

Ein quantitatives Tonmodell für Ibibio

Entwicklung eines Prädiktionsmoduls
für das BOSS-Sprachsynthesystem

Magisterarbeit
zur Erlangung des Grades eines
Magister Artium M.A.

vorgelegt
der
Philosophischen Fakultät
der Rheinischen Friedrich-Wilhelms-Universität
zu Bonn

von
Arne Bachmann
aus
Köln

An Eides statt versichere ich, dass die Arbeit

Ein quantitatives Tonmodell für Ibibio

Entwicklung eines Prädiktionsmoduls für das BOSS-Sprachsynthesystem

von mir selbst und ohne jede unerlaubte Hilfe angefertigt wurde, dass sie noch keiner anderen Stelle zur Prüfung vorgelegen hat und dass sie weder ganz, noch im Auszug veröffentlicht worden ist. Die Stellen der Arbeit - einschließlich Tabellen, Karten, Abbildungen usw. -, die anderen Werken dem Wortlaut oder dem Sinn nach entnommen sind, habe ich in jedem einzelnen Fall als Entlehnung kenntlich gemacht.

Ich danke

Meinen Eltern

Stefan Breuer

und meinen Rechtschreib-Korrektoren:

Eva Lasarczyk,

Verena Pyschny

und

Claudia Heimbach.

Erstellt am 22. Mai 2007, 13:05 Uhr.

Inhaltsverzeichnis

| | | |
|----------|---|-----------|
| 1 | Einführung | 1 |
| 1.1 | Ziel der Arbeit | 1 |
| 1.2 | Konkatenative Sprachsynthese | 2 |
| 1.3 | Tonsprachen | 4 |
| 1.3.1 | Das Ibibio | 5 |
| 1.3.2 | Deklination vs. Downdrift | 6 |
| 2 | Intonationsmodelle | 8 |
| 2.1 | Beschreibung von Intonationskonturen | 8 |
| 2.1.1 | Phonetisch vs. Phonologisch | 9 |
| 2.1.2 | Symbolisch vs. Parametrisch | 10 |
| 2.1.3 | Superposition vs. Tonsequenzen | 12 |
| 2.1.4 | Zielpunkte vs. Bewegungen | 14 |
| 2.1.5 | Weitere Betrachtungen | 16 |
| 2.1.6 | Ausgewählte Parametrische Modelle | 18 |
| 2.2 | Approximations-Algorithmen | 21 |
| 2.3 | Zusammenfassung | 21 |
| 3 | Prädiktionsmethoden | 23 |
| 3.1 | Maschinelles Lernen | 23 |
| 3.1.1 | Regelbasiert vs. Datenbasiert | 23 |
| 3.1.2 | Überwachtes Lernen vs. Nicht überwachtes Lernen | 25 |
| 3.1.3 | Ausgewählte datenbasierte Lernmethoden | 25 |
| 3.2 | Prädiktionsmerkmale | 30 |
| 3.3 | Zusammenfassung | 32 |
| 4 | Entwurf | 34 |
| 4.1 | Syntheseebenen | 34 |
| 4.2 | F ₀ -Modellierung | 35 |
| 4.2.1 | Polynom-Regression | 37 |
| 4.2.2 | Vektorquantisierung | 41 |
| 4.3 | Prädiktionsmethode | 45 |
| 4.4 | Einheitenauswahl | 49 |
| 4.4.1 | Vorauswahl | 49 |

| | | |
|----------|---|-----------|
| 4.4.2 | Einheitenauswahl | 50 |
| 4.4.3 | Verkettungskosten | 51 |
| 4.4.4 | Einheitenkosten | 51 |
| 4.5 | Behandlung von Dauer-Phänomenen | 53 |
| 4.6 | Kontextklassen | 54 |
| 5 | Implementation | 55 |
| 5.1 | Das Korpus | 56 |
| 5.2 | XML-Format | 57 |
| 5.2.1 | Vereinbarungen für den Client | 57 |
| 5.2.2 | Vereinbarungen für den Server | 58 |
| 5.3 | Quelltext | 60 |
| 5.3.1 | CART-Reader | 60 |
| 5.3.2 | Intonationsmodul | 60 |
| 5.3.3 | Einheitenauswahlmodul | 62 |
| 5.3.4 | Signalmanipulation | 62 |
| 6 | Evaluation | 64 |
| 6.1 | Quantitative Evaluation | 64 |
| 6.1.1 | Die Vektorquantisierung | 64 |
| 6.1.2 | CART-Prädiktion | 65 |
| 6.1.3 | Maße für die Grundfrequenzdifferenz | 69 |
| 6.1.4 | Stimulierzeugung | 70 |
| 6.2 | Qualitative Evaluation | 71 |
| 7 | Schluss | 73 |
| A | BOSS-Konfigurationsdatei | 75 |
| B | Das Korpus | 78 |
| B.1 | Inhalt | 78 |
| B.1.1 | Sätze des Korpus | 78 |
| B.1.2 | Testsätze | 80 |
| B.2 | Synthesebeispiele | 81 |
| C | Codebooks | 82 |
| C.1 | Das Tonschablonen-Codebook | 82 |
| C.1.1 | Codevektoren | 82 |
| C.1.2 | Silbenkonturen | 84 |
| C.2 | Das Tonschablonen-Klassen-Codebook | 86 |
| C.2.1 | Codevektoren | 86 |
| C.2.2 | Silbenklassen-Konturen | 87 |
| C.3 | Der Tonschablonen-Entscheidungsbaum | 87 |

| | |
|--------------------------------------|-----------|
| D Tools | 91 |
| D.1 Korpustools | 91 |
| D.2 CART-Training | 92 |
| D.3 Datenbankerstellung | 94 |
| D.4 Grafische Aufarbeitung | 96 |
| Literaturverzeichnis | 97 |

Tabellenverzeichnis

| | | |
|-----|---|----|
| 1.1 | Vokale des Ibibio nach Urua (2004) | 5 |
| 2.1 | Klassifikation von Intonationsbeschreibungsmodellen | 15 |
| 3.1 | F ₀ -Prädiktionsmerkmale in der Literatur | 32 |
| 4.1 | Messwertabweichungen der Grundfrequenzstilisierung bei unterschiedli- chen Skalen und Polynomgraden | 37 |
| 4.2 | Messwertabweichungen der Grundfrequenzstilisierung bei unterschiedli- chen Interpolationsmethoden | 38 |
| 4.3 | Vergleich der Vektorquantisierung verschiedener Codebookgrößen und Polynomgrade bei Interpolation aller stimmlosen Silbenabschnitte. An- gabe von Distortion (dist.) und Signal-zu-Rausch-Energieverhältnis (SNR). 42 | |
| 4.4 | Vergleich verschiedener Interpolationsgrade und Codebookgrößen bei bi- quadratischer Polynomisierung | 44 |
| 4.5 | Genutzte Prädiktionsmerkmale für das Ibibio-Intonationsmodul. T - Nut- zung im Tonschablonen-CART, D - Nutzung im Dauer-CART | 48 |
| 4.6 | Konfiguration der Einheitenvorauswahl. | 50 |
| 4.7 | Gewichtungen (Zähler) und Normalisierungen (Nenner) der Kostenterme. 53 | |
| 4.8 | Phone des Ibibio mit ihren Artikulationsorten | 54 |
| 5.1 | Attribute des Transfer-XML-Formats | 58 |
| 5.2 | Attribute des Korpus-XML-Formats | 59 |
| 5.3 | Attribute des internen Server-XML-Formats | 60 |
| 6.1 | Ergebnisse des CART-Trainings für Dauer- und Tonschablonenprädikti- on dreier Trainingsmengen | 67 |
| 6.2 | Codevektorauswahl des Entscheidungsbaums nach <code>sylphrase</code> | 68 |
| C.1 | Codebook der Intonationschablonen | 83 |
| C.2 | Codebook der Intonationschablonen-Klassen | 86 |

1 Einführung

1.1 Ziel der Arbeit

Diese Arbeit entstand im Rahmen einer Zusammenarbeit mit Frau Prof. Eno-Abasi Urua, Moses Ekpenyong (University of Uyo), Prof. Dafydd Gibbon (Universität Bielefeld), Prof. Wolfgang Hess und Stefan Breuer M.A. (Universität Bonn) zur Erstellung einer Sprachsynthese für das Ibibio. In dieser Arbeit wird das *Bonn Open Synthesis System* BOSS um ein Intonationsmodul für das Ibibio erweitert. Zu diesem Zwecke wird ein Prädiktionsmodul in C++ entwickelt und an die spezifischen Anforderungen der Einheiten Auswahl in einer konkatenativen Sprachsynthese angepasst. Zur Kommunikation innerhalb des Projektes wurde ein Wiki-System (<http://wiki.org>, 2006-08-16) auf einem Server des Instituts für Kommunikationsforschung und Phonetik (IKP¹) eingerichtet. Die ersten Kapitel dieser Arbeit geben einen Überblick über den Untersuchungsgegenstand, bisherige Forschungsergebnisse und meine Herangehensweise. Die restlichen Kapitel beschäftigen sich mit dem Entwurf, der Implementation und der Auswertung des Ibibio-Intonationsmoduls für BOSS:

- Kapitel 1 führt in grundlegende Begriffe und das Thema dieser Arbeit ein,
- Kapitel 2 zeigt verschiedene Ansätze zur Intonationsmodellierung,
- Kapitel 3 ist eine Einführung in maschinelle Lernverfahren und deren Applikation auf die Sprachsynthese,
- In Kapitel 4 wird der Entwurf eines Systems für die Intonation des Ibibio vorgestellt,
- In Kapitel 5 zeige ich Schritte zur Implementation des BOSS-Moduls,

¹<http://ikp.uni-bonn.de> (2006-08-16)

- In Kapitel 6 wird die Evaluation vorgestellt,
- Das Schlusswort schließt sich in Kapitel 7 an

Hinweise zur Formatierung

Dieses Dokument wurde vollständig mit dem Textverarbeitungssystem L^AT_EX 2_ε gesetzt; dazu wurde die Dokumentenklasse `scrbook` des Koma-Paketes genutzt (<http://www.komascript.de>, 2006-08-16).

Englische und lateinische Fremdwörter werden bei Erstnennung durch Schrägstellung (*slanted text*) gekennzeichnet; gibt es keine übliche deutsche Übersetzung, so werden diese Begriffe nachfolgend in normaler Schrift großgeschrieben. Spezielle Symbole, Parameter und Merkmale werden durch nichtproportionale Schrift angegeben (**typewriter font**). Einzelne phonetische und andere kategoriale Symbole werden in Fettschrift formatiert (**bold face**). Zahlen werden durchgängig mit Dezimalpunkt statt Komma und ohne Tausendermarkierung angegeben.

1.2 Konkatenative Sprachsynthese

Die Sprachsynthese im *Bonn Open Synthesis System* hat zur Aufgabe, einen elektronisch verfügbaren geschriebenen Text in einen lautsprachlichen Text umzuwandeln (*text-to-speech*, *TTS*). Somit handelt es sich um einen Vorleseautomaten, der dann gebraucht werden kann, wenn jemand aus unterschiedlichen Gründen nicht in der Lage ist, etwas abzulesen. Dies kann durch Behinderung (Blindheit), soziale Gründe (Analphabetismus) oder andere Umstände, bei denen die Augen sich bereits auf andere Dinge konzentrieren müssen (im Autoverkehr, in der Lagerverwaltung) oder der vorgelesene Text nicht greifbar oder nicht darstellbar ist (am Telefon), verursacht sein.

BOSS realisiert eine konkatenative Sprachsynthese. Dies bedeutet, dass Ausschnitte natürlichsprachlicher Aufnahmen neu verkettet (konkateniert) werden, um den eingegebenen Text zu erzeugen. Konkatenative Sprachsynthese hat in den letzten Jahren durch steigende Leistung der Rechner und weithin frei verfügbare Korpora die meisten Fortschritte gemacht und übertrifft alternative Ansätze (wie die artikulatorische Sprachsynthese und die Formantsynthese) besonders durch ihre Natürlichkeit. Dies liegt an der weitgehenden Erhaltung segmentaler (spektraler) Eigenschaften und der

durch große Korpora herbeigeführten großen Auswahl an prosodischen Realisierungen segmental gleichartiger Einheiten.

Der grobe Ablauf in einem Sprachsynthesystem wie BOSS ist wie folgt: Der eingegebene Text wird in einer Vorverarbeitung orthografisch vervollständigt (beispielsweise werden Zahlen ausbuchstabiert) sowie von überflüssigen Zeichen befreit (Satzzeichen werden nicht mitgesprochen). Eine linguistische Vorverarbeitung untersucht den Text auf linguistische Einheiten wie Sätze, Phrasen, Wörter und Silben und baut aufgrund dieser Information eine interne Repräsentation des Textes auf (einen XML²-Baum). Alle Einheiten werden in eine maschinenlesbare phonetische Lautschrift transkribiert (*grapheme-to-phoneme conversion G2P*). Nun werden die suprasegmentalen Eigenschaften für die Äußerung bestimmt (wie Grundfrequenzverlauf, Dauer und Intensität der Phone). Aufgrund dieser Informationen werden alle passenden konkatenierbaren Einheiten aus dem Korpus herausgesucht (*candidate selection*) und deren optimale Auswahl bestimmt (*unit selection*). Die ausgewählten Einheiten werden im letzten Schritt signaltechnisch verkettet, um sie dann per Lautsprecher ausgeben, über ein Netzwerk verschicken oder auf einem Datenträger abspeichern zu können.

BOSS

Geschichte

Das BOSS-System ging aus dem vom Bundesministerium für Bildung, Wissenschaft, Forschung und Technologie (BMBF) geförderten Verbundforschungsprojekt *Verbmobil* mit Teilnehmern aus Unternehmen, Universitäten und Forschungszentren hervor (Wahlster, 1997). Die Entwicklung an BOSS begann im Dezember 1998 unter der Leitung von Karlheinz Stöber. Im Laufe der Zeit wurde das gesamte System neu implementiert und mit zeitgemäßen, frei verfügbaren und technisch optimierten Datenstrukturen versehen (BOSS II ab dem Jahr 2000). BOSS 3 (seit Juli 2004) wurde im Dezember 2005 im Internet unter <http://sourceforge.net/projects/boss-synth> (2006-08-16) als *open source*³-Projekt veröffentlicht.

²eXtensible Markup Language, <http://www.w3.org/XML/>, 2006-08-16

³<http://www.opensource.org>, 2006-08-16

Charakteristika

BOSS ist weniger eine fertige Applikation, als vielmehr eine Forschungsplattform für die Sprachsynthese, ähnlich anderer Syntheseprojekte wie MBROLA (Dutoit et al., 1996), dem Festival System (Taylor et al., 1998; Black und Taylor, 1997b), dem Dresdener DRESS-System (Hoffmann et al., 1999) oder dem MARY TTS System (Schröder und Trouvain, 2003).

BOSS stellt eine Client-Server-Architektur zur Verfügung, wodurch eine Arbeitsteilung zwischen einerseits räumlich getrenntem spezialisierten Clienten und leistungstarkem Server auf der anderen Seite möglich ist. Darüber hinaus trennt BOSS relativ sauber zwischen Daten und Algorithmen, wodurch die Forschungsarbeit erleichtert wird. Entwickelt wurden bisher Korpora für das Holländische (2001), das Polnische (seit 2003), das britische Englisch, das Deutsche und sogar ein Korpus mit der Stimme Marcel Reich-Ranickis.

1.3 Tonsprachen

Die meisten Forschungsarbeiten haben sich bisher auf Akzentsprachen und deren Modellierung von Intonation bezogen. In Tonsprachen wie z. B. dem Chinesischen spricht man allerdings nicht von Intonation sondern von Tönen (*tone*), wenn man sich auf eine niedrigere Ebene unterhalb der Äußerung bezieht:

”Pitch functionality at the phonological and morphological ranks (which determine word prosody) is usually referred to as tone, and at the syntactic, text and dialogue levels (which determine discourse prosody) as intonation.”
(Gibbon et al., 2006)

”A tone language may be defined as a language having lexically significant, contrastive, but relative pitch on each syllable.” (Pike, 1948, S. 3)

Für afrikanische Sprachen wie dem Ibibio kommen evtl. zur Modellierung der lexikalischen Toneme (welche semantische Minimalpaare durch ihren Tonunterschied bilden) noch morphemische Töne hinzu (bedeutungstragende Tonmuster oder -schablonen):

”[...] the tone system of Ibibio is better classified as morphophonemic and morphotactic tone system than simply as a phonemic ’lexical tone’ system”
(Gibbon et al., 2006, 6)

1.3.1 Das Ibibio

Ibibio ist eine Niger-Congo-Sprache, die hauptsächlich im Akwa Ibom State und im Cross River State im südöstlichen Teil Nigerias gesprochen wird (Urua, 2001). Zur Gruppe der Niger-Congo-Sprachen gehören über 1500 einzelne lebende Sprachen, weshalb eine weitere Sprachsubklassifikation sinnvoll ist (hier nach Gordon 2005): Ibibio gehört zusammen mit Anaang, Efik und Ukwa (in immer feinerer Klassifikation) zur Gruppe der → Niger-Congo-Sprachen (1514) → Atlantic-Congo (1418) → Volta-Congo (1344) → Benue-Congo (961) → Cross River (67) → Lower Cross (23) → Obolo (23) und → Efik (4).

In Nigeria leben etwa 135 Mio. Menschen auf 923.768 km² und es gibt 510 lebende Sprachen (Gordon, 2005). Ibibio ist die hauptsächliche Handelssprache von Akwa Ibom State und wird dort beispielsweise an den Universitäten benutzt und in den Schulen gelehrt. Ibibio wird seit einigen Jahrzehnten mit lateinischen Buchstaben geschrieben und wird im Fernsehen und Radio übertragen. Töne werden nicht mitgeschrieben sondern können von Muttersprachlern aus dem Kontext abgeleitet werden. Die Anzahl der Ibibiosprecher wird auf 2-4 Millionen geschätzt (Gibbon et al. 2006; Essien 1991 nach Urua 2004). Es gibt einige Dialekte des Ibibio, deren Isoglossen sich an Clangrenzen befinden, Urua (2004) nennt Ibiono, Uruan, Uyo, Iman, Ikot Abasi/Mkpat Enin und Ikono; Gordon (2005) nennt darüber hinaus Enyong, Central Ibibio, Itak und Nsit. Die im Korpus dieser Arbeit untersuchte Varietät wird in der Gegend von Uruan und Uyo (der Hauptstadt von Akwa Ibom State) gesprochen.

Ibibio besitzt 13 phonemische Konsonanten **b(p),t,d,k,kp,m,n,ɲ,f,s,j,w** sowie 7 Vokale **i(i),u(u),e,ɛ,o(ə),ɔ,a** (Urua, 2004; priv. Komm.).

| | | |
|----|-----|----|
| i• | i•u | u• |
| e• | ə | o• |
| | ɛ | ɔ• |
| | a• | |

Tabelle 1.1: Vokale des Ibibio nach Urua (2004)

Dabei ist **kp** der labial-velare Plosiv, welcher die häufigste Doppelartikulation in den Sprachen der Welt darstellt und besonders in Afrika stark vertreten ist. Die Dauer von

Nasalen (einfach oder geminiert) ist in vokalischer Umgebung distinktiv: /jòmó/ "be noisy" vs. /jòmmó/ "boo at" (Urua, 2004). **p** tritt in komplementärer Distribution zu **b** auf und ist nicht kontrastiv. Zusätzlich zu den in Urua (2004) genannten Phonemen werden im Korpus noch weitere Symbole genutzt, siehe dazu Tabelle 4.8. Silben im Ibibio haben keine großen Konsonantenverbünde (größtenteils C, Cr, Cj, Cw im Anlaut) einen kurzen oder langen Vokal im Nukleus und nicht mehr als einen Konsonanten in der Coda. Zwischen zwei Vokalen werden die Konsonanten abgeschwächt und geben dem Ibibio seinen weichen Klang: [p,b] → [β]; t,d → [ɾ]; k → [ɽ],[w]⁴. Interessanter sind die tonalen Merkmale des Ibibio:

Ibibio ist eine Register⁵-Terassen⁶-Tonsprache mit drei kontrastierenden Tönen: Einem Hochtone (*level high tone*) **H**, Tieftone (*level low tone*) **L** und einem Downstep-Hochtone (*downstepped high tone*) **!H**, im folgenden mit **D** bezeichnet. Darüber hinaus gibt es zwei Konturtöne, welche sich als Kombination der Leveltöne bestimmen lassen: *Falling HL* und *rising LH* - im Folgenden **F** und **R** - welche nicht miteinander kontrastieren. Ibibio besitzt sowohl für L- als auch für H-Töne automatischen Downstep, was im folgenden Abschnitt näher erklärt wird. Die Abbildung 1.1 zeigt beispielhaft einen Ausschnitt aus Satz 53 des Korpus.

1.3.2 Deklination vs. Downdrift

Connell (2001) versucht die Unterschiede zwischen Deklination, *downdrift* und *downsteps*, die man alle unter dem Oberbegriff *downtrends* zusammenfassen kann, herauszuarbeiten. Deklination bezeichnet ein Phänomen, welches besonders (aber nicht ausschließlich) in Akzentsprachen vorzufinden ist: Im Verlaufe einer Äußerung sinkt die durchschnittliche Sprachgrundfrequenz. Messbar ist das an der *topline* und der *baseline*: Den begrenzenden (gedachten) Kurven, welche die F₀-Gipfel und Täler einhüllen. Nach Cohen et al. (1982, S. 270) kann man die *baseline* als die verlässlichere Bezugslinie annehmen. Die Deklination der *topline* scheint steiler zu verlaufen als die der *baseline*, aber auf der logarithmischen Skala lassen sich beide Deklinationen als nahezu parallele Linien darstellen (Cohen et al., 1982, S. 264).

⁴[ɽ] bezeichnet hier ein stimmhaftes uvulares Tippen, für das es kein Symbol im IPA gibt (IPA, 1999).
[w] ist der stimmhafte velare Approximant.

⁵"When a language has a small, restricted, number of pitch contrasts between level tonemes, theses contrastive levels are conveniently called REGISTERS." (Pike, 1948, S. 5)

⁶Der Begriff "Terasse" hängt mit der Tonsprachen-Eigenschaft des *downstep* zusammen und wird im Abschnitt 1.3.2 erklärt.

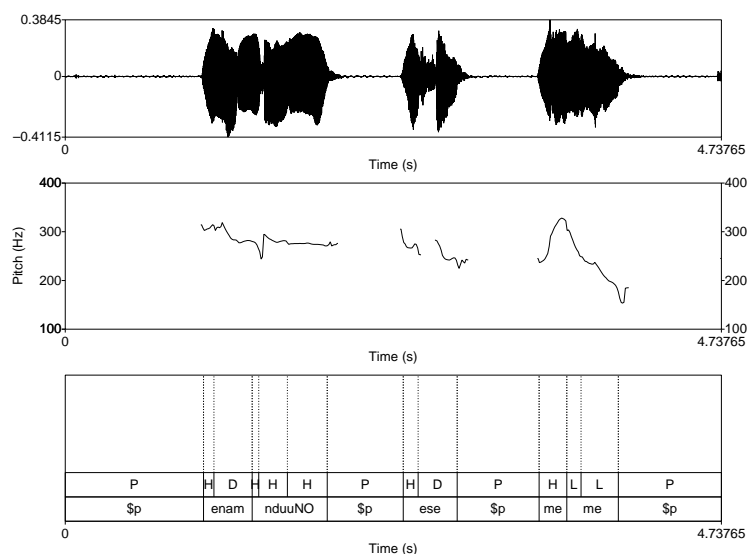


Abbildung 1.1: Oszillogramm, Grundfrequenz, Töne und Wörter von Satz 53.

Das Phänomen des Downdrift findet man hingegen in Tonsprachen. Connell (2001) unterscheidet zwischen automatischem und nicht-automatischem Downstep. *Automatic downstep* bezeichnet dabei das Absinken der Grundfrequenz des zweiten H-Tones in einer HLH-Sequenz (und auch das Absinken des zweiten L-Tones in einer LHL-Sequenz). Nicht-automatischer Downdrift findet in Sequenzen von H-Tönen statt, wenn ihnen entweder ein *underlying* oder *floating tone* zugrunde liegt oder dieser historisch verloren gegangen und zu einem *downstepped high tone* !H geworden ist. Hombert (1974) nach Connell (2001): Der Begriff Downdrift meint "the lowering of like tones (consecutive or not)". Sprachen, deren Register-Töne eine solche Downdrift besitzen, nennt man auch "terracing" (Terrassen-Sprachen).

Deklination und Downdrift sind jedoch nicht einfach zwei Ausprägungen eines Phänomens sondern können auch kombiniert werden: Eine Folge von H-Tönen kann in manchen Sprachen (z. B. im Hausa) eine Deklination aufweisen oder (wie im Ibibio) sich auf einer konstanten Tonhöhe bewegen, unabhängig von vorhandenem Downdrift. Bei der Deklination handelt es sich zwar um einen phonetisch-physiologischen Mechanismus, welcher aber willentlich vom Sprecher ausgesetzt werden kann.

2 Intonationsmodelle

2.1 Beschreibung von Intonationskonturen

Definition Den Begriff Intonation benutze ich in dieser Arbeit hauptsächlich für den Verlauf der Sprachgrundfrequenz innerhalb einer Äußerung. Gewöhnlich wird der Sprachgrundfrequenzverlauf in Hertz oder in Halbtönen gemessen und als Funktion im Zeit/Frequenz-Diagramm (evtl. dem Spektrogramm überlagert) dargestellt. Nach Hirst und Cristo (1998, S. 4) wird der Begriff "Intonation" auf physikalischer Ebene wie folgt benutzt:

"intonation is used to refer to variations of one or more acoustic parameters. Of these, fundamental frequency (F_0) is universally acknowledged to be the primary parameter."

Da Dauereffekte im zeitlichen Verlauf der Intonation bereits inbegriffen sind, beschränkt sich meine Begriffsnutzung auf die reine F_0 über die Zeit. Die Betrachtung der Sprachintensität wird oft vernachlässigt und ihr wird keine hohe Relevanz beigemessen, da das menschliche Gehör Intensitätsänderungen weitaus schlechter wahrnehmen kann als Frequenzänderungen (Zwicker, 1982). Hirst und Cristo (1998, S. 9) meinen zwar

"an adequate characterisation of tonal phenomena in Chinese needs to account for both pitch and intensity variations",

aber selbst wenn dies auch für das Ibibio gelten würde, wird eine Untersuchung der Intensitätsphänomene nicht Bestandteil dieser Arbeit sein.

Während die Prosodie auch lexikalische Eigenschaften wie Betonung, Töne und Quantität beinhalten kann, verstehe ich - dem Beispiel von Hirst und Cristo (1998) folgend - unter Intonation einzig die nicht-lexikalische "intonation proper", also die "eigentliche" Intonation, welche aus der Sprachgrundfrequenz über der Zeit besteht. Da ich mich

ausschließlich mit dieser auseinandersetzen werde, wird der Begriff Intonation auch nur in diesem so engen Rahmen genutzt werden.

Im Folgenden werde ich die drei wesentlichen Klassifizierungsmerkmale bestehender Intonationsmodelle vorstellen und einige in der Literatur angeführte Unterscheidungen beschreiben.

2.1.1 Phonetisch vs. Phonologisch

Bei der Entscheidung über die Wahl eines geeigneten Beschreibungsmodells wird man vor die Frage gestellt, ob man rein deskriptiv arbeitet oder einige Vorannahmen über die Intonation trifft. Die erste Herangehensweise versucht phonetisch zu beschreiben, wie sich der Grundfrequenzverlauf verhält, ohne diesem ein irgend geartetes linguistisches Wissen zu Grunde zu legen (z. B. was eine Silbe ist und wie sie aufgebaut ist). Die zweite Möglichkeit bezeichne ich als "phonologisch", da wir hier bereits generelle Annahmen über die Funktionsweise des Lautsystems einer Sprache machen und unsere Beobachtungen der intonatorischen Phänomene in unsere Beschreibung mit einbeziehen. Darunter fällt die Erkenntnis, dass es z. B. in europäischen Sprachen das Phänomen der Akzente gibt, welche sich im Grundfrequenzverlauf u. a. durch ihren Berg- oder Talcharakter und dessen zeitlicher Position zum Silbenkern auszeichnen. Louw und Barnard (2004) schreiben Folgendes zu phonologischen Beschreibungen:

"Such a representation is descriptive and discrete. It uses an inventory of abstract phonological categories, with each category having its own linguistic function."

Ein typisches phonologisches Beschreibungsmodell stellt der autosegmental-metrische Ansatz (AM: Ladd 1996) dar. Für Tonsprachen gilt die Beobachtung, dass verschiedene Tonhöhen bedeutungsunterscheidend sein können.

Man kann also sagen, dass man zwar den Grundfrequenzverlauf jeder beliebigen Äußerung irgendwie phonetisch-akustisch (z. B. als mathematische Funktion) beschreiben kann, dass für eine kompakte (i. S. v. funktional äquivalente) Beschreibung allerdings viele Feinheiten der F_0 -Bewegung nicht relevant sind. Um diese Redundanzen weglassen zu können benötigt man das phonologische Vorwissen. Hirst und Cristo (1998) nennen diese Unterscheidung "physikalisch" vs. "formal / linguistisch". Hirst et al. (2000) entwickeln ein 4-Ebenen-Modell für eine umfassende Theorie der Intonation.

Die vier Beschreibungsebenen sind tiefe Phonologie, Oberflächen-Phonologie, Phonetik und Oberflächen-Phonetik. Letztere unterliegt den physikalischen und akustischen *constraints* (Randbedingungen), die allen menschlichen Sprachen gemeinsam sind. Nach den Autoren betrachten die meisten existierenden Intonations-Modelle nur eine der vier Ebenen und vergeuden so mögliches Potential. Das weiter unten besprochene Fujisakimodell (Fujisaki und Nagashima, 1969) beispielsweise beschreibt die Oberflächen-Phonetik sehr akkurat und liefert eine Erklärung für Intonationsphänomene im physiologischen Bereich, die anderen Ebenen werden allerdings nicht mit in das Modell eingeschlossen. Besonders die Möglichkeit, die Informationstransformation zwischen den Ebenen in beide Richtungen modellieren zu können, stellt für die Autoren ein nützliches Kriterium in der phonetischen Wissenschaft dar.

2.1.2 Symbolisch vs. Parametrisch

Die zweite Dichotomie findet zwischen symbolischen und parametrischen Beschreibungen der Intonation statt.

Symbolische Beschreibungssysteme besitzen ein endliches (diskretes) Symbolinventar, mit dem eine qualitative Beschreibung der Intonation ermöglicht wird. Gewöhnlich streben diese Beschreibungssysteme nicht nach einer völlig exakten Datenbeschreibung sondern bieten sowohl dem Phonetiker als auch nicht phonetisch geschulten Menschen ein Werkzeug an, um Intonationskonturen mit einem einfach zu erlernenden System zu beschreiben. Dabei kann es sich einerseits um phonologische Beschreibungsmodelle wie dem *Tone sequence model TSM* (Pierrehumbert, 1980) und dem darauf basierenden Markierungssystem *Tone and Break Indices ToBI* (Silverman et al., 1992; Liberman und Pierrehumbert, 1984) handeln, welche sich eignen, die Akzent- und Grenztöne von Akzentsprachen zu beschreiben, aber auch um rein phonetisch motivierte symbolische Beschreibungssysteme wie dem *international transcription system for intonation INTSINT* (Estelle Campione, Daniel Hirst, 2000; Hirst und Cristo, 1998), mit dessen Hilfe sich sowohl relative als auch absolute Tonhöhen qualitativ beschreiben lassen. Das INTSINT-Modell kann i. Ggs. zu ToBI ohne weiteres sprachspezifisches Wissen auf alle Sprachen angewendet werden. Für phonologische symbolische Systeme muss es jeweils einzelne Sprachvarianten geben (z. B. G-ToBI für das Deutsche, E-ToBI für das Englische oder K-ToBI für das Koreanische):

”[the ToBI system] presupposes that the inventory of tonal patterns of the

language is already established”. (Hirst und Cristo, 1998, S. 14)

Auf der anderen Seite befinden sich die parametrischen Modelle, welche die Intonation in kontinuierliche quantitative Parameter überführen. Ein solches Modell kann erst einmal keinen linguistischen Anspruch erfüllen, da es rein deskriptiv arbeitet und keine Verallgemeinerungen trifft. Es gibt inzwischen eine große Vielfalt an parametrischen Beschreibungen. Dabei wird die zeitliche Bewegung der Sprachgrundfrequenz gewöhnlich in Kurvenabschnitte zerlegt und diese durch eine (perzeptiv möglichst äquivalente) mathematische Näherungsfunktion beschrieben; darunter fallen beispielsweise verschiedene Polynome (Möhler, 1998a), quadratische *splines* (Hirst, 1993; Mixdorff, 2000; Mouline et al., 2004), Bézierkurven dritten (Escudero et al., 2002) und vierten Grades (Agüero et al., 2004; Agüero und Bonafonte, 2005), kritisch gedämpfte Impuls- und Sprungfunktionen (Fujisaki und Nagashima 1969; Hirose et al. 1983; Fujisaki et al. 1984; Fujisaki 1988; Fujisaki et al. 1996; Fujisaki und Ohno 1996; Fujisaki 2004; Gu et al. 2004, Mixdorff 1998 und viele andere), das Tiltmodell (Dusterhoff und Black, 1997; Syrdal et al., 1998; Möhler, 1998b; Dusterhoff et al., 1999; Taylor, 2000) und einige zusammengesetzte Funktionen wie das *parametric intonation event*-Modell (Möhler, 1998b,a; Möhler und Conkie, 1998; Möhler, 2001; Syrdal et al., 1998) oder die maximumbasierte Beschreibung nach Heuft (1999), welche Grundfrequenzmaxima mit einer \cos^2 -Kurve stilisiert.

Die Schnittstelle zwischen symbolischen und parametrischen Systemen ergibt sich dort, wo aufgrund vorhandener parametrischer Stilisierungen ein minimales Symbolinventar entworfen werden soll oder - in anderer Richtung - aufgrund einer symbolischen Intonationsbeschreibung eine perzeptiv korrekte F_0 -Kontur synthetisiert werden soll.

Die erste Variante findet sich beispielsweise schon in der Beschreibung des IPO¹-Modells, welche die Intonation durch eine perzeptiv identische Approximation durch die kleinstmögliche Anzahl von Geradenstücken ersetzt und aufgrund dieser Stilisierung ein Inventar an *”perceptually relevant pitch movements”* und darauf aufsetzenden *intonational patterns* (’t Hart et al., 1990, S. 40, S. 59) herausarbeitet. Das Verfahren wird *”close-copy stylisation”* genannt.

Das umgekehrte Verfahren, Symbol-zu-Intonation, scheint viele Schwierigkeiten zu bereiten. Es gibt zwar ein handgeschriebenes Regelsystem zur Projektion von Labels auf eine konkrete F_0 -Kontur (Anderson et al., 1984) und auch statistisch trainierte Me-

¹*Instituut voor Perceptie Onderzoek*

thoden (Black und Hunt 1996), aber oft werden ToBI-Labels nur als eines von vielen Prädiktionsmerkmalen behandelt um über künstliche neuronale Netze (NN) oder Entscheidungsbäume die Werte eines parametrischen Modells vorherzusagen (s. z. B. für das Tilt-Modell Dusterhoff und Black 1997 oder für das PaIntE-Modell Möhler und Conkie 1998).

2.1.2.1 Diachrone Betrachtung

Diachron betrachtet entwickelten sich die Beschreibungsmodelle für die Grundfrequenzkontur von den linguistisch motivierten symbolischen Modellen zu den parametrischen, nicht zuletzt wegen der heute frei verfügbaren Audioanalyse-Programme und verfügbarer Rechenkapazität. Dabei gehen Erkenntnisse aus früheren phonologisch-symbolischen Modellen an anderer Stelle wieder in die parametrischen Modelle mit ein. Dazu gehört allein schon der Gedanke, dass eine Sprache eine endliche Zahl an relevanten Intonationsmustern besitzt. Selbst die potentiell unendlich große Zahl an Realisierungen ändert nichts daran, dass Menschen nur eine endliche Zahl funktional relevanter Bewegungsmuster differenzieren können, und dies wirkt sich schlussendlich auf die möglichen Modelle aus. Daher wurden auch einige Modelle entwickelt, welche explizit Intonations-„Schablonen“ (engl. *templates*) benutzen.

„Linguistisch orientiert wird ein solches Schema dann, wenn wenige, kategorial unterschiedliche Formen [...] angewendet werden und ein phonologisches Intonationsmodell dahinter steht.“ (Hess, 2004a, F. 22)

Zu diesen Schablonenmodellen gehört z. B. die „Vogelsilhouetten“-Stilisierung (Delattre et al. 1965²) oder die Akzentschablone nach Grønnum (1988, nach Hess 2004a). Und auch in der Vektorquantisierung (VQ) findet sich dieser Gedanke wieder.

2.1.3 Superposition vs. Tonsequenzen

Für parametrische Modelle gibt es noch die wichtige Dichotomie zwischen Superposition und Tonsequenz; spätestens mit dem Aufkommen von Fujisakis Intonationsbeschreibungmodell setzte sich auch vermehrt eine neue Art der Intonationsdekom-

²„A concrete image - that of a bird - will simplify our description [...]. The complete pitch pattern of continuation, at the end of a German sense group, invariably takes a shape which suggests the body outline of a bird singing, seen from profile with its tail on the left and its beak raised up on the right. The completion of its shape seems to depend on the degree of stress involved.“

position durch: Das Superpositionsmodell. In additiven (superponierenden) Modellen wird der Grundfrequenzverlauf nicht nur zeitlich in einzelne überlappungsfrei (oder leicht überlappend) zu modellierende Abschnitte zerlegt, sondern es wird auch noch eine "spektrale" Aufteilung der F_0 -Kontur in schnelle und langsame Bewegungselemente unternommen. Diese wurden dann entweder gewissen physiologischen Prozessen beigegeben (Translations- und Rotationsbewegung des Cricothyroidmuskels im Kehlkopf: Fujisaki 1988, 2004; vgl. "chest pulses und stress pulses" in der Rhythmusforschung bei Abercrombie 1967) oder mit linguistischen Entitäten in Verbindung gebracht (Phrasen, Akzentgruppen: Möbius 1993). Die letztere Auslegung der additiven Modelle besticht durch ihre Klarheit, da sie linguistische Information direkt auf akustische Form projizieren kann und dadurch eine starke Vereinfachung der nötigen Parameter ermöglicht. Im Fujisakimodell wird eine sprecher- und sprechstilabhängige Basisfrequenz F_{min} mit Phrasen- und Akzentkomponenten überlagert. Sie bestehen aus kritisch gedämpften Schwingungen. Es existieren auch andere superponierende Ansätze (z. B. Sakai 2004), aber Fujisakis hat in den letzten Jahren ohne Zweifel die meiste Aufmerksamkeit erhalten und wie kein anderes Modell die Erforschung der parametrischen Intonationsmodelle und einer großen Zahl an Softwaretools zur effizienten Bestimmung der Parameter vorangetrieben. Einen interessanten neueren Ansatz bieten van Santen et al. (2004) mit einem *general superpositional intonation model* und dem von ihnen vorgeschlagenen (*simplified*) *linear alignment model* (LAM). Das LAM besitzt eine ähnliche additive Struktur wie das Fujisakimodell und unterscheidet zwischen Phrasen-, Akzent- und Perturbationsebene. Das Modell ist aber (durch das *alignment*) weitaus stärker linguistisch motiviert und macht sehr spezifische Vorgaben, wie Akzente und Grenzen in den einzelnen Ebenen ausgerichtet werden.

"However, if one allows multiple components, then one necessarily faces the problem that there is no unique solution in the decomposition of a single f_0 time series into multiple components. Any such decomposition depends on a model of the speech process, and is only as good as the underlying model." (Kochanski und Shih, 2001b)

Im Gegensatz zu den superponierenden Modellen existieren die stückweise modellierenden Ansätze ohne überlagernde Komponenten. Ein leicht erkennbares Problem stellt die schwierige Verallgemeinerung ihrer Parameter dar. Beispiel: Zwei Akzente zu Beginn und gegen Ende einer Äußerung haben absolut gesehen sehr unterschiedliche F_0 -Werte,

aber ihre Form ist exakt gleich. Ohne Zugrundelegung einer Deklinationsbewegung (oder anderer Theorien) über die Äußerung lässt sich die Akzentbewegung nicht generalisieren. Daher werden stückweise Modellierungen oft mit einem Regelinventar kombiniert um zu brauchbaren Ergebnissen zu gelangen; z.B. in der F_0 -Stilisierung mit Registern (Ladd 1989 nach Möbius 1993, S. 63) bedarf es phonologischer Regeln um die unsinnige und inflationäre Anwendung von Downsteps zu verhindern. Auf S. 62 schreibt Möbius allerdings,

”daß die strenge dichotomische Klassifizierung von Intonationsmodellen in Tonsequenz-Modell einerseits und hierarchische Intonationsmodelle, die mehrere Komponenten einander superponieren, andererseits, artifiziell ist und sich aufgrund inhaltlicher und methodischer Kriterien nicht durchhalten läßt.”

Ein frühes Beispiel für ein additives Intonationsmodell liefert Grønnum (1988): Hier wird für kurze dänische Sätze eine satzmodusabhängige Phrasenkomponente mit einer Akzentschablone überlagert.

Einen einleuchtenden Vorteil eines hierarchischen Modells nennt Yu et al. (2002): Das Trainingskorpus für die datenintensiven maschinellen Lernverfahren darf hier relativ klein sein, da jede Ebene des additiven Modells einzeln trainiert werden kann.

Tabelle 2.1 zeigt noch einmal einen Überblick über die bisher getroffene Differenzierung der Beschreibungsmodelle.

2.1.4 Zielpunkte vs. Bewegungen

Eine weitere klassische Unterscheidung betrifft die Frage, ob man Intonationsbewegungen durch Beschreibung ihrer Zielpunkte oder ihrer Bewegungen charakterisiert. Diese Unterscheidung hat eine lange Tradition. Schon Pike (1945) beschrieb die Intonation des amerikanischen Englisch in vier Tonhöhen und auch Tonsprachen besitzen bis zu vier distinktive Tonhöhen:

”a LEVEL toneme³ is one in which, within the limits of perception, the pitch of a syllable does not rise or fall during its production. [...] When a

³Toneme bezeichnen nach Pike (1948, S. 4) - ähnlich den Phonemen - die kleinsten tonal und segmental kontrastierenden lexikalischen Einheiten.

| Art | Symbolisch | Parametrisch | |
|--------------|------------------------|--|---|
| Phonologisch | TSM+ToBI IPO KIM | LAM (van Santen et al., 2004) | |
| | | Additiv | Sequentiell |
| Phonetisch | INTSINT | Fujisaki et al. (1996) Sakai (2004) Grønnum (1988) | Schablonen Polynome Bézierkurven Splines Tilt-Modell PaIntE-Modell |

Tabelle 2.1: Klassifikation von Intonationsbeschreibungsmodellen

language has a small, restricted, number of pitch contrasts between level tonemes, these contrastive levels are conveniently called REGISTERS. The number of permitted registers in various languages seems to be limited to two, or three, or four;” (Pike, 1948, S. 5)

Pike macht aber auch klar, dass es sich hier nicht um absolute (Ziel-)Werte handelt, sondern nur syntagmatisch verschiedene Tonregister entstehen können:

”The pitch levels appear to be nearly or completely meaningless by themselves. It is the intonation contour as a whole which carries the meaning while the pitch levels contribute end points, beginning points, or direction-change points to the contours - and as such are basic building blocks which contribute to the contours and hence contribute to the meaning.” (Pike, 1945, S. 26)

”Tone languages have a major characteristic in common: it is the relative height of their tonemes, not their actual pitch, which is pertinent to their linguistic analysis. [...] The important feature is the relative height of a syllable in relation to preceding and following syllables.” (Pike, 1948, S. 4)

Die Kontroverse zwischen absoluten ”levels” (Tonstufen) und den ”configurations” (Konturen) wurde zugunsten der *configurations* entschieden (nach Möbius 1993, S. 53):

”Bolinger (1951) führt den Nachweis, daß die Tonstufen nicht phonologisch interpretiert werden dürfen. Die Tonhöhenbereiche der ”Phoneme” schließen sich nicht wechselseitig aus, sondern können sich überlappen.”

Pierrehumbert (1980) beschrieb Akzente durch das Vorhandensein von zwei Zieltönen (welche der Deklination überlagert sind). Durch Wechsel des Tonniveaus kann der Sprecher einen Akzent markieren. Adriaens (1991) erweiterte das System für das Deutsche auf vier Tonniveaus. Auch in Tonsprachen macht es Sinn, von Zieltönen zu sprechen: Wie z. B. Connell (2001) zeigt, erreichen H-Töne im Mambila nach einem L-Ton ihre Ziel-Tonhöhe erst einige Silben später. Auch im MoMel (*modélisation de melodie*: Hirst und Espesser 1993; Campione et al. 1997) kommen Zielpunkte zum Einsatz. Bei solchen parametrischen Verfahren gibt es allerdings auch oft einen Parameter für die Steilheit der Intonationskurve zum Zielpunkt hin und von diesem fort, durch den die Transition - d. h. die Bewegung - zwischen den Zielpunkten genauer charakterisiert wird.

Die andere Möglichkeit ist nun, nicht Töne als Diskriminationsmerkmal zu betrachten, sondern Tonbewegungen. Parallel zur Erkenntnis in der artikulatorischen Phonetik, dass nicht Stationarität der Artikulation sondern gerade die Transitionen zwischen den einzelnen Lauten die meiste Information trägt, kann man auch annehmen, dass nicht feste Tonhöhen und das exakte Erreichen eines F_0 -Zielpunktes relevant sind, sondern vielmehr die Steilheit und Form einer F_0 -Bewegung. Diese Beschreibung kann man auch auf Konturtöne von Tonsprachen anwenden. Es gibt solche, welche sich zwischen Tonniveaus bewegen und solche, welche nur ”fallend” oder ”steigend” sind und damit auch auf kein Tonniveau direkt passen. Als Beispiel der 2. (steigende) und 4. (fallende) Ton des Chinesischen:

”the high and low tones in Mandarin are referred to as static targets whereas the rise and fall tones are dynamic targets” (Sun, 2002a)

2.1.5 Weitere Betrachtungen

Unterspezifizierung vs. volle Spezifizierung Man kann die Intonationsmodelle danach unterscheiden, wie umfangreich sie die Grundfrequenz modellieren: Das Tilt-Modell (s. u.) definiert die Tonkontur von Akzenten, lässt aber die unakzentuierten Abschnitte dazwischen undefiniert; diese werden beispielsweise linear interpoliert. Andere Modelle besitzen eine volle Spezifikation, so dass an jeder (stimmhaften) Stelle

des Signals ein F_0 -Wert bestimmt werden kann (z. B. bei Fujisaki). Kochanski und Shih (2001b) schreiben dazu:

”The advantage of using an under-specified accent shape is that it allows sufficient distance between specified accent targets to allow a smooth f_0 transition, typically by way of interpolation. The drawback is that it ignores changes of shape between specified targets. On the other hand, a system with fully specified accents leaves little room to resolve conflicting targets. A simple concatenation of fully-specified accents will result in a pitch curve with unnatural jumps at the concatenation joints. Many systems, such as Fujisaki (1983, 1988), use filters to smooth out abrupt changes in f_0 . Alternatively, van Santen (1997, 2000) requires each accent to begin and end at zero to ensure smooth connections between accents.”

Intonationstypologie Typologisch lassen sich Sprachen in Akzent-, Ton und Tonakzentsprachen unterscheiden (*stress, tone and pitch accent languages*). Akzentsprachen kommt zu Gute, dass es bereits viele symbolische Beschreibungssysteme für Akzenttöne und Akzentbewegungen gibt. Auch die meisten parametrischen Modelle wurden gezielt für Akzentsprachen entwickelt und stilisieren die Intonation durch passende Funktionen (beispielsweise Gipfel an relevanten Positionen). Das erfolgreiche Fujisakimodell lässt sich allerdings auch für den Gebrauch in Tonsprachen erweitern (Fujisaki, 2004).

Tonheit Eine von den meisten Modellen ignorierte Tatsache ist, dass Grundfrequenz und wahrgenommene Tonhöhe nicht zwangsläufig übereinstimmen. Dies ist umso erstaunlicher, da andere Komponenten eines TTS-Systems durchaus gehörrichtige Normierungen vornehmen (z. B. die segmentalen MFCC-Parameter⁴ in der Einheitenwahl von BOSS). Das Fujisakimodell nimmt eine einfache Anpassung der Grundfrequenz an die Tonheit wahr, indem es die Intonation nur auf der logarithmischen Hertz-Skala modelliert. Auch das IPO-Modell ist ein perzeptives Modell der Intonation (’t Hart et al., 1990) und führt die Stilisierung durch Geradenstücke im Logarithmischen durch. Dadurch ist eine weitgehend annehmbare perzeptive Korrektur vorgenommen. Durch eine solche Transformation kann das Modell eine genauere Repräsentation der psychologischen Realität von Intonation ermöglichen. d’Alessandro (1995) erarbeitet ein

⁴Mel frequency cepstral coefficients

quantitatives Modell basierend auf tonaler Wahrnehmung.

Mikroprosodie Die Mikroprosodie wird im allgemeinen bei der Modellierung der Intonation ignoriert. Für die Parametrisierung des Grundfrequenzverlaufs bei der Modellbildung wird gewöhnlich eine Glättung oder Filterung durchgeführt, die viele mikroprosodische Auswirkungen von vornherein egalisiert. Die Natürlichkeit der Intonation nimmt somit schon in der Vorbereitung der Parametrisierung ab, ermöglicht allerdings auch eine starke Datenreduktion. Das rein regelbasierte Kieler Intonationsmodell (KIM: Kohler 1991a,b, 1997b,a) berücksichtigt hingegen explizit die mikroprosodischen Effekte, indem es spezielle kontextsensitive Regeln einführt: So wird beispielsweise die Grundfrequenz in stimmhaften Plosiven um 10 Hz vom prädictierten Wert gesenkt. Auch in der Einheiten Auswahl von BOSS kommen kontextsensitive Regeln zum Einsatz, welche die lautliche Umgebung (Nasal, Plosiv o. ä.) mit in die Einheitenkosten (*unit cost*) einberechnen.

2.1.6 Ausgewählte Parametrische Modelle

Da in dieser Arbeit ein parametrisches Modell entwickelt und implementiert werden soll, werde ich in diesem Abschnitt einige solche Modelle genauer untersuchen.

2.1.6.1 Fujisaki

Für das Fujisakimodell wurden inzwischen viele Hilfsprogramme entwickelt, und es ist bereits in der deutschen Version von BOSS integriert. Die Realisierung einer Grundfrequenzkontur durch Fujisaki-Kommandos sollte damit sehr leicht möglich sein, wenn man sowohl positive als auch negative Akzentkommandos zulässt. Das Modell wurde bereits erfolgreich für das Mandarin-Chinesische eingesetzt, welches tonal dem Ibibio ähnlich ist, außer dass es im Ibibio statt des gleichbleibenden Tones **S** (*steady*) einen Downstep-Ton **D** gibt (Mandarin, fünf Toneme: H,L,F,R,S; Ibibio, fünf Töne: H,L,F,R,D). Im Ibibio kann man nicht von fünf Tonemen sprechen, da R und F zusammengesetzte Töne und nicht kontrastiv sind. Jeder Ton ist mit einer Silbe verknüpft. Die Phrasenkomponente und die minimale Grundfrequenz F_{min} kann in Tonsprachen genau wie in Akzentsprachen realisiert werden. Statt der Akzentkomponenten kommen jedoch Tonkomponenten hinzu, welche eine additive Verknüpfung zweier aufeinander folgender Akzentkomponenten möglicherweise umgekehrten Vorzeichens darstellen.

Diese ermöglichen die Realisierung von Konturtönen. Betrachtet man im Fujisakimodell wieder die physiologische Begründung für die Nutzung von kritisch gedämpften Schwingungen, so mögen die zwei Akzentkomponenten den gegensätzlich beschleunigten Muskelbewegungen im Kehlkopf entsprechen. Sollte sich jedoch herausstellen, dass mit Hilfe einer viel einfacheren Parametrisierung eine perzeptiv gleichwertige Stilisierung der Grundfrequenz gelingt, so darf man sich fragen, wie viel physikalische Realität hinter Fujisakis Modell steckt. Hinzu kommt die Tatsache, dass die Approximation der Parameter in Fujisakis Modell keine geschlossene analytische Form hat und sie nur mit hohem Rechenaufwand näherungsweise bestimmt werden können. Außerdem lassen sich mit unterschiedlichen Parameterkombinationen perzeptiv äquivalente Approximationen der Intonation herstellen, doch nicht jede Parameterkombination lässt sich linguistisch begründen (Möbius, 1993, S. 82). Der Approximationsalgorithmus benötigt daher linguistische Randbedingungen (*constraints*), um nicht mit absurden Parametern, die weder linguistisch noch physiologisch zu erklären sind, das zu erreichen, was mit anderen Werten ebenso leicht möglich wäre. So könnte es beispielsweise passieren, dass jeder Akzent mit einer Phrasenkomponente erzeugt wird oder dass mehr Akzentkommandos in der Phrase sind als es Akzente gibt. Wir müssen also *a posteriori* Beschränkungen auferlegen, um eine linguistische Begründung für das genutzte Modell liefern zu können, was keinen besonders erstrebenswerten Zustand darstellt.

2.1.6.2 Tilt-Modell

Das Tiltmodell (Dusterhoff und Black, 1997; Taylor, 1998) und seine Variante in Möhler (1998a) stellen eine kompakte parametrische Beschreibung für annotierte Intonationsereignisse dar (markiert werden Akzenttöne **a** (*accent*), Grenztöne **b** (*boundary*) und Verbindungstöne **c** (*connection*) sowie Pausen **sil** (*silence*). Jede Tiltkurve kann durch ein 5-Tupel (F_0 , dur, amp, ppos, tilt) beschrieben werden, wobei es die folgenden Parameter gibt: F_0 = *start-frequency* [Hz], dur= *duration* [s], amp= *amplitude* [Hz], ppos= *peak position* [s] und tilt [-1...+1]. Das Tiltmodell besticht durch einen klar gegliederten Aufbau mit einfach nachvollziehbaren Parametern. Jedoch sind die mit diesem Modell erzeugbaren Intonationsereignisse eingeschränkt:

”It is based on the general findings of the Tone-Sequence Model. Hence, the function should be able to model F_0 movements that are either rising or falling or of the rising-falling type.” (Möhler, 1998a)

Fallend-steigende Konturen aus Tonsprachen sind mit einem Tilt-Ereignis nicht beschreibbar. Damit verbietet das Modell eine Relation zwischen phonologischen Ausprägungen fallendsteigender Art, wie z. B. Tal-Akzente oder auch Intonationskonturen in Tonsprachen. Darüber hinaus zeigt eine subjektive Evaluation in Syrdal et al. (1998) die Unterlegenheit des Tiltmodells gegenüber dem PaIntE-Modell (s. folgender Abschnitt) im Bezug auf die Hörerurteile.

Ein Vorteil ist, dass Tiltereignisse nur wenige Freiheitsgrade besitzen, welche approximiert werden müssen. Mit der Prädiktion von Tiltereignissen beschäftigen sich Dusterhoff et al. (1999) und Taylor (2000). Das Modell hat den Vorteil, bereits mit wenigen Markierungen (*labels*), welche selbst dem einfachsten TTS System vorliegen, auf Tiltereignisse trainierbar zu sein. Darüber hinaus vertritt Taylor die Meinung, dass das Tiltmodell eine universellere Beschreibungsmöglichkeit für phrasenweise ansteigende Akzente (z. B. *upsteps*) bietet als das Fujisakimodell und dabei dennoch eine linguistische Erklärung bietet - welche durch willkürliche Verkettung von Fujisakikommandos nicht gegeben wäre - da keine Phrasenkommandos zur Simulation von steigenden Akzenten benutzt werden brauchen.

Das Tilt-Modell eignet sich gut zur Modellierung der von dem TSM vorgeschlagenen F_0 -Bewegungen. F_0 -Täler müssten aber durch die Kombination von zwei Tilt-Ereignissen simuliert werden und hebeln damit die gute Parametrisierbarkeit aus, da silbenorientierte Toneme prädiiziert werden sollen und die einzelnen Silben nicht mehr direkt mit den Parametern eines Tilt-Ereignisses assoziiert werden könnten.

2.1.6.3 PaIntE

Im *PA*rametric *INT*onation *Event* PaIntE-Modell (Möhler, 1998b,a; Möhler und Conkie, 1998; Möhler, 2001; Syrdal et al., 1998) wird die komplette Grundfrequenzkontur silbenweise in jeweils zwei zeitlich versetzte Sigmoide stilisiert, welche sich am höchsten Punkt treffen. Wenn man Möhler folgt, erhält man so 6 reellwertige Parameter (a_1 , a_2 , b , c_1 , c_2 , d). a_1 und a_2 bezeichnen die Steilheit des Anstiegs bzw. Abfalls der beiden Sigmoid-Abschnitte, b bezeichnet die Verschiebung der Funktion zur Silbe, c_1 und c_2 bezeichnen die Höhe des Anstiegs und Abfalls und d bezeichnet die Spitzenfrequenz in Hz. Auf diese Weise lässt sich jede akzentuierte Silbe und jeder Grenzton modellieren. Möhler führt weiterhin eine Normalisierung der Grundfrequenzspanne (*pitch range*) innerhalb jeder Intonationsphrase durch, indem er den höchsten Parameter $d = 1$

normalisiert und die Parameter d der anderen Silben sich in Bruchteilen auf diesen beziehen.

Wie beim Tilt-Modell sind auch im PaIntE-Modell nur bestimmte qualitative F_0 -Bewegungen möglich. Insbesondere fallend-steigende Intonationsbewegungen können mit diesem parametrischen Modell nicht generiert werden, obwohl es ansonsten ziemlich flexibel ist.

2.2 Approximations-Algorithmen

Um die Werte eines parametrischen Modells an eine extrahierte Folge von F_0 -Werten anzugleichen, bedient man sich numerischer Näherungsalgorithmen, von denen ich einige nennen möchte. In der Literatur finden sich die *hill-climbing* (Rojc, 2006) bzw. *gradient-decent-technique*, der *Nelder-Mead*⁵-Algorithmus für nichtlineare Regression, die *asymmetrical modal quadratic regression* und das mit dem MoMel verbundene Verfahren für quadratische *splines* (Hirst und Espesser, 1993; Mouline et al., 2004) sowie die *wavelet*-Transformation (van Santen et al., 2004).

2.3 Zusammenfassung

Die meisten entwickelten parametrischen Modelle nutzen das Vorwissen, welches aus früheren Arbeiten stammt. So liegt den meisten das Tonsequenzmodell von Janet Pierrehumbert zu Grunde, das hohe und tiefe Zieltöne in der Grundfrequenzspanne des Sprechers postuliert. Die grobe Unterscheidung in zwei Tonebenen scheint für die meisten Sprachen eine gelungene Annahme zu sein, da sich bereits viele linguistische Funktionen allein durch diese Unterscheidung treffen lassen (Isačenko und Schädlich 1966, nach Mixdorff 1998). Und selbst in Tonsprachen wie dem Mandarin oder auch dem Ibibio kann man zwei Töne (plus Downstep) sowie die Konturtöne als Transition dazwischen annehmen.

Abschließend kann man sagen, dass beinahe alle parametrischen Modelle entsprechend meiner Klassifikation als rein phonetisch und parametrisch zu betrachten sind und erst durch nachträgliche Untersuchungen linguistisch / phonologisch interpretierbar sind. Rein phonologische Modelle sind gewöhnlich zu präskriptiv. Eine volle Spezifizie-

⁵auch (*downhill*) *simplex-search*-Algorithmus genannt.

rung der Intonation ist sinnvoll, da der tonale Verlauf sowohl innerhalb der Silben als auch zwischen ihnen in einer Tonsprache bedeutungsunterscheidend sein kann.

3 Prädiktionsmethoden

Die Hauptaufgabe der Prosodiemodule eines Sprachsynthesystems besteht in der präzisen Bestimmung (Prädiktion) der prosodischen Parameter einer verständlichen und natürlichen Sprache.

In symbolischer Sprachsynthese verschiebt sich die Hauptarbeit der F_0 -Generierung von der Prädiktion eines gewissen Symbols für ein Intonationseignis auf dessen Umwandlung in eine Grundfrequenzkontur. In regelbasierten Prädiktionsverfahren wie dem KIM gelangt man aufgrund der semantischen und syntaktischen Analyse über (meist binäre) Entscheidungen direkt zu numerischen Werten für die Parametrisierung der Intonation. In einem parametrischen System geht es jedoch nicht darum, die Symbole zu präzisieren, aufgrund deren sich die Grundfrequenzwerte ableiten lassen, sondern direkt über bestimmte automatische Verfahren zu einer Grundfrequenzkontur zu gelangen. Wie in Kapitel 2 werde ich zunächst eine Klassifikation der Prädiktionsmethoden vornehmen. Dem schließt sich eine Übersicht über mögliche Prädiktionsparameter an.

3.1 Maschinelles Lernen

Die Intonationsprädiktion stellt eine typische Anwendung des Maschinellen Lernens (ML) dar. Darunter versteht man die künstliche Generierung von Wissen durch ein vom Menschen geschaffenes System über die automatische Erkennung von Regularitäten aus gegebener Information und deren Verallgemeinerung zur Anwendung auf bisher unbekannte Eingabedaten. In den folgenden Abschnitten stelle ich einige Dichotomien und Möglichkeiten des maschinellen Lernens vor.

3.1.1 Regelbasiert vs. Datenbasiert

”In rule-based systems, a set of human-crafted rules are defined based on existing knowledge. On the other hand, in a datadriven system, rules are

automatic[ally] derived from existing data.” (Sun, 2002a, S. 46)

In regelbasierten Systemen können von den Eingangsdaten über bestimmte Regeln direkt die Ausgabewerte abgeleitet werden. Bei den Regeln kann es sich z. B. um mathematische Formeln handeln, aber auch Klassifikation aufgrund der Eingangswerte oder eine Transformationsgrammatik sind in diesem Sinne Regeln. Ein regelbasierter Ansatz ist beispielsweise der Entscheidungsbaum, wie er im KIM genutzt wird. Ein solches System kann mit Expertenwissen erstellt werden und benötigt viel Aufwand und Feineinstellungen über Versuch und Irrtum, liefert dafür aber ein leicht verständliches Regelwerk, welches dem menschlichen Verstand zugänglich, diskutierbar sowie erweiterbar bleibt. Der Vorteil von solchen Regelwerken ist, dass sie sehr robust arbeiten (d. h. immer Werte liefern). Der Nachteil liegt in ihrer geringen Variationsbreite. Die Regeln erzeugen immer die gleichen Ausgabemuster passen und sich nicht selbstständig an ein neues Korpus an.

Während in regelbasierten Ansätzen ein menschlicher Experte aufgrund extensiver Untersuchungen Regeln aufstellt, können heute Maschinen die mühsame Suche nach Regularitäten übernehmen, wie es im datenbasierten Ansatz geschieht.

”In rule-based models [...], we need linguistic experts to write rules. It was a boring and labor-intensive task. Besides, it is difficult to create enough rules, which cover all of the cases in a language.” (Yu et al., 2002)

Dies bedeutet nicht notwendigerweise, dass die Regeln ausformuliert werden müssen. Beim Ansatz des *memory-based learning* (Moore, 2005; Daelemans et al., 2004) arbeitet der Algorithmus direkt auf den Trainingsdaten und kann mit Hilfe geschickt gewählter Kostenterme Parametermengen direkt klassifizieren.

Es besteht aber oft das Problem, eine ausreichende Menge an annotierten Daten für das automatische Training zur Verfügung zu haben:

”Data driven models suffer from data sparsity and can be difficult to generalise. Rule based models suffer from being over prescriptive and insensitive to the contents of the unit selection database.” (Aylett, 2004)

Dafür jedoch ermöglichen automatische Lernverfahren eine Adaptierbarkeit beliebiger Korpora und damit Sprachen, Sprecher und Sprechstile. Zu den Verfahren datenbasierten maschinellen Lernens gehören u. a. künstliche neuronale Netze (NN), *support*

vector machines (SVM), *classification and regression trees* (C&RT) und *belief networks* (Bayes Netzwerke, *bayesian networks*).

3.1.2 Überwachtes Lernen vs. Nicht überwachtes Lernen

Bei überwachtem Lernen (*supervised learning*) lernt die Maschine die Beziehung zwischen den Eingangsdaten im Vektor X und einen bestimmten Ausgabewert y . Dabei kann man dem "Fixieren" bzw. Vorgeben des richtigen Ausgabewertes in der Lernphase den Aspekt der Überwachung zuordnen. Man unterscheidet überwachtes Lernen für globale Modelle und solche, welche lokale Elemente enthalten (z. B. *nearest neighbour classifier*). Nicht-überwachtes Lernen (*unsupervised learning*) bezeichnet eine Menge von Eingangsdaten, deren mögliche Relationen untereinander trainiert werden. Das Modell erlernt somit selbstständig Regularitäten und wie jede Variable von den übrigen abhängt. Zusätzlich kann man noch den Modus des bestärkenden Lernens betrachten (*reinforcement learning*), bei dem die einzige Rückmeldung an die Maschine die binäre Information "richtig" oder "falsch" ist.

3.1.3 Ausgewählte datenbasierte Lernmethoden

3.1.3.1 Künstliche neuronale Netze

Künstliche neuronale Netze (NN) stellen den Versuch dar, die internen Vorgänge von Nervensystemen wie die parallele Datenverarbeitung zu simulieren und ihre Vorteile nutzbar zu machen. Die meist sequentielle Simulation in Computern nimmt dem Netz zwar den stärksten Vorteil - seine Parallelität, aber Dank der immer größeren Verarbeitungsgeschwindigkeit der heutigen Datenverarbeitung und dem speziellen theoretischen Hintergrund der NN haben sie sich als nützlich erwiesen. Heute existieren die verschiedensten Arten von NN für unterschiedliche Lernaufgaben. Für die Sprachsignalverarbeitung haben sich *feed-forward*¹ (FF-) NN (Rao und Yegnanarayana, 2004) und *bidirectional recurrent* NN (Möhler, 1998a) hervorgetan, welche sich für das Lernen von zeitlichen Vorgängen besonders eignen. Die Verarbeitung von Daten in NN ähnelt der Informationsverarbeitung eines Gehirns: Sie können nach einer gewissen Lernzeit

¹Ein Feedforward-Netz besitzt klar abgetrennte Schichten von Neuronen: Eine Eingabeschicht, eine Ausgabeschicht und beliebig viele innere, von außen nicht sichtbare Verarbeitungsschichten. Verbindungen sind nur zu Neuronen der jeweils nächsten Schichte erlaubt." (Kriesel, 2006)

”beliebige” (d. h. auch nicht-lineare und klassifizierende) Funktionen erlernen, stellen aber eine *black box* dar, da die inneren Beziehungen der ”Neuronen” nicht oder nur sehr schwer zu interpretieren sind (ein generelles Problem für die Neurowissenschaftler bei dem Verständnis der Funktionsweise des Gehirns). NN können mit Hilfe eines überwachenden Lernverfahrens (*back-propagation*) auf abstrakte Weise aus Eingangsdaten Regeln ableiten und aufgrund der gewonnen inneren Struktur auch auf den Eingang unbekannter Daten eine approximierte Ausgabe liefern:

”Neural networks are known for their ability to generalize and capture the functional relationship between the input-output pattern pairs and have the ability to predict, after an appropriate learning phase, even patterns not presented before” (Rao und Yegnanarayana, 2004)

Hingegen sehen Kochanski und Shih (2001a) die Situation eher pessimistisch und votieren für ein offenes System, in dem sich Regelwerk und sprecherspezifische Parameter trennen lassen:

”NNs are, mathematically, an interpolater whose structure is unrelated to the system that they model, so one cannot expect the output of a NN to behave well in situations outside its training set”

Ein weiteres Problem besteht in einer sinnvollen Wahl für die Anzahl der Trainingsrunden (Epochen) und die Lernrate (Eta-Wert) η der einzelnen Neuronen. Werden diese Parameter ungünstig gewählt, so kann das NN unter- oder übertrainiert (zu generalisiert oder zu spezialisiert) werden und falsche Werte liefern. Darüber hinaus stellt sich dem Entwickler die Frage, welche Topologie (Netzaufbau) das NN besitzen soll, d. h. aus wie vielen ”Neuronen” welcher Art mit welchen Verknüpfungen das Netz aufgebaut werden soll, um die optimalen Ergebnisse für eine bestimmte Aufgabenstellung zu erreichen. Man kann zwar einige Kriterien beim Entwurf heranziehen (wie z. B. die Anzahl der Prädiktorvariablen als Anzahl der Eingangsneuronen), aber manche Entscheidungen können bisher nur durch Versuch und Irrtum vorgenommen werden².

3.1.3.2 Support vector machines

Die SVN ist ein Spezialfall der *kernel machines* und löst einige Probleme, welche in NN vorkommen. Dabei werden die Daten in einen höherdimensionalen Merkmalsraum

²Die Anwendung von NN in der Spracherkennung wird z. B. in Tebelskis (1995) ausführlich behandelt.

transformiert, in welchem ihre Eigenschaften linear trennbar sind. Durch einen höherdimensionalen Raum kann man auch komplexe Funktionen ausdrücken. Der Nachteil liegt in einem höheren Rechenaufwand und dem weiter unten erwähnten Problem der vielen Dimensionen (*curse of dimensionality*). Außerdem ergibt sich ein Problem im Auffinden der am besten geeigneten Hyperebene, um die Daten zu klassifizieren, da es von ihnen möglicherweise sehr viele gibt. Durch die Nutzung vieler Dimensionen können auch *Scheinregularitäten* gefunden werden, die sich in einer anderen Menge von Testdaten nicht bestätigen würden. SVMs haben sehr gute Ergebnisse beispielsweise in der Texterkennung (*optical character recognition OCR*) und generell in der Mustererkennung erzielt; ob sie sich für die Sprachsynthese eignen, muss sich noch herausstellen.

3.1.3.3 Entscheidungsbäume

Die Datenstruktur des Baumes ist (als Untermenge der Graphen) eine der grundlegendsten Datenstrukturen überhaupt und in vielen Fachgebieten bekannt. Überall, wo man über sukzessive Entscheidungen (z. B. zwischen "ja" und "nein") zu einem Ergebnis kommen möchte, lassen sich Entscheidungsbäume anwenden. Entscheidungsbäume können sowohl durch menschliches Expertenwissen (d. h. regelbasiert) als auch maschinell (datenbasiert) erzeugt werden. Die Entscheidungen können auch in der Maschine selbst getroffen werden: Auf der Basis einer großen Datenmenge wird algorithmisch ein Entscheidungsbaum erzeugt (der weiterhin dem menschlichen Verstand zugänglich ist) und wird vom Rechner zur Entscheidungsfindung benutzt.

Classification and regression trees C&RT bezeichnen Entscheidungsbäume (d. h. Folgen aus wenn-dann Entscheidungen), mit denen sich kontinuierliche oder kategoriale abhängige Variablen bestimmen lassen. Der klassische C&RT-Algorithmus stammt von Breiman et al. (1984) und ist bekannt unter der im Folgenden genutzten Abkürzung "CART®". CART ist ein Oberbegriff für zwei Arten von Bäumen:

- Soll ein Entscheidungsbaum ein kategoriales Merkmal präzisieren, handelt es sich um einen *classification tree* (Klassifikationsbaum), an dessen Blättern jeweils die wahrscheinlichste Kategorie festgehalten ist.
- Soll ein kontinuierlicher Parameter bestimmt werden, ist es ein *regression tree* (Regressionsbaum), an dem alle Knoten jeweils mit Mittelwert und Standardabweichung markiert sind.

Besonders in der klinischen Medizin, wo schnelle, nachvollziehbare und statistisch fundierte Entscheidungen bei vielen Eingabeparametern gefällt werden müssen, macht der Einsatz von CARTs Sinn (Lewis, 2000). Als Vorteile nennt Lewis:

- Es müssen keine Annahmen über mögliche Werteverteilungen der Prädiktorvariablen getroffen werden. Das bedeutet auch, dass die Beziehung zwischen Prädiktorvariablen und der abhängigen Variable weder linear noch monoton sein muss.
- CARTs können mit einer unvollständigen Menge an Prädiktorvariablen umgehen: Die fehlenden Werte können durch abgeleitete ersetzt werden und die Entscheidungsfindung kann trotz ungenügender Datenlage fortgesetzt werden.
- "The CART building algorithm implicitly deals with sparseness of units in that it will only split a cluster if there are sufficient examples and significant difference to warrant it." (Black und Taylor, 1997a)
- CARTs können mit relativ wenig Aufwand für den Analytiker maschinell trainiert werden und die Bäume können auch von Nicht-Statistikern einfach angewendet, nachvollzogen und erweitert werden.
- Die binäre Struktur von CARTs erlaubt darüberhinaus eine effiziente Speicherung in digitalen Rechenmaschinen; es gibt aber auch buschigere Algorithmen, z.B. C4.5.

Bei CARTs handelt es sich um *greedy*-Algorithmen:

"It can be seen that CART is a greedy algorithm in that at each stage it finds the locally best question that makes a split. Once the partitions are determined, the algorithm continues the same process and never looks back to make changes. This is suboptimal, but a global optimization would be very computationally expensive. Fortunately, in practice it seems this suboptimality doesn't pose serious problems in most cases." (Sun, 2002a)

Wie auch bei neuronalen Netzen gibt es in CARTs das Problem der *sparsity* (Datenknappheit), von Richard Bellman (dem Erfinder der dynamischen Programmierung) auch "*curse of dimensionality*" genannt. Das heißt, dass sich die Trainingsdaten in einem hochdimensionalen Raum befinden (nicht selten mit 20 Dimensionen oder mehr)

und es aufgrund der begrenzten Menge an Trainingsdaten kaum zu Ballungen in diesem Hyperraum kommen kann, auf Grund derer man klassifizieren könnte. CARTs trennen den Merkmalsraum in immer kleinere Partitionen auf; es kann jedoch schnell passieren, dass nach einigen Trennpunkten (*splitters*) keine Trainingsdaten mehr für eine sinnvolle Partitionierung vorhanden sind; dann würde der Entscheidungsbaum auf der Grundlage eines einzelnen Datenpunktes übergeneralisieren. Dazu kommt, dass eine unglückliche Partitionierung in einem frühen Entscheidungsschritt zu Fehlern und redundanten Entscheidungen in allen tieferen Ebenen führen kann. Daher macht es manchmal Sinn, einem CART einige Entscheidungen *a priori* aufzuzwingen. Außerdem eignen sich CARTs nicht gut zur Klassifikation von additiven Strukturen; dafür gibt es die *multivariate adaptive regression splines* MARS (Tutz, 2005). Im Gegensatz zu NN mit ihren arbiträr großen Ausgabeschichten können CARTs nur eine Variable prädisieren, so dass man für jede Ausgabevariable ein eigenes CART trainieren muss. Eine komfortable Implementation von CART namens *wagon* befindet sich beispielsweise im Festival-System (Black und Taylor, 1997b; Taylor et al., 1998) und wird auch in BOSS benutzt.

Ensemble learning Werden einzelne Klassifizierer wie NN oder CARTs benutzt, erhält man eine hohe Genauigkeit in der Prädiktion, wenn die Eingabevariablen aus den Trainingsdaten stammen, aber Abweichungen am Eingang können zu Abweichungen des Prädiktionsergebnisses führen. Hier setzt das *ensemble learning* an, in dem mehrere Instanzen des Klassifizierers über verschiedene Teilmengen der Trainingsdaten trainiert werden und die letztendliche Klassifizierung über eine Gewichtung der Einzelprädiktionen durchgeführt wird. Man kann *ensemble learning* unterteilen in *bagging* (unabhängig konstruierte Ensembles) und *boosting* (verknüpfte Konstruktion von Ensembles). *Ensemble learning* zeigt sehr gute Ergebnisse und kommt an menschliche Klassifikationsleistung heran, hat aber meines Wissens noch keine breite Anwendung in der Sprachtechnologie gefunden.

3.1.3.4 Memory-based learning

Die Methode des *memory-based learning* (MBL³) basiert auf dem einfachen Gedanken des analogen Schließens: In dem man ein neues Ereignis mit bereits bekannten

³Auch unter den Namen *similarity-based*, *example-based*, *exemplar-based*, *case-based*, *instance-based*, *kernel-based*, *analogical* und *lazy learning* sowie *nonparametric regression* bekannt.

vergleicht, kann man es klassifizieren.

”Memory-Based Learning (MBL) is based on the idea that intelligent behavior can be obtained by analogical reasoning, rather than by the application of abstract mental rules as in rule induction and rule-based processing. In particular, MBL is founded in the hypothesis that the extrapolation of behavior from stored representations of earlier experience to new situations, based on the similarity of the old and the new situation, is of key importance. [...]”

Historically, memorybased learning algorithms are descendants of the k-nearest neighbor (henceforth k-NN) algorithm (Cover and Hart, 1967; De-
vijver and Kittler, 1982; Aha, Kibler, and Albert, 1991).” (Daelemans et al., 2004)

Erst durch die geschickte Auswahl eines Distanzmaßes und effizienter Datenstrukturen wird aus diesem einfachen Ansatz eine lohnenswerte Anwendung des maschinellen Lernens.

3.2 Prädiktionsmerkmale

In Tabelle 3.1 sind einige Prädiktionsmerkmale zusammengetragen, wie sie in gängige F₀-Prädiktionsverfahren eingehen. Wie man sieht, handelt es sich fast ausschließlich um Merkmale von Akzentsprachen. Für eine Tonsprache wie dem Ibibio werden weitere Merkmale benötigt, welche sich besonders auf die tonalen Eigenschaften beziehen. Meine Vorschläge dazu werde ich in Tabelle 4.5 vorstellen.

| Art | Merkmale | |
|----------|--------------|--------------------------------|
| | Einheit | Bezug (+ zusätzlicher Kontext) |
| Position | Nukleus | in der Silbe |
| | Phon | Anlaut, Nukleus oder Coda |
| | Silbe | im Wort |
| | Silbe | in der Intonationsphrase |
| | Wort | in der Intonationsphrase |
| | Wort | im Satz |
| | Akzentgruppe | in der Intonationsphrase |

| Art | Merkmale | |
|------------|---|--|
| | Einheit | Bezug (+ zusätzlicher Kontext) |
| | Akzentgruppe Intonationsphrase | im Satz im Satz |
| Anzahl | Phone Silben Silben Silben Akzente Akzente Worte Akzentgruppen | Silbe Wort + linkes Wort Akzentgruppe Intonationsphrase bis vorangehende größerer Grenze bis nächster größerer Grenze Intonationsphrase Intonationsphrase |
| Distanz | Silben Silben Akzentgruppen | zum linken Akzent/Grenztone zum rechten Akzent/Grenztone bis zum Ende des Satzes |
| Dauer | Anlaut Reim Prä-stabile Phase Stabile Phase Post-stabile Phase Silbe Phrase Pause | Silbe Silbe Silbe Silbe Silbe in ms, ± 1 Silbe in Silben vor und nach Silbe |
| Kategorial | ToBI-Label Silbenposition Wortposition Satzteil (POS) Satzteil Satztyp Akzent Akzent | Silbe ± 2 Silben Initial, Medial, Final, Einzel Satz-/Phrasen- Initial/Final Wort ± 1 Wort Erstes und letztes Wort der Akzentgruppe Deklaration, Interrogation, Ja-Nein-Frage, Exklamation Wort ± 1 Wörter Silbe ± 2 Silben |

| Art | Merkmale | |
|----------------|--|--|
| | Einheit | Bezug (+ zusätzlicher Kontext) |
| | Vokal Letzter stimmhafter Konsonant Grenztonstärke Ton | Silbe Silbe (0-4) Silbe \pm 1 Silbe |
| Typ | Silbenart Phon-Klasse Akzent Grenzton Akzentgruppe Intontationsphrase | V,CV,VC oder CVC Sonoranten, Frikative,... |
| Kontinuierlich | Akzentkommando | letztes Kommando in Phrase (im Fujisakimodell) |

Tabelle 3.1: F₀-Prädiktionsmerkmale in der Literatur

3.3 Zusammenfassung

Während früher tendentiell eher regelbasierte Systeme genutzt wurden (vgl. z. B. Kohler 1991a), basieren die erfolgreichsten heutigen Ansätze größtenteils auf großen Datenbasen, mit deren Hilfe Prädiktionen über ML-Algorithmen trainiert werden. Da die Rechner schneller geworden sind und inzwischen ausreichend große Arbeitsspeicher besitzen, ist heute auch Korpusssynthese möglich, welche momentan die artikulatorische und die Formantsynthese qualitativ überholt hat. Unter den ML-Methoden sind besonders CARTs ein vielversprechender Ansatz, da sie im Gegensatz zu NNs und SVMs keine global optimierte "schwarze Kiste" darstellen. Man kann die einzelnen Schritte des Entscheidungsbaumes nachvollziehen und manuelle Eingriffe vornehmen. Zusätzlich sind CARTs extrem schnell in der Ausführungszeit und belegen wenig Speicherplatz zur Laufzeit.

Im folgenden Kapitel werde ich den Entwurf des Intonationsmoduls und die grundlegenden Methoden der Prädiktion erläutern. Zusätzlich folgt die Begründung für die

Auswahl Ibibio-spezifischer Prädiktionsmerkmale und der Kostenterme der Einheiten-
auswahl.

4 Entwurf

4.1 Syntheseebenen

Bei der Entwicklung der Kommunikationsschnittstelle zwischen Client und Server entstand die Frage, ob die einzelnen Ebenen der Sprachsynthese (Sätze, Wörter, Silben, Phone) in eins-zu-n-Beziehungen hierarchisch geordnet sein sollten und wenn nicht, wie die Daten repräsentiert werden können. Innerhalb von etwa 9.11 % aller Silben (ohne Pausen) gibt es eine Wortgrenze; 74.04 % aller Wörter besitzen mindestens einen Rand, der nicht gleichzeitig eine Silbengrenze ist.

Ursachen dafür sind unter anderem, dass im Gegensatz zum Deutschen vor initialen Vokalen kein Glottalverschluss gesetzt wird und der Vokal als Silbenkern mit dem vorangehenden Konsonant als Anlaut eine Silbe bildet. Im Ibibio gibt es viele Fälle, bei denen ein wortfinaler Konsonant mit dem initialen Vokal des folgenden Wortes verschmilzt (/apiŋ ufoh ɔ basi odo/ → /a.pi.ŋu.fɔ.hɔ.ba.sio.do/).

Da wir innerhalb des Projektes bisher zu keiner Übereinkunft gekommen sind wie die Überschneidungen der verschiedenen Ebenen zu behandeln sind, entscheide ich mich für die einfachste lauffähige Lösung: Silben werden als eine direkte Unterebene von Wörtern betrachtet, wobei Silben, welche sich über eine Wortgrenze erstrecken, das Attribut **Border=1** erhalten, sonst **Border=0**. Die Wortgrenze ist in der Silbentranskription **TReal** mit einem # gekennzeichnet. Wörter, die mindestens an einem Rand eine inkonsistente Silbengrenze besitzen, erhalten das Attribut **Broken=1**, sonst **Broken=0**. Sie werden von der Vorauswahl nicht ausgewählt, bis eine Lösung für dieses Problem gefunden wird. Dadurch wird die Anzahl der auffindbaren Wörter allerdings auf gerade einmal knapp 26 % aller Wörter des Korpus reduziert. Da aber ursprünglich eine Synthese auf reiner Silbenebene geplant war, können diese wenigen Wörter evtl. doch noch etwas zur Synthesequalität beitragen.

4.2 F₀-Modellierung

Ich möchte - der Klassifikation aus Kapitel 2 folgend - eine voll spezifizierte silbenweise phonetisch-parametrische Intonationsmodellierung realisieren. Dies hat den Vorteil, die Parameter einfach und vollautomatisch extrahieren und direkt als numerische Werte repräsentieren zu können. Der Nachteil einer rein phonetisch-deskriptiven Repräsentation besteht in der Tatsache, dass man den extrahierten Werten nur nachträglich eine linguistische Bedeutung auferlegen kann.

Die Stilisierung ist vollständig, d. h. es werden keine Zieltöne an bestimmten Zeitpunkten parametrisiert, sondern silbenweise Polynome lückenlos über den gesamten Satz hinweg. Beispiele für einfache Polynome liefert Abbildung 4.1.

Dabei wird keine der Wahrnehmung entsprechende Korrektur der Grundfrequenzskala vorgenommen, da eine vollständige Parametrisierung in jedem Fall alle Eigenschaften der Intonation berücksichtigen sollte und da es sich andererseits auch in anderen einfachen Modellen gezeigt hat, dass dies perzeptiv gar nicht notwendig ist (z. B. 't Hart et al. 1990). Experimentell werden die Grundfrequenzwerte dennoch logarithmiert und auf diesen Werten eine Stilisierung durchgeführt. Die Ergebnisse waren jedoch durchweg schlechter als auf der linearen Skala (dazu im Folgenden Tabelle 4.1).

Es wird keine Superposition modelliert sondern ein sequentielles Modell entwickelt. Im Ibibio ersetzen Downstep und Downdrift (welche man möglicherweise als phonologische Komponenten ansehen könnte) die globale Deklinations-Komponente, welche weitestgehend als phonetischer Effekt betrachtet wird (Connell, 2001). Da sich die Down-trends des Ibibio leicht parametrisieren lassen, gehen sie als Merkmale in die Prädiktion ein und bilden *keine* eigene Ebene in einem additiven Modell. Bei additiven Modellen müssen die einzelnen Ebenen getrennt voneinander prädiziert und dann wieder zusammengefügt werden (z. B. Sakai 2004), was aber den Vorteil hat, weniger Trainingsdaten je Ebene zu benötigen (Yu et al., 2002). Die typischen ebenenübergreifenden Informationen wie "Position der Silbe in der Phrase" gehen in meinem sequentiellen Modell über die Prädiktion einer entsprechenden Tonschablone mit ein. Gleiches gilt für die Mikroprosodie, welche in der Parametrisierung des Kontextes implizit mit eingefangen werden soll (z. B. durch Charakterisierung des Silbentyps für vokalspezifische Grundfrequenz und koartikulatorische F₀-Variationen) und durch die Einheiten Auswahl aus einem großen Korpus reproduziert werden kann.

Dieses einfache silbische Intonationsmodell sollte sich leicht auf andere Sprachen mit

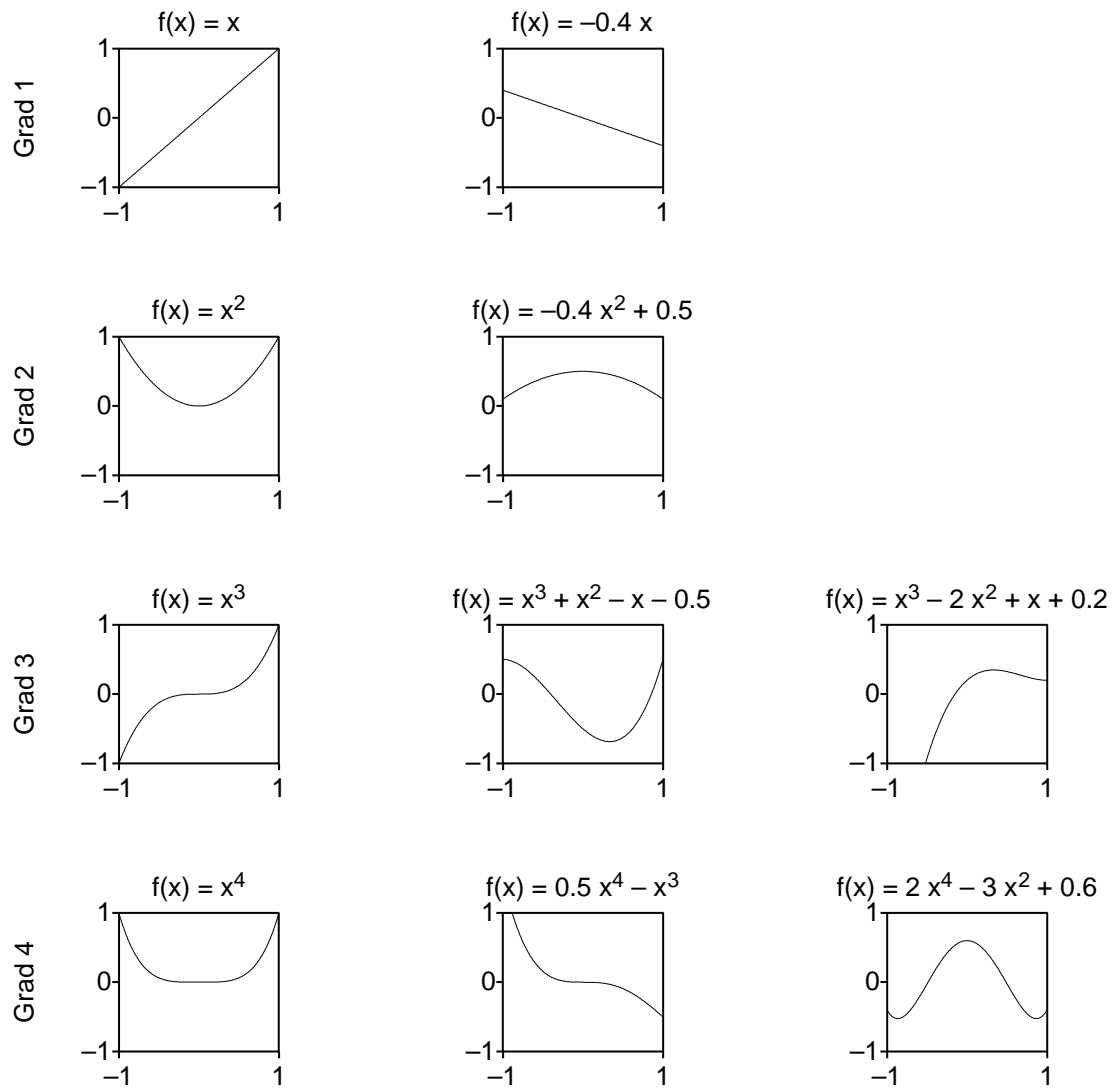


Abbildung 4.1: Beispiele für einfache Polynome

silbischen Tönen (z. B. das Chinesische) anwenden lassen und bietet viel Raum für weitere Entwicklungen, möglicherweise auch in Akzentsprachen.

4.2.1 Polynom-Regression

4.2.1.1 Untersuchung

Um die Randbedingungen der parametrischen Intonationsmodellierung festzulegen, wird in einer vorangehenden Untersuchung die optimale Kombination von Einstellungen für die Polynom-Regression (Stoer, 2005, S. 42) bestimmt. Zu den Einstellungen gehört der Polynomgrad, die Interpolation stimmloser Abschnitte und die Achsenskalierung¹. Dazu wird das gesamte Korpus stilisiert, resynthetisiert und ein Fehlermaß zwischen den Grundfrequenzkonturen aller Sätze bestimmt. Die folgende Tabelle zeigt die Ergebnisse des *root mean square error RMSE*, des *mean absolute errors MAE* und des Korrelationskoeffizienten r bzw. r^2 :

| Polynom- grad | Skala: | RMSE | | MAE | | r | | r ² | |
|------------------|--------|-------|-------|------|------|-------|-------|----------------|------|
| | | lin | log | lin | log | lin | log | lin | log |
| 1 | | 10.55 | 20.14 | 6.74 | 7.88 | 0.964 | 0.878 | 0.93 | 0.77 |
| 2 | | 6.98 | 18.36 | 4.26 | 5.20 | 0.985 | 0.901 | 0.97 | 0.81 |
| 3 | | 5.47 | 17.89 | 3.17 | 4.18 | 0.991 | 0.907 | 0.98 | 0.82 |
| 4 | | 4.53 | 17.66 | 2.49 | 3.53 | 0.994 | 0.910 | 0.98 | 0.82 |
| 5 | | 4.02 | 17.36 | 2.14 | 3.17 | 0.995 | 0.914 | 0.99 | 0.84 |

Tabelle 4.1: Messwertabweichungen der Grundfrequenzstilisierung bei unterschiedlichen Skalen und Polynomgraden

Abbildungen 4.2 und 4.3 zeigen beispielhaft, wie exakt die einzelnen Stilisierungen den Originalgrundfrequenzverlauf einfangen.

Da die Ergebnisse bei Logarithmierung der Grundfrequenzskala durchweg schlechter waren und in der folgenden Vektorquantisierung auch nicht quantitativ vergleichbar sind, wurde die Logarithmierung verworfen.

Bei der Polynom-Regression wurden nur *frames* (Messwerte) berücksichtigt, die 50 Hz $< F_0 < 1000$ Hz erfüllten. Alle anderen Werte gelten als 0 mit der Bedeutung "stimmlos"

¹Experimentell wurden auch Normalisierungen der Dauer und Frequenz vorgenommen und eine Polynom-Symmetrie eingeführt. Beim momentanen Ansatz haben diese Eigenschaften jedoch keine Auswirkung mehr.

und werden gegebenenfalls interpoliert. Wenn nicht mindestens ein Drittel aller Werte stimmhaft ist, gilt die Silbe als stimmlos bzw. nicht reliabel genug und wird wie eine Pause behandelt und alle Koeffizienten bzw. Stützstellen werden auf Null gesetzt.

| Interpolation | Grad | RMSE | MAE | r | r ² |
|------------------------|------|-------|------|-------|----------------|
| Mitte | 1 | 10.66 | 6.87 | 0.964 | 0.93 |
| | 2 | 7.07 | 4.36 | 0.984 | 0.97 |
| | 3 | 5.63 | 3.30 | 0.990 | 0.98 |
| | 4 | 4.69 | 2.64 | 0.993 | 0.99 |
| | 5 | 4.17 | 2.29 | 0.995 | 0.99 |
| Mitte und Ränder | 1 | 10.84 | 7.05 | 0.962 | 0.93 |
| | 2 | 7.46 | 4.63 | 0.982 | 0.96 |
| | 3 | 5.85 | 3.48 | 0.989 | 0.98 |
| | 4 | 4.88 | 2.78 | 0.992 | 0.98 |
| | 5 | 4.33 | 2.41 | 0.994 | 0.99 |

Tabelle 4.2: Messwertabweichungen der Grundfrequenzstilisierung bei unterschiedlichen Interpolationsmethoden

Tabelle 4.2 zeigt die Messwertabweichungen bei verschiedenen Arten der Interpolation stimmloser Abschnitte. *Mitte* bezeichnet die lineare Interpolation von stimmlosen Regionen innerhalb der Silbe und *Ränder* bezeichnet das Kopieren des ersten bzw. letzten vorhandenen Grundfrequenzwertes in die stimmlosen Regionen zu Beginn und am Ende der Silbe. Die Werte sind in beiden Fällen schlechter als in einer Stilisierung ohne Interpolation. Das ist nicht verwunderlich, da besonders die linear interpolierten Werte mehr Einschränkungen für die Polynomregression bedeuten (die jede Kurve global optimieren muss). Es macht dennoch Sinn, diese Interpolation durchzuführen, da die so extrahierten Polynome völlig andere Formen besitzen als ohne und daher die sie beschreibenden Punkt andere Ballungen im n-dimensionalen Polynomraum bilden können.

4.2.1.2 Ergebnis

Man muss die Werte in den Tabellen 4.1 und 4.2 vorsichtig interpretieren, da nur ein Teil des gesamten Korpus vorlag und dieses in seiner Laut- und Tonverteilung nicht unbedingt als repräsentativ für eine typische Äußerung des Ibibio sein muss. Für ei-

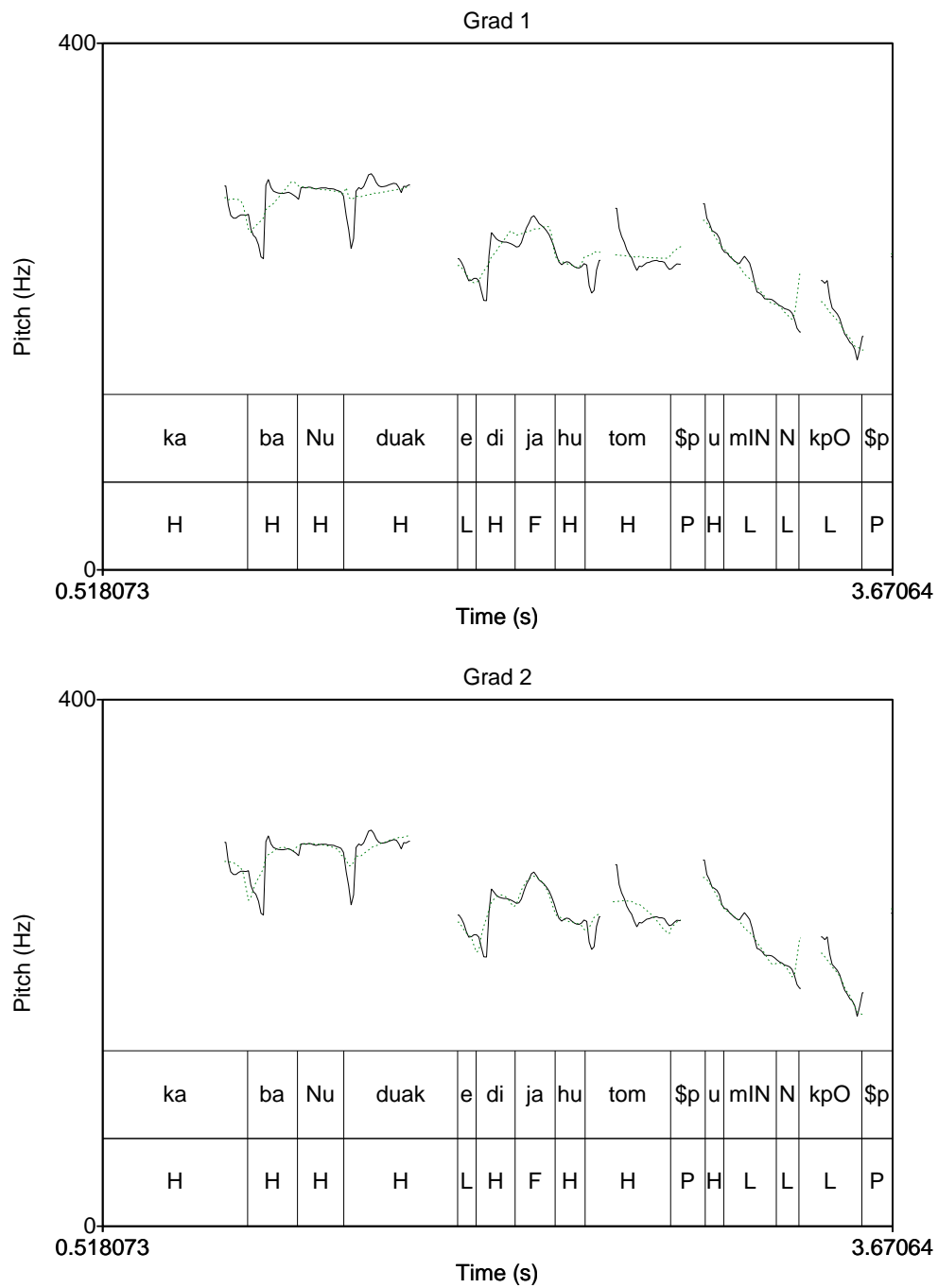


Abbildung 4.2: Stilisierungsgenauigkeit mit Polynomen 1. und 2. Grades. Die gestrichelte Linie gibt die Stilisierung an.

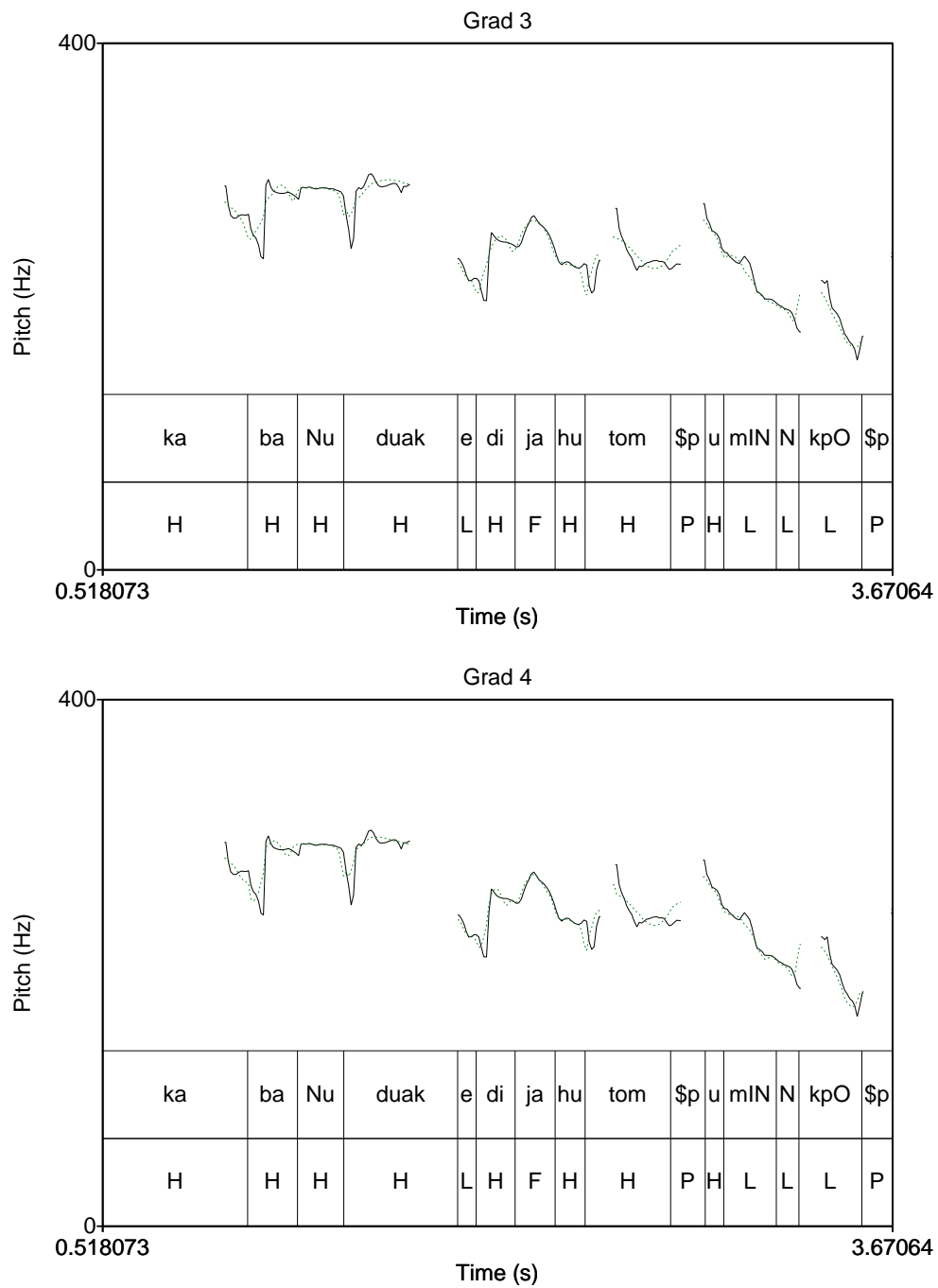


Abbildung 4.3: Stilisierungsgenauigkeit mit Polynomen 3. und 4. Grades. Die gestrichelte Linie gibt die Stilisierung an.

ne vorläufige Betrachtung reichen die Werte jedoch aus. Mit steigendem Grad nimmt zwar der RMSE ab, andererseits aber die Anzahl der Silben zu, für die keine Polynomregression durchführbar ist. Dies liegt beispielsweise daran, dass das Gleichungssystem aufgrund der zu geringen Anzahl an Stützwerten nicht gelöst werden kann. Solche Fälle werden genau wie Pausen behandelt und aus der Berechnung des RMSE herausgenommen. Die Interpolation verschlechtert den RMSE offenbar, aber eine endgültige Entscheidung über den Polynomgrad und Anwendung einer Interpolation muss noch aufgeschoben werden, da sich erst im folgenden Abschnitt zeigen wird, ob eine bestimmte Parameterkonfiguration sinnvoll ist.

4.2.2 Vektorquantisierung

Bei Vektorquantisierung (VQ) handelt es sich um eine verlustbehaftete Datenkomprimierung, welche einfach und vollautomatisch durchführbar ist:

”In the earlier days, the design of a vector quantizer (VQ) is considered to be a challenging problem due to the need for multi-dimensional integration. In 1980, Linde, Buzo, and Gray (LBG) proposed a VQ design algorithm based on a training sequence. The use of a training sequence bypasses the need for multi-dimensional integration. A VQ that is designed using this algorithm are referred to in the literature as an LBG-VQ.”
(<http://www.data-compression.com/vq.shtml>, 2006-08-16.)

Der ursprüngliche Artikel zum LBG-Algorithmus findet sich in Linde et al. (1980), nach Wendemuth (2004). Um ein Inventar der prototypischen Silbenkonturen zu erhalten, wende ich die VQ auf die Menge aller Silbenpolynome an. Da sich die n Koeffizienten eines Polynoms nicht direkt interpretieren lassen und verschiedene Dimensionen besitzen $(x^0, x^1, x^2, \dots, x^n)$, eignen sie sich nicht als Koordinaten im n -dimensionalen Raum für einen quantitativen Vergleich. Jedes Polynom wird daher durch $n+1$ (nicht notwendigerweise) äquidistante Stützstellen repräsentiert, aus denen per Horner-Schema eindeutig alle Funktionswerte rückgewonnen werden können. Da die Vektorquantisierung VQ den Merkmalsraum sukzessive halbiert, ist die Anzahl der Codevektoren $|C| = 2^N$, wobei N die Zahl der Vektorraumteilungen bezeichnet. Die Codevektoren entsprechen Punkten im $(n+1)$ -dimensionalen Raum und stellen prototypische Punkte dar, welche alle extrahierten Datenpunkte (d. h. Stilisierungskurven) am besten repräsentieren.

4.2.2.1 Untersuchung

Alle gültigen Silbenkonturen des Korpus, repräsentiert durch jeweils $n + 1$ Stützstellen des sie konstituierenden Polynoms, werden mit unterschiedlichen Codebookgrößen zweimal vektorquantisiert. Die Bedingung für die Tauglichkeit der Stützstellen $b_0 \dots b_n$ des Polynoms n. Grades ist willkürlich festgesetzt auf: $\forall_{i=0}^n : -4000 < b_i < +4000 \wedge b_i \neq 0$. Die Stützstellen könnten und dürften zwar theoretisch Null sein, aber ein solcher Wert wird als Markierung für Pausen gebraucht und Pausensilben werden herausgefiltert. Für den Fall einer Interpolation aller stimmlosen Abschnitte ist dieses Intervall natürlich viel zu weit und unwirksam.

| Grad | # Silben | $ C_1 $ | dist. | SNR | $ C_2 $ | dist. | SNR |
|------|----------|---------|--------|--------|---------|---------|--------|
| 3 | 2333 | 64 | 230.31 | -23.62 | 8 | 568.68 | -27.55 |
| 4 | 2322 | | 224.03 | -23.50 | | 531.99 | -27.26 |
| 5 | 2067 | | 240.54 | -23.81 | | 5656.15 | -37.52 |
| 3 | 2333 | 64 | 230.31 | -23.62 | 16 | 314.46 | -24.98 |
| 4 | 2322 | | 224.03 | -23.50 | | 290.29 | -24.63 |
| 5 | 2067 | | 240.54 | -23.81 | | 4836.02 | -36.84 |
| 3 | 2333 | 128 | 156.82 | -21.95 | 16 | 447.28 | -26.51 |
| 4 | 2322 | | 156.35 | -21.94 | | 389.75 | -25.91 |
| 5 | 2067 | | 170.50 | -22.32 | | 3762.66 | -35.75 |
| 3 | 2333 | 128 | 156.82 | -21.95 | 32 | 309.64 | -24.91 |
| 4 | 2322 | | 156.35 | -21.94 | | 258.54 | -24.13 |
| 5 | 2067 | | 170.50 | -22.32 | | 2744.57 | -34.38 |

Tabelle 4.3: Vergleich der Vektorquantisierung verschiedener Codebookgrößen und Polynomgrade bei Interpolation aller stimmlosen Silbenabschnitte. Angabe von Distortion (dist.) und Signal-zu-Rausch-Energieverhältnis (SNR).

4.2.2.2 Ergebnis

In Tabelle 4.3 werden die Signal-zu-Rausch-Energieverhältnisse (*signal-to-noise-ratios* SNRs) der VQ bei unterschiedlichen Polynomgraden und Codebookgrößen bei Interpolation an den Silbenrändern und innerhalb der Silbe verglichen. Die SNR ergibt sich rechnerisch zu $-10 * \log_{10} dist$ aus der distortion der VQ. Je mehr dieser Wert gegen Null geht, umso besser repräsentieren die Codevektoren die Datenpunkte. $|C_2|$ gibt die

Werte einer nochmaligen VQ des Codebooks C_1 an (s. u.). Die SNR der VQ bei Grad 5 ist deutlich schlechter als bei den Graden 3 und 4, selbst wenn eine Vergrößerung des Codebooks auch hier noch eine Verbesserung der Werte mit sich bringt. Ein Grund für den Abfall der Qualität liegt möglicherweise in der Menge an Silben, deren Stützstellen eine nicht zu lösende Regressionsgleichung darstellen. Die Zahl der gültigen Silben sinkt daher mit dem Polynomgrad und einzelne Abweichungen fallen stärker ins Gewicht. Ich entscheide mich für Polynome 4. Grades und untersuche den Einfluss der Codebookgröße näher. Tabelle 4.4 zeigt, wie Distortion und SNR mit zunehmender Codebookgröße gegen Null gehen. Außerdem zeigt die Tabelle, dass eine Beschränkung auf eine Interpolation in der Silbenmitte keinen großen Gewinn gegenüber keiner Interpolation mit sich bringt (womöglich, weil nur wenige Messfehler zu Nullwerten innerhalb der stimmhaften Abschnitte geführt haben, die interpoliert werden könnten).

4.2.2.3 Entscheidung über Polynomgrad, Interpolation und Codebookgrößen

Die Silbenintonationsstilisierung wird mit biquadratischen Polynomen vorgenommen (engl. *quartic*, 4. Grad). Zum einen ist der Zugewinn für RMSE und MAE zwischen 3. und 4. Grad signifikant und zum anderen ist die Zahl der nicht lösbaren Systeme weitaus geringer als bei der Regression 5. Grades (255 Silben weniger). Die Anzahl der Parameter pro Silbenkontur (5) ist beispielsweise vergleichbar mit der Datenmenge des Tilt-Modells (s. 2.1.6.2). Zudem kann jede Silbe mit ihren 5 Stützstellen so in 4 äquidistante Abschnitte zerteilt werden, dass man an ihr direkt die Grundfrequenz am linken und rechten Rand sowie in der Silbenmitte ablesen kann, was man beispielsweise in der späteren Berechnung von Verkettungskosten nutzen könnte. Es findet keine Amplitudennormalisierung statt, da die Information über den absoluten Startwert und die Grundfrequenzspanne jeder Silbe nicht separat prädiziert werden soll sondern implizit in einer Schablone enthalten sein sollte.

Ich entscheide mich für eine Codebookgröße von 64. Diese Menge an Tonschablonen stellt einen Kompromiss zwischen zu starker Generalisierung und zu starker Spezialisierung dar (jede Silbe hat einen eigenen Codevektor). Hinzu kommt, dass das Korpus generell schon ziemlich klein ist, was eine kleinere Codebookgröße sinnvoll macht. Das Codebook selbst wird noch einmal in 16 "Meta"-Codevektoren vektorquantisiert und man erhält Klassen von Codevektoren (im folgenden "Tonschablonenklassen"), welche in der Einheitenvorauswahl (4.4.1) wichtig sind.

| Inter- polation | Codebook | | | Codebook 2 | | |
|------------------------|----------|------------|--------|------------|------------|--------|
| | $ C_1 $ | distortion | SNR | $ C_2 $ | distortion | SNR |
| keine | 16 | 20632.88 | -43.15 | 4 | 57687.44 | -47.61 |
| | 32 | 16763.54 | -42.24 | 4 | 111901.26 | -50.49 |
| | 32 | 16763.54 | -42.24 | 8 | 65084.27 | -48.13 |
| | 64 | 10239.96 | -40.10 | 8 | 145336.99 | -51.62 |
| | 64 | 10239.96 | -40.10 | 16 | 80999.92 | -49.08 |
| | 128 | 3345.01 | -35.24 | 16 | 105556.25 | -50.23 |
| | 128 | 3345.01 | -35.24 | 32 | 33561.92 | -45.26 |
| | 256 | 536.90 | -27.30 | 32 | 45780.03 | -46.61 |
| | 256 | 536.90 | -27.30 | 64 | 8279.91 | -39.18 |
| Mitte | 16 | 11723.75 | -40.69 | 4 | 93919.49 | -49.73 |
| | 32 | 5527.12 | -37.43 | 4 | 156825.63 | -51.95 |
| | 32 | 5527.12 | -37.43 | 8 | 85513.08 | -49.32 |
| | 64 | 2363.56 | -33.74 | 8 | 126115.16 | -51.01 |
| | 64 | 2363.56 | -33.74 | 16 | 71372.66 | -48.54 |
| | 128 | 948.08 | -29.77 | 16 | 42281.21 | -46.26 |
| | 128 | 948.08 | -29.77 | 32 | 14551.41 | -41.63 |
| | 256 | 501.98 | -27.01 | 32 | 11827.91 | -40.73 |
| | 256 | 501.98 | -27.01 | 64 | 4716.79 | -36.74 |
| Mitte und Ränder | 16 | 428.97 | -26.32 | 4 | 779.82 | -28.92 |
| | 32 | 343.70 | -25.36 | 4 | 1007.32 | -30.03 |
| | 32 | 343.70 | -25.36 | 8 | 378.16 | -25.78 |
| | 64 | 224.03 | -23.50 | 8 | 531.99 | -27.26 |
| | 64 | 224.03 | -23.50 | 16 | 290.29 | -24.63 |
| | 128 | 156.35 | -21.94 | 16 | 389.75 | -25.91 |
| | 128 | 156.35 | -21.94 | 32 | 258.54 | -24.13 |
| | 256 | 110.43 | -20.43 | 32 | 364.82 | -25.62 |
| | 256 | 110.43 | -20.43 | 64 | 203.95 | -23.10 |

Tabelle 4.4: Vergleich verschiedener Interpolationsgrade und Codebookgrößen bei bi-quadratischer Polynomisierung

4.3 Prädiktionsmethode

Der Klassifikation von Kapitel 3 folgend möchte ich ein überwachtes, datenbasiertes maschinelles Lernverfahren per Klassifikationsbaum durchführen. Wie bereits in Kapitel 3 erwähnt gibt es einfach zu handhabende Tools zur Erstellung von CARTs (z. B. *wagon* des Festival-Sprachsynthesystems, welches auch zum Training von Dauer-Regressionsbäumen im deutschen BOSS-Modul genutzt wird). Die erzeugten Entscheidungsbäume lassen Aufschlüsse über die Wichtigkeit einzelner Prädiktionsmerkmale zu. Darüber hinaus liefern CARTs robuste Ergebnisse, selbst wenn einzelne Unterräume nur spärlich besiedelt sind oder Prädiktionsmerkmale fehlen (sie können approximiert werden, s. Breiman et al. 1984).

Bei einer Polynom-Stilisierung von Silben macht es keinen Sinn, jedes Polynomglied einzeln zu prädictieren, da jede Kurve global optimiert wurde und die einzelnen Koeffizienten keine direkte Interpretation der Kurve zulassen und nur gemeinsam eine eindeutige Form besitzen (im Gegensatz z. B. zum Tilt-Modell, wo jeder Parameter eine direkte Interpretation zulässt). Mit der von mir gewählten Stützstellen-Repräsentation der Polynome wäre eine getrennte Prädiktion zwar möglich, aber ich ziehe den eleganten Ansatz per Vektorquantisierung vor, da er zudem ein linguistisch interpretierbares Codebook liefert.

Der Klassifikationsbaum (oder auch jede andere Prädiktionsmethode) muss daher nicht die einzelnen Parameter prädictieren, sondern nur den am ehesten passenden Codevektor ermitteln. Die Prädiktion ist im Gegensatz zur Intonationsmodellierung stark symbolisch geprägt: Viele Prädiktionsmerkmale (*features*) sind kategorischer / symbolischer Natur (wie z. B. der markierte Ton), einige haben auch kontinuierliche Werte (wie die Silbendauer oder die Anzahl der Silben in der Phrase). Ich werde im Folgenden jedes Einzelne in die Prädiktion eingehende Merkmal besprechen.

Genutzte Merkmale

Die folgende Liste enthält die Merkmale, welche offensichtlich in die Prädiktion einer Tonsprache mit eingehen sollten.

- Der Ton **tone** einer Silbe sollte eine entscheidende Rolle bei der Auswahl der Intonationschablone haben und wird daher als kategorisches Merkmal in fünf Symbolen (H,L,R,F,D) eingehen. Parallel zum segmentalen Kontext, der an den

Phrasenrändern auf ”\$p” (Pause) gesetzt wird, gilt für den tonalen Kontext am Satzrand eine Tonmarkierung **P**. Zusätzlich wird der tonale Kontext in den vorangehenden (**ltone5** .. **ltone1**) und folgenden (**rtone1** .. **rtone5**) fünf Silben festgehalten. Dies ermöglicht eine Modellierung des *start-up* Effektes (Connell nach Urua 2001) und des *final falls* (finales Absinken) im Ibibio. Der *start-up effect* bewirkt, dass die Zielfrequenz eines H-Tones nach einem L-Ton oder am Phrasenbeginn nicht schon innerhalb der tontragenden Silbe erreicht wird, sondern möglicherweise erst in der nächsten oder übernächsten:

”usually this [reaching the High tone target] is attained in the second or third syllable after which the subsequent High tones stabilise” (Urua, 2001, Abschnitt 4.1)

Es ist jedoch zu beachten, dass der *start-up effect* in kurzen HH-Sequenzen ausgesetzt wird: Hier beginnt der Ton sogar etwas über H-Niveau. Dies kann der Lernalgorithmus über den tonalen Kontext unterscheiden lernen.

Initiale und einzelne L-Töne beginnen gewöhnlich auf H-Niveau um danach abzusinken und produzieren damit ähnlich den H-Tönen eine Art *start-up effect*. Um das finale Absinken korrekt zu modellieren, benötigt man ein Fenster von mindestens einem folgenden Kontextton, um das Phrasenende zu erkennen, denn finale L-Töne fallen nach Urua zusätzlich um 10 Hertz ab. Ein auf der linken Seite weit zurückblickendes und wenig vorausschauendes Fenster macht auch insofern Sinn, dass sich tonologische Prozesse im Ibibio immer von links nach rechts (progressiv) auswirken.

- Die Anzahl an vorangehenden Downsteps **d** sollte in die Prädiktion mit eingehen. Nach Urua (2001) setzt der Downstep-Ton **D** eine neue *topline*, die etwa 30 Hz niedriger liegt als die vorangehender H-Töne.
- Ebenso lässt sich Downdrift modellieren: Für H-Töne ist das Merkmal ”Anzahl vorangehender H-L-Wechsel” (ohne Betrachtung der aktuellen Silbe) bestimmt, für L-Töne ”Anzahl vorangehender L-H-Wechsel”. Da die Konturtöne **F** und **R** historisch aus den Tonwechseln H-L und L-H entstanden und mit diesen formgleich sind, wird die Anzahl der entsprechenden Konturtöne auf die beiden Merkmale **f** und **r** mit aufaddiert.

Eine Unterscheidung zwischen Downdrift und Downstep scheint notwendig zu sein, da

”Both the overt Low Tone in a HLH sequence and the floating Low tone in a H!H sequence exerted some degree of lowering. However, there was a difference in the degree of lowering. The overt Low tone seemed to exert a higher degree of lowering than the L_\circ tone judging by the difference in the F0 values.” (Urua, 2001)

(Das Ausrufezeichen kennzeichnet einen *downstep*, während der Kreis einen *floating tone* (einen keinem Segment zugehörigen Ton) bezeichnet.)

- Die Wortart **pos** sollte als binäres Merkmal ”Verb” / ”kein Verb” mit eingehen, da beispielsweise Verben (zumindest in der Grundform) bestimmte Tonschablonen tragen. Diese Daten lagen mir leider bis Abschluss der Arbeit nicht vor.
- Die Position der Silbe im Wort **sylword** ist in Bezug auf die Wortart und die Tonschablonen interessant.
- Die Anzahl der Silben im Wort **sylsword** ist von Bedeutung, um das vorgenannte Merkmal interpretieren zu können.
- Die Anzahl der Silben in der Phrase **sylsphrase** wird benötigt, um den Grad von Downdrift und Downstep zu bestimmen:

”a shorter utterance [...] has a larger pitch difference than a longer utterance (Gibbon, Urua and Gut 2000; Cook 1985) [...] This seems to indicate an overlaid look-ahead component” (Urua, 2001)
- Der erste Konsonant der Silbe **firstcons** wird angegeben, um mikroprosodische Effekte abzudecken:

”the Ibibio data clearly seems to suggest that fundamental frequency values are raised in the vicinity of voiceless consonants, especially strident fricatives” (Urua, 2001, Abschn. 4.6)
- Ebenso der erste Vokal der Silbe **vowel**.

Darüber hinaus habe ich im Laufe der Entwicklung weitere experimentelle Parameter hinzugefügt, welche sich zum Teil als wichtig erwiesen haben. Sie sind der Tabelle 4.5

| T | D | Name | Merkmal | Bereich |
|---|---|---------------|------------------------------------|------------------------|
| | * | tone | Ton der Silbe | H,L,R,F,D (P) |
| * | * | ltone4..1 | Linker Kontextton | H,L,R,F,D (P) |
| * | * | rtone1..3 | Rechter Kontextton | H,L,R,F,D (P) |
| | * | d | # Vorangehende D (downsteps) | 0.. |
| | * | phrasedown | # Downsteps in Phrase | 0.. |
| * | * | f | # H-L + # F (H-downdrift) | 0.. |
| * | * | phrasefall | # H-L - Wechsel + F-Töne in Phrase | 0.. |
| * | * | r | # L-H + # R (L-downdrift) | 0.. |
| | * | phraserise | # L-H - Wechsel + R-Töne in Phrase | 0.. |
| | * | pos | Wortart | V / N (P) |
| | * | sylword | Silbenposition im Wort | 1.. |
| * | * | sylsword | # Silben im Wort | 1.. |
| * | * | firstcons | Erster Konsonant in Silbe | "-" sonst |
| | | vowel | (erster) Vokal in Silbe | "-" sonst |
| * | * | catsylword | Silbenposition im Wort | s,i,m,f |
| | * | catsylphrase | Silbenposition in der Phrase | s,i,m,f |
| * | * | catwordphrase | Wortposition in Phrase | s,i,m,f |
| * | * | sylphrase | Silbenposition in der Phrase | 1.. |
| | * | sylsphrase | # Silben in der Phrase | 1.. |
| * | * | wordphrase | Wortposition in Phrase | 1.. |
| * | * | wordsphrase | # Wörter in der Phrase | 1.. |
| - | - | cb | Codevektor der Silbe | 0..63 |
| - | * | cbc | Codevektor-Klasse | 0..15 |
| - | | lastcb | Codevektor der linken Silbe | 0..63 |
| - | * | lastcbc | Codevektorklasse der linken Silbe | 0..15 |
| * | | phonessyl | # Phone in der Silbe | 1.. |
| | - | dur | Dauer der Silbe | in [ms] |
| * | * | sylstruc | Silbenstruktur | (C, V, N) ⁺ |
| | | bodil | Distanz zum Beginn der Phrase | 0.. |
| | * | bodir | Distanz zum Ende der Phrase | 0.. |

Tabelle 4.5: Genutzte Prädiktionsmerkmale für das Ibibio-Intonationsmodul. T - Nutzung im Tonschablonen-CART, D - Nutzung im Dauer-CART

angefügt. Wie man in der Tabelle erkennt, existieren für einige numerischen Maße mit Werten $\in \mathbb{N}$ auch kategoriale Merkmale, die nur in *initial*, *medial*, *final* und *single* unterschieden werden. Das gilt für die Silbenposition im Wort **catsylword**, Silbenposition in der Phrase **catsylphrase** und die Wortposition in der Phrase **catwordphrase**.

Von der Prädiktion ausgeschlossen werden alle Parameter-Vektoren, die entweder den Pausenton **tone P**, den Pausen-Codebookeintrag **cb -1** oder eine unmögliche Silbenstruktur **sylstruc** (z. B. **CCCC**, **CVCV** o.ä.) tragen. Darüber hinaus werden Silben ausgeschlossen, denen zwar keine Pause vorangeht, aber deren vorangehende Codevektorklasse **lastcbc** mit **-1** unbestimmt ist (notwendig für die Dauerprädiktion, die die vorangehende Tonschablonenklasse berücksichtigen darf).

4.4 Einheitenauswahl

In einem Sprachsynthesystem basierend auf *non-uniform unit selection NUU* steht und fällt die Synthesequalität mit einer sinnvollen Einheiten(vor-)auswahl. Die Auswahl und Gewichtung der Kriterien stellt entscheidende Anforderungen an das phonetische und phonologische Verständnis der zu synthetisierenden Sprache. Da die Prädiktion nur einen Codevektor liefert, reicht es prinzipiell, ein silbisches Segment mit dem passenden Codevektor aus dem Korpus zu extrahieren. Da jedoch nicht jede segmentale Realisierung aufgrund von Datenknappheit und intonatorischen Randbedingungen auch in allen Codevektor-Variationen vorkommt, muss die Einheitenauswahl ihren Fokus erweitern können. Dies geschieht bei BOSS sukzessive durch Wegfallenlassen der obligatorischen Attribute in der Kandidatenvorauswahl.

Um das zu erleichtern, werden alle Intonationsschablonen noch einmal vektorquantisiert, um Schablonenklassen zu erhalten, denen jeder Codevektor eindeutig zuordenbar ist.

4.4.1 Vorauswahl

Die Einheitenvorauswahl findet in BOSS stufenweise statt: Beginnend auf Wortebene wird per **mySQL** (<http://www.mysql.com>, 2006-08-16) eine Suchanfrage (engl. *query*) an die Datenbank abgeschickt (**mySQL**; DuBois, 2006). Liefert die Anfrage keine Daten zurück, so wird der Suchkontext erweitert oder - bei weiterhin leerem Suchergebnis - auf Silbenebene gesucht. Eventuell wird noch auf der Phonebene gesucht. Halbphone sind

in der Ibibio-Version für BOSS bisher nicht vorgesehen. Für das Ibibio schlage ich die in Tabelle 4.6 genannten Vorauswahlparameter vor. Die einzelnen Parameter beziehen

| Ebene | Übereinstimmende Attribute |
|--------|---|
| Wort | TReal, Tones, Broken = 0, PMode |
| Silben | TReal, Tone, CCLeft2, LTone1, Border = 0, CB..CBC, CCRight2, RTone, PMode |
| Phone | TReal, PhPos, Tone, CB .. CBC, CCLeft2, LTone1, CCRight2, RTone |

Tabelle 4.6: Konfiguration der Einheitenvorauswahl.

sich auf XML-Attribute bzw. Tabelleneinträge in der mySQL-Datenbank und werden in Kapitel 5 erläutert.

4.4.1.1 Aufweitung des Kontextes

Um ein sukzessives Zurücknehmen von obligatorischen Attributen zu ermöglichen, existiert ein einfacher Mechanismus in BOSS. In einer Textdatei für jede Ebene der Vorauswahl (Wort, Silbe, Phon) stehen zeilenweise benötigte Attribute und evtl. ein geforderter Wert. Mehrere solcher Attribut-Listen sind als Blöcke mit Leerzeilen voneinander getrennt und repräsentieren jeweils eine Aufweitung des Kontextes. Ich habe ein Tool geschrieben, welches eine einfachere Beschreibung der Zurückstufung obligatorischer Attribute ermöglicht und die entsprechenden BOSS-Dateien automatisch erstellt. Mehr dazu in Anhang D.

4.4.2 Einheitenauswahl

Aus den Listen möglicher passender Elemente, welche die Vorauswahl von der Datenbank erhält, wird ein Suchgraph erstellt. Der genaue Vorgang wurde schon an vielen Stellen beschrieben, hier sei nur auf Black und Campbell (1995) verwiesen.

Unverändert zur deutschen BOSS-Einheitenauswahl geht die Grundfrequenzdifferenz und die MFCC-Differenz an den Einheitenrändern auf allen Ebenen in die Verkettungskosten (*transition cost*, *join cost*) mit ein. Die Einheitenkosten (*unit cost*) errechnen sich als Summe von ebenenabhängigen Einzelkosten, die im Folgenden beschrieben werden.

4.4.3 Verkettungskosten

Die Gesamtkosten für N Kostenterme berechnen sich für zwei aufeinander folgende Einheiten l und r wie folgt:

$$Cost_{Transition}(l, r) = \frac{1}{N} \sum_{i=1}^N w_i * Cost_i(l, r) \quad (4.1)$$

mit den beiden Kostentermen für den euklidischen Abstand der *Mel frequency cepstral coefficients MFCC*

$$Cost_1(l, r) = Cost_{MFCC}(l, r) = \sqrt{\sum_{j=1}^{CC} (MFCC_j(l) - MFCC_j(r))^2} \quad (4.2)$$

und der absoluten Grundfrequenzdifferenz

$$Cost_2(l, r) = Cost_{F_0}(l, r) = |F_0(l) - F_0(r)| \quad (4.3)$$

4.4.4 Einheitenkosten

Die Gesamtkosten zwischen prädiziertem Wert p und einer Korpuseinheit r im Auswahlgraphen für N Kostenterme

$$Cost_{Unit}(p, r) = \frac{1}{N} \sum_{i=1}^N w_i * Cost_i(p, r) \quad (4.4)$$

setzen sich zusammen aus sechs gewichteten Kostentermen, wobei

$$Cost_1 = Cost_{Dur}(p, r) = \frac{|Dur(p) - Dur(r)|}{PNr} \quad (4.5)$$

die auf *ms pro Phon* normalisierte absolute Dauerdifferenz ist (Normalisierung notwendig für Silben- und Worteinheiten),

$$Cost_2 = Cost_{Phrase}(p, r) = \begin{cases} 0, & \text{wenn } PMode(p) = PMode(r) \\ 2, & \text{wenn } PMode(p) = \text{""} \text{ und } PMode(r) \neq \text{""} \\ 1, & \text{wenn } PMode(p) \neq PMode(r) \end{cases} \quad (4.6)$$

Abweichungen vom Satzmodus genau wie im deutschen BOSS bestraft und

$$Cost_3 = Cost_{Tone}(p, r) = \begin{cases} 0, & \text{wenn } Tone(p) = Tone(r) \\ 2, & \text{wenn } Tone(p) - Tone(r) = 0 \\ |Tone(p) - Tone(r)|, & \text{sonst} \end{cases} \quad (4.7)$$

$$Tone(x) = \begin{cases} 0.0, & x = "L" \\ 1.0, & x = "R" \\ 1.0, & x = "F" \\ 1.5, & x = "D" \\ 2.0, & x = "H" \end{cases} \quad (4.8)$$

ein Maß zur Bestrafung von falschen Tönen auf der Silbenebene liefert. Zusätzlich zu den Verbindungskosten aufgrund von Grundfrequenzdifferenzen füge ich für Wörter und Phone einen Kostenterm für die *Einheitenkosten* aufgrund der Grundfrequenzdifferenz hinzu. Dies macht Sinn, da die absolute Tonhöhe im Ibibio als Spiegelbild einer bestimmten Tonschablone auf der Silbenebene linguistisch signifikant sein kann.

Für Silben werden direkt die Stützstellen des prädizierten Codevektors mit denen des Stilisierungspolynoms der Korpuseinheit verglichen und ein Fehlermaß auf Basis des RMS der Stützstellen errechnet:

$$Cost_4 = Cost_{Basic_values}(p, r) = \sqrt{\frac{1}{n} \sum [Basic_value_i(p) - Basic_value_i(r)]^2} \quad (4.9)$$

Für Wörter wird nur die Grundfrequenz am linken und rechten Rand, abgeleitet aus den Stützwerten der konstituierenden Silben, verglichen:

$$Cost_5 = Cost_{F0}(p, r) = \frac{|\Delta F0_l(p, r)| + |\Delta F0_r(p, r)|}{2} \quad (4.10)$$

Auf der Phonebene gibt es noch einen zusätzlichen Kostenterm für die kategoriale

Position des Phons (*initial*, *medial*, *final* und *single*) innerhalb der Silbe:

$$Cost_6 = Cost_{PhPos}(p, r) = |PhPos(p) - PhPos(r)| \quad (4.11)$$

$$PhPos(x) = \begin{cases} 0.0, & x = \text{"i"} \\ 1.0, & x = \text{"m"} \\ 1.0, & x = \text{"s"} \\ 2.0, & x = \text{"f"} \end{cases} \quad (4.12)$$

Die Tabelle 4.7 zeigt, welche Kostenterme mit welchen Normalisierungen und Gewichtungen auf welchen Ebenen genutzt werden.

| Kostenart | Term | Wort | Silbe | Phon |
|------------|----------------|---------------------|---------------------|----------------|
| Verkettung | MFCC | ... | $\frac{1}{26}$ | ... |
| | F ₀ | ... | $\frac{1}{238.745}$ | ... |
| Einheit | Dur | $\frac{1}{60}$ | $\frac{1}{60}$ | $\frac{1}{60}$ |
| | Phrase | $\frac{1}{2}$ | $\frac{1}{2}$ | - |
| | Tone | - | $\frac{1}{1}$ | - |
| | BV | - | $\frac{1}{50}$ | - |
| | F ₀ | $\frac{1}{238.745}$ | - | - |
| | PhPos | - | - | $\frac{1}{1}$ |

Tabelle 4.7: Gewichtungen (Zähler) und Normalisierungen (Nenner) der Kostenterme.

4.5 Behandlung von Dauer-Phänomenen

Es zeigt sich, dass die F₀-Differenz bei Downsteps und Downdrift von der Phrasenlänge abhängt. Da ich völlig auf eine Modellierung einer Phrasen-Ebene verzichten möchte, muss solche Information wie z. B. Phrasenlänge in die Prädiktion mit eingehen, damit das maschinelle Lernen Generalisierungen bilden kann. Die Dauer-Prädiktion benutzt die selben Parameter wie die Codevektor-Prädiktion und liefert die Information, wie lang jede Silbe sein soll, und die Polynom-Schablone wird auf diese Länge angewendet. Der Vorteil der Polynomstützstellen liegt auch darin, keine Normalisierung der Länge vornehmen zu müssen, da die Position der Stützstellen immer relativ zur Dauer der

Silbe bestimmt werden und einer Normalisierung entsprechen. Dauernormalisierungen wurde auch schon für das PaIntE-Modell angewendet: *”The syllable length is defined as unity”* (Möhler und Conkie, 1998) oder

”we modify the time stamps of the F0 values extracted from the selected units linearly. Thus, the extracted portions of F0 curves are stretched or contracted to fit the duration of the segments.” (Raux und Black, 2003)

4.6 Kontextklassen

Die Phone des Ibibio wurden in Klassen gleichen Anregungsortes grob kategorisiert. Diese Information wird in der Vorauswahl als Parameter `CCLeft2` und `CCRight2` genutzt, um die Suche eines Phons im segmentalen Kontext auf Kontexte mit gleichem Artikulationsort aufzuweiten. Tabelle 4.8 listet alle Phone des Ibibio mit ihrem Artikulationsort auf. Die Tabelle enthält das in BOSS genutzte Symbol für den Artikulationsort, der vol-

| Symbol | Ort | IPA | SAMPA | Orthographie |
|--------|----------------|-------------------|-----------|--------------|
| PAU | Pause | | \$p | |
| FRO | Vordervokal | i,i,e | i,I,e | i,i,e |
| CEN | Zentralvokal | ə,a | @,a | ə,a |
| BAC | Hintervokal | o,ɔ,u,ʊ,ʌ | o,O,u,U,V | o,ɔ,u,ʊ,ʌ |
| BIL | Bilabial | b,m,p,w | b,m,p,w | b,m,p,w |
| BIL* | Bilabial-velar | \widehat{kp} | kp | kp |
| LAB | Labial | f | f | f |
| DEN | Dental | $\underset{̣}{t}$ | t | t |
| ALV | Alveolar | d,n,r,s | d,n,r,s | d,n,r,s |
| PAL | Palatal | ɲ,j | J,j | ɲj,y |
| VEL | Velar | g,k,ŋ | g,k,N | g,k,ng |
| GLO | Glottal | h | h | h |

Tabelle 4.8: Phone des Ibibio mit ihren Artikulationsorten

le Name des Ortes sowie die Transkription aller entsprechend kategorisierten Phone in IPA- (IPA, 1999) und Sampanotation (<http://www.phon.ucl.ac.uk/home/sampa/home.htm>, 2006-08-16). Im rechten Kontext gilt \widehat{kp} als velar, sonst als bilabial (BIL*). /w/ gilt wie im deutschen BOSS-Modul immer als bilabial.

5 Implementation

Als Ausgangspunkt zur Entwicklung der Ibibio-Synthesemodule diente die deutsche BOSS-Version. Folgende Teile des Systems mussten verändert oder komplett neu geschrieben werden:

- Die CART-Klasse (`boss_cartreader/`)
- Das Intonationsmodul (`boss_intonation/`)
- Das Einheitenauswahlmodul (`boss_unitselection/`)
- Einige Grundklassen (`Boss_Node`, `boss_utility/dom_tools`)
- Die meisten Tools (`tools/*`)

Als grundsätzliches Prinzip wurde festgesetzt, möglichst wenig Klassen der deutschen BOSS-Distribution zu verändern und - wo immer möglich - Ableitungen (Vererbungen) bestehender Basisklassen durchzuführen. Dies gelang fast immer, mit folgender entscheidender Ausnahme: Die Klasse "BOSS_Node" musste leicht verändert werden, da sie den zentralen Drehpunkt in der gesamten Einheitenauswahl ausmacht und noch zu sehr auf die Merkmale der deutschen Sprache zugeschnitten war (hier besonders das Attribut `stress` für den Wortakzent. In kommenden Versionen von BOSS wird die Sprachunabhängigkeit des System noch steigen). Da in der Ibibio-Version von BOSS die Bedeutungen der einzelnen Einheiten Ebenen anders gewichtet sind als im Deutschen (kleines Korpus, keine Halbphone, Silben als wichtigste Einheit), müssen an einigen Stellen Verzweigungen im Quelltext geändert werden.

In diesem Kapitel möchte ich die erstellten Tools, die Struktur der Datenbank und wichtige Methoden des Quelltextes vorstellen.

5.1 Das Korpus

Die Aufbereitung erfordert viele Schritte, und viele Aufgaben müssen manuell erledigt werden. Ich möchte nur einen kurzen Überblick über die wichtigsten Schritte geben, die nötig sind, sobald ein annotiertes, segmentiertes und vereinheitlicht formatiertes Korpus vorliegt. Eine eingehendere Dokumentation der einzelnen Tools findet sich in Anhang D.

- Die Sprachdaten werden mit Hilfe des Tools `sox` (<http://sox.sourceforge.net>, 2006-08-16) in das Format 16 kHz, Mono, 16 bit konvertiert. Die Segmente und Annotationen liegen als Praat-TextGrid-Dateien vor (<http://www.fon.hum.uva.nl/praat/>, 2006-08-16). Jeder Satz wird - nach BOSS-Konvention - in einer eigenen Datei abgespeichert. Zu den 165 Sätzen des Ibibio-Korpus lagen mir allerdings nur 94 segmentierte und annotierte TextGrids vor.
- Mit Hilfe von `find-avg-f0.pl` wird die mittlere Grundfrequenz bestimmt, die später für die Normalisierung der Gewichtungsfaktoren der Kostenterme der Einheiten Auswahl benötigt wird.
- Es wird für jeden Satz alle 10 ms die Grundfrequenz mit Hilfe des Tools `get_f0` des ESPS-Paketes bestimmt (<http://computing.ee.ethz.ch/sepp/esps-5.3.1-vj>), in ein einfaches Dateiformat umgewandelt (`prune-esps.pl`, `esps2f0.pl`) und silbenweise extrahiert (`extract_syl_f0s.psc`). Die Tatsache, dass sich Silbengrenzen nie exakt an den Positionen der *frames* befinden (100 *frames* pro Sekunde bzw. ein Messwert je 10 ms) ist relativ unerheblich, da jeder einzelne F_0 -Wert selbst über ein Fenster mit Länge > 10 ms bestimmt wird und verfahrensbedingt kein Wert einem exakten lokalen Wert entspricht.
- Mit `extract-pulses.psc` werden die Verschlusszeitpunkte der Glottis bestimmt.
- Die TextGrids werden mit annotierten Silben- und Wortsegmenten ergänzt (`add-syls.psc` und `add-words.psc`).
- Für jede Silbe wird die Polynomregression durchgeführt (`polyreg.pl`). Dabei kann es vorkommen, dass für einzelne Silben keine Polynomregression durchgeführt werden kann, weil das zu Grunde liegende System aus Zeit/ F_0 -Paaren unlösbar ist oder unerlaubte Werte enthält.

- Mit Hilfe von `loo.pl` werden Testsätze aus dem Korpus extrahiert und von der weiteren Verarbeitung ausgeschlossen.
- Es wird ein Codebook aller Polynome berechnet und eine Codevektor-Klassifizierung vorgenommen (`vq.pl`).
- Die Prädiktionsmerkmale werden extrahiert und ein CART mit dem *wagon*-Tool damit trainiert (`extract-params.psc`, `filter.pl`).
- Die Annotationsdaten werden mit `tg2xml.pl` in das XML-Format umgewandelt.
- Erzeugen der leeren MySQL-Datenbank mit `create_tables.sql`.

Dann werden die Standard-BOSS-Tools auf die XML-Dateien angewendet um weitere Attribute hinzuzufügen:

- Segmentale Kontextinformationen inkl. des Tonkontextes (`addcontext`)
- Grundfrequenzwerte auf allen Ebenen (`addf0`)
- Angleichen der Segmentgrenzen an Nulldurchgängen (`optbounds`)
- Mel-Cepstral-Koeffizienten an den Segmentgrenzen auf allen Ebenen (`melbounds`)
- Schreiben der XML-Dateien in die MySQL-Tabellen (`blfxml2db`).

5.2 XML-Format

5.2.1 Vereinbarungen für den Client

Da das Intonationsmodul parallel zur Textvorverarbeitung anderer Projektteilnehmer entwickelt werden sollte, musste eine Vereinbarung darüber getroffen werden, wie Ton-Information in der Übertragung zwischen Client und Server kodiert werden soll. Das XML-Dokument, welches der Client an den Server schickt, enthält die Attribute aus Tabelle 5.1. Da BOSS primär auf der Wortebene arbeitet und erst dann auf Silben-, Phon- oder gar Halbphonebene nach Einheiten sucht, wenn ein gesuchtes Wort nicht im Korpus existiert, müssen die silbischen Töne innerhalb eines Wortes markiert werden. Dazu wird parallel zum XML-Attribut "TKey" (für die kanonische Transkription) ein Attribut "Tones" eingeführt, welches alle Töne als eine Zeichenkette repräsentiert. Die Länge der Zeichenkette entspricht damit direkt der Anzahl der Silben im Wort.

| Ebene (tag) | Attribut | Beschreibung |
|-----------------------|-----------------|---|
| Sentence | Type | Aussage "." oder Frage "?" |
| Word, Syllable, Phone | TReal TKey | Transkription kanonische Transkription |
| Word | Tones Broken | Kette aller Silbentöne 0: Wortgrenze ist auch Silbengrenze, 1: sonst |
| Syllable, Phone | Tone | Ton der Silbe (H,L,R,F,D,P) |
| Syllable | Border | 1: Wortgrenze innerhalb der Silbe, 0: sonst |

Tabelle 5.1: Attribute des Transfer-XML-Formats

5.2.2 Vereinbarungen für den Server

Für die Erstellung der Datenbank und die interne Kommunikation der einzelnen Module des BOSS-Servers wird ein eigenes XML-Format definiert. Es entspricht weitestgehend dem des Client, hat aber einige Attribute mehr, welche in Tabelle 5.2 aufgelistet sind.

Das letzte Wort und die letzte Silbe innerhalb einer Phrase (d. h. hier innerhalb eines Satzes) erhält das Attribut **PMode** mit dem Wert des **Type**-Attributes des **sentence**-tags (momentan nur "." oder "?"). Theoretisch könnte das Intonationsmodul auch innerhalb eines Satzes Phrasengrenzen über das **PMode**-Attribut definieren. Dies ist jedoch eine Information, die der Client liefern müsste, der zum Abgabetermin dieser Arbeit noch nicht existierte. Jede nichtfinale Wort- und Silbeneinheit erhält den leeren Wert "" für **PMode**. Das Attribut **PInt** bestimmt die Stärke des Grenztons, wobei der Wert 5 entsprechend der deutschen BOSS-Version für Aussagesätze festgesetzt wurde. Alle anderen Einheiten bekommen den Wert 0; auf Phonebene sind **PMode** und **PInt** nicht definiert. In der Ibilio-Implementation für BOSS wird zwar nur die Transkription **TReal** der tatsächlichen lautlichen Realisierung benötigt, aber die kanonische Transkription **TKey** enthält vorerst den selben Wert wie **TReal**, weil viele Klassen des BOSS dieses Attribut auf Inhalt prüfen und umgeschrieben werden müssten. Durch die simple Kopie des Wertes entsteht zwar eine Menge redundanter Daten in der Datenbank, die Entwicklung wird aber erheblich erleichtert. Hier kann man später bei einer expliziten Optimierung ansetzen.

Weitere Attribute werden in der internen Modul-Kommunikation, besonders zwischen Intonations- und Einheitenwahlmodul, benötigt. Sie sind in Tabelle 5.3 zusammen-

| Ebene (tag) | Attribut | Beschreibung |
|-----------------------|---------------------------------------|---|
| Sentence | File Type | Dateiname der Wave-Datei (nur Korpus) Aussage "." oder Frage "?" |
| Word, Syllable, Phone | TReal TKey First Last | Transkription kanonische Transkription Samplenummer des linken Einheitenrandes Samplenummer des rechten Einheitenrandes |
| Word, Syllable | PMode PInt PNr | Grenzton ".", "?", "" Stärke der Grenze (0 oder 5) Anzahl Phone in Einheit (1..) |
| Word | Tones Broken | Kette aller Silbentöne 0: Wortgrenze ist auch Silbengrenze, 1: sonst |
| Syllable, Phone | Tone CB CBC BV0..4 PC0..4 | Ton der Silbe (H,L,R,F,D,P) Codebook-Eintrag (Codevektor, -1..63) CB - Klasse (-1..7; -1: Pause) 5 Stützstellen der Silbe v.l. n. r. 5 Polynomkoeffizienten nach Grad |
| Syllable Phone | Border PhPos | 1: Wortgrenze innerhalb der Silbe, 0: sonst Kategoriale Position in Silbe (i,m,f,s) |

Tabelle 5.2: Attribute des Korpus-XML-Formats

gefasst.

| Ebene (tag) | Attribut | Beschreibung |
|-----------------|----------|--|
| Syllable | LTone1 | Ton der vorangehenden Silbe |
| | RTone | Ton der folgenden Silbe |
| Syllable, Phone | CLeft | Phon der vorangehenden Silbe |
| | CRight | Phon der folgenden Silbe |
| | CCLeft | Randsegment der vorangehenden Silbe |
| | CCRight | Randsegment der folgenden Silbe |
| | CCLeft2 | Phonklasse des vorangehenden Randsegments |
| | CCRight2 | Phonklasse des folgenden Randsegments |
| Word | Dur | Dauer des Wortes, $Dur_{Word} = \sum_i Dur_{Syllable}(i)$ |
| Phone | Dur | Dauer des Phones, $Dur_{Phone} = \frac{Dur_{Syllable}}{PNr}$ |

Tabelle 5.3: Attribute des internen Server-XML-Formats

5.3 Quelltext

5.3.1 CART-Reader

Das Dauerprädiktionsmodul der deutschen BOSS-Version kann nur Regressionsbäume einlesen, an deren Blättern jeweils Standardabweichung und Mittelwert eingetragen sind. Für den Klassifikationsbaum müssen an jedem Blatt eine Liste aus 64 Wahrscheinlichkeiten übergangen und die wahrscheinlichste Klasse eingelesen werden. Zusätzlich ist das neue Modul in der Lage, nicht nur kategorische Entscheidungen (**feature is element**), sondern auch numerische Vergleiche (z. B. **feature < value**) durchzuführen.

5.3.2 Intonationsmodul

constructor Bei der Initialisierung des Intonationsmoduls werden zuerst das Codebook und die Codebook-Klassifikationsdatei der Größe `cbsize * (order+1)` über die Parameter `codebook` und `codebook2` entsprechend den Angaben in der Konfigurationsdatei eingelesen. Dann werden der Klassifikationsbaum für die Tonschablonen `tonecart` und der Regressionsbaum für die Dauerprädiktion `durcart` geladen.

destructor Der Destruktor entfernt den allozierten Speicher der Codebooks und der Entscheidungsbäume.

entry point Der Aufruf des Intonationsmoduls erfolgt über den überladenen Operator (). Hier wird die Methode `setFeatures` aufgerufen, welche die Parameter für die CART-Bäume bestimmt, die Entscheidung durchführt und die Ergebnisse in das interne XML-Dokument schreibt. Daraufhin werden die temporär angelegten Parameter über `removeFeatures` wieder entfernt und der Speicher wieder freigegeben.

setFeatures Das Vorgehen der Prädiktionsparameterbestimmung ist das gleiche wie das des Tools `cart/extract-params.psc`, nur dass im Intonationsmodul immer nur ein Satz bearbeitet wird. Es wird eine Liste aller Silben aus der XML-Struktur extrahiert und dann silbenweise mit Parametern versehen. Diese werden jedoch nur insoweit in der XML-Struktur selbst gespeichert, wie es für die Einheiten(vor)auswahl wichtig ist; die meisten Parameter werden ausschließlich von den Entscheidungsbäumen gebraucht und werden daher in einer effizienten Datenstruktur unter dem Zeiger `*tmp` abgelegt (`map` aus der STL¹). Die extrahierten Parameter entsprechen denen aus Tabelle 4.5, wobei nie alle gleichzeitig von den trainierten CARTs genutzt werden, aber für weitere Entwicklungen zur Verfügung stehen. Nach Bestimmung der Parameter wird zuerst die Dauerprädiktion über die Methode `dur_tree→RegrClassify(*tmp)` und dann mit dem zusätzlich gewonnenen Parameter `dur` die Tonschablonenprädiktion per `tree→Classify(*tmp)` aufgerufen.

Nachdem die Codevektoren bestimmt sind, werden die entsprechenden Stützstellen aus dem Codebook extrahiert und an den Silbenrändern durch einfaches Mittelwertbilden einander angeglichen. Das hat zwar noch keine Auswirkung auf die tatsächliche Grundfrequenzsynthese, sondern nur auf die Einheitenauswahl. Aber von der Vorauswahl werden erst einmal nur solche Einheiten gesucht, die den korrekten Codevektor besitzen. Unabhängig von der Menge der gefundenen Einheiten werden danach von der Einheitenauswahl diejenigen über den Stützstellen-Kostenterm bevorzugt, deren Stützstellen eher den angeglichenen Werten der Silbe entsprechen.

Einzelne Parameter werden von der Silben- auf die Phon- und Wortebene propagiert: Die Dauer der Einheiten `dur` und der Parameter `phonessyl` (Anzahl der Phone in der

¹Standard template library: Plauger et al. (2000)

Silbe) werden aufaddiert. Diese Propagierung macht die Methode etwas unübersichtlich, langsamer und redundant, was aber für meine Ergebnisse keine Rolle spielt.

5.3.3 Einheitenauswahlmodul

constructor Als erstes werden die einzelnen Ebenen der Einheitenauswahl erstellt: Über den Aufruf `new UnitLevel ([...], preselection-file, UnitLevel next)` wird eine neue Ebene angelegt und mit der niedrigeren Ebene `next` verknüpft. Wenn es keine niedrigere Ebene gibt, wird `NULL` übergeben.

destructor Im Destruktor werden alle erstellten Ebenen und die Kostenfunktionen wieder entfernt.

entry point Wie in allen BOSS-Modulen ist der Modul-Aufrufpunkt der überladene Operator `()`. Hier wird als erstes die Vorauswahl über die Methoden `unitSelect` und `PreSelection` durchgeführt. Daraufhin wird der Einheiten-Graph über die Methode `createUnitGraph` erstellt und mit virtuellen (d. h. leeren) Start- und Endknoten versehen. Dann wird der Kürzeste-Pfade-Algorithmus aufgerufen, das *backtracking* des besten Pfades durchgeführt, die gefundenen Einheiten in die XML-Struktur geschrieben und das Modul beendet.

unitSelect Die Methode ruft die Vorauswahl für eine bestimmte Ebene auf und ruft sich selber auf der nächstniedrigeren Ebene rekursiv auf, wenn für die Suchanfrage keine Ergebnisse existieren. Die Vorauswahl in der hier aufgerufenen Methode `PreSelection` schickt eine `mySQL`-Anfrage an die Datenbank und liefert das Ergebnis zurück.

createUnitGraph Diese Methode legt für jede gesuchte Einheit eine Liste (`vector` in der STL) aller im Korpus verfügbaren Einheiten an. So entsteht ein zweidimensionaler Graph mit allen zu untersuchenden Einheiten, durch den der beste Pfad gefunden werden muss.

5.3.4 Signalmanipulation

Aufgrund der aus dem Codebook extrahierten und an den Einheitenrändern angeglichenen Stützwerte wäre eine Signalmanipulation mit *pitch synchronous overlap add*

(*PSOLA*) möglich. Eine Entwicklung des entsprechenden Moduls war weder Thema der Magisterarbeit noch war genügend Zeit vorhanden, um mehr als einige Tests zu unternehmen. Dennoch möchte ich eine kurze Richtlinie geben, wie die Verarbeitung implementiert werden könnte:

- Da das Intonationsmodul nicht weiß, welche Einheiten das Einheitenauswahlmodul endgültig für die Synthese auswählt, können die Stützwerte nur vor der Auswahl an den Silbenrändern gemittelt werden. Bevor die eigentliche Signalmanipulation stattfindet, muss daher für jede ausgewählte Einheit gleich welcher Ebene die Intonationskontur berechnet und geglättet werden. Dabei müssen Phone ihrer tatsächlichen Dauer entsprechend als proportionale Anteile der gesamten prädizierten Silbenlänge betrachtet werden. Für Wörter wird die gesamte Kontur aus den einzelnen Silben- und Phonkonturen zusammengesetzt.
- Die gesamte Menge aller prädizierten F_0 -Werte wird den jeweiligen Einheiten zugeordnet. Mit dieser Information kann das BOSS-Modul *conman* (*concatenation and manipulation*) dann das Sprachsignal manipulieren.

Eine Signalmanipulation kann sicherlich die Synthesequalität verbessern. In einem NUU-System ist aber ein großes, ausgewogenes und vielseitiges Korpus weitaus wichtiger als Signalmanipulation. Jede rechnerische Veränderung des Originalsignals kann Störungen einbringen und die Sprache unnatürlich klingen lassen.

6 Evaluation

6.1 Quantitative Evaluation

6.1.1 Die Vektorquantisierung

In die Vektorquantisierung gehen 2322 gültige Silben aus 94 Sätzen ein. Ausgenommen sind Pausen und Polynome, deren Stützstellen extreme Werte besitzen (willkürlich auf den Bereich von -4000..4000 Hz festgesetzt). Das Codebook wird mit einer Größe von 64 Codevektoren erzeugt und besitzt eine *distortion* von 224.03 bzw. ein Signal-zu-Rausch-Energieverhältnis (SNR) von -23.50. Das Codebook findet sich in Tabelle C.1 in Anhang C.

Das Codebook wird wiederum selbst vektorquantisiert und die so erhaltenen Codevektoren nenne ich (Tonschablonen-) Klassen. Dass die SNR hier mit -24.63 (*distortion* 290.29) schlechter ist verwundert nicht, da die Datenpunkte der zweiten VQ im 5-dimensionalen Raum die Codevektoren der ersten Vektorquantisierung sind, die bereits eine optimale Aufteilung des Datenraumes repräsentieren. Dennoch kann diese einfache Herangehensweise eine gewisse Verbesserung in der Einheitenvorauswahl ermöglichen.

6.1.1.1 VQ-Besprechung

Bei der VQ kann es passieren, dass ein Codevektor nur Nullwerte besitzt und nicht genutzt wird. Dies liegt an der Eigenheit des VQ-Algorithmus, dass während der Vektorquantisierung in jeder Iteration alle Datenvektoren zu den neu gebildeten Codevektoren zugeordnet werden. Fällt nach einer solchen Partitionierung in einen Codevektor kein einziger Datenwert mehr, so bleibt dieser Codevektor in allen folgenden Aufteilungen des Vektorraums komplett unbenutzt und erzeugt bei jeder folgenden Teilung (*split*) zwei weitere leere Codevektoren und gilt als verloren. In gewissen Datenkonstellationen kann es somit passieren, dass ein Großteil des Vektorraums ungenutzt bleibt und die

Separierung der verschiedenen *cluster* ungenügend funktioniert und der Datenverlust sehr hoch wird.

Bei Betrachtung des Codebooks kann man die Codevektorpolynome allein schon optisch in Klassen zusammenfassen. Gewöhnlich kann man bei jeweils zwei oder vier benachbarten Kurven Ähnlichkeiten entdecken und die durchschnittliche Grundfrequenz der Intonationskurven steigt mit der Codevektornummer. Daran kann man die Funktionsweise der Vektorquantisierung wiedererkennen: Jeweils zwei aufeinander folgende Codevektoren tragen immer kleinere und größere Werte, da bei jeder Vektorraumaufteilung zwei neue Codevektoren um den jeweils alten Datenpunkt der letzten Iteration aufgebaut werden. Gleiches gilt nicht nur für zwei aufeinander folgende Codevektoren, sondern auch wenn man zwei aufeinander folgende Blöcke von 2, 4, oder 2^n Codevektoren miteinander vergleicht. Scheinbar geht die Form des Polynoms weniger in die Vektorquantisierung mit ein als dass vielmehr die mittlere Grundfrequenz über die Zugehörigkeit zu einer der Tonschablonenklassen entscheidet. Im Codebook gibt es nur eine einzige deutliche Talkontur (CV 1), wodurch eine Stilisierung mit Hilfe der anderen parametrischen Modelle ebenso denkbar wäre.

Bevor die Entscheidung für die Interpolation stimmloser Silbenabschnitte gefällt wurde, gab es viele Codevektoren mit extremen Steigungsgraden und sehr hohen und tiefen Werten weit außerhalb der menschlichen Grundfrequenzspanne. Die Auswahl eines solchen Codevektors mit absurden Werten wie einer Grundfrequenz < 0 Hz oder > 600 Hz ist nicht von vornherein als falsch zu verurteilen. Solche Codevektoren rühren daher, dass viele Silben des Korpus mit einem stimmlosen Konsonant beginnen und die F_0 -Werte im Anlaut der Silbe undefiniert sind. Daher hat die Polynomregression hier die Freiheit, extreme Werte zuzulassen, um die vorhandenen Stützwerte besser approximieren zu können. In der Einheitenauswahl schadet die Auswahl eines solchen Codevektors auch nicht, nur in der Signalmanipulation kann es möglicherweise Probleme geben, da es im Korpus stimmhafte Signalabschnitte geben kann, für die aufgrund der Polynomform kein (realistischer) F_0 -Wert prädiziert wird. Daher ist eine Interpolation sehr sinnvoll.

6.1.2 CART-Prädiktion

Da aufgrund der momentanen Programmstruktur keine vollautomatische *leave-one-out*-Analyse möglich war, wurden dreimal ca. 10% aller Silben aus dem Korpus entfernt und mit den restlichen Sätzen ein CART trainiert. Die Ergebnisse sind in Tabelle 6.1

zusammengefasst. Die Entscheidungsbäume wurden mit 1695-1748 Datenvektoren zu je 40 Parametern und einem Stopwert von 10 schrittweise trainiert (Funktion `stepwise` des `wagon`-tools). Zum Training wurden 84 der 94 annotierten Sätze genutzt (leave-one-out von 10.6 % bzw. 8.7 - 11.4 % der Silben). Ein Stopwert von 10 bedeutet, dass eine weitere Klassifikation ausgesetzt wird, sobald ein Blatt des Entscheidungsbaums 10 Datenvektoren abdeckt. Bei kleinen Werten kann es zu einem Übertrainieren kommen; da das Ibibio-Korpus aber selbst so wenig Daten enthält, halte ich den Wert 10 in diesem Fall für gerechtfertigt, wobei die Voreinstellung von `wagon` bei 50 liegt.

Da im Ablauf des Intonationsmoduls die Dauerprädiktion der Tonschablonen-Prädiktion vorangeht, steht letzterer der Parameter `dur` zusätzlich zur Verfügung. Um Rückkopplungen zu vermeiden, gehen die Parameter `lastcb` und `lastcbc` nicht in die Prädiktion der Tonschablonen mit ein. In der Dauerprädiktion sind die Parameter `cbc` und `lastcbc` erlaubt, werden allerdings nicht vom CART genutzt, wie Tabelle 4.5 zeigt.

6.1.2.1 Dauer-Prädiktion

Der Entscheidungsbaum für die Dauer-Prädiktion nutzt 25 Parameter, von denen die wichtigsten `sylstruc`, `rtone`, `sylphrase`, `firstcons` und `tone` sind. Der Parameter `sylstruc` repräsentiert die Silbenstruktur in der Form einer Zeichenkette mit den Symbolen C für Konsonant, N für Nasal und V für Vokal. Ich nutze für die Ibibio-Version zusätzlich das Symbol N, da im Ibibio sehr häufig Nasale vorkommen und so eine weitere Differenzierung in der Silbenform möglich wird. Allein durch diesen Parameter können 81.22 % aller Dauerwerte aus den Trainingsdaten korrekt prädiziert werden. Die nächsten vier Parameter heben den Wert auf 89.34 % und die restlichen Parameter auf 90.90 %. Der RMSE für die Dauer liegt bei 36.80 ms, der MAE bei 25.44 ms.

Lässt man die Prädiktion mit den Testdaten auf den drei LOO-CARTs laufen, so erhält man RMSEs von 45.14 - 55.50 ms, MAEs von 33.06 - 37.99 ms und Korrelationskoeffizienten r von 0.822 - 0.839 (r^2 0.68 - 0.70).

6.1.2.2 Tonschablonen-Prädiktion

Im Intonationsmodul folgt der Dauerprädiktion die Codevektorprädiktion, die nun den Parameter `dur` zusätzlich nutzen darf. Tatsächlich sind aber `sylphrase`, `wordphrase`, `phonessyl`, `sylstruc` und `ltone4` die wichtigsten fünf der 17 genutzten Prädiktions-

| Wert | 6 (LO 101-110) | 7 (LO 111-120) | 8 (LO 121-130) |
|------------------|-------------------|------------------|------------------|
| Gültige Silben | 219 | 166 | 166 |
| Korrekte CV | 103 | 98 | 64 |
| Korrekte CVC | 118 | 113 | 71 |
| Zuwachs CVC | 15 | 15 | 7 |
| Korrekte CV % | 47.03 | 59.04 | 38.55 |
| Korrekte CVC % | 53.88 | 68.07 | 42.77 |
| %-Zuwachs | 14.56 | 15.30 | 10.94 |
| Rel. Zuwachs (%) | 6.85 | 9.04 | 4.22 |
| # CART Silben | 1695 | 1748 | 1748 |
| T CART korrekt | 1197 | 1256 | 1292 |
| Korrekt (%) | 70.62 | 71.85 | 73.91 |
| # T Features | 16 | 16 | 17 |
| T Feature 1 | sylphrase 0.630 | sylphrase 0.623 | sylphrase 0.656 |
| T Feature 2 | phrasefall 0.647 | wordphrase 0.645 | wordphrase 0.668 |
| T Feature 3 | phonessyl 0.657 | sylstruc 0.664 | sylstruc 0.684 |
| T Feature 4 | wordphrase 0.668 | rtone3 0.674 | ltone4 0.696 |
| T Feature 5 | wordsphrase 0.677 | d 0.683 | rtone4 0.705 |
| D CART RMSE | 37.53 | 37.03 | 37.01 |
| D Test RMSE | 45.14 | 55.50 | 49.32 |
| D CART MAE (ms) | 25.92 | 25.62 | 25.17 |
| D Test MAE (ms) | 33.06 | 37.99 | 35.39 |
| D CART r | 0.908 | 0.908 | 0.909 |
| D Test r | 0.839 | 0.822 | 0.823 |
| D CART r^2 | 0.82 | 0.82 | 0.83 |
| D Test r^2 | 0.70 | 0.68 | 0.68 |
| # D Features | 21 | 19 | 25 |
| D Feature 1 | sylstruc 0.811 | sylstruc 0.807 | sylstruc 0.811 |
| D Feature 2 | rtone 0.862 | rtone 0.857 | rtone 0.862 |
| D Feature 3 | sylphrase 0.880 | sylphrase 0.874 | sylphrase 0.877 |
| D Feature 4 | firstcons 0.888 | firstcons 0.883 | tone 0.885 |
| D Feature 5 | bodir 0.891 | bodir 0.887 | firstcons 0.893 |

Tabelle 6.1: Ergebnisse des CART-Trainings für Dauer- und Tonschablonenprädiktion dreier Trainingsmengen

parameter, wobei der Parameter **sylphrase** (Position der Silbe in der Phrase [1..n]) den größten Beitrag zur Prädiktion leistet (62.96 % korrekt allein mit diesem Parameter), **dur** überhaupt nicht vorkommt und die anderen Parameter die Genauigkeit mit 1376 von 1914 korrekt erkannten Vektoren nur noch auf 71.89 % steigern können. In der Verwechslungsmatrix ist immerhin eine eindeutige Diagonale erkennbar. Die Entropie ist 0.359 und die Perplexität liegt bei 1.283. Der Entscheidungsbaum befindet sich in Anhang C.3.

Bei Anwendung der LOO-Entscheidungsbäume auf die ausgelassenen Testdatenmengen (10 von 94 Sätzen bzw. 9.50 - 12.92 % der Silben), werden nur 38.55 - 59.04 % der Tonschablonen korrekt prädiziert. 42.77 - 68.07 % der Codevektoren liegen in der richtigen Klasse, was einer Verbesserung um 4.22 - 9.04 % entspricht.

6.1.2.3 CART-Diskussion

Aus dem Klassifikationsbaum für die Codevektoren geht hervor, dass die Position der Silbe in der Phrase (**sylphrase**) die größte Rolle in der Auswahl der Tonschablone spielt. Betrachtet man die Auswahl von Codevektoren zu bestimmten Werten von **sylphrase**, so gewinnt man den Eindruck einer globalen Deklinationskomponente bis zur 10. Silbe, und zwar unabhängig von Downsteps oder Downdrift. Die Tabelle 6.2 gibt einen Überblick über einige Prädiktionsergebnisse und die ungefähre Grundfrequenz. Für hohe Werte von **sylphrase** steigt die prädizierte Grundfrequenz wieder. Das könnte

| sylphrase | 1 | 2 | 3 | 4 | 10 | 15 | 20 | 30 | 40 |
|------------------|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| CV | 50 | 53 | 45 | 25 | 8 | 16 | 8 | 15 | 9 |
| CVC | 11 | 12 | 10 | 6 | 2 | 3 | 2 | 5 | 2 |
| $\approx F_0$ | 280 | 290 | 270 | 230 | 180 | 195 | 180 | 210 | 185 |

Tabelle 6.2: Codevektorauswahl des Entscheidungsbaums nach **sylphrase**

darauf hindeuten, dass lange, als Satz segmentierte Einheiten des Korpus eigentlich aus mehreren Phrasen bestehen und ein Zurücksetzen (*reset*) der Grundfrequenz stattfindet.

Leider haben die Parameter **d**, **r** und **f** für die Anzahl der vorangehenden Downsteps sowie steigender und fallender Tonwechsel nicht die erhoffte Erklärungskraft im CART entfaltet. Der Parameter **wordspphrase** könnte auf eine interessante Interaktion zwischen

silbischen Tönen und Wörtern hinweisen. Hier bietet sich eine weitere Nachforschung an.

6.1.3 Maße für die Grundfrequenzdifferenz

Ein großes Problem bei der NUU stellt die Schwierigkeit dar, Ergebnisse zu vergleichen: Die meisten Autoren kennzeichnen ihre Prosodie-Modellgüte durch den *root mean square error* RMSE (Wurzel der quadratischen mittleren Abweichung) zwischen Original- und erzeugter Grundfrequenzkontur. Der RMSE ist sehr ähnlich zur empirischen Standardabweichung definiert als

$$RMSE_{F_0} = \sqrt{\frac{1}{N} \sum_{i=1}^N [F_{0_synthesized}(i) - F_{0_original}(i)]^2} \quad (6.1)$$

mit N : Anzahl der zu vergleichenden Instanzen (F_0 -Werte, *frames*).

Der RMSE stellt aber nur ein rein akustisches Differenzmaß dar und kann weder zeitliche Verschiebungen noch Verzerrungen berücksichtigen und stellt auch kein perzeptives Maß dar. Darüber hinaus lassen sich die Fehlerwerte nicht zwischen mehreren Korpora vergleichen, da deren Grundfrequenzspanne unterschiedlich sein kann und damit zu absolut unterschiedlichen Werten führt.

Ein weiteres typisches Maß ist der Mahalanobis-Abstand (Black und Taylor, 1997a), der den Vorteil hat, zeitliche Verzerrungen zwischen zwei Einheiten linear zu interpolieren, einzelne Parameter über ihre Standardabweichung zu normalisieren und der darüber hinaus von einer Unkorreliertheit der Vektoren untereinander ausgeht (Nukaga et al., 2004). Dieses Maß wird jedoch gewöhnlich in der Kostenberechnung der Einheitenauswahl benutzt und nicht in der Evaluation der Tonkontur. Es ist definiert für $|V| > |U|$ als

$$Adist(V, U) = \frac{WD * |U|}{|V|} * \sum_{i=1}^{|U|} \sum_{j=1}^n \frac{W_j * (abs(F_{ij}(U) - F_{(i*|V|/|U|)j}(V)))}{SD_j * n * |U|} \quad (6.2)$$

mit $|U|$ der Anzahl Messwerte (*frames*) in der Einheit U , $F_{ij}(U)$ der Parameter j des Frames i , SD_j die Standardabweichung des Parameters j und W_j der Gewichtung für den Parameter j . WD ist ein Kostenfaktor für Abweichungen in der Anzahl von Messwerten (d. h. Dauerdifferenzen). Bei einem simplen Grundfrequenzvergleich ist die Anzahl der

Parameter $n = 1$ und $W_1 = 1$, wodurch sich die Formel stark vereinfacht und der der mittleren absoluten Abweichung ähnelt, welche ein selten genutztes Maß ist:

$$e_z = \frac{1}{n} \sum_{i=1}^n |x_i - Z| \quad (6.3)$$

wobei Z den Median bezeichnet.

Rao und Yegnanarayana (2004) nutzen folgende Maße:

”In order to evaluate the prediction accuracy (objective measures) between predicted and actual values, mean absolute error (MAE) and correlation coefficient (γ) are computed using the following formulations.”

Der MAE wird berechnet als

$$MAE(x, y) = \frac{1}{N} \sum_{i=1}^N |(x_i - y_i)| \quad (6.4)$$

und der (Maß-) Korrelationskoeffizient r (Sun, 2002b), welcher der Korrelation γ entspricht, als:

$$r = \frac{SAQ_{xy}}{\sqrt{SAQ_x * SAQ_y}} \quad (6.5)$$

$$\gamma = \frac{\sum_i |(x_i - \bar{x})| * |(y_i - \bar{y})|}{\sqrt{\sum_i (x_i - \bar{x})^2} \sqrt{\sum_i (y_i - \bar{y})^2}} \quad (6.6)$$

Oft wird auch der *mean square error* MSE (Agüero et al., 2004) genutzt, da man sich in zeitkritischen Systemen bemüht, Formeln ohne Wurzeln zu nutzen und die Maße im Quadratischen zu lassen, um Rechenzeit zu sparen.

6.1.4 Stimulierzeugung

Die zehn aus dem Synthesekorpus ausgelassenen Testsätze wurden *offline* (d. h. nicht über eine Netzverbindung sondern direkt aus einer lokalen XML-Datei) auf dem BOSS-Server synthetisiert. Dabei traten mehrere Probleme auf:

- Bei einem Satz brach der Server vorzeitig ab. Es zeigte sich, dass in den mySQL-Tabellen, welche die Zuordnung von Phonem zu Silben, Silben zu Wörtern und

Wörtern zu Sätzen repräsentieren, eine Einheit fehlte und die Sprachdatei für die Synthese nicht gefunden wurde. Dieser Fehler könnte an einer nicht-hierarchischen Segmentierung liegen, aber die Frage ist noch offen.

- Die auf Silbenebene geplante und in einigen Fällen auf Wortebene mögliche Einheiten Auswahl fand fast keine einzige passende silbische Einheit im Korpus, selbst wenn der Suchfokus maximal aufgeweitet wurde. Dementsprechend fand fast ausschließlich eine Synthese auf Phonebene statt. Über die Attribute `phonepos` und `CB` wäre möglicherweise eine noch hinnehmbare Synthesequalität machbar gewesen, aber aufgrund der zu geringen Datenmenge im Korpus wurde selbst auf Phonebene der Suchfokus maximal aufgeweitet, wodurch Phone aus falschen segmentalen und tonalen Kontexten ausgewählt wurden. Hierbei tritt das Problem auf, dass Vokale oft assimiliert werden und mit dem folgenden Vokal verschmelzen. Werden diese in einem anderen Kontext ausgewählt, so stimmt das Label nicht mehr mit der Realisierung überein.
- In einigen Fällen wurden *überhaupt keine* passenden Einheiten gefunden. Die Notlösung für solche Fälle ist die Synthese per Neutralvokal /ə/. In vokalischem Kontext mag das in Notfällen funktionieren, in konsonantischem eher nicht. Wenn diese Ersetzung zu häufig vorkommt, ist die Synthese völlig unbrauchbar.

Ich entscheide mich aufgrund der momentan allzu unbefriedigenden Ergebnisse, keine quantitative Analyse durchzuführen. Es wird ein größeres Korpus benötigt.

6.2 Qualitative Evaluation

Allgemeine Synthesequalität

Eine qualitative Untersuchung aufgrund folgender Beurteilungskriterien war geplant: Fünf Bewertungskriterien in jeweils sieben Abstufungen:

1. Wohlklang (*pleasantness: very pleasant .. very unpleasant*)
2. Natürlichkeit (*naturalness: very natural .. very unnatural*)
3. Segmentale Verständlichkeit (*intelligibility: very intelligible .. not at all intelligible*)

4. Gesamteindruck (*intonation: has very good intonation .. has very bad intonation*)
5. Einverständnis zum Vorlesen eines Zeitungstextes mit Hilfe dieser Synthese (*would really like to listen to a newspaper read by the system .. wouldn't like to listen at all to a newspaper read by the system*)

Die zehn Sätze der Testdatenmenge 8 (Satz 121-130) wurden mit BOSS resynthetisiert und sollten jedem Hörer zwei Mal vorgespielt werden (für die Intra-Hörer-Reliabilität). Das ergibt bei 10 Sätze x 2 Wiederholungen x 5 Kriterien x ca. 10 Hörer z.B. 1000 Datenwerte.

Zum jetzigen Stand der Entwicklung mit dem kleinen Korpus macht eine subjektive Untersuchung der Synthesequalität noch keinen Sinn.

7 Schluss

Ziel dieser Arbeit war es, ein lauffähiges Intonationsmodul für BOSS zu entwickeln. Grundlage dafür war ein minimal segmentiertes und annotiertes Korpus. Nach der Festlegung der Modellart (parametrisch, phonetisch, lückenlos, sequentiell, polynominell) und der Prädiktionmethode (Klassifikationsbaum mit CART) sowie der Auswahl angemessener Prädiktionsparameter wurde das Modul in C++ implementiert. Ich möchte hier noch einmal die wichtigsten Ergebnisse zusammenfassen.

Die Grundfrequenzstilisierung (Abschnitt 4.2.1) besitzt einen RMSE von 4.88 Hz für die von mir gewählten biquadratischen Polynome, was als sehr gut einzuschätzen ist. Das Silbenpolynom ist jedoch einzig für die jeweilige Silbe optimiert und berücksichtigt nicht den intonatorischen Kontext an den Silbenrändern. Dazu kommt, dass die Silbe für den Fall, dass weniger als ein Drittel stimmhaft ist, gar nicht berücksichtigt wird. Das führt dazu, dass mögliche wichtige Silbenkonturen gar nicht eingefangen werden können.

In der Vektorquantisierung (Abschnitt 4.2.2) kann das dazu führen, wichtige prototypische Silbenkonturen auszulassen. Die SNRs von -23.50 bzw. -24.63 sind zwar keine sehr guten Werte aber für die aktuelle Situation durchaus zufriedenstellend. Ziel des Codebooks war u. a. auch, Prototypen der Silbenkonturen des Ibibio zu identifizieren, was subjektiv (visuell) geglückt ist. Eine Codebookgröße $|C_1|$ von 64 scheint für das Korpus im momentanen Entwicklungsstand angemessen zu sein. In einem großen bis sehr großen Korpus wird eine Vergrößerung des Codebooks sinnvoll und produktiv sein. Die Verdoppelung der Codebookgröße $|C_2|$ von $\sqrt{|C_1|} = 8$ auf 16 brachte eine deutliche Verbesserung der SNR und konnte mehr distinktive Grundformen der Silbenintonation des Ibibio einfangen.

Die Prädiktionsmerkmale aus Abschnitt 4.3 zeigen in der Dauerprädiktion per CART ein sehr gutes Ergebnis von 90.90 % korrekt prädizierter Werte bei einem RMSE von 45.14 - 55.50 ms auf den Testdatenmengen. Die Tonschablonenprädiktion mit einer Genauigkeit von 72.26 %¹ hat auf den ersten Blick ein enttäuschendes Ergebnis von nur 38.55 - 59.04 % korrekt prädizierter Codevektoren auf den Testdatenmengen, liegt aber in der Klassifikation der Codevektorklassen bei immerhin 10.94 - 15.30 % höher. Hier kann man überlegen, ob noch weitere Prädiktionsmerkmale hinzuzufügen sind, oder ob es an der geringen Datenmenge liegt.

Das Korpus , das allen Untersuchungen zu Grunde lag, muss für eine konkatenative Sprachsynthese als zu klein angesehen werden. Mit 5.5 Minuten Sprache kommt es nicht annähernd an das in Portele (1998, nach Hess 2004b, F. 13) beschriebene Minimum von 20 Minuten Sprachdaten für die NUU heran:

”Portele (1998) [...] wies die grundsätzliche Machbarkeit einer derartigen Synthese auch schon mit einer Datenbasis von 20 Minuten Dauer nach und zeigte, dass der Arbeitsaufwand für die Initialisierung eines solchen Systems, ist die Datenbank einmal vorhanden, vergleichsweise gering ist.”

In die weitere Forschung und Entwicklung sollten folgende Punkte eingehen: Eine weitere Untersuchung der Prädiktionsparameter: Welche Rolle spielen Downstep und Downtrend wirklich? Ein Downtrend kann beobachtet werden, aber aus meinem CART geht vorerst eher eine phonetische Rolle hervor (Deklination) statt der postulierten möglichen phonologischen (Connell, 2001). Das Modul für die Signalmanipulation sollte noch in die Ibibio-Synthese integriert werden. Dazu muss die Grundfrequenzkontur des gesamten Zielsatzes berechnet und geglättet werden und an das Manipulations- und Konkatenationsmodul übergeben werden.

Besonders wichtig ist es aber, vor allem ein größeres Korpus zu labeln, mit dem neue Rahmenwerte für die Codebookgrößen bestimmt werden und weitere Erfahrungen mit der Prädiktion gesammelt werden können.

¹Dieser Wert gilt für ein Training auf dem gesamten Korpus; letzter Stand 2006-08-31.

A BOSS-Konfigurationsdatei

Die folgende Liste nennt alle neuen und von der deutschen Version abweichenden Optionen der zentralen BOSS-Konfigurationsdatei **boss3conf.xml**; Werte in Klammern geben den aktuell benutzten Standardwert an:

usewords (1) Schaltet die Synthese auf Wortebene ein. Während der Entwicklung wurde die Wortebene zuerst ignoriert.

order (4) Polynomgrad

cbsize (64) Größe des Codebooks

tonecart (data/ibb/eno/runtime/boss_sampa/ibb_eno_tone.cart) Pfad zum Tonschablonen-Klassifikationsbaum.

durcart (data/ibb/eno/runtime/boss_sampa/ibb_eno_dur.cart) Pfad zum Dauer-Regressionsbaum.

codebook (data/ibb/eno/runtime/boss_sampa/ibb_eno.cb) Pfad des Codebooks

codebook2 (data/ibb/eno/runtime/boss_sampa/ibb_eno.class) Pfad für die Codevektor-Klassifikationsdatei

pshift (10.0) Abstand der F_0 -frames, hier 10 ms

tcw_melc (1.0) Verkettungs-Gewichtungsfaktor für MFCC-Differenzen

tcw_f0 (1.0) Verkettungs-Gewichtungsfaktor für F_0 -Differenzen

ucw_dur (1.0) Gewichtungsfaktor für Dauerdifferenzkosten

ucw_phrase (0.5) Gewichtungsfaktor für Abweichungen der Phrasengrenzeninformation auf Wort- und Silbenebene

ucw_tone (1.0) Gewichtungsfaktor für Abweichungen des Tones auf Silben- und Phonenebene

ucw_bv (1.0) Gewichtungsfaktor für die Stützstellenkosten auf der Silbenebene

ucw_f0 (1.0) Gewichtungsfaktor für die F_0 -Randwertkosten auf Wortebene

- ucw_phpos** (1.0) Gewichtungsfaktor für Abweichungen der Phonposition in der Silbe
- tcw_melc_norm** (26.0) Verkettungs-Normalisierungsfaktor für MFCC-Differenzen
- tcw_f0_norm** (238.745) Verkettungs-Normalisierungsfaktor für F₀-Differenzen
- ucw_dur_norm** (60.0) Normalisierungsfaktor für Dauerdifferenzkosten
- ucw_phrase_norm** (1.0) Normalisierungsfaktor für Abweichungen der Phrasengrenzeninformation auf Wort- und Silbenebene
- ucw_tone_norm** (1.0) Normalisierungsfaktor für Abweichungen des Tones auf Silben- und Phonebene
- ucw_bv_norm** (50.0) Normalisierungsfaktor für die Stützstellenkosten auf der Silbenebene
- ucw_f0_norm** (238.745) Normalisierungsfaktor für die F₀-Randwertkosten auf Wortebene
- ucw_phpos_norm** (1.0) Normalisierungsfaktor für Abweichungen der Phonposition in der Silbe
- espsdir** (data/ibb/eno/inventory/esps) Pfad zu den ESPS-Dateien, die die 4 benötigten Spalten der get_f0 - Informationen enthalten
- f0dir** (data/ibb/eno/inventory/f0) Pfad zu den f0-Dateien, die satzweise extrahierte F₀-Werte enthalten
- f0sdir** (data/ibb/eno/inventory/f0s) Pfad zu den f0s-Dateien, die silbenweise extrahierte F₀-Werte enthalten
- cbdir** (data/ibb/eno/inventory/cb) Pfad zu den cb-Dateien, die die silbenweise am besten passenden Codevektoren enthalten
- polydir** (data/ibb/eno/inventory/poly) Pfad zu den poly-Dateien, die silbenweise Stützstellen und Polynom-Koeffizienten enthalten
- tgdir** (data/ibb/eno/inventory/textgrid) Pfad zu den Praat-TextGrid-Dateien, die Segmentierung und Annotationen enthalten
- xmldir** (data/ibb/eno/inventory/xml) Pfad zu den xml-Dateien, die aus den TextGrids erstellt wurden
- rawdirt** (data/ibb/eno/inventory/raw) Pfad zu den raw-Dateien, die die Sprachstücke satzweise in 16 kHz, 16 bit, mono enthalten

- pulsedir** (data/ibb/eno/inventory/pulse) Pfad zu den PointTier-Dateien, die die Zeitpunkte der Glottisverschlüsse enthalten
- espsext** (.f0_ascii) Erweiterung der ESPS-Dateien, die die 4 benötigten Spalten der get_f0 - Informationen enthalten.
- f0ext** (.f0) Erweiterung der f0-Dateien, die satzweise extrahierte F_0 -Werte enthalten
- f0sext** (.f0s) Erweiterung der f0s-Dateien, die silbenweise extrahierte F_0 -Werte enthalten
- cbext** (.cb) Erweiterung der cb-Dateien, die die silbenweise am besten passenden Co-devektoren enthalten
- polyext** (.poly) Erweiterung der poly-Dateien, die silbenweise Stützstellen und Polynom-Koeffizienten enthalten
- tgext** (.TextGrid) Erweiterung der Praat-TextGrid-Dateien, die Segmentierung und Annotationen enthalten
- xmlext** (.xml) Erweiterung der xml-Dateien, die aus den TextGrids erstellt wurden
- rawext** (.raw) Erweiterung der raw-Dateien, die die Sprachstücke satzweise in 16 kHz, 16 bit, mono enthalten
- pulseext** (.PointProcess) Erweiterung der PointProcess-Dateien, die die Zeitpunkte der Glottisverschlüsse enthalten

B Das Korpus

B.1 Inhalt

Das Korpus besteht insgesamt aus 165 Sätzen. Von diesen waren 94 segmentiert und annotiert. Im Folgenden sind alle Sätze aufgelistet, die in die Synthese integriert wurden (21-70, 91-120 und 131-134).

B.1.1 Sätze des Korpus

- 21: /ke abaŋa uduak edijak utom umiŋŋkpɔ ake ukara nnɔ mbonowo/
22: /njobio owo nseabasi ette/
23: /ke emum umiaŋ nditɔ akwa ibom edueehe mbet ufɔk utom emi/
24: /ke ekpesuk odo eeŋimme ebo ke ibaan esɔŋ itak afara unamŋkpɔ/
25: /ke idaha esukko enam mbeeŋeidem ebaŋa edikoono akpa iduɔk nditɔ ufɔkŋwet ukpeepŋkpɔ mbaŋa ibɔk mbakara mme nsɔŋidem ndɔŋ/
26: /dɔkta desi wiidism/
27: /mbɔk idakka do innɔ ke edinim utom emi nɔ akpeene ntom/
28: /nniehe idoteŋen ke idem owo emi atɔɔŋke utom ɔmɔ je ŋkara ke esit/
29: /ekamba awetŋwet ke ufɔkutom enamma ŋkpɔ ebaŋa isɔŋ mme ubɔɔp ufɔk aketaŋ ami/
30: /eŋe akedo aŋwaan ete jɔɔn esien abaaha ke ufɔk utom usuan mbuk ake ekebe ndise ujo/
31: /editwak nda ke mben usuy/
32: /amaakwɔɔɔ obo nɔ ikɔt abasi ejaak uwem ɔmmɔ esm abasi ke ubɔk mbaak ebo edidiɔŋ abasi/
33: /owo ibarake ikaŋ ikeene daŋa uduɔk ikɔt ɔɔɔŋ/
34: /eebɔppɔ ke idit keetekeet anaasime ŋkpɔ siben ufɔn isɔk ibaan/
35: /ekamba ase ŋkpɔ mbaŋa kooberatif defedopmen baŋk/
36: /mbaak ɔmmɔ ekeme ediwuɔ nda ke usuy ɔfɔnnɔ/
37: /ikiduɔɔŋ ke iŋaŋ/
38: /nɔ eŋe je ikɔt utom ɔmɔ eewɔt ŋɔm ŋkem ɔmmɔ ɔjɔhɔɔjɔhɔ/
39: /mbon unam nduuyɔ eben ewɔt ebo ke otu bidion owo ke afit unadot eedoŋŋoke ɔkpɔikpɔi uweene/
40: /amaanam ediwɔŋ ke esai mme itie adiduuyɔ uduɔŋ itiaita ke ijiip mme owo mi ke sted ŋɔm/

- 41: /je nditɔ ete ebe ɔmmɔ ke ini mme ebe ɔmmɔ ekpaaha ekpɔɔ ɔmmɔ/
42: /eje ikisitreke editiik idem ke ini ke ini/
43: /ke imɔ itie nte asasawrek/
44: /ɔbɔɔɔ fiktɔ attah amaakɔɔm mbon ufɔkutom emi/
45: /ete joo akenofa amaabo nɔ mbon se ibɔ mfɔn ami etim ekama ɔkuk ami/
46: /mfɔn jems ejɔ ake ufɔkutom usuan mbuk ake ekebe ndise ujo/
47: /ɔmmɔ emaedɔɔ idem ɔmmɔ okpoho ke nnuun nte idioɔɔ edidiana keet/
48: /nsinia okon amaadian obo ke ufɔn ntunɔ odo ke adikpura owo nɔ akappa ɔfɔn/
49: /emaesim ebeenje ebo nɔ ukara enɔ mmoto bɔɔs/
50: /anam nɔ ntɔkejm esim idem ɔmmɔ ke edinam mme ido ammiikpinaaha nte ɔmmɔ enam/
51: /nɔm iminim ite ke epe ɔbɔ nduɔk odudu/
52: /ɔɔɔɔ ɔkɔɔ mme aɔam urua odo ke edimaana ntɔɔɔ utom adibɔɔ mme ufɔk urua odo ke idem ɔmmɔ/
53: /enam nduɔɔ ese me mme edinam se eduak ediifɔn die je ibaan/
54: /mbɔkɔɔ inɔɔ odo mbɔkɔɔ edat/
55: /kristian kuujeeme ke ini aɔwan/
56: /odooho kɔmred emma esooke/
57: /etɔk ɔsuhɔ krais amaajem itimme ɔkpɔɔ utom epe akedihe/
58: /ekenimme ke umuahia/
59: /amaakop iduɔesit abaɔa unɔmɔ ake bɔmb akesiine ke deɔs/
60: /daɔa ekenɔɔɔ mme ɔkpooɔ owo ke ufɔk utom usuan mbuk ake ekebe ndise ɔkuk udeeme ɔkpodɔro ufɔk/
61: /ufɔk abasi odo ekesooboke ke isua itiaba akeboijoke do/
62: /ɔɔɔɔ akɔt obo ke eesɔɔɔ mme mbet akɔtte obo ke owoɔwaan ɔɔ ndɔ oto ke upim-me ɔmɔ/
63: /aakpenamma nɔ mme owo esifiik unen ibaan enieehe ke ntak edido owoɔwaan/
64: /saɔa saɔa ɔnɔ item ke sains mme teknoroji/
65: /muusɔppɔɔ idem udakka ke mmi/
66: /enɔ mbio efiommoke ibaan ukpeep ɔkpɔ ke editre mme uto eduuwem ntom/
67: /ibaaha owo se ikpikikan inɔk epe isim ke itie utaɔ ikɔ/
68: /eɔɔɔ enam nduɔɔ ekɔt ke enam mme ukpuuɔɔ ɔkpɔ/
69: /ke andidiito ikan inam nɔ akaisaɔ asɔɔɔ ɔfɔn ufuhɔ/
70: /ekɔt ebo ke ekpeme ibaan je uboikpa eedɔɔɔoke ke mme itie ekɔɔ aɔwanake/
71: /edioɔɔke ibio ibio nte akruwasan aasiak mme itie edibɔ mmɔɔ esajukwak akanna ikie itiokeet/
72: /ufɔk fedraad sekriteriad ke ujo amaabuuro eti eti ke idaha mbon napep ke ikpehe ujo/
73: /abasi mbɔk ɔaɔa ubon owo/
74: /mma teedma ɔwokedi/
75: /ejauwɔt afɔt se akenam ke ndibe aɔwa aɔwa/

- 96: /ideppedep/
97: /ke ini ekebirpe ɔmmɔ mme mbuumo ke abaja itie odo/
98: /ke mme mfina ake mme ibaan ufɔkndɔ ajaamaana afeere/
99: /amaadido se ekokonno eda ekɪt koo/
100: /abaaha ke ekamba ufɔkɪwɛt ntaifɔk nnamdi asikiwe ke ɔka/
101: /ndion ke mbre adiɪwam ke uto nneeɛidem ami/
102: /ke ntak inem mmm je ukɔt emi ɔmmɔ ekedatta nte akpan ufan ɔmmɔ/
103: /kpa ntro ŋko ke akaisaŋ anenekke atuut ɔfɔŋ ufɔhɔ ɔfuk idem ɔmɔ/
104: /ebo jak ukara enam mbet ebaja mme eduuwem ake miijakka ibaan ekop inemu-
wem je nsɔɪidem/
105: /tɔɔŋɔ ke ŋkanika itiaita ubahausen mbaak edioɔŋɔ mme mme anamutom kɔfmen
enam ujo eekenɔɔhɔ/
106: /mmuudiaha unɔ usan/
107: /mme itie akeododo ke sted/
108: /mma duisa ubɔɔŋ amaadian do obo ke ŋketi esɛŋwa odo ŋkpɔ uto ake ɔsɔbɔke
eti eti/
109: /okop ujukkɔ abaja utom akaahaiso ke usɔŋ itu mbon usɔ mme arosukwu/
110: /fɔn ido jem ŋkpɔ mfɛn dia/
111: /unek unek je uŋwɔŋ ŋkpɔ ke mme ufɔk udia uwem emi ekedidakka kwa efaak
kwa efaak ke iduŋ/
112: /ɛpɛ amaasua ɔnɔ ubioŋ utire utom/
113: /akedooho ŋkanika abee itiaɔa/
114: /edet itohoakwa/
115: /ekebaaha ke ndopnda ke ŋkpɔ nte isua iba mme akanna odo/
116: /ndion ekpak ufɔkmbet ukara sted ebo nɔ enam mbet/
117: /ɔɪuŋ ataj obo nɔ afɪt mbon ŋka odo edian ubɔk emenne utom sednet/
118: /kuufere ke idomo mme ke ukuut/
119: /nsikak eduɔk amaataŋ ke adipeepɪk mbon abasi mbo nɔ esm ubɔk ke ekpat/
120: /iibohoke ke iwuot afɪt ibaan/
121: /itie unam spɔɔt/
122: /ɛpɛ amaano ufɔk uman ake ufɔk abasi odo beed nnaa anieehe sibriŋ duoɛɛba/
123: /ɛpɛ amaabo ke ufɔkɪwɛt odo osiooŋɔ mme oku abasi ke mme nsio nsio ufɔk abasi
mme ikpehe/
124: /amaataŋ obo ke udɔŋ adikpura aɪuŋ ufɔk abasi odo ikitooho ndausuŋ ake sibiɪt
abasi/

B.1.2 Testsätze

- 121: /ana ekop nsɔɪidem odo ɔɔhɔ ɔɔhɔ ke ifuuro uwem/
122: /ekæriku odo akwa owo eto/

123: /mmɔɔŋ idim ukana amaasidenŋe/

124: /usioudat usioudat/

125: /ke ukara ebiooro adism uɲwam nnɔɔ ufɔkɲwet ukpeep mme ɔkwɔɔikɔ ke mme utom ŋkɔt adidippe ufɔkɲwet odo/

126: /epe ikifreke ediwakka ufɔk abasi ŋkebɔŋ akam ŋɲɲ ŋkɔɔm abasi ke uɲwam je ndausuŋ ɔmɔ/

127: /ete edet nduɔpoi/

128: /aɲwaan mme ebe emi emaewɔt ima ke adikpikke kek ndɔ nnɔ keet eken/

129: /enam nɔ ibaan enie nɲmɲɔidem/

130: /rattawu/

B.2 Synthesebeispiele

Das LOO-Synthesekorpus Nr. 8 besteht aus 84 Sätzen, 728 Wörtern, 1750 Silben und 3446 Phonemen sowie 244 Pausen (auf allen Ebenen). Es hat eine Gesamtlänge von 9 Minuten und 46 Sekunden, wovon 4 Minuten und 32 Sekunden auf Pausen entfallen. Die Sätze 121-130 stellen die Testsätze für das Korpus Nr. 8.

Die Testsätze wurden mit BOSS synthetisiert und befinden sich auf der dieser Arbeit beigelegten CD-ROM. Zur Synthese wurden die auf dem gesamten Korpus trainierten CARTs und Codebooks genutzt, jedoch die Attribute CB und CBC in der Vorauswahl ausgeschaltet, da die Codevektoren der einzelnen Silben momentan bei jeder (LOO-)Korpuserstellung neu erzeugt werden und die XML-Attribute CB und CBC des LOO-Korpus daher nicht mit denen des vollständigen Korpus übereinstimmen.

Zusätzlich wurde die prädisierte Intonation der Testsätze mit Hilfe der MBROLA-Diphonsynthese (Dutoit et al., 1996) versuchsweise reproduziert. Die Ergebnisse sind ebenso der CD beigelegt. Hierbei wurden allerdings nur die Stützstellen der Polynome als Zielpunkte übergeben und die Interpolation wurde MBROLA überlassen.

C Codebooks

C.1 Das Tonschablonen-Codebook

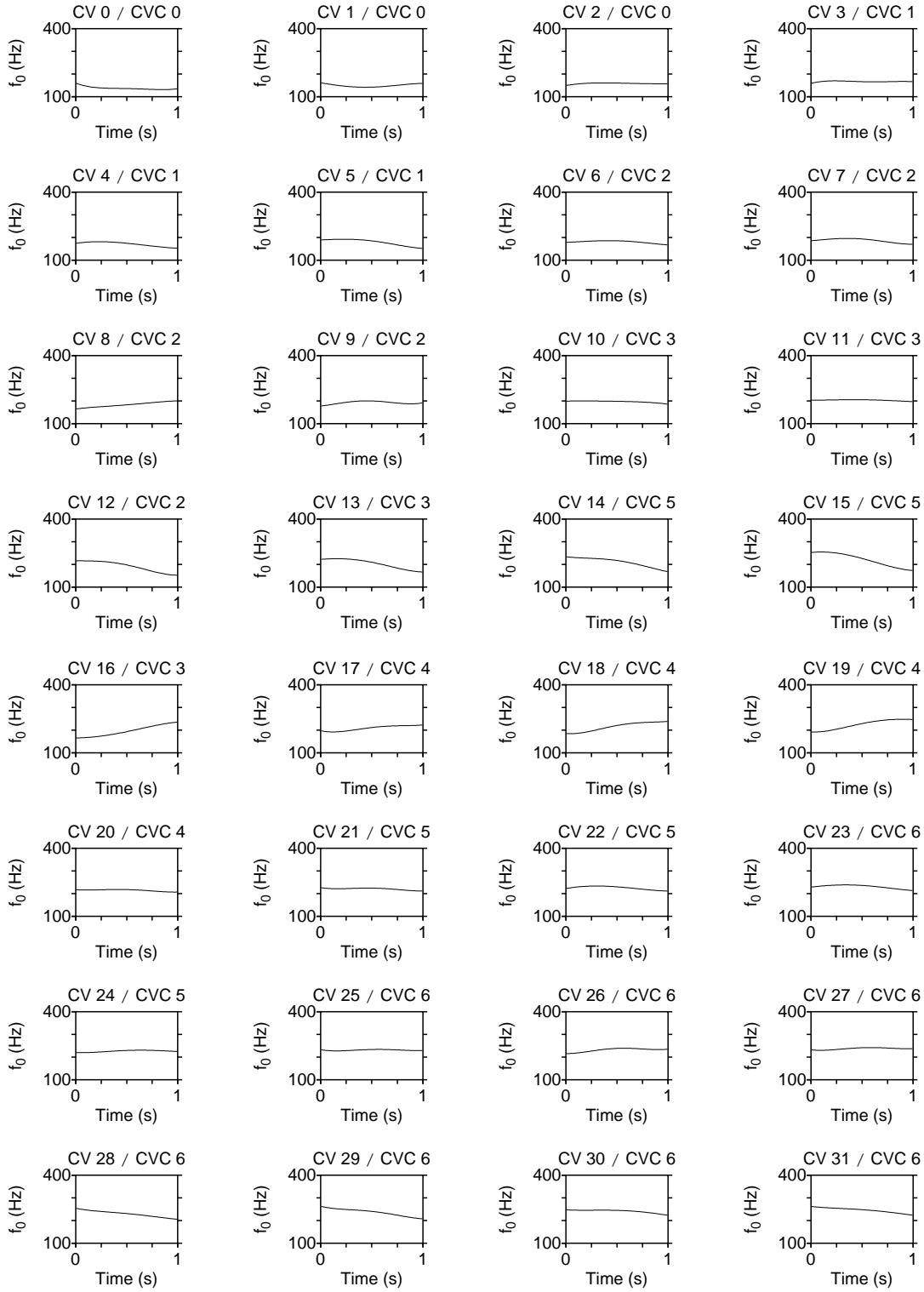
C.1.1 Codevektoren

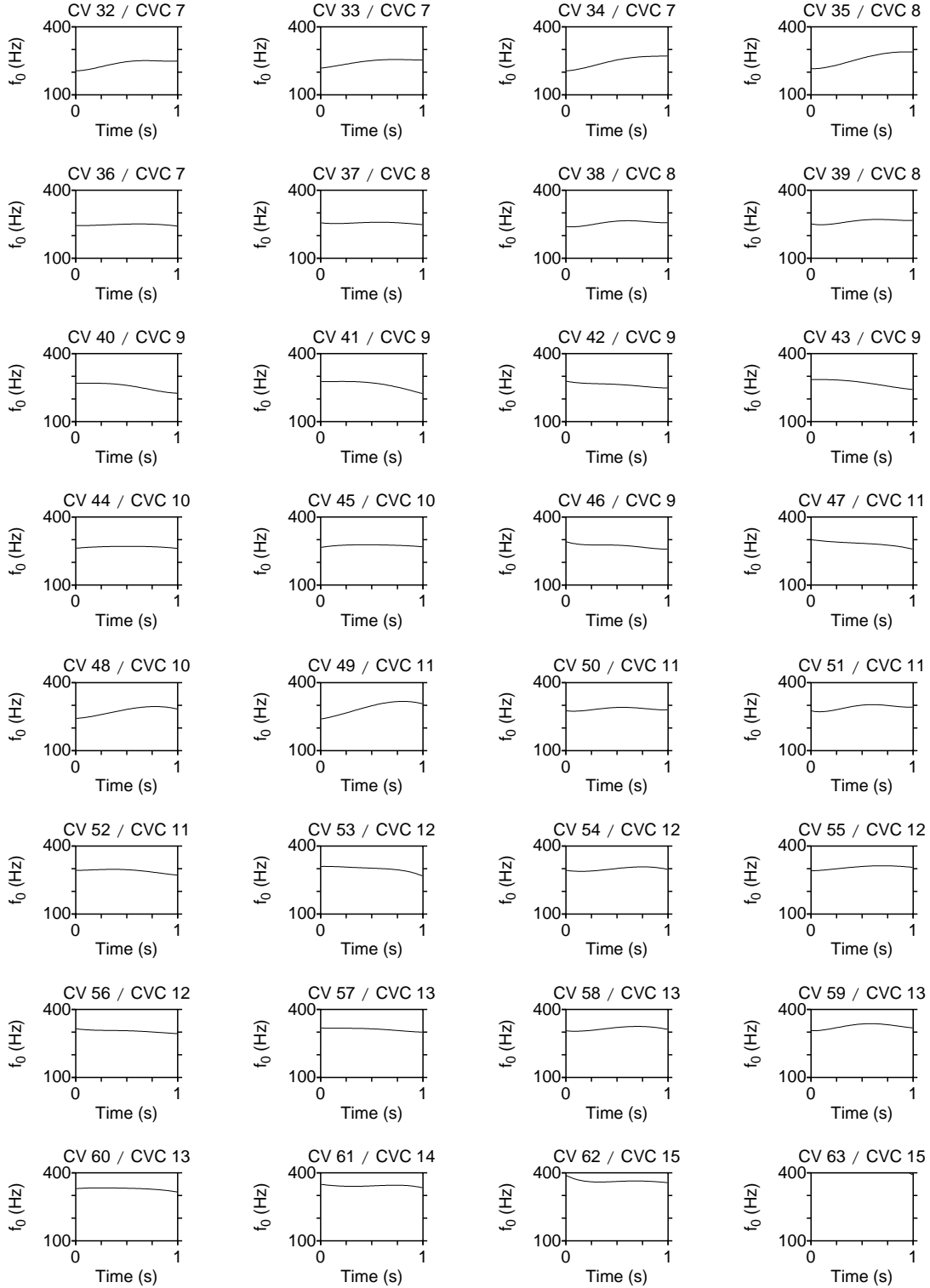
| CV | 0 | 1 | 2 | 3 | 4 |
|----|--------|--------|--------|--------|--------|
| 0 | 159.34 | 138.69 | 136.07 | 132.43 | 135.03 |
| 1 | 161.06 | 145.71 | 142.31 | 150.46 | 158.84 |
| 2 | 149.67 | 160.04 | 160.25 | 157.97 | 156.86 |
| 3 | 158.95 | 169.90 | 166.39 | 166.27 | 166.31 |
| 4 | 174.54 | 181.42 | 174.69 | 162.55 | 152.55 |
| 5 | 189.42 | 192.30 | 185.16 | 167.68 | 151.98 |
| 6 | 178.66 | 184.24 | 185.57 | 178.37 | 167.48 |
| 7 | 185.97 | 194.69 | 193.57 | 180.47 | 170.41 |
| 8 | 165.00 | 174.91 | 182.82 | 192.62 | 200.49 |
| 9 | 177.72 | 194.30 | 200.04 | 190.91 | 191.60 |
| 10 | 199.22 | 199.79 | 198.41 | 195.20 | 186.17 |
| 11 | 203.42 | 204.83 | 205.35 | 202.29 | 197.13 |
| 12 | 216.31 | 213.12 | 197.79 | 170.24 | 152.63 |
| 13 | 222.60 | 223.95 | 209.48 | 184.23 | 166.87 |
| 14 | 233.82 | 226.68 | 217.36 | 195.32 | 168.41 |
| 15 | 252.75 | 250.02 | 226.39 | 194.65 | 173.18 |
| 16 | 165.49 | 174.13 | 193.27 | 217.52 | 235.30 |
| 17 | 197.01 | 196.40 | 211.63 | 219.28 | 222.83 |
| 18 | 185.31 | 196.44 | 220.46 | 233.41 | 238.43 |
| 19 | 192.06 | 204.19 | 228.79 | 245.21 | 247.08 |
| 20 | 217.38 | 217.09 | 217.89 | 211.81 | 206.86 |
| 21 | 226.21 | 222.10 | 224.32 | 218.15 | 211.82 |
| 22 | 222.51 | 233.27 | 229.91 | 219.49 | 211.24 |
| 23 | 228.97 | 237.67 | 236.18 | 225.63 | 213.24 |
| 24 | 219.81 | 222.91 | 229.15 | 229.61 | 224.74 |
| 25 | 232.11 | 228.37 | 233.49 | 231.74 | 228.91 |
| 26 | 215.48 | 226.68 | 238.26 | 235.85 | 234.93 |
| 27 | 231.68 | 233.40 | 241.14 | 239.85 | 237.20 |
| 28 | 253.47 | 239.88 | 230.72 | 218.58 | 205.80 |

| CV | 0 | 1 | 2 | 3 | 4 |
|-----------|----------|----------|----------|----------|----------|
| 29 | 262.51 | 248.61 | 240.40 | 223.86 | 207.87 |
| 30 | 247.18 | 245.40 | 244.86 | 237.90 | 223.42 |
| 31 | 261.77 | 254.17 | 247.95 | 237.77 | 223.75 |
| 32 | 206.14 | 224.43 | 246.11 | 250.68 | 249.21 |
| 33 | 218.25 | 235.69 | 251.27 | 255.80 | 253.58 |
| 34 | 206.12 | 226.76 | 253.68 | 268.08 | 271.62 |
| 35 | 215.46 | 231.19 | 261.57 | 284.00 | 289.24 |
| 36 | 243.84 | 246.33 | 250.44 | 250.24 | 241.73 |
| 37 | 256.02 | 253.99 | 258.39 | 256.64 | 248.35 |
| 38 | 239.82 | 248.06 | 263.57 | 262.83 | 256.83 |
| 39 | 250.99 | 253.48 | 268.03 | 270.19 | 266.60 |
| 40 | 269.28 | 268.78 | 260.02 | 240.35 | 225.43 |
| 41 | 277.01 | 276.91 | 271.08 | 252.38 | 223.63 |
| 42 | 278.08 | 267.88 | 264.07 | 255.61 | 248.43 |
| 43 | 285.46 | 283.57 | 273.44 | 256.84 | 241.76 |
| 44 | 262.24 | 268.71 | 270.54 | 268.94 | 261.60 |
| 45 | 266.20 | 276.10 | 277.34 | 274.93 | 268.53 |
| 46 | 290.55 | 276.79 | 275.56 | 266.71 | 258.19 |
| 47 | 299.51 | 289.36 | 283.70 | 276.01 | 258.41 |
| 48 | 242.04 | 259.17 | 281.79 | 294.29 | 283.64 |
| 49 | 240.17 | 266.79 | 297.95 | 316.43 | 306.61 |
| 50 | 275.82 | 280.29 | 290.49 | 286.62 | 280.16 |
| 51 | 275.34 | 281.09 | 300.59 | 299.41 | 292.44 |
| 52 | 292.51 | 296.29 | 295.63 | 285.26 | 272.02 |
| 53 | 310.17 | 307.03 | 302.22 | 295.17 | 267.98 |
| 54 | 292.52 | 289.94 | 300.84 | 308.13 | 296.94 |
| 55 | 290.61 | 299.53 | 309.77 | 312.76 | 305.99 |
| 56 | 314.45 | 307.66 | 305.89 | 300.61 | 293.29 |
| 57 | 317.77 | 316.70 | 314.37 | 307.02 | 299.84 |
| 58 | 305.40 | 307.86 | 320.16 | 324.57 | 312.18 |
| 59 | 305.83 | 319.36 | 335.64 | 332.62 | 318.68 |
| 60 | 330.36 | 333.38 | 331.94 | 327.61 | 315.12 |
| 61 | 348.90 | 340.49 | 342.16 | 345.23 | 333.83 |
| 62 | 387.83 | 360.14 | 361.69 | 363.55 | 356.40 |
| 63 | 414.98 | 419.81 | 417.68 | 422.94 | 390.49 |

Tabelle C.1: Codebook der Intonationschablonen

C.1.2 Silbenkonturen





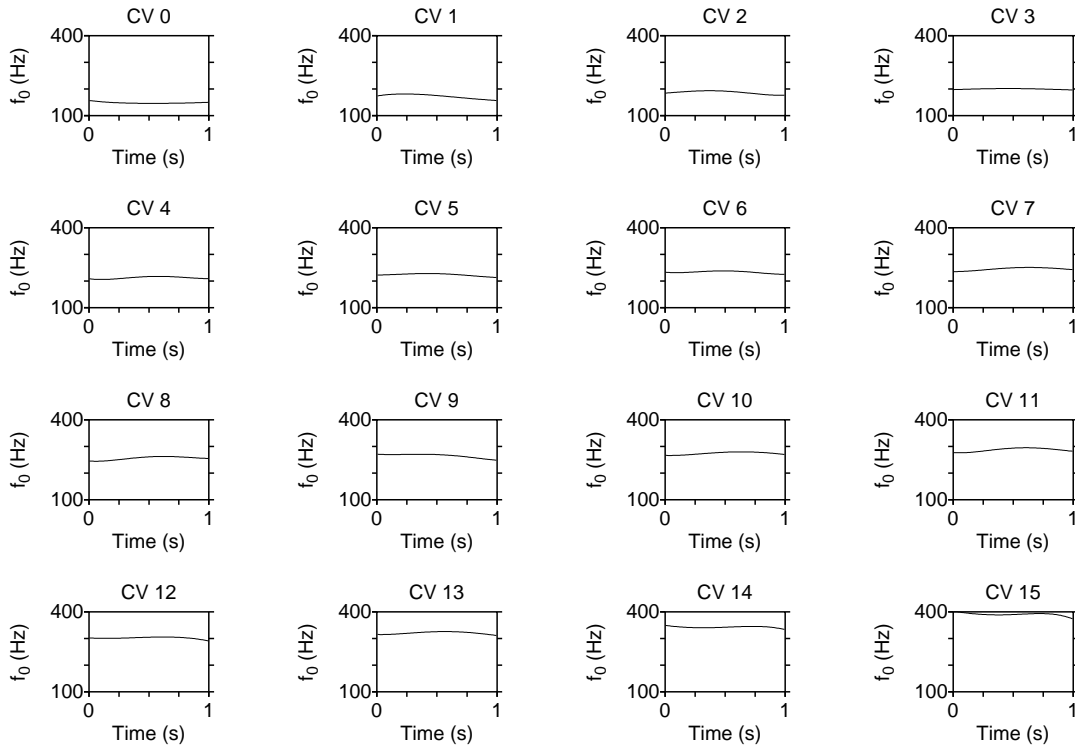
C.2 Das Tonschablonen-Klassen-Codebook

C.2.1 Codevektoren

| Nr | 0 | 1 | 2 | 3 | 4 |
|----|--------|--------|--------|--------|--------|
| 0 | 156.69 | 148.15 | 146.21 | 146.95 | 150.24 |
| 1 | 174.3 | 181.21 | 175.41 | 165.5 | 156.94 |
| 2 | 184.73 | 192.25 | 191.96 | 182.52 | 176.52 |
| 3 | 197.68 | 200.67 | 201.63 | 199.81 | 196.36 |
| 4 | 208.38 | 209.15 | 216.84 | 214.96 | 209.13 |
| 5 | 222.67 | 226.5 | 227.71 | 221.42 | 213.61 |
| 6 | 232.91 | 234.15 | 238.04 | 232.31 | 225.31 |
| 7 | 235.43 | 241.67 | 249.64 | 249.96 | 242.82 |
| 8 | 245.15 | 250.5 | 260.89 | 260.96 | 254.97 |
| 9 | 270.76 | 270.11 | 269.43 | 260.79 | 248.4 |
| 10 | 266.26 | 270.69 | 278.23 | 278.64 | 270.12 |
| 11 | 276.67 | 282.76 | 293.67 | 292.75 | 281.93 |
| 12 | 301.94 | 301.04 | 304.68 | 304.17 | 291.05 |
| 13 | 314.84 | 319.33 | 325.53 | 322.95 | 311.46 |
| 14 | 348.9 | 340.49 | 342.16 | 345.23 | 333.83 |
| 15 | 401.41 | 389.97 | 389.69 | 393.25 | 373.45 |

Tabelle C.2: Codebook der Intonationschablonen-Klassen

C.2.2 Silbenklassen-Konturen



C.3 Der Tonschablonen-Entscheidungsbaum

Der Entscheidungsbaum wurde von den unübersichtlichen Klassenwahrscheinlichkeiten befreit und nur die jeweils wahrscheinlichste Klasse wird als Blatt angegeben; genauso arbeitet das BOSS-Modul.

```
((sylphrase < 15.7)
  ((sylphrase < 8)
    ((sylphrase < 3.4)
      ((ltone2 is P)
        ((ltone1 is P)
          ((phonessyl < 1.3)
            (50)
            (53)
            (50)
          )
          ((f < 0.1)
            ((wordphrase < 1.2)
              (45)
            )
          )
        )
      )
    )
  )
```

```
((catsylword is i)
  (45)
  (50)
  (45)
  ((sylphrase < 4.3)
    ((rtone2 is L)
      (25)
      ((ltone3 is L)
        (25)
        ((ltone2 is D)
          (25)
          ((wordsphrase < 8)
            )
        )
      )
    )
  )
```

| | |
|----------------------|-----------------------|
| (25) | (26) |
| ((wordphrase < 15.5) | ((rtone2 is H) |
| (45) | (34) |
| (25) | ((wordphrase < 5.5) |
| ((sylphrase < 6.2) | ((catsylword is f) |
| ((sylphrase < 5.1) | (26) |
| ((ltone4 is L) | ((ltone4 is H) |
| (32) | (26) |
| ((ltone3 is D) | (34) |
| (32) | (26) |
| ((rtone2 is L) | ((sylphrase < 10.1) |
| (32) | ((ltone3 is P) |
| (25) | (34) |
| ((catsylword is f) | ((ltone4 is L) |
| (32) | (40) |
| ((ltone1 is H) | ((r < 0.4) |
| (37) | (40) |
| ((ltone3 is H) | ((wordphrase < 8.4) |
| ((wordphrase < 3.2) | (40) |
| (37) | ((ltone1 is H) |
| (32) | (40) |
| (37) | (34) |
| ((rtone4 is H) | ((rtone4 is P) |
| (25) | ((catwordphrase is m) |
| ((ltone1 is L) | (22) |
| (25) | (40) |
| ((f < 1.2) | ((wordphrase < 14.2) |
| ((wordphrase < 6.2) | (22) |
| (25) | (40) |
| ((sylsword < 2.5) | ((sylphrase < 13.2) |
| (25) | ((sylphrase < 12.1) |
| (37) | ((sylsword < 3.4) |
| (25) | ((sylstruc is CV) |
| ((sylphrase < 11.5) | (10) |
| ((sylphrase < 9.2) | ((rtone4 is P) |
| ((sylphrase < 8.1) | (10) |
| ((f < 1.2) | (22) |
| ((ltone2 is L) | ((sylsword < 4.3) |
| (26) | (22) |
| ((wordphrase < 8.7) | (10) |
| (26) | ((sylstruc is NV) |
| ((ltone4 is H) | (10) |
| (26) | ((ltone2 is H) |
| (25) | (13) |

| | |
|---------------------|----------------------|
| ((catsylword is f) | (7) |
| (13) | ((ltone2 is H) |
| ((wordphrase < 6.5) | (8) |
| (10) | (7) |
| (13) | (8) |
| ((sylphrase < 14.1) | ((sylphrase < 23) |
| ((ltone3 is H) | ((sylphrase < 20.3) |
| (16) | ((wordphrase < 10) |
| ((phonessyl < 1.2) | ((wordphrase < 9.2) |
| (16) | (17) |
| ((rtone4 is P) | (8) |
| (13) | (17) |
| (16) | ((sylphrase < 21.2) |
| ((ltone4 is H) | ((ltone2 is L) |
| (27) | (17) |
| ((rtone2 is H) | ((rtone4 is P) |
| (27) | ((wordphrase < 11.2) |
| ((ltone3 is H) | (17) |
| (27) | (21) |
| (16) | (21) |
| ((sylphrase < 26.2) | ((sylphrase < 22.1) |
| ((sylphrase < 20) | ((catsylword is m) |
| ((sylphrase < 17.2) | (11) |
| ((sylphrase < 16.1) | ((ltone4 is H) |
| ((r < 1.5) | (21) |
| (23) | (11) |
| ((wordphrase < 16) | ((ltone4 is L) |
| ((sylsword < 3.5) | (11) |
| (23) | (17) |
| (27) | ((sylphrase < 24.2) |
| (23) | ((ltone1 is L) |
| ((sylsword < 2.2) | (17) |
| (9) | ((wordphrase < 13.8) |
| ((ltone3 is L) | (17) |
| (23) | (24) |
| (9) | ((sylphrase < 25.1) |
| ((sylphrase < 18.1) | ((rtone4 is L) |
| ((catsylword is m) | (12) |
| ((ltone4 is L) | ((sylsword < 2.2) |
| (9) | (12) |
| (7) | ((phrasefall < 5.7) |
| (7) | (24) |
| ((phonessyl < 2.2) | (12) |
| ((catsylword is f) | ((sylsword < 2.2) |

| | |
|---------------------|----------------------|
| (56) | (15) |
| (12) | (19) |
| ((sylphrase < 31.6) | ((sylphrase < 35.6) |
| ((sylphrase < 28.2) | ((sylphrase < 33.2) |
| ((sylphrase < 27.1) | ((sylphrase < 32.1) |
| ((ltone4 is H) | ((wordphrase < 19.5) |
| (52) | (19) |
| ((wordphrase < 15) | (42) |
| (56) | ((rtone4 is P) |
| (52) | (42) |
| ((phrasefall < 6.1) | (45) |
| (52) | (28) |
| (31) | ((sylphrase < 38.8) |
| ((sylphrase < 29.2) | ((sylphrase < 37) |
| ((phonessyl < 2) | (11) |
| (31) | (18) |
| (36) | ((sylphrase < 42.3) |
| ((sylphrase < 30.1) | ((sylphrase < 39.3) |
| ((rtone4 is P) | (25) |
| (36) | (56) |
| (15) | (9) |
| ((sylsword < 3.7) | |

D Tools

Die folgenden Tools sind weitestgehend in der Reihenfolge aufgeführt, wie sie während der Entwicklung und Korpuserstellung aufgerufen werden müssen. Dieser Anhang kann keine vollständige Dokumentation ersetzen, für Detailinformationen und Übergabeparametersyntax lohnt es sich oft in den Quelltext zu schauen.

D.1 Korpustools

bits/loo.pl

Erzeugt aus allen für die Synthese bestimmten Sätzen 10 unterschiedliche *leave-one-out*-Listen und die entsprechenden Listen von Testsätzen. Als Urliste wird die durch die Option `list` verknüpfte Datei benutzt ("files.list"). Die N Ausgabelisten heißen dann "files.listN_loo" und die Testlisten "files.listN_test".

pre/convert/addsyls.psc

Fügt den Original-TextGrids ein *tier* mit Silben hinzu. Die Information über die Silbengrenzen wird dem Tone-Tier entnommen.

pre/convert/addwords.psc

Fügt einen Tier mit Wörtern hinzu. Die Information über Wortgrenzen entstammt dem Phontier, in dem Wortgrenzen per `#` markiert sind.

bits/find-avg-f0.pl

Bestimmt die durchschnittliche Grundfrequenz des gesamten Korpus.

bits/extract-pulses.psc

Das Praat-Skript bestimmt die glottalen Verschlusszeitpunkte aller Klangdateien des Korpus.

esps2f0/prune-esps.pl

Extrahiert aus den mit Hilfe eines ESPS-Skript gewonnenen *.f0_ascii - Dateien die vier von BOSS benötigten Spalten F_0 , prob_voice, rms und ac_peak.

esps2f0/esps2f0.pl

Extrahiert die F_0 -Werte der 4-spaltigen *.f0_ascii-Dateien in das zeilenweise .f0-Format.

esps2f0/extract_syl_f0s.psc

Sammelt silbenweise alle F_0 -Werte ein und legt sie mit Tabulatoren getrennt, jeweils eine Silbe pro Zeile, in den *.f0s-Dateien ab.

polyreg/polyreg.pl & polyreg/polysylloop.pl

Führt für alle Sätze des Korpus die Polynomregression mit dem Skript polysylloop.pl durch. Dessen drei Parameter sind die XML-Datei, der Name der *.poly-Datei, in welche die Ergebnisse geschrieben werden, und der Dateiname der Polynom-Konfigurationsdatei poly.conf. Das Skript erstellt für jede Silbe eine temporäre Datei mit den möglicherweise normalisierten und logarithmierten Grundfrequenzwerten und ruft das Skript polysylsub.pl auf, welches die eigentliche Regression durchführt.

Der ursprüngliche Quelltext stammt von <http://www.we.fh-osnabrueck.de/fbwe/vorlesung/edv2>.

polyreg/resynth.psc

Resynthetisiert Sätze mit polynominellen Formen. Dies kann entweder per Codebook geschehen oder mit den originalen Regressionswerten der Polynome (siehe Tool plain-cb.pl).

polyreg/rmse.pl

Das Skript vergleicht originale und resynthetisierte Grundfrequenzwerte und bestimmt diverse Abstandsmaße dazu.

D.2 CART-Training

vectorq/collect-poly.pl

Dieses Script sammelt die Polynome aller für das Korpus bestimmten Sätze für die Bearbeitung mit dem folgenden Tool in einer großen Datei. Dabei werden nur Polynome

akzeptiert, deren Stützstellen keine 0 enthalten und die im Intervall $-4000 \dots +4000$ liegen.

vectorq/vq.pl

Führt die Vektorquantisierung für die gesammelten Polynome durch. Ruft `prepare2.pl` auf und vektorquantisiert die durch das Tool vorbereiteten Codevektoren erneut für die Codevektorklassen. Der ursprüngliche Quelltext stammt von <http://www.data-compression.com/vq>

vectorq/redistribute-cb.pl

Verteilt die erhaltene Liste aller Codevektoren zurück auf die einzelnen Sätze des Korpus (unter `inventory/cb`).

vectorq/plain-cb.pl

Erzeugt für jede Silbe einen eigenen Codebookeintrag mit genau den Stützstellen dieser Silbe. Entspricht einem beliebig großen Codebook und hundertprozentig korrekter Prädiktion und wird für die reine Stilisierungsgenauigkeitsmessung benötigt.

cart/concat-textgrids.psc & cart/concat-part-two.psc

Mit Hilfe dieser beiden Skripte wird in zwei Schritten aus allen für das Korpus bestimmten TextGrid-Dateien eine einzige große Datei erzeugt. Liest Listeninformation aus der Korpus-Liste `files.list`.

cart/concat-waves.psc

Fügt alle für das Korpus bestimmten Klangdateien zusammen.

cart/extract-params.psc

Extrahiert alle möglichen für die Tonschablonen- und Dauerprädiktion benötigten Parameter in eine `*.pred` - Datei.

cart/filter.pl

Löscht alle unerwünschten Zeilen aus der Prädiktionsparameterdatei und speichert die Ergebnisse in einer `*.predf` - Datei.

cart/testcbc.pl

Testet den Tonschablonenklassifikationsbaum auf der Testdatenmenge. Die Testdatenmenge wird so erstellt: Aufruf von `collect-poly.pl`, `vq.pl` und `redistribute-cb.pl` zur Erstellung des LOO-VQ. Dann nochmaliger Aufruf von `collect-poly.pl` und `redistribute-cb.pl` mit der Test-Liste. Das erzeugt die Datei `all.cb_`, welche die korrekten CV enthält. Jetzt wird für die Testdatenmenge ein TextGrid erstellt und mit `extract_params.psc` wie gewohnt die `*.predf`-Datei erzeugt. Mit dieser wird `wagon-test` gestartet und die Ergebnisse per `testcbc.pl` verglichen.

cart/testdur.pl

Testet den Dauerregressionsbaum auf der Testdatenmenge.

D.3 Datenbankerstellung

tg2xml.pl

Erzeugt aus allen TextGrids des Korpus XML-Dateien und füllt sie mit allen verfügbaren Attributen auf (`Type`, `File`, `Tkey`, `TReal`, `Tones`, `PMode`, `PInt`, `PNr`, `Broken`, `First`, `Last`, `Border`, `CB`, `CBC`, `BV0..BV5`, `PC0..PC5`, `PhPos`). Weitere Attribute werden dann später direkt mit BOSS-Tools in die XML-Dateien eingefügt.

bits/create-selection

Die Shellskripte `create-selection.bat` (für das Windows-) und `create-selection.sh` (für das Linux-Betriebssystem) rufen das Perl-Skript `create-selection.pl` für die drei Vorauswahlebenen Wort, Silbe und Phon auf.

Das Tool erwartet eine Eingabedatei mit Attributnamen und erzeugt die von BOSS eingelesenen *feature pruning*-Dateien (Fokusaufweitung der `mySQL`-Anfragen). Über die Standardausgabe werden die Suchschlüssel für die Definition der `mySQL`-Datenbank ausgegeben und in die entsprechenden `*.key`-Dateien umgelenkt. Das Format der Eingabedatei ist wie folgt definiert:

In jeder Zeile steht eine Liste von Attributnamen, durch Kommata getrennt. Die Liste muss mindestens ein Element haben (`Attribut_1 [,Attribut_2, ..., Attribut_n]`). Soll ein Attribut einen fixen Wert bekommen und nicht den von den anderen Modulen prädierten, so kann dies über die Syntax `Attribut=Wert` festgesetzt werden. Jede Suchanfrage besteht aus allen Attributen, aber nach und nach werden Attribute von unten beginnend fallengelassen. Für n Zeilen ergeben sich somit n Suchanfragen und Suchschlüssel. Soll die Suche nicht weiter aufgeschlüsselt werden, so setzt man ein `$`-Zeichen vor die erste Stelle der letzten benötigten Zeile. Zeilen können per `#`-Zeichen auskommentiert werden. Diese Syntax ist noch recht unflexibel, daher wird für

jede Anfrage (vor jedem Wegfallenlassen des untersten Attributes) jedes Listenelement einmal als gesuchtes Attribut eingesetzt, so dass alle Kombinationen vorkommen.

addcontext

Erweitertes Tool der deutschen BOSS-Distribution. Fügt auf allen Ebenen die Attribute CLeft, CRight, LTone1, RTone, CCLeft, CCRight, CCLeft2, CCRight2 hinzu.

addf0

Original-Tool der deutschen BOSS-Distribution. Fügt auf allen Ebenen die Attribute LF0, RF0, AVGF0 und RMS hinzu. Die Werte berechnen sich als Durchschnitt der ersten und letzten 33 % der Grundfrequenzwerte (LF0 & RF0) bzw. Gesamtdurchschnitt und Gesamtenergie der Einheit (AVGF0 & RMS).

optbounds

Gleicht die Einheitengrenzen an den Nulldurchgängen der Sprachsignale an (Original-BOSS-Tool).

melbounds

Original-Tool, das in die XML-Struktur die Information über die Mel-Cepstral-Koeffizienten hinzufügt (LM0..LM10 und RM0..RM10).

mySQL

Die mySQL-Skripte `create_tables.sql` und `drop_tables.sql` erzeugen und entfernen alle benötigten Tabellen der BOSS-Ibibio-Datenbank. Das Skript `go.sql` ruft beide Skripte auf und löscht und erzeugt die Datenbank neu.

blfxml2db

Das Tool fügt alle Inhalte der XML-Dateien des Korpus in die vorbereitete, leere Datenbank ein. Der Name "blf" stammt von BOSS-LabelFormat, dass in der deutschen Version genutzt wird. In BOSS_Ibibio wird die XML-Datei aber direkt und ohne Umweg über BLF erstellt.

D.4 Grafische Aufarbeitung

gfx/drawCodebook.psc

Erstellt *Encapsulated PostScript* EPS-Dateien mit den Polynomformen aller Codevektoren. Zu finden in Anhang C.

gfx/drawTones.psc

Zeichnet die Grafik aus Abbildung 1.1.

gfx/polys.psc

Erzeugt die einfachen Polynome auf Abbildung 4.1.

polyreg/draw-diff.psc

Erstellt die Grafiken der Polynomstilisierung in Abbildung 4.2.

Literaturverzeichnis

- Abercrombie, D. (1967): *Elements of general phonetics*. Edinburgh University Press.
- Adriaens, L. (1991): *Ein Modell deutscher Intonation. Eine experimentell-phonetische Untersuchung nach den perzeptiv relevanten Grundfrequenzveränderungen in vorgelesenen Text*. Dissertation, Technische Universität Eindhoven.
- Agüero, P. D.; A. Bonafonte (2005): “Consistent Estimation of Fujisaki’s Intonation Model Parameters.” In: *SPECOM 2005*, Patras, Greece.
- Agüero, P. D.; K. Wimmer; A. Bonafonte (2004): “Automatic Analysis and Synthesis of Fujisaki’s Intonation Model for TTS.” In: *Speech Prosody 2004*, Nara, Japan.
- Anderson, M.; J. Pierrehumbert; M. Liberman (1984): “Synthesis by rule of English intonation patterns.” *ICASSP*, 281–284.
- Aylett, M. (2004): “Merging data driven and rule based prosodic models for unit selection tts.” In: *5th ISCA Speech Synthesis Workshop*, Rhetorical Systems Ltd.
- Black, A. W.; W. N. Campbell (1995): “Optimising selection of units from speech databases for concatenative synthesis.” In: *Eurospeech*, Madrid, Spain, 581–584.
- Black, A. W.; A. J. Hunt (1996): “Generating F0 contours from ToBI labels using linear regression.” In: *Proceedings of ICSLP 96*, Philadelphia, PA, USA.
- Black, A. W.; P. Taylor (1997a): “Automatically clustering similar units for unit selection in speech synthesis.” In: *Proc. Eurospeech ’97*.
- Black, A. W.; P. A. Taylor (1997b): *The Festival Speech Synthesis System: System documentation*. University of Edinburgh, Scotland, UK: Human Communication Research Centre,
<http://www.cstr.ed.ac.uk/projects/festival.html> (2006-08-11).
- Breiman, L.; J. H. Friedman; R. A. Olshen; C. J. Stone (1984): *Classification and Regression Trees*. New York: Chapman and Hall (Wadsworth, Inc.).
- Campione, E.; E. Flachaire; D. Hirst; J. Veronis (1997): “Stylisation and symbolic coding of F_0 : A quantitative model.” In: *ESCA 1997*.

- Cohen, A.; R. Collier; J. 't Hart (1982): "Declination: Construct or intrinsic feature of speech pitch." In: *Phonetica*, Bd. 39, 254–273.
- Connell, B. (2001): "Downdrift, Downstep and Declination." In: *Typology of African Prosodic Systems Workshop*, Bielefeld University.
- Daelemans, W.; J. Zavrel; K. van der Sloot; A. van den Bosch (2004): *TiMBL: Tilburg Memory-Based Learner. Version 5.1, Reference Guide*. Induction of Linguistic Knowledge, Computational Linguistics, Tilburg University, <http://ilk.uvt.nl> (2006-08-11).
- d'Alessandro, C. (1995): "Automatic pitch contour stylization using a model of tonal perception." In: *Computer Speech and Language*, Bd. 9, 257–.
- Delattre, P.; E. Poenack; C. Olsen (1965): "Some characteristics of German intonation for the expression of continuity and finality." In: *Phonetica*, Bd. 13, 134–161.
- DuBois, P. (2006): *MySQL Administrator's guide and language reference, 2nd edition*. MySQL Press.
- Dusterhoff, K.; A. W. Black (1997): "Generating F0 contours for speech synthesis using the tilt intonation theory." In: *Proceedings of the 1997 ESCA Workshop on intonation*, Athens, Greece.
- Dusterhoff, K.; A. W. Black; P. Taylor (1999): "Using Decision Trees within the Tilt Intonation Model to Predict F₀ Contours."
- Dutoit, T.; V. Pagel; N. Pierret; F. Bataille; O. van der Vrecken (1996): "MBROLA project: Towards a set of high quality speech synthesisers free of use for non commercial purposes." In: *Proceedings of the 4th International Conference of Spoken Language Processing*, Philadelphia, USA, 1393–1396, <http://tcts.fpms.ac.be/synthesis/mbrola.html>, 2006-08-11.
- Escudero, D.; V. Cardenoso; A. Bonafonte (2002): "Corpus based extraction of quantitative prosodic parameters of stress groups in Spanish." In: *ICASSP 2002*.
- Essien, O. E. (1991): "The nature of tenses in African languages: a case study of the morphemes and their variants." In: *Archiv Orientalni*, Bd. 59, 1–11.
- Estelle Campione, J. V., Daniel Hirst (2000): "Automatic stylisation and symbolic coding of F0: Implementations of the INTSINT model." In: Botinis, A., Hg., *Intonation. Research and Applications*, Cambridge: Cambridge University Press.
- Fujisaki, H. (1988): "A note on the physiological and physical basis for the phrase and accent components in the voice fundamental frequency contour." In: Fujimura, O., Hg., *Vocal Physiology: Voice Production, Mechanisms and Functions*, New York: Raven Press, Kap. 30, 165–175.

- Fujisaki, H. (2004): "Information, Prosody, and Modeling – with Emphasis on Tonal Features of Speech." *Speech Prosody 2004* .
- Fujisaki, H.; K. Hirose; N. Takahashi; M. Yoko'o (1984): "Realization of accent components in connected speech." In: *Transactions of the Committee on Speech Research*, Acoustic Society of Japan, Bd. S84, 279–286.
- Fujisaki, H.; S. Nagashima (1969): "A model for the synthesis of pitch contours of connected speech." In: *Annual Report of the Engineering Research Institute*, University of Tokyo, Bd. 28, 53–60.
- Fujisaki, H.; S. Ohno (1996): "Prosodic Parameterization of Spoken Japanese Based on a Model of the Generation Process of F₀ Contours." In: *Proceedings of the ICSLP*, Philadelphia, PA, Bd. 4, 2439–2442.
- Fujisaki, H.; S. Ohno; O. Tomita (1996): "On the levels of accentuation in spoken japanese." In: *Proceedings of the ICSLP*, Philadelphia, PA, Bd. 2, 634–637.
- Gibbon, D.; E.-A. Urua; M. Ekpenyong (2006): "Problems and solutions in African tone language Text-To-Speech." In: *ISCA Workshop on Multilingual Speech and Language Processing (MULTILING 2006)*, Stellenbosch, South Africa: Center for Language and Speech Technology, Stellenbosch University, paper 014.
- Gordon, R. G., Hg. (2005): *Ethnologue: Languages of the World, Fifteenth edition*. Dallas, Texas: SIL International, <http://www.ethnologue.com> (2006-08-11).
- Grønnum, N. (1988): "Standard Danish intonation." In: *ARIPUC*, Univ. Copenhagen, Bd. 22, 1–23.
- Gu, W.; K. Hirose; H. Fujisaki (2004): "A Method for Automatic Tone Command Parameter Extraction for the Model of F₀ Contour Generation for Mandarin." *Speech Prosody 2004* .
- 't Hart, J.; R. Collier; A. Cohen (1990): *A perceptual study of intonation*. Cambridge: Cambridge University Press.
- Hess, W. (2004a): "Prosodie." Folien zur Vorlesung "Prosodie".
- Hess, W. (2004b): "Sprachsynthese." Folien zur Vorlesung "Systeme der akustischen Mensch-Maschine-Kommunikation 1,2", kapitel 3.
- Heuft, B. (1999): *Eine prominenzbasierte Methode zur Prosodieanalyse und -synthese*. Frankfurt am Main, Berlin, Bern, New York, Paris, Wien: Lang.

- Hirose, K.; H. Fujisaki; M. Yamaguchi (1983): "Automatic estimation of characteristic parameters of fundamental frequency contours." In: *Reports of the 1983 Spring Meeting*, Acoustic Society of Japan, Bd. 1, 93–94.
- Hirst, D. J. (1993): "Automatic modelling of fundamental frequency using a quadratic spline function."
- Hirst, D. J.; A. D. Cristo (1998): "A survey of intonation systems." In: Hirst, D. J.; A. D. Cristo, Hg., *Intonation systems. A survey of twenty languages*, Cambridge University Press, Kap. 1, 1–44.
- Hirst, D. J.; A. D. Cristo; R. Espesser (2000): "Levels of Representation and Levels of Analysis for the Description of Intonation Systems." In: Horne, M., Hg., *Prosody: Theory and Experiment*, Kluwer Academic Press.
- Hirst, D. J.; R. Espesser (1993): "Automatic modelling of fundamental frequency using a quadratic splinefunction." In: *Travaux de l'Institut de Phonétique d'Aix*, 15, 71–85.
- Hoffmann, R.; U. Kordon; S. Kürbis; B. Ketzmerick; K. Fellbaum (1999): "An interactive course on speech synthesis." In: *Proceedings of the ESCA/SOCRATES Workshop MATISSE*, 61–64.
- IPA (1999): *Handbook of the International phonetic association. A guide to the use of the international phonetic alphabet*. Cambridge: CUP, <http://www.cambridge.org> (2006-08-11).
- Isačenko, A. V.; H. J. Schädlich (1966): "Untersuchungen über die deutsche Satzintonation." In: *Studia Grammatica*, Bd. VII, 7–64.
- Kochanski, G.; C. Shih (2001a): "Automated modelling of Chinese intonation in continuous speech."
- Kochanski, G.; C. Shih (2001b): "Prosody modelling with soft templates."
- Kohler, K. J. (1991a): "A model of German intonation." In: *Arbeitsberichte des Instituts für Phonetik und digitale Sprachverarbeitung der Universität Kiel (AIPUK)*, Bd. 25, 295–360.
- Kohler, K. J. (1991b): "Terminal intonation patterns in single-accent utterances of German: Phonetics, phonology, and semantics." In: *Arbeitsberichte des Instituts für Phonetik und digitale Sprachverarbeitung der Universität Kiel (AIPUK)*, Bd. 25, 115–185.
- Kohler, K. J. (1997a): "Modelling prosody in spontaneous speech." In: Sagisaka, Y.; N. Campbell; N. Higuchi, Hg., *Computing Prosody*, New York: Springer, 187–210.

- Kohler, K. J. (1997b): "Parametric control of prosodic variables by symbolic input in TTS synthesis." In: van Santen, J. P. H.; R. W. Sproat; J. P. Olive; J. Hirschberg, Hg., *Progress in Speech Synthesis*, New York: Springer, 459–475.
- Kriesel, D. (2006): "Ein kleiner Überblick über Neuronale Netze." Vorläufige Version.
- Ladd, D. (1989): "Metrical representation of pitch register." In: Kingston, J.; M. Beckman, Hg., *Papers in laboratory phonology I*, Cambridge: Cambridge University Press, 35–57.
- Ladd, R. (1996): *Intonational Phonology*. Cambridge University Press.
- Lewis, R. J. (2000): "An Introduction to Classification and Regression Tree (CART) Analysis."
- Lieberman, M. Y.; J. B. Pierrehumbert (1984): "Intonational invariance under changes in pitch range and length." In: Aronoff, M.; R. Oehrle, Hg., *Language Sound Structure*, The MIT Press, 157–233.
- Linde, Y.; A. Buzo; R. M. Gray (1980): "An algorithm for vector quantizer design." In: *IEEE transactions on communications*, Bd. 28, 84–95.
- Louw, J. A.; E. Barnard (2004): "Automatic intonation modeling with INTSINT." *Proceedings of the 15th Annual Symposium of the Pattern Recognition Association of South Africa*, 107–111.
- Möbius, B. (1993): *Ein quantitatives Modell der deutschen Intonation: Analyse und Synthese von Grundfrequenzverläufen*. Tübingen: Max Niemeyer Verlag.
- Möhler, G. (1998a): "Describing intonation with a parametric model." In: *Proc. ICSLP*.
- Möhler, G. (1998b): "Theoriebasierte Modellierung der deutschen Intonation für die Sprachsynthese." Dissertation.
- Möhler, G. (2001): "Improvements of the PaIntE model for F_0 parametrization. Technical report (draft version)."
- Möhler, G.; A. Conkie (1998): "Parametric modeling of intonation using vector quantization."
- Mixdorff, H. (1998): *Intonation patterns of German. Model based quantitative analysis and synthesis of F_0 contours*. Dissertation, Technische Universität Dresden.
- Mixdorff, H. (2000): "A novel approach to the fully automatic extraction of Fujisaki model parameters."

- Moore, A. W. (2005): "Instance-based learning." Folien-Präsentation.
<http://www.cs.cmu.edu/~awm/tutorials> (2006-08-11).
- Mouline, S.; O. Boeffard; P. Bagshaw (2004): "Automatic adaptation of the MoMel F0 stylisation algorithm to new corpora." In: *INTERSPEECH-2004*, 753–756,
http://www.isca-speech.org/archive/archive_papers/interspeech_2004/i04_0753.pdf
(2006-08-11).
- MySQL (2006): "MySQL 5.0 Reference Manual."
<http://www.mysql.org> (2006-08-11).
- Nukaga, N.; R. Kamoshida; K. Nagamatsu (2004): "Unit selection using pitch synchronous cross correlation for japanese concatenative speech synthesis." In: *5th ISCA Speech Synthesis Workshop*, 43–48.
- Pierrehumbert, J. (1980): *The phonology and phonetics of English intonation*. MIT Press, Cambridge.
- Pike, K. L. (1945): *The intonation of American English*. Ann Arbor: University of Michigan Press.
- Pike, K. L. (1948): "Tone languages. A technique for determining the number and type of pitch contrasts in a language, with studies in tonemic substitution and fusion."
- Plauger, P.; A. A. Stepanov; M. Lee; D. R. Musser (2000): *The C++ standard template library*. Prentice Hall PTR.
- Portele, T. (1998): "Just concatenation - A corpus-based approach and its limits." In: Campbell, N.; A. Breen; J. van Santen; J. Vonwiller, Hg., *Proceedings of the Third ESCA / COCOSA workshop on speech synthesis*, Jenolan Caves, Blue Mountains, Australia.
- Rao, K. S.; B. Yegnanarayana (2004): "Intonation modeling for Indian languages."
- Raux, A.; A. W. Black (2003): "A unit selection approach to F0 modeling and its application to emphasis."
<http://www.cs.cmu.edu/~awb/papers/asru2003/f0clunits.pdf> (2006-08-11).
- Rojc, M. (2006): "Improving the fitting accuracy of re-constructed F0 contours." In: Kačič, Z., Hg., *Advances in speech technology: proceedings / the twelfth international workshop*, Maribor, Slovenien: University of Maribor, Faculty of electrical engineering and computer science, 21–26.
- Sakai, S. (2004): " F_0 modeling with multi-layer additive modeling based on a statistical learning technique."

- van Santen, J. P. H.; T. Mishra; E. Klabbers (2004): "Estimating Phrase Curves in the General Superpositional Intonation Model." In: *5th ISCA Speech Synthesis Workshop*, Pittsburgh.
- Schröder, M.; J. Trouvain (2003): "The German Text-to-Speech Synthesis System MARY: A Tool for Research, Development and Teaching."
<http://mary.dfki.de> (2006-08-11).
- Silverman, K.; M. Beckman; J. Pitrelli; M. Ostendorf; C. Wightman; P. Price; J. Pierrehumbert; J. Hirschberg (1992): "ToBI: A Standard for Labeling English Prosody." In: *Proc. ICSLP*, 867–870.
- Stoer, J. (2005): *Numerische Mathematik 1. Eine Einführung - unter Berücksichtigung von Vorlesungen von F. L. Bauer*. Berlin: Springer, neunte Auflage.
- Sun, X. (2002a): *The Determination, Analysis, and Synthesis of Fundamental Frequency*. Dissertation, Northwestern University.
- Sun, X. (2002b): " F_0 generation for speech synthesis using a multitier approach."
- Syrdal, A. K.; G. Möhler; K. Dusterhoff; A. Conkie; A. W. Black (1998): "Three methods of intonation modelling."
- Taylor, P. A. (1998): "The tilt intonation model." In: *Proceedings of the ICSLP*, 4, Sydney, 1383–1386.
- Taylor, P. A. (2000): "Analysis and Synthesis of Intonation using the Tilt Model." *Journal of the Acoustical Society of America* 107(3), 1697–1714.
- Taylor, P. A.; A. W. Black; R. Caley (1998): "The architecture of the festival speech synthesis system." In: *The Third ESCA Workshop in Speech Synthesis*, Jenolan Caves, Australia, 147–151,
<http://www.cstr.ed.ac.uk/projects/festival/> (2006-08-11).
- Tebelskis, J. (1995): *Speech Recognition using Neural Networks*. Dissertation, School of Computer Science, Carnegie Mellon University, Pittsburgh, Pennsylvania.
- Tutz, G. (2005): "CART – Classification and Regression Trees." Folien-Präsentation.
- Urua, E.-A. E. (2001): "The tone system of Ibibio." In: *Typology of African Prosodic Systems Workshop*.
- Urua, E.-A. E. (2004): *Ibibio*, Nr. 34/1 in Journal of the international phonetic association, International phonetic association, Kap. Ibibio. 105–109.

- Wahlster, W. (1997): “VERBMOBIL: Erkennung, Analyse, Transfer, Generierung und Synthese von Spontansprache.” *Informatik 97 – Informatik als Innovationsmotor Heidelberg*, 215–224
<http://www.dfki.de/~wahlster> (2006-08-11).
- Wendemuth, A. (2004): *Grundlagen der stochastischen Sprachverarbeitung*. München, Wien: Oldenbourg Verlag.
- Yu, M.-S.; N.-H. Pan; M.-J. Wu (2002): “A statistical model with hierarchical structure for predicting prosody in a Mandarin text-to-speech system.”
- Zwicker, E. (1982): *Psychoakustik*. Berlin: Springer.