \cup F^s$
$n = n + 1$

Rückwärtsschritte:
WHILE (*true*)
$f = \underset{m \in F^s}{\text{argmin}}(\mathsf{T}(F^s \setminus \{m\}))$
IF $\mathsf{T}(F^s \setminus \{f\}) < \mathsf{T}(F^s)$ und
$(F^s \setminus \{f\}) \notin States$
$F^s = F^s \setminus \{f\}$
$States = States \cup \{F^s\}$
$n = n - 1$
ELSE break
END
$n = n + 1$
END

[2] In unserem Fall wird zur Ermittlung der Fehlklassifikationrate eine Kreuzvalidierung durchgeführt.

Anmerkungen:

- Die Menge "States" wird in diesem Algorithmus benötigt um Endlosschleifen zu verhindern. Nachdem das Merkmal gefunden wurde, dessen Entfernen die größte Verbesserung für die Fehlklassifikationsquote bedeutet, muss stets verglichen werden, ob die dadurch entstehende Menge nicht bereits in "States" enthalten ist. In einem solchen Fall würden wir das Merkmal nicht entfernen, da sonst eine Endlosschleife entstehen würde.

- Der Aufwand des Rückwärtschrittes steigt mit Anzahl der selektierten Merkmale, während der Aufwand des Forwärtschrittes sinkt. Im Vergleich zu *Forward Search* verdoppelt sich bei *SFFS* der Rechenaufwand, wenn man den Algorithmus zu Ende laufen lässt (alle Merkmale selektiert) und der Rückwärtsschritt nicht eintritt. Sobald einer oder mehrere Rückwärtsschritte eingetreten sind, ist es jedoch nicht mehr möglich eine Aussage über die Komplexität zu treffen. Beschränkt wird sie trivialerweise durch die Größe des kombinatorischen Suchraums.

3.2 Ergebnisse

Für konkrete Versuchsergebnisse haben wir den beschriebenen Algorithmus auf einen Beispieldatensatz mit Grillengesängen angewandt. Eine nähere Beschreibung zu Extraktion der Merkmale aus den Originaldaten findet sich in [1]. Unsere Tests beziehen sich auf einen 56-dimensionalen Datensatz. Die Merkmale sind folgendermaßen auf die Dimensionen aufgeteilt:

- T - Zeitliche Struktur der Impulse (4-dim.)
- D - Impulslänge (4-dim.)
- F - Impulsfrequenz (4-dim.)
- E - Energiekontur der Impulse (20-dim.)
- E^{--} - Energiekontur der Impulse, zeitlich normalisiert (20-dim.)
- S - Maximale Amplitude innerhalb eines Impulses (4-dim.)

Zur Einschätzung der Güte der verwendeten Merkmale eignet sich Abbildung 1. Hier sind die Fehlklassifikationsraten aufgetragen, welche sich ergeben, wenn nur ein einziges Merkmal zu Klassifikation verwendet wird.

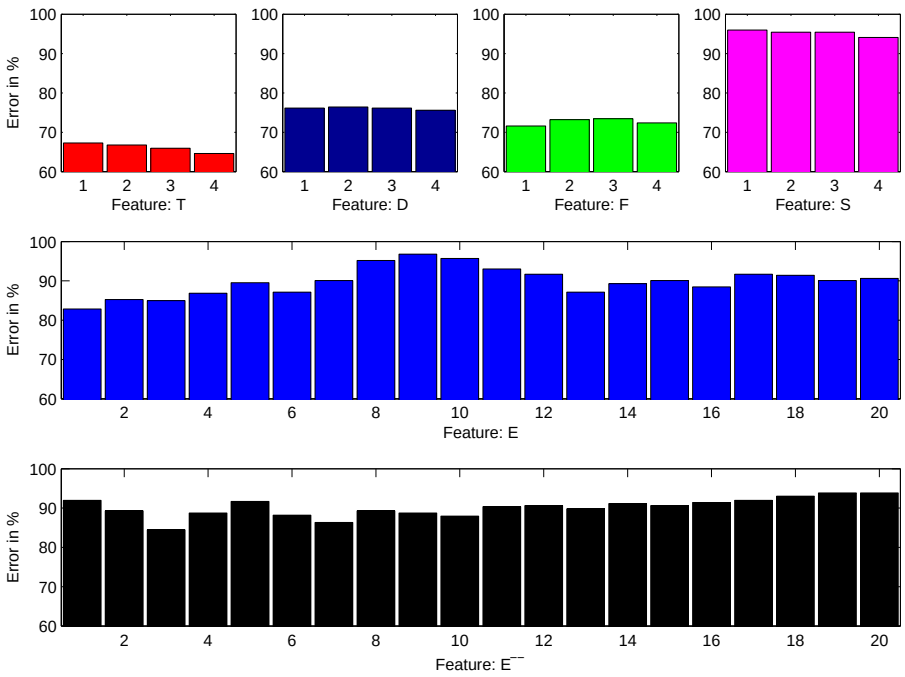

Abbildung 1: Fehlklassifikationsraten bei einer Klassifikation mit einzelnen Merkmalen

Beim Betrachten der Ergebnisse (siehe Abbildung 2) fällt auf, dass die Rückwärtsschritte in unserem Fall erst bei den letzten selektierten Merkmalen auftreten. In den ersten 50 Selektionsschritten wird kein einziges Mal die Möglichkeit genutzt, ein Merkmal zu entfernen, demnach unterscheiden sich die erreichten Fehlerquoten kaum von denen der Vorwärtssuche. Dies entspricht nicht unseren Erwartungen, da wir uns durch die Verwendung einer Floating-Search-Methode eine Verbesserung des Ergebnisses erhofft haben. Durch die Nichtmonotonie im Suchverhalten findet in jedem Schritt eine Erweiterung der Suche auf in der Vorwärtssuche nicht betrachtete Merkmalsmengen statt. In den ersten 10-15 Selektionsschritten stehen erfahrungsgemäß noch gute Merkmale zur Auswahl, die Funktion ist in diesem Bereich mit hoher Wahrscheinlichkeit monoton. Sobald man nur noch aus für die Klassifikation weniger geeigneten Merkmalen die Wahl hat, hätten wir uns eine Verbesserung erhofft. Desweiteren traten Rückwärtsschritte nur in Testläufen mit einer ausreichend großen Anzahl an Kreuzvalidierungen ein. Dies ist darauf zurückzuführen, dass der Algorithmus nur dann einen Rückwärtsschritt ausführt, wenn die Fehlerquote dabei kleiner wird, bei Gleich-

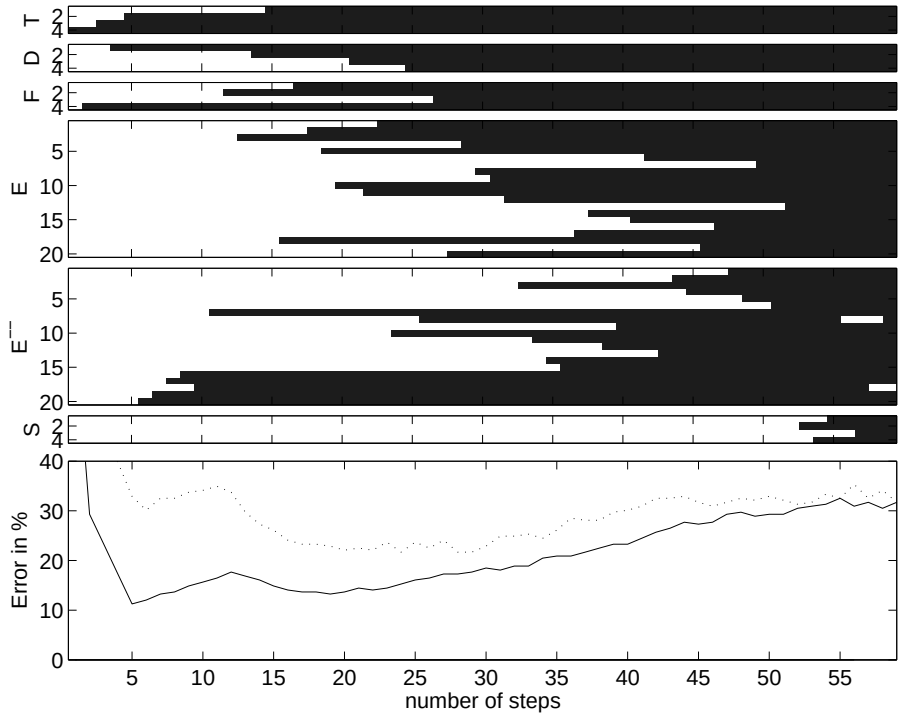

Abbildung 2: Kompletter SFFS Durchlauf auf 56 Merkmalen

heit wird nur hinzugefügt. Bei einer geringeren Kreuzvalidierungsrate sind, nachdem bereits viele Merkmale selektiert worden sind, bei etlichen der Merkmalsmengen die Fehlerquoten identisch. Demnach ist eine höhere Kreuzvalidierungsrate nötig, um eine genauere Fehlerquote zu erreichen und dadurch differenzieren zu können. Die Ergebnisse lassen sich eventuell verbessern, dass man den Rückwärtsschritt dadurch wahrscheinlicher macht, dass er bereits bei gleichen Fehlerquoten durchgeführt wird.

4 Mehrklassifikatorsysteme

Damit ein Mehrklassfikatorsystem bessere Vorhersagen über die Klassenzugehörigkeit eines Eingabevektors machen kann als seine einzelnen Komponenten müssen zwei Forderungen an die einzelnen Klassifikatoren gestellt werden. Zum einen müssen die Klassifikatoren richtig (accurate) sein. Ein richtiger Klassifikator ist ein Klassifikator, dessen Fehlerrate besser ist als diejenige, die bei zufälligem Raten der Klassenzugehörigkeit erreicht würde. Das zweite Attribut, das die einzelnen Komponenten erfüllen sollten, ist die Unkorreliertheit. Unkorreliert sind zwei Klassifikatoren dann, wenn sie bei der Eingabe von neuen Daten unterschiedliche Fehler machen.

Wären die einzelnen Klassifikatoren nicht unkorreliert, so würden sie bei der gleichen Eingabe die gleichen oder zumindest ähnliche Fehler machen. Die Fehlerrate des Systems wäre dann nicht geringer als die seiner einzelnen Komponenten. Sind die Klassifikatoren allerdings unkorreliert, so besteht die Möglichkeit, dass der Fehler eines Klassifikators durch die Korrektheit eines anderen ausgeglichen wird. Die Mehrheitsentscheidung kann dann zum richtigen Ergebnis führen. Präziser: Wenn die Fehlerraten von L Hypothesen kleiner sind als $0,5$ und die Fehler unkorreliert sind, so ist die Wahrscheinlichkeit, dass die Mehrheitsentscheidung falsch ist, im Bereich unterhalb der Binomialverteilung, in der mehr als $L/2$ Hypothesen falsch sind [2]. Genau genommen kommt zu den genannten zwei Anforderungen an die Klassifikatoren also noch jene hinzu, dass die individuellen Fehlerraten kleiner als $0,5$ sein sollten. Ist dies nicht gegeben, so kann die Mehrheitsentscheidung sogar schlechter sein die jeweiligen Einzelentscheidungen.

Dass ein Mehrklassifkatorsystem auch praktisch besser ist als ein einzelner Klassifikator - und nicht nur theoretisch, wie oben erläutert - wird durch die folgenden drei Argumente deutlich.

- **Statistische Gründe**: Der Suchraum, den der Lernalgorithmus auf der Suche nach dem besten Klassifikator durchsucht, kann als Hypothesenraum betrachtet werden. Ein statistisches Problem ergibt sich

dann, wenn die Menge der Trainingsdaten zu klein ist im Vergleich zur Größe des Hypothesenraums. Der Lernalgorithmus findet in diesem Fall u.U. viele verschiedene Hypothesen mit der gleichen Genauigkeit. Indem aus diesen ein Mehrklassifikatorsystem erstellt wird, können die einzelnen Vorhersagen gemittelt werden. Dies reduziert das Risiko, den falschen Klassfikator auszuwählen.

- **Gründe der Komplexität**: Viele Lernalgorithmen basieren auf lokaler Suche. Dadurch sind sie natürlich gefährdet in einem lokalen Optimum zu stoppen. Auf der anderen Seite ist das optimale Training zum Auffinden der globalen Optimas oft NP-vollständig (z.B. für neuronale Netze, Entscheidungsbäume). Damit ist dies auch kein möglicher Weg. Dieses Problem kann auch trotz ausreichender Trainingsdaten, und damit unter Wegfall der obigen statistischen Gründe, eintreten. Gewinnt man nun über lokale Suchen von unterschiedlichen Punkten des Suchraums aus verschiedene lokale Optimas, so kann ein Mehrklassifikatorsystem eine bessere Annäherung an das globale Optimum darstellen als jedes einzelne lokale Optimum/jeder einzelne Klassifikator.

- **Gründe der Repräsentation**: In manchen Fällen ist es unmöglich die tatsächliche Abbildung f (die fehlerfreie Klassifikation) durch eine einzelne Hypothese darzustellen, d.h. f liegt außerhalb des Hypothesenraums. In diesen Fällen kann der Suchraum u.U. durch das gewichtete Aufsummieren mehrerer Hypothesen erweitert werden, so dass f durch das so geschaffene Mehrklassifikatorsystem approximiert wird.

5 Vorwärtssuche über mehrere Klassifikatoren

Aufgrund der Gründe, die für die Güte einer Klassifikation mit mehreren Klassifikatoren sprechen, haben wir das Verfahren der Vorwärtssuche auf mehrere Klassifikatoren erweitert. Man beschränkt sich also nicht nur auf einen Klassifikator, sondern hat die Möglichkeit, die Merkmale auf mehrere Klassifikatoren zu verteilen. Dahinter steckt zunächst die Idee, dass bei jeder Erweiterung der Anzahl der Merkmale, das neue Merkmal zunächst bei den bereits vorhandenen Klassifikatoren angehängt und dort die Fehlerquote bestimmt wird. Anschließend testet man den Fall, dass das neue Merkmal einzeln als Eingabe für den neuen Klassifikator dient, die alten Merkmalssequenzen der restlichen Klassifikatoren bleiben erhalten. Um die Gesamtfehlerquote für eine bestimmte Merkmals-Klassifikatoren-Kombination

zu bekommen, müssen die Fehlerquoten der einzelnen Klassifikatoren fusioniert werden. Dabei wird bei den einzelnen Klassifikatoren in unserem Fall stets ein *fuzzy-k-nearest-neighbour*-Klassifikator verwendet. Diese Methode der Vorwärtssuche über mehrere Klassifikatoren werden wir im folgen auch mit *SFS-M (Sequential Forward Selection for Multiple Classifier Systems)* bezeichnen.

Allerdings ist in der Praxis normalerweise nicht zu erwarten, dass die Erweiterung um einen Klassifikator wirklich eintritt. Ein Klassifikator mit nur einem Merkmal führt bei realen Daten normalerweise zu einem schlechteren Klassifikationsergebnis wie ein Klassifikator mit mehreren Merkmalen. Somit dürften mit hoher Wahrscheinlichkeit die Erweiterungen der bisher verwendeten Klassifikatoren in der Fusion einen geringeren Fehler erzeugen, verglichen mit dem, der bei einer Fusion mit dem neuem Klassifaktor entsteht.

Um dennoch auf eine ausreichende Zahl von Klassifikatoren zu kommen, müssen in der Initialisierungphase entsprechende Änderungen vorgenommen werden. Eine intuitive Methode ist hierbei das Füllen einer vorgegebenen Anzahl an Klassifikatoren mit Merkmalen, von denen bekannt ist, dass sie stochastisch relativ unabhängig sind.

Da dies in der Realität schwer zu ermitteln ist, haben wir zur Vollautomatisierung unseres Algorithmus eine andere Initialisierungsphase gewählt. Hier wird am Anfang der Algorithmus nur derart modifizierter, dass die Erweiterung um einen Klassifikator die einzige Methode zur Wahl eines neuen Merkmals ist. Die ersten k selektierten Merkmale werden jeweils einem der k Anfangsklassifikatoren zugewiesen.

5.1 Algorithmus

Seien im folgenden

- F - Menge, die alle Merkmale enthält
- F_k^s - Menge der Merkmale, die bereits für den k-ten Klassifikator ausgewählt wurden
- $n_{\max}$ - maximale Anzahl der Merkmale, die für die Klassifikation verwendet werden dürfen
- $k_{\max}$ - maximale Anzahl an Klassifikatoren
- error_i - geringste Fehlklassifikationsrate, wenn ein Merkmal zu Klassifikator i hinzugefügt wird
- f_i - zu error_i gehöriges Merkmal

- T - Testfunktion, die für gegebene Merkmalsmenge eine Fehlklassifikationsquote ermittelt
- fusion - fusioniert die Fehlklassifikationsraten von mehreren Klassifikatoren zu einer Gesamtfehlerquote

Algorithmus SFS-M

Setze: $F_k^s = \emptyset,\ k = 1 \ldots k_{\max};\ n = 0;\ k = 0$
WHILE $n < n_{\max}$

Wahl der Merkmale für einzelne Klassifikatoren:
FOR $i = 1 \ldots k$
$f_i = \underset{m \in (F \setminus F^s)}{\text{argmin}}\ (\text{fusion}(\mathsf{T}((F_i^s \cup \{m\}), \{F_j^s \mid j = 1 \ldots k; j \neq i\})))$
$\text{error}_i = (\text{fusion}(\mathsf{T}((F_i^s \cup \{f_i\}), \{F_j^s \mid j = 1 \ldots k; j \neq i\})))$
END
Erweiterung um einen Klassifikator:
IF $k < k_{\max}$
$k = k + 1$
$f_k = \underset{m \in (F \setminus F^s)}{\text{argmin}}\ (\text{fusion}(\mathsf{T}((F_k^s \cup \{m\}), \{F_j^s \mid j = 1 \ldots k-1)))$
$\text{error}_k = \text{fusion}(\mathsf{T}((F_k^s \cup \{f_k\}), \{F_j^s \mid j = 1 \ldots k-1)))$
END

Aufnahme des Merkmals zum zugehörigen Klassifikator:
$\tilde{i} = \underset{i=1 \ldots k}{\text{argmin}}(\text{error}_i)$
$F_{\tilde{i}}^s = F^s \cup \{f_{\tilde{i}}\}$
IF $k \neq \tilde{i}$
$k = k - 1$
END
$n = n + 1$
END

Anmerkung: Der Algorithmus kann in seiner Komplexität erheblich reduziert werden, wenn man Zwischenergebnisse abspeichert. Bei der Ermittlung der Fehlklassifikationsrate werden nur in einer Eingabekomponente der

Fusion Änderungen vorgenommen, folglich müssen die Ergebnisse der nicht veränderten Klassifikatoren nicht neu ermittelt werden. Auf diese Weise steigt der Rechenaufwand im Vergleich zur Vorwärtsuche über einen Klassifikator nur linear mit der Anzahl der Klassifikatoren.

5.2 Ergebnisse

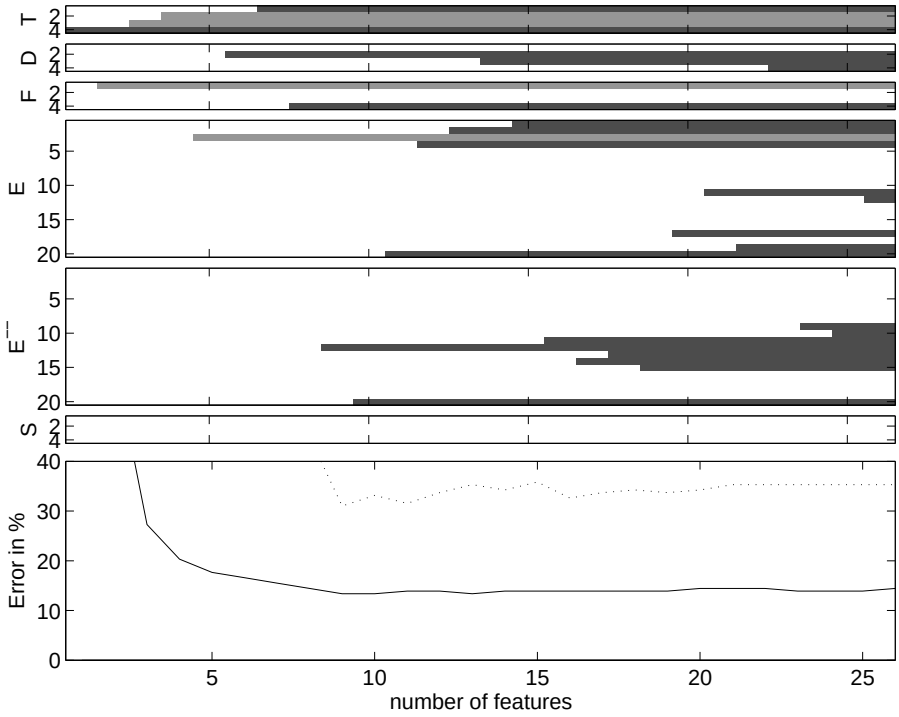

Abbildung 3: SFS-M, mit 2 Klassifikatoren initialisiert

Bei den Egebnissen von *SFS-M* fällt zunächst auf, dass die Fehlerkurven einen wesentlich stabileren Verlauf haben (siehe Abbildungen 3, 4 und 5). Bei der Hinzunahme von einem neuen Merkmal erfolgt also keine sprunghafte Veränderung der Fehlerquote. Dies ist damit begründbar, dass aufgrund der Möglichkeit, die neue Merkmalskomponente zu einem bestimmten Klassifikator hinzuzunehmen, die Ergebnisse ausgeglichener bleiben. Zum einen

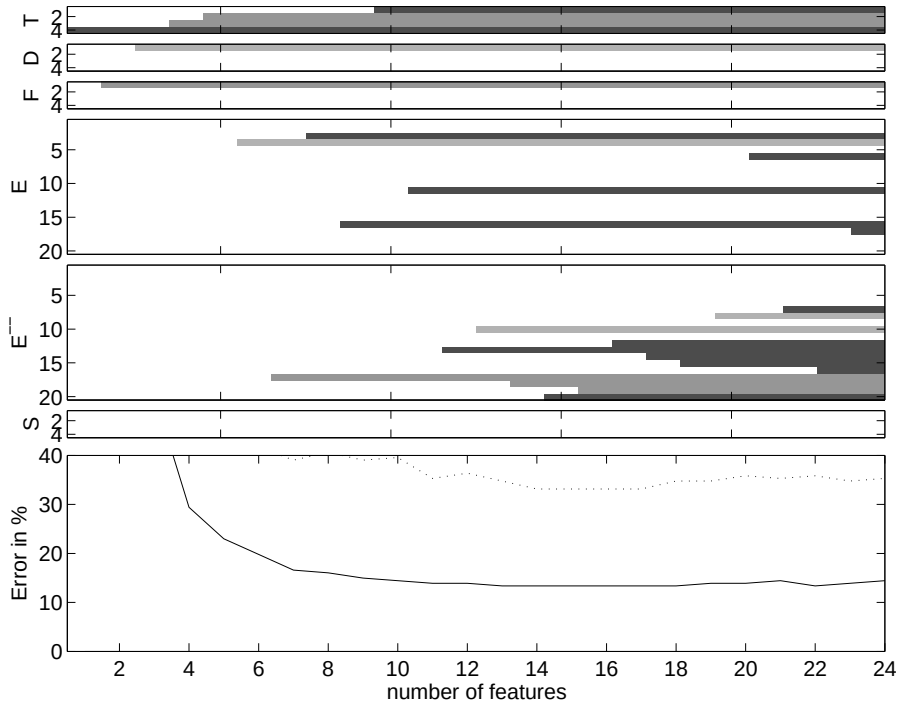

Abbildung 4: SFS-M, mit 3 Klassifikatoren initialisiert

wird nur der Klassifikator erweitert, zu dem ein neues Merkmal am besten passt, zum anderen gleichen die anderen Klassifikatoren in der Fehlerquote mögliche leichte Schwankungen aus.

Wenn man die Reihenfolge der Auswahl der Merkmalskomponente betrachtet, fällt auf, dass zunächst stets Merkmalskomponenten aus den Merkmalen F, D und T verwendet werden. Die Auswahl von Komponenten des Merkmals T erfolgt dabei stets als erstes. Wenn man Abbildung 1 betrachtet, sieht man, dass diese Merkmale für sich allein schon eine verhältnismäßig gute Klassifikation ermöglichen. Es besteht also ein direkter Zusammenhang zwischen der Güte eines Merkmals und dem Zeitpunkt der Auswahl. Diese Ergebnisse bestätigt die Aussage von Nischk, der in seiner Doktorarbeit[3] gezeigt hat, dass die Merkmale T, D und F besonders aussagekräftig für die Klassifikation von Grillen sind. Im Gegensatz dazu fällt auf, dass Komponenten der hochdimensionalen Merkmale E und E^{--} erst relativ spät gewählt werden. Dann werden allerdings stets mehrere Komponenten in einen Klassifikator gewählt, es findet also eine gewisse Spezialisierung der einzelnen Klassifikatoren auf bestimmte Merkmale statt. Dies ist sinnvoll, da eine einzelne Komponente eines hochdimensionalen Merkmals für sich noch keine gute Klassifikationsleistung erbringen kann.

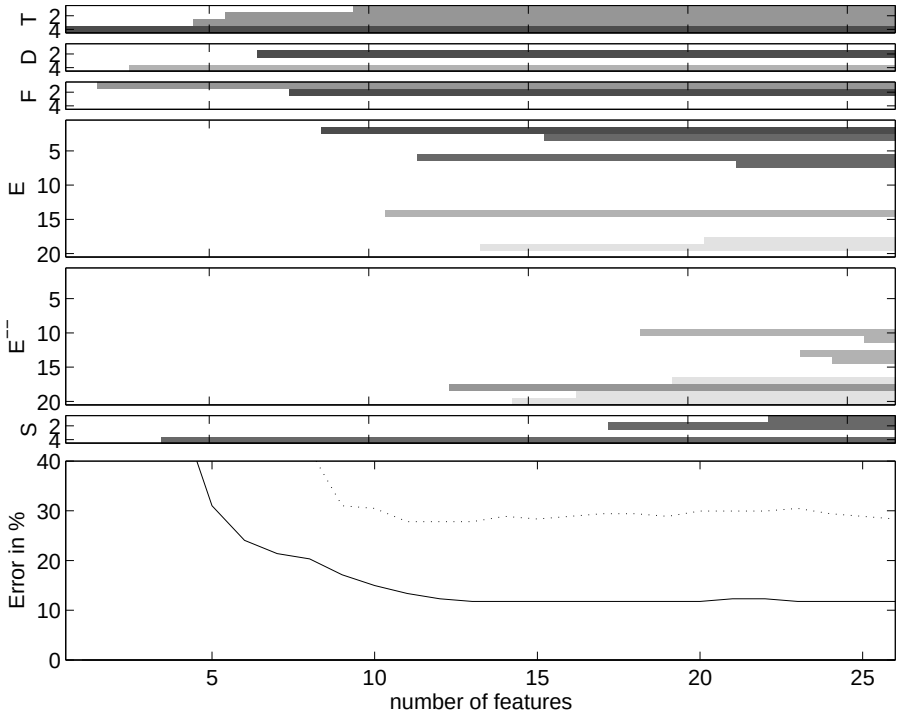

Abbildung 5: SFS-M, mit 4 Klassifikatoren initialisiert

6 Zusammenfassung

Obwohl der Ansatz der *Floating Search* Methoden in unserem Fall keine wirkliche Verbesserung der Fehlerquote brachte, ist es uns gelungen, die Klassifikationsergebnisse im Vergleich zur Vorwärtssuche zu verbessern. Dies und auch eine Stabilisierung der Ergebnisse brachte die Erweiterung der Vorwärtssuche auf Mehrklassifikatorsysteme. Desweiteren konnten die Merkmale, von denen schon vorher von Seite der Biologen bekannt war, dass sie für die Klassifikation gut geeignet sind, durch die Merkmalsselektionsalgorithmen bestätigt werden. Die unmittelbare Auswahl der guten Merkmale und die Gruppierung ähnlicher Merkmale in einzelnen Klassifikatoren ist als Indiz für die Eignung von Mehrklassifikatorsystemen bei diesem konkreten Problem anzusehen.

Interessant wäre es noch, ob eine Erweiterung der *SFS-M* um *Floating Search* Methoden, zum Beispiel ein optionaler Rückwärtschritt für jeden Klassifikator (analog zu *SFFS*), zu einer weiteren Verbesserung führen könnte. Aufgrund der nicht ermutigenden Ergebnisse bei *SFFS* halten wir dies aber für eher unwahrscheinlich.

Literatur

[1] C. Dietrich, F. Schwenker, K. Riede, and G. Palm. Classification of bioacoustic time series utilizing pulse detection, time and frequency features and data fusion. *www.informatik.uni-ulm.de/pw/berichte* Ulmer Informatik-Berichte 2001-04, University of Ulm, 2001.

[2] Josef Kittler and Fabio Roli, editors. *Multiple Classifier Systems, First International Workshop, MCS 2000, Cagliari, Italy, June 21-23, 2000, Proceedings*, volume 1857 of *Lecture Notes in Computer Science*. Springer, 2000.

[3] F. Nischk. *Die Grillengesellschaften zweier neotropischer Waldökosysteme in Ecuador*. PhD thesis, University of Köln, Germany, 1999. Memorandum UCB/ERL–M89/29 (in German).

[4] P. Pudil, F. Ferri, J. Novoviĉová, and J. Kittler. Floating search methods for feature selection with nonmonotonic criterion functions. In *Proceedings of the IEEE Intl. Conf. on Pattern Recognition*, pages 279–283, 1994.

[5] K. Riede. Acoustic monitoring of orthoptera and its potential for conversation. *Journal of Insect Conservation*, 2:217–223, 1998.