Karlsruhe Institute of Technology

Automatische Segmentierung und Gruppierung natürlicher Sprache anhand verschiedener Sprecher

Masterarbeit
von

Ludwig Linhuber

An der Fakultät für Informatik
Am Institut für Anthropomatik

Erstgutachter:	Prof. Dr. Alex Waibel
Zweitgutachter:	Dr. Sebastian Stüker
Betreuender Mitarbeiter:	Dipl.-Inform. Michael Heck

Bearbeitungszeit: 01. Juli 2013 – 31. Dezember 2013

Ich versichere wahrheitsgemäß, die Arbeit selbstständig angefertigt, alle benutzten Hilfsmittel vollständig und genau angegeben und alles kenntlich gemacht zu haben, was aus Arbeiten anderer unverändert oder mit Abänderungen entnommen wurde.

Karlsruhe, 31.12.2013

. . . . . .

(**Ludwig Linhuber**)

Abstract

In dieser Arbeit wird versucht den Ansatz aus [GLA98] zur Segmentierung und Gruppierung unbekannter Sprecher zu implementieren und durch die Verwendung verschiedener Distanzmaße und Mergefunktionen die Qualität der Ergebnisse zu verbessern. Zuerst werden die theoretischen Grundlagen der automatischen Spracherkennung erläutert. Danach wird der Ansatz aus [GLA98] genauer erklärt. Es werden verschiedene Experimente durchgeführt um die Auswirkung der verschiedenen Distanzmaße und Mergefunktionen auf die Qualität der Ergebnisse zu zeigen. Das Ziel ist es, ein bereits bestehendes Altsystem zur Segmentierung und Gruppierung von Sprache zu ersetzen und dessen Leistung mindestens zu erreichen.
Die Experimente zeigen, dass die Verwendung von adaptierten Universal Background Models die Qualität des Ergebnisses stark erhöht. Außerdem kann die Leistung des Baseline-Systems zwar erreicht, jedoch kaum übertroffen werden. Das in dieser Arbeit entwickelte Verfahren dient als wichtige Grundlage für weitere Forschungen.

Inhaltsverzeichnis

1. Einleitung

Das Ziel dieser Arbeit besteht darin ein System zur Segmentierung und Gruppierung von Audiodaten zu entwickeln. Das bedeutet es sollen verschiedene Sprecher in beliebigen Audiodateien, die neben Sprache auch Geräusche, Musik oder Stille beinhalten, wiedererkannt werden. Relevant ist dieses Thema im Bereich der Spracherkennung. Der erste Schritt bei der modernen Spracherkennung ist die Wiedererkennung einzelner Sprecher, sodass daraufhin eine Sprechernormalisierung durchgeführt werden kann. Je ungenauer die einzelnen Sprecher erkannt werden, umso schlechter sind auch die Ergebnisse des nachgeschalteten Spracherkenners.

Eine Sprechererkennung wird üblicherweise nach dem folgenden oder einem vergleichbaren Schema durchgeführt: Zuerst werden Geräusche, Musik und Stille mit Hilfe eines Hidden Markov Models, dessen Emissionswahrscheinlichkeiten von einem Gaussian Mixture Model beschrieben werden, detektiert und anschließend entfernt. Danach wird die verbleibende Sprache mit einer beliebigen Metrik (z.B. Kullback-Leibler-Divergenz) in beispielsweise 10s lange gleich klingende Stücke vorsegmentiert. Im letzten Schritt werden die einzelnen Stücke mit einem Clustering-Algorithmus (z.B. hierarchisch agglomeratives Clustering-Verfahren) und einem entsprechenden Distanzmaß solange vereint, bis ein gesetzter Schwellwert überschritten wird.

Das Problem bei dieser Vorgehensweise besteht darin, dass die Grenzen der vorsegmentierten Stücke meist nicht neu angepasst werden und die einzelnen Stücke auch nicht neu trainiert werden. Das kann zu schlechteren Ergebnissen führen, wenn ein vorsegmentiertes Stück mehr als nur einen Sprecher enthält. Im Rahmen der vorliegenden Untersuchung soll eine Sprechersegmentierung entwickelt werden, die die Grenzen neu berechnet und jedes Stück neu trainiert, um dadurch die Güte der Sprechersegmentierung zu verbessern. Ein System zur Segmentierung und Gruppierung (QClust) ist bereits vorhanden. Da es sich bei diesem System um eine Fremdentwicklung handelt, bei der weder der Quellcode noch die genaue Funktionsweise bekannt oder veränderbar sind, soll dieses durch die im Rahmen dieser Arbeit entwickelten Methode für den Einsatz am KIT ersetzt werden. Die Anforderungen bestehen darin, dass sich die grundsätzliche Funktionsweise des zu entwickelnden Systems an dem in [GLA98] vorgestellten Algorithmus orientieren soll und dass sich die Qualität der Spracherkennung des ARS-Systems aufgrund der neuen Komponente nicht verringert.

Es können die Computer des IfA (Institute for Anthropomatics) des Karlsruher Instituts für Technologie benutzt werden. Rechenintensive Berechnungen dürfen auf dem Serverclus-

ter des IfA durchgeführt werden. Es existieren bereits verschiedene Systeme zur Vorsegmentierung der Daten, welche benutzt werden können. Auch ein System zur automatischen Spracherkennung (ASR-System) ist bereits vorhanden, das die segmentierten und gruppierten Daten im gegebenen Dateiformat einlesen und weiterverarbeiten kann.

In **Kapitel 2** werden die theoretischen Grundlagen erläutert. Dazu gehören der Spracherzeugungs und -wahrnehmungsprozess beim Menschen und der aktuelle Stand der Technik im Bereich der automatischen Spracherkennung.

Danach wird in **Kapitel 3** erklärt warum die Segmentierung und Gruppierung von Audiodaten als Vorverarbeitung für moderne Spracherkenner sinnvoll ist und das Segmentierungsverfahren, wie es in [GLA98] vorgestellt wird, wird erläutert und verschiedene Distanzmaße und Mergefunktionen werden vorgestellt. Außerdem wird das Janus Recognition Toolkit (JRTk) vorgestellt, mit dem der Segmentierungsprozess implementiert wurde.

Kapitel 4 behandelt die Evaluation. Dazu gehören die Erläuterung der verwendeten Testdaten, alle unbekannten Parameter, die verwendeten Evaluationsverfahren sowie die durchgeführten Experimente und deren Ergebnisse.

Kapitel 5 beinhaltet eine Zusammenfassung der Problemstellung und der gewählten Lösung. Außerdem wird die Bedeutung der Ergebnisse der Experimente und des entstandenen Segmentierungsprozesses erläutert.

2. Theorie

In diesem Kapitel werden die theoretischen Grundlagen beschrieben, die für das weitere Verständnis nötig sind. Dies beinhaltet die Spracherzeugung und -wahrnehmung des Menschen, die maschinelle Spracherkennung und die Segmentierung.

2.1 Spracherzeugung und -wahrnehmung

Abbildung 2.1 zeigt eine Übersicht über die menschliche Spracherzeugung und -wahrnehmung.

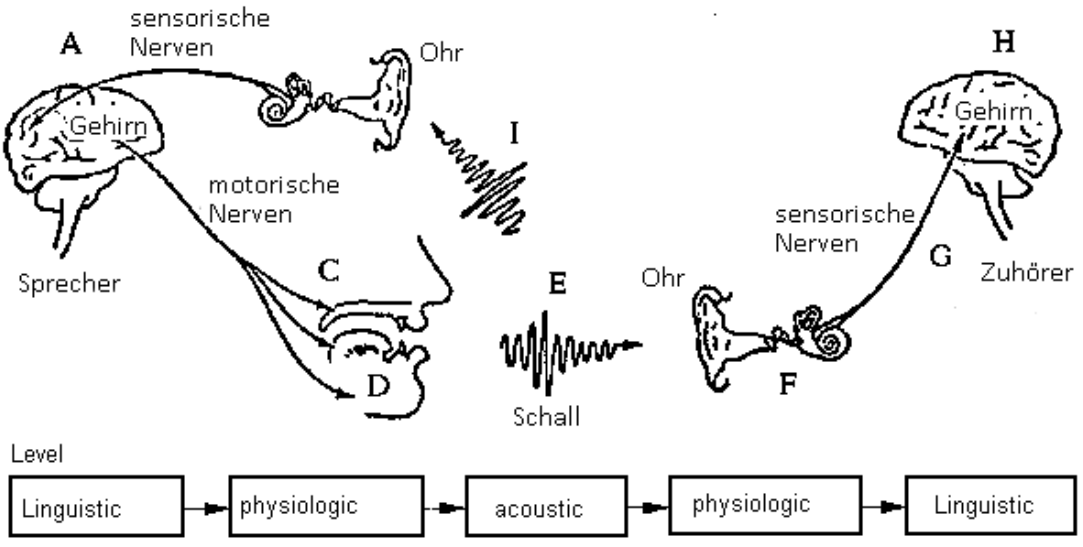

Abbildung 2.1: Überblick: Menschliche Spracherzeugung und -wahrnehmung von Mensch zu Mensch nach [DP93]

Eine Nachricht soll zu einem Zuhörer übertragen werden. Das entspricht einem neurophysiologischen Prozess im Gehirn. Die Nachricht wird in eine Reihe vom Wörtern zerlegt. Über die motorischen Neuronen werden verschiedene Artikulatoren angesteuert. Daraus resultieren verschiedene Repositionierungen und Bewegungen der Artikulatoren und es wird

ein akustisches Sprachsignal generiert. Das Signal wird als per Schalldruck messbare Wellen übertragen. Das Signal passiert über das Außen- und Mittelohr des Zuhörers die Cochlea im Innenohr, die eine Frequenzanalyse durchführt. Die Ergebnisse werden als Nervenimpuls über die sensorischen Neuronen zum Gehirn übertragen. Ein neuro-physiologischer Prozess übersetzt die Nervenimpulse zurück in Wörter und interpretiert diese als Nachricht.

2.1.1 Erzeugung der Sprache

Die menschliche Spracherzeugung besteht im Wesentlichen aus der Stimmgebung, eine Anregung der Stimmbänder (Glottis) beim Ausatmen, und der Resonanzabildung, eine Modulation des Schalls im Vokaltrakt durch Rachen-, Mund- und Nasenraum. Beim Stimmbildungsprozess strömt also Luft von der Lunge durch die Luftröhre, regt beim Passieren die Stimmbänder zur Schwingung an und wird dann im Vokaltrakt zu verschiedenen Lauten ausgeformt.

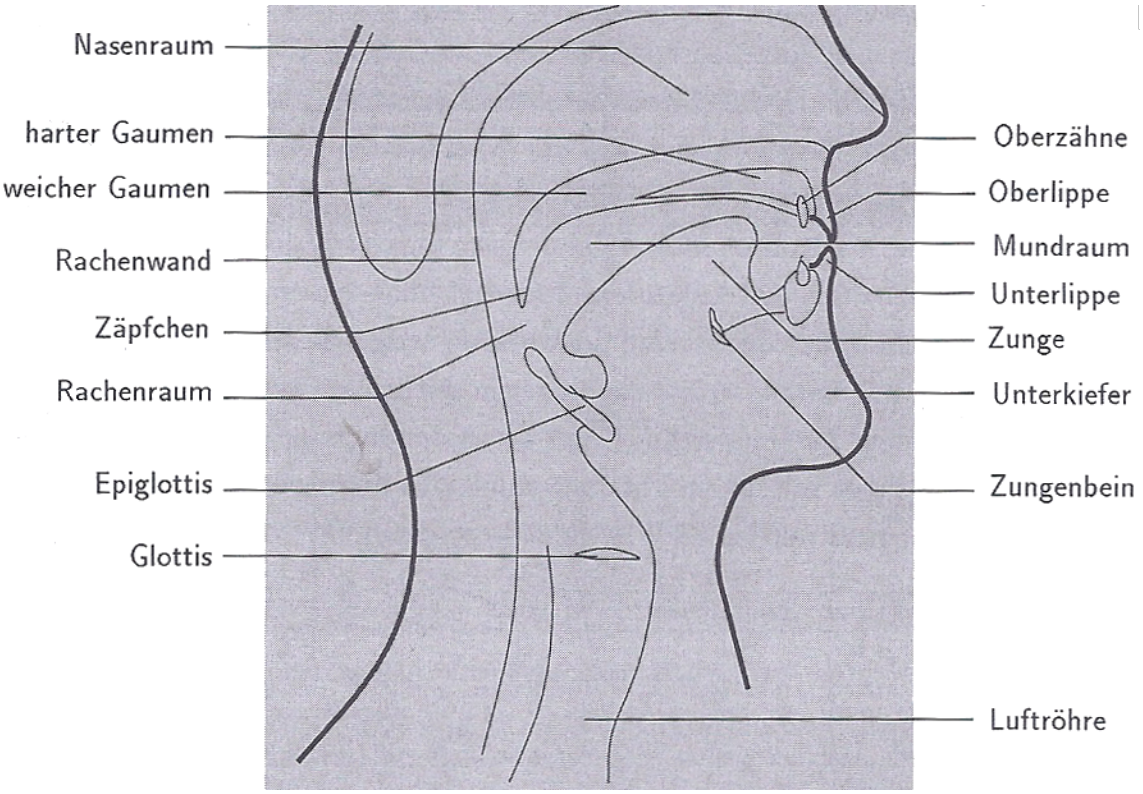

Abbildung 2.2: Artikulationsapparat des Menschen [ST95]

Diese schematische Darstellung zeigt die wichtigsten Bestandteile des menschlichen Artikulationsapparates [ST95].

Bei der Stimmgebung entsteht ein Öffnungs- und Schließbewegungszyklus der Stimmbänder. Wenn die Stimmbänder geschlossen sind verhindern sie, dass die Luft aus den Lungen passieren kann. Ab einem gewissen Punkt werden durch den steigenden Luftdruck die Stimmbänder aufgedrückt und die Luft kann hindurchströmen. Dadurch fällt der Luftdruck und die Stimmbänder verschließen sich wieder. Dieser Vorgang wiederholt sich mehrere Male pro Sekunde. Wie lange ein einzelner Öffnungs-Verschluss-Zyklus dauert ist abhängig von Konsistenz und Größe der Stimmbänder und vom Luftdruck. Bei großen Männern wiederholt sich dieser Zyklus ca. 60 mal pro Sekunde, bei Kindern ca. 300 mal. Durch die Schwingung der Stimmbänder wird ein Ton mit einer Grundfrequenz und einer Reihe von

Obertönen erzeugt. Je nach Resonanz des Vokaltraktes (Anhängig von Position der Zunge, Zähne, Gaumen, ...) werden manche Obertöne verstärkt und andere gedämpft. Obertöne die in der Nähe der Resonanzfrequenzen liegen und somit verstärkt werden nennt man Formanten. Die ersten beiden Formanten können genutzt werden um Vokale zu unterscheiden.

2.1.1.1 Laute

Vom menschlichen Artikulationssystem können verschiedene Laute erzeugt werden. Alle Laute können in stimmlos (wie [t] oder [s]) oder stimmhaft (z.B. [m] oder [n]) eingeteilt werden. Diese werden jeweils dadurch gebildet, dass die Stimmbänder weit offen stehen und die Luft dadurch ungehindert hindurchströmen kann oder dass aufgrund der Stimmgebung die Stimmbänder in Schwingung versetzt werden.

2.1.1.2 Konsonanten

Konsonanten können in verschiedene Kategorien eingeteilt werden. Die wichtigsten im Deutschen sind Plosive, Nasale, Frikative, Approximanten und Laterale.

Plosive oder Verschlusslaute (z.B. [p] oder [b]) entstehen durch einen kurzzeitigen Verschluss der Artikulatoren und eine anschließende Öffnung. Bei Nasalen (z.B. [n] oder [m]) tritt der Luftstrom ganz oder teilweise durch die Nase aus. Wegen einer Änderung des Resonanzverhaltens entsteht so eine Klangveränderung. Reibelaute oder Frikative (z.B. [s] oder [f]) entstehen durch eine Verängung der Artikulatoren. Bei Approximanten (z.B. [w]) und Laterale (z.B. [l]) strömt die Luft im Mundraum entweder zentral oder an einem Hindernis (Zunge) seitlich vorbei. Für weitere Informationen über die verschiedenen Laute siehe [ST95] Kapitel 2.1.1.

2.1.1.3 Vokale

Im Gegensatz zu Konsonanten gibt es bei Vokalen keine feststehende Reihe von Kategorien, in die sie eingeteilt werden können, sondern vier Merkmale die jeden einzelnen Vokal charakterisieren.

Die vertikale Position der Zuge steht für den Öffnungsgrad des Mundes. Dieses Merkmal reicht von geschlossen bis offen. Die horizontale Position der Zunge beschreibt wie weit vorne bzw. hinten sich die Zunge befindet. Ein weiteres Merkmal ist die Stellung der Lippen. Sie sind entweder gerundet oder ungerundet. Die Dauer eines Vokals unterscheidet Vokale wie [ε] in ”Bett” oder [ε:] in ”Fähre”.

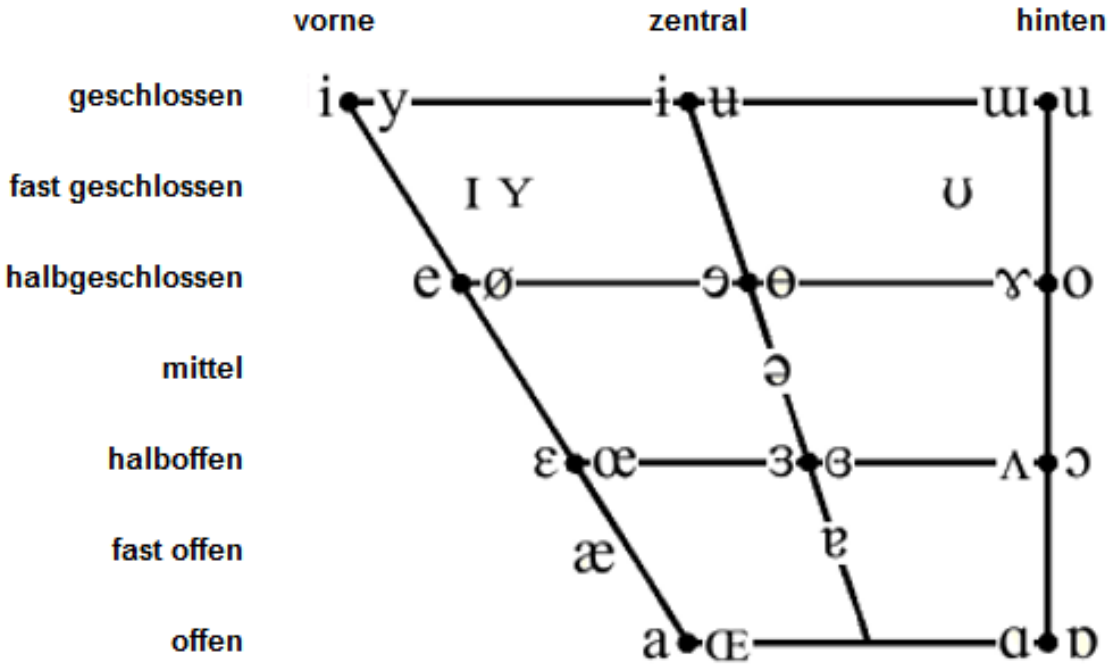

Abbildung 2.3: Vokalviereck [IPAa]

Das Vokalviereck stellt anschaulich die wichtigsten Vokale anhand der ersten drei Merkmale dar. Die Vokaldauer wird nicht beachtet. Auf der Horizontalen ist die horizontale Position der Zuge aufgetragen, auf der Vertikalen die vertikale Position. Wenn sich zwei Laute nur durch die Stellung der Lippen unterscheiden stehen sie im Vokalviereck nebeneinander [ST95, S. 23-25].

Alternativ könnten die beiden Achsen des Vokalvierecks auch mit den beiden ersten Formanten beschriftet werden.

2.1.1.4 Phone und Phoneme

Ein Phon ist die kleinste unterscheidbare Lauteinheit. Ein Phonem hingegen ist die kleinste bedeutungsunterscheidende Lauteinheit. Ein Phonem ist also eine abstrakte Klasse die mehrere Phone zusammenfasst, die in einer bestimmten Sprache dieselbe Bedeutung haben.

Ein Beispiel dafür wäre das Phonem /r/ das z.B. in ”rot” vorkommt. Je nachdem ob man es gerollt oder frikativ ausspricht erzeugt man verschiedene Laute bzw. Phone, das Wort ”rot” hat aber immer dieselbe Bedeutung. Deshalb sind beide Phone ([ʁ] und [r]) eine Realisierung des Phonems /r/. Solche Phone werden Allophone genannt.

Das Internationale Phonetische Alphabet (IPA) hat zum Ziel, alle vom Menschen erzeugbaren Laute festzuhalten. Das IPA wurde von der International Phonetic Association entwickelt und ist aktuell das am weitesten verbreitete Lautschriftsystem. Im IPA wird jedem Laut ein spezifisches Zeichen zugeordnet. Bei den Zeichen des Vokalvierecks aus Abbildung 2.3 handelt es sich um IPA-Zeichen. Eine vollständige Liste aller Zeichen ist in [IPAb] aufgeführt.

2.1.2 Übertragung von Sprache

Schall ist eine Druckwelle die von einem vibrierenden Objekt erzeugt wird. Die Vibration setzt die Partikel des umgebenden Mediums in vibrierende Bewegung. Da sich die Partikel parallel zur Bewegungsrichtung der Welle bewegen entsteht eine Longitudinalwelle. Dadurch entstehen Verdichtungen und Verdünnungen innerhalb des Mediums. Bei 20 Grad Celsius breitet sich Schall in Luft mit ca. 343 m/s aus. Die Frequenz einer Welle wird gemessen als die Anzahl der Vor- und Zurückbewegungen eines Partikels pro Sekunde. Die Einheit ist Herz. Je größer die Frequenz umso höher klingt der Ton.

$$1Hertz = 1Vibration/Sekunde \tag{2.1}$$

Die Schallintensität (Einheit: Pascal, $1Pa = 1N/m^2$) ist des Maß für die Lautstärke des Tons. Hörbare Töne liegen zwischen 10^{-5} und 10^2 Pascal. Aufgrund dieses großen Bereichs ist es üblich die Schallintensität mit einer logarithmischen Skala zu messen. Die hierfür übliche Größe ist Dezibel [dB]. Dabei wird die Größe der Schallintensität relativ zu einer Referenzgröße angegeben. Deshalb hat Dezibel auch keine physikalische Einheit. Dezibel ist wie folgt definiert:

$$L_{dB} = 10 \times log_{10}(\frac{P_1}{P_0}) \tag{2.2}$$

P_1 und P_0 sind hier zwei Schallintensitäten und L_{dB} das Verhältnis in Dezibel. In der Akustik wird P_0 gewöhnlich auf die untere Schwelle der gerade noch hörbaren Schallintensität ($P_0 := 0dB, P_0 = 10^{-5}Pa$) gesetzt.

2.1.3 Wahrnehmung von Sprache

Das Ohr besteht aus drei Hauptteilen. Das Außenohr dient dazu den Schall aufzufangen und ans Mittelohr weiterzuleiten.
Das Mittelohr transformiert die Energie des Schalls in Vibrationen der internen Knochenstruktur und leitet diese Vibrationen als Druckwelle zum Innenohr weiter.
Das Innenohr wandelt mit Hilfe der Innenohrflüssigkeit diese Druckwelle in Nervenimpulse um, die zum Gehirn übermittelt werden.

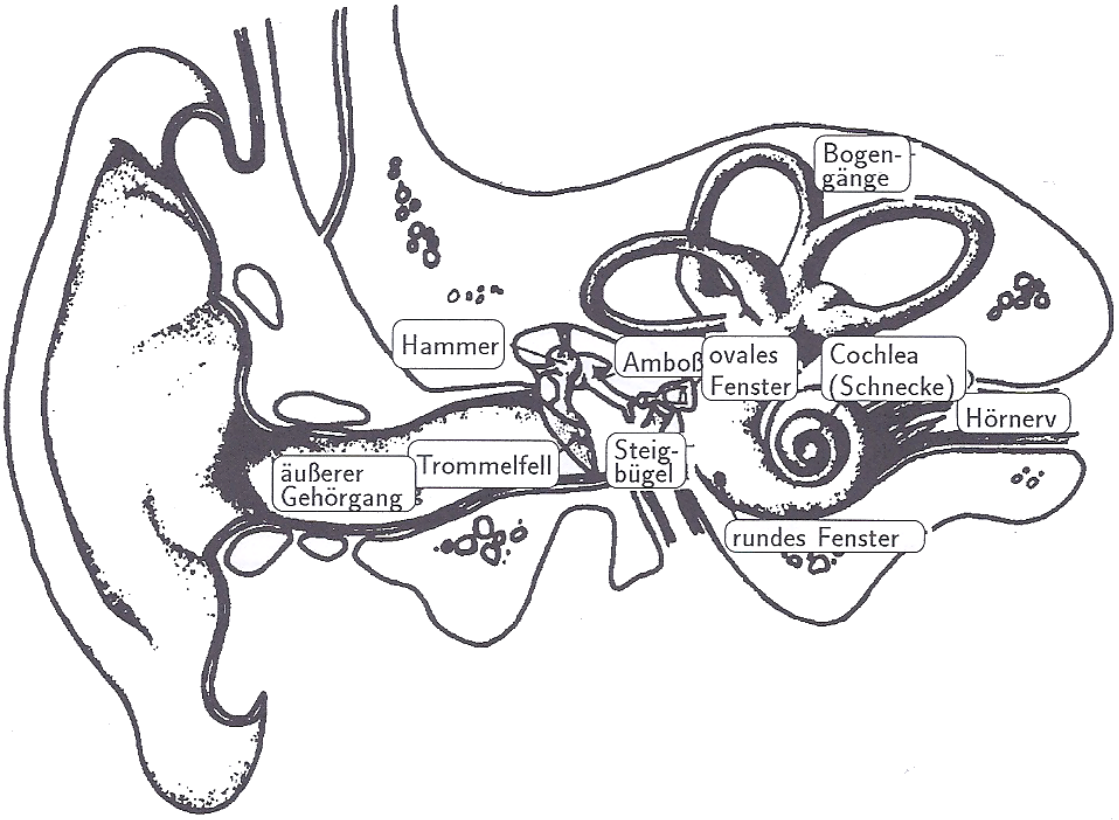

Abbildung 2.4: Anatomie des menschlichen Ohrs [Hol91]

Der Schall wird von Außenohr zum Mittelohr geleitet. Dort versetzt er das Trommelfell in Bewegung. Ein mechanischer Umwandler, bestehend aus Hammer, Ambos und Steigbügel, der an der anderen Seite des Trommelfells anliegt, konvertiert den Schall in Vibrationen am ”runden Fenster”, dem Eingang zur Hörschnecke (Cochlea). Die Hörschnecke zerlegt die Welle in einzelnen Frequenzen und wandelt diese in elektrische Impulse um.

Mel-Skala

Die Mel-Skala beschreibt den Zusammenhang zwischen tatsächlicher Tonhöhe und wahrgenommener Tonhöhe. Sie wurde 1937 von Stanley Smith Stevens, John Volkman und Edwin Newmann aufgestellt. Die wahrgenommene Tonhöhe wird mit Mel (engl: Melody) bezeichnet.

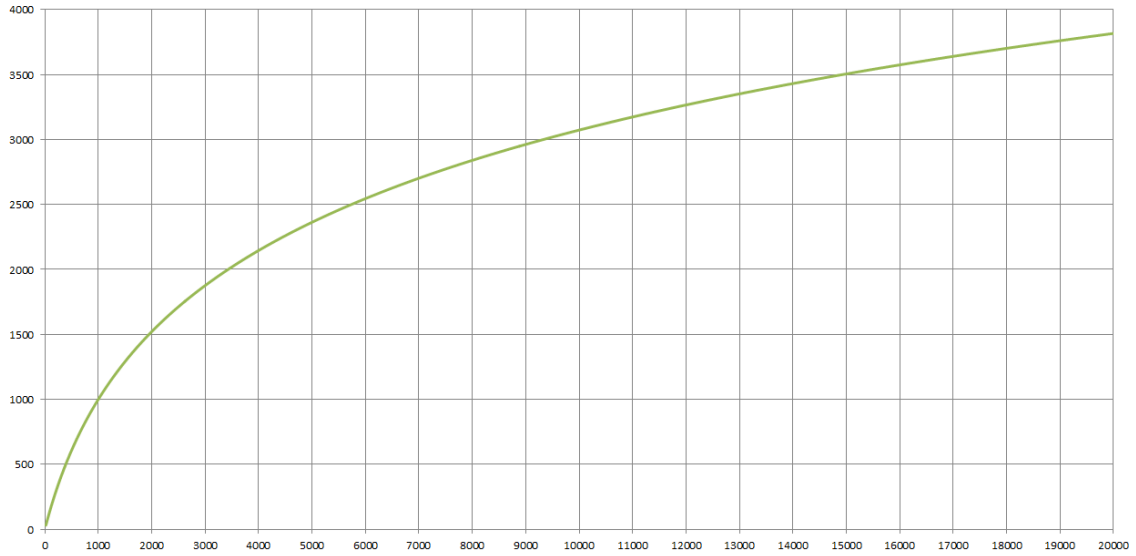

Abbildung 2.5: An der x-Achse ist die tatsächliche Tonhöhe aufgetragen und an der y-Achse die wahrgenommene Tonhöhe (in Mel) nach [O'S87].

Wie zu sehen ist, können bei steigender Frequenz die Tonhöhen immer schlechter unterschieden werden.

2.2 Automatische Spracherkennung

Die automatische Spracherkennung, abgekürzt ASR (Automatic Speech Recognition) ist die maschinelle Umwandlung von menschlich gesprochener Sprache in ihre zugehörige textuelle Repräsentation als Sequenz von Worteinheiten. Es existieren ein paar verschiedene grundlegende Möglichkeiten wie automatische Spracherkennungssysteme arbeiten. Beim akustisch phonetischen Ansatz (siehe [RJ93]) wird versucht mehrere Detektoren zu entwickeln, die entweder statistische oder wissensbasierte Informationen enthalten und die gesprochene Sprache zu annotieren und zu segmentieren. Der künstlich intelligente Ansatz (siehe [RJ93]) versucht die Intelligenz menschlicher Experten als Reihe von Regeln bzw. Sammlung von Wissen in eine automatisch auswertbare Form zu bringen und auf dieser Basis die Klassifikation durchzuführen. Beim konnektionistischen Ansatz (siehe [WL90]) werden Neuronale Netze verwendet. Es gibt auch Spracherkenungssysteme die verschiedene Ansätze kombinieren, deshalb ist eine eindeutige Zuordnung nicht immer möglich. Da der aktuelle Stand der Technik jedoch die statistische Spracherkennung ist, wird im Folgenden nur darauf eingegangen.

2.2.1 Aufbau

Bei der statistischen Spracherkennung wird die Erkennung von Sprache als Mustererkennungsproblem betrachtet. Das Muster wäre dabei eine Reihe von Merkmalsvektoren einer beliebige Audioaufnahme X. Die Klassen wären alle möglichen Wortfolgen W, die aus einer Menge von Wörtern gebildet werden können. Die Klassifikationsaufgabe besteht nun darin die Wortfolge zu finden, die am besten zur Audioaufnahme passt.

$$\hat{W} = \arg\max_W P(W|X) = \arg\max_W \frac{P(X|W) \times P(W)}{P(X)} = \arg\max_W P(X|W) \times P(W) \quad (2.3)$$

Die Wortfolge $\hat{W}$ ist diejenige, deren Wahrscheinlichkeit bei gegebener Audioaufnahme X maximal ist. Mit Hilfe des Satzes von Bayes wird die Formel nun umgeformt. $p(X|W)$ ist dabei die Wahrscheinlichkeitsverteilung der Merkmalsvektoren gegeben einer bestimmten Wortfolge W, $p(W)$ ist die a-priori-Wahrscheinlichkeit dafür, dass die Wortfolge W gesprochen wird und $p(X)$ ist die Wahrscheinlichkeit, dass X überhaupt beobachtet wird.

Da die Aufgabe darin besteht die wahrscheinlichste Wortfolge zu bestimmen und $p(X)$ für alle möglichen Wortfolgen gleich ist, kann dieser Wert weggelassen werden.

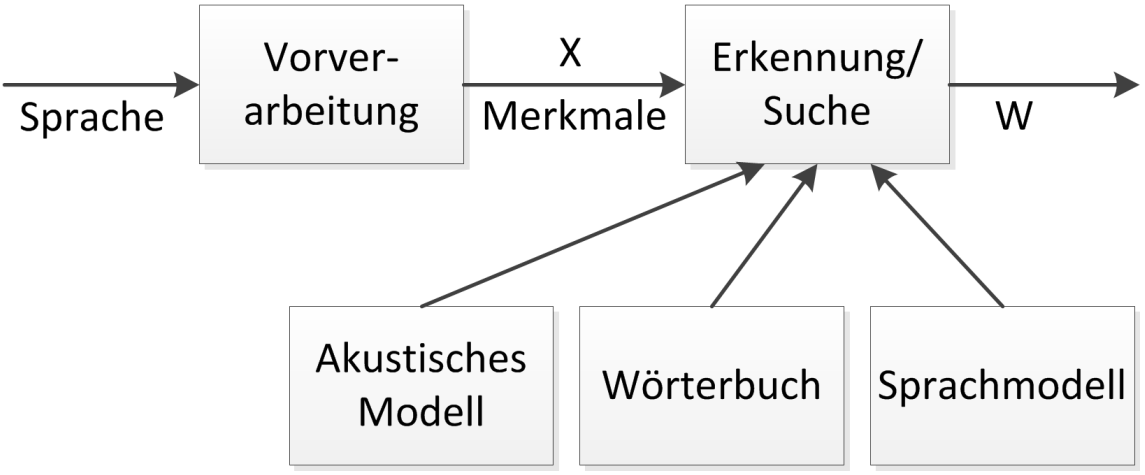

Abbildung 2.6: Übersicht über ein ASR-System

In Abbildung 2.6 wird eine schematische Übersicht über ein statistisches ASR-System gegeben. Anfangs wird mittels einer Vorverarbeitung aus dem Audiosignal eine Reihe von Merkmalsvektoren extrahiert. Anhand dieser Merkmalsvektoren wird dann mit einer Suche/Erkennung die wahrscheinlichste Wortfolge ermittelt. Das akustische Modell ist dabei eine Näherung der Wahrscheinlichkeitsverteilung $p(X|W)$ und das Sprachmodell eine Näherung der A-priori-Wahrscheinlichkeit $p(W)$.

2.2.2 Vorverarbeitung

Die Vorverarbeitung ist der erste Schritt des Spracherkennungsprozesses. Aus dem Audiosignal werden in mehreren Schritten akustische Merkmalsvektoren extrahiert. Das Ziel besteht darin, das Audiosignal für die Erkennung aufzubereiten und möglichst nur Informationen zu extrahieren, die für die Spracherkennung wichtig sind. Nutzlose Informationen wie z.B. Stimmlage des Sprechers, Störgeräusche, usw. sollen so gut wie möglich herausgefiltert werden.

1. Im ersten Schritt muss das Audiosignal digitalisiert werden. Bei der Abtastung muss das Nyquist-Shannon-Theorem eingehalten werden um Aliasing zu vermeiden. Es muss mit einer Frequenz abgetastet werden, die größer ist als das Zweifache der größten im Signal vorkommenden Frequenz. Zur Bandbegrenzung kann ein Tiefpassfilter verwendet werden. Üblich sind 16 oder 44 kHz. Bei der Quantisierung wird der Wertebereich auf eine Skala von mehreren festgelegten Schritten abgebildet. Dadurch entsteht Rauschen, das so genannte Quantisierungsrauschen. Menschliche Sprache hat einen Schallleistungspegel von bis zu ca. 60 dB, deshalb sollte das Signal-Rausch-Verhältnis (SNR) in einer ähnlichen Größenordnung liegen. SNR berechnet sich ähnlich wie dB. Der Unterschied besteht nur darin, dass die Schallleistung nicht relativ zur kleinsten hörbaren Schallleistung angegeben wird, sondern

relativ zur Rauschleistung. Rein rechnerisch wäre eine Quantisierung mit 12 Bit (2^{12} Schritte) ausreichend. In der Praxis werden aber entweder 8 oder 16 Bit verwendet. Da die Amplituden eines Sprachsignals nicht gleichverteilt sind, sind gleichmäßige verteilte Quantisierungsstufen nicht ideal. Daher wird meist das μ-law-Koding, wie es im Puls-Code-Modulation-Verfahren [Kam08] benutzt wird, verwendet.

2. Im zweiten Schritt wird auf das abgetastete Signal eine schnelle Fouriertransformation angewandt um das Signal in den Spektralbereich zu transformieren bzw. ein komplexes Spektrum zu erhalten. Um das Leistungsspektrum zu erhalten werden die Real- und Imaginäranteile jeweils quadriert und dann summiert. Das Problem dabei besteht darin, dass es keinen Sinn machen würde die Fouriertransformation auf das gesamte Signal anzuwenden, da sich die einzelnen Frequenzen über die Zeit ändern und das resultierende Frequenzspektrum keine Informationen darüber enthalten würde. Daher wird das Frequenzspektrum auf mehreren Abschnitten des Signals berechnet. Bei steigender Anzahl der Abschnitte werden diese immer kürzer, so erhöht sich zwar die zeitliche Auflösung, es sinkt jedoch die Frequenzauflösung. Deshalb muss ein guter Kompromiss zwischen Frequenz- und Zeitauflösung gefunden werden. Üblich ist eine Fenstergröße von 16ms und eine Verschiebung um je 10ms. Da ein einfaches Ausschneiden eines Abschnitts aus dem Signal einer Multiplikation mit einem Reckteckfenster entspricht und eine Multiplikation im Zeitbereich einer Faltung in Spektralbereich entspricht, würde das resultierende Frequenzspektrum dadurch verfälscht. D.h. eine Fensterfunktion deren Spektrum ein Impuls ist, wäre ideal. Leider existiert eine solche Fensterfunktion nicht. Häufig wird das Hamming-Fenster [HAH01] verwendet, da dessen Spektrum einem Impuls zumindest ähnelt.

3. Als nächstes werden die Frequenzspektren entsprechend der Mel-Skala zusammengefasst.

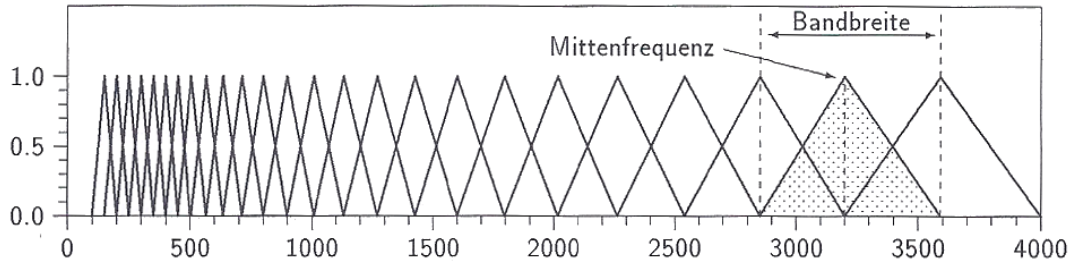

Abbildung 2.7: Dreieckfilterbank für 25 Frequenzgruppen [ST95]

Die Zusammenfassung der einzelnen Frequenzen innerhalb des Spektrums erfolgt durch eine gewichtete lineare Kombination. Die Gewichte sind in Abbildung 2.7 dargestellt. Dadurch werden die Dimensionen entsprechend der menschlichen Wahrnehmung reduziert. Die Anzahl der Bänder ist je nach System unterschiedlich.

4. Im letzten Schritt wird das Mel-Spektrum logarithmiert und anschießend eine diskrete Kosinustransformation darauf angewendet. Von den so entstehenden Mel-Frequenz-Cepstrum-Koeffizienten werden meist nur die unteren Koeffizienten verwendet, damit nur die Makrostruktur des Mel-Spektrums erhalten bleibt. Häufig werden die unteren 13 oder 16 Koeffizienten verwendet. So soll der Einfluss des Übertragungskanals minimiert werden. Die verbleibenden Mel-Frequenz-Cepstrum-Koeffizienten nennt man Cepstum.

2.2.3 Akustisches Modell

Die Aufgabe des akustischen Modells besteht nun also darin die Wahrscheinlichkeitsdichteverteilung der Merkmalsvektoren, die von der Vorverarbeitung geliefert werden, bei beliebigen gegebenen Wortfolgen anzunähern. Dazu wird das akustische Modell mit Trainingsdaten trainiert, um den zugrunde liegenden Parametern einen Wert zuzuweisen. Daraus kann dann für die Erkennung unbekannter Sprache die Wahrscheinlichkeitsverteilung der Merkmalsvektoren geschätzt werden.

2.2.3.1 Gauß-Mischverteilung

Eine Gauß-Mischverteilung (eng. gaussian mixture model (GMM)) besteht aus einer Menge mehrdimensionaler gewichteter Normalverteilungen. Dadurch kann, je nach Anzahl der Normalverteilungen, jede beliebige Wahrscheinlichkeitsdichteverteilung approximiert werden.
Eine mehrdimensionale Normalverteilung ist wie folgt definiert:

$$N(x|\mu,\Sigma) = \frac{1}{(2\pi)^{\frac{p}{2}} \times |\Sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)} \tag{2.4}$$

$N(x|\mu,\Sigma)$ definiert eine Dichtefunktion einer reellen p-dimensionalen Zufallszahl x mit Hilfe des Erwartungswertvektors μ und der Kovarianzmatrix Σ.

$$p(x) = \sum_{k=1}^{K} P_k \times N(x|\mu_k,\Sigma_k) \tag{2.5}$$

K ist die Anzahl der Normalverteilungen, P_k ist das Gewicht der K^{ten} Normalverteilung. Wenn hinreichend viele Beobachtungen der Zufallszahl x bekannt sind, kann mit erwartungstreuen Schätzern μ und Σ berechnet werden [HAH01].

Da ein GMM eine lineare Kombination aus mehreren Normalverteilungen ist, müssen nicht nur μ, Σ und die Gewichte geschätzt werden, sondern es muss auch entschieden werden welche Zufallszahl x zu welcher Normalverteilung gehört. Dafür wird der Expectation-Maximization-Algorithmus [Mit97] (EM-Algorithmus) verwendet. Grundsätzlich kann der EM-Algorithmus für viele verschiedene Modelle (in unserem Fall GMMs) verwendet werden. Der Expectation-Schritt, die Zuordnung der Daten zu den Modellen, und der Maximization-Schritt, die Berechnung neuer Parameter, gegeben der zuvor ermittelten Zuordnung, wechseln sich dabei laufend ab.

1. Es werden Initialisierungswerte für die Parameter aller Normalverteilungen festgelegt. Die Initialisierungswerte können entweder Zufallswerte sein oder durch Clusterverfahren (z.B. k-means-Algorithmus [Mac03, S. 285-290]) festgelegt werden.
2. Alle Datenpunkte werden der Normalverteilung zugeordnet, die für den jeweiligen Datenpunkt den maximalen Funktionswert ergibt. Der Datenpunkt kann, abhängig von der genauen Funktionsweise, entweder absolut oder anteilig zugewiesen werden.
3. Erwartungstreue Schätzer berechnen, entsprechend der vorher zugewiesenen Datenpunkte, die Parameter neu.

Die Schritte 2 und 3 werden so lange wiederholt bis sich die Parameter nicht mehr ändern oder die Änderung der Parameter unterhalb eines bestimmten Schwellwertes liegt. Die Gewichte der Normalverteilungen werden anhand der Anzahl der Datenpunkte pro Normalverteilung festgelegt.

2.2.3.2 Universal Background Model

Ein Problem beim Training eines GMMs mit Hilfe des EM-Algorithmuses besteht darin, dass eine bestimmte Mindestmenge an Trainingsdaten vorhanden sein muss, um alle Parameter (μ, Σ und P_k) korrekt und stabil bestimmen zu können. Daher wird häufig ein Universal Background Model (UBM) verwendet wenn die Menge der vorhandenen Trainingsdaten gering ist.

Ein UBM ist ein GMM das mit dem EM-Algorithmus auf vielen Stunden Sprache von vielen verschiedenen Sprechern trainiert wurde und mehrere hundert oder tausend Normalverteilungen enthält. Mit dem UBM wird der Versuch unternommen, eine sprecherunabhängige Verteilung von Merkmalsvektoren zu repräsentieren. Wenn ein GMM auf einer geringen Menge von Daten trainiert werden soll, werden die Normalverteilungen des UBMs gesucht, die auf die Daten am besten passen und entsprechend der Daten angepasst. Die angepassten Normalverteilungen bilden ein neues GMM, das auf die Daten besser passt. So werden die Informationen, die aufgrund der wenigen Trainingsdaten fehlen, durch Vorwissen (allgemeine Sprachinformationen des UBMs) ersetzt.

Es ist möglich alle Parameter anzupassen oder nur einen Teil davon. Eine in [aTFQD00] vorgestellte Methode passt sowohl die Mittelwerte (μ) als auch die Kovarianzmatrix (Σ) an. Ein umfassender Überblick über die möglichen Methoden zur Anpassung der Parameter wird in [KL10] gegeben.

2.2.3.3 Hidden Markov Models

Ein Hidden Markov Model (HMM) ist eine Markov-Kette bei der der aktuelle Zustand unbekannt ist. HMMs werden verwendet um die zeitliche Abfolge der Merkmalsvektoren zu modellieren. Ein HMM besteht aus einer Anzahl von Zuständen. In jedem Zeitschritt geht das HMM von einem Zustand in denselben oder einen anderen über. Die Zustandsübergänge sind durch Wahrscheinlichkeiten festgelegt, die nur vom aktuellen Zustand abhängen und über die Zeit gleich bleiben. Es ist unbekannt in welchen Zustand sich das HMM aktuell befindet. Stattdessen sind jedem Zustand Ausgabesymbole zugeordnet, die Beobachtet werden können. Die Emissonswahrscheinlichkeiten bestimmen welcher Zustand welche Ausgabesymbole erzeugt.

Ein HMM ist definiert durch:

- Eine Menge S mit N Zuständen $S = s_1, ..., s_N$.
- Ein Alphabet V mit M Symbolen $V = v_1, ..., v_M$. Es kann sich auch um eine Menge handeln z.B. die rationalen Zahlen.
- Eine Übergangsmatrix $A \in R^{n \times n}$. Mit a_{ij} als Wahrscheinlichkeit, dass vom Zustand s_i in den Zustand s_j gewechselt wird.
- Eine Emissionswahrscheinlichkeitsverteilung $B \in R^{n \times m}$. Gibt die Wahrscheinlichkeit an, im Zustand s_i die Beobachtung v_j zu machen.
- Die Anfangsverteilung $\pi \in R^n$. Gibt die Wahrscheinlichkeit an, dass s_i der Startzustand ist.

Die Emissionswahrscheinlichkeitsverteilung pro Zustand wird meist von einem Gaussian Mixture Model (GMM) bestimmt.
Die Zustände und Übergänge eines HMMs können zwar beliebige Topologien bilden, aber da damit die menschliche Sprache modelliert werden soll, ist es üblich Links-Rechts Modelle zu verwenden.

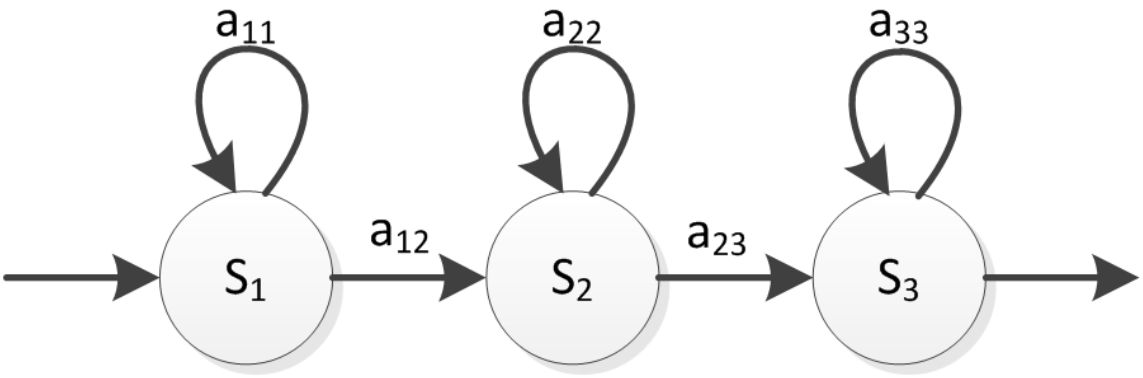

Abbildung 2.8: Beispiel für ein HMM eines Phonems

Es ist üblich für jedes Phonem ein HMM anzulegen. Abb. 3.2 zeigt ein mögliches HMM für ein Phon. Da der Anfang eines Phons anders klingt als die Mitte oder das Ende, hat ein solches HMM häufig drei Zustände. Da Phoneme abhängig von ihrer Umgebung stark unterschiedlich klingen können (ein /a/ auf das ein /t/ folgt klingt z.B. anders als ein /a/ auf das ein /m/ folgt) verwendet man meist Polyphone. D.h. es werden je nach Kontext andere HMMs verwendet. Wenn man als Kontext z.B. den direkten Vorgänger und Nachfolger betrachtet spricht man Triphonen, bei zwei Vorgänger und zwei Nachfolgern von Quintphonen usw. Der Vorteil bei der Verwendung von Polyphonen besteht darin, dass sich dadurch die Genauigkeit der geschätzten Wahrscheinlichkeitsdichteverteilung deutlich verbessert. Der Nachteil ist, dass sich die Anzahl der Ployphone, mit der Länge des Kontextes, exponentiell erhöht. Da z.B. ein Triphon eine geordnete Stichprobe von drei Phonemen darstellt und sich die Phoneme Wiederholen dürfen, gibt es bei 50 Phonemen 50^3 Triphone. Da man für jedes HMM mindestens eine bestimmte Menge an Trainingsdaten braucht um die Parameter der GMMs hinreichend gut bestimmen zu können, nimmt die Menge der nötigen Trainingsdaten deutlich zu. Des Weiteren ist es häufig so, dass manche Polyphone trotz ausreichender Trainingsdaten kaum oder gar nicht vorkommen, weil eine spezifische Phonkombination in der gegebenen Sprache schlicht selten auftritt. Außerdem können verschiedene Polyphone zufällig ähnlich klingen, dann wäre es überflüssiger Aufwand mehrere HMMs zu verwenden.
Daher wird häufig ein sogenanntes Clusterverfahren verwendet um die Polyphone in mehrere Gruppen zusammenzufassen. Im Wesentlichen geht es darum, Polyphone, deren Wahrscheinlichkeitsdichteverteilung der Merkmalsvektoren ähnlich ist, zusammenzufassen, also nur ein Modell zu verwenden das für alle Polyphone der Gruppe hinreichend gut passt. Es sollen bei möglichst wenig Verlust an Diskriminationsfähigkeit möglichst viel an Sprecher, Rechenaufwand und Trainingsdaten eingespart werden.

Mit Hilfe sogenannter Aussprachewörterbücher kann ein HMM für ein ganzes Wort gebildet werden. Darin sind alle Wörter und deren Flexionen aufgeführt die vom ASR-System erkannt werden können. Zu jedem Wort ist dabei die Aussprache, also eine Folge von Phonen, mit angegeben. Die HMMs der einzelnen Phone werden dann entsprechend der Aussprache zusammengesetzt. Da relativ viel Arbeit nötig ist um ein solches Aussprachewörterbuch manuell zu erstellen, gibt es diverse Algorithmen um diese automatisch zu generieren bzw. einen Teil der Arbeit abzunehmen (siehe [Sea10], [BN08] oder [KK94]).

Um HMMs für die Spracherkennung einsetzen zu können müssen folgende Probleme gelöst werden:

Evaluierung: Bei einem gegebenen HMM H und einer Ausgabe O muss die Wahrscheinlichkeit bestimmt werden können, mit der O von H ausgegeben wurde. Dies ist wichtig, um bei gegebenen Audiodaten bestimmen zu können, welches HMM besser passt.

Dekodierung: Bei einem gegebenen HMM H und einer Ausgabe O muss die Zustandsfolge S mit der höchsten Wahrscheinlichkeit gefunden werden. Dies wird benötigt um z.B. die Übergangswahrscheinlichkeiten eines HMMs zu bestimmen.

Training: Bei einem gegebenen HMM und einer Menge Trainingsdaten müssen Parameter gefunden werden, die besser passen. D.h. das HMM muss so eingestellt werden, dass die Wahrscheinlichkeit, dass es die Trainingsdaten tatsächlich produziert hätte, maximal wird.

Um das Evaluierungsproblem zu lösen könnte man theoretisch einfach die Wahrscheinlichkeit dafür berechnen, dass irgendeine bestimmte Zustandskombination, ein sogenannter Pfad, die gegebene Ausgabe erzeugt hat und dann alle möglichen Pfade aufsummieren. Das würde jedoch einen sehr hohen Rechenaufwand bedeuten. Aus diesem Grund wird der Forward- bzw. Backward-Algorithmus [HAH01] verwendet. Dabei werden die Pfade rekursiv Schritt für Schritt, entweder von Hinten oder von Vorne, abgearbeitet. Die dadurch entstehende deutliche Reduktion des Rechenaufwandes kommt dadurch zu Stande, dass nach jedem Schritt die Zwischenergebnisse in einer Matrix gespeichert werden. So muss jeder Pfad nicht komplett neu berechnet werden.

Ähnlich wie beim Evaluierungsproblem könnte das Dekodierungsproblem dadurch gelöst werden, indem die Wahrscheinlichkeit für alle Pfade berechnet wird und der Pfad mit der maximalen Wahrscheinlichkeit gewählt wird. Aber auch hier hätte man einen zu hohen Rechenaufwand. Das Problem wird mit dem Viterbi-Algorithmus [HAH01] gelöst. Die Funktionsweise ähnelt dem Forward-Algorithmus. Der Unterschied besteht darin, dass in jedem Schritt die Zwischenergebnisse nicht jeweils summiert werden, sondern nur das Maximum verwendet wird. So kann der Aufwand deutlich reduziert werden und das Problem ist auch für viele Zustände und lange Beobachtungsfolgen berechenbar.

Der Baum-Welch-Algorithmus passt ein gegebenes HMM besser an gegebene Trainingsdaten an, indem der die Übergangsmatrix A, die Emissionswahrscheinlichkeiten B und die Anfangswahrscheinlichkeiten π anpasst. Die Topologie des HMMs bleibt unverändert (siehe [HAH01]).

2.2.4 Sprachmodell

Wie bereits erwähnt berechnet ein Sprachmodell (Language Model (LM)) die a-priori-Wahrscheinlichkeit $p(W)$. D.h. ein Sprachmodell berechnet die Wahrscheinlichkeit, dass eine bestimmte Wortfolge in der gegebenen Sprache vorkommt. Dazu wird das LM auf großen Mengen Text trainiert und lernt welche Wörter in welchem Kontext wahrscheinlich sind. Es gibt mehrere Modelle dies zu berechnen, aber da die stochastischen Sprachmodelle bis heute die weiteste Verbreitung fanden, werden hier nur diese beschrieben.
Ein LM erfüllt viele wichtige Aufgaben. Es korrigiert Rechtschreibfehler wie z.B. /t/ statt /d/ oder /z/ statt /t//z/ und semantisch sinnlose Sätze die wegen Hintergrundgeräuschen ganz einfach falsche Wörter enthalten oder die auf der Zweideutigkeit von Homophonen beruhen wie z.B. ”man” und ”Mann” oder ”wahre” und ”Ware”.

Die Wahrscheinlichkeit, dass eine bestimmte Wortfolge W bestehend aus N Wörtern $W = w_0, ..., w_N$ in einer gegebenen Sprache Sinn macht, kann zerlegt werden in mehrere bedingte

Wahrscheinlichkeiten.

$$\begin{aligned} P(W) &= P(w_0, ..., w_N) \\ &= P(w_0) \times P(w_1|w_0) \times P(w_2|w_1, w_0) \times ... \times P(w_n|w_{n-1}, w_{n-2}, ..., w_0) \end{aligned} \quad (2.6)$$

Diese bedingten Wahrscheinlichkeiten wären zwar theoretisch berechenbar, indem die Anzahl der Vorkommnisse der gesamten Folge durch die Anzahl der Vorkommnisse der Folge der vergangen Wörter geteilt würde, aber aufgrund der extrem großen Menge möglicher Sätze wäre es unmöglich genug Trainingsdaten zu sammeln.
Deshalb werden sogenannte n-Gramme verwendet. Dabei wird die Wahrscheinlichkeit $P(W)$ angenähert indem man die Folge der vergangenen Wörter auf N begrenzt. Bei z.B. Bigrammen werden insgesamt zwei Wörter betrachtet, das aktuelle und das vorherige, bei Trigrammen das aktuelle und die zwei letzten usw.

n-gramme: $P(w_j|w_{j-1}, w_{j-2}, ..., w_0) \approx P(w_j|w_{j-1}, w_{j-2}, ..., w_{j-(n-1)})$
Unigramme: $P(w_j|w_{j-1}, w_{j-2}, ..., w_0) \approx P(w_j)$
...
Trigramme: $P(w_j|w_{j-1}, w_{j-2}, ..., w_0) \approx P(w_j|w_{j-1}, w_{j-2})$
...

Mit wachsendem n steigt zwar die Qualität der LMs, aber es werden auch deutlich mehr Trainingsdaten benötigt. Wenn es trotzdem passiert, dass z.B. ein bestimmtes Trigramm in den Trainingsdaten nicht häufig genug vorkommt, dann kann der Wert mit Hilfe von Bi- oder Unigrammen interpoliert werden (siehe [HAH01]).

2.2.5 Suche

Jetzt können zwar mit der Vorverarbeitung die Merkmalsvektoren, mit dem akustischen Modell $P(X|W)$ und mit dem Sprachmodell $P(X)$ berechnet werden und somit auch $P(W|X)$, aber es muss noch die Suche durchgeführt werden. D.h. der Suchraum, bestehend aus allen möglichen Wortfolgen, muss durchsucht werden, bis die Wortfolge mit der maximalen Wahrscheinlichkeit gefunden ist. Allerdings ist es aufgrund des riesigen Suchraums nicht möglich eine vollständige Suche durchzuführen. Ein intelligenter Algorithmus muss den Suchraum nach der besten oder einer guten Wortfolge durchsuchen, ohne dabei alle möglichen Wortfolgen zu berechnen.

Um intelligente Suchalgorithmen anwenden zu können, werden die HMMs aller Phone aller Wörter aller Wortfolgen als riesiger gerichteter Graph interpretiert. So können in jedem Schritt z.B. nur die N Besten Pfade weiterverfolgt werden, oder nur die Pfade, deren Wahrscheinlichkeit über einem gewissen Schwellwert liegt. Es existieren auch bereits verschiedene Algorithmen die bei derartigen Problemen angewendet werden können, z.B. der A*-Algorithmus [Ste94]. Für weitere Informationen zu möglichen Suchstrategien (siehe [HAH01]).

2.3 Segmentierung

Im Folgenden sollen die theoretischen Grundlagen zur Segmentierung und Gruppierung von Audiodaten näher erläutert werden.

2.3.1 Gruppierung

Um eine Audiodatei in homogene Segmente zu unterteilen und diese in Gruppen bzw. Klassen einzuteilen, gibt es zwei verschiedene Vorgehensweisen.
Beim überwachten Lernen wird eine feste Anzahl an Klassen vorgegeben, in die die zu klassifizierenden Daten, aufgrund geeigneter, auf den Daten berechneter Merkmale und mit Hilfe eines Klassifikators, eingeteilt werden. Dieses Verfahren besteht aus zwei verschiedenen Phasen, die Trainings- und die Ausführungsphase. Für die Trainingsphase werden sogenannte gelabelte Daten benötigt. Das sind Daten für die die korrekte zugehörige Klasse, entweder manuell oder durch andere Klassifikatoren, bereits ermittelt wurde. Während der Trainingsphase werden die dem Klassifikator zugrunde liegenden Parameter durch geeignete Trainingsmethoden bestimmt. So können, während der Ausführungsphase, für unbekannte Daten die jeweiligen Klassen ermittelt werden [Buc96].
Beim unüberwachten Lernen hingegen werden die Daten automatisch in Klassen eingeteilt, die sich durch charakteristische Muster möglichst maximal voneinander unterscheiden sollen. Häufig werden dazu Clusterverfahren verwendet.
Da es sich bei dem in dieser Arbeit vorgestellten Verfahren um ein unüberwachtes Lernverfahren handelt werden die überwachten Lernverfahren nicht weiter erläutert.

2.3.2 Clusterverfahren

Clusterverfahren teilen eine Menge an Daten in mehrere Gruppen ein, sodass die Daten innerhalb einer Gruppe (sogenannte Cluster) möglichst wenig Variabilität aufweisen. Dies wird Innerklassenvariabilität genannt. Die Variabilität zwischen dein Clustern, die Intraklassenvariabilität, hingegen soll maximiert werden. Es gibt viele verschiedene Algorithmen die sich in ihrer Vorgehensweise, der Anzahl und Position der resultierenden Cluster und ihrer Effizienz stark unterscheiden.

Hierarchische Clusterverfahren stellen eine Gruppe von Clusterverfahren da, die alle nach dem selben Prinzip arbeiten. Dabei werden die Cluster schrittweise immer weiter verfeinert oder vergöbert, sodass eine Hierarchie entsteht. Die Cluster können darüber hinaus nicht mehr verändert werden.

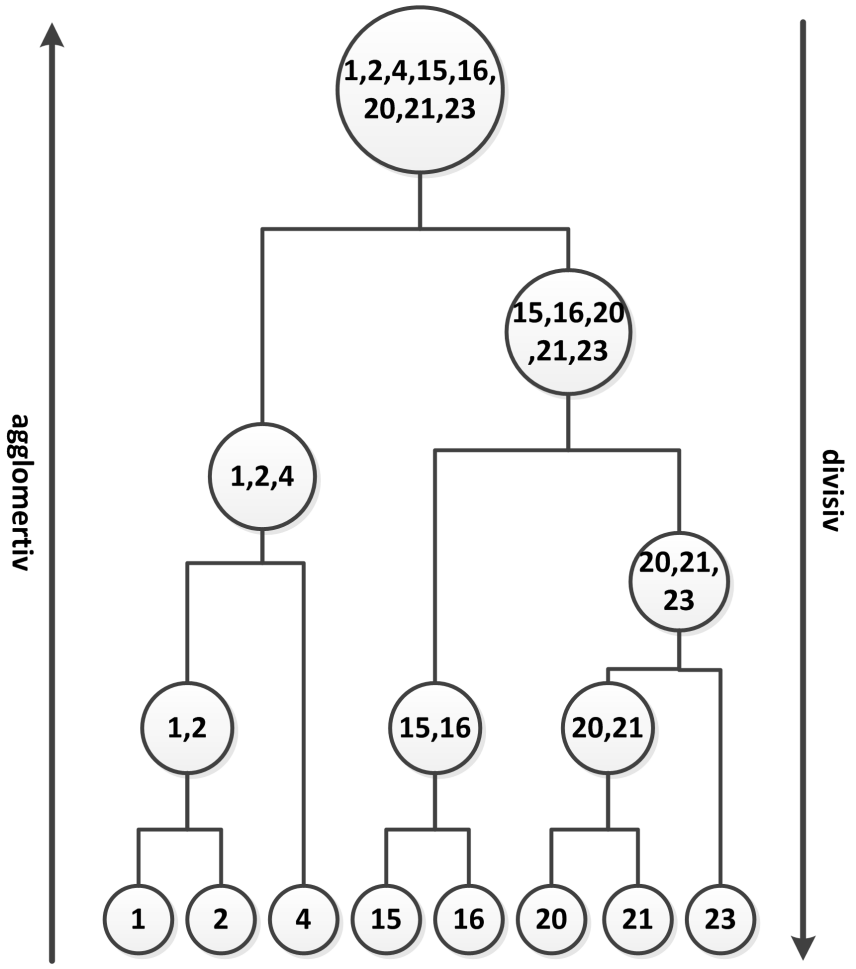

Abbildung 2.9: Beispiel für ein agglomeratives bzw. divisives Clusterverfahren

Beim divisiven Clusterverfahren, dem sogenannten Top-down-Verfahren sind anfangs alle Daten demselben Cluster zugeordnet und werden dann schrittweise immer weiter verfeinert.
Das agglomerative Clusterverfahren arbeitet umgekehrt. Dort sind zu Beginn alle Datenpunkte in einem eigenen Cluster und werden schrittweise zusammengefasst.
In jedem Fall sind ein Distanzmaß, das die Distanz bzw. die Ähnlichkeit der einzelnen Cluster misst und eine Funktion, die zwei Cluster zusammenfasst bzw. ein Cluster aufteilt, nötig. Je nach Art der Objekte bzw. Darstellung der Cluster können die unterschiedlichsten Distanzmaße und Merge- bzw. Splitfunktionen verwendet werden.

Eine weitere große Gruppe stellen die partitionierenden Clusterverfahren dar. Diese Clusterverfahren starten mit einer fest vorgegebenen Anzahl von Clustern, denen in einem iterativen Prozess immer wieder neu Daten zugeordnet und daraufhin neu berechnet werden. Der bereits vorgestellte EM-Algorithmus kann zu dieser Klasse der Clusterverfahren gezählt werden.
Ein weiterer sehr bekannter Vertreter ist der bereits erwähnte k-Means-Algorithmus. Dieser basiert auf folgenden vier Schritten:

1. Initialisierung: (zufällige) Auswahl von k Clusterzentren
2. Zuordnung: Jedes Objekt wird dem ihm am nächsten liegenden Clusterzentrum zugeordnet
3. Neuberechnung: Es werden für jeden Cluster die Clusterzentren neu berechnet

4. Wiederholung: Falls sich nun die Zuordnung der Objekte ändert, weiter mit Schritt 2, sonst Abbruch

3. Audio-Segmentierung und Gruppierung

In diesem Kapitel wird die verwendete Methode zur Segmentierung und Gruppierung der Audiodaten genauer erklärt. Außerdem werden verschiedene Distanzmaße und Mergefunktionen beschrieben, die dabei zu Verfügung stehen.

3.1 Systemüberblick

Der Aufbau des Systems ist in Abbildung 3.1 dargestellt. Als Eingabe ist jedes beliebige Audiosignal geeignet, das Sprache, Musik, Geräusche und Stille enthalten kann. Dabei spielt es keine Rolle, wie viele Sprecher im Signal vertreten sind, unter welchen akustischen Bedingungen sie aufgezeichnet wurde oder ob sie sich mit Musik oder Geräuschen zersetzt bzw. hinterlegt ist. Die Vorverarbeitung entfernt möglichst alle Musik-, Geräusch- und Stillesegmente und zerteilt die verbleibenden Sprachsegmente in kürzere homogene Teile. Diese kurzen Sprachsegmente werden dann neu festgelegt und in Gruppen mit ähnlichen akustischen Bedingungen eingeteilt. Dazu werden die vorverarbeiteten Daten neu segmentiert und gruppiert und an ein ASR-System weitergegeben, das die einzelnen Gruppen verschriftet.

Abbildung 3.1: Überblick über die am Prozess der Spracherkennung beteiligten Systeme

- Die Vorsegmentierung besteht aus der Entfernung der Segmente, die aus Musik, Geräusche oder Stille bestehen und der anschließenden Unterteilung der übrigen Sprachsegmente in kürzere Segmente. Beim Entfernen von Nicht-Sprach-Segmenten sollte darauf geachtet werden, dass Sprache, Musik und Geräusche häufig nicht sauber von der sie umgebenden Sprache getrennt sind. Musik und Geräusche können im Hintergrund auftreten oder Sprache und Musik können sich in einem fließenden

Übergang abwechseln indem die Musik im Hintergrund immer lauter wird. Außerdem sollten die Sprachsegmente möglichst homogen sein. Das heißt, ein Segment sollte im Idealfall nur Sprache von einem Sprecher unter denselben akustischen Bedingungen enthalten. Außerdem dürfen die Segmente eine gewisse Mindestlänge nicht unterschreiten.

- Die von der Vorsegmentierung gelieferten Segmente, die häufig nicht ganz genau auf die Sprecherwechsel passen, werden hier genau ausgerichtet und entsprechend ähnlicher akustischer Bedingungen und Stimmcharakteristiken in Gruppen zusammengefasst.
- Das Spracherkennungssystem verschriftet die in Gruppen zusammengefassten Sprachsegmente und gibt den entsprechenden Text aus.

Wie in der Einleitung bereits erwähnt ist es sinnvoll Audiodaten zu segmentieren bevor mit einem Spracherkenner Wörter und Sätze erkannt werden. Welche Vorteile eine gute Segmentierung hat wird im Folgenden erläutert:

- Sprache auf mehreren kurzen Segmenten zu erkennen benötigt deutlich weniger Speicher und Rechenleistung als auf einem langen Segment. Vor allem ist die Suche einfacher, da der Rechenaufwand mit der Anzahl der Zustände des Graphen meist quadratisch wächst.
- Natürlich ist es auch von Vorteil wenn die Segmente nur ganze Sätze enthalten. Abgeschnittene Sätze oder Wörter, können leicht zu einer falschen Erkennung führen, da das Sprach- bzw Audiomodell auf ganzen Sätzen bzw. Wörtern trainiert wurde.
- Wenn in der Audiodatei mehrere verschiedene Sprecher enthalten sind z.B. bei Nachrichten, Talkshows usw. kann dadurch die Erkennungsrate deutlich sinken. Es gibt mehrere Möglichkeiten dieses Problem zu lösen. Die einfachste besteht darin, das ASR-System mit Trainingsdaten zu trainieren, die möglichst viele verschiedene Sprecher enthalten. So kann das ASR-System zwar für alle Sprecher Sprache erkennen, es leidet aber die Erkennungsrate. Man spricht von sogenannten sprecherunabhängigen Spracherkennern.
  Eine weitere Möglichkeit ist die Verwendung einer Segmentierung, die die Segmente, die vom selben Sprecher stammen, in Gruppen zusammenfasst. Auf den einzelnen Gruppen können dann entweder spezialisierte Spracherkenner angewendet werden, oder es kann eine Sprecheradaption mittels z.B. einer Vokaltraktnormalisierung durchgeführt werden. Vokaltraktnormalisierung ist ein Überbegriff für Ansätze, die den Einfluss verschiedener Sprecher auf die Merkmalsvektoren minimieren, wie z.B. erläutert in [ZJ91] oder [BE86]. Das führt in der Regel zu besseren Erkennungsergebnissen.

3.2 Vorverarbeitung

Ein möglicher Ansatz zur Entfernung von Nicht-Sprach-Segmenten besteht darin mehrere GMMs zu benutzen, die auf verschiedenen Arten von Audiodaten trainiert werden. Ein GMM wird z.B. mit einer Stunde beliebiger Sprache trainiert, eines mit Musik und eines mit beliebigen Geräuschen/Stille. Ähnlich wird in [GLA98] vorgegangen. Je nachdem welches der drei GMMs die größere Wahrscheinlichkeit aufweist, können Abschnitte in unbekannten Audiodaten in Sprach- oder Nicht-Sprachsegmente klassifiziert werden.
Es gibt auch die Möglichkeit eine Klassifikation aufgrund bestimmter Merkmale durchzuführen, die direkt vom Signal abgeleitet werden. Dazu gehören z.B. die Energie oder die Anzahl der Nulldurchgänge (siehe [Rea10]). Dieser Ansatz basiert darauf, dass ein digitalisiertes Audiosignal je nach Art des Klangs unterschiedliche Charakteristiken aufweist. So hat menschliche Sprache eine, über die Zeit hinweg, deutlich stärkere Schwankung der Amplitude als z.B. weißes Rauschen.

Als nächstes müssen die übrigen Sprachsegmente in mehrere kurze Segmente mit einer gewissen Mindestlänge unterteilt werden. Wichtig dabei ist, dass die einzelnen kurzen Segmente möglichst rein sind, d.h. es sollte pro Segment möglichst nur ein Sprecher vertreten sein. Wenn die Segmente zu lang sind besteht eine erhöhte Wahrscheinlichkeit, dass das Segment mehr als nur einen Sprecher enthält. Sind die Segmente jedoch zu kurz enthalten sie zu wenig Merkmalsvektoren um ein GMM hinreichend stabil trainieren zu können. Es muss also ein Kompromiss gefunden werden.

Häufig wird mit einem sogenannten sliding Window und einem Distanzmaß gearbeitet, das den Unterschied zwischen zwei Wahrscheinlichkeitsverteilungen berechnen kann, z.B. mit der Kullback-Leibler-Distanz. Dabei wird ein Fenster mit einer bestimmten Länge x, z.B. 2s und einem bestimmten Versatz y, z.B. 1s über das Segment geschoben und jeweils die Wahrscheinlichkeitsverteilung berechnet. Dann wird mit der Kullback-Leibler-Distanz je die Distanz zum Vorgängerfenster berechnet. So erhält man eine Reihe von Funktionswerten und das Segment kann z.B. an allen Maxima geteilt werden. Ein solches Verfahren wird in [SJRS97] beschrieben.

Ein anderes Verfahren stützt sich auf die Annahme, dass zwischen Sprecher- oder Satzwechseln längere Pausen auftreten als zwischen zwei Wörtern innerhalb eines Satzes. Es werden dann alle beliebig kurzen (evtl. nur Bruchteile einer Sekunde lange) Pausen bzw. Stilleabschnitte erkannt und das Sprachsegment an den längsten Pausen unterteilt.

Um die ersten beiden Schritte umzusetzen kann auch ein bereits existierender allgemeiner Spracherkenner benutzt werden. Es wird eine Transkription der gesamten Audiodatei erstellt und anhand der Abschnitte, für die Wörter erkannt wurden, und des Abstandes zwischen den einzelnen Wörtern können die Sprach- und Nicht-Sprachsegmente und die Sprecherwechsel relativ zuverlässig erkannt werden. Ein allgemeiner Spracherkenner hat aber meist eine schlechtere Erkennungsrate als ein spezialisierter und so können einzelne Wörter leicht falsch erkannt werden. Das ist in diesem Fall jedoch unwichtig, da die Bedeutung der einzelnen Wörter hier keine Rolle spielt.

3.3 Clusterverfahren

Zur Segmentierung und Gruppierung wird ein agglomeratives Clusterverfahren verwendet wie es in [GLA98] vorgestellt wird. Die Cluster werden dabei durch GMMs dargestellt und nach jedem Durchgang werden mit dem Viterbi-Algorithmus die Segmente neu ausgerichtet.

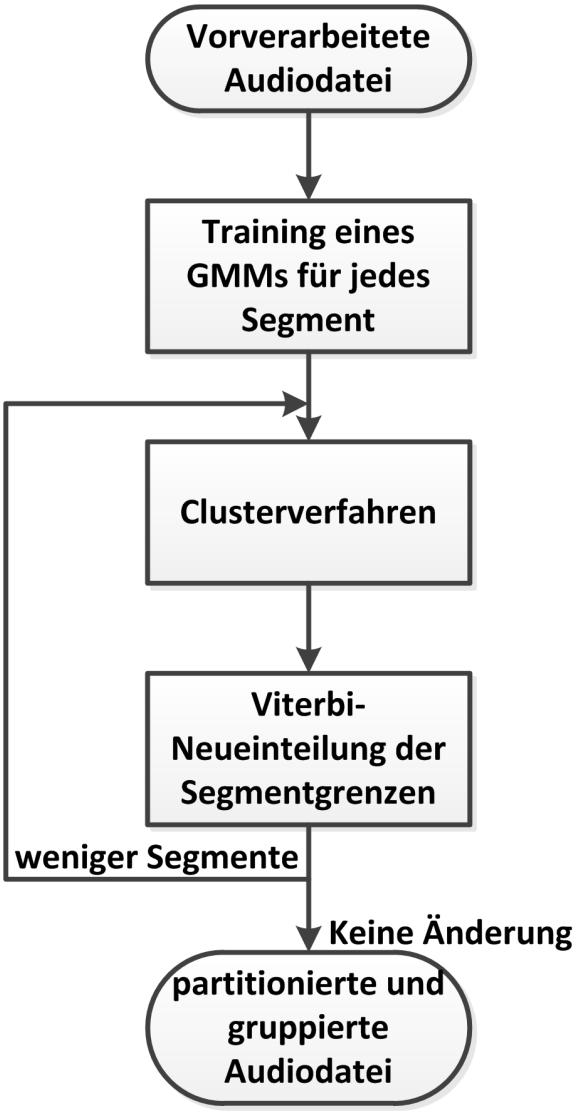

Abbildung 3.2: Schematische Darstellung eines Clusterverfahrens

Das Clusterverfahren gruppiert in einem iterativen Prozess die einzelnen Segmente schrittweise und schätzt die Grenzen der Segmente mit dem Viterbi-Algorithmus jedes Mal neu. Dazu wird zuerst für jedes Segment ein GMM trainiert. Im folgenden iterativen Prozess wird eine Distanz, die angibt wie ähnlich sich zwei GMMs sind, zwischen allen möglichen GMM-Paaren berechnet und alle GMMs, deren Distanz unterhalb eines bestimmten Schwellwertes liegt, werden zu einem GMM zusammengefasst. Danach werden die Grenzen der Segmente mit dem Viterbi-Algorithmus neu berechnet. Dieser Vorgang wird so lange wiederholt bis alle Distanzen größer als ein Schwellwert sind.

Das Clusterverfahren besteht aus folgenden Schritten.

1. Dieser Schritt trainiert für jedes Segment ein GMM. Welche Möglichkeiten es gibt um ein GMM zu trainieren wird in Kapitel 2.2.3 erläutert. Ein Sprachsegment enthält

grundsätzlich zwei unterschiedliche Informationen. Das sind zum einen die akustischen Bedingungen, wie z.B. Hintergrundgeräusche, Nachhall oder Sprecherstimme und zum anderen die Folge der gesprochenen Laute bzw. Wörter. Da die Folge der gesprochenen Laute bzw. Wörter pro Segment als unterschiedlich angenommen werden kann, werden die Wahrscheinlichkeitsdichteverteilungen der Merkmalsvektoren zweier Segmente, deren akustische Bedingungen gleich oder ähnlich sind, ähnlicher sein als die zweier Segmente, deren akustische Bedingungen sich stark unterscheiden.

2. Im nächsten Schritt werden alle GMM-Paare, deren Unterschied, bezogen auf ein bestimmtes Distanzmaß, kleiner als ein festgelegt Schwellwert ist, zu einem GMM kombiniert. Die möglichen Distanzmaße und Mergefunktionen werden in Kapitel 3.4 vorgestellt.

3. Alle GMMs werden nun in der Reihenfolge der Segmente in ein Links-Rechts-HMM eingefügt und die Grenzen der Segmente mit dem Viterbi-Algorithmus neu festgelegt.

Die Schritte 1-3 werden so lange wiederholt bis im zweiten Schritt keine GMM-Paare mehr gefunden werden können.

3.4 Distanzmaße und Mergefunktionen

Welche Distanzmaße im Clusterverfahren verwendet werden ist abhängig von der Art der Daten und vom Modell das der Cluster darstellt. Wie in der Spracherkennung üblich werden Cepstralkoeffizienten und GMMs verwendet. Ein Distanzmaß sollte also den Unterschied zwischen zwei Wahrscheinlichkeitsdichteverteilungen messen.

Es gibt mehrere Distanzmaße die in Frage kommen.

- Ein möglicher Ansatz zur Berechnung der Distanz wird in [GLA98] vorgestellt. Der Unterschied zweier Normalverteilungen kann, wie in [KOR94, GSR91] vorgestellt, einfach berechnet werden. Dabei wird das Verhältnis zwischen der Wahrscheinlichkeit, dass die Daten von einer Normalverteilung generiert wurden und der Wahrscheinlichkeit, dass die Daten von zwei verschiedenen Normalverteilungen generiert wurden berechnet. Oder anders ausgedrückt, es wird berechnet wie gut zwei Datensätze, die von zwei verschiedenen Normalverteilungen generiert wurden, zu einer Normalverteilung passen würden, die auf der Grundlage beider Datensätze berechnet wurde. Zur Berechnung sind nur die Parameter (Kovarianz und Mittelwertsvektor) nötig. Um dieses Verfahren nun auf zwei GMMs anzuwenden, von denen jedes z.B. 8 Normalverteilungen enthält, kann man ein agglomeratives Clusterverfahren auf der Ebene der Normalverteilungen durchführen, bis von den anfangs 16 Normalverteilungen nur noch 8 vorhanden sind. Zwei Normalverteilungen können zu einer kombiniert werden indem das der Anzahl der jeweils zugrundeliegenden Merkmalsvektoren entsprechend gewichtete Mittel der Parameter berechnet wird. Der berechnete "Verlust" der Wahrscheinlichkeit in jedem Schritt muss aufaddiert werden.

- Es besteht auch die Möglichkeit den Unterschied zwischen zwei Wahrscheinlichkeitsdichteverteilungen mit der symmetrischen Kullback-Leibler-Divergenz direkt zu berechnen (siehe [KL51]).

$$KL(P,Q) = \sum_{x \in X} P(x) \cdot log \frac{P(x)}{Q(x)} \tag{3.1}$$

$$SKL(P,Q) = \frac{KL(P,Q) + KL(Q,P)}{2} \tag{3.2}$$

KL bezeichnet die diskrete Berechnung der Kullback-Leibler-Divergenz für die Wahrscheinlichkeitsdichteverteilungen P und Q. SKL ist die symmetrische Kullback-Leibler-Divergenz.

- Es kann auch ein Distanzmaß verwendet werden, das mit dem Funktionswert der Normalverteilungen der GMMs an der Stelle der jeweiligen Datenpunkte rechnet. Um die Distanz zwischen Cluster bzw. GMM A und B zu berechnen kann der Durchschnitt der Wahrscheinlichkeit aller Merkmalsvektoren aller Segmente die zum Cluster A gehören auf Cluster B berechnet werden und umgekehrt.

$$dist(A,B) = (score_B(A) - score_A(A)) + (score_A(B) - score_B(B)) \tag{3.3}$$

$$score_A(B) = -\ln(lh_A(B)) \tag{3.4}$$

$dist(A,B)$ definiert die Distanz zwischen Cluster A und B. Die Distanz von GMM A bzw. B zu sich selbst wird subtrahiert um Schwankungen der Distanz auszugleichen, die auf Daten zurückzuführen sind, die wenig Normalverteilt sind. $lh_A(B)$ ist der Durchschnitt der Wahrscheinlichkeit aller Merkmalsvektoren in Cluster B auf GMM A.

Um zwei Normalverteilungen zu kombinieren gibt es auch mehrere Möglichkeiten.

- Ein Ansatz wurde bereits bei den Distanzmaßen vorgestellt. Der Unterschied besteht lediglich darin, dass der während des agglomerativen Clusterverfahrens anfallende "Verlust" der Wahrscheinlichkeit nicht aufsummiert werden muss, da nur das Ergebnis des Clusterverfahrens interessant ist.
- Es ist natürlich auch möglich ein GMM auf den Daten der beiden zum kombinierenden GMMs, mit Hilfe des in Kapitel 2.2.3 vorgestellten EM-Algorithmus neu zu trainieren.
- Das GMM kann auch mit Hilfe von UBMs in jedem Durchgang neu adaptiert werden.

3.5 Janus

Das Janus Recognition Toolkit (JRTk) ist ein in C++ geschriebenes Framework, das eine umfangreiche Funktionalität bietet um aktuelle ASR-Systeme erstellen zu können. Es wurde am Karlsruhe Institut für Technologie und der Carnegie Mellon University in Pittsburgh entwickelt.

Das JRTk ist objektorientiert und bietet Klassen an, die alle nötigen Funktionen für ein ASR-System abdecken. Mit der Skriptsprache Tcl/Tk [tcl, Ous94] können alle Funktionen und Objekte angesprochen und so ein genauer Ablauf festgelegt werden. Der integrierte IBIS Decoder [SMFW01] durchsucht den Suchraum, bestehend aus allen möglichen Wortkombinationen, sehr effizient nach der besten Hypothese.
Für alle durchgeführten Experimente und implementierten Abläufe und Funktionalitäten wurde das JRTk verwendet.

4. Evaluation

In diesem Kapitel wird vorgestellt wie und unter welchen Bedingungen die verschiedenen Distanzmaße und Mergefunktionen evaluiert wurden und welche dabei am besten abschnitten. Außerdem wird der beste Ansatz mit einer manuell erstellten Ground Truth und mit dem Baseline-System verglichen.

4.1 Testdaten

Die Testdaten bestehen aus Audioaufnahmen elf verschiedener TV-Sendungen, die aus Nachrichtensendungen und Talkshows zusammengesetzt sind. Die Testdaten haben eine Gesamtlaufzeit von ca. 330 Minuten. Sie enthalten verschiedene männliche und weibliche Sprecher, unterschiedlich viele Sprecher pro Datei, die sowohl einmal als auch mehrmals enthalten sind, spontane Sprache z.B. bei Interviews und sowohl Geräusche als auch Musik. Zu jeder Datei existiert eine sogenannte Ground Truth, die aus einer manuellen Clustereinteilung besteht. D.h. es wurden manuell die Bereiche markiert an denen die Sprache eines Sprechers zu hören ist und die Sprache desselben Sprechers enthalten.

4.2 Vorgaben und Parameter

Bei der Entwicklung eines Systems zur Segmentierung und Gruppierung von Audiodaten im Rahmen dieser Arbeit sind verschiedene Vorgaben gegeben. Das sind zum einen die Nachbarkomponenten Vorsegmentierung und ASR-System und zum anderen die grundsätzliche Funktionsweise die auf einem agglomerativen Clusterverfahren mit einer Viterbi-Neueinschätzung der Segmentgrenze beruht. Die variablen Parameter die evaluiert wurden werden im Folgenden aufgezählt.

- Variation des Clusterverfahrens: Die Vorgabe besteht zwar darin, dass es sich wie in [GLA98] vorgegeben um ein agglomeratives Clusterverfahren mit einer Viterbi-Neueinschätzung der Segmentgrenzen nach jedem Schritt handeln soll, es ist aber nicht angegeben wie ein solcher Schritt genau definiert ist. Es existieren zwei naheliegende Möglichkeiten. Es können in jedem Schritt die beiden ähnlichsten Segmente vereint werden, deren Distanz unterhalb des Schwellwertes liegt. Danach wird der Viterbi-Algorithmus durchgeführt und es werden alle Distanzen erneut berechnet.

Im nächsten Schritt werden dann wieder die beiden ähnlichsten Segmente vereint usw. Die andere Möglichkeit besteht darin in jedem Schritt alle Segmentpaare, deren Distanz unterhalb des Schwellwertes liegt und deren Segmente nicht schon mit anderen Segmenten vereint worden sind, zu vereinen. Bei der zweiten Möglichkeit wird angenommen, dass sich die Lage der Segmentpaare, die nicht am ähnlichsten sind, deren Distanz aber trotzdem unterhalb des Schwellwertes liegt, durch das Vereinen des ähnlichsten Segmentpaares nicht wesentlich ändern würde. Die zweite Möglichkeit sollte jedoch weniger Rechenkapazität benötigen, da der Viterbi-Algorithmus und das Neuberechnen aller Distanzen seltener durchgeführt werden muss. Ob dies tatsächlich der Fall ist werden die Experimente zeigen.

- Mindestlänge der Segmente: Um die Parameter der GMMs hinreichend stabil trainieren zu können, ist eine bestimmte minimale Menge an Daten nötig, die zum Training zur Verfügung stehen. Da im Initialschritt für jedes Segment ein GMM trainiert wird, muss die Mindestlänge der Segmente, die von der Vorsegmentierung geliefert werden, bestimmt werden.

- Normalverteilungen pro GMM: Die Anzahl der Normalverteilungen für die GMMs können frei gewählt werden. Je mehr Normalverteilungen gewählt werden, umso detaillierter kann eine Wahrscheinlichkeitsdichteverteilung angenähert werden. Es steigt jedoch auch die Menge der benötigten Trainingsdaten und das Risiko zum Übertraining. Grundsätzlich gilt: So viele Normalverteilungen wie nötig, aber so wenige wie möglich.

- Deltas und Deltadeltas: Bei der Vorverarbeitung der Daten (Kapitel 2.2.2) gibt es zwar viele verschiedene Parameter (z.B. Fenstergröße und -versatz, Fensterart, Anzahl der Mel-Skala-Filter, ...), die meisten fließen in die Evaluation jedoch nicht mit ein, da sich im Laufe der Zeit Standardwerte durchgesetzt haben. Es muss jedoch getestet werden ob die Verwendung von Deltas und Deltadeltas Verbesserungen der Ergebnisse ergeben. Dabei handelt es sich um die erste und zweite Ableitung der einzelnen Dimensionen, die als weitere Dimensionen an den Merkmalsvektor angehängt werden. Die Verwendung von Deltas und Deltadeltas kann in manchen Fällen zu einer Verbesserung der Performance führen.

- Schwellwert: Der Schwellwert, der angibt ab wann das agglomerative Clusterverfahren abgebrochen wird, kann beliebig gewählt werden und ist direkt abhängig vom verwendeten Distanzmaß. Ein niedriger Schwellwert würde zu mehreren Clustern führen, deren Segmente sehr ähnlich zueinander wären. Ein hoher Schwellwert hingegen führt zu weniger Clustern, deren Segmente zueinander weniger ähnlich wären. Es ist schwierig einzuschätzen ob ein hoher Schwellwert nun besser oder schlechter als ein niedriger ist.

- Distanzmaße: Es müssen alle in Kapitel 3.4 vorgestellten Distanzmaße getestet werden. Es handelt sich dabei um die Kullback-Leibler-Distanz, die Distanz, die den Wahrscheinlichkeitsverlust auf Ebene der Normalverteilungen berechnet und die Distanz, die die durchschnittliche Wahrscheinlichkeit der Segmente auf dem jeweils anderen GMM berechnet.

- Mergefunktionen: Auch hier gibt es verschiedene Möglichkeiten die in Kapitel 3.4 vorgestellt wurden. Es handelt sich um die in [GLA98] vorgestellte Mergefunktion,

die die Normalverteilungen in einem hierarchischen Clusterverfahren kombiniert, die zwei Standardtrainingsverfahren auf Basis des EM-Algorithmus und eine Adaption durch UBMs.

4.3 Evaluationsverfahren

Da es sich bei der Segmentierung und Gruppierung von Audiodaten nur um einen Vorverarbeitungsschritt eines ASR-Systems handelt, ist lediglich die Leistung des ASR-Systems ausschlaggebend um die Clustereinteilungen untereinander zu vergleichen.

Um die Leistung eines ASR-Systems zu messen hat sich die Wortfehlerrate (word error rate (WER)) durchgesetzt. Das Ziel dabei ist es die Gleichheit der vom ASR-System ausgegebenen Wortsequenz und einer gegebenen Referenzwortsequenz zu messen. Die Wortsequenzen können dabei aus unterschiedlichen Wörtern verschiedener Reihenfolge und abweichender Länge bestehen. Die WER wird aus der Anzahl der Ersetzungen, Löschungen und Einfügungen von Wörtern berechnet, die nötig ist um aus der Wortsequenz des ASR-Systems die Referenzwortsequenz zu machen, geteilt durch die Anzahl der Wörter der Referenzwortsequenz.

$$\text{WER} = \frac{\#\text{Löschungen} + \#\text{Einfügungen} + \#\text{Ersetzungen}}{\#\text{Wörter in der Referenzwortsequenz}} \tag{4.1}$$

Die WER kann mit dem Prinzip der dynamischen Programmierung effizient berechnet werden.

Die ideale Vorgehensweise bei der Evaluation wäre es, für jede mögliche Parameterkombination die Testdaten zu segmentieren und zu gruppieren, die resultierenden Cluster an das ASR-System weiterzugeben und die WER zu vergleichen. Aufgrund der großen Anzahl der Parameterkombinationen (ca. 200) und der Tatsache, dass ein vollständiger Testlauf des ASR-Systems ca 24 Stunden dauert, wäre dies jedoch zu aufwändig. So wurden zur Evaluation Methoden verwendet um Untergruppen von Parametern einzeln zu testen und so verschiedene Kombinationen frühzeitig ausschließen zu können.

Zur Evaluation der Vorverarbeitung und des Distanzmaßes werden die Testdaten mit Hilfe der Vorsegmentierung in kurze Sprachsegmente zerlegt. Danach werden alle möglichen Kombinationen von je zwei Segmenten gebildet. Von jedem Paar aus zwei Segmenten ist aufgrund der Ground Truth bekannt, ob es zum selben Sprecher gehört oder nicht. Außerdem wird die Distanz aller Paare berechnet und anhand eines Schwellwertes festgelegt ob das Paar laut dem Distanzmaß zum selben Sprecher gehört. Aus diesen Daten wird die Sensitivität und die Falsch-Positiv-Rate berechnet und die Raten aller Testdateien werden gemittelt.

$$\text{Sensitivität} = \frac{r_p}{r_p + f_n} \tag{4.2}$$

$$\text{Falsch-Positiv-Rate} = \frac{f_p}{r_n + f_p} \tag{4.3}$$

r_p: Anzahl der Fälle in denen das Paar vom selben Sprecher stammt und das auch so erkannt wird.
r_n: Anzahl der Fälle in denen das Paar nicht vom selben Sprecher stammt und das auch

so erkannt wird.
f_n: Anzahl der Fälle in denen das Paar vom selben Sprecher stammt, fälschlich jedoch als Paar von verschiedenen Sprechern erkannt wird.
f_p: Anzahl der Fälle in denen das Paar nicht vom selben Sprecher stammt, fälschlich jedoch als Paar vom selben Sprechern erkannt wird.

Die einzelnen Schwellwerte können als Punkte in eine Grenzwertoptimierungskurve (Receiver Operating Characteristic (ROC-Kurve)) eingetragen werden. Sensitivität und Falsch-Positiv-Rate bilden die beiden Achsen der ROC-Kurve. So kann die Qualität der Distanzmaße gut beurteilt und verglichen werden. Außerdem kann ein grober Richtwert für den Schwellwert im Clusterverfahren ermittelt werden.

Es ist wichtig die Qualität des Ergebnisses der Segmentierung und Partitionierung auch ohne das ASR-System beurteilen zu können. Dazu wurde die Cluster Purity und Cluster Coverage aus [HJJ+98] verwendet, jedoch auf Frame-Ebene berechnet. Dabei wird die Clustereinteilung mit einer manuell erstellten Ground Truth verglichen. Daraus können zwar keine direkten Rückschlüsse auf die WER gezogen werden, die bei einem Testdurchgang des ASR-Systems gemessen würden, da z.B. zwei verschiedene Sprecher mit einer sehr ähnlichen Stimme die sich im selben Cluster befinden, verglichen mit der Ground Truth, falsch wären, die WER des ASR-Systems davon aber durchaus profitieren könnte. Mehrere Clustereinteilungen können damit jedoch objektiv und einfach miteinander verglichen werden.
Die Cluster Purity definiert wie viel Prozent eines Clusters, der im Idealfall nur Segmente des selben Sprechers enthalten sollte, von dem Sprecher abgedeckt wird, der in dem Cluster am häufigsten enthalten ist. Ein Cluster der z.B. zu einem Prozent aus Segmenten des Sprechers A und zu 99 Prozent aus Segmenten des Sprechers B besteht hätte einen höheren Cluster Purity-Wert als ein Cluster bei dem die Sprecher A und B mit 40 und 60 Prozent verteilt wären. Die Cluster Coverage ist genau das Gegenteil. Sie definiert auf wie viele Cluster und zu welchen Teilen die Sprache eines Sprechers aufgeteilt ist. Cluster Purity und Coverage werden relativ zur manuell erstellten Ground Truth berechnet.

$$purity(M, G) = \frac{1}{N} \sum_{k} \max_{j} |m_k \cap g_j| \tag{4.4}$$

wobei $M = \{m_0, m_1, ..., m_K\}$ die Menge aller zu messender Cluster und $G = \{g_0, g_1, ..., g_J\}$ die Menge aller Cluster der Ground Truth ist. Jeder Cluster besteht aus Segmenten und jedes Segment aus Frames. N ist die Summe der Länge aller Cluster in M. Die Länge eines Clusters ist die Anzahl aller Frames aller enthaltenen Segmente.

$$coverage(M, G) = purity(G, M) \tag{4.5}$$

um die Cluster Coverage zu berechnen müssen beide Mengen einfach vertauscht werden.

4.4 Experimente

Die ersten Experimente zeigen welche Kombination aus Vorverarbeitung und Distanzmaß am sinnvollsten ist. Dazu wird, wie in Kapitel 4.3 erläutert, eine ROC-Kurve berechnet, die die Fehlerrate der Klassifikation von Segmentpaaren in die Klassen selber Sprecher und anderer Sprecher unter verschiedenen Schwellwerten darstellt. Bei zwei verschiedenen Möglichkeiten der Vorverarbeitung (ohne Deltas und Deltadeltas (16 Dimensionen) und mit Deltas und Deltadeltas (47 Dimensionen)) und drei verschiedenen Distanzmaßen (Wahrscheinlichkeit der Frames auf GMMs,Kullback-Leibler-Distanz und Wahrscheinlichkeitsverlust auf NV Ebene [GLA98]) ergeben sich sechs Kombinationsmöglichkeiten.

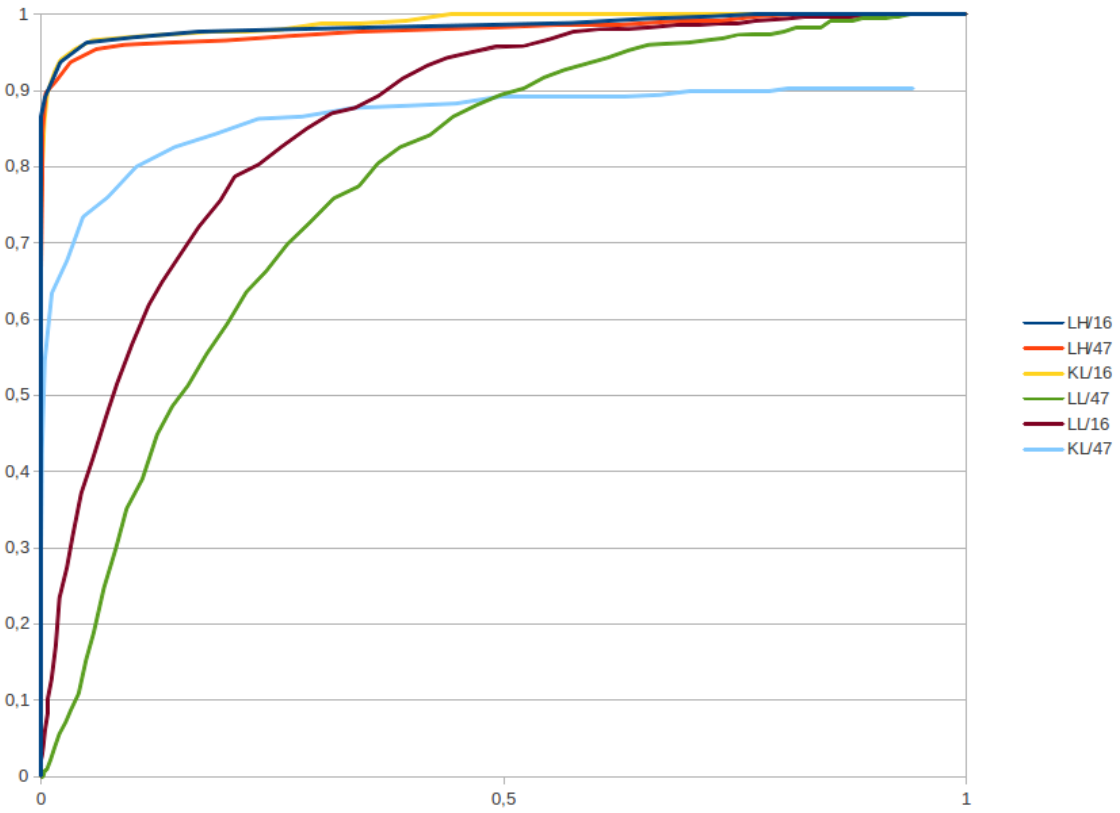

Abbildung 4.1: Erstes Experiment, das Distanzmaße und Vorverarbeitungsvarianten miteinander vergleicht

Wie in Abbildung 4.1 zu sehen ist gibt es erhebliche Unterschiede zwischen den einzelnen Distanzmaßen. Je weiter sich eine ROC-Kurve der linken obere Ecke bzw. dem Punkt (0,1) annähert umso mehr Segmentpaare wurden korrekt klassifiziert. LL steht für likelihood loss und bezeichnet das Distanzmaß, das den Verlust der Wahrscheinlichkeit auf Ebene der Normalverteilungen berechnet. LH steht für likelihood und bezeichnet die Berechnung der mittleren Wahrscheinlichkeit der Frames auf den GMMs. KL steht für das Kullback-Leibler-Distanzmaß. 16 bzw. 47 steht für die Anzahl der Dimensionen und bezeichnet somit die Vorverarbeitung mit und ohne Deltas und Deltadeltas. Die Verwendung von Deltas und Deltadeltas liefert immer schlechtere Ergebnisse. Die besten Ergebnisse werden vom LH/16 und KL/16-Verfahren erzeugt. Da das LH/16-Verfahren den geringeren Rechenaufwand erfordert wurde in allen weiteren Experimenten diese Vorverarbeitungs- und Distanzmaß-Kombination verwendet.

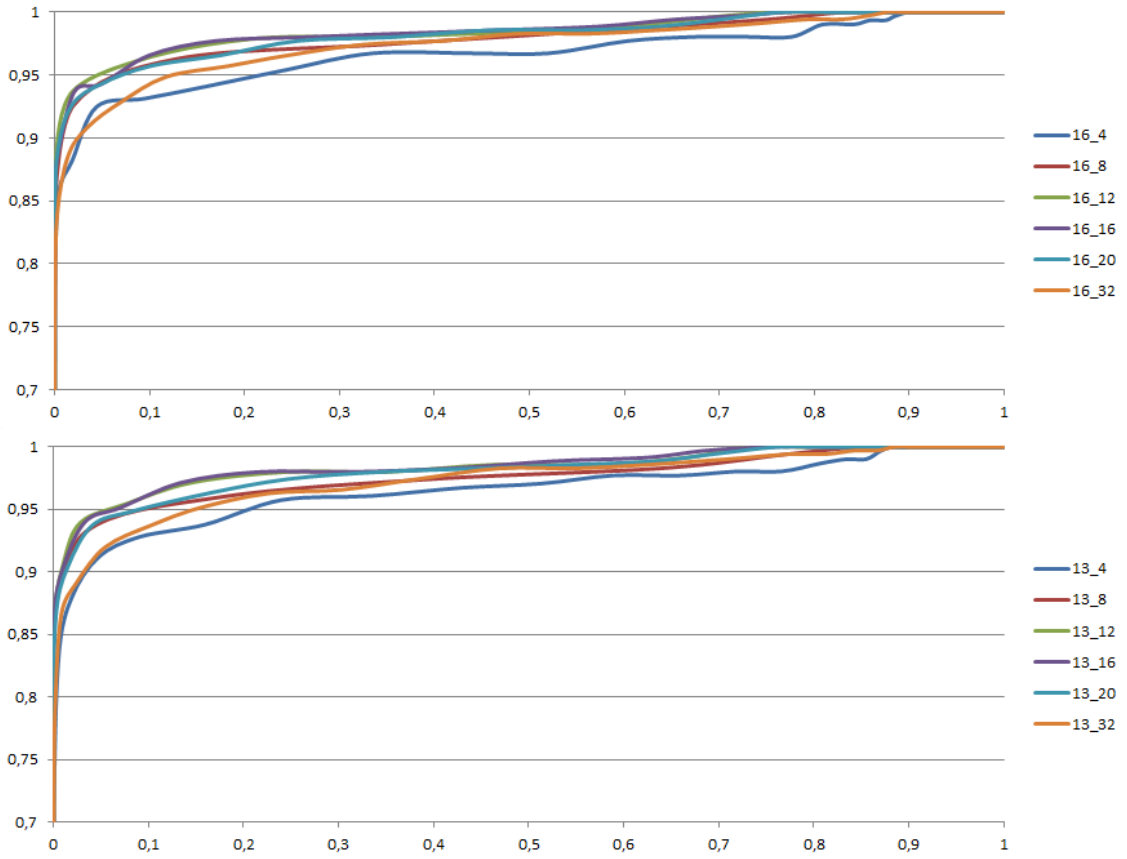

Abbildung 4.2: Vergleich von 13 und 16 Dimensionen bei unterschiedlich vielen Normalverteilungen

Abbildung 4.2 zeigt die ROC-Kurven bei 13- und 16-dimensionaler Vorverarbeitung für 4, 8, 12, 16, 20 und 32 Normalverteilungen pro GMM.

	13 Dimensionen	16 Dimensionen
4	10,00522775	8,710769755
8	7,677884896	7,59214752
12	6,660915116	6,355894118
16	6,845457977	6,623170618
20	7,485313576	7,484301054
32	9,683315212	9,769285728

Tabelle 4.1: Fehlerrate der besten Schwellwerte für verschieden viele Normalverteilungen

Die Anzahl der Dimensionen macht keinen großen Unterschied, jedoch liefern 16 Dimensionen häufig die etwas besseren Ergebnisse. Bei der Wahl von 4 Normalverteilungen kann die Wahrscheinlichkeitsdichteverteilung nicht detailliert genug abgebildet werden und 32 Normalverteilungen bieten zu viele Parameter, sodass das GMM zum übertrainieren neigt. So sinkt in beiden Fällen die Performance. Es werden 8 Normalverteilungen gewählt (so wenig wie möglich).

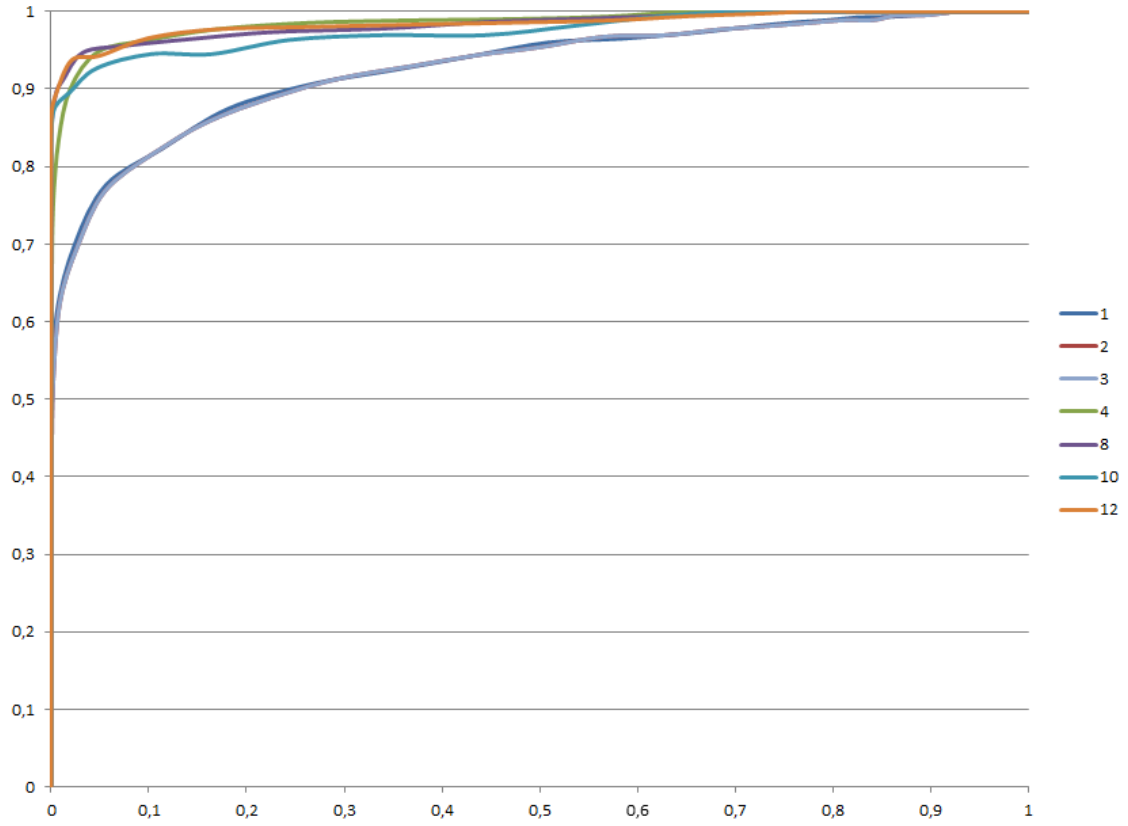

Abbildung 4.3: Vergleich verschiedener Mindestlängen der Segmente

Verglichen werden verschiedene Vorsegmentierungsvarianten, die Segmente mit einer Mindestlänge von 1, 2, 3, 4, 8, 10 oder 12 Sekunden liefern. Wie zu sehen ist, fällt die Erkennungsrate bei Mindestlängen von weniger als 4 Sekunden stark ab. Deshalb sollten die Segmente mindestens 4 Sekunden lang sein.

Bei den nächsten Experimenten wurde der Segmentierungs- und Gruppierungsprozess durchgeführt und anschließend die Cluster Coverage und Cluster Purity der Ergebnisse berechnet. Dabei wurden die drei Mergefunktionen (Mergefunktion nach [GLA98] (LL), Neutraining (Train) und Adaption durch UBMs (UBM)) unter verschiedenen Schwellwerten getestet. Außerdem wurde getestet inwiefern sich die Ergebnisse der beiden Variationen des Clusterverfahrens voneinander unterscheiden.

	LL		Train		UBM	
	Purity	Coverage	Purity	Coverage	Purity	Coverage
1,8	88,68	28,86	88,71	33,56	87,59	89,47
2	88,75	39,15	88,75	41,21	86,55	91,90
2,2	88,76	40,21	88,76	42,52	-	-
2,4	88,75	45,66	88,75	52,98	75,04	93,46
2,5	-	-	-	-	75,04	93,46
2,6	88,75	49,31	88,73	58,67	66,69	93,88
3	89,39	64,54	89,03	71,56	42,50	95,24
3,2	89,38	68,01	89,06	72,08	-	-
3,4	89,38	70,75	89,04	72,13	-	-
3,6	89,77	72,52	80,34	74,35	-	-
4	89,98	76,08	73,62	79,51	-	-
4,5	47,10	90,67	49,71	89,25	-	-
5	45,24	92,35	48,26	89,68	-	-

Tabelle 4.2: Cluster Coverage und Purity für die drei verschiedenen Mergefunktionen bei unterschiedlichen Schwellwerten

Tabelle 4.2 zeigt die Ergebnisse in Cluster Coverage und Purity. Zwischen einem Neutraining und der Mergefunktion nach [GLA98] bestehen keine wesentlichen Unterschiede. Die Adaption mit Hilfe von UBMs liefert mit Abstand die besten Ergebnisse. Vor allem kann eine sehr hohe Cluster Coverage erreicht werden, ohne dass dadurch die Cluster Purity zu weit absinkt. Daher wurde dieses Verfahren für die abschließenden Experimente, bei denen die WER des ASR-Systems berechnet wird, gewählt.

	Var2 (Train)		Var1 (Train)	
	Purity	Coverage	Purity	Coverage
2	88,75	41,21	-	-
2,2	88,76	42,52	-	-
2,4	88,75	52,98	88,75	42,68
2,6	88,73	58,67	-	-
3,4	89,04	72,13	-	-
3,6	80,34	74,35	88,80	54,95
4	73,62	79,51	-	-
4,2	-	-	89,64	73,96
4,5	49,71	89,25	79,98	74,73

Tabelle 4.3: Cluster Coverage und Purity für die zwei verschiedenen Variationen des Clusterverfahrens bei unterschiedlichen Schwellwerten

Tabelle 4.3 zeigt die Ergebnisse der ersten Variation, bei der in jedem Schritt nur das ähnlichste Segmentpaar vereint wird, und der zweiten Variation, bei der in jedem Schritt alle ähnlichen Segmentpaare vereint werden. Da die Güte der Ergebnisse der beiden Clusterverfahren relativ zueinander unabhängig von der Güte des verwendeten Distanzmaßes und der verwendeten Mergefunktion sein sollte, wurden die Ergebnisse nur für eine Mergefunktion (Neutraining der Segmente) berechnet. Da sich die Ergebnisse kaum voneinander unterscheiden, wird aufgrund der geringeren Laufzeit die zweite Variation verwendet.

Die Experimente haben gezeigt, dass das Distanzmaß LH, eine Vorverarbeitung, die 16-dimensionale Merkmalsvektoren liefert, GMMs, die mit einer Adaption eines UMBs trainiert werden und eine Vorsegmentierung mit einer minimalen Segmentlänge von 4s die besten Ergebnisse liefern. Im letzten Experiment wird der Segmentierungs- und Gruppierungsprozess für verschiedene Schwellwerte durchlaufen und die auf Basis der resultierenden Cluster berechnete WER des ASR-Systems verglichen.

	WER (%)
Ground Truth	17,54
QClust	19,67
Schwellwert 1,8	19,68
Schwellwert 2,0	19,68
Schwellwert 2,2	19,64
Schwellwert 2,4	19,62
Schwellwert 3,0	20,03

Tabelle 4.4: Wortfehlerraten

Um die WERs bei unterschiedlichen Schwellwerten einschätzen zu können, sind in Tabelle 4.4 auch die WERs für die Ground Truth und die Baseline QClust eingetragen. Die WER der Ground Truth stellt den bestmöglichsten Wert dar, der bei einer absolut korrekten, idealen Segmentierung erreicht würde. Die WERs schwanken je nach Schwellwert um die WER von QClust. Der beste Wert wird für den Schwellwert 2,4 erzeugt, mit einer Verbesserung der WER von 0,05 gegenüber QClust.

4.5 Laufzeit

Obwohl es keine Vorgaben bezüglich der Laufzeit gibt, sollte diese dennoch beachtet werden. Es ist nicht vorhersehbar in welchen Anwendungen, das im Rahmen dieser Arbeit entwickelte Clusterverfahren verwendet werden wird und welche davon laufzeitkritisch sein werden. Daher sollte Rechenleistung nicht unnötig verschwendet werden.

Die Laufzeit ist im Wesentlichen von drei Faktoren abhängig. Es handelt sich um das Distanzmaß, die Mergefunktion und die Variation des Clusterverfahrens. Diese drei Faktoren werden im Folgenden näher betrachtet.

Im Initialisierungsschritt und nach jedem Clusterschritt müssen die Distanzen für alle Segmentpaare neu berechnet werden. Die Anzahl der möglichen Segmentpaare steigt mit der Anzahl der Segmente exponentiell. Da die Anzahl der Segmente, die von der Vorsegmentierung geliefert werden, kaum beeinflussbar ist, spielt nur die durchschnittliche Dauer zur Berechnung einer Distanz eine Rolle.

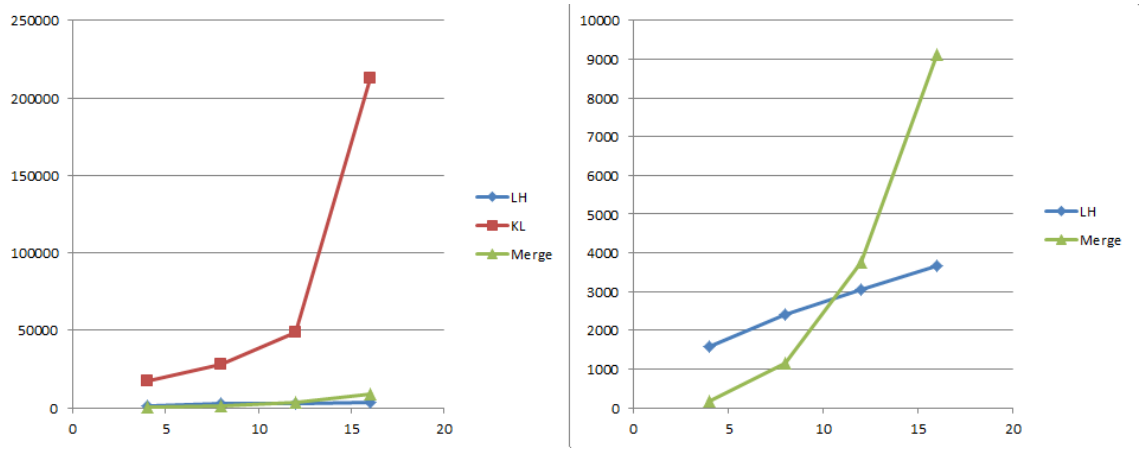

Abbildung 4.4: Vergleich der Dauer zur Berechnung einer Distanz für die drei Distanzmaße. Y-Achse: Dauer in Mikrosekunden, X-Achse: Anzahl der Normalverteilungen pro GMM

Wie zu sehen ist, ist das Berechnen der Kullback-Leibler-Distanz das mit Abstand aufwändigste bzw. langsamste Verfahren. Da der Gewinn an Rechenzeit bei dem Merge-Distanzmaß gegenüber dem LH-Distanzmaß bei weniger als ca. zehn Normalverteilungen nur sehr gering ausfällt, das Merge-Distanzmaß jedoch deutlich schlechtere Ergebnisse liefert, ist das LH-Distanzmaß die bessere Wahl. Davon abgesehen steigt beim LH-Distanzmaß die Rechendauer mit der Anzahl der Normalverteilungen nur linear.

Um die Laufzeit der beiden Variationen des Clusterverfahrens, unabhängig von verwendetem Distanzmaß und Mergefunktion, vergleichen zu könnnen, wird die Art und Anzahl der jeweils durchlaufenen Schritte verglichen. Dabei werden Schwellwerte verwendet, die zu ähnlichen Ergebnissen führen.

Schwellwert	Variation 1		Variation 2	
	Distanzen/Viterbi	Merge	Distanzen/Viterbi	Merge
3,6/2,4	23	22	8	21
4,2/3,4	39	38	7	32
4,5/3,6	42	41	7	34

Tabelle 4.5: Vergleich Anzahl der durchgeführten Distanzberechnungen, Viterbi-Algorithmen und Mergefunktionen für die zwei verschiedenen Variationen des Clusterverfahrens bei einer 15 Minuten Testaudiodatei mit initial 79 Segmenten

Wie Tabelle 4.3 zu entnehmen ist, produzieren die in Tabelle 4.5 gewählten Schwellwerte in etwa dieselben Ergebnisse, sodass der Aufwand vergleichbar ist. Bei ähnlich vielen Mergefunktionsaufrufen ist die Anzahl der durchgeführten Viterbi-Algorithmen und Distanzmaßberechnungen bei Variation zwei deutlich geringer. Daher sollte die zweite Variation verwendet werden.

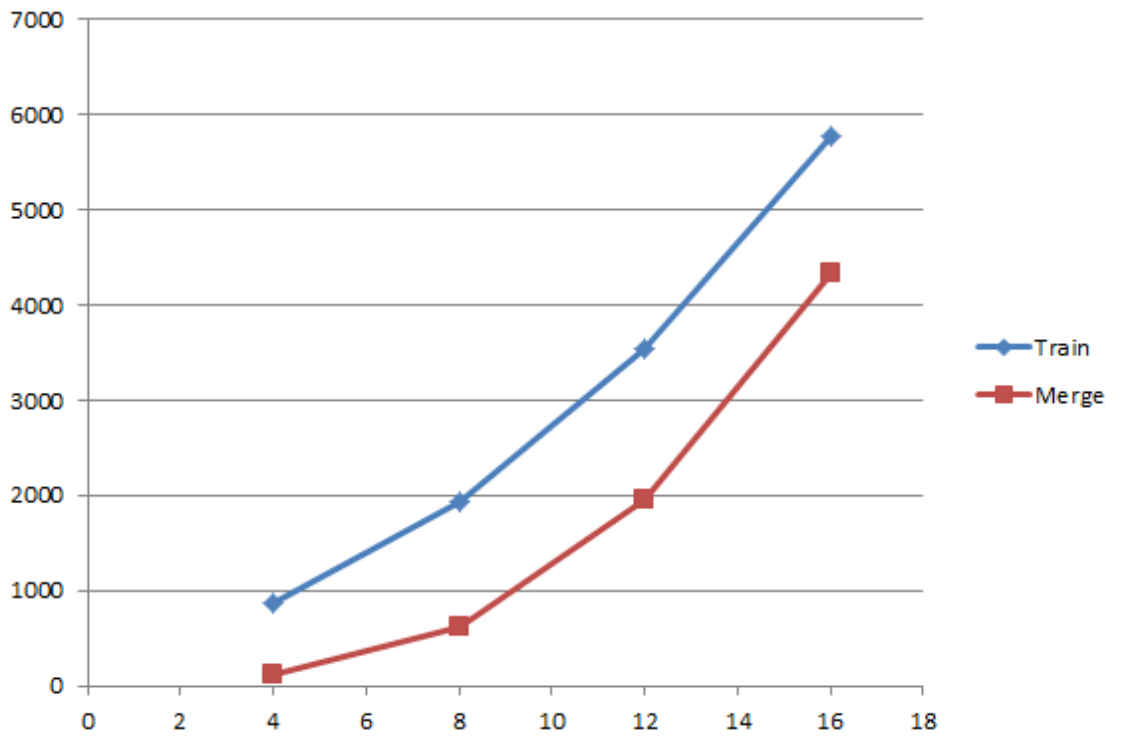

Abbildung 4.5: Vergleich der Laufzeit der Mergefunktion nach [GLA98] (Merge) und des Neutrainings (Train). Y-Achse: Dauer in Mikrosekunden, X-Achse: Anzahl der Normalverteilungen pro GMM

Wie zu sehen ist, steigt die Laufzeit mit der Anzahl von Normalverteilungen zwar exponentiell an, ist jedoch trotzdem sehr gering. Da die Anzahl der Mergefunktionsaufrufe nicht größer als die Anzahl der Segmente sein kann, und selbst eine 60-minütige Audiodatei, bei einer durchschnittlichen Segmentlänge von 15 Sekunden, nur ca. 240 Segmente hätte, fällt die Laufzeit der Mergefunktion bei der Gesamtlaufzeit des Clusterverfahrens kaum ins Gewicht.

UBM

Da es sich bei einem UBM um ein GMM mit 2048 Normalverteilungen handelt, hat ein GMM, das durch die Adaption des UBMs an ein bestimmtes Segment berechnet wird, auch 2048 Normalverteilungen. Eine so große Anzahl an Normalverteilungen erhöht die Dauer der Berechnung der Distanzen und der Mergefunktion erheblich.

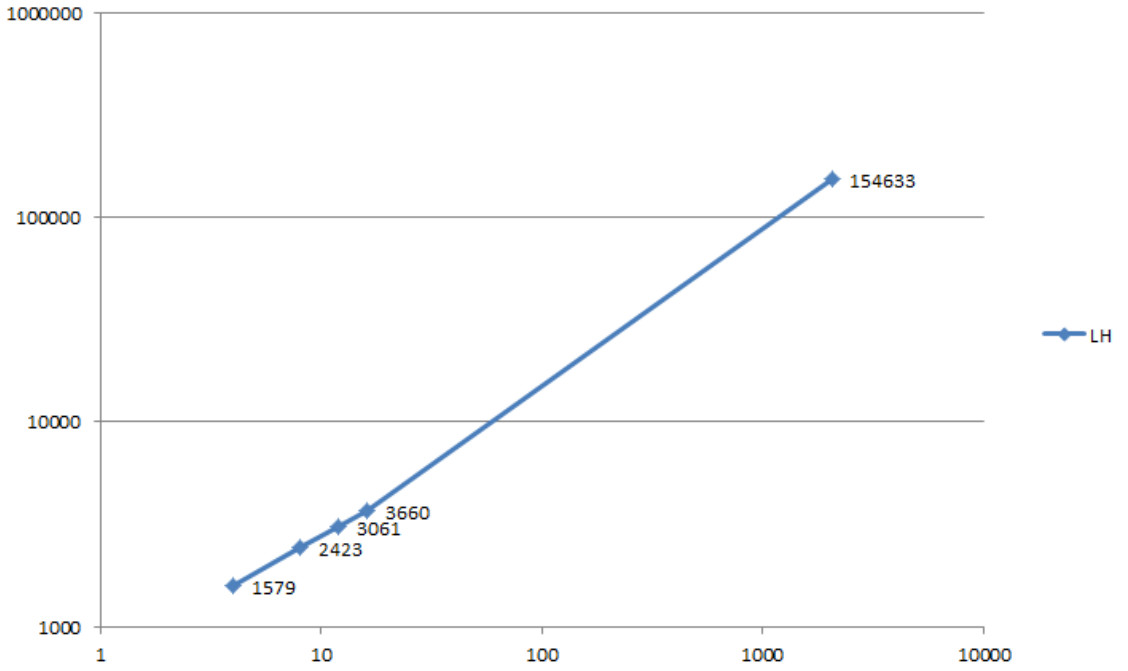

Abbildung 4.6: Durchschnittliche Dauer zur Berechnung einer Distanz. Y-Achse: Dauer in Mikrosekunden, X-Achse: Anzahl der Normalverteilungen pro GMM

Da nur das LH-Distanzmaß mit der Anzahl der Normalverteilungen linear steigt, ist nur dieses Distanzmaß für 2048 Normalverteilungen berechenbar. Jedoch sind 154633 Mikrosekunden pro Berechnung trotzdem relativ hoch.

	Train (16 NV)	Merge (16 NV)	Adaption UBM
Laufzeit (Mikrosekunden)	5771	4329	1491096

Tabelle 4.6: Vergleich der Laufzeiten der drei Mergefunktionen (NV: Normalverteilungen)

Die durchschnittliche Dauer um ein UBM an ein Segment zu adaptieren ist deutlich am höchsten. Jedoch läge selbst bei mehreren hundert Segmenten die Dauer aller benötigter Adaptionen im einstelligen Minutenbereich.

Unter Berücksichtigung der gemessenen Zeiten ist zu erkennen, dass es nur bei der Verwendung von UBMs zu Laufzeitproblemen kommen kann. Dabei hat die Berechnung der Distanzen den größten Einfluss auf die Laufzeit, da sie mit der Anzahl der Segmente exponentiell wächst. Alle anderen Einflussfaktoren wie z.B. Mergefunktion oder Viterbi-Algorithmus steigen entweder linear mit der Anzahl der Segmente oder haben nur eine Laufzeit im Bereich von wenigen Minuten.

5. Zusammenfassung

In den letzten Jahren wurden im Bereich der automatischen Spracherkennung große Fortschritte gemacht. Obwohl es noch viele Herausforderungen gibt, wie z.B. den Cocktailparty-Effekt, Hintergrundgeräusche, spontane oder undeutliche Sprache oder Dialekte, wurde das Problem der unterschiedlichen Sprecher bzw. Stimmen weitestgehend gelöst. Viele Audioaufnahmen wie z.B. Nachrichtensendungen, Talkshows oder Telefongespräche enthalten häufig die Stimmen verschiedener männlicher und weiblicher Sprecher, die sich zufällig abwechseln. Da das Segmentieren und Gruppieren der unterschiedlichen Sprecher mehrere Cluster mit je nur einem Sprecher erzeugt, dient die Sprechersegmentierung als wichtiger Vorverarbeitungsschritt für moderne Spracherkenner. So kann die Sprache in jedem Cluster separat erkannt werden, wodurch die Erkennungsleistung deutlich steigt. Es gibt bereits viele Ansätze zur Segmentierung und Gruppierung von Sprache in Audiodaten. Jedoch gibt es viel Spielraum zur Modifikation und Verbesserung der bestehenden Vorgehensweisen.

In dieser Arbeit wurde versucht den Ansatz aus [GLA98] zu implementieren und durch die Verwendung verschiedener Distanzmaße und Mergefunktionen die Qualität der Ergebnisse zu verbessern. Außerdem sollte ein bereits bestehendes Baseline-System zur Segmentierung und Gruppierung von Sprache ersetzt werden und dessen Leistung mindestens erreicht werden. Indem statt einer Funktion die zwei GMMs vereint oder ein GMM neu trainiert, angepasste UBMs verwendet wurden und eine eigene Distanzfunktion entwickelt wurde, konnten beide Anforderungen erfüllt werden. Wie die durchgeführten Experimente zeigen, konnte der Ansatz aus [GLA98] verbessert und auch die Leistung des Baseline-Systems erreicht werden.

Das im Rahmen dieser Arbeit entwickelte System zur Segmentierung und Gruppierung von verschiedenen Sprechern in Audiodaten soll das Problem der verschiedenen Stimmen für moderne Spracherkenner lösen. Um die in [GLA98] vorgestellte Vorgehensweise zu implementieren und die Leistung zu verbessern mussten verschiedene Parameter, Distanzmaße und Mergefunktionen getestet werden. Es wurden drei verschiedene Distanzmaße und drei verschiedene Mergefunktionen getestet. Als Distanzmaße wurde das in [GLA98] vorgestellte Maß, das auf einem agglomerativen Clusterverfahren auf Ebene der Normalverteilungen und dem Aufsummieren des Wahrscheinlichkeitsverlustes aus [KOR94] beruht, ein Maß, das die durchschnittliche Wahrscheinlichkeit der Frames auf den jeweils anderen GMMs berechnet und die Kullback-Leibler-Distanz getestet. Die getesteten Mergefunktionen waren die ebenfalls in [GLA98] vorgestellte Funktion, ein einfaches Neutraining

eines GMMs und die Adaption mittels eines UBMs. Außerdem war der genaue Ablauf des Clusterverfahrens, die ideale Anzahl der Dimensionen der Merkmalsvektoren und Anzahl der Normalverteilungen pro GMM, die Verwendung von Deltas und Deltadeltas und die Mindestlänge der vorsegmentierten Sprachsegmente unklar.

5.1 Diskussion

Im ersten Experiment wurde die beste Distanzfunktion bei einer Vorverarbeitung mit und ohne Deltas und Deltadeltas ermittelt. Dazu wurden beliebige Paare aus je zwei Sprachsegmenten gebildet, die entweder vom selben oder von verschiedenen Sprechern stammen, und diese anhand der Distanz und eines Schwellwertes klassifiziert. Die Sensitivität und Falsch-Positiv-Rate wurden bei variablem Schwellwert in eine ROC-Kurve eingetragen. Alle Distanzmaße erbrachten bei der Verwendung von Deltas und Deltadeltas eine schlechtere Leistung. Deltas und Deltadeltas erhöhen zwar die Menge der Informationen, weil die Merkmalsvektoren ebenfalls die erste und zweite Ableitung enthalten. Da die erste und zweite Ableitung jedoch für je alle Dimensionen berechnet werden muss, verdreifacht sich dadurch jedoch auch die Anzahl der Dimensionen. Dies ist bei ohnehin schon knappen Trainingsdaten kontraproduktiv. Die Kullback-Leibler-Distanz und die Distanz, die den Mittelwert der Wahrscheinlichkeiten berechnet, schnitten am besten ab.

Im nächsten Experiment wurden eine 13- und 16-dimensionale Vorverarbeitung mit einer unterschiedlichen Anzahl von Normalverteilungen pro GMM getestet. Die Anzahl der Dimensionen ergab nur einen geringen Unterschied, die Anzahl der Normalverteilungen pro GMM hingegen ist von größerer Bedeutung. Die größte Anzahl von Normalverteilungen mit 32 und die kleinste mit 4 schnitten am schlechtesten ab. Das ist dadurch zu erklären, dass bei 4 Normalverteilungen die Wahrscheinlichkeitsdichteverteilung nicht genau genug abgebildet werden kann und 32 Normalverteilungen bei zu wenig Trainingsdaten zum Übertraining neigen.

Der komplette Segmentierungs- und Gruppierungsprozess wurde mit den drei verschiedenen Mergefunktionen durchgeführt. Die Leistung wurde gemessen, indem die Cluster Coverage und Purity der Ergebnisse zu einer Ground Truth berechnet wurden. Die Mergefunktion aus [GLA98] wurde in [GLA98] nur als Ersatz für ein Neutraining verwendet, da ein Neutraining zur Zeit der Veröffentlichung von [GLA98] zu rechenintensiv gewesen wäre. Daher ist es nicht verwunderlich, dass ein Neutraining ebenso gute oder die besseren Ergebnisse liefert. Das haben die Experimente bestätigt. Die Verwendung von angepassten UBMs hingegen führte zu einer ganz erheblichen Verbesserung der Leistung. Ein GMM mit 8 Normalverteilungen und 16 Dimensionen besitzt insgesamt 264 Parameter (8 Normalverteilungen mit je 16 Parameter für den Mittelwertsvektor und 16 Parameter für die diagonale Kovarianzmatrix und einem Gewicht), denen im Training ein Wert zugewiesen werden muss. Ein Sprachsegment hingegen, das 4-10 Sekunden lang ist besitzt 400-1000 Merkmalsvektoren. Im Verlauf des Clusterverfahrens werden zwar immer mehr Segmente gruppiert, sodass die Menge der Trainingsdaten steigt, im Initialisierungsschritt jedoch muss pro Segment ein GMM trainiert werden. Dies sind zu wenige Vektoren um alle Parameter stabil trainieren zu können. Da UBMs genau diesem Problem entgegenwirken indem sie Vorwissen in Form von allgemeinen sprecherunabhängigen Normalverteilungen liefern sind diese Ergebnisse nicht überraschend.

Da bei der Verwendung von UBMs GMMs mit 2048 Normalverteilungen verwendet werden, erhöht sich die Laufzeit dadurch beträchtlich. Da das Distanzmaß, das den Mittelwert

der Wahrscheinlichkeiten berechnet, das einzige ist, dessen Berechnungsaufwand mit der Anzahl der Normalverteilungen nur linear steigt, ist dieses Distanzmaß die beste Wahl. Da eine Distanz für alle Segmentpaare berechnet werden muss, steigt der Aufwand der Berechnung mit der Anzahl der Segmente exponentiell. Dies führt dazu, dass bei laufzeitkritischen Anwendungen eine gewisse Maximallänge der Audiodateien eingehalten werden muss.

5.2 Ausblick

Obwohl die Wortfehlerraten, die das ASR-System in Verbindung mit dem im Rahmen dieser Arbeit entwickelten Segmentierungsprozesses liefert, nur minimal besser sind als die Wortfehlerraten in Verbindung mit dem Baseline-System, ist der entwickelte Prozess dennoch ein wichtiger Schritt, da er eine Basis für weitere Forschungen liefert. Da weder der Quellcode noch die genaue Funktionsweise des Baseline-Systems bekannt sind, kann es auch nicht verändert oder angepasst werden. Der im Rahmen dieser Arbeit entwickelte Prozess hingegen kann beliebig angepasst und verbessert werden, durch z.B. eine bessere Vorverarbeitung, Distanzmaße oder Mergefunktionen. Dadurch ist es bestens geeignet für eine künftige Weiterentwicklung und kann auch in Zukunft auf einem angemessenen Leistungsniveau gehalten werden.

Literaturverzeichnis

[aTFQD00] Douglas A. Reynolds ans Thomas F. Quatieri und Robert B. Dunn: *Speaker Verification Using Adapted Gaussian Mixture Models.* In: *Digital Signal Processing 10*, Seite 19, 2000.

[BE86] M. Blomberg und K. Elenius: *Nonlinear frequency warp for speech recognition.* In: *In Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, Seiten 2631–2634, 1986.

[BN08] Maximilian Bisani und Hermann Ney: *Joint-sequence models for grapheme-to-phoneme conversion.* In: *Speech Communication 50, Issue 5*, Seite 434, 2008.

[Buc96] Jan Constantin Buckow: *Klassifizierung und Erkennung von Sprachsegmenten.* Diplomarbeit, Universität Karlsruhe, 1996.

[DP93] P. Denes und E. Pinson: *The Speech Chain: The Physics and Biology of Spoken Language.* Worth Publishers, 1993.

[GLA98] Jean Luc Gauvain, Lori Lamel und Gilles Adda: *PARTITIONING AND TRANSCRIPTION OF BROADCAST NEWS DATA.* In: *ICSLP*, 1998.

[GSR91] H. Gish, M. Siu und R. Rohlicek: *Segregation of Speakers for Speech Recognition and Speaker Identification.* In: *ICASSP-91*, Seiten 873–876, May 1991.

[HAH01] Xuedong Huang, Alex Acero und Hsiao Wuen Hon: *Spoken Language Processing: A Guide to Theory, Algorithm and System Development.* Prentice Hall PTR, 2001.

[HJJ+98] Thomas Hain, Sue Johnson, S. E. Johnson, Andreas Tuerk, Steve Young, Philip Woodland und S. J. Young: *Segment Generation and Clustering in the HTK Broadcast News Transcription System.* In: *In Proceedings of the Broadcast News Transcription and Understanding Workshop*, Seiten 133–137, 1998.

[Hol91] John N. Holmes: *Sprachsynthese und Spracherkennung.* Oldenbourg, 1991.

[IPAa] *IPA-Vokalviereck.* `http://web.uvic.ca/ling/resources/ipa/charts/IPAlab/IPAlab.htm`. [Online; accessed 22-Dezember-2013].

[IPAb] *Liste aller IPA Zeichen.* `http://de.wikipedia.org/wiki/Liste_der_IPA-Zeichen`. [Online; accessed 22-Dezember-2013].

[Kam08] Karl Dirk Kammeyer: *Nachrichtenübertragung : mit ... 35 Tabellen.* Vieweg + Teubner, 2008.

[KK94] Ronald M. Kaplan und Martin Kay: *Regular models of phonological rule systems.* COMPUTATIONAL LINGUISTICS, 20:331–378, 1994.

[KL51] S. Kullback und R. A. Leibler: *On information and sufficiency.* Annals of Mathematical Statistics, 22, 1951.

[KL10] Tomi Kinnunen und Haizhou Lib: *An Overview of Text-Independent Speaker Recognition: from Features to Supervectors.* In: *Speech Communication 52, Issue 1*, Seite 12, 2010.

[KOR94] A. Kannan, M. Ostendorf und J.R. Rohlicek: *Maximum Likelihood Clustering of Gaussians for Speech Recognition.* In: *IEEE Trans. Speech and Audio*, Band 2, July 1994.

[Mac03] David J.C. MacKay: *Information Theory, Inference, and Learning Algorithms.* Cambridge University Press, 2003.

[Mit97] Tom M. Mitchell: *Machine Learning.* McGraw-Hill Science/Engineering/Math, 1997.

[O'S87] Douglas O'Shaughnessy: *Speech Communications: Human and Machine.* John Wiley Sons, 1987.

[Ous94] John K. Ousterhout: *Tcl and the Tk Toolkit*, 1994.

[Rea10] Bachu R.G. und et al.: *Voiced/Unvoiced Decision for Speech Signals Based on Zero-Crossing Rate and Energy*, 2010.

[RJ93] Lawrence Rabiner und Biing Hwang Juang: *Fundamentals of Speech Recognition.* Prentice Hall, 1993.

[Sea10] Tim Schlippe und et al.: *Wiktionary as a Source for Automatic Pronunciation Extraction*, 2010.

[SJRS97] Matthew A. Siegler, Uday Jain, Bhiksha Raj und Richard M. Stern: *Automatic Segmentation, Classification and Clustering of Broadcast News Audio.* In: *Proc. DARPA Speech Recognition Workshop*, Seiten 97–99, 1997.

[SMFW01] Hagen Soltau, Florian Metze, Christian Fügen und Alex Waibel: *A One-Pass Decoder Based on Polymorphic Linguistic Context Assignment*, 2001.

[ST95] Ernst Günter Schukat-Talamazzini: *Automatische Spracherkennung.* Friedr. Vieweg Sohn Verlagsgesellschaft, 1995.

[Ste94] Anthony Stentz: *Optimal and Efficient Path Planning for Partially-Known Environments.* In: *In Proceedings of the IEEE International Conference on Robotics and Automation*, Seiten 3310–3317, 1994.

[tcl] *Tcl/Tk.* `http://www.tcl.tk/`. [Online; accessed 22-Dezember-2013].

[WL90] Alexander Waibel und Kai Fu Lee: *Readings in Speech Recognition.* Morgan Kaufmann, 1990.

[ZJ91] S. A. Zahorian und A. J. Jagharghi.: *Speaker normalization of static and dynamic vowel spectra features.* In: *J. Acoust. Soc. Am.*, Seiten 67–75, 1991.