Fachbereich Informatik und Medien

Studienarbeit

Erstellung eines Klassifikationsmodells zur Identifikation von Vogelarten anhand ihrer Singstimme mit Merkmalsextraktion durch Marsyas

Vorgelegt von: Josef Mögelin

am: 28.02.2014

Betreuer: Dipl.-Inf. Ingo Boersch

Inhaltsverzeichnis

1 Einleitung

1.1 Motivation

Vögel können eine wichtige Rolle im Leben und Kultur der Menschen darstellen. Selbst in Großstädten kann man sie hören und fast jeder erkennt besonders markante Vögel an ihrer Singstimme. Für einige Menschen, wie zum Beispiel Musiker, können ihre Lieder eine Quelle der Inspiration sein. Doch Vögel singen nicht nur aus Vergnügen. [CS95] schrieb, dass Geräusche nur produziert werden, wenn sie auch benötigt werden und damit hat jedes Geräusch eine Bedeutung. So nutzen also Vögel oder Tiere im Allgemeinen ihre Stimme zur Kommunikation untereinander. Für biologische Forschungen und der Umweltüberwachung ist die Identifizierung von Tieren sehr bedeutsam. Insbesondere bei der Lokalisierung kann sie eine große Rolle spielen, schließlich werden Tiere oftmals zuerst gehört, bevor sie gesehen werden[KM98, LLH06]. Flugzeugunternehmen setzen bereits Systeme zur Vogellokalisierung ein, um Kollisionen zu vermeiden[CM06]. Es gibt also eine Vielzahl von sinnvollen Einsatzgebieten. Eine Menge Wissenschaftler, Umweltaktivisten und Biologen sind an der automatischen Klassifizierung interessiert. Darüber hinaus müssen oftmals Experten eingesetzt werden, um Vogelarten zu identifizieren. Durch Klassifizierungssysteme könnten Ornithologen entlastet werden und effizienter arbeiten. An der FH Brandenburg wird geplant, bei der Buga 2014, eine Applikation zur Echtzeiterkennung von Vogelarten zu entwickeln und einzusetzen.

1.2 Zielstellung und Abgrenzung

Ziel dieser Arbeit ist die Entwicklung eines Setups zur Identifikation von Vogelarten anhand ihrer Singstimme. Dabei soll die Merkmalsextraktion durch das bereits entwickelte Framework Marsyas durchgeführt werden. Um diesen Prozess zu entwickeln wird eine Datenmenge erstellt und aufbereitet. Dann werden die Merkmale durch das OpenSource Framework Marsyas extrahiert. Mit der extrahierten Merkmalsmenge soll ein Klassifikator trainiert und ein Modell erstellt werden. Bei der Datenmenge handelt es sich um Tracks von Vogelstimmen-CDs, welche keine

Lärm- oder Nebengeräusche besitzen. Diese Arbeit soll Grundlagen für weitere Arbeiten liefern, welche zum Beispiel eine Echtzeit-Identifikation von Vogelstimmen adressieren. Insbesondere soll herausgefunden werden, ob die berechneten Merkmale von Marsyas ausreichen, um Vogelarten erfolgreich zu identifizieren.

1.3 Aufbau der Arbeit

Zum besseren Verständnis, wird in kurzen Zügen der Aufbau dieser Arbeit im Folgenden beschrieben. Nach dem Einführungskapitel, welches Motivation und Aufgabenstellung umschreibt, folgt ein Grundlagenkapitel. In diesem werden Grundlagen zum Thema Audio, Vögel und Extraktion von Audiomerkmalen geliefert. Darüber hinaus wird kurz auf die verwendete Software eingegangen. Im dritten Kapitel, dem Hauptkapitel, wird die Umsetzung beschrieben und erläutert. Es werden alle Arbeitsschritte festgehalten und Ergebnisse protokolliert und interpretiert. Am Ende folgt eine Zusammenfassung und ein Ausblick für zukünftige Schritte.

1.4 Verwandte Arbeiten

Die Klassifizierung von Vogelstimmen wird in vielen verschiedenen Artikeln untersucht und beschrieben. Die ersten Ansätze zur automatischen Erkennung lieferten [ADM96] und [KM98]. Sie verwendeten Dynamic-Time-Warping und Hidden Markov Modelle um Zebrafink (*Taeniopygia guttata*) und Indigofink (*Passerina cyanea*) zu erkennen. Es wurden Spektrogramme zur Darstellung von Silben eingesetzt und mittels vordefinierten Prototypen eine Klassifizierung vorgenommen. Die Ergebnisse ihrer Untersuchungen zeigten, dass diese Methode sich besonders gut bei Aufnahmen mit wenigen Störgeräuschen eignet.

Mittlerweile ist die Bestimmung von eindeutigen Silben ein gängiges Verfahren zur Klassifizierung von Vogelarten. Auf ihnen werden Merkmalsvektoren erstellt und anschließend mit üblichen Klassifikatoren, wie zum Beispiel k-NN oder SVM, eine Vorhersage getroffen[SHF06, Fag07, Här03, LLH06, Fag04]. [MC97] waren einer der Ersten, die eine automatische Klassifizierung auf einer größeren Anzahl an Vogelarten untersuchten. Sie nutzten neuronale Netze zur Klassifizierung von sechs Vogelarten aus Kanada. Neuronale Netze wurden auch bereits erfolgreich zur Erkennung und Klassifizierung von akustischen Signalen bei anderen Tieren, wie zum Beispiel Walen und Fröschen eingesetzt[PR98, DFS99]. [Här03] untersuchte insgesamt 14 Vogelarten aus Nordeuropa. Sein Ziel war es festzustellen, ob Vogelarten sich direkt von Silben oder Elementen erkennen

lassen. Einen etwas anderen Ansatz versuchte [VEVT06]. Er verwendete Data Mining Techniken zur Klassifizierung und Analyse und verglich diese mit traditionellen Ansätzen.

Marsyas wurde bereits in [LJKK11] erfolgreich eingesetzt, wird ansonsten aber in der Literatur hauptsächlich vom Entwickler George Tzanetaki verwendet[TC99, TC00, Tza09].

2 Grundlagen

2.1 Audio

Akustik ist die Lehre vom Schall und seiner Ausbreitung. Schallwellen besitzen vier Eigenschaften: Geschwindigkeit, Amplitude, Wellenlänge und Frequenz. Durch diese physikalischen Eigenschaften wird jedes akustisches Signal (Ton, Geräusch) definiert. So dient zum Beispiel die Amplitude, also die Höhe eines Wellenberges, als Indikator für die Lautstärke. Die Frequenz hingegen, welche die Anzahl der Schwingungen in einem Zeitintervall wiedergibt, dient zur Festlegung der Tonhöhe. Das menschliche Gehör erkennt Töne bis zu einer maximalen Frequenz von 22kHz. Die Aufnahme von akustischen Signalen ist die Umwandlung des Schalldrucks in elektrische Spannung, welche kontinuierlich in der Zeit variiert. Diese Repräsentation ist allerdings analog. Für die Bearbeitung am Computer muss das Signal digitalisiert werden. Dabei werden die Spannungswerte an vielen Zeitpunkten gemessen, also diskretisiert. Dieser Vorgang wird auch als Sampling bezeichnet. Die Zahl der Samples (Messpunkte) pro Sekunde wird Abtastrate genannt. Je öfter eine Schallwelle abgetastet wird, desto genauer wird der originale Ton digital repräsentiert.[Wie13]

Ein hoher Ton wird nur dann genau abgetastet, wenn auch die Abtastrate entsprechend hoch ist. Das Theorem von Nyquist besagt, dass die Abtastrate mindestens doppelt so hoch sein muss, wie die höchste Frequenz, die noch gespeichert werden soll[Sha49]. Da menschliche Ohren Schallwellen von maximal 22.000 Hz wahrnehmen können, liegt die erforderliche Nyquist-Rate bei 44.000 Hz. Qualitativ hochwertige Audiosignale haben deswegen eine Abtastrate von 44kHz. Zur Aufzeichnung von digitalen Signalen gibt es verschiedene Verfahren. Das bekannteste ist die Pulse Code Modulation (PCM). Dabei wird für jeden Abtastzeitpunkt ein Signalwert gemessen. Zu den bekanntesten Audioformaten, welche PCM verwenden, gehören WAV und AIFF. Das Wave-Format wurde von Microsoft und IBM entwickelt und ist seit Windows 3.1 Standard auf PCs. Es unterstützt alle Sample-Raten und Kompressionen. Die Dateiendung lautet *.wav.

Vogelgeräusche können ähnlich untersucht und interpretiert werden wie die menschliche Sprache. Oftmals werden dafür Frequenz-Zeit Darstellungen wie zum Beispiel Spektrogramme verwendet. In der Literatur zur Vogelkunde wird auch von Sonogrammen gesprochen.[MC97]

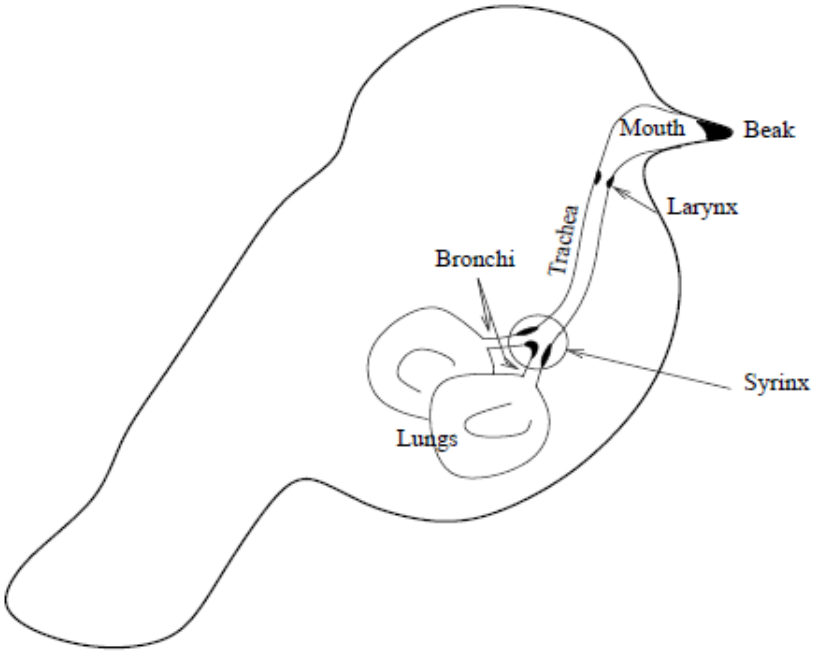

Abbildung 2.1: Bestandteile eines Vogels zur Geräuscherzeugung und deren Annordung [Fag04]

Audiodaten sind Zeitreihen, bei der die y-Achse den aktuellen Ausschlag (Amplitude) der Lautsprechermembran zu einer bestimmten Zeit auf der x-Achse repräsentiert. Sie sind univariat, endlich und äquidistant.[MM05]

2.2 Vogelstimmen

Vögel sind in der Lage eine Vielzahl von Klängen zu erzeugen. Die Luft aus der Lunge wird durch die Bronchien (*bronchia*) in die Syrinx gedrückt. Die Syrinx ist der Stimmkopf der Vögel und die Hautquelle der Geräuscherzeugung. Der Ton aus der Syrinx geht durch die Resonanzstrukturen von Luftröhre (*trachea*), Kehlkopf (*larynx*), Mund (*mouth*) und Schnabel (*beak*).[CM06] In Abbildung 2.1 werden die Bestandteile der Stimmerzeugung von Vögeln grafisch veranschaulicht. Es gibt viele Möglichkeiten Vögel in Gruppen zu unterteilen. Eine Möglichkeit basiert auf der Grundlage ihrer Tonerzeugung, welche tonal oder unharmonisch sein kann. Unharmonische Töne beispielsweise sind oftmals nur kurz und liegem im Frequenzbereich nah beinander. Vogelstimmen können sowohl Lieder (*songs*) als auch Rufe (*calls*) enthalten. Beide Geschlechter produzieren solche Geräusche über das ganze Jahr verteilt. Rufe sind oftmals kurz und einfach, Lieder hingegen besitzen eine komplexere Struktur und werden im Allgemeinen zur Revierverteidigung und Paarung genutzt[SHF06]. Nach [STT07] besitzen Vögel zwischen 5 und 15 unterschiedliche Rufe, welche verschiedene Bedeutungen wie “Achtung” oder “Aufregung” repräsentieren können.

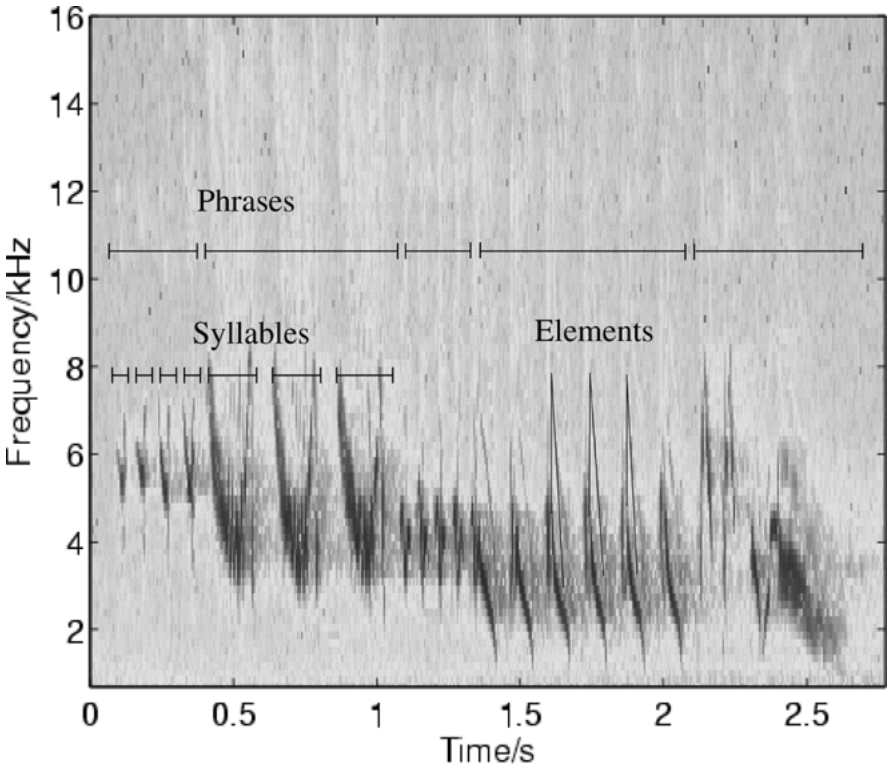

Abbildung 2.2: Einteilung einer Vogelstimme in verschiedene Hierarchielevel [CM06]

Zusätzlich können individuelle und regionale Unterschiede bei Vogelstimmen derselben Art entstehen[BL03]. Dies erschwert eine globale, weltweite Klassifikation von Vogelarten.

[KK80] und [CS95] gliedern Vogelstimmen des Weiteren in Phrasen (*phrases*), Silben (*syllables*) und Elemente (*elements*). Eine Silbe beinhaltet eine oder mehrere Elemente und ist üblicherweise bis zu einige hundert Millisekunden lang. Phrasen sind kurze Gruppierungen von Silben.[CM06] In Abbildung 2.2 wird die Einteilung einer Vogelstimme grafisch veranschaulicht.

2.3 Merkmalsextraktion

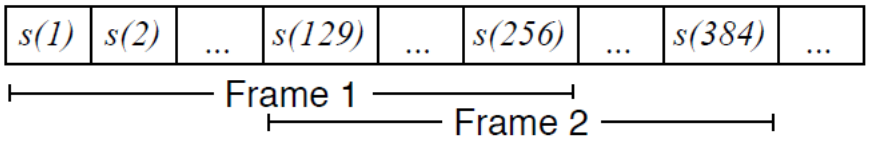

Abbildung 2.3: Zerlegung einer Reihe von Samples $s(t)$ in gleich große, sich überlappende Frames [BRF09]

Ein System zur Klassifizierung von Audiodaten beinhaltet und beginnt üblicherweise mit der

Extraktion von akustischen Merkmalen von Audiosignalen[BRF09]. Dabei werden kompakte, nummerische Werte berechnet, um Audiosegmente zu charakterisieren. Die Wahl der Merkmale ist oftmals die wichtigste und größte Herausforderung bei Klassifikationssystemen[TC02]. Zur Erkennung und Klassifizierung von Audiosignalen werden jene Merkmale genutzt, welche physikalische Charakteristiken der Geräuscherzeugung repräsentieren. So sind einige bekannte physikalische Merkmale, wie zum Beispiel die Schwerpunktwellenlänge (*spectral centroid*), die Signalbandbreite (*signal bandwidth*) und der Nulldurchgang (*zero crossing*), mathematisch berechnete Charakteristiken von Schallwellen. Neben physikalischen Merkmalen existieren auch Wahrnehmungsmerkmale, welche das Geräuschempfinden von Menschen widerspiegeln. Beispiele hierfür wären die Lautstärke und Tonhöhe. Generell können Audiomerkmale in temporale (Zeitbereich) und spektrale (Frequenzbereich) Merkmale unterteilt werden. Temporale Merkmale werden direkt auf der Schallwelle berechnet, wohingegen spektrale Merkmale erst berechnet werden können, wenn das Signal mit einer Fouriertransformation in einen Frequenzbereich transformiert wurde[Fag04]. Eine häufig verwendete Darstellungsart im Frequenzbereich ist das Spektrogramm. Um ein Spektrogramm zu erstellen wird ein Audiosignal in gleich große, sich überlappende Frames zerlegt (siehe Abbildung 2.3) und auf jedem Frame eine Fouriertransformation angewendet. Daraus ergeben sich Fourierkoeffizienten, dessen Beträge Aussage über die Intensität eines Geräusches liefern[BRF09]. Um gebräuchliche Algortihmen zur Klassifikation einzusetzen, müssen Audiosignale durch einen Vektor fester Länge repräsentiert werden. Ein üblicher Ansatz einen solchen Vektor zu erstellen ist die Identifizierung interessanter Frames durch Segmentierung (siehe Abschnitt 2.4). Anschließend werden Merkmale auf diesen Frames berechnet und der Mittelwert über alle Frames berechnet[BRF09]. Sobald alle Merkmale extrahiert wurden, können Techniken des maschinellen Lernens zur Klassifikation eingesetzt werden.

2.4 Segmentierung

Unter Segmentierung versteht man die Unterteilung einer Wertereihe in mehrere Segmente. Durch sie wird die Identifizierung von homogenen Bereichen ermöglicht. Elemente innerhalb eines Segments sollten möglichst bedeutenden Objekten beziehungsweise Ereignissen des unterliegenden Prozesses entsprechen[Wit83]. Die einfachste Art der Segmentierung stellt die statische Aufteilung einer Wertereihe in N gleichgroße Segmente dar. Der Nachteil dieses Vorgehen ist, dass die Segmente kein datenbasiertes Homogenitätskriterium erfüllen. Eine andere Möglichkeit stellt die Betrachtung der zugrundeliegenden Domaine dar. Im Bereich der Vogelkunde ist es üblich ein Audiosignal so zu zerlegen, dass ein Segment einer Silbe entspricht[BRF09]. Um eine solche

Segmentierung vorzunehmen, wird ein iterativer Algorithmus im Zeitbereich durchgeführt[Fag04]. Zuerst wird für jeden Frame die Energie berechnet. Dann wird ein anpassungsfähiger Schwellwert berechnet um Silben von Hintergrundgeräuschen zu trennen. Silben welche nah beieinander liegen werden gruppiert um einen so genannten Border-Effekt zu vermeiden[LSDM01]. Genauere Beschreibungen und weitere Algorithmen zur Segmentierung könnenn in [SHF06], [Fag04] und [CM06] nachgelesen werden.

2.5 Verwendete Software

2.5.1 CDex und Audacity

Um Audiodaten von einer CD zu extrahieren wird ein CD Ripper benötigt. Hierfür wurde das OpenSource Werkzeug CDex eingesetzt. Die Audiodaten können sowohl unkomprimiert als auch komprimiert gespeichert werden. Zur Bearbeitung der extrahierten Audiodaten wird das Programm Audacity verwendet. Dieses wurde in C++ programmiert und bietet eine übersichtliche, grafische Schnittstelle.

2.5.2 Marsyas

Zur Extrahierung von Merkmalen wird das OpenSource Framework Marsyas (*Music Analysis Retrieval and Synthesis for Audio Signals*), entwickelt von George Tzanetaki, eingesetzt. Es bietet die Möglichkeit Applikationen zur Analyse und Darstellung von Audioinhalten zu entwickeln. Der Schwerpunkt der entwickelten Systeme liegt dabei auf Music Information Retrieval-Systemen[TC00].

2.5.3 RapidMiner

Für die Erstellung eines Modells wird eine Umgebung für maschinelles Lernen, RapidMiner genannt, eingesetzt. Sie bietet die Möglichkeit, Prozesse zur Klassifikation möglichst einfach zu erstellen. Dabei werden Operatoren über eine grafische Oberfläche miteinander verbunden. Der Aufbau wird allerdings durch XML beschrieben und kann so ebenfalls bearbeitet werden. RapidMiner wurde 2001 vom Lehrstuhl für künstliche Intelligenz der Technischen Universität Dortmund entwickelt. Damals wurde das Projekt noch YALE genannt, was für *Yet Another Learning Environment* steht.

3 Umsetzung

3.1 Vorgehen

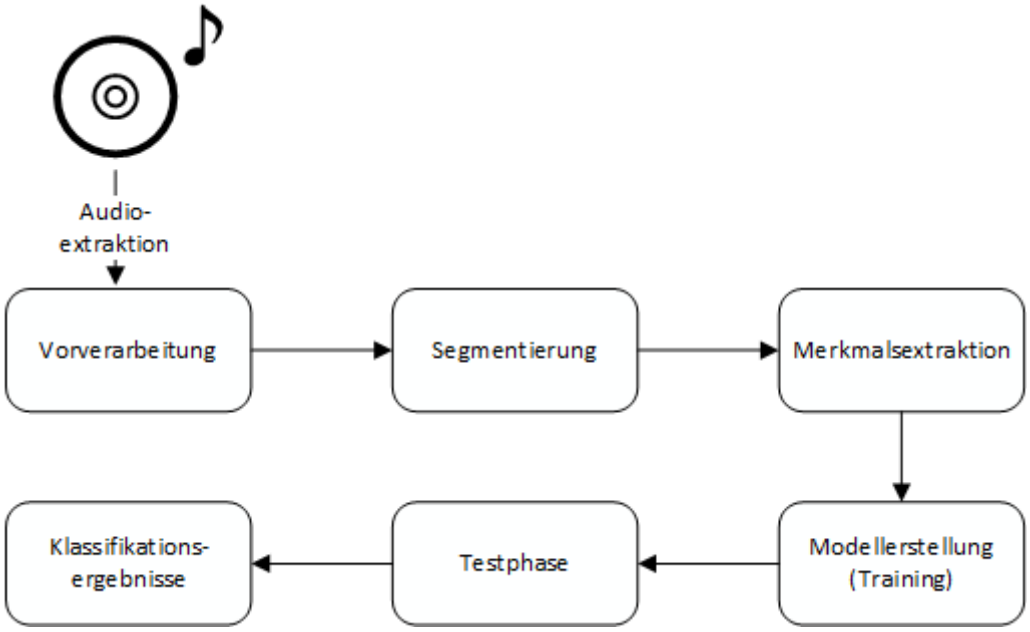

Abbildung 3.1: Arbeitsschritte zur Entwicklung eines Vorhersagemodells für Vogelrufe

Eine automatische Klassifizierung von Vogelstimmen kann durch Lärm- oder Nebengeräusche stark erschwert werden. Daher wurden Daten für diese Studie von Vogelstimmen-Audio-CDs (mit möglichst wenig Nebengeräuschen) extrahiert und in das Wave-Dateiformat konvertiert. Es wurden die Folgenden CDs verwendet: Jean C. Roché - Die Vogelstimmen Europas, Karl-Heinz Dingler - Unsere heimische Vogelwelt und Nigel Tucker - Zauberhafte Vogelstimmen. Um eine möglichst große Datenbasis aufzubauen, wurden nur Vögel ausgesucht, welche auf allen drei Datenquellen vorhanden sind. Sechs Vogelarten erfüllten diese Bedingung und wurden für diese Testreihe ausgesucht: Amsel (*Turdus merula*), Buchfink (*Fringilla coelebs*), Feldlerche (*Alauda arvensis*), Kohlmeise (*Parus major*), Nachtigall (*Luscinia megarhynchos*) und Rauchschwalbe (*Hirundo rustica*). Die Tracks der verschiedenen Datenquellen unterscheiden sich signifikant in der Länge der Aufnahme und den Variationen der Vogelrufe. Zum einen wird dadurch eine

Tabelle 3.1: Übersicht der, auf zwei Sekunden gesampelten, Datenmenge

Typ	Trainingsamples	Testsamples	Gesamtlänge (min)
Amsel (*Turdus merula*)	172	42	7:24
Buchfink (*Fringilla coelebs*)	131	31	5:37
Feldlerche (*Alauda arvensis*)	140	35	6:01
Kohlmeise (*Parus major*)	181	43	7:40
Nachtigall (*Luscinia megarhynchos*)	208	52	8:54
Rauchschwalbe (*Hirundo rustica*)	112	26	4:51

automatische Klassifizierung erschwert, zum anderen jedoch die Vogelwelt realistisch dargestellt. Vogelrufe können sich je nach Individuum und Region unterscheiden[STT07].

In Abbildung 3.1 werden die einzelnen Arbeitsschritte, welche im Folgenden genauer erläutert werden, vereinfacht grafisch dargestellt. Bei der Vorverarbeitung (*preprocessing*) der Datenmenge werden die extrahierten Daten in das Wave-Dateiformat konvertiert und einheitlich auf 22kHz gesampelt. Zusätzlich werden Pausen am Anfang und Ende eines jeden Tracks entfernt. Die Überlegung Pausen zwischen den einzelnen Vogelrufen zu entfernen oder zu verringern wurde verworfen. Denn es ist nicht ganz klar, ob die Länge der Pause zwischen zwei Rufen ein signifikantes Merkmal zur Identifizierung der Vogelart darstellt oder nicht.

Die Segmentierung würde üblicherweise wie in Kapitel 2.4 beschrieben durchgeführt. In dieser Studie soll jedoch getestet werden, inwiefern das Framework Marsyas ausreicht, um Merkmale zu extrahieren. Die ausgewählten Tracks werden in kleinere Samples von je zwei Sekunden zerteilt.

Das Programm Audacity bietet eine einfache Möglichkeit diese Zerlegung durchzuführen. Zuerst werden die Tracks durch die Funktion *Regular Interval Labels* mit einem Intervall von zwei Sekunden gelabelt. Das letzte Label wird in seiner Länge an die Titellänge angepasst und kann daher variieren und kürzer sein. Audacity bietet die Möglichkeit die Labels zu nummerieren und mit einem Pre- oder Postfix zu versehen. Anschließend können die Labels über den Menüpunkt *File* und der Funktion *Export Multiple* als eigenständige Dateien gespeichert werden. Aufgrund der überschaubaren Anzahl an Tracks (18) ist dieser semi-automatisierte Ansatz nicht mit zu viel Aufwand verbunden und daher ausreichend. Für eine deutlich größere Datenmenge würde es sich anbieten ein Pythonskript zur Segmentierung zu schreiben.

Die Samples werden nun durch ein Python-Skript (siehe Listing 3.1) in Trainings- und Testmenge im Verhältnis 4 zu 1 geteilt. Das Skript berechnet die Anzahl der Dateien im aktuellen Ordner und wie viele davon zur Testmenge gehören. Dann wird ein Ordner für die Testmenge angelegt. Es werden so viele Samples wie benötigt zufällig verschoben. Die restlichen Dateien werden als Trainingsdaten eingestuft und ebenfalls in einen seperaten Ordner verschoben. Damit ergibt sich

die in Tabelle 3.1 festgehaltenen Datenmengen.

Listing 3.1: Python-Skript zur Unterteilung der Samples in Tranings- und Testmenge

```
import os, fnmatch, shutil, random

c_dir = os.getcwd() #get path of current dir
dst_test = c_dir+'\\test' #destination path for test set
dst_training = c_dir+'\\training' #dest. path for training set
path, dirs, files = os.walk(c_dir).next() #assign file system variables
k = len(files)-1 #count samples excluding this script

os.mkdir('test')
for i in range(k/5): #20percent test set
    file = random.choice(os.listdir(c_dir)) #choose random file
    if fnmatch.fnmatch(file, '*.wav'):
        shutil.move(file, os.path.join(dst_test, file))

os.mkdir('training')
for file in os.listdir(c_dir):
    if fnmatch.fnmatch(file, '*.wav'):
        shutil.move(file, os.path.join(dst_training, file))
```

Die Merkmalsberechnung (*feature calculation*) wird dann mit Hilfe des Frameworks Marsyas durchgeführt. Aufgrund einiger Probleme bei Kompilierung und Ausführung unter Windows wurde Marsyas auf einer virtuellen Umgebung mit Ubuntu 12.10 x86 installiert und eingesetzt. Eine ausführlichere Beschreibung der Merkmalsberechnung wird in Abschnitt 3.2 vorgenommen. Die berechneten Merkmale werden dann an RapidMiner übergeben um ein Modell zur Klassifikation der Vogelarten anhand ihrer Singstimme zu entwickeln. Bei der Modellerstellung gibt es zwei Phasen: Tranings- und Testphase. In der ersten Phase wird ein Modell auf einer bestimmten Menge gebaut beziehungsweise trainiert. Zur Überprüfung des Erfolgs wird in der Testphase eine dem Modell unbekannte Menge übergeben. Das Ergebnis aus dieser Phase ist dann repräsentativ für den gesamten Prozess.

3.2 Merkmalsberechnung

Aus der extrahierten und gesampelten Datenmenge werden nun mittels Marsyas Merkmale extrahiert. Das Framework besitzt verschiedene Werkzeuge die über eine Konsole eingesetzt werden können. Für die Merkmalsberechnung werden die beiden Werkzeuge *mkcollection* und

bextract benötigt.

3.2.1 mkcollection

Marsyas arbeitet generell auf *Collection-Files*, also Sammlungen von Dateien. In diesen werden die Speicherorte der Samples festgehalten. Es besteht zusätzlich die Möglichkeit, die Samples mit einem Label zu versehen. Dies ist insbesondere für die Merkmalsberechnung und Klassifizierung sehr wichtig. In Listing 3.2 wird gezeigt, wie eine Collection angelegt werden kann. Standardgemäß wird die Dateiendung *.mf für eine *Collection* verwendet.

Listing 3.2: Erstellung eines Collection-Files mittels mkcollection

```
./mkcollection -c amselSamples.mf -l amsel /home/ubuntu/data/sound/amsel
```

Der Parameter -c steht für *Collection* und bestimmt den Namen der erstellten Datei. Mit -l werden alle Dateien innerhalb des angegebenen Verzeichnisses gelabelt. Marsyas erzeugt eine Liste aller sich in dem Verzeichnis befindenden Sounddateien (nur *.wav und *.au). Die Daten werden allerdings nicht validiert, es findet also keine Fehlerüberprüfung statt. Die Entwickler empfehlen die Sounddateien vorher auf 22kHz zu samplen, da Marsyas keine automatische Sampling-Konvertierung vornimmt[PT14]. Ein Auszug einer erstellten *Collection* ist in Listing 3.3 dargestellt.

Listing 3.3: Auszug einer erstellten Kollektion

```
/home/ubuntu/sound/amsel/01.wav     amsel
/home/ubuntu/sound/amsel/22.wav     amsel
/home/ubuntu/sound/amsel/114.wav    amsel
/home/ubuntu/sound/amsel/83.wav     amsel
...
```

Es werden für jede Vogelart, insgesamt sechs, *Collection-Files* mit zugehörigem Label angelegt. Diese können nun vom Framework Marsyas als Parameter weiterverarbeitet werden.

3.2.2 bextract

Marsyas' Fokus liegt auf der Verarbeitung von Audiosignalen. Das Werkzeug *bextract* extrahiert beziehungsweise berechnet auf Audio spezialisierte Merkmale. Obwohl dieses Werkzeug die Möglichkeit bietet ein Klassifikationsmodell zu erstellen, wird in dieser Arbeit darauf verzichtet.

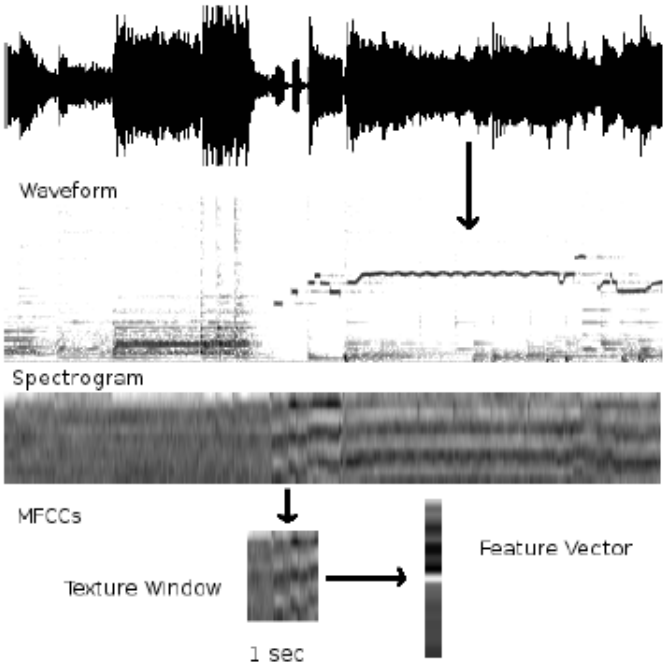

Abbildung 3.2: Prozessvisualisierung der Merkmalsextraktion durch Marsyas[Tza09]

Der Grund dafür liegt in den nur beschränkten Möglichkeiten zur Modellerstellung und Performanzberechnung. *Bextract* bietet eine Vielzahl an Kommandozeilenparametern. In Tabelle 3.2 werden die Wichtigsten vorgestellt und erläutert. Weitere Parameter können in der Betriebsanleitung [PT14] nachgeschlagen werden. Marsyas kann derzeit Folgende Merkmale berechnen[1]: Zero-Crossing, Spectral Centroid, Spectral Flux, Spectral Rolloff, Chroma und Mel-Frequency Cepstral Coefficients (MFCC). Im Folgenden werden die einzelnen Merkmale kurz vorgestellt.

Zero Crossings Misst Geräuschhaftigkeit (*noisiness*) durch Berechnung der Anzahl der Vorzeichenwechsel.

$$Z_t = \frac{1}{2} \sum_{n=1}^{N} |sign(x[n]) - sign(x[n-1])| \tag{3.1}$$

Sign()-Funktion besitzt Rückgabewert 1 für positive und 0 für negative Argumente. $x[n]$ repräsentiert das Signal der Zeitdomäne für Frame t.

Spectral Centroid Maß zur Bestimmung der Klangfarbe. Bestimmt Mittelpunkt des Frequenzspektrums. Im Allgemeinen wird die Spektralform beschrieben.

[1] es werden nur die englischen Fachbegriffe der Merkmale verwendet, eine Übersetzung ins Deutsche folgt nicht

$$C_t = \frac{\sum_{n=1}^{N} M_t[n] * n}{\sum_{n=1}^{N} M_t[n]} \tag{3.2}$$

$M_t[n]$ ist der Betrag der Fouriertransformation im Frame t und Frequenz-Bin n.

Spectral Rolloff Ein weiteres Maß zur Beschreibung der Spektralform. Beschreibt Schiefe (*skewness*) des Powerspektrums.

$$\sum_{n=1}^{R_t} M_t[n] = 0.85 * \sum_{n=1}^{N} M_t[n]. \tag{3.3}$$

Die Frequenz R_t, unterhalb derer 85% der Energieverteilung erreicht wird.

Spectral Flux Beschreibt wie schnell und oft sich das Powerspektrum verändert. Vergleicht Powerspektrum eines Frames mit Powerspektrum des vorherigen Frames.

$$F_t = \sum_{n=1}^{N} (N_t[n] - N_{t-1}[n])^2 \tag{3.4}$$

$N_t[n]$ und $N_{t-1}[n]$ sind die normalisierten Beträge der Fouriertransformation im aktuellen und vorherigen Frame.

Mel-Frequency Cepstral Coefficents Kompakte Darstellung des Frequenzspektrums. Überwiegend in der Spracherkennung eingesetzt. Für eine ausführlichere Erläuterung wird auf [Log00] verwiesen.

Chroma Ein Merkmal zur Beschreibung der Tonhöhe. Teilt das Frequenzspektrum in 12 Bins. Ein Bin entspricht dabei einem der 12 Halbtöne (*chroma*) einer Oktave. Die Verteilung der Chroma kann dabei Informationen des Audiosignals enthalten, welche im Frequenzspektrum nicht offensichtlich sind.

Um diese Merkmale zu berechnen, wird das Audiosignal mit einer Kurzzeit-Fourier-Transformation (STFT) in den Spektralbereich transformiert. Das Signal lässt sich nun in Frames oder Fenster unterteilen. Bei Marsyas ist das Analysefenster und die Verschiebung des Fensters standardgemäß 512 Samples groß. Beide Werte können individuell eingestellt werden (siehe Tabelle 3.2). Die eigentlichen Merkmale werden dann durch Mittelwert (*mean*) und Standardabweichung (*standard deviation*) über die letzten M Frames berechnet:

$$m\Phi(t) = mean[\Phi(t - M + 1), \ldots, \Phi(t)] \tag{3.5}$$

$$s\Phi(t) = std[\Phi(t - M + 1), \ldots, \Phi(t)] \tag{3.6}$$

$\Phi(t)$ entspricht dem ursprünglichen Einzelmerkmalsvektor (*single feature vector*). Für M wurde der Wert 40 verwendet, was einem Fenster von ungefähr 1 Sekunde entspricht. Dieser Prozess ist noch einmal in Abbildung 3.2 veranschaulicht.

In Listing 3.4 wird der in dieser Arbeit verwendete Aufruf für bextract dargestellt.

Listing 3.4: Aufruf von bextract zur Merkmalsberechnung

```
./bextract -fe -sv -c amsel.mf buchfink.mf nachtigall.mf rauchschwalbe.mf feldlerche.mf
    kohlmeise.mf -w features.arff
```

Die berechneten Merkmale werden in der Datei *features.arff* gespeichert. Insgesamt wurden 125 Merkmale inklusive Label berechnet. Es handelt sich dabei um ein für Weka entwickeltes Dateiformat und ist auf maschinelles Lernen spezialisiert. Die Datei besteht aus zwei Teilen: Header und Daten. Im Header werden die Merkmale definiert. Die Daten werden dann zeilenweise und die Merkmale durch Komma separiert dargestellt. Die durch Marsyas generierten Merkmale besitzen lange, sich selbst beschreibende Namen. Ein Merkmalsname beschreibt den Prozess der Berechnung. Eine Zeile im Datenteil der Weka-Datei entspricht einem Merkmalsvektor und damit einem extrahierten Sample von zwei Sekunden. Zur leichteren Zuordnung werden die Namen der Audiodateien als Kommentare integriert. Ein Auszug der generierten ARFF-Datei ist in Listing 3.5 zu finden.

Listing 3.5: Beispielauszug einer generierten ARFF-Datei

```
% Created by Marsyas
@relation features.arff
@attribute output {amsel,buchfink,feldlerche,kohlmeise,nachtigall,rauchschwalbe}
@attribute Mean_Acc5_Mean_Mem20_ZeroCrossings_HopSize512_WinSize512 real
@attribute Mean_Acc5_Mean_Mem20_Centroid_powerFFT_WinHamming_HopSize512_WinSize512 real
@attribute Mean_Acc5_Mean_Mem20_Rolloff_powerFFT_WinHamming_HopSize512_WinSize512 real
...


@data
% filename ../../sound/amsel/01.wav
amsel,0.198119,0.169417,0.252675, ...
```

```
% filename ../../sound/amsel/02.wav
amsel,0.242027,0.238858,0.288132, ...
...
```

Zum besseren Verständnis wird die Namensgebung im Folgenden genauer erläutert. Der Name ist nach dem Schema A_B_C aufgebaut, wobei A, B und C für Funktionen stehen, welche dem Prinzip $A(B(C)))$ folgen. Das bedeutet im Umkehrschluss, dass C vor B und B vor A berechnet wird. Als Beispiel eine Erklärung mit dem Merkmal:

Mean_Acc5_Mean_Mem20_MFCC2_Power_powerFFT_WinHamming_HopSize512_WinSize1024.

Alle 512 Samples wird ein Fenster von 1024 Samples erstellt. Dieses Fenster wird mit der Hammingfunktion multipliziert. Es wird das Powerspektrum gebildet und der dritte Koeffizient der Mel-Frequenz-Cepstrum-Koeffizienten berechnet. Dann wird der Mittelwert der letzten 20 berechneteten MFCC2-Koeffizienten gebildet. Zum Abschluss wird noch einmal der Mittelwert aus den fünf vorherigen Mittelwerten berechnet. Acc steht für Accumulator und verhält sich wie folgt:

$$AccN : out(i) = [in(i * N), in(i * N - 1), in(i * N - 2), \ldots, in(i * N - N + 1)].$$

Mem hingegen steht für Memory und lässt sich ebenfalls mathematisch ausdrücken:

$$MemN : out(i) = [in(i), in(i - 1), in(i - 2), \ldots, in(i - N + 1)].$$

Für Mem gilt die Anzahl der Ausgabessamples entspricht gleich der Anzahl der Eingabesamples, wohingegen für den Accumulator die Ausgabesamples gleich dem Quotienten der Eingabesamples zu N sind. Die Parameter für Acc und Mem können mittels -as und -m geändert werden.

Es ist wichtig zu erwähnen, dass Marsyas eine Kollektion von Audiodateien als eine große Audiodatei behandelt. Dies hat zur Folge, dass die Grenzen zwischen Audiodateien überschritten werden. Eine Überanpassung (*overfitting*) an die Datenmenge kann somit nicht direkt verhindert werden. Die Performanzwerte könnten also beeinflusst werden. Um dies zu vermeiden wird die gesampelte Datenmenge bereits zu Beginn in Tranings- und Testmenge unterteilt. Es werden daraufhin zwei Kollektion je Vogelart erstellt, eine jeweils für Trainings- und Testmenge. Somit wird die Überschneidung zwischen diesen beiden Mengen aufgehoben. Eine weitere Möglichkeit wäre es, die ersten $m - 1$ Deskriptoren der Audiodatei n wegzulassen, denn diese beinhalten

Audioinformationen der Datei $n-1$. Diese Möglichkeit bietet sich aber nur an, wenn man Marsyas in Kombination mit Python oder C++ verwendet.

Tabelle 3.2: Wichtige Kommandozeilenparameter und deren Bedeutung für *bextract*

Parameter	Bedeutung
-c	Collection-Files, welche genutzt werden sollen
-fe	Es werden nur Merkmale extrahiert, aber kein Klassifizierer trainiert
-w	Dateiname der generierten .arff Datei (Weka)
-ws	Fenstergröße (Standard: 512)
-hp	Fensterverschiebung (Standard: 512)
-sv	Attribute werden als Einzelvektor berechnet

3.3 Modellerstellung in RapidMiner

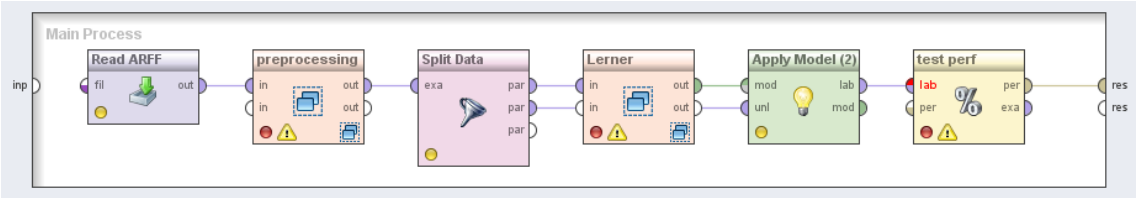

Abbildung 3.3: Prozesskette zur Modellerstellung in RapidMiner

Zur Erstellung eines Klassifikationsmodells wird RapidMiner 5.3 eingesetzt. Dieses Werkzeug bietet die Möglichkeit Operatoren über eine visuelle Schnittstelle miteinander zu verbinden. Somit können schnell und einfach verschiedene Setups erstellt und getestet werden. Der Grundaufbau bleibt für Klassifikationsmodelle eigentlich immer gleich. Zu Beginn wird eine Datenmenge mit allen berechneten Merkmalen und zugehörigem Klassifikationslabel eingelesen. Auf dieser können dann Transformationen, wie zum Beispiel die Entfernung von unnötigen Attributen, vorgenommen werden. Eine Unterteilung in Trainings- und Testmenge ist zu empfehlen, denn dadurch kann eine Überanpassung (*overfitting*) an die Datenmenge verhindert werden. Es folgt die Erstellung eines Modells, welches sich je nach Lerner unterscheiden kann. Ein gängiges Verfahren ist die Kombination einer 10fache-Kreuzvalidierung mit einem Lerner. Bei manchen Modellen, wie zum Beispiel neuronalen Netzen oder Random Forest, ist dies aus Zeitgründen jedoch nicht immer optimal. Es ist daher situationsbedingt zu entscheiden, ob sich der Zeitaufwand einer Kreuzvalidierung zur Verbesserung der Performanz beziehungsweise Sicherheit lohnt. Abschließend wird der Testfehler auf den ungesehenen Daten berechnet. In Abbildung 3.3 ist eine solche Prozesskette aus RapidMiner grafisch dargestellt. Die in dieser Arbeit verwendete Prozesskette

weicht minimal von der in der Abbildung dargestellten ab. Anstatt zu Beginn die gesamte Datenmenge einzulesen, werden Trainings- und Testmenge seperat eingelesen (siehe Ende von Teilabschnitt 3.2.2).

Das vorherzusagende Merkmal wird Label genannt. Dieses Label kann je nach Situation numerical, bi- oder polynominal sein. In unserem Fall handelt es sich um ein polynominales Label, da wir insgesamt sechs Vogelarten vorraussagen wollen. Diese Eigenschaft ist wichtig bei der Wahl eines Lerners. Nicht alle Lerner können mit numerischen oder polynominalen Labels umgehen. Um die Performanz eines Lerners einschätzen zu können empfiehlt es sich einen Initial-Performanzwert zu berechnen. Hierfür wird das DefaultModel verwendet. Für das richtige Modell werden Entscheidungsbaum, neuronales Netz und RandomForest verwendet und getestet.

3.4 Ergebnisse und Auswertung

Das eingesetzte DefaultModel basiert auf dem Mittelwert der Trainingsdaten. Es wählt die Vogelart mit den meisten Trainingsdaten und gibt als Antwort exakt diese für jede zu klassifizierende Testreihe. Bei binominalen Labels können dadurch sogar relativ hohe Performanzwerte (>50%) erzielt werden. Für polynominale Labels werden hingegen deutlich schlechtere Werte erzielt. Das DefaultModel wählte aufgrund der Verteilung der Trainingsdaten die Nachtigall als Antwort für alle Testdaten. Die Resubstitutionserfolgsrate im Traning lag dabei nur bei 22,03%. Auf dem Testset konnte eine leichte Verbesserung der Erfolgsrate (*accuracy*) von 22,71% verzeichnet werden. Es wurden insgesamt 52 von 229 Testdaten richtig klassifiziert. Die Erkennungsrate von **22,71%** wird als Initialwert angesehen. Die trainierten Lerner sollten diesen Wert deutlich übersteigen, ansonsten würde sich die Merkmalsgenerierung durch Marsyas zur Klassifizierung von Vogelarten auf Grundlage der Singstimme nicht eignen.

Als Erstes wird ein Entscheidungsbaum mit Entscheidungskriterium auf der relativen Entropie (Kullback-Leibler-Divergenz, *information gain*) erstellt. Es sei angemerkt, dass der Baum nicht gestutzt (*no prunning*) wird. Die Erkennungsrate auf den Trainingsdaten liegt bei 94,07%. Dieser Wert ist zwar sehr gut, jedoch nicht wirklich entscheidend, da das erstellte Modell alle Daten gesehen hat und sich diesen angepasst hat. Viel entscheidender ist der Performanzwert auf den Testdaten. Hier konnte der Entscheidungsbaum eine Erfolgsrate von **74,67%** erzielen. Dies stellt schon einmal eine deutliche Verbesserung gegenüber dem Initialwert des DefaultModels dar. Die Kontingenztabelle des Entscheidungsbaumes wird in Tabelle 3.3 dargestellt. Es wurden 58 Testdaten von insgesamt 229 falsch klassifiziert. Die Rauchschwalbe wurde am besten erkannt, die Kohlmeise am schlechtesten. Es sollte jedoch beachtet werden, dass die Rauchschwalbe die

geringste Anzahl an Testdaten besitzt. Ein weiterer interessanter Aspekt ist die Tatsache, dass der Einsatz einer 10fachen Kreuzvalidierung keinen Einfluss auf die Erfolgsrate hatte, weder auf den Trainings-, noch auf den Testdaten.

Das zweite Modell wurde durch das Klassifikationsverfahren Random Forest von Weka erstellt. Dabei handelt es sich um eine Vielzahl von Entscheidungsbäumen, welche während dem Training durch eine bestimmte Art der Randomisierung gewachsen sind. Jeder erstellte Baum darf eine Entscheidung zur Klassifizierung treffen. Die Klasse mit den meisten Stimmen wird als endgültige Entscheidung verwendet. Als Parameter wurden 100 generierte Bäume ausgewählt. Auf den Trainingsdaten konnte eine Erkennungsrate von 100% erreicht werden. Dies zeigt, wie gut ein Verfahren der Mehrheitsbestimmung zur Klassifikation geeignet ist. Andererseits liegt die Vermutung nahe, dass eine zu starke Anpassung an die Trainingsdaten vorliegen könnte. Dennoch konnte auf dem Testset eine ebenfalls sehr gute Erfolgsrate von **87,34%** erzielt werden. Die Kontingenztabelle für Random Forest ist in Tabelle 3.4 zu finden. Die Nachtigall wurde lediglich einmal falsch klassifiziert und wurde damit mit Abstand am besten erkannt. Die Feldlerche wurde mit acht Falsch identifizierten am schlechtesten bestimmt. Random Forest erstellt zwar ein transparentes Modell, dieses ist jedoch durch die Hohe Komplexität und Anzahl der Bäume kaum im Ganzen zu erfassen. Eine tiefgründigere Analyse ist daher mit viel Aufwand verbunden.

Das letzte Modell wird durch ein neuronales Netz mit 500 Traningszyklen erstellt. Die Resubstitutionserkennungsrate liegt bei 99,47%. Auch hier könnte eine zu starke Anpassung an die Traningsdaten vorliegen. Dennoch erreichte das neuronale Netz eine Erkennungsrate von **88,65%** auf den Testdaten und ist damit das erfolgreichste Modell. Die Performanzwerte können in Tabelle 3.5 eingesehen werden. Die Amsel wurde mit 95,24% Recall am besten erkannt, dicht gefolgt von der Nachtigall. Die Rauchschwalbe wird mit 80,77% zwar am schlechtesten, aber immernoch gut erkannt. Das neuronale Netz besitzt allerdings den Nachteil, dass es keine Transparenz bietet und realtiv lange für das Training benötigt. Weder Kreuzvalidierung noch eine erhöhte Anzahl an Trainingszyklen konnten eine Performanzsteigerung dieses Modells bewirken.

Insgesamt konnten alle drei Modelle eine solide Leistung bei der Klassifikation erzielen. Das neuronale Netz und Random Forest sind jedoch dem einfachen Entscheidungsbaum vorzuziehen. Die Performanzwerte zeigen, dass eine Klassifizierung der Vogelarten durch die generierten Merkmale von Marsyas möglich ist. Es ist schwer Gemeinsamkeiten der drei Modelle zu finden, da ein Modell den beiden anderen oftmals widerspricht. So wird zum Beispiel der Buchfink beim neuronalen Netz und Random Forest hauptsächlich mit der Kohlmeise falsch klassifiziert. Das Modell auf Basis eines einfachen Entscheidungsbaumes klassifizierte jedoch die Amsel am häufigsten als Kohlmeise falsch. Es kann daher nicht exakt festgestellt werden, welche Vogelart

einer anderen zu sehr ähnelt, so dass diese leicht Verwechselt werden. Generell kann gesagt werden, dass Amsel und Nachtigall in allen drei Modellen durchweg sehr gut erkannt wurden. Dies könnte an der größeren zur Verfügung gestellten Trainingsmenge liegen, als auch an den gut erkennbaren Vogelrufen.

Tabelle 3.3: Kontingenztabelle auf Basis eines Entscheidungsbaums

	true amsel	true nachtigall	true buchfink	true kohlmeise	true rauchschwalbe	true feldlerche	class precision
pred. amsel	35	7	5	2	0	2	68,63%
pred. nachtigall	4	42	1	5	0	3	76,36%
pred. buchfink	1	1	20	8	0	2	62,5%
pred. kohlmeise	1	1	2	26	0	2	81,25%
pred. rauchschwalbe	1	1	2	1	23	1	79,31%
pred. feldlerche	0	0	1	1	3	25	83,33%
class recall	83,33%	80,77%	64,52%	60,47%	88,46%	71,43%	

Tabelle 3.4: Kontingenztabelle auf Basis von Weka-RandomForest

	true amsel	true nachtigall	true buchfink	true kohlmeise	true rauchschwalbe	true feldlerche	class precision
pred. amsel	36	0	0	1	0	0	97,3%
pred. nachtigall	4	51	1	4	0	4	79,69%
pred. buchfink	1	0	27	1	0	0	93,1%
pred. kohlmeise	0	1	3	37	0	0	90,24%
pred. rauchschwalbe	1	0	0	0	22	4	81,48%
pred. feldlerche	0	0	0	0	4	27	87,1%
class recall	85,71%	98,08%	87,10%	86,05%	84,62%	77,14%	

Tabelle 3.5: Kontingenztabelle auf Basis eines neuronalen Netzes

	true amsel	true nachtigall	true buchfink	true kohlmeise	true rauchschwalbe	true feldlerche	class precision
pred. amsel	40	1	0	2	0	0	93,02
pred. nachtigall	0	49	1	0	0	0	98,0%
pred. buchfink	1	1	27	5	0	0	79,41%
pred. kohlmeise	1	1	3	36	1	1	83,72%
pred. rauchschwalbe	0	0	0	0	21	4	84,0%
pred. feldlerche	0	0	0	0	4	30	88,24%
class recall	95,24%	94,23%	87,10%	83,72%	80,77%	85,71	

4 Zusammenfassung

4.1 Fazit

Ziel dieser Arbeit war es, einen Prozess zur Identifikation einer Vogelart anhand seiner Singstimme zu erstellen. Dabei sollte das Framework Marsyas zur Merkmalsextraktion eingesetzt werden. Der Prozess wurde für die Fachhochschule Brandenburg an einem Beispiel mit sechs Vogelarten erfolgreich erstellt. Es wurden Merkmale durch Marsyas extrahiert und in einem Modell in RapidMiner eingesetzt. Diese Arbeit erläutert Vorgehen und Durchführung und bietet Möglichkeiten zur Weiterentwicklung des Prozesses an verschiedenen Stellen. Die Ergebnisse zeigen, dass eine Klassifizierung durch automatisch generierte Merkmale von Marsyas möglich ist. Das beste Ergebnisse erreichte ein neuronales Netz mit einer Erkennungsrate von 88,65%. Damit wurden nur 26 der 229 vorhandenen Testdaten falsch klassifiziert. Eine höhere Anzahl an Trainings- und Testdaten bewirkte bei der Erkennung von Amsel und Nachtigall auf allen drei Modellen eine durchweg gute Erkennungsrate. Dennoch kommt es viel mehr auf die Qualität dieser Daten, anstatt auf die Quantität an. Die Kohlmeise hatte ebenfalls eine relativ hohe Anzahl an Daten zur Verfügung und wurde Teilweise am schlechtesten erkannt. Inwiefern die erreichten Ergebnisse zufriedenstellend oder ausreichend sind, hängt von der jeweiligen Situation oder dem Aufgabenumfeld ab. Anhand der drei erstellten Modelle ist zu erkennen, dass die Wahl des Modells durchaus eine große Rolle spielt. Eine Verbesserung des Modells durch Veränderung der Parameter wäre denkbar. Sowohl eine bessere Segmentierung, als auch andere Merkmale könnten ebenfalls einen positiven Einfluss auf die Performanzwerte haben. Alles in allem sind die Ergebnisse überraschend positiv. Mit relativ geringem Aufwand, das bedeutet keine aufwändige Segmentierung oder Merkmalsgenerierung, konnte ein recht zuverlässiges Klassifizierungsmodell erstellt werden. Die Trennung zwischen Merkmalsgenerierung und Modellerstellung ermöglicht einen schnellen Austausch beziehungsweise Wechsel des Vorgehens. So können leicht andere Merkmale generiert, aber die gleiche Prozesskette zur Erstellung des Modells verwendet werden. Die Arbeit bietet einen guten Einstieg in die Thematik zur Klassifizierung von Vogelstimmen und kann für weiterführende Arbeiten sicherlich eine Hilfestellung sein.

4.2 Ausblick

Der in dieser Arbeit erstellte Prozess bietet viele Möglichkeiten für weiterführende Aufgaben. Die Segmentierung ist dabei wohl eine der wichtigsten und wurde in dieser Arbeit bereits im Grundlagenkapitel angesprochen. Anstatt einer statischen Segmentierung sollten Silben erkannt und extrahiert werden. Einen guten Einstieg bietet dabei das Werkzeug Soundruler. Um Marsyas jedoch mit einer solchen Segmentierung verwenden zu können, müsste das Framework durch C++ oder Python eingesetzt werden und nicht über die Konsole. Eine weitere Möglichkeit zur Verbesserung des Prozesses wäre die Veränderung der Parameter. Sowohl die Merkmalsgenerierung von Marsyas als auch RapidMiner bieten genügend Optionen dafür an. Jede Veränderung am Prozess sollte dabei protokolliert und mit den Ergebnissen dieser Arbeit verglichen werden. Des Weiteren könnten Teile der Prozesskette ausgetauscht werden. So könnte beispielsweise die Programmiersprache R die Modellerstellung durch RapidMiner ersetzen. Im Laufe dieser Arbeit wurde ein weiteres Werkzeug zur Merkmalsextraktion entdeckt: jAudio. Dieses bietete eine grafische Oberfläche und eine Vielzahl von Audiomerkmalen an. Ein Vergleich dieses Werkzeugs mit Marsyas wäre durchaus interessant. Als letzten Punkt steht die tiefere Analyse der erzielten Ergebnisse. Eine Zusammensetzung mit einem Ornithologen und einer Analyse seines Expertenwissens könnte bessere Einblicke in die Thematik liefern. Da Experten an einem Spektrogramm Vogelarten erkennen können, müssen signifikante Merkmale existieren. Insbesondere für die Vogelarten die in dieser Arbeit schlecht erkannt wurde, wäre eine Identifizierung dieser Merkmale interessant.

Anhang

A Literaturverzeichnis

B Abbildungsverzeichnis

C Tabellenverzeichnis

A Literaturverzeichnis

[ADM96] Anderson, Sven E. ; Dave, Amish S. ; Margoliash, Daniel: Template-based automatic recognition of birdsong syllables from continuous recordings. In: *Journal of the Acoustical Society of America* 100 (1996), August, Nr. 2, S. 1209–1219

[BL03] Baker, Myron C. ; Logue, David M.: Population Differentiation in a Complex Bird Sound: A Comparison of Three Bioacoustical Analysis Procedures. In: *Ethology* (2003)

[BRF09] Briggs, Forrest ; Raich, Raviv ; Fern, Xiaoli Z.: Audio Classification of Bird Species: A Statistical Manifold Approach. In: 0010, Wei W. (Hrsg.) ; Kargupta, Hillol (Hrsg.) ; Ranka, Sanjay (Hrsg.) ; Yu, Philip S. (Hrsg.) ; Wu, Xindong (Hrsg.): *ICDM*, IEEE Computer Society, 2009. – ISBN 978–0–7695–3895–2, 51-60

[CM06] Chen, Z. ; Maher, R. C.: Semi-automatic classification of bird vocalizations using spectral peak tracks. In: *J Acoust Soc Am* 120 (2006), November, Nr. 5 Pt 1, 2974–2984. `http://view.ncbi.nlm.nih.gov/pubmed/17139754`. – ISSN 0001–4966

[CS95] Catchpole, C K. ; Slater, P J B.: *Bird Song - Biological Themes and Variations*. Cambridge University Press, 1995. – 7ff. S.

[DFS99] Deecke, V. B. ; Ford, J. K. B. ; Spong, P.: Quantifying complex patterns of bioacoustic variation: Use of a neural network to compare killer whale (*Orcinus orca*) dialects. In: *Journal of the Acoustical Society of America* 105 (1999), Nr. 4, S. 2499–2507

[Fag04] Fagerlund, Seppo: *Automatic recognition of bird species by their sounds*, Helsinki University of technology, Diss., 2004

[Fag07] Fagerlund, Seppo: Bird Species Recognition Using Support Vector Machines. In: *EURASIP J. Appl. Signal Process.* 2007 (2007), Januar, Nr. 1, 64–64. `http://dx.doi.org/10.1155/2007/38637`. – DOI 10.1155/2007/38637. – ISSN 1110–8657

[Här03] Härmä, Aki: Automatic identification of bird species based on sinusoidal modeling of syllables. In: *Proceedings of the 2003 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '03)* Bd. 5, 2003, S. 545–548

[KK80] Krebs, J. R. ; Kroodsma, D. E.: Repertoires and geographical variation in bird song. In: *Advances in the Study of Behavior* 11 (1980), S. 143–177

[KM98] Kogan, Joseph A. ; Margoliash, Daniel: Automated recognition of bird song elements from continuous recordings using dynamic time warping and hidden Markov models: A comparative study. In: *Journal of the Acoustical Society of America* 103 (1998), April, Nr. 4, S. 2185–2196

[LJKK11] Lopes, Marcelo T. ; Junior, Carlos Nascimento S. ; Koerich, Alessandro L. ; Kaestner, Celso Antonio A.: Feature set comparison for automatic bird species identification. In: *SMC*, IEEE, 2011. – ISBN 978–1–4577–0652–3, 965-970

[LLH06] Lee, Chang-Hsing ; Lee, Yeuan-Kuen ; Huang, Ren-Zhuang: Automatic Recognition of Bird Songs Using Cepstral Coefficients. In: *Journal of Information Technology and Applications* 1 (2006), Nr. 1, S. 17–23

[Log00] Logan, Beth: Mel Frequency Cepstral Coefficients for Music Modeling. In: *In International Symposium on Music Information Retrieval*, 2000

[LSDM01] Li, Dongge ; Sethi, Ishwar K. ; Dimitrova, Nevenka ; McGee, Tom: Classification of general audio data for content-based retrieval. In: *Pattern Recognition Letters* 22 (2001), Nr. 5, 533 - 544. `http://dx.doi.org/http://dx.doi.org/10.1016/S0167-8655(00)00119-7`. – DOI http://dx.doi.org/10.1016/S0167–8655(00)00119–7. – ISSN 0167–8655. – Image/Video Indexing and Retrieval

[MC97] Mcilraith, A. L. ; Card, H. C.: Birdsong recognition using backpropagation and multivariate statistics. In: *Signal Processing, IEEE Transactions on [see also Acoustics, Speech, and Signal Processing, IEEE Transactions on]* 45 (1997), Nr. 11, 2740–2748. `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=650100`

[MM05] Mierswa, Ingo ; Morik, Katharina: Automatic feature extraction for classifying audio data. In: *Machine Learning Journal* 58 (2005), S. 127–149

[PR98] Phelps, S. M. ; Ryan, M. J.: Neural networks predict response biases of female túngara frogs. In: *Proceedings. Biological sciences / The Royal Society* 265 (1998), Februar, Nr. 1393, 279–285. `http://dx.doi.org/10.1098/rspb.1998.0293`. – DOI 10.1098/rspb.1998.0293. – ISSN 0962–8452

[PT14] Percival, Graham ; Tzanetakis, George: *Marsyas User Manual*, 2014. http://marsyas.info/assets/docs/manual/marsyas-user.pdf

[Sha49] Shannon, C. E.: Communication in the Presence of Noise. In: *Proc. Institute of Radio Engineers* 37 (1949), Nr. 1, S. 10–21

[SHF06] Somervuo, Panu ; Härmä, Aki ; Fagerlund, S.: Parametric Representations of Bird Sounds for Automatic Species Recognition. In: *IEEE Transactions on Audio, Speech & Language Processing* 14 (2006), Nr. 6, 2252-2263. http://dblp.uni-trier.de/db/journals/taslp/taslp14.html#SomervuoHF06

[STT07] Selin, Arja ; Turunen, Jari J. ; Tanttu, Juha T.: Wavelets in Recognition of Bird Sounds. In: *EURASIP J. Adv. Sig. Proc.* 2007 (2007). http://dblp.uni-trier.de/db/journals/ejasp/ejasp2007.html#SelinTT07

[TC99] Tzanetakis, G. ; Cook, P.: Multifeature audio segmentation for browsing and annotation. In: *Applications of Signal Processing to Audio and Acoustics, 1999 IEEE Workshop on* (1999), 103–106. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=810860

[TC00] Tzanetakis, George ; Cook, Perry: *MARSYAS: A framework for audio analysis.* 2000

[TC02] Tzanetakis, G. ; Cook, P.: Musical genre classification of audio signals. In: *IEEE Transactions on Speech and Audio Processing* 10 (2002), Juli, Nr. 5, 293–302. http://dx.doi.org/10.1109/tsa.2002.800560. – DOI 10.1109/tsa.2002.800560. – ISSN 1063–6676

[Tza09] Tzanetakis, George: Marsyas submissions to MIREX 2009. In: *In: MIREX*, 2009

[VEVT06] Vilches, Erika ; Escobar, Ivan A. ; Vallejo, Edgar E. ; Taylor, Charles E.: Data Mining Applied to Acoustic Bird Species Recognition. In: *ICPR (3)*, IEEE Computer Society, 2006. – ISBN 0–7695–2521–0, 400-403

[Wie13] Wieners, Jan G.: *Vorlesungsfolien Audio der HKI Universität Köln.* http://www.hki.uni-koeln.de/sites/all/files/courses/839/ton.pdf. Version: 2013

[Wit83] Witkin, Andrew P.: Scale-space Filtering. In: *Proceedings of the Eighth International Joint Conference on Artificial Intelligence - Volume 2.* San Francisco, CA, USA : Morgan Kaufmann Publishers Inc., 1983 (IJCAI'83), 1019–1022

B Abbildungsverzeichnis

C Tabellenverzeichnis