

SEG_TOOL

Ein automatisiertes Segmentationsverfahren zur Verarbeitung gesprochener Sprache auf Phonebene durch die Analyse von Formanten

Modulprüfung des Schwerpunktmoduls 1 „Verarbeitung von Texten“ im Seminar:
Korpuslinguistik

vorgelegt von:

Anne-Kathrin Gerlach

Matrikelnummer: 5986095

E-Mail: agerlac3@smail.uni-koeln.de

vorgelegt am: 25.09.2019

Prüfer: Prof. Dr. Andreas Witt

Inhaltsverzeichnis

1. Einleitung.....	3
2. Literatur und Programme zur phonetischen Analyse.....	3
3. Idee und Hypothesen.....	6
4. Vorbereitung der Daten.....	8
5. Programmierung des SEG_TOOLS.....	9
5.1 Klassenstruktur.....	9
5.2 Analysestruktur.....	10
5.2.1 Einlesen & Aufbereiten der Daten.....	10
5.2.2 Kernanalyse – Verarbeitung der Daten.....	11
5.2.3 Ausgabe der Daten.....	15
6. Anwendung und Ergebnisse.....	16
7. Fazit und Ausblick.....	20
Literaturverzeichnis	II.....21
Abbildungs- und Tabellenverzeichnis	III.....22
Anhang	IV.....23
Formblatt	V.....29

1. Einleitung

Kann eine akustische Sequenz auf phonebene segmentiert werden? Das ist die „Frage nach der Segmentierbarkeit“, die Machelett (1996) anspricht und insofern bejaht, als dass bei einem Spektrogramm mit dem bloßen Auge bereits Segmentgrenzen erkennbar seien. In dieser schriftlichen Ausarbeitung soll dieser Frage nachgegangen werden; dafür wurde ein Segmentationstool (SEG_TOOL) entwickelt, welches versucht, diese mit bloßem Auge erkennbaren Segmentgrenzen ohne visuellen Input aufgrund einer Datenbasis aus Formantwerten automatisiert zu erkennen. Zunächst wird ein Überblick über phonetische Analysen gegeben und neben der Literatur auch über die Tools PRAAT und BAS Webservice gesprochen. Anschließend wird die Fragestellung genauer erläutert und die Idee für das Projekt skizziert. Um Segmentationen zu erstellen, werden sowohl Input als auch Vergleichsdaten benötigt; die Aufbereitung dieser wird im nächsten Kapitel beschrieben. Dann folgt der Hauptteil, in dem das geschaffene SEG_TOOL in seiner Programmierung vorgestellt wird, einmal die Struktur der Klassen und dann die Analysestruktur. Ebenso wird darüber gesprochen, wie das Einlesen und Ausgeben der Daten funktioniert. Anschließend folgt ein Anwendungsbeispiel und der Versuch, die Ergebnisse durch verschiedene Rechenmöglichkeiten zu verbessern. Die Analyse mit den besten Ergebnissen wird abschließend bei weiteren Audioaufnahmen getestet und die endgültigen Ergebnisse diskutiert. Zuletzt wird ein Fazit gezogen.

2. Literatur und Programme zur phonetischen Analyse

Zunächst soll ein kurzer Überblick über den wissenschaftlichen Hintergrund zur Verarbeitung gesprochener Sprache gegeben werden.

FORMANTEN Grundlage des SEG_TOOLS ist die Analyse der Formantenstruktur einer Audiodatei. Formanten haben besonders bei Vokalen sehr charakteristische Konturen. Bei der Artikulation eines Lautes kommt es durch die Schwingungen der Stimmlippen zu einer sogenannten Grundschiwingung; die dadurch entstandene Luftsäule wird wiederum durch Teilräume im Bereich zwischen Stimmlippen und Lippen zum Schwingen angeregt (Becker 2012: 46). Hierbei sind sowohl „Kehlkopfhöhe, Rachenenge, Zungenposition und -höhe sowie die Lippenstellung“ an der Modulierung der Töne beteiligt (vgl. Machelett 1996). Diese, dadurch entstehenden Eigenschwingungen werden verstärkt und die Grundschiwingung gedämpft. Es kommt zu sogenannten „Lautstärkemaxima im Obertonspektrum eines Vokals“, die als

Formanten bezeichnet werden (vgl. Becker 2012:46f.). Betrachtet man nun eine Audiodatei im Spektrogramm, können Formanten visualisiert dargestellt werden. Durch die Dreidimensionalität dieser, werden Sprachsignale neben Frequenz und Bandbreite, auch auf zeitlicher Ebene dargestellt (vgl. Becker 2012: 50). Die Zeit (ms) wird somit auf der x-Achse abgebildet, die Frequenz(Hz) auf der y-Achse und die Energie(dB) auf der z-Achse. Der Begriff Energie wird hier äquivalent zu Lautstärkemaximum eines Formanten verwendet und durch den Begriff Bandbreite ausgedrückt. Erkennbar sind die Formanten durch einen Schwärzungsgrad im Spektrogramm. Je höher der Bandbreitenwert ist, desto intensiver der Schwärzungsgrad (vgl. Reichel 2007).

LAUTKLASSEN Da sich die ersten drei Formanten besonders für die Vokaldeskription eignen, stellt sich die Frage, inwiefern andere Lautklassen sich voneinander unterscheiden lassen, besonders in Hinblick auf die Formantenkontur (vgl. Machelett 1996). Grundsätzlich unterscheiden sich Konsonanten von Vokalen durch einen niedrigeren Formant F1, da bei der Produktion von Konsonanten eine „größere artikulatorische Enge [entsteht]“ (vgl. Machelett 1996).

Plosive zeichnen sich durch ein sehr charakteristisches Bild im Spektrogramm aus. Akustisch entsteht ein Plosiv durch drei Phasen: Zuerst kommt die Verschlussbildung, die sich durch einen „abrupte[n] Abfall der spektralen Energie“ zeigt. Anschließend folgt die Verschlussphase, die „durch eine völlige ‚akustische Stille‘ in allen Frequenzbereichen des Spektrums“ erscheint. Bei der Verschlusslösung kommt es zu einem plötzlichen „Anstieg der spektralen Energie“. Visuell sind Plosive deshalb relativ leicht auffindbar. Energiemaxima lassen sich bei Plosiven (vorwiegend stimmhaften) meist nur im Bereich des F1 Formanten orten und beim der Verschlusslösung (vgl. Machelett 1996).

Nasale werden durch eine Verschlussbildung der Lippen/Zungenspitze/Zungenrücken und das Absenken des Velums gebildet (umformen!) (vgl. Machelett 1996). Der Formant F1 befindet sich meistens in einem niedrig frequentem Bereich; er dominiert das Spektrum (Fry 1979: 119). F2 ist sehr schwach ausgeprägt oder fehlt völlig. Mehrere höhere Formanten geringer Intensität sind manchmal zu erkennen. Einer von ihnen liegt bei etwa 2200 Hz. (umformen) (vgl. Machelett 1996).

Laterale werden durch das Berühren der Alveolen mit der Zungenspitze artikuliert. Laterale verfügen meist über eine relativ ausgeprägte Formantenstruktur, hat allerdings auch konsonantische Merkmale (niedriger F1). Durch die Art der Artikulation sind laterale im Vergleich zu Vokalen meist etwas gedämpft (vgl. Machelett 1996).

Trills sind Laute, die aus „einer Folge von Verschlüssen und Verschußlösungen besteh[en].“ Dadurch sind sie im Spektrogramm sehr eindeutig zu erkennen (vgl. Machelett 1996).

Frikative sind Laute, die durch eine „Engbildung im Mundraum“ entstehen, durch die dann ein Luftstrom gedrückt wird. „Der Luftstrom, der diese Enge passiert, wird turbulent“. „Das Frikativspektrum weist wesentlich mehr Intensität in den höheren Frequenzbereichen oberhalb von 2500 Hz auf als in den unteren Frequenzbereichen. Je nach Artikulationsort konzentriert sich dieses 'Rauschen' auf bestimmte Frequenzbereiche.“ (Bereiche in denen sonst die F4/5 liegen) (vgl. Machelett 1996). Weiter gibt es noch Affrikaten, Glottalstops (...)

Auf den ersten Blick lassen sich die beschriebenen Lautklassen im Spektrogramm relativ leicht ausfindig machen. Die genaue Segmentsetzung zwischen den einzelnen Lauten ist allerdings schwieriger. Einmal spielen hier die umliegenden Laute eine wichtige Rolle. Beispielsweise bleibt die Formantkontur während eines Vokals nur „relativ“ konstant. Formanten bewegen sich fallend oder steigend hin Abhängigkeit von den den Laut umgebenden Lauten. Diese Formantbewegungen werden Transitionen genannt vgl. Becker 2012: 53f.). Diese spielen eine wichtige Rolle bei der Koartikulation, wobei sich Laute ihren Umgebungen und deren Charakteristiken anpassen (vgl. Hardcastle 2006: 7). Trotzdem soll im Folgenden versucht werden eine Audiodatei, zu segmentieren. Zwar überlappen sich Laute, aber trotzdem verfügen die Lautklassen über relativ einzigartige Charakteristika, die durch die Formanten beschrieben werden können.

Für die phonetische Analyse von Audiodateien eignet sich beispielsweise das OpenSource Programm **PRAAT** (Weenink 2019): Mit diesem können Audiodateien manipuliert, gelabelt und in vielerlei Hinsicht analysiert werden¹ (Boersma 2018). Beispielhaft sieht man hier die Transkription einer Audiodatei (Abbildung1). Diese wird einmal im Oszillogramm dargestellt und im Spektrogramm. Die darunterliegende Transkription stammt vom BAS Webservice (s.u.). Die roten Punkte im Spektrogramm visualisieren einzelne Formantenmesspunkte auf der darunterliegenden schwarzverfärbten Formantenkontur. Hierbei ist die zeitliche Achse (x-Achse) in Sekunden angegeben. Die einzelnen Formantenmesspunkte sind allerdings vertikal in Frames geordnet die jeweils 0.00625s betragen. Diese kann bei Praat als Formantenanalyse ausgegebenwerden (vgl. Boersma 2018) (Anleitung im Anhang s. Tab. 4).

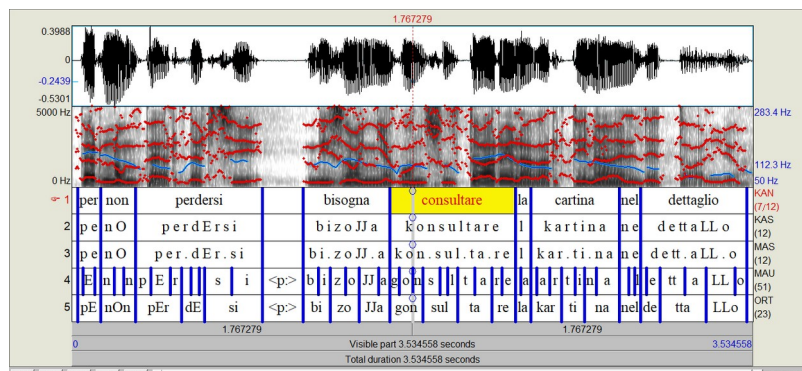


Abbildung 1: Transkription des BAS Tools

Für Segmentationsprozesse und automatisierte Transkriptionen gibt es beispielsweise den Webservice des Bavarian Archive for Speech Signals (**BAS**), der Universität München² (Kisler 2017). Dieses steht seinen Nutzern kostenfrei zur Verfügung und liefert Transkriptionen/Segmentationen auf allen sprachlichen Ebenen hier beispielhaft eine Transkription von BAS, die bei PRAAT geöffnet wurde (s. Abb. 1).

3. Idee und Hypothesen

Die Idee zu diesem Projekt entstand bei meiner Arbeit als Hilfskraft im Institut für deutsche Sprache und Literatur I im Sonderforschungsbereich *Prominenz in Sprache* für die Doktorandin Caterina Ventura. Sie forscht zu prosodischen Strukturen in Italienisch und Deutsch. Im Zuge ihrer Dissertation entstanden viele Audiodateien, die ich transkribiert habe. Da sie Transkriptionen auf Phonebene benötigt hat, war die Anfertigung dieser sehr zeitaufwendig. Zwar gibt es wie oben (s.u. Kapitel 2.)

¹ Praat-Homepage unter: www.praat.org (Boersma 2018)

² BAS-Homepage unter: <https://www.bas.uni-muenchen.de/Bas/> (Kisler 2017)

beschrieben beispielsweise den Webservice von BAS, aber auch hier sind die Transkriptionen auf Phonebene nicht immer genau. Zu einer Audiodatei wurde eine Transkription mit BAS angefertigt und eine händisch selbst erstellt. Dabei zeigt sich, dass bei 50 Segmentgrenzen lediglich 12 übereinstimmend waren; wobei hier eine Übereinstimmung der Segmentgrenzensetzung auf das Frame genau gemeint ist. Da ein Frame allerdings nur 0.00625s lang ist, und Laute oft in einander überlappen, wurden auch Segmentgrenzen als Treffer gewertet, die bis zu ± 2 Frames von derselben Trennstelle des Vergleichsdokumentes abweichen. Diese Grenze ist vertretbar, da es oftmals keine auf die (ms) richtige Trennstelle gibt, sondern eher einen Bereich, welcher mehrere Frames enthält, die als richtig gelten können. Mit Einbezug dieser Grenztoleranz lässt sich die Übereinstimmung zwischen der BAS Transkription und der manuell gefertigten auf 72% Übereinstimmung anheben (s. Anhang, Tabelle 1). Trotzdem wird deutlich, dass gerade bei einer phonetisch engen Transkription auch Onlinetools wie BAS an ihre Grenzen kommen und eine „richtige“ Segmentation auch von Fragestellung abhängt. Deshalb entstand die Idee zu dem Projekt Seg_Tool, um bestenfalls ein Segmentationstool zu erschaffen, welches eine ähnliche oder sogar bessere Segmentation liefern soll.

Hypothese: Wie in der Literatur erläutert, haben Laute gewisse Charakteristika, die sich im Spektrogramm durch mehr oder weniger eindeutige Muster widerspiegeln. Die Formanten dienen hierbei als eine Art ‚Fingerabdruck‘ für Laute, die über ihre Lautsegmente hinweg, eine relativ konstante Formantenkontur besitzen. Da meist Laute verschiedener Lautklassen aufeinander folgen, sollte es somit einen Wechsel an Formantenmustern von Laut zu Laut geben. Diese Wechsel sollten sich in den Formanten, bestehend aus Frequenz- (Hz) und Bandbreite(dB)-werten widerspiegeln. Dabei werden die Veränderungen dieser Werte von Frame zu Frame als Differenzwerte berechnet. Bei konstantbleibenden Werten über eine gewisse Anzahl an Frames hinweg, handelt es sich wahrscheinlich um einen Laut. Sollten sich die Differenzwerte aber signifikant zum nächsten Frame hin verändern, könnte dies auf eine Segmentgrenze hindeuten.

Umsetzung: Dieses Projekt ist wie in Abbildung 2 ersichtlich, strukturiert:

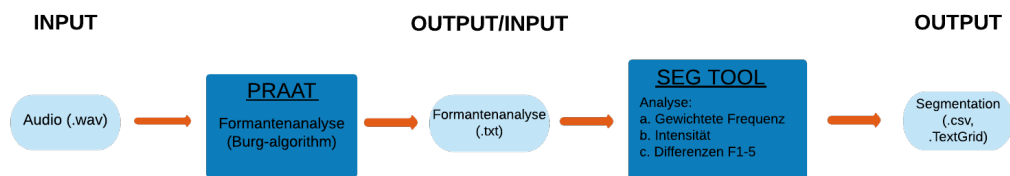


Abbildung 2: Segmentationsverfahren von Input zu Output

Als Inputdatei wird eine Audiodatei gewählt, die in PRAAT geöffnet wird. Wie zu Abbildung 1 erläutert, kann hier die Formantenanalyse extrahiert werden. Dies geschieht in einer .txt Version, die Werte pro Frame für die Formanten jeweils über Frequenz (in Hz) und Bandbreite (in dB) enthält. Diese Datei ist die Outputdatei aus PRAAT und zeitgleich die Inputdatei für das SEG_TOOL. Hier wird die Formantenanalysedatei eingelesen, aufbereitet und die Werte über die Frames hinweg analysiert. Daraus lassen sich Treffer für Segmentgrenzen ermitteln, die dann einmal für die weitere statistische Analyse als CSV-Datei und zudem als TextGrid für die Visualisierung in PRAAT erstellt werden.

4. Vorbereitung der Daten

Wie zuvor bereits angedeutet, benötigt es einiger Aufbereitung der Daten für die Erstellung, Benutzung und Auswertung des Segmentationstools. Grundsätzlich werden drei Datentypen benötigt:

Audiodateien als Input (1): Die verwendeten Audiodateien stammen aus deinem Dissertationsexperiment von Caterina Ventura und werden mit ihrer Erlaubnis für dieses Projekt genutzt. Es handelt sich dabei um Aufnahmen eines italienischen Muttersprachlers, der in insgesamt 10 Aufnahmen, jeweils einen Satz spricht. Die Sätze haben alle eine ähnliche Struktur und doppelten sich teilweise nur mit anderer Akzentsetzung (s. Anhang, Tabelle 2). Die Label wurden so übernommen. Dadurch, dass sie aus einem anderen universitären Projekt stammen, ist gewährleistet, dass die Aufnahmen qualitativ hochwertig, an einem ruhigen Ort aufgenommen und deutlich gesprochen sind. (s. Tabelle 2 im Anhang)

S_07_1_6_NF

Per sedersi comodamente bisogna mettere il cuscino sullo sgabello.

Um bequem zu sitzen, muss man ein Kissen auf den Hocker legen.

S_07_1_7_NF

Quando si arreda la casa bisogna scegliere il divano per il soggiorno.

Wenn man das Haus einrichtet, muss man ein Sofa für das Wohnzimmer auswählen.

Die Audiodateien liefern die Grundlage für die beiden anderen Dokumententypen zur Vorbereitung:

Formantenanalyse aus PRAAT als Input/Output (2): Wie bereits beschrieben werden die Audiodateien bei PRAAT eingelesen und die Formantenanalyse extrahiert. Dies geschieht durch den Burg-Algorithmus, den PRAAT verwendet, bei dem es sich um einen *linear prediction coefficient* Verfahren handelt, bei dem bei der spektralen Analyse u.a. Formanten unter 50Hz und über 50 Hz subtrahiert vom höchsten Formanten (maximum formant) als Artefakte rausgefiltert werden. Dadurch erhält man eine gefilterte Formantenanalyse, die die „wirklichen“ Formanten darstellt (Boersma 2018, s. Praat manual). Die Output Datei (Formantenanalyse.txt) ist so verschachtelt, dass jedes Frame[n] Informationen über die Anzahl an Formanten, den max. Intensity-Wert des Frames und die jeweils eingerückten Formanten[n] enthält, die wiederum aus Frequenz (Hz) und Bandbreite(dzB) bestehen (s. Abbildung 4, Kapitel 5) (vgl. Boersma 2018, s. Praat manual).

Transkriptionen von BAS (3): Um Transkriptionen von BAS zu extrahieren, benötigt man neben den Audiodateien (.wav) noch jeweils Textdateien (.txt) mit dem in der Audiodatei gesprochenen Sätzen (orthographisch). Diese lädt man zusammen hoch. Für dieses Projekt wurde die Pipeline *G2P -> MAUS-> Phy2SyL* verwendet, um Transkription und Segmentation auf allen sprachlichen Ebenen zu erhalten (auch wenn eigentlich nur die phonemische relevant ist). Anschließend erhält man eine TextGrid Datei. Da die Transkriptionen von BAS als Vergleichskonstanten für das SEG_TOOL genutzt werden, wurden anschließend noch CSV Dateien mit den Segmentationsgrenzen aus dem BAS TextGrid erstellt.

Wie in Abbildung 4 ersichtlich, ist somit ein statistischer Vergleich zu den Segmentgrenzen zwischen BAS und SEG_TOOL leichter möglich. Außerdem liegen beide Transkriptionen später ebenfalls als TextGrid vor, was eine Visualisierung und Verwendung bei PRAAT erleichtert.

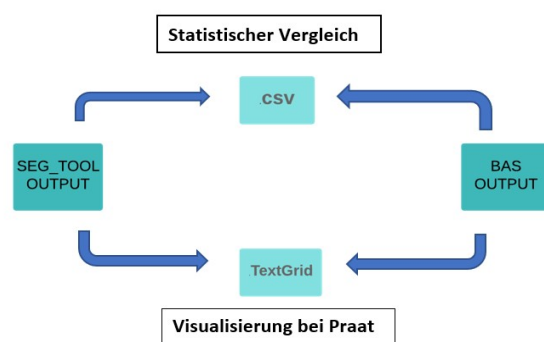


Abbildung 3: Vergleich BAS & SEG Output

5. Programmierung des SEG_TOOLS

Das Projekt wurde in der Java-Umgebung Eclipse erstellt und in Java programmiert.

5.1 Klassenstruktur

Das einzulesende Dokument (Formantenanalyse.txt) wird in der Datenstruktur der Klassen des SEG_TOOLS so repliziert, dass die zu verarbeitenden Werte des Input-

Dokumentes als instanziierte Objekte durch diese Klassenstruktur beschrieben werden können. Dies ist vergleichbar mit einem unique Label für hierarchisch geordnete Daten und spiegelt den Aufbau des Dokumentes Formantenanalyse.txt wider (genauere Informationen s. Anhang unter 3).

Hier sieht man in Abb. 4 den Inhalt des Dokumentes Formantenanalyse.txt. Unter der Klasse **Datensatz** wird das gesamte Dokument gefasst; mit den Attributen *xmin*, *xmax*, *nx*, *dx*, *x1* und einer Arrayliste mit Plätzen für die Objekte der Klasse **Frame**. Die Klasse **Frame** wiederum spiegelt die Struktur eines jeden *frames* wider. Hier zu gehört ein *intensity* Wert, der die maximale Lautstärke eines frames markiert und eine Arrayliste mit Plätzen für Objekte der Klasse **Formant**.

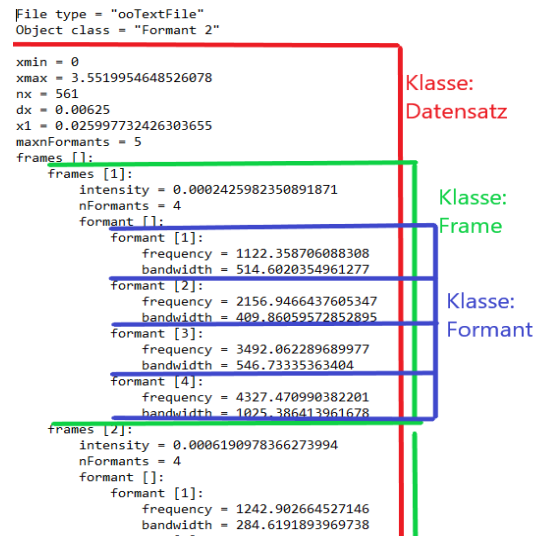


Abbildung 4: Klassenstruktur am Beispiel der Formantenanalyse

Die Klasse **Formant** umfasst auf unterster Ebene jeweils den Formanten an sich, bestehend aus einem Frequenz- und Bandbreitewert. Diese sind die Kernelemente der Verarbeitung. Die Anzahl der Formanten, die ein Frame umfasst, wird durch die Variable *nFormants* ausgedrückt. Durch das Verwenden von Arraylists wird *nFormants* überflüssig, da sich diese Listen der Anzahl an Einträgen dynamisch anpassen. Nach der Analyse und Verarbeitung der Daten gibt es für die Speicherung der gefundenen Trennstellen zwei weitere Klassen. Die Klasse **Loesungssatz** umfasst eine dynamische Arraylist mit Platzhaltern für Objekte der Klasse **Trennstelle**. Die Klasse **Trennstelle** wiederum beinhaltet ein Array welches die Kennwerte einer Trennstelle enthält und eine Variable *Position*, die die Framenummer wiedergibt, an welcher sich der Treffer befindet. Eingabe- und Ausgabestrukturen sind somit durch hierarchische Array(list)-Gebilde strukturiert, die Verwendung von Arraylisten zieht sich dementsprechend auch durch die Analyse der Daten.

5.2 Analysestruktur

Für die eigentliche Verarbeitung der Daten wurde die Klasse **Analyzer** erstellt. Hier wird das Input Dokument im ersten Schritt (5.2.1) eingelesen und aufbereitet. Dann werden die Daten im Kern des Programmes verarbeitet und verrechnet (5.2.2) und letztlich werden die Trennstellen ausgegeben (5.2.3).

5.2.1 Einlesen & Aufbereiten der Daten

Einlesen: Die Datei (Formantenanalyse.txt) wird über ein Auswahlfenster ausgewählt, nach Überprüfung, ob es sich auch um ein Dokument vom Typ (.txt) handelt. Anschließend wird eine ArrayList<content> erzeugt, um die Inhalte der Datei einzulesen. Die Datei wird dann zeilenweise eingelesen und in die ArrayList <content> geschrieben.

Aufbereiten: Dann wird die ArrayList<content> wiederum zeilenweise durchgegangen und deren Inhalt anhand des Gleichheitszeichens in Präfixe und Suffixe aufgeteilt. Die Präfixe verweisen auf den Platz, den der Inhalt (Suffix) in der Datenstruktur zugewiesen bekommen soll.

Bandwidth	=	667.8547744830787
Präfix	=	Suffix

Die Suffixe werden dann durch die Präfixe in ein Objekt der Klasse Datensatz <daten_aus> geschrieben. Dieses bildet dann die Struktur des ursprünglichen Input-Dokuments mit den gesamten Werten ab; nun kann aber auf die Werte als numerische Instanzen zugegriffen werden. Die Daten sind aufbereitet und bereit verarbeitet zu werden.

5.2.2 Kernanalyse – Verarbeitung der Daten

Die Verarbeitung der Daten ist im SEG_TOOL in drei Analyseschritte aufgeteilt, die alle dynamisch erweiterbar sind: Zunächst können aus den aufbereiteten Daten Kennwerte definiert werden. Danach werden diese über ihre Differenzen von Frame zu Frame verrechnet und durch das Setzen von Bedingungen zu vorläufigen Trennstellen zugelassen. Zuletzt werden die vorläufigen Treffer auf ihre Streuung in den Frames auf zeitlicher Ebene analysiert und falsche Treffer entnommen. Alle drei Analyseschritte sind in Arrays organisiert, um eine dynamische Verarbeitung und spätere Erweiterung zu ermöglichen.

I. Kennwerte definieren: Da die Formanten durch die Werte Frequenz und Bandbreite beschrieben werden, wurde zunächst nach einer Kategorie gesucht, die beide Werte für alle Formanten *eines* Frames gemittelt wiedergibt. Der hier generierte Kennwert wird im Folgenden als **freqband** bezeichnet und beschreibt den Wert eines jeden Frames bestehend aus gemittelten Frequenz und Bandbreiten Werten. Der Frequenzwert ist für die Verortung eines Formanten im Spektrogramm relevant und der Bandbreitenwert für das Intensitätsmaximum eines Formanten, bzw. somit dessen Charakterisierung (s.o.). Deshalb wurde jeder Frequenzwert eines Formanten mit dem Bandbreitenwert desselbigen multipliziert und die Ergebnisse aller Formanten pro Frame zusammen

addiert. Dieser Wert wurde dann durch den gesamten Bandbreitenwert eines Frames dividiert.

$$freqband = \frac{((f1 \cdot b1) + (f2 \cdot b2) + (f3 \cdot b3) + (f4 \cdot b4) + (f5 \cdot b5))}{(bges)}$$

Abbildung 5: Gleichung für freqband

Somit entsteht der Kennwert **freqband**, der pro frame Informationen über einen gemittelten Wert der Formanten in Hinblick auf Frequenz und Bandbreite enthält.

Ein weiterer Wert, der für die spätere Betrachtung über die Frames hinweg interessant sein könnte, ist die **Intensity**. Diese ist als Attribut in jedem Frame genannt und unterscheidet sich zu dem Bandbreitenwert, der sich auch auf Lautstärkemaxima bezieht dadurch, dass nicht ein Formant, sondern das Lautstärkemaximum eines Frames angegeben wird. Dieser könnte zur Differenzfindung zwischen Vokalen und Konsonanten hilfreich sein, da durch Vokale meist „die Lautstärke der Sprache erzeugt [wird und] über die Konsonanten die Informationen übermittelt [werden]“ (Ecophon 2018). Da **Intensity** nicht gemittelt oder anderweitig verrechnet wird, kann dieser direkt extrahiert werden.

Ähnlich verhält es sich auch mit der dritten Kategorie der Kennwerte. Hier werden die Frequenz- und Bandbreitenwerte eines jeden Formanten im Frame einzeln als Kennwerte genommen, um so später bspw. die Entwicklung des ersten Formanten von einem Frame zum nächsten untersuchen zu können, sowohl in Hinblick auf Frequenz als auch Bandbreite. Auch diese Werte müssen nicht gewichtet werden, sondern können in ihrem Rohzustand extrahiert werden. Diese Kennwerte bekommen folgende Label: handelt es sich um die Frequenz des ersten Formanten wird das Label **F1F** gewählt; ebenso verhält es sich mit den Formanten 2-5 (F2F etc.). Wird der Bandbreitenwert genutzt, wird dieser unter dem Label **F1B** für den ersten Formanten gefasst; ebenso verhält es sich wieder für die Formanten 2-5 (F2B etc.).

Wie bereits angedeutet, sind die eingelesenen Daten des ursprünglichen Dokuments durch Arraylisten und Arrays organisiert (Objekt <daten_aus> der Klasse Datensatz). Dies zieht sich ebenso durch die eigentliche Verarbeitung. Das Objekt <daten_aus> wird durchlaufen und die Kennwerte wie zuvor beschrieben ermittelt und in das lokale Array **kennwerte** eingetragen (Tabelle 5).

I.Schritt: Kennwerte definieren								Tab. 5
Array I : kennwerte (lokal)								
Kennwerte	FB	I	F1F	F2F	(...)	F1B	F2B	(...)
Frame aktuell	KW	KW	KW	KW	KW	KW	KW	KW



Array II: werte_past (aus kennwerte) (lokal)

Kennwerte	FB	I	F1F	F2F	(...)	F1B	F2B	(...)
Frame past	KW	KW	KW	KW	KW	KW	KW	KW

Zeitgleich wird das Array **werte_past** erstellt, welches die Kennwerte aus dem aktuellen Frame erhält, sobald sie in diesem nicht mehr benötigt werden. Das Array **werte_past** ist für den folgenden Teil der Analyse von Interesse.

II. Kennwerte über Frames analysieren: Um Trennstellen zu identifizieren, sollen die Frames durchlaufen werden und wenn sich zwischen zwei Frames signifikante Differenzen finden lassen, wird diese Stelle als Trennstelle gewertet. Dementsprechend werden die Kennwerte als Ausgangspunkte eines jeden Frames gewertet, die von Frame zu Frame untersucht werden sollen. Wie in Kapitel 2 erläutert, sollen diese Differenzen auf wechselnde Lautklassen hinweisen, die unterschiedliche Formantenstrukturen haben. Daraus lässt sich schließen, dass die Kennwerte in Hinblick auf die Unterschiede zu den Kennwerten des vorherigen Frames untersucht werden sollen. Dafür wird das Array **werte_diff** instanziiert, welches aus der Subtraktion des vorherigen Arrays **kennwerte** vom Array **werte_past** entsteht (s. Tab 6).

II.a Schritt: Differenzen berechnen**Tab. 6****Array III: werte_diff = werte_past – kennwerte (lokal)**

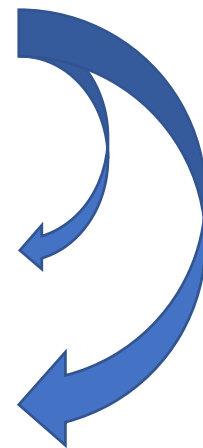
	FB	I	F1F	F2F	(...)	F1B	F2B	(...)
frame (n)	diff	diff	diff	diff	diff	diff	diff	diff

Array IV: daten_max (fest) (aus allen werte_diff)

	FB	I	F1F	F2F	(...)	F1B	F2B	(...)
Frame (n) im Schnitt an daten_max	rel. max	rel. max	rel. max	rel. max	rel. max	rel. max	rel. max	rel. max

Arraylist V: liste_diff = werte_diff

	FB	I	F1F	F2F	(...)	F1B	F2B	(...)
frame0	diff	diff	diff	diff	diff	diff	diff	diff
frame1	diff	diff	diff	diff	diff	diff	diff	diff
frame2	diff	diff	diff	diff	diff	diff	diff	diff
frame(...)	(...)	(...)	(...)	(...)	(...)	(...)	(...)	(...)



Aus dem Array **werte_diff** werden anschließend zwei weitere generiert. Einmal wird das Array **daten_max** instanziiert, welches des maximalen Wert pro Kennwert über alle Frames hinweg enthält (Abgleich mit jedem aktualisiertem Frame aus **werte_diff**). Also den höchsten freqband Wert in allen Frames oder den höchsten Intensity Wert über alle Frames hinweg. Dieses Array wird für den nächsten Schritt benötigt. Zunächst werden aber alle lokalen Einträge aus **werte_diff** in die Arraylist **liste_diff** geschrieben. Somit

verfügt das SEG_TOOL nun über eine Liste, die die Differenzen zwischen den einzelnen Frames in gewissen Kennwerten ausdrückt.

Im nächsten Schritt (s. Tab. 7) werden die bisherigen absoluten Differenzwerte in relative Werte verrechnet, um sie gegen gegebene Konstantwerte aufrechnen zu können. Dafür wird zunächst das Array **erg_konstante** instanziiert. In diesem werden die Werte der Liste **liste_diff** durch die maximalen Werte von **daten_max** dividiert. Dadurch erhält man die Differenzwerte in Relation zu ihrem maximalen Wert pro Kennwert. Dann werden abschließend im Array **konstante** pro Kennwert Konstanten manuell festgelegt, anhand derer Werte die relativen Differenzwerte bewertet werden. Wird bspw. für die Werte des Kennwertes **freqband** eine Konstante von 0.05 (5%) angewandt, gelten alle Werte die größer als anteilig 5% des maximalen Wertes (daten_max) liegen als vorübergehende Trennstellen. Genauer gesagt, wird der maximale Wert als 100% angesehen; die untersten 5% davon, sind die Grenze, die ein relativer Differenzwert überschreiten muss, um als Trennstelle zu gelten. Das bedeutet auch, dass an dieser Stelle Bedingungen formuliert werden können, die zur Auswahl von Trennstellen führen. Sind die Konstanten festgelegt, können auch die einzelnen Kennwertgruppen untereinander in Relation zueinander gestellt werden. Ein Beispiel ist:

Konstanten: Konstante[freqband] = 0.05; Konstante[F2F] = 0.10;

Bedingung: IF((rel.Diff[freqband] > konstante[freqband])&& (rel. Diff.[F2F] > konstante[F2F]))

Die Bedingungen für eine Trennstelle sind hier, dass der relative Differenzwert von **freqband** größer sein muss, als 5% des Anteils an dem maximalen Wert von **freqband**. Zusätzlich muss aber auch erfüllt sein (&&), dass der relative Differenzwert von **F2F** größer ist als 10% des Anteils an dem maximalen Wert von **F2F**. Zusätzlich kann noch ein Variable als Bedingung (grenz_treffer) gesetzt werden, die einen Treffer erst zulässt, wenn eine bestimmte Anzahl an Bedingungen erfüllt ist.

II.b Schritt: Konstanten festlegen & Bedingungen setzen

Tab. 7

Array VI: erg_konstante = liste_diff/daten_max

	FB	I	F1F	F2F	(...)	F1B	F2B	(...)
Frame 0	rel.diff	rel.diff	rel.diff	rel.diff	rel.diff	rel.diff	rel.diff	rel.diff

Array VII: konstante

	FB	I	F1F	F2F	(...)	F1B	F2B	(...)
Konstante	?	?	?	?	?	?	?	?

erg_konstante abhängig von konstante = **Trennstellen**

Trennstellen werden in ArrayList VIII: loesungs_aus geschrieben

	Position	Kennwerten [erg_konstante]																
Stelle [0]	3	<div>Array Vt: <u>erg_konstante</u> = liste_diff/daten_max</div> <table><tr><td>FB</td><td>l</td><td>F1F</td><td>F2F</td><td>(...)</td><td>F1B</td><td>F2B</td><td>(...)</td></tr><tr><td>Frame 0</td><td>rel.diff</td><td>rel.diff</td><td>rel.diff</td><td>rel.diff</td><td>rel.diff</td><td>rel.diff</td><td>rel.diff</td></tr></table>	FB	l	F1F	F2F	(...)	F1B	F2B	(...)	Frame 0	rel.diff	rel.diff	rel.diff	rel.diff	rel.diff	rel.diff	rel.diff
FB	l	F1F	F2F	(...)	F1B	F2B	(...)											
Frame 0	rel.diff	rel.diff	rel.diff	rel.diff	rel.diff	rel.diff	rel.diff											
Stelle [1]	5	<div>Array Vt: <u>erg_konstante</u> = liste_diff/daten_max</div> <table><tr><td>FB</td><td>l</td><td>F1F</td><td>F2F</td><td>(...)</td><td>F1B</td><td>F2B</td><td>(...)</td></tr><tr><td>Frame 0</td><td>rel.diff</td><td>rel.diff</td><td>rel.diff</td><td>rel.diff</td><td>rel.diff</td><td>rel.diff</td><td>rel.diff</td></tr></table>	FB	l	F1F	F2F	(...)	F1B	F2B	(...)	Frame 0	rel.diff	rel.diff	rel.diff	rel.diff	rel.diff	rel.diff	rel.diff
FB	l	F1F	F2F	(...)	F1B	F2B	(...)											
Frame 0	rel.diff	rel.diff	rel.diff	rel.diff	rel.diff	rel.diff	rel.diff											

Die gefundenen Trennstellen werden abschließend in die Arraylist **loesungs_aus** geschrieben, die sich einmal die Position(framenummer) des Treffers merkt und die Kennwerte, die in relativen Differenzwerte aus **erg_konstante** weitergibt.

III. Treffer eingrenzen: Da nach dem vorherigen Schritt häufig zu viele Trennstellen gefunden werden und es nach Augenmerk so erscheint, als ob viele Trennstellen sich mit anderen „gruppieren“ oder Intervalle bilden, bietet es sich an ein Streuungsmaß mit einzubeziehen. Dabei wird das Objekt <daten_aus> der Klasse Datensatz, welches alle Frames ohne vorherige Verarbeitung enthält, noch einmal durchgegangen. Stößt man nun auf ein Frame, welches im gleichen Frame in **loesungs_aus** als Trennstelle markiert ist, wird dieses Frame als der Start eines Intervalles markiert. Erst wenn das Programm in **daten_aus** auf ein Frame trifft, welches keine aufeinanderfolgende Trennstelle ist in **loesungs_aus**, wird das vorherige Frame als das Ende eines Intervalls markiert. Auch wenn zwischen mehreren vorkommenden Trennstellen ein Frame ohne Trennstelle liegen sollte, wird dieses „leere“ Frame mit ins Intervall einbezogen, d.h. die Grenze für Intervalle liegt bei 0-1.

Wenn ein Intervall mit Start und Endpunkt gefunden wurde, wird dieses direkt analysiert. Dafür muss eine Bedingung festgelegt werden, nach der das Frame als Trennstelle bestimmt wird, welches am ehesten einen wirklichen Treffer darstellt. Nach vorheriger Betrachtung der vorläufigen Trennstellen aus Analyseteil II., erscheint es, dass innerhalb eines Intervalls oft der Frame eine Trennstelle ist, dessen **frequband**-Wert der kleinste ist, im Vergleich zu denen der anderen Frames in demselben Intervall. Diese Bedingung kann allerdings beliebig angepasst werden. Diese überarbeiteten und endgültigen Trennstellen werden dann wieder in das Objekt **loesungs_aus** geschrieben und zur Ausgabe weitergegeben.

Abschließend stellt die folgende Grafik noch einmal kurz da, wie die Analysestruktur aufgebaut ist:

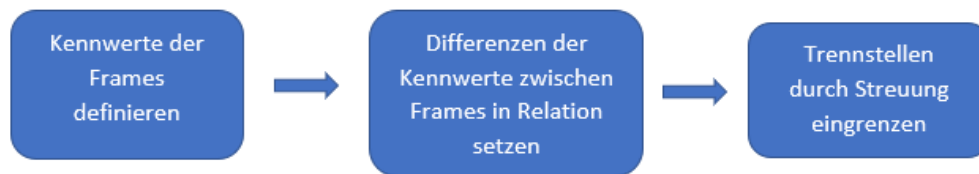


Abbildung 6: Analyseverfahren in drei Schritten

Der Vorteil daran ist, dass jeder Schritt beliebig erweitert und angepasst werden kann. Im ersten Schritt können beispielsweise die Anzahl an Kennwerten vergrößert oder verkleinert werden. Im zweiten Schritt können die Konstanten an denen die relativen Differenzwerte gemessen werden, manuell festgelegt werden und die Bedingungen, welche dieser rel. Differenzwerte (+Konstanten) als vorläufige Trennstelle zu gelten hat, können je nach Wunsch angepasst werden. Auch die Streuung zur Eingrenzung der vorläufigen Trennstellen kann abgeändert werden.

5.2.3 Ausgabe der Daten

Abschließend gibt es zwei Ausgabedokumente: Das erste ist eine TextGrid Datei, in der die Trennstellen mit ihren Zeitpunkten an denen sie starten und enden eingetragen werden. Dieses Dokument dient letztendlich zum visuellen Abgleich mit der Audiodatei bei PRAAT bspw. Die zweite Ausgabe ist eine CSV-Datei. Diese besteht zu allererst aus einer Zeile in der pro spalte die Nummer eines Frames in steigender Abfolge eingetragen wird. Sollte es sich bei einem Frame um eine Trennstelle handeln, wird in der Zeile darunter in der Spalte des jeweiligen Frames eine „1“ eingetragen und ansonsten eine „0“. Zusätzlich werden die Kennwerte aus **erg_konstante** unter die Trennstellentreffer gesetzt. Diese Form der Darstellung ermöglicht einen Abgleich mit den Transkriptionen des BAS-Onlinetools bei Excel.

6. Anwendung und Ergebnisse

Nachdem nun der Aufbau und die Funktion des SEG_TOOLS erläutert wurde, soll nun auch ein Anwendungsbeispiel gegeben werden. Für die Einstellung der Bedingungen wurde die Audiodatei mit dem Label S07_1_10_NF verwendet und die Ergebnisse wurden mit den Ergebnissen der BAS Transkription abgeglichen. Die daraus generierte Formantenanalyse wurde unter folgenden drei Szenarien getestet:

Im ersten Versuch wurden die Kennwerte **freqband** und **intensity** gewählt, weil in beiden Werte definiert sind, die gemittelt ein ganzen Frame definieren. Es wird davon ausgegangen, dass hier Trennstellen gefunden werden können, da alle Charakteristika eines Formanten miteinbezogen werden. Dementsprechend wurde die Konstante für

freqband auf 0.05 gesetzt und die Konstante für **intensity** auf 0.10. Die Bedingung lautet wie folgt:

```
erg_k[INDEX_FREQBAND] > konst[INDEX_FREQBAND] || erg_k[INDEX_INTENSITY] > konst[INDEX_INTENSITY]
```

Da es sich nur um zwei Bedingungen handelt, bleibt der Grenztreffer auf 1 gesetzt. Daraufhin wurden bei einer Anzahl von 561 Frames insgesamt, 386 Trefferstellen identifiziert (s. Tab. 8). In diesen sind nach einem Vergleich mit der BAS Transkription auch alle 100% der BAS 50 Treffer enthalten, was allerdings auch der Masse an Trefferstellen geschuldet ist, woraus sich für das SEG_TOOL nach diesem ersten Rechenversuch lediglich 13% oder 386 Trefferstellen als echte Treffer identifizieren lassen.

Im zweiten Versuch wurden die Kennwerte **F1B** und **F2B** analysiert. Diese erscheinen vielversprechend, weil in diesen Frequenzbereichen auch die Stimmhaftigkeit von Lauten ausgedrückt wird. Dementsprechend könnten die Analyse dieser Kennwerte zur Trennstellenfindung durch Stimmhaftigkeit und Stimmlosigkeit führen. Hier wurden die beiden Konstanten jeweils auf 0.01 gesetzt und folgende Bedingungen definiert:

```
erg_k[FORMANT1_BAND] < konst[FORMANT1_BAND] || erg_k[FORMANT2_BAND] < konst[FORMANT2_BAND]
```

Auch hier wurde der Grenztrefferwert nicht erhöht. Daraufhin wurden 299 Trennstellen gefunden, in denen 90% der BAS Treffer enthalten sind. Das macht auch hier lediglich einen Anteil von 15% an Treffern von 299 im SEG_TOOL aus.

Im dritten Versuch wurden die übrigen Kennwerte aufgegriffen. Hier sollte untersucht werden, wie sich die Frequenz des ersten Formanten von der Frequenz desselben Formanten des vorherigen Frames verhält [F1F – F1F], in Abhängigkeit von allen anderen Formanten zu den jeweiligen Formanten der vorhergegangenen Frames. Das steht dann wiederum in Abhängigkeit zu demselben Verfahren mit dem Bandbreitenwert:

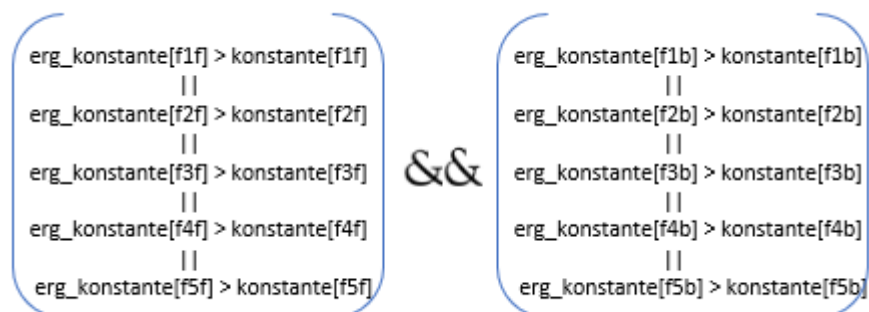


Abbildung 7: Bedingung für Rechenweg 3 (f1-5 über freq und band)

Bei den gemittelten Kennwerten könnten Feinheiten rausgerechnet sein, deshalb wird hier versucht jeden Kennwert in seiner Feinheit zu verrechnen. Die Variable **grenz_treffer** wird hier auf 4 erhöht; womit eine Trennstelle erfüllt ist, wenn mehr als 4

oder 4 Bedingungen zutreffen. Die Konstanten liegen bei allen Kennwerten bei 0.10. Hierbei werden 188 Trennstellen gefunden, worin 82% der Trennstellen aus BAS noch enthalten sind; woraus sich ergibt, dass 22% der 188 vorläufigen Trennstellen des SEG_Tools als richtige Trennstellen zu werten sind (vgl. Tab. 8).

Tabelle: Auswertung vor/nach Streuung			Beispiel einer Audio: S_07_1_10_NF		
Nr.	Rechnung:	ohne Streuung		mit Streuung	
		Tool:	Tool	Nach Streuung (<freqband)	
		abs. Tr.	SEG Tr.	abs. Tr.	SEG Tr.
1.	Freqband (0.05) Int(0.10) (grenztrefter 1)	BAS	50 100%	BAS	50 36%
		SEG	386 13%	SEG	36 50%
2.	f1b f2b beide 0.01 (grenztrefter bleibt 1)	BAS	50 90%	BAS	50 50%
		SEG	299 15%	SEG	57 44%
3.	f1-5 (über freq () && über band() konst 0.10 grenztrefter 4	BAS	50 82%	BAS	50 58%
		SEG	188 22%	SEG	55 53%

Tabelle 8: Auswertungen vor/nach Streuung

Da hier bei allen drei Analyseversuchen eine Vielzahl an vorläufigen Trennstellen gefunden wurde, worin immernoch viele richtige Trennstellen (vgl. BAS) enthalten sind, bietet es sich an die Streuung der Treffer zu analysieren. Hierbei wurden die Bedingungen wieder wie zuvor erläutert jeweils den drei Rechnungen angepasst. Da, nach Betrachten der Kennwerte der vorläufigen Treffer, es den Anschein macht, dass der kleinste **freqband** Wert innerhalb eines Intervalls an Trennstellen die richtige Trennstelle häufig markiert, wurde dies als Bedingung für den Streuungsfaktor gesetzt. Daraus ergeben sich wie in Tabelle 8 ersichtlich folgende Ergebnisse: Bei der ersten Rechnung (freqband 0.05 || int 0.10) werden anschließend nur noch 36 Trennstellen gefunden, worin nur noch 36% der Treffer in BAS vorkommen. Dies ergibt allerdings durch die Verringerung der Trennstellen allgemein eine Trefferquote von 50% für das SEG_TOOL. Besser fallen die Ergebnisse für die zweite Rechnung aus (f1b 0.01 || f2b 0.01). Hier werden 57 Trennstellen gefunden, worin noch 50% (25) der BAS Treffer enthalten sind. Daraus ergibt sich für das SEG_TOOL eine Trefferquote von 44% von 57 gesetzten Trennstellen. Am besten fallen die Ergebnisse bei der dritten Rechnung aus (f1-5(über freq&band 0.10). Hier werden 55 Trennstellen identifiziert, von denen 58% der Trennstellen mit denen des BAS Tools übereinstimmen. Daraus ergibt sich für das SEG_TOOL eine Trefferquote von 53% für die 55 gesetzten Trennstellen.

Da sich aus dem dritten Rechenversuch die besten Ergebnisse extrahieren lassen, ist dieser als Default im Programm selbst eingestellt (Variation jederzeit möglich). Diese Analyse wurde dann mit 10 weiteren Audiodateien vom selben Sprecher ausgetestet und

jeweils mit den dafür zum Abgleich generierten Transkriptionen des BAS Tools verglichen (s. Tabelle 9).

Audio	BAS Treffer %	SEG Treffer %
S_07_1_1_NF	45	55
S_07_1_2_NF	57	50
S_07_1_2_PF	52	52
S_07_1_3_BF	52	62
S_07_1_3_NF	33	47
S_07_1_4_PF	43	51
S_07_1_5_NF	30	33
S_07_1_6_BF	43	48
S_07_1_6_NF	46	45
S_07_1_7_NF	47	57
S_07_1_10_NF	60	55
GESAMT:	50	55

Tabelle 9: Finale Auswertung der Ergebnisse (11 Audios)

Daraus ergibt sich letztlich, dass bei allen Treffern die gefunden werden im Schnitt 50% der im BAS Abgleich markierten Treffer vorkommen und in Relation zu den jeweils gefundenen Trennstellen ergibt sich für das SEG_TOOL eine Trefferquote von 55%.

Diskussion: Das Ergebnis kann zwiespältig interpretiert werden. In Hinblick auf die Komplexität der Segmentation gesprochener Sprache und der Schwierigkeit Laute voneinander zu isolieren, ist eine Trefferquote von 55% als relativ gut zu bewerten.

Bei der Audiodatei S07_1_10_NF sind 53% der 55 gesetzten Treffer als richtige Treffer identifiziert. Das bedeutet aber nicht, dass die 26 „falschen“ Treffer wirklich komplett falsch gesetzt wurden. Denn beim Abgleich mit der BAS Transkription wurden die Stellen als Treffer gewertet, die entweder mit der des BAS Ergebnisses genau im selben Frame übereinstimmten oder 1-2 Frames daneben gesetzt waren. Da ein Frame nur 0.00625s lang ist, ist diese Variation durchaus möglich. Trotzdem könnten auch 3-4 Frames noch als Toleranzgrenze gesetzt werden und die teilweise 26 „falschen“ Treffer bspw. noch als richtig gewertet werden.

Weiter stellt sich die Frage, ob die BAS Transkriptionen wirklich am besten geeignet sind für den Abgleich mit den Segmentgrenzen des SEG_TOOLS. Wie zu Beginn erläutert, deckt sich die BAS Transkription mit der manuell erstellten lediglich zu 72% (s. Tab. 1). Der Vergleich der manuell erstellten Transkription zu der Datei S07_1_10_NF zeigt, dass im SEG_TOOL 55 Treffer gefunden wurden, die 60% der Manuell gesetzten Treffer abdecken. Damit erhält das SEG_TOOL im Vergleich zur manuelle Transkription eine Trefferquote von 55%. Bei derselben Audiodatei ergab sich im Vergleich zu BAS eine Trefferquote des SEG von 53% (Trefferabdeckung mit

BAS 58%). Insofern gibt es hier eine minimale Verbesserung von 53% → 55%. des SEG_TOOLS durch die Wahl einer anderen Vergleichstranskription.

Tool			
	abs. Tr.	SEGTr.	
Manuell	50	60%	
SEG	55	55%	

Tabelle 10: Vergleich SEG & Manuell

Durch den variablen und dynamischen Aufbau des Tools, können die Bedingungen, Konstanten und Kennwerte weiter verfeinert werden, um die Ergebnisse zu verbessern. Die drei bisherigen Rechenwege könnten in weiteren Analysen gegebenenfalls ausgebaut oder entfernt werden. Gleiches gilt auch für das Streuungsmaß; zwar reduziert sich die Trefferanzahl dadurch fast immer auf ungefähr die gewünschte Anzahl an Trennstellen, trotzdem werden aber auch häufig 20-30% an richtigen Trennstellen rausgelöscht. Der Streuungsfaktor ist eventuell zu „eng“ eingestellt und Bedarf einiger Anpassungen.

Gegebenenfalls würde sich auch eine Herangehensweise über die Charakteristika der Lautklassen anbieten. Wie in Abb. 9 erkennbar werden einzelne Lautklassen wie hier der Frikativ /s/ als solcher erkannt. Es könnte analysiert werden, welche Lautklassen bisher gute Trennstellenerkennung haben und welche eher weniger. Aus dieser Information könnten wiederum neue Bedingungen für den Analyseteil entwickelt werden.

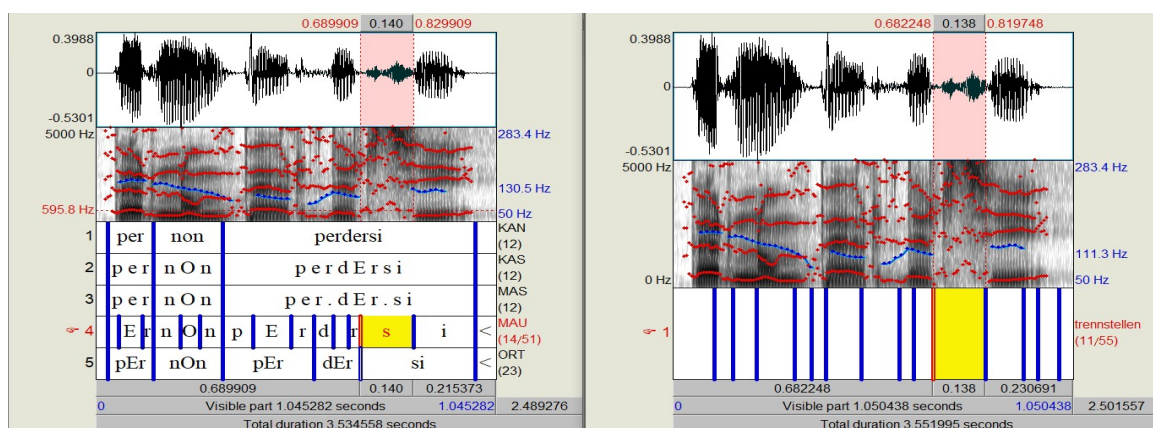


Abbildung 8 (Vergleich der Segmentationen von /s/: links BAS & rechts SEG)

7. Fazit und Ausblick

Abschließend lässt sich sagen, dass wie in den Hypothesen formuliert, Trennstellen durch die Analyse von Differenzen zwischen Formanten über Frames hinweg generiert werden konnten. Das deutet auf eine systematische Datenbasis hin, die relevante Daten

mit wiederkehrenden Mustern beinhaltet. Nach einigen Analyseversuchen und Anpassungen derer, ließ sich ein Ergebnis von 55% Trefferquote des SEG_TOOLS erzielen. Weitergehend könnte der Datensatz von bisher 11 Audiodateien erweitert werden und die Analyseverfahren variiert und angepasst werden. Bessere Ergebnisse lassen sich eventuell auch durch andere Herangehensweisen wie eine Lautklassenidentifizierung erzielen oder die Integration eines Deep-Learning/Machine Learning - Ansatzes. Zuletzt kann allerdings gesagt werden, dass die Segmentation von gesprochener Sprache schwierig ist, da Sprache sich nicht in genaue Einheiten isolieren lässt. Gerade bei der Analyse von Sprachdaten auf Phonebene ist Genauigkeit gefragt, weshalb ein Segmentationstool auf dieser Ebene sinnvoll ist, welches ggf. noch genauer funktioniert als das BAS Webtool. Das hier geschaffene SEG_TOOL kommt an diese Genauigkeit noch nicht ran. Trotzdem sollten sich durch weitere Versuche bessere Ergebnisse erzielen lassen.

Literaturverzeichnis

II

Sekundärliteratur

FRY, D. B.; Anderson, S. R.; Bresnan, J.; Comrie, B.; Dressler, W.; Ewen, C. J. (1979): The Physics of Speech: Cambridge University Press.

HARDCASTLE, William J.; Hewlett, Nigel (2006): Coarticulation. Theory, data and techniques. Cambridge, New York: Cambridge University Press.

KISLER, Thomas; Reichel Uwe D.; Schiel, Florian (2017): Multilingual processing of speech via web services, Computer Speech & Language, Volume 45, September 2017, pages 326–347.

Internetquellen

BOERSMA, Paul; Weenink, David (2018): Praat: doing phonetics by computer [Computer program]. Version 6.0.37, retrieved 14 March 2018 from <http://www.praat.org/>, (Stand:19.09.2019).

BECKER, Thomas (2012): Einführung in die Phonetik und Phonologie des Deutschen.

Darmstadt: WBG Wiss. Buchges (Einführung Germanistik). Unter:
http://content-select.com/index.php?id=bib_view&ean=9783534727636 (Stand
 19.09.2019).

ECOPHON (2018): Sprache und Hören. Unter: <https://www.ecophon.com/de/know-how/acoustic-knowledge/basic-acoustics/sprache-und-horen/> (Stand
 22.09.2019).

MACHELETT, Kirsten (1996): Das Lesen von Sonagrammen V1.0 - Kapitel II. Die
 Lautklassen im Sonagramm: Bestimmung des Artikulationsmodus. Hg. v.
 Institut für Phonetik und Sprachliche Kommunikation, Universität München.
 Unter:
<https://www.phonetik.uni-muenchen.de/studium/skripten/SGL/SGLKap2.html>
 (Stand: 19.09.2019).

REICHEL, Uwe (2007): Lesen von Sonagrammen I: Grundlagen. Hg. v. L.
 MünchenM.U. IPS. Unter:
https://www.phonetik.uni-muenchen.de/~reichelu/kurse/sonagramme/sona_slide_s_1.pdf (Stand: 19.09.2019).

WEENINK, David (2019): Die Praatpfanne. Universität Amsterdam. Unter:
<http://praatpfanne.lingphon.net/> (Stand: 19.09.2019).

Abbildungs- und Tabellenverzeichnis

III

Tab.	A:	Realisation	der	Modalverben	
<i>potere/dovere/volere</i>					6
Tab.1.b:					8
Abb.1.b:					8
Tab. 2:					9
Abb.2:					9
Tab. 3:					9

Abb.3:	10
--------------	----

1. Tabellen

Tabelle 1: Vergleich der Transkriptionen BAS und ANNE:

Auswertung BAS+ANNE					
BAS Treffer	50				
Anne Treffer	50				
		Anteil an BAS			Anteil an An
Gem. Treffer	6	Anteil Gem.	12		12
Treffer +1	20	Treffer +1+G	52		52
Treffer +2	10	Treffer +2+1-	72		72

Tabelle 2: Transkriptionen der Audios (orthographisch)

S_07_1_1_N F	Per amministrare la città bisogna votare il partito alle elezioni
S_07_1_2_N F	Quando si frigge bisogna proteggere il vestito dagli schizzi
S_07_1_2_P F	Quando si frigge bisogna proteggere il vestito dagli schizzi
S_07_1_3_B F	Per suonare nell'orchestra bisogna imparare lo spartito perfettamente
S_07_1_3_N F	Per suonare nell'orchestra bisogna imparare lo spartito perfettamente
S_07_1_4_P F	Durante l'inverno bisogna scaldare il locale per i clienti
S_07_1_5_N F	Per avere una bocca sana bisogna disinfettare le gengive con il collutorio
S_07_1_6_B F	Per sedersi comodamente bisogna mettere il cuscino sullo sgabello
S_07_1_6_N F	Per sedersi comodamente bisogna mettere il cuscino sullo sgabello
S_07_1_7_N F	Quando si arreda la casa bisogna scegliere il divano per il soggiorno

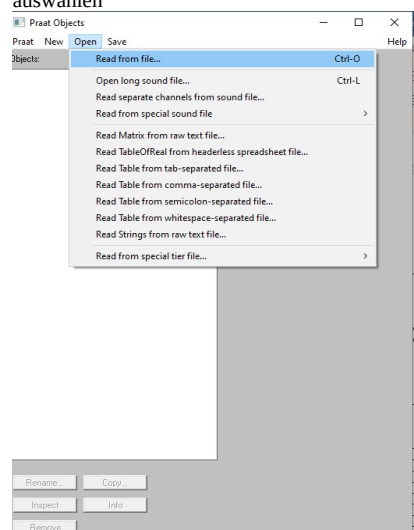
Tabelle 3: Klassenstruktur

Klasse	Name	Datentyp	Beschreibung
Datensatz	xmin	double	Startpunkt der Audiodatei in (s)
	xmax	double	Endpunkt der Audiodatei in (s)
	nx	int	Anzahl der Frames gesamt
	dx	double	Dauer eines Frames in (s)
	x1	double	Startpunkt des ersten Frames in (s)
	Frame	ArrayList<Frame>	Arrayliste mit Platzhaltern für Objekte der Klasse Frame
Frame	intensity	double	Maximaler Lautstärkenwert eines Frames
	formants	ArrayList<Formant>	Arrayliste mit Platzhaltern für Objekte der Klasse Formant
Formant	frequency	double	Frequenzwert eines Formanten
	bandwidth	double	Bandbreitenwert eines Formanten
Loesungssatz	trennstellen	ArrayList<Trennstellen>	Arrayliste mit Platzhaltern für Objekte der Klasse Trennstellen

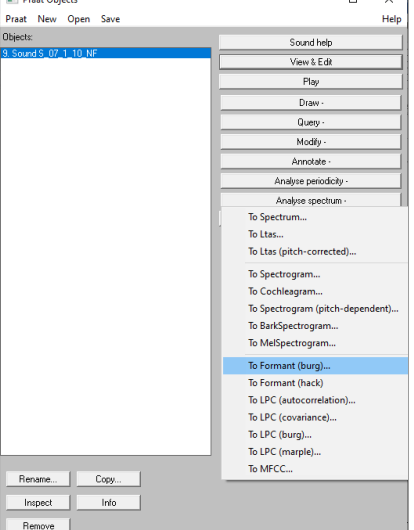
Trennstellen	position	int	Angabe der Framenummer einer Trennstelle
	kennwerte	double Array[]	Array für die Kennwerte der Analyse

Tabelle 4: PRAAT Anleitung - Formantenanalyse generieren

1. Praat Objects Fenster öffnen
2. Open -> Read from file... -> Audio auswählen



3. Analyse spectrum -> to Formant(burg) -> ok



4. Generiertes Objekt: Save -> Save as text file

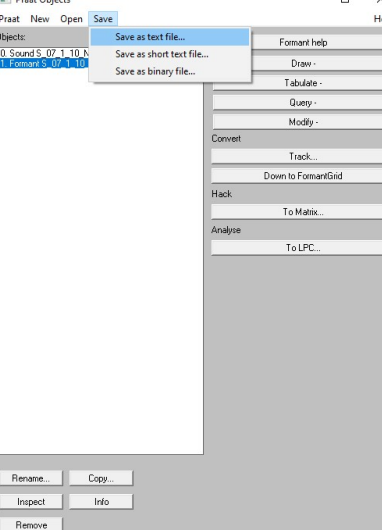


Tabelle 5: I. Schritt: Kennwerte definieren

I.Schritt: Kennwerte definieren									Tab. 5
Array I : kennwerte (lokal)									
Kennwerte	FB	I	F1F	F2F	(...)	F1B	F2B	(...)	
Frame aktuell	KW	KW	KW	KW	KW	KW	KW	KW	
Array II: werte_past (aus kennwerte) (lokal)									
Kennwerte	FB	I	F1F	F2F	(...)	F1B	F2B	(...)	
Frame_past	KW	KW	KW	KW	KW	KW	KW	KW	

Tabelle 6: II.a. Schritt: Differenzen berechnen

II.a Schritt: Differenzen berechnen									Tab. 6
Array III: werte_diff = werte_past – kennwerte (lokal)									
	FB	I	F1F	F2F	(...)	F1B	F2B	(...)	
frame (n)	diff	diff	diff	diff	diff	diff	diff	diff	
Array IV: daten_max (fest) (aus allen werte_diff)									
	FB	I	F1F	F2F	(...)	F1B	F2B	(...)	
Frame (n) im Schnitt an daten_max	rel. max	rel. max	rel. max	rel. max	rel. max	rel. max	rel. max	rel. max	
Arraylist V: liste_diff = werte_diff									
	FB	I	F1F	F2F	(...)	F1B	F2B	(...)	

frame0	diff	diff	diff	diff	diff	diff	diff	diff
frame1	diff	diff	diff	diff	diff	diff	diff	diff
frame2	diff	diff	diff	diff	diff	diff	diff	diff
frame(...)	(...)	(...)	(...)	(...)	(...)	(...)	(...)	(...)

Tabelle 7: II.b. Schritt: Konstanten festlegen & Bedingungen setzen

II.b Schritt: Konstanten festlegen & Bedingungen setzen

Tab. 7

Array VI: erg_konstante = liste_diff/daten_max

	FB	I	F1F	F2F	(...)	F1B	F2B	(...)
Frame 0	rel.diff	rel.diff	rel.diff	rel.diff	rel.diff	rel.diff	rel.diff	rel.diff

Array VII: konstante

	FB	I	F1F	F2F	(...)	F1B	F2B	(...)
Konstante	?	?	?	?	?	?	?	?

erg_konstante abhängig von konstante = Trennstellen

Trennstellen werden in ArrayList VIII: loesungs_aus geschrieben

	Position	Kennwerten [erg_konstante]																
Stelle [0]	3	<div>Array VI: erg_konstante = liste_diff/daten_max</div> <table> <tr> <td>FB</td> <td>I</td> <td>F1F</td> <td>F2F</td> <td>(...)</td> <td>F1B</td> <td>F2B</td> <td>(...)</td> </tr> <tr> <td>rel.diff</td> <td>rel.diff</td> <td>rel.diff</td> <td>rel.diff</td> <td>rel.diff</td> <td>rel.diff</td> <td>rel.diff</td> <td>rel.diff</td> </tr> </table>	FB	I	F1F	F2F	(...)	F1B	F2B	(...)	rel.diff	rel.diff	rel.diff	rel.diff	rel.diff	rel.diff	rel.diff	rel.diff
FB	I	F1F	F2F	(...)	F1B	F2B	(...)											
rel.diff	rel.diff	rel.diff	rel.diff	rel.diff	rel.diff	rel.diff	rel.diff											
Stelle [1]	5	<div>Array VI: erg_konstante = liste_diff/daten_max</div> <table> <tr> <td>FB</td> <td>I</td> <td>F1F</td> <td>F2F</td> <td>(...)</td> <td>F1B</td> <td>F2B</td> <td>(...)</td> </tr> <tr> <td>rel.diff</td> <td>rel.diff</td> <td>rel.diff</td> <td>rel.diff</td> <td>rel.diff</td> <td>rel.diff</td> <td>rel.diff</td> <td>rel.diff</td> </tr> </table>	FB	I	F1F	F2F	(...)	F1B	F2B	(...)	rel.diff	rel.diff	rel.diff	rel.diff	rel.diff	rel.diff	rel.diff	rel.diff
FB	I	F1F	F2F	(...)	F1B	F2B	(...)											
rel.diff	rel.diff	rel.diff	rel.diff	rel.diff	rel.diff	rel.diff	rel.diff											

Tabelle 8: Auswertungen vor/nach Streuung

Tabelle: Auswertung vor/nach Streuung				Beispiel einer Audio : S_07_1_10_NF			
		ohne Streuung		mit Streuung			
Nr.	Rechnung:	Tool:		Tool	Nach Streuung (<freqband)		
		abs. Tr.	SEG Tr.		abs. Tr.	SEG Tr.	
1.	Freqband (0.05) Int(0.10) (grenztrefter 1)	BAS	50	100%	BAS	50	36%
		SEG	386	13%	SEG	36	50%
2.	f1b f2b beide 0.01 (grenztrefter bleibt 1)	BAS	50	90%	BAS	50	50%
		SEG	299	15%	SEG	57	44%
3.	f1-5 (über freq ()) && über band()) konst 0.10 grenztrefter 4	BAS	50	82%	BAS	50	58%
		SEG	188	22%	SEG	55	53%

Tabelle 9: Finale Auswertung der Ergebnisse (11 Audios)

Audio	BAS Treffer %	SEG Treffer %
S_07_1_1_NF	45	55
S_07_1_2_NF	57	50
S_07_1_2_PF	52	52
S_07_1_3_BF	52	62
S_07_1_3_NF	33	47
S_07_1_4_PF	43	51
S_07_1_5_NF	30	33
S_07_1_6_BF	43	48
S_07_1_6_NF	46	45
S_07_1_7_NF	47	57
S_07_1_10_NF	60	55
GESAMT:	50	55

Tabelle 10: Vergleich SEG & Manuell

Tool	abs. Tr.	SEGTr.
Manuell	50	60%
SEG	55	55%

2. Abbildungen:

Abbildung 9: Transkription des BAS Tools

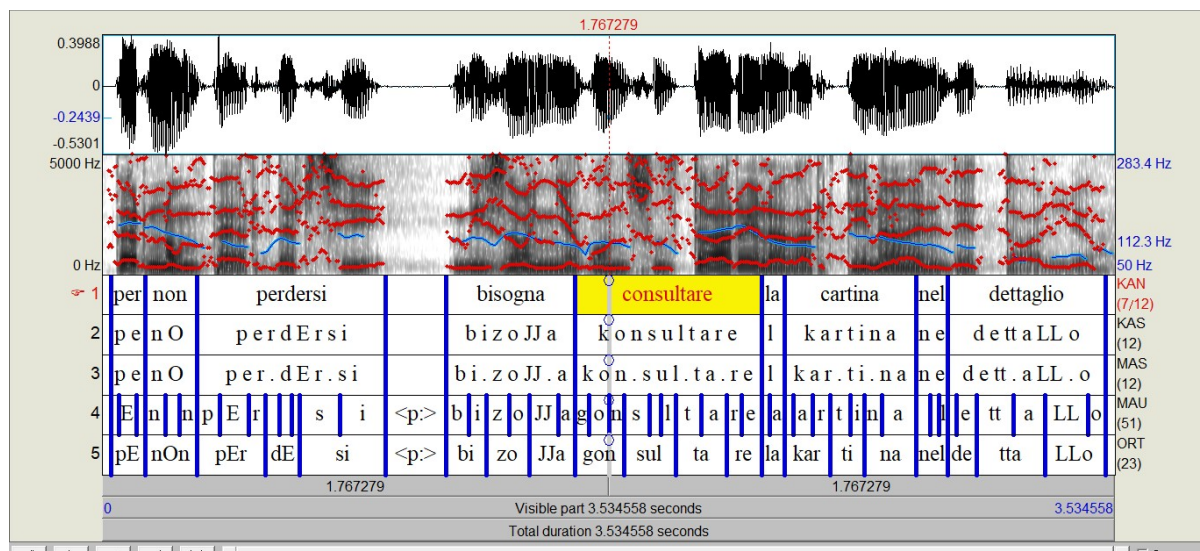


Abbildung 10: Segmentationsverfahren von Input zu Output

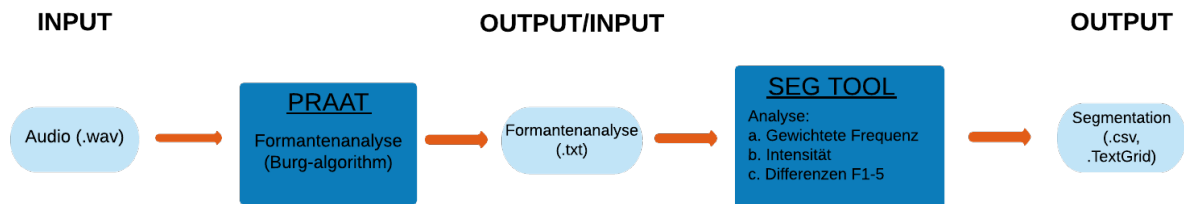


Abbildung 3: Vergleich BAS & SEG Output

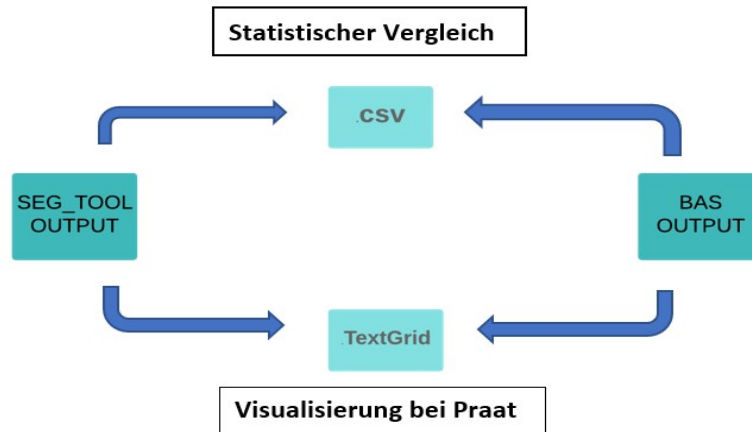


Abbildung4: Klassenstruktur am Beispiel der Formantenanalyse

```

File type = "ooTextFile"
Object class = "Formant 2"

xmin = 0
xmax = 3.5519954648526078
nx = 561
dx = 0.00625
x1 = 0.025997732426303655
maxnFormants = 5
frames []:
frames [1]:
  intensity = 0.0002425982350891871
  nFormants = 4
  formant []:
  formant [1]:
    frequency = 1122.358706088308
    bandwidth = 514.6020354961277
  formant [2]:
    frequency = 2156.9466437605347
    bandwidth = 409.86059572852895
  formant [3]:
    frequency = 3492.062289689977
    bandwidth = 546.73335363404
  formant [4]:
    frequency = 4327.470990382201
    bandwidth = 1025.386413961678
frames [2]:
  intensity = 0.0006190978366273994
  nFormants = 4
  formant []:
  formant [1]:
    frequency = 1242.902664527146
    bandwidth = 284.6191893969738
  ...
  
```

Klasse: Datensatz

Klasse: Frame

Klasse: Formant

Abbildung 5: Gleichung für freqband

$$freqband = \frac{((f1 \cdot b1) + (f2 \cdot b2) + (f3 \cdot b3) + (f4 \cdot b4) + (f5 \cdot b5))}{(bges)}$$

Abbildung 6: Analyseverfahren in drei Schritten

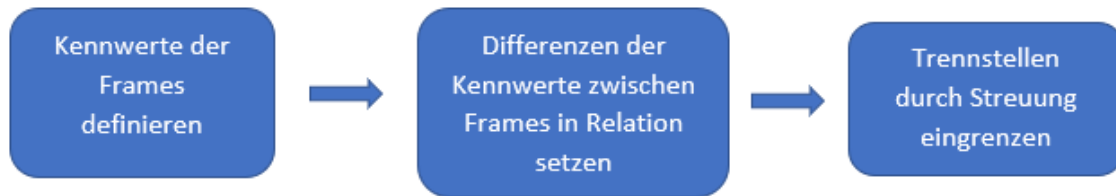
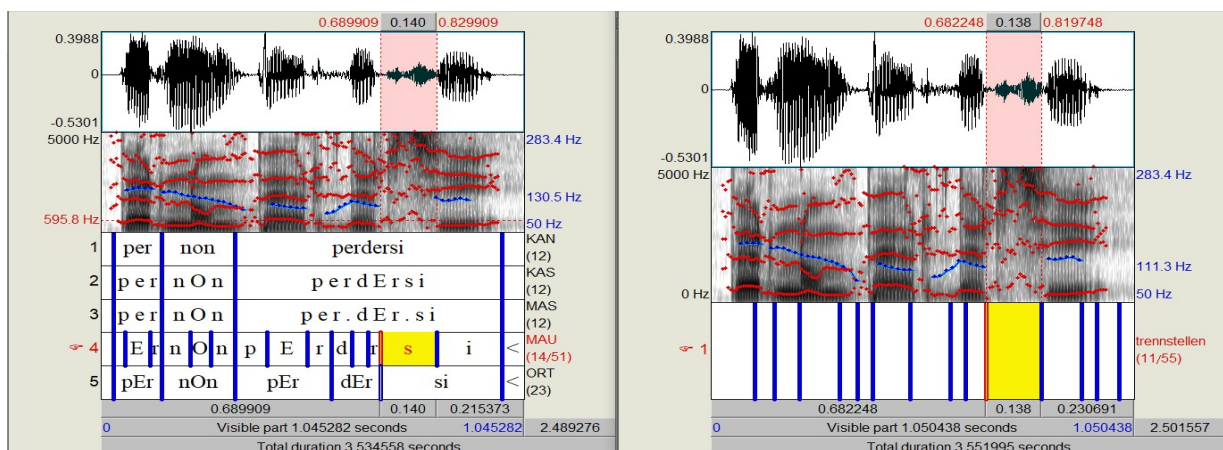


Abbildung 7: Bedingung für Rechenweg 3 (f1-5 über freq und band)

$$\begin{aligned}
 & \left(\begin{array}{l} \text{erg_konstante}[f1f] > \text{konstante}[f1f] \\ || \\ \text{erg_konstante}[f2f] > \text{konstante}[f2f] \\ || \\ \text{erg_konstante}[f3f] > \text{konstante}[f3f] \\ || \\ \text{erg_konstante}[f4f] > \text{konstante}[f4f] \\ || \\ \text{erg_konstante}[f5f] > \text{konstante}[f5f] \end{array} \right) \&\& \left(\begin{array}{l} \text{erg_konstante}[f1b] > \text{konstante}[f1b] \\ || \\ \text{erg_konstante}[f2b] > \text{konstante}[f2b] \\ || \\ \text{erg_konstante}[f3b] > \text{konstante}[f3b] \\ || \\ \text{erg_konstante}[f4b] > \text{konstante}[f4b] \\ || \\ \text{erg_konstante}[f5b] > \text{konstante}[f5b] \end{array} \right)
 \end{aligned}$$

Abbildung 11: Vergleich der Segmentationen von /s/: links BAS & rechts SEG



Formblatt

V

Universität zu Köln

Institut für DH

MA Informationsverarbeitung

Titel der Arbeit: SEG_TOOL - Ein automatisiertes Segmentationsverfahren zur
Verarbeitung gesprochener Sprache auf Phonebene durch die Analyse von Formanten


Name: Anne-Kathrin Gerlach

smail-Kennung: agerlac3

Matrikelnummer: 5986095
Abgabesemester: Sommersemester 2019

Kurstitel: Korpuslinguistik
Kurssemester: Wintersemester 2018
Leitung: Prof. Dr. Andreas Witt
Modul: Schwerpunktmodul 1 – Verarbeitung von Texten
Datum: 25.09.2019

Eigenständigkeitserklärung: Hiermit bestätige ich, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Hilfsmittel benutzt habe. Die Stellen der Arbeit, die dem Wortlaut oder dem Sinn nach anderen Werken (dazu zählen auch Internetquellen) entnommen sind, wurden unter Angabe der Quelle kenntlich gemacht.

25.09.2019 
Datum, Unterschrift