Topologieoptimierung phonetischer Hidden-Markov-Modelle mittels Erkennungssimulation

Dissertationsschrift
zur Erlangung des Grades
Doktor der Naturwissenschaften

am Department Informatik
der Universität Hamburg

vorgelegt von
DIRK KNOBLAUCH
Dipl. Physiker
aus Berlin

2006

Danksagung

An dieser Stelle möchte ich allen Personen und Institutionen danken, die die vorliegenden Arbeiten ermöglicht haben.

Mein besonderer Dank gilt:

Herrn Prof. Dr. Walther von Hahn für die Möglichkeit am Fachbereich Informatik der Universität Hamburg, meine wissenschaftliche Arbeit anfertigen zu können.

Meinem Betreuer und Gutachter, Herrn Prof. Dr. Ing. Wolfgang Menzel, der mir jederzeit für Diskussionen und Anregungen zur Verfügung stand.

Herrn Dr. Ing. habil. Gernot A. Fink für Erstellung des Zweitgutachtens.

Herrn Dr. Florian Schiel für anregende Diskussionen und ausführliche Auskünfte über das Verbmobil Korpus.

Herrn Michael Daum für die Bereitstellung und ständiger Wartung des Compute-Clusters, ohne den die Anfertigung dieser Arbeit nicht möglich gewesen wäre.

Herrn Kilian Foth für Diskussionen über Implementationsdetails.

Dirk Knoblauch
Berlin, den 5. Dezember 2005

Zusammenfassung

Eine Grundlage für die Entwicklung leistungsfähiger Spracherkennungssysteme ist die Modellierung geeigneter akustischer Modelltopologien. In der vorliegenden Arbeit wird ein Verfahren vorgestellt, welches verschiedene bestehende akustische monophone Modelltopologien anhand von Spracherkennungssimulationen qualitativ bewertet und somit Rückschlüsse auf einzelne Topologiegüten zulässt. Eine Spracherkennungs-Simulation erfolgt, indem theoretisch eine Dekodierung errechnet wird, jedoch ohne dem System Echtdaten zur Verfügung zu stellen. Durch die qualitative Bewertung wird eine akustische Topologieoptimierung möglich.

Als Grundlage für die Versuche wird ein Verfahren implementiert, welches anhand trainierter akustischer Modelle eine qualitative Bewertung der untersuchten Modelltopologien zulässt.

Die durch das Verfahren optimierten Topologien werden mit bekannten Topologien, die in der Spracherkennung eingesetzt werden, verglichen und diskutiert. Die Resultate zeigen, dass es möglich ist, akustische Modelltopologien qualitativ zu bewerten, ohne explizit Spracherkennungsergebnisse zu errechnen. Durch die optimierten Modelltopologien ist es schließlich möglich, die Trainiertheit einzelner Phonmodelle und damit mögliche Spracherkennungssysteme zu verbessern.

Inhaltsverzeichnis

Kapitel 1

Einleitung

Spracherkennung ist heutzutage in vielen Bereichen ein Bestandteil des alltäglichen Lebens geworden. In vielen Anwendungen stellen Spracherkennungssysteme eine einfache und wirkungsvolle Schnittstelle zwischen Mensch und technischer Umwelt dar. Die Spracherkennungstechnologie hat in den vergangenden Jahrzehnten insbesondere durch datengetriebene Verfahren große Fortschritte in der Erkennungsleistung erzielt. Zu diesen Verfahren zählen im wesentlichen das Training von Phon- bzw. Wortmodellen, das Training von Sprachmodellen, datengetriebene Clustering-Verfahren und Verfahren zur Modellierung geeigneter Aussprachevarianten eines Lexikons.

Durch die Möglichkeit der akustischen Topologiemodellierung, worunter Modellierungen der Zustandsanzahl und der Übergänge zwischen den Zuständen eines Hidden-Markov-Modells verstanden wird, lassen sich Erkennungsraten verbessern bzw. in Abhängigkeit der Modellierung untersuchen. Welche Topologie jedoch am geeignetsten ist, ein bestimmtes Spracherkennungsproblem optimal zu beschreiben, bleibt zunächst unklar, so dass zunächst auf einfache uniforme Topologien zurückgegriffen wird.

Ebenso kann Wissen über die Phonetik der Sprache zu Hilfe genommen werden, um einen Satz optimaler Modelle bereitzustellen, indem z.B. kurze lautliche Realisierungen mit wenigen Modellzuständen und längere lautliche Realisierungen durch mehr Modellzustände modelliert werden.

Eine Alternative hierzu ist, geeignete akustische Topologien datengetrieben zu ermitteln. So besteht die Möglichkeit, Topologien mit einer möglichst großen Variationsbreite einzusetzen, und während des Trainings unattraktive Übergänge zu eliminieren. Unter Topologievariationsbreite wird verstanden, wieviele verschiedene Topologien implizit in einem akustischen Phonmodell enthalten sind, wobei an dieser Stelle mögliche Zustandswiederholungen außer Acht gelassen werden.

Die sog. BAKIS-Topologie, die über drei linear verknüpfte Zustände verfügt, die ihrerseits einzeln übersprungen werden können, lässt z.B. eine datengetriebene Topologieoptimierung zu, indem die anfängliche maximale Variationsbreite während des Trainings schrittweise reduziert wird. Die Reduzierung erfolgt, indem Übergangswahrscheinlichkeiten von Zuständen, die einen gewählten Schwellenwert während des Trainings unterschreiten, eliminiert werden. Schließlich liegt die optimierte phonspezifische Topologie vor. Der unbestrittene Vorteil dieses Ansatzes ist, dass der Experimentator sich um die Topologieoptimierung keine Gedanken machen muss, jedoch nimmt der BAKIS-Ansatz keine Rücksicht auf eventuelle Datenknappheit und wird außerdem bei einer hohen Ausprachevariationsbreite die Topologievariationsbreite nicht wie gewünscht reduzieren. Ebenfalls zeigt sich in Vorexperimenten zu dieser Arbeit, dass eine triviale Variationsreduzierung anhand der Übergangswahrscheinlichkeiten zu keinen verbesserten Spracherkennungsergebnissen führt. Wäre es nun möglich, die Topologievariationsbreite einer trainierten BAKIS-Topologie nicht nur schrittweise zu reduzieren, sondern auch eine qualitative Aussage über die verbleibenden Topologievariationen zu treffen, könnte die qualitativ hochwertigste Topologie für weitere Experimente bzw. für einen Spracherkenner herangezogen werden.

Eine der bisher vernachlässigten Aufgaben ist die Fragestellung der qualitativen Beurteilung einzelner phonetischer Hidden-Markov-Modelle verschiedener Topologien ohne jedoch Erkennungsleistungen zu messen, um z.B. zwischen zwei Modellen für ein und dasselbe Phon die bessere Topologie auswählen zu können. Die Qualität eines phonetischen Hidden-Markov-Modells wird in

der vorliegenden Arbeit durch Simulation probabilistisch errechnet, indem eine idealisierte Beobachtungssequenz für dieses Modell angenommen wird. In dem Fall, dass zwei Topologien gleichzeitig auf denselben Daten trainiert werden, wird angenommen, dass die geeignetere Topologie diejenige ist, die für die idealisierte Beobachtungssequenz die höhere Produktionswahrscheinlichkeit erzeugt und somit die höhere Modellierungsgenauigkeit aufweist. Die vorliegende Arbeit beschäftigt sich daher hauptsächlich mit der Frage, wie die oben beschriebene Simulation mathematisch zu fassen und algorithmisch umzusetzen ist.

Vorgehensweise Die Grundlage der heutigen Topologiemodellierung von phonetischen Hidden-Markov-Modellen bilden akustisch motivierte Beobachtungen, um für die Spracherkennung geeignete Phonmodelle zu entwickeln. Die daraus folgenden phonetischen Modelle weisen auf Basis der phonetischen Feinstruktur z.B. eine dreiteilige Topologie auf. Sie besteht aus einer Anglitts-, einer quasi-stationären und einer Abglittsphase. So ist es nicht verwunderlich, dass sich als eines der am häufigsten eingesetzten Modelle das lineare links-rechts Modell mit drei emittierenden Zuständen zur phonetischen Modellierung etabliert hat.

Diese Wahl stellt jedoch möglicherweise nicht die optimale, sondern lediglich eine mögliche Topologie dar, Phone durch Modelle geeignet zu repräsentieren. Zusätzlich wird angenommen, dass alle Phone durch ein und dieselbe Topologie gleich gut beschrieben werden können, was vermutlich nicht optimal ist.

Um geeignete lautspezifische Topologien zu erhalten, kann auf Wissen über spezielle Lautcharakteristiken wie z.B. die Dauer zurückgegriffen werden. Das Trainingsverfahren gewährleistet jedoch nicht, dass diejenigen Daten, die unserer lautlichen Vorstellung entsprechen, auch tatsächlich durch die zugeordneten Phonmodelle konsumiert werden, da z.B. Phonsegmentgrenzen während des Trainings unbeachtet bleiben. Aus diesem Grund stützt sich die vorliegende Arbeit auf die These, dass eine datengetriebene Auswahl bzw. Optimierung besser geeignet ist.

Eine attraktive Möglichkeit der automatischen Topologiemodellierung stellt

die Initialisierung der einzelnen Phonmodelle durch eine allgemeine Topologie dar, welche auf einzelnen Zuständen Wiederholungen zulässt. Ebenfalls können alle Zustände des Hidden-Markov-Modells einzeln oder mehrfach übersprungen werden, was eine große Topologievariationsbreite zulässt. Während des Trainingsverfahrens werden die einzelnen Modelltopologien anhand der vorliegenden Sprachdaten modelliert. In Abhängigkeit von den Daten werden nun phonspezifische Topologievarianten im Modell bevorzugt, bis hin zur völligen Eliminierung bestimmter Varianten. Dieser Algorithmus arbeitet jedoch unabhängig von der Anzahl vorliegender Beobachtungen, was dazu führen kann, keine ausreichende Verallgemeinerung des Phonmodells zu erhalten. Sofort drängt sich der Gedanke des prunings einzelner Topologievariationen auf, die z.B. kleine Transitionswahrscheinlichkeiten aufweisen, um die übriggebliebene Topologievariationsbreite weiter zu reduzieren. Experimente dieser Art führen jedoch zu keiner wesentlichen Verbesserung.

Um dennoch eine qualifizierte datengetriebene Auswahl zu realisieren, werden für die einzelnen Phone des Inventars parallel unterschiedliche Topologien auf den vorliegenden Daten trainiert. Unter parallelem Training wird im Grunde, ähnlich dem Training des BAKIS-Modells, welches gleichzeitig mehrere Topologien beinhaltet, verstanden, zwei phonspezifische Topologien parallel aber unkorreliert innerhalb eines Modells gleichzeitig zu trainieren. Unter unkorrelierten Topologien wird verstanden, dass die Wahrscheinlichkeitsdichtefunktionen der einzelnen Modellzustände jeweils voneinander unabhängige Parameter aufweisen.

Um aus diesen Topologien die jeweils optimierte Topologie auszuwählen wäre es notwendig, systematisch Erkennungsergebnisse aller Topologiepermutationen zu errechnen. Dies ist in angemessener Zeit nicht realisierbar, so dass in dieser Arbeit ein heuristischer Ansatz zur Topologieauswahl vorgestellt wird, der sich der theoretischen Differenzierungsqualität der verschiedenen Topologien bedient.

Die Modellqualität einer parametrischen Verteilungsdichtefunktion äußert sich durch die Größe bzw. Breite ihrer Varianz, so dass Modelltopologien mit einer kleineren Gesamtvarianz zu bevorzugen sind.

In der vorliegenden Arbeit werden zunächst verschiedene Phontopologien parallel trainiert, um anschließend die qualitativ besser bewerteten zu jeweils einem optimierten Topologieinventar zusammenzufassen.

Mit den optimierten Modelltopologien werden anschließend drei Spracherkenner bis zur Erkennungskonvergenz trainiert. Anschließend werden die daraus resultierenden Spracherkennungsergebnisse mit den Ergebnissen der in dieser Arbeit verwendeten Standardtopologien verglichen.

Für den Vergleich mit den Ergebnissen aus dieser Arbeit werden typische Basistopologien herangezogen, die heutzutage sowohl in praktischen Anwendungen als auch in wissenschaftlichen Publikationen Verwendung finden. Hierzu gehören die einfache lineare links-rechts Topologie, die BAKIS-Topologie, ein phonspezifisches Topologieinventar und eine weitere reduzierte BAKIS-Topologie, die jedoch während der Initialisierung lediglich über zwei Topologievarianten verfügt. Die Referenztopologien werden in dieser Arbeit zum Zwecke des besseren Vergleichs erneut trainiert und evaluiert.

Die Untersuchungen werden auf dem spontansprachlichen Korpus Verbmobil Phase I und Phase II durchgeführt und bewertet. Das Korpus stellt eine besondere Herrausforderung dar, da Instanzen von schneller und undeutlich gesprochener Sprache aufgenommen und transkribiert wurden.

Aufbau der Arbeit: die vorliegende Forschungsarbeit zerfällt in zwei Teile. Im ersten Hauptteil werden die grundsätzlichen Verfahren und Ansätze stochastischer Spracherkennung eingeführt und erläutert.

Zu diesen gehören die Signalverarbeitung, die Klassifikation von Sprachmerkmalen insbesondere durch Hidden-Markov-Modelle, die akustisch motivierte Wortmodellierung und der Einsatz von Sprachmodellen in der Spracherkennung.

Der zweite Hauptteil stellt den wissenschaftlichen Ansatz und seine experimentelle Umsetzung dar.

In ihm wird zunächst der stochastische Ansatz zur Qualitätsbestimmung einzelner Phonmodelle heuristisch motiviert und mathematisch gefasst. Anschlie-

ßend wird die mathematische Gleichung in einen geeigneten Algorithmus umgeformt. Nach der Beschreibung des Algorithmus werden geeignete Experimente vorgestellt, mit denen die Eignung der vorgestellten Heuristik untermauert wird.

Der experimentelle Teil umfasst die Vorstellung der verschiedenen Experimentiersysteme. Hierzu gehört die Beschreibung der verschiedenen Referenzsysteme und die technischen Durchführung der Experimente. Schließlich werden die erzielten Ergebnisse ausgewertet und diskutiert. Im Anhang der Arbeit finden sich Korpusinformationen, das verwendete Phoninventar, Auszüge des Aussprachewörterbuchs, die Modellierung der Pausenmodelle sowie Informationen zur Signifikanzermittlung.

Kapitel 2

Stochastische Sprachverarbeitung

Stochastische Spracherkennung löste in den 80iger Jahren, die bis dahin bekannte abstandsbasierte Mustererkennung (DTW) ab und begann ihren triumphalen Siegeszug in diesem Anwendungsbereich der Informatik.

Die wesentlichen stochastischen Verfahren, die in den letzten Jahrzehnten zur Spracherkennung beigetragen haben, werden im Hinblick auf den in dieser Arbeit vorgestellten Ansatz zielgerichtet dargestellt und erörtert.

2.1 Signalverarbeitung und Merkmalsextraktion

Das zu untersuchende Sprachsignal liegt unverarbeitet in analoger Form vor. Es enthält daher alle möglichen Informationen, die für den späteren Klassifikationsprozess irrelevant sind. Hierzu gehören alle Signale die nicht zur Sprache gehören (Hall, Hintergrundgeräusche, usw.) Aus diesem Grund ist es nicht weiter verwunderlich, dass sich innerhalb der Signalverarbeitung ein eigener Forschungszweig etabliert hat. Im wesentlichen werden dort nicht nur grundsätzliche Probleme der Merkmalsextraktion, sondern insbesondere die Schwierigkeiten der Extraktion unter speziellen Randbedingungen, wie z.B. räumliche

Aufnahmen, Störgeräusche, unvollständige Signale, etc. diskutiert.

In den folgenden Paragraphen werden das für diese Arbeit verwendete Extraktionsverfahren und die daraus resultierenden Merkmalsvektoren erläutert. Zusätzlich werden typische Parameter zur Merkmalsgewinnung in den einzelnen Abschnitten angegeben, die heutzutage standardmäßig Verwendung finden und in der vorliegenden Arbeit ebenfalls verwendet werden.

Im darauf folgenden Abschnitt werden typische Klassifikatoren vorgestellt, die zunächst generell auf die Merkmale trainierbar sind und diese probabilistisch zusammenfassen. Anschließend wird der Begriff des Hidden-Markov-Modells eingeführt. Durch das Hidden-Markov-Modell lassen sich schließlich ganze Merkmalsequenzen zusammenfassen.

Quantisierung Für einen Klassifikator werden quantisierte Daten benötigt, die zunächst in einem unverarbeiteten Sprachsignal nicht vorliegen. Die durch das Signal gegebenen Amplitudenwerte werden mit Hilfe einer geeigneten Abtastperiode, in dieser Arbeit 16 kHz, quantisiert. Hierdurch entstehen, bei einer Auflösung von 16Bits/s, 256 kBit Daten pro Sekunde.

Frequenzspektrum Aus diesen Werten werden unter der Annahme der kurzzeitigen Stationarität durch die Kurzzeitanalyse (FT) (Fourier, 1822) Linienspektren erzeugt. Als Fensterfunktion wurde in dieser Arbeit das Hamming-Fenster mit einer Breite von 20ms und einem Vorschub von 10ms gewählt. In diesem Frequenzspektrum sind nun die für uns relevanten Frequenzen einer Beobachtung (z.B. eines Phons) enthalten, die sich zwischen 0 und ca. 8000Hz bewegen.

mel-Frequenzen Um aus diesen Spektren geeignete "charakteristische" Merkmale zu gewinnen, werden sie durch eine Dreiecksfilterbank, die sog. mel-Filterbank (Davis and Mermelstein, 1980), weiter reduziert. Mel-Filterbanken weisen üblicherweise 20-25 Frequenzgruppen auf und zeichnen sich durch die Simulation der menschlichen Hörverzerrung aus, indem die Filterbank mit zunehmenden Frequenzen in den Bandbreiten zunimmt.

Cepstrum Koeffizienten Das Sprachsignal kann als Überlagerung von Anregungssignal und Impulsantwort beschrieben werden(Fant, 1960). Das Signal liegt somit gefaltet vor. Um sich des für die Spracherkennung unwesentlichen Anregungssignals zu entledigen, welches das Spektrum verschmiert, werden schließlich die sog. *Cepstrum*-Koeffizienten der bereits ermittelten mel-Frequenzen m_j durch eine diskrete Kosinus-Transformation errechnet (Ahmed and Nasir, 1975) (Gleichung 2.1).

$$c_i = \frac{2}{N}\sum_{j=1}^{N} m_j \cos\left(\frac{\pi \cdot i}{N}(j - 1/2)\right) \tag{2.1}$$

In der vorliegenden Arbeit werden zwölf cepstrale Koeffizienten errechnet.

Delta-Koeffizienten Aus den cepstralen Koeffizienten werden zusätzlich noch ihre ersten beiden Ableitungen bestimmt, die in der Literatur als Delta-Koeffizienten bekannt sind (Furui, 1986). Durch die Delta-Koeffizienten wird der dynamische Verlauf in der Sprache modelliert. Zur Ermittlung der Delta-Koeffizienten wird auf das Verfahren der linearen Regression zurückgegriffen, welches mit Hilfe von drei Stützstellen die jeweilige Steigung approximiert.

Energie-Koeffizienten Letztendlich wird zusätzlich aus den cepstralen Koeffizienten ein Energiewert und seine Änderung, bez. Beschleunigung berechnet. Die Energie wird aus den cepstralen Koeffizienten ermittelt, indem der negative Logarithmus der Koeffizientensumme gebildet wird.

Merkmalsvektoren Die durch die oben beschriebenen Verfahren ermittelten Merkmalsvektoren verfügen über 39 Dimensionen und charakterisieren 10ms des Sprachsignals. Diese Art der Merkmalsextraktion ist heutzutage sehr häufig in Anwendungen zur Spracherkennung wiederzufinden. Die benötigten Merkmale werden mit Hilfe des Tools `HCopy` aus dem HTK Toolkit (Young et al., 2002) errechnet.

2.2 Klassifikation von Merkmalsvektoren

Die ermittelten Merkmalvektoren stellen Punkte im 39-dimensionalen Raum dar. Leider ist es nicht möglich, anhand der reinen Merkmalsvektoren ein Phon zu klassifizieren. Dies liegt daran, dass selbst gleiche Sprecher ein und dasselbe Wort unterschiedlich aussprechen können. Zusätzlich werden bei verschiedenen Sprechern die Merkmalspunkte aufgrund der Sprechercharakteristik noch gröber im Raum verteilt sein.

Um diesen Sachverhalt sinnvoll zu modellieren, werden alle Frames eines Phons zusammengefasst und durch sog. *Repräsentanten* dargestellt. In der Spracherkennung wird an dieser Stelle zwischen Ermittlung der Repräsentanten (auch Prototypen genannt) durch Vektorquantisierung und der Parameterinitialisierung kontinuierlicher Verteilungsdichten unterschieden.

Der in dieser Arbeit vorgestellte heuristische Ansatz zur Qualitätsbestimmung einzelner Modelltopologien lässt sich auf beide Arten von Klassifikatoren ohne weitere Probleme anwenden. Bei diskreten Verteilungen muss jedoch zusätzlich eine geeignete Varianzinformation vorliegen.

In dieser Arbeit werden Spracherkenner implementiert, die kontinuierliche Verteilungsdichtefunktionen verwenden. Es wird davon ausgegangen, dass der noch vorzustellende Simulationsalgorithmus in seiner Anwendung auf alle Arten von probabilistischen Verteilungen gleichermaßen anwendbar ist. Für dessen Herleitung und Evaluation reicht daher ein typischer Vertreter von Verteilungsfunktionen aus. Die Wahl von kontinuierlichen Dichteverteilungen ist darin begründet, dass diese Verteilungen aufgrund ihrer Parameter besonders angenehm in ihrer Handhabung erscheinen.

2.2.1 Klassifikation durch Vektorquantisierung

Das wesentliche Ziel der Vektorquantisierung ist Prototypen zu finden, die eine bestimmte Menge Merkmalsvektoren unter einem geeignetem Distanzmaß $d(\cdot, \cdot)$ in dem sog. Codebuch zusammenfasst. Das Klassifikationsverfahren arbeitet anschließend entweder distanzbasiert, z.B. durch Errechnung des eukli-

dischen Abstandes zum Prototyp, oder probabilistisch.

Der wesentliche Vorteil des sog. *diskreten Ansatzes* ist ohne Zweifel die durch die Gruppenbildung erzielte Reduktion der freien Parameter bzw. die Umwandlung des kontinuierlichen Signals in diskrete Observationsgruppen, die jeweils durch einen Prototypen charakterisiert sind. Die Klassenbildung erfolgt durch Vektorquantisierung (Gersho and Cuperman, 1983, Juang et al., 1982), die durch ein geeignetes Abstandsmaß $d(\cdot, \cdot)$, eine zuvor definierte Anzahl von Gruppen füllt und damit das sog. Codebuch generiert. Die Gruppenbildung hat zur Folge, dass handliche Modelle entwickelt werden können, die durch ihre schnelle Performanz hervorstechen. Dieser Vorteil wird jedoch durch eine oftmals zu scharfe Klassentrennung erkauft. Scharfe Gruppentrennung wird zwar auch durch Verteilungsdichtefunktionen erreicht, jedoch liegen im Gegensatz zur Gruppentrennung noch probabilistische Informationen über das fehlklassifizierte Merkmal vor, die in die Gesamtwahrscheinlichkeit der Klassifikation einfließen können. Klassifizierung durch Gruppenbildung führt daher u.U. zu schlechteren Erkennungsergebnissen. Umfassend wird das Thema Vektorquantisierung in (Gersho and Gray, 1992) diskutiert. Eine Zusammenfassung findet sich unter anderem bei (Huang et al., 2001).

Eine Alternative zum rein diskreten Ansatz stellen parametrische Verteilungsdichtefunktionen dar.

2.2.2 Parametrische Verteilungsdichtefunktionen

Parametrische Verteilungsdichtefunktionen (Mergel and Ney, 1985, Zhao et al., 1991) erlauben, die Klassenzugehörigkeit probabilistisch zu errechnen. Nicht nur die Tatsache, dass die Klassifikationsräume durch wenige Parameter beschrieben werden können, sondern auch, dass in Überlappungsgebieten zunächst keine eindeutige Entscheidung getroffen werden muss, zählen zu den großen Vorteilen. Dies wird dadurch begründet, dass während der Klassifikation von Sprache lokale Klassenzugehörigkeit weniger wichtig ist als die Auffindung von Pfaden im Merkmalsraum.

So kann auch in Überlappungsgebieten durch den Dichtewert des Merkmalsvektors vorerst eine Zugehörigkeitswahrscheinlichkeit ermittelt werden, die

dann zu einer Gesamtwahrscheinlichkeit eines noch zu definierenden Hidden-Markov-Modells führen wird. Zusätzlich können mehrere Verteilungen durch Linearkombination zusammengefasst werden, was zu einer besseren Raumapproximationen führen wird.

Üblich sind hierfür heutzutage Gaußsche Verteilungsdichten, die folgend motiviert werden (Schukat-Talamazzini, 1995a). Falls sich das kontinuierliche Signal hinreichend durch eine Normal- oder Gaußverteilung modellieren lassen würde, könnte man einfach

$$b_j(\vec{x}) = \mathcal{N}(\vec{x}|\vec{\mu}_{jk}, \mathbf{\Sigma}_{jk}) \tag{2.2}$$

annehmen. In 2.2 sind $\vec{x}$ der Merkmalsvektor, $\vec{\mu}_{jk}$ der errechnete Mittelwertsvektor und $\mathbf{\Sigma}_{jk}$ die inverse Kovarianzmatrix. Leider handelt es sich beim Sprachsignal nicht um einfach modale Observablen, so dass eine Linearkombination aus theoretisch unendlich vielen Verteilungsdichten zur Modellierung herangezogen werden muss, um eine möglichst genaue Approximation zu erhalten. Da aber nur ein begrenzter Vorrat an Daten zur Verfügung steht, wird die Linearkombination aus endlich vielen Funktionen gebildet, die durch Koeffizienten c_{jk} gewichtet werden (Yakowitz, 1970).

$$b_j(\vec{x}) = \sum_{k=1}^{\infty} c_{jk}\mathcal{N}(\vec{x}|\vec{\mu}_{jk}, \mathbf{\Sigma}_{jk}) \approx \sum_{k=1}^{M} c_{jk}\mathcal{N}(\vec{x}|\vec{\mu}_{jk}, \mathbf{\Sigma}_{jk}) \tag{2.3}$$

mit

$$\sum_{k=1}^{K} c_{jk} = 1 \quad und \quad c_{jk} \geq 0 \tag{2.4}$$

Kontinuierliche Dichteverteilungen finden auch heute noch, insbesondere aufgrund ihrer angenehmen Handhabung, weite Verbreitung in der Spracherkennung. Alternativen wären die Klassifikation durch neuronale Netze oder Support-Vector-Machines, auf die in dieser Arbeit nicht weiter eingegangen wird.

In einigen Fällen sind Richterverteilungen (Richter, 1986) von Vorteil. Richterverteilungen zeichnet sich dadurch aus, dass innerhalb einer Linearkombination der Mittelwertvektor $\vec{\mu}$ identisch bleibt und nur die Gewichtungsfaktoren c der jeweiligen Verteilung modelliert werden. Dadurch kann das Abklingverhalten der Verteilungsdichtefunktionen besser modelliert werden. Dies wirkt sich positiv bei den Modellen aus, bei den wenige Daten zur Verfügung stehen.

Der unbestrittene Vorteil, der sich durch die Modellierung mittels Verteilungsdichten ergibt, ist, dass eine solche Funktion zunächst durch zwei Kernparameter, nämlich Mittelwertvektor $\vec{\mu}$ und Varianz $\mathbf{\Sigma}$, geschätzt werden kann. Dieser Vorteil wird jedoch bei zunehmender Anzahl von Verteilungsdichten abgeschwächt. Die Parameteranzahl erhöht sich dramatisch gegenüber diskreten Ansätzen. Dennoch hat dieser Modellierungsansatz aufgrund seiner unendlichen Raumausdehnung gegenüber der diskreten Klassenbildung den unschätzbaren Vorteil, auch während des Trainings nicht gesehene Beobachtungen, später noch berechnen zu können, da jederzeit ein Wahrscheinlichkeitsdichtewert der Beobachtung errechenbar ist.

Um die Anzahl der zu lernenden Parameter zu reduzieren, kann ein globaler Vorrat an Gaußschen Komponentendichten $g_k(\vec{x})$ in einem Codebuch zusammengefasst werden. Die Raummodellierung erfolgt anschließend nur noch über die Modellierung der Gewichte c_{jk}.

Gleichung 2.3 lässt sich dann folgendermaßen umformen.

$$b_j(\vec{x}) = \sum_{k=1}^{M} c_{jk} g_k(\vec{x}) \tag{2.5}$$

Diese Hidden-Markov-Modelle werden in der Literatur als *semi-kontinuierliche Modelle* bezeichnet und wurden von (Huang and Jack, 1989, Huang et al., 1990) entwickelt. Sie stellen das Bindeglied zwischen diskreten und kontinuierlichen Modellen dar. Die nicht-probabilistische harte Klassenzuordnung wird durch globale Komponentendichten, bei gleichzeitiger Reduktion der zu lernenden Parameter, aufgehoben. Diese Art der Modellierung wird heutzutage als der Standard in der Spracherkennung angesehen.

Der Nachteil der bisher behandelten Klassifikationsvarianten ist jedoch, dass

sie nur statische Klassifikationsentscheidungen treffen können. Da das vorliegende Sprachproblem jedoch einen dynamischen Prozess darstellt, wird auf Hidden-Markov-Modelle zurückgegriffen, die dynamische Prozesse modellieren können. Im anschließenden Abschnitt wird gesondert auf die Klassifikation mit Markov-Ketten und das Training der Markov-Modelle eingegangen.

2.3 Hidden-Markov-Modelle in der stochastischen Sprachverarbeitung

Hidden-Markov-Modelle, im folgenden mit HMM abgekürzt, sind doppelt stochastische Automaten, die bei gegebenen Verteilungsdichten einem beliebig langen Merkmalsstrom eine probabilistische Bewertung zuteilen können. Daher eignen sich HMMs besonders gut, um Sprachsignale, die als unidirektionaler, dynamischer Fluss von Merkmalen vorliegen, zu modellieren.

2.3.1 Definition

In der einschlägigen Literatur (Lee, 1989, Huang et al., 2001, Jelinek, 1997, Wendemuth, 2004, Schukat-Talamazzini, 1995a) finden sich folgende Definitionen zum HMM-Formalismus, die hier zusammenfassend dargestellt werden:

Zunächst sei

$$\mathcal{Q} = \{s_1 \dots s_N\} \tag{2.6}$$

eine endliche Menge von Zuständen s_i, und

$$q = q_1 \dots q_T \qquad q_t \in \mathcal{Q} \tag{2.7}$$

eine Folge von Zuständen q_t.

In einem einfachen kausalen und stationären Prozess, wobei hier im wesentlichen die Abhängigkeit der Zustände q_t von vorherigen Zuständen q_{t-1} von Bedeutung ist, erhält die Übergangswahrscheinlichkeit die folgende Form

$$P(q_t|q_1 \dots q_{t-1}) \approx P(q_t|q_{t-1}) \tag{2.8}$$

und kann als $N \times M$ Matrix $\mathbf{A}$ dargestellt werden.

$$\begin{aligned} \mathbf{A} &= a_{ij} \\ mit \quad a_{ij} &= P(q_t = s_j|q_{t-1} = s_i) \\ i = \{1 \dots N\} \quad und \quad j &= \{1 \dots M\} \end{aligned} \tag{2.9}$$

Weiterhin wird der Vektor $\vec{\pi}$

$$\vec{\pi} := \pi_i = P(q_1 = s_i) \tag{2.10}$$

definiert, der die möglichen Anfangszustände mit ihren Eintrittswahrscheinlichkeiten formuliert.

Das Markov-Modell, das eine Folge von Zuständen stochastisch beschreibt, ist damit bereits definiert. Sollen die Zustände über ihre Folge hinaus noch Symbole als Folge darstellen, ist es notwendig, jedem Zustand eine Symbolverteilung b_j zuzuordnen. Es wird im Augenblick bewusst von einer *Verteilung* gesprochen, da diese je nach Aufgabenstellung variieren kann.

Die Verteilungen repräsentieren dabei ein endliches Ausgabealphabet $\mathcal{K}$

$$\mathcal{K} = \{v_i \dots v_K\} \tag{2.11}$$

Eine stochastisch erzeugte Folge von Symbolen v_i aus dem Vorrat $\mathcal{K}$ sei

$$O = O_1 \dots O_T \tag{2.12}$$

Der Weg, diese diskrete Symbolfolge durch eine Folge von q_t zu produzieren, bleibt versteckt. Die Produktionwahrscheinlichkeit der Symbole lässt sich als

$$P(O_t|q_t) \tag{2.13}$$

angeben. Die Emissionverteilungen lassen sich nun zu

$$\begin{aligned} b_{jk} &= b_j(v_k) \\ &= P(O_t = v_k | q_t = s_j) \end{aligned} \tag{2.14}$$

schreiben.

Aus stochastischen Normierungsgründen muss

$$\sum_j a_{ij} = 1, \quad mit \quad a_{ij} \geq 0 \tag{2.15}$$

$$\sum_i \pi_i = 1, \quad mit \quad \pi_i \geq 0 \tag{2.16}$$

$$\sum_j b_{jk} = 1, \quad mit \quad b_{jk} \geq 0 \tag{2.17}$$

gefordert werden.

Der gesamte doppelt stochastische Prozess ist somit durch den Parametersatz

$$\lambda = (\vec{\pi}, \mathbf{A}, \mathbf{B}) \tag{2.18}$$

die Größe K des Ausgabealphabets und die Anzahl N der möglichen Zustände definiert. In der Literatur (Levinson et al., 1983) wird das Modell, welches durch den Parametersatz 2.18 beschrieben wird, als *Hidden-Markov-Modell* bezeichnet.

2.3.2 Produktionswahrscheinlichkeiten

Man möchte ermitteln, wie groß die Wahrscheinlichkeit ist, dass ein gegebenes HMM λ eine Beobachtung $\mathbf{O} = (O_1 \dots O_T)$ produziert. Diese lässt sich ermitteln, indem zunächst die Wahrscheinlichkeiten für alle zulässigen Pfade $\mathcal{S}$ berechnet werden und dann über $\mathcal{S}$ aufsummiert wird.

$$P(\mathbf{O}|\lambda) = \sum_{\mathcal{S}} P(\mathbf{O}|\lambda, \mathcal{S}) \tag{2.19}$$

mit

$$P(\mathbf{O}|\lambda, \mathcal{S}) = \pi_{q_0} \prod_{t=1}^{T} a_{q_{t-1}q_t} b_{q_{t-1}q_t}(O_t) \tag{2.20}$$

Diese Berechnung erscheint zu unhandlich. Schließlich ist über N^T Zustandsfolgen zu summieren. Die Anzahl der Multiplikationen ergibt sich etwa zu $2T \cdot N^T$ und wächst damit exponentiell mit der Dauer der Beobachtungssequenz.

Als eine geeignete Alternative ist die Berechnung der sog. *Vorwärts-* und *Rückwärtswahrscheinlichkeiten* bekannt. Aus der Bezeichnung geht bereits hervor, dass es sich bei der Berechnung der Vorwärtswahrscheinlichkeiten $\alpha_t(j)$ um einen Berechnungsalgorithmus in Richtung der Zeitachse der Sequenz und bei den Rückwärtswahrscheinlichkeiten $\beta_t(i)$ um einen Berechnungsalgorithmus handelt, der entgegen der Zeitachse, also vom Ende einer Sequenz startet. Zu jedem Zeitpunkt t werden n Koeffizienten bestimmt. Daraus ergibt sich, dass diese Algorithmen einen Berechnungsaufwand linear zur Sequenzlänge T aufweisen. Insgesamt ergibt sich ein Berechnungsaufwand von $2N^2T$ für den Algorithmus.

Der Rekursionsalgorithmus lässt sich leicht herleiten und kann in kurzer Form dargestellt werden.

- Initialisierung:

  für $j = 1 \ldots N$ und $i = 1 \ldots N$ ist

  $$\alpha_1(j) = \pi_j b_j(O_1) \tag{2.21}$$
  $$\beta_T(i) = 1 \tag{2.22}$$

- Rekursion:

  $$\alpha_t(j) = \left(\sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij} \right) b_j(O_t);\ t > 1 \tag{2.23}$$
  $$\beta_t(i) = \sum_{j=1}^{N} \beta_{t+1}(j) a_{ij} b_j(O_{t+1});\ t < T \tag{2.24}$$

- Terminierung

$$P(\mathbf{O}|\lambda) = \sum_{j=1}^{N} \alpha_T(j) \tag{2.25}$$

$$P(\mathbf{O}|\lambda) = \sum_{j=1}^{N} \pi_j b_j(O_1)\beta_1(j) \tag{2.26}$$

Die oben bereits erwähnte zeitliche Richtung wird in Gleichung 2.23 und 2.24 sehr schön ersichtlich. Bemerkenswert ist, dass zu jedem Zeitpunkt t gilt:

$$\alpha_t(j)\beta_t(j) = P(\mathbf{O}, q_t = j|\lambda) \tag{2.27}$$

und

$$P(\mathbf{O}|\lambda) = \sum_{j=1}^{N} \alpha_t(j)\beta_t(j) \tag{2.28}$$

Dieser Sachverhalt wird nachfolgend für die Schätzung der Modellparameter durch den Baum-Welch-Algorithmus ausgenutzt.

2.3.3 Schätzung der Modellparameter

Die Aufgabenstellung dieses Kapitels ist, die HMM-Parameter $\lambda = (\vec{\pi}, \mathbf{A}, \mathbf{B})$ in geeigneter Form zu schätzen. Allgemein wird ein Verfahren gesucht, welches ein neues Modell

$$\hat{\lambda}(\hat{\vec{\pi}}, \hat{\mathbf{A}}, \hat{\mathbf{B}}) \geq \lambda(\vec{\pi}, \mathbf{A}, \mathbf{B}) \tag{2.29}$$

findet bzw. ein Verfahren welches das Modell in seiner Produktionswahrscheinlichkeit *maximiert*. Ein solches Verfahren wurde erstmalig von (Baum and Petrie, 1966, Baum and Eagon, 1967) erwähnt. Obgleich das Verfahren wesentlich auf der Berechnung der Vorwärts- und Rückwärtswahrscheinlichkeiten beruht, ist es häufig unter dem Namen *Baum-Welch-Verfahren* [BW] zu finden. Ein Beweis für die Gültigkeit des Verfahrens wird in (Baum and Eagon, 1967, Baum and Sell, 1968) vorgestellt. Das BW-Verfahren soll nun

folgend kurz dargestellt werden.

Ausgehend von einer Observation $\mathbf{O} = (O_1 \dots O_T)$ der Länge T und einem HMM λ mit diskreten Ausgabeverteilungen lässt sich die a posteriori Wahrscheinlichkeit für einen Zustand $q_t = s_j$ zu einem beliebigen Zeitpunkt t unter Zuhilfenahme von 2.23, 2.24 und 2.27 aufschreiben. Die Wahrscheinlichkeit in einem bestimmten Zustand t zu sein, sei $\gamma_t(i)$.

$$\begin{aligned} \gamma_t(i) &= \\ P(q_t = j|\mathbf{O}, \lambda) &= \frac{P(\mathbf{O}, q_t = j|\lambda)}{P(\mathbf{O}|\lambda)} \\ &= \frac{\alpha_t(j)\beta_t(j)}{\sum_i \alpha_t(i)\beta_t(i)} \end{aligned} \tag{2.30}$$

Für einen Zustandsübergang $q_t = s_i \rightarrow q_{t+1} = s_j$ mit der Übergangswahrscheinlichkeit $\xi_t(ij)$ gilt:

$$\begin{aligned} \xi_t(ij) &= \\ P(q_t = i, q_{t+1} = j|\mathbf{O}, \lambda) &= \frac{P(\mathbf{O}, q_t = i, q_{t+1} = j|\lambda)}{P(\mathbf{O}|\lambda)} \\ &= \frac{\alpha_t(i)a_{ij}b_j(O_{t+1})\beta_{t+1}(j)}{\sum_i \alpha_t(i)\beta_t(i)} \end{aligned} \tag{2.31}$$

Wenn außerdem über alle in der zeitlichen Abfolge entstehenden λ-bedingten Erwartungswerte summiert wird, erhalten wir durch geeignete Substitutionen das folgende Gleichungssystem:

$$\hat{\pi}_i = \gamma_1(i) \tag{2.32}$$

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(ij)}{\sum_{t=1}^{T-1} \gamma_t(i)} \tag{2.33}$$

$$\hat{b}_{ik} = \frac{\sum_{t=1|O_t=v_k}^{T-1} \gamma_t(j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \tag{2.34}$$

Die Schätzung geeigneter neuer Parameter verlangt im wesentlichen die wiederholte Berechnung der Vorwärts- und Rückwärtswahrscheinlichkeiten.

Es sei ausdrücklich darauf hingewiesen, dass die Verteilungen $b_t(ik)$ in dieser Betrachtung zunächst diskret sind. Nur deshalb fungiert die Schätzgleichung als *Zähler*, wie oft das im Ausgabealphabet $\mathcal{K}$ enthaltene Symbol emittiert wurde. Auf das Schätzen kontinuierlicher Verteilungen wird im nächsten Kapitel ausführlich eingegangen.

Eine weitere Möglichkeit der Herleitung dieser Gleichungen wurde erstmalig ebenfalls von Baum (Baum et al., 1970) über die Kullbach-Leiber-Statistikgleichung gezeigt. Auf die Darstellung soll an dieser Stelle verzichtet werden, da auch sie zu einer weiteren Bestätigung der oben hergeleiteten Gleichungen führt.

Bei genauer Analyse der Schätzgleichungen fallen zwei Besonderheiten auf. Zunächst ist die starke Abhängigkeit von den initialen λ_0 zu bemerken. Der iterativ- und rekursive Algorithmus ist nur in Lage, das der Initialisierung nächst liegende Maximum aufzufinden. Außerdem wird während der lokalen Maximumsermittlung keine Rücksicht darauf genommen, ob die konsumierten Daten noch mit den Daten übereinstimmen, die wir aus einer phonetischen Sichtweise den einzelnen Phonen zuordnen würden. Das Verfahren verzerrt somit unsere ursprüngliche phonetische Sichtweise zu Gunsten der lokalen Maximumsfindung.

2.3.4 Schätzung kontinuierlicher Mischverteilungen

Im Kapitel 2.2 "Klassifikation von Merkmalsvektoren" wurden bereits Vor- und Nachteile verschiedener Klassifikationsvarianten behandelt. Die vorliegende Arbeit verwendet zur Modellklassifikation Hidden-Markov-Modelle. An die emittierenden Knoten der HMMs, werden kontinuierliche Gaußsche Mischverteilungsdichten zur probabilistischen Merkmalsklassifikation "angehängt". Es ist nun die Aufgabe solche HMMs zu trainieren und den Baum-Welch-Algorithmus geeignet umzuformen.

Die Schätzformeln für kontinuierliche Mischverteilungsdichten (Gl. 2.3) lassen sich aus der Produktionswahrscheinlichkeit ableiten. Die Herleitung lässt sich sowohl für diagonale als auch für Kovarianzmatrizen analog nachvollziehen.

Zu jedem Zeitpunkt t wird ein Zustand q_t eingenommen und in Abhängigkeit davon eine Komponente k_t der Mischverteilung, die einen Ausgabevektor $\mathbf{x}_t$ erzeugt. Völlig analog zu 2.31 lässt sich die Produktionswahrscheinlichkeit angeben:

$$\begin{aligned} \zeta_t(j,k) &:= \\ P(q_t = i, k_t = k|\mathbf{O},\lambda) &= \frac{P(\mathbf{O}, q_t = i, k_t = k|\lambda)}{P(\mathbf{O}|\lambda)} \qquad (2.35) \\ &= \begin{cases} \frac{\sum_i \alpha_{t-1}(i) a_{ij} c_{jk} g_{jk}(O_t) \beta_t(j)}{P(\mathbf{O}|\lambda)} & t > 1 \\ \frac{\sum_i \pi_j c_{jk} g_{jk}(O_1) \beta_1(j)}{P(\mathbf{O}|\lambda)} & t = 1 \end{cases} \qquad (2.36) \end{aligned}$$

Unter Verwendung von $\gamma_t(i)$ aus Gleichung 2.31 lassen sich die Baum-Welch-Gleichungen folgendermaßen entwickeln:

$$\hat{c}_{jk} = \frac{\sum_{t=1}^{T} \zeta_t(jk)}{\sum_t \gamma_t(j)} \tag{2.37}$$

$$\hat{\mu}_{jk} = \frac{\sum_{t=1}^{T} \zeta_t(jk)\vec{x}_t}{\sum_t \zeta_t(jk)} \tag{2.38}$$

$$\hat{\mathbf{\Sigma}}_{jk} = \frac{\sum_{t=1}^{T} \zeta_t(jk)\vec{x}_t\vec{x}_t^T - \hat{\mu}_{jk}\hat{\mu}_{jk}^T}{\sum_t \zeta_t(jk)} \tag{2.39}$$

Die Schätzgleichungen für die initialen Zustände $\hat{\pi}$ und die Koeffizienten der Transitionsmatrix $\hat{\mathbf{A}}$ werden aus den Gleichungen 2.32 und 2.33 übernommen.

2.3.5 Parameterkoppelung kontinuierlicher Verteilungen

Durch M Linearkombinationen der einzelnen n-dimensionalen Gaußschen Verteilungsfunktionen entstehen $M \cdot \frac{n^2+3n}{2}$ Parameter (c_i, μ_j *und* Σ) pro emittierendem Zustand eines HMMs. Dadurch entstehen unter Umständen sehr viele Parameter, die errechnet werden müssen. Im Allgemeinen reicht das vorhandene Trainingsmaterial nicht aus, um alle Parameter ausreichend bestimmen zu können. Es ist daher notwendig verschiedene Parameter auf der Modellebene zu koppeln, um einen verringerten Parametersatz zu erhalten. Dazu sind eine Vielzahl von Techniken bekannt. Generell lassen sich die Techniken in wissensbasierte und datengetriebene Verfahren aufteilen.

Wissensbasierte Verfahren finden sich hauptsächlich im Bereich der Triphonmodellierung wieder. Die Technik der Triphonmodellierung berücksichtigt linke und rechte phonetische Kontexte eines Phons und ist daher in der Lage, den Nachteil der unzureichenden kontextuellen Modellierung der Phone aufzuheben. Auf die Modellierung durch Triphone wird im Kapitel 3 näher eingegangen.

Mangels Daten ist es im allg. nicht möglich, alle benötigten Triphonparameter einer Domäne geeignet zu schätzen. Theoretisch wären bei einem verwendetem Phoninventar von 42 Phonen bis zu 42^3 Triphonkombinationen möglich. Selbst unter der Berücksichtigung, dass diese hohe Anzahl an Kombinationen nicht auftritt, bleibt eine Anzahl von Modellparametern übrig, die nicht ausreichend berechnet werden kann. Es werden daher lautlich ähnliche linke und rechte Kontexte in Gruppen zusammengefasst und gekoppelt (Deg et al., 1988).

Ein weiteres Verfahren, in dem ebenfalls ähnliche linke und rechte Kontexte wissensbasiert zusammengefasst werden, ist in der Literatur unter dem Namen *Phonem Environment Clustering* oder auch *Tree-Based Clustering* bekannt und erstmals von (Sagayama, 1989) vorgestellt worden. Ausgehend vom Zentralphon eines Triphons wird die Gruppierung der Kontexte durch einfache Ja/Nein-Entscheidungen herbeigeführt. Dieses Verfahren erlaubt eine schrittweise binäre Untersuchung des Musterraumes.

Datengetriebene Verfahren koppeln Parameter z.B. aufgrund der euklidischen Abstände der Mittelwertsvektoren und treffen dadurch eine Ähnlichkeitsentscheidung, ob sich zwei oder mehr Wahrscheinlichkeitsräume vereinigen lassen. Es gibt außerdem noch eine Reihe anderer Verfahren der Distanzberechnung, die sich zusammengefasst u.a. in (Huang et al., 1991) finden lassen. Datengetriebene Verfahren lassen sich sowohl auf Phon- als auch auf Triphonmodellierungen anwenden. Erstmalig wurden entsprechende Vorgehensweisen in (Niemann, 1983) aufgezeigt.

2.3.6 Ermittlung der optimalen Zustandsfolge

Ein trainiertes Hidden-Markov-Modell gibt uns zunächst keinen Aufschluss darüber, durch welche mögliche Zustandfolge $\mathbf{q}$, eine gegebene Beobachtung $\mathbf{O} = (O_1 \dots O_T)$, optimal beschrieben wird.

Gesucht wird diejenige Zustandfolge $\mathbf{q}$, die bei einem gegebenem λ die Beobachtung $\mathbf{O}$ mit maximaler a posteriori Wahrscheinlichkeit erzeugt.

$$P(q|\mathbf{O},\lambda) = \frac{P(\mathbf{O},q|\lambda)}{P(\mathbf{O}|\lambda)} \tag{2.40}$$

Der Beitrag der Produktionswahrscheinlichkeit im Nenner ist jedoch für die Maximumsbestimmung unerheblich, so dass sich die Gleichung zu

$$P^*(\mathbf{O}|\lambda) := \tag{2.41}$$
$$P(\mathbf{O},q^*|\lambda) = \max_q P(\mathbf{O},q|\lambda) \tag{2.42}$$

umformen lässt. Es können jedoch mehrere Folgen q vorhanden sein, die die Maximumsforderung erfüllen.

Dieses Optimierungsproblem wird mit dem sog. *Viterbi-Algorithmus* (Viterbi, 1967) gelöst.

Hierzu werden Wahrscheinlichkeiten $\vartheta_t(j)$ definiert, die partiell optimale Pfade beschreiben und im Zustand j enden.

$$\vartheta_t(j) = \max_{q_1 \ldots q_{t-1}} P(\mathbf{O_t}, q_1 \ldots q_{t-1}, q_t = j|\lambda) \tag{2.43}$$

Der Viterbi-Algorithmus nutzt die Maximierungsforderung, aus indem er rekursiv jeweils das Produktionsmaximum akkumuliert.Um die Folge zu dekodieren, wird während der Rekursion die Rückverzweigungsmatrix ψ aufgebaut, um abschließend die produzierte Zustandfolge zu ermitteln.

Formal lässt sich der Algorithmus für $j = 1 \ldots N$ wie folgt aufschreiben:

- Initialisierung:

$$\vartheta_1(j) = \pi_j b_j(O_1) \tag{2.44}$$
$$\psi_1(j) = 0; \tag{2.45}$$

- Rekursion:

$$\vartheta_t(j) = \max_i(\vartheta_{t-1}(i)a_{ij})b_j(O_t)) \tag{2.46}$$
$$\psi_t(j) = \arg\max_i \vartheta_{t-1}(i)a_{ij} \tag{2.47}$$

- Terminierung:

$$P^*(\mathbf{O}|\lambda) = \max_j \vartheta_T(j) \tag{2.48}$$

$$q_T^* = \arg\max_i \vartheta_T(j) \tag{2.49}$$

- Rückverfolgung für $t = T-1 \ldots 1$:

$$q_t^* = \psi_{t+1}(q_{t+1}^*) \tag{2.50}$$

Der wesentliche Unterschied zur oben abgeleiteten Produktionswahrscheinlichkeit ist, dass nach der Viterbisuche nur *die* Zustandsfolge zur Berechnung verwendet wird, die die Modellwahrscheinlichkeit maximiert. Wird statt der Wahrscheinlichkeitsberechnung ein Abstandsmaß und statt der Multiplikationsoperation die Addition verwendet, deckt sich der Viterbi-Algorithmus mit dem DTW-Algorithmus (Schukat-Talamazzini, 1995b).

Kapitel 3

Akustisch phonetische Wortmodellierung

Nachdem in Kapitel 2 grundlegend der mathematische Zugang zur Mustererkennung mit Hidden-Markov-Modellen geschaffen wurde, ist es notwendig, das System mit Wissen über die zu erkennende Sprache anzureichern.

Spracherkennung und Systemdesign stehen immer im Kontext zu der zu modellierenden Domäne. In der derzeitigen Forschungs- und Entwicklungsgemeinschaft existieren verschiedene große oder kleine Domänen, die aufgabenbezogen Verwendung finden. Je nach Größe des in der Domäne verwendeten Wörterbuchs lassen sich Ganzwortmodelle, Subwortmodelle oder Phon- bzw. Triphonmodelle zur Modellierung heranziehen.

Eine Abschätzung: das in dieser Arbeit verwendete Korpus umfasst ca. 9500 Worte, die zum großem Teil nur selten auftreten. Ca. 60% der Worte tauchen im Trainingsmaterial nur bis zu dreimal auf.

Um abzuschätzen, ob das vorhandene Trainingsmaterial für eine Ganzwortmodellierung ausreicht, wird zunächst aus den transkribierten Trainingsdaten die mittlere Wortlänge in Graphemen errechnet. Es ergibt sich eine mittlere Wortlänge von zehn Zeichen. Angenommen, es würde bei der Ganzwortmodellierung ein emittierender Zustand pro Zeichen angesetzt, müssten 95000

verschiedene Modellzustände berechnet werden. Kommen zu den Zuständen noch Mixturen hinzu, erhöht sich die Parameterzahl zusätzlich. Unter einer Mixtur wird eine einzelne Verteilung der Linearkombination 2.3 verstanden. Das bedeutet für ein Set aus initialen Modellen ca. 119 MByte Speicherbedarf, der mit zunehmenden Anzahl von Mixturen pro HMM-Zustand linear ansteigen wird. Weiterhin stehen ca. 17 Mio Merkmalvektoren zur Verfügung. Es ist berechenbar, dass durch diese Anzahl von Merkmalvektoren, den initialen Modellen im Durchschnitt pro Zustand maximal ca. 178 Vektoren zur Verfügung stünden.

Diese exemplarischen Abschätzungen führen immer zu dem Ergebnis, dass es unmöglich scheint 9500 Ganzwortmodelle zu schätzen. Weiterhin muss berücksichtigt werden, dass in einem geeigneten Testset u.U. Worte auftreten können, die vorher nicht gesehen wurden.

Aus dieser Abschätzung folgt, dass die vorhandenen Trainingsdaten möglichst gut zusammengefasst werden müssen, um geeignete Wortuntereinheiten ausreichend trainieren zu können. An diese Wortuntereinheiten werden die nachfolgenden Anforderungen gestellt.

3.1 Paradigmen der phonetischen Wortmodellierung

Die Anforderungen die an eine geeignete Wortuntereinheit gestellt werden, lassen sich folgendermaßen zusammenfassen (Lee, 1989, Fink, 1991):

1. **Präzision**
   Die Untereinheit sollte im Inventar möglichst gut unterscheidbar gegenüber anderen Einheiten sein. Nur so wird eine ausreichende Trennschärfe gewährleistet.

2. **Robustheit**
   Nur durch ausreichend vorhandene Observationen einer Untereinheit lassen sich gute Konfidenzen erreichen.

3. **Modularität**
   Das Inventar der Untereinheiten sollte so gewählt werden, dass sich alle Worte der gewählten Domäne durch sie modular darstellen lassen.

4. **Transfer**
   Die Untereinheiten sollten so generell sein, dass sich daraus auch neue, unbekannte Worte generieren lassen.

Aufgrund dieser Anforderungen wird nun versucht, nach geeigneten Wortuntereinheiten hoher Präzision zu suchen. Die wesentlichen Ansätze, die insbesondere für den vorliegenden Forschungsbeitrag notwendig sind, werden in den folgenden Abschnitten behandelt.

3.2 Phonetisch basierte Wortuntereinheiten

Als kleinste phonetische Einheiten werden die Phone definiert, die abhängig von der jeweiligen Sprache 20-60 Einträge im Phoninventar haben können (Shoup, 1980). Für das Deutsche wird die Zahl von 48 Phonen nach (Meinhold and Stock, 1982) angegeben.

3.2.1 Kontextfreie Phonmodellierung

Durch Phone lassen sich Worte folgendermaßen abbilden:

Guten	$\longrightarrow$	`g u: t @ n`
Abend	$\longrightarrow$	`Q a: b @ n t`

Abbildung 3.1: *Darstellung von Worten durch Phone*

Durch diese Art der Modellierung sind Robustheit, Modularität und Transfer bereits gewährleistet. Um die Trennschärfe noch zu erhöhen, kann auf die Technik der Triphonmodellierung (Abschnitt 3.2.2), in der linke und rechte phonetische Kontexte berücksichtigt werden, zurückgegriffen werden.

Die in Abbildung 3.1 dargestellte Wortmodellierung bezeichnet man als kontextfreie Modellierung, da die späteren Phonmodelle alle auftretenden linken und rechten Kontexte in ihrem Modell vereinigen und damit eine Differenzierung nach Kontexten nicht möglich ist. Weiterhin wird diese Modellierung in Hinblick auf die einzig vorhandene Aussprache eines Wortes im Wörterbuch als kanonisch bezeichnet. Um Aussprachevariationen modellieren zu können, ist es notwendig, das Wörterbuchinventar geeignet zu erweitern. Ein typisches Beispiel zur Aussprachemodellierung ist in Abbildung 3.2 am Wort "haben" dargestellt. In der ersten Zeile wird mit der hochdeutschen Aussprache begonnen, um anschließend das Wort zunehmend zu verschleifen.

haben	$\longrightarrow$	`h a: b @ n`
haben	$\longrightarrow$	`h a: b n`
haben	$\longrightarrow$	`h a: m`
haben	$\longrightarrow$	`h a m`
haben	$\longrightarrow$	`h o m`
haben	$\longrightarrow$	`h e n`
haben	$\longrightarrow$	`h e b @`

Abbildung 3.2: *Aussprachevarianten des Wortes "haben"*

Im Gegensatz zur Ganzwortmodellierung wirkt sich die Modellierung durch kontextfreie Phone auf die Erkennungsleistung eher negativ aus (Bahl et al., 1980), da durch die Ganzwortmodellierung linke und rechte Kontexte in der Modellierung berücksichtigt sind. Der Vorteil der Ganzwortmodellierung wird jedoch mit ansteigender Wortanzahl bei etwa gleichbleibender Menge Trainingsdaten schnell zunichte gemacht, so dass die phonetische Wortmodellierung bei großen Wortschätzen zu bevorzugen ist (Svendsen et al., 1989).

Einen weiteren vielversprechenden Ansatz zur Wortmodellierung stellt die Modellierung durch Sprechsilben dar. Es wird davon ausgegangen, dass alle koartikulatorischen Aussprachevarianten innerhalb einer Silbe verbleiben und daher ihre Trennschärfe erheblich sein sollte. Leider sind in der deutschen Sprache

sehr viele Silben vorhanden, so dass nach (Ortmann, 1980) zur Modellierung einer Domäne mit 1000 Worten ca. 1000 Silben benötigt werden, wodurch wieder eine zu hohe Anzahl von Modellen trainiert werden müsste.

3.2.2 Kontextabhängige Phonmodellierung

Um den Nachteil der unzureichenden kontextuellen Modellierung der Phone aufzuheben wird konzeptionell das Triphon (Schwartz et al., 1985) eingeführt. Durch Berücksichtigung der jeweiligen linken und rechten Nachbarphone werden so kontextabhängige Phone, die sog. *Triphone*, gebildet. Die Umsetzung diese Konzeptes ist in Abbildung 3.3 am Wort "Beispiel" dargestellt.

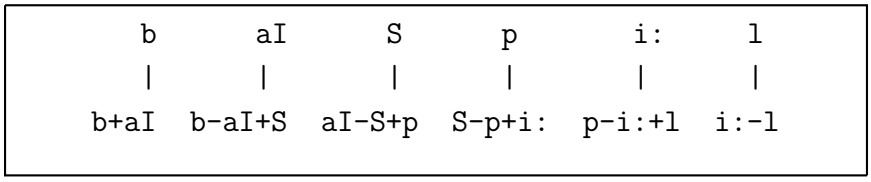

Abbildung 3.3: *Triphonrepräsentation des Wortes "Beispiel"*

Der kontextuelle Einfluss eines Phons wird nun durch die beiden benachbarten Phone gekennzeichnet. Üblicherweise treten zusätzlich zu den Triphonen noch *Biphone* an Satzanfängen und -enden in den zu lernenden Korpora auf (Lee et al., 1992). Oft ist es notwendig, einige kontextfreie Phone zusätzlich im Modellinventar vorrätig zu haben. Diese werden in der Literatur mit *Monophonen* bezeichnet.

Die Leistungsverbesserung der Worterkennungsrate gegenüber kontextunabhängigen Phonen, ist jedoch vom Überlappungsgrad der Wortschätze des Trainingssets und des Testsets abhängig (Derouault, 1987).

Ganzwortmodellierung von Funktionswörtern Eine weitere vielversprechende kontextabhängige Phonmodellierung stellt die phonspezifische Ganzwortmodellierung einzelner kurzer Funktionswörter dar.

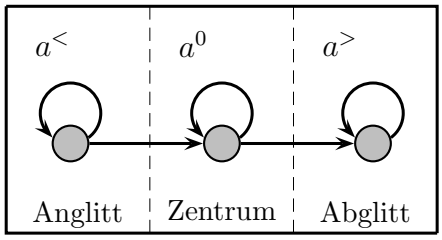

Abbildung 3.4: *Subphonetische Feinstruktur*

Funktionswörter tragen nach (Lee et al., 1992) bis zu 65% der Fehler in der Spracherkennung bei. Der Grund hierfür ist, dass wenige Funktionswörter einen großen Teil der Gesamtanzahl der Wörter ausmachen.

Für das in dieser Arbeit untersuchte Korpus Verbmobil II machen z.B. 0.3% des Wörterbuches, welche Observationen mit einer Frequenz > 3000 im Trainingsset aufweisen, den Anteil von 36,8% aller Observationen des Trainingssets aus. Dieses Phänomen wurde in der vorliegenden Arbeit jedoch nicht berücksichtigt.

3.2.3 Phonetische Feinstruktur

Akustische Prozesse sind oft von nicht-stationärer (unifrequenter) Natur. Dies wird insbesondere bei Diphthongen klar, da diese ganz offensichtlich über mehrere Produktionsphasen verfügen. Aufgrund dieser Erkenntnis wird bei der phonetischen *Feinstruktur* (Abbildung 3.4) von An- und Abglittsphasen und von einer quasi-stationären Phase, dem sog. Nukleus, gesprochen (Schwartz et al., 1985, Bußmann, 1990).

Die Analogie zu Hidden-Markov-Modellen ist in Abbildung 3.4 durchaus beabsichtigt. Dargestellt wird der Sachverhalt, dass ein Phon durch drei separate Zustände modelliert werden kann, die in zeitlicher Abfolge durchlaufen werden. Da nicht bekannt ist, wie lange eine einzelne akustische Phase anhält,

kann jeder Zustand je nach Anforderung wiederholt durchlaufen werden, so dass beliebige Sequenzlängen erzeugt werden können.

3.3 Aussprachewörterbücher

Die im Kapitel 3.2.1 beschriebene Wahl eines Phoninventars wird in so genannte Aussprachewörterbücher übertragen. Hier wird im einfachsten Fall die kanonische Aussprache für ein Wort abgebildet. Zusätzlich können alternative Aussprachevarianten berücksichtigt werden.

Der Suchalgorithmus benötigt das Wörterbuch, bzw. seine Einträge, um die entsprechende phonetische HMM-Folge für ein Wort zu generieren und damit im Suchnetz das "wahrscheinlichste" Wort bei vorliegendem Sprachsignal zu errechnen.

Außerdem besteht durch das verwendete Programm `HTK` (Young et al., 2002) die Möglichkeit, bei mehreren Aussprachevarianten ein sog. *forced alignment* durchzuführen. Üblicherweise dient diese Technik zur Annotierung von Modell-Segmentgrenzen im Trainingsmaterial. Das verwendete Programm verfügt zusätzlich über die Möglichkeit, die wahrscheinlichere Aussprachevariation zu errechnen und in die Transkriptionsdateien einzutragen. Die dadurch entstandenen verbesserten phonbasierten Transkriptionen können anschließend zu einem weiteren Training herangezogen werden.

Die Modellierung eines geeigneten Aussprachewörterbuches kann unter Umständen viel Zeit und Erfahrung in Anspruch nehmen. Das in der vorliegenden Arbeit benutzte Aussprachewörterbuch wurde aus dem Verbmobilprojekt übernommen und beinhaltet keine Aussprachevariationen. Es beinhaltet somit haupsächlich die kanonische Aussprache, wie sie aus dem Hochdeutschen bekannt ist.

3.4 Basistopologien in der Spracherkennung

Im wesentlichen werden in der Literatur vier grundsätzliche HMM-Topologien, die zur Repräsentation einer sprachlichen Äußerung $\mathcal{O}$ bzw. ihrer phonetischen Untereinheiten O_p sinnvoll erscheinen, erwähnt. Zulässig sind nur dem dynamischen Sprachprozess entsprechende, quasizeitlich aufeinanderfolgende Zustände. Dies gewährleistet eine wichtige Randbedingung, nämlich die, das Modell stets wieder verlassen zu müssen. Die üblicherweise an den Knoten sitzenden Klassifikationsfunktionen spielen in der jetzigen Darstellung eine untergeordnete Rolle und werden zunächst nicht berücksichtigt.

In den folgenden Abbildungen werden die verschiedenen Modellierungsansätze mit jeweils zunehmender Komplexität der Topologien dargestellt. Die jeweiligen Zustandsübergänge werden auf zwei verschiedene Arten dargestellt. Einerseits werden sie in einfacher Matrixform dargestellt, wobei ein Punkt einen möglichen Übergang von $Spalte_i(i \in 1..N)$ nach $Zeile_j(j \in 1..M)$ kennzeichnet. Im unteren Teil der Abbildung werden dieselben Übergangsmöglichkeiten durch Verlaufspfeile gekennzeichnet, wobei die grauen Kreise die jeweiligen Zustände des Modells zeigen.

Üblich sind HMM-Modelle mit drei emittierenden Zuständen, abgeleitet aus der phonetischen Vorstellung heraus wie sie im vorangegangenem Abschnitt beschrieben wurde, bis hin zu sieben emittierenden Zuständen. Die vorliegende Arbeit beschäftigt sich mit der Topologieoptimierung ausgehend von drei emittierenden Zuständen pro phonetischem HMM.

Die einfachste und wegen der minimalen Anzahl der zu schätzenden Parameter ist die lineare links-rechts Topologie (Abbildung 3.5) ohne Zustandsauslassungen (i.f. Skip genannt). Diese Topologie ist insbesondere bei kleinen Datenmengen eine der meist verwendeten Topologien. In diesem Fall sind fünf Transitionskoeffizienten und drei Emissionsverteilungen zu berechnen. Das Durchlaufen aller drei Zustände q_i wird erzwungen. Jedoch ist die Modellierungsqualität dadurch eingeschränkt, dass es unmöglich ist artikulatorische

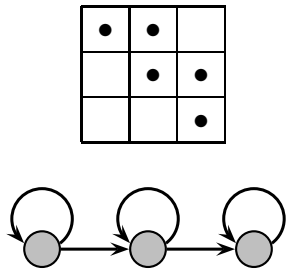

Abbildung 3.5: *Lineares links-rechts Modell*

Verschleifungen, die kürzer als drei Sequenzeinheiten sind, zu modellieren.

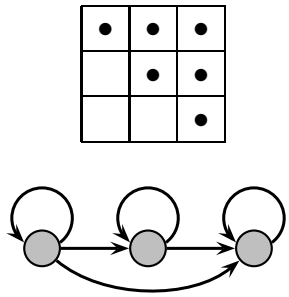

Abbildung 3.6: *Reduzierte BAKIS-Topologie mit zentralem Sprung*

Eine weniger bekannte Topologie ist die links-rechts Topologie mit einem Skip. Der Ursprung dieser Modellierung ist nicht bekannt. Diese Topologie (Abbildung 3.6) findet sich z.B. in (Bechetti and Ricotti, 1999). Sie erlaubt auch die Modellierung kürzerer Sequenzen bis zu zwei Zuständen. Aus welchem Grund das zentrale Subphon übersprungen werden soll, bleibt jedoch unklar. Die Komplexität nimmt um einen weiteren Transitionskoeffizienten zu. Aufgrund der Ähnlichkeit zum BAKIS-Modell wird diese Variante nachfolgend mit "reduzierter BAKIS-Topologie" bezeichnet.

In Abbildung 3.7 ist die nach (Bakis, 1974) benannte Topologie (nachfolgend als BAKIS-Topologie benannt) dargestellt. Sie weist zusätzliche Skips innerhalb des Modells auf und verfügt daher über eine höhere Komplexität als die vorherigen Modelle. Die Skips erlauben weitere Weglassungen in der phonetischen Sprachproduktion. Da sowohl der Eingangszustand als

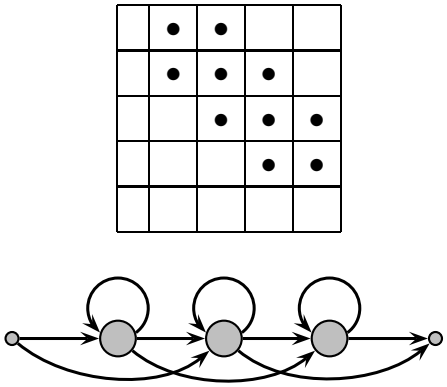

Abbildung 3.7: *Topologie nach BAKIS*

auch der Ausgangszustand übersprungen werden kann, weist die Darstellung zwei Pseudoknoten auf. Sie sagen aus, dass kommend aus dem vorherigem Phon-HMM der erste Zustand des betrachteten Modells übersprungen werden kann. Dies gilt gleichermaßen für den letzten Zustand, falls ein weiteres Modell folgt. Die Pseudozustände sind als kleine Kreise dargestellt. Da nicht jedes Phon über alle drei Phasen verfügen muss bzw. die artikulatorischen Verschleifungen bestimmte Phasen auslassen, ermöglicht diese Topologie eine verbesserte Repräsentation des Sprachsignals.

Die vollständige Links-Rechts-Topologie aus Abbildung 3.8 ist mit dreizehn zu schätzenden Transitionswahrscheinlichkeiten das komplexeste Links-Rechts-Modell, welches in der Topologiemodellierung eingesetzt werden kann. Im Vergleich zu den beiden oben dargestellten Topologien können hier ganze Teile phonetischer Realisierungen weggelassen werden, was eine ausgesprochen genaue Modellierung zulässt. Theoretisch wäre es sogar möglich, das gesamte Modell zu überspringen. Diese Topologie wird aufgrund nicht ausreichender Trainingsdaten selten verwendet.

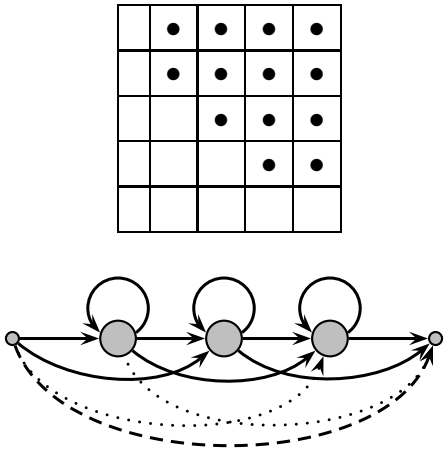

Abbildung 3.8: *Vollständige Links-Rechts-Topologie*

3.5 Sprachmodelle

Die menschliche Fähigkeit gesprochene Sprache gut zu verstehen ist zum großen Teil darauf zurückzuführen, dass der Mensch mehr als nur einen akustischen Klassifikationsprozess zur Decodierung einsetzt. Der Mensch verfügt z.B. über komplexes Grammatikwissen und über Verstand, der es ihm ermöglicht selbst total Unverständliches noch zu verstehen. Es sind noch eine Vielzahl von anderen Techniken und Fähigkeiten zu Sprachverstehen des Menschen bekannt, die hier nicht nur den Umfang der Arbeit sprengen, sondern auch thematisch irrelevant sind.

In der Erkennung von gesprochener Sprache werden zumindest Sprachmodelle benutzt, um weiteres Wissen dem Erkennungsverfahren zuzuführen. Grundsätzlich wird zwischen grammatikalisch basierten Sprachmodellen, die in (Salomaa, 1978, Hopcroft and Ullman, 1979) beschrieben werden, und den

stochastischen Grammatiken (Ney, 1992) unterschieden.

Grammatikbasierte Sprachmodelle zeichnen sich dadurch aus, dass sie eine feste Abfolge von Worten bzw. Wortklassen vorgeben, aus denen Sätze gebildet werden. Weicht der Sprecher von der vorgegeben Folge in irgendeiner Form ab, entstehen unweigerlich grobe Erkennungsfehler.

Eine Alternative sind stochastische Sprachmodelle. Sie haben den großen Vorteil, abhängig von der jeweiligen Domäne, das Sprachmodell aufgrund des vorliegenden Trainingsmaterials errechnen zu können und Wortfolgen probabilistisch zu erfassen. Es werden dabei einfach Wortfolgehäufigkeiten errechnet, die dem Suchverfahren als weitere Information zur Verfügung stehen. Zu der Familie der stochastischen Sprachmodelle zählen die sog. m-Gramm Sprachmodelle, die in der vorliegenden Arbeit Verwendung finden.

3.5.1 Stochastische m-Gramm Sprachmodelle

Es wird nun der mathematische Formalismus eines m-Gramms vorgestellt, beginnend beim Unigramm über das Bigramm bis hin zur allgemeinen Form des m-Gramms. Eine allgemeine Darstellung findet sich u.a. in (Jelinek et al., 1982).

Liegt eine Wortfolge $\mathcal{W} = w_1 \dots w_n$ in einem Unigramm-Modell vor, wird angenommen, dass sich die Wahrscheinlichkeit für ein Wort n folgendermaßen errechnet:

$$P(w_n|w_1 \dots w_{n-1}) = P(w_n) \tag{3.1}$$

Die Wahrscheinlichkeit der gesamten Wortfolge ist dann:

$$P(w_1 \dots w_N) = \prod_{n=1}^{N} P(w) \tag{3.2}$$

Die Wahrscheinlichkeiten der einzelnen Worte können durch Auszählen

der Frequenzen aus dem Trainingskorpus ermittelt werden. Ebenso kann errechnet werden, wie häufig bestimmte Worte auf andere folgen. Durch diese Wortfolge-Wahrscheinlichkeiten ergibt sich das sog. Bigramm-Modell.

Die Wahrscheinlichkeit, dass ein Wort w_n im Kontext eines vorangegangenen Wortes w_{n-1} folgt, ergibt sich durch ein Bigramm-Modell:

$$P(w_n|w_1 \dots w_N) = P(w_n|w_{n-1}) \tag{3.3}$$

Analog ergibt sich für eine mögliche Wortfolge aus dem Bigramm-Modell die Wahrscheinlichkeit:

$$P(w_1 \dots w_N) = P(w_1) \prod_{n=2}^{N} P(w_n|w_{n-1}) \tag{3.4}$$

Die Analogie lässt sich entsprechend fortführen. Zum Schluss lässt sich die allgemeine Form eines *m-Gramm* Sprachmodells folgendermaßen darstellen:

$$P(w_n|w_1 \dots w_N) = P(w_n|w_{n-m+1} \dots w_{n-1}) \tag{3.5}$$

Als Beispiel sind in Tabelle 3.1 einige Einträge aus den in dieser Arbeit verwendeten Unigramm-Modellen und Bigramm-Modellen dargestellt. Im Unigramm-Modell sind die logarithmischen Wahrscheinlichkeiten der einzelnen Worte eingetragen. Im Bigramm-Teil der Tabelle ist die logarithmische Wortfolge-Wahrscheinlichkeit eingetragen, die Aufschluss über die errechnete Wortkombination gibt.

Aus dem Unigramm der Tabelle 3.1 lässt sich ersehen, dass Verabredungen getroffen werden bzw. Gespräche zu Zeitpunkten stattfinden, die nicht eindeutig Abends oder Morgens stattfinden. Jedoch lässt sich aus der Wortfolge-Wahrscheinlichkeit des Bigramm-Teils der Tabelle ablesen, dass die allgemeine Begrüßungsformel "Guten Tag" deutlich häufiger Verwendung findet als "Guten Morgen" oder "Guten Abend". Das Wort "Kolloquium" taucht während der Terminabsprachen weniger häufig auf und wird daher innerhalb der Domäne selten gezählt.

log P	
Unigramm	
-2.6371	guten
-3.5853	Morgen
-2.4343	Tag
-3.8445	Abend
-5.0576	Kolloquium
Bigramm	
-0.1162	guten Tag
-1.0949	guten Morgen
-2.2444	guten Abend

Tabelle 3.1: *Unigramm- und Bigramm-Wahrscheinlichkeiten*

Die Möglichkeit, bei einer nicht geschätzten Wortkombination auf eine geringere Granularitätsstufe des Sprachmodells auszuweichen, ist durch das parallel vorhandene Unigramm jederzeit gegeben. Diese Eigenschaft der Sprachmodellglättung wird in der Literatur als "Backoff m-Gramm" bezeichnet. Diese Möglichkeit und die Tatsache, dass im Sprachmodell nie ganze Sätze sondern lediglich Kontextinformationen stochastisch abgebildet werden, verschafft den m-Grammen einen großen Vorteil gegenüber den grammatikalisch basierten Sprachmodellen. Während der Suche nach der wahrscheinlichsten Symbol- bzw. Wortfolge werden in der Erkennung die errechneten Sprachmodell-Wahrscheinlichkeiten hinzugezogen. In der Spracherkennung kann mit zunehmender Komplexität des m-Gramm Sprachmodells die Erkennungsleistung verbessert werden, falls ausreichend Trainingsmaterial zur Verfügung steht. Jedoch finden aufgrund der wenig zu berechnenden Parameter meist Bigramme oder Trigramme Verwendung (Jelinek, 1985).

In Tabelle 3.2 sind Wortfehlerraten in Abhängigkeit vom jeweiligen Sprachmodell bei gleichbleibendem Sprachmodell-Skalierungsfaktor dargestellt. Der Sprachmodell-Skalierungsfaktor entscheidet, wie groß der Einfluss des Sprachmodells während des Dekodierungsschrittes tatsächlich ist und kann als Systemparameter gewählt werden.

Diese Ergebnisse wurden im Laufe der Arbeit im Zuge verschiedener Vorexperimente ermittelt, um einen Eindruck über die Auswirkung der verschiedenen Sprachmodelle auf die Erkennungsleistung zu erhalten.

Sprachmodell	WFR [%]
Zerogramm	≈ 60
Unigramm	≈ 50
Bigramm	≈ 35

Tabelle 3.2: *Verminderung der Wortfehlerrate [WFR] durch m-Gramm Sprachmodelle bei konstanter Skalierung des Sprachmodells*

Das sog. Zerogramm weist keinerlei Wortfrequenzen auf, so dass hier alle Worte während des Decodierungsprozesses gleich gewichtet sind. Die große Bedeutung des Einsatzes von Sprachmodellen in der stochastischen Spracherkennung wird hier deutlich. Der akustische Klassifikator ist bei großem Vokabular nur in der Lage, vierzig Prozent aller Worte korrekt zu bestimmen. Diese Leistung erhöht sich bei Einsatz von Sprachmodellen, die Wortfolge-Wahrscheinlichkeiten beinhalten, mit zunehmendem Anstieg von m.

Geeignete Trigramme sind nur selten bestimmbar, da i.A. zu wenig Trainingsdaten vorliegen. Bei einem Vokabular von 10^4 Worten gäbe es theoretisch maximal 10^{12} Tripel zu berechnen, so dass oft Bigramme als günstige Alternative in der Spracherkennung verwendet werden. Insbesondere unterstützt das in dieser Arbeit verwendete HTK-Toolkit nur Backoff-Bigramm-Modelle, so dass die Wahl des Sprachmodells an dieser Stelle eingeschränkt ist.

3.5.2 Perplexität

Um die durchschnittliche Streuung des Sprachmodells zu erfassen, wird der Begriff der Perplexität definiert. Sie stellt den mittleren Verzweigungsgrad zwischen Wörtern pro Wortposition in Abhängigkeit der Historie h_n dar.

$$PP = \left[\prod_{n=1}^{N} P(w_n|h_n) \right]^{-\frac{1}{N}} \tag{3.6}$$

Das Ziel ist, die Perplexität eines Sprachmodell möglichst zu minimieren, um an der gerade zu berechnenden Wortposition die Entropie möglichst gering zu halten. Die Veränderung der Wortfehlerrate verhält sich etwa proportional zur Wurzel der Differenz zweier Perplexitäten (Kimball et al., 1986).

$$\triangle WER \sim \sqrt{\triangle PP} \tag{3.7}$$

Kapitel 4

Topologieoptimierung durch Erkennungssimulation

Datengetriebene Topologiemodellierung wurde bereits Ende der achtziger Jahre von (Rabiner and Juang, 1993, Lee, 1989) gefordert. Die Optimierung von HMM-Topologien, worunter in dieser Arbeit die Modellierung der Zustandsanzahl verschiedener emittierender Zustände oder die Modellierung verschiedener Pfadmöglichkeiten durch Loops oder Skips verstanden wird, wird in der Literatur selten beschrieben.

Die Ursache hierfür ist der große systematische Aufwand, der betrieben werden müsste, um z.B. Unterschiede in der Erkennungsrate zwischen zwei voneinander unabhängigen Phontopologien messen zu können. Unter "unabhängig" wird lediglich verstanden, dass zwei Topologien unterschiedliche Daten eines Korpusses konsumieren.

Ausgehend von zwei unabhängigen Phontopologien müsste jede Topologiekombination für jedes Wort im gesamten Kontext des vorliegenden Korpus trainiert und evaluiert werden. Dieser Aufwand stellt sich als sehr groß dar, so dass auf kombinierte Modelle wie zum Beispiel auf das BAKIS Modell 3.7 zurückgegriffen wird. Der kombinatorische Aufwand entfällt beim BAKIS-Modell sofort, jedoch bleibt die optimale Topologie innerhalb des Modells verborgen, da die Daten durch mehrere gleichzeitig vorhandenen Topologien modelliert werden. Dieser Umstand kann bei zu geringen Da-

tenmengen in Abhängigkeit der Modellkomplexität zu Erkennungsverlusten führen, auch wenn innerhalb des Modells die optimale Topologie vorhanden ist.

Um aus diesem Modell die nicht geeigneten impliziten Topologien zu eliminieren, wird ein Bewertungsalgorithmus für HMMs, bzw. einzelner Topologien benötigt, der in Abschnitt 4.2 motiviert wird.

4.1 Optimierung durch parallele Hidden-Markov-Topologien

Um eine optimierte Modelltopologie aus verschiedenen implizit vorhandenen Topologien eines Phon-HMMs zu ermitteln, sind bestimmte Vorraussetzungen notwendig, die im folgenden Text erörtert werden. Grundsätzlich soll ermittelt werden, welche Modelltopologie eines HMMs sich für die Qualitätsberechnung besonders gut eignet. Die maximale Topologievariationsbreite stellt ein vollständiges Links-Rechts-HMM dar und bietet sich für eine solche Untersuchung zunächst an. Ausgehend von diesem Modell wird ein Multi-Paralleles HMM motiviert, welches für einen datengetriebenen Modellierungsansatz als Alternative zu einem vollständigen Links-Rechts-HMM herangezogen werden kann. Zusätzlich wird die Möglichkeit diskutiert, ob es möglich ist, anhand von unabhängig trainierten Topologien eine Auswahl zu treffen.

Vollständige Links-Rechts HMMs zur Topologieoptimierung Im folgenden wird das Akronym LR-HMM anstelle des vollständigen Namens (Links-Rechts-Hidden-Markov-Modell) verwendet. Ein LR-HMM mit z.B. drei emittierenden Zuständen besteht grundsätzlich aus einer BAKIS-Topologie und zusätzlichen übergreifenden Sprüngen, die das Auslassen von mehr als einem Zustand erlauben.

In Abbildung 4.1 sind, im Gegensatz zu den typischen BAKIS-Transitionen, die übergreifenden Transitionen gestrichelt dargestellt. In Abbildung 4.1 sind zusätzlich Transitions- und Emissionswahrscheinlichkeiten eingetragen, die in der einfachen Abbildung 3.8 aus Übersichtsgründen noch weggelassen wurden.

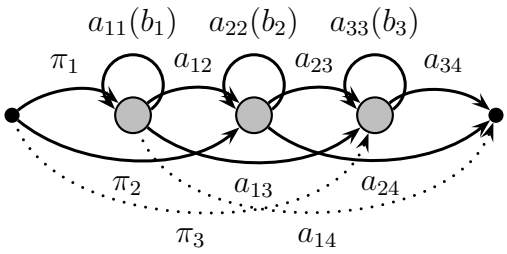

Abbildung 4.1: *Darstellung eines vollständigen unidirektionalen Hidden-Markov Modells*

Die Transitionswahrscheinlichkeiten werden dem Formalismus folgend mit a_{ij} und die Initialisierungswahrscheinlichkeiten mit π_{ij} bezeichnet. Geeignete Verteilungsfunktionen b_i sind der Übersicht halber in der Abbildung nicht dargestellt, befinden sich aber an den emittierenden Zuständen, die durch die größeren Kreise gekennzeichnet sind. Die kleinen schwarzen Punkte stellen die symbolischen Anfangs- und Endzustände des jeweiligen Phonmodells dar. An ihnen werden verschiedene Phonmodelle zu Wörtern gemäß eines geeigneten Aussprachewörterbuchs konkateniert. Es lässt sich ausrechnen, dass dieses Modell insgesamt sieben implizite Topologien beinhaltet, die durch die verschiedenen Reihenfolgen der emittierenden Zustände gebildet werden können. Die verschiedenen Topologien können anhand ihrer jeweiligen Zustandsanzahl benannt werden. Es handelt sich konkret um eine Dreier-, drei Zweier- und drei Einer-Topologien. Die berechneten Transitionswahrscheinlichkeiten a_{ij} der eben erwähnten Variationen sind keineswegs unabhängig voneinander zu verstehen, sondern über das gemeinsame Datensharing, entstehend durch Parameterkopplung der Verteilungsfunktionen b_i, miteinander korreliert.

Dieser Sachverhalt wird in der Abbildung 4.2 verdeutlicht, in der die Aussprachevarianten, die durch das Modell ermöglicht werden, getrennt dargestellt sind. Da die Loopwahrscheinlichkeiten für alle in den Spalten dargestellten Zustände immer gleich sind, werden sie in der Darstellung nur einmal in der obersten Zeile der Abbildung mit ihren zugehörigen Verteilungen b_i aufgeführt. Auf die Loopdarstellungen der einzelnen Zustände wurde in der

Abbildung aus Übersichtsgründen verzichtet. Die gestrichelte Linie kennzeichnet die Zustände, die Aufgrund des Datensharings dieselben Sprachmerkmale konsumieren sollen. Hieraus ergibt sich, dass die Transitions- bzw. die Initialisierungswahrscheinlichkeiten in der Art berechnet werden, dass sie gleichzeitig verschiedene Topologien repräsentieren, also eine genaue Modellierung eines phonetischen Vorkommens nicht präzise darstellen können.

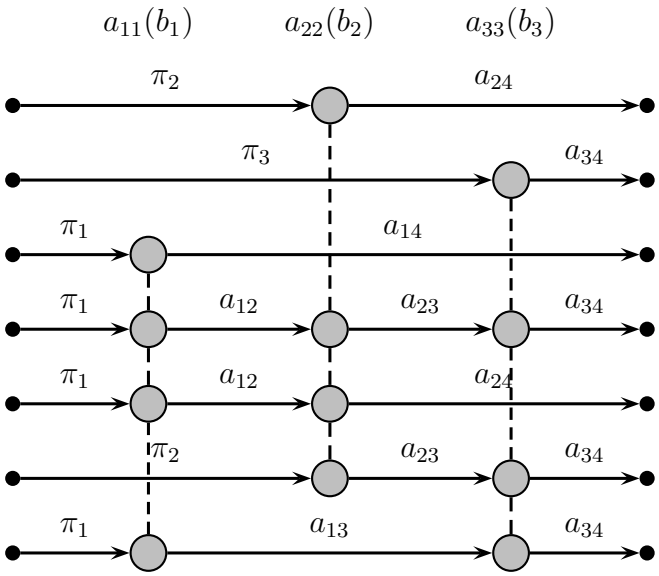

Abbildung 4.2: *Darstellung von miteinander verbundenen Zuständen eines komplexen Hidden-Markov-Modells*

Der in dieser Arbeit vorgestellte Ansatz verfolgt jedoch die Strategie, diejenige Topologie zu ermitteln, welche die phonetische Beobachtung mit der größten möglichen Wahrscheinlichkeit beschreibt. Es wäre zwar möglich, die sieben Topologien im dargestellten LR-HMM für eine bestimmte durchschnittliche Merkmalssequenzlänge auf ihre jeweilige Topologiewahrscheinlichkeit zu untersuchen, und dann den verallgemeinerten Pfad, der die größte Wahrscheinlichkeitsmasse beinhaltet, als den geeignetsten zu bezeichnen. Unter dem verallgemeinerten Pfad wird im folgendem Text derjenige Pfad durch eine Topologie verstanden, der unabhängig von den benötigten Loops eine gegebene

Folge von Merkmalsvektoren konsumiert.

Aufgrund des vorher beschriebenen Datensharings wird die gewählte Topologie jedoch nicht das gewünschte Ergebnis hervorbringen, da ihre Zustände noch durch Anteile anderer verallgemeinerter Pfade beeinflusst sind. Eine Alternative stellen die so genannten Multi-Parallelen Hidden-Markov-Modelle dar, bei denen gleichzeitig zwei voneinander unabhängige Topologien geschätzt werden können.

Multi-Parallele Hidden-Markov-Modelle zur Topologieoptimierung: ein Multi-Paralleles Hidden-Markov Modell, kurz MPP (Multi Parallel Path), ist ein Phonmodell, in dem mindestens zwei voneinander unabhängige Topologien als separate Zweige, die in Konkurrenz zueinander stehen, gleichzeitig trainiert werden. In (Jay et al., 2001) wird ein MPP mit zwei parallelen Zweigen vorgestellt, um Hidden-Markov-Topologien zur Erkennung von handschriftlichem Hangul (koreanische Schrift) zu optimieren. Im Gegensatz zum Spracherkennungsproblem wird in ihrer Arbeit in erster Linie die Zustandsanzahl der jeweiligen HMMs optimiert und auf mögliche Skips nicht eingegangen. Auch weisen ihre optimierten Modelle große Zustandslängen auf, die in der Spracherkennung unüblich sind. Dennoch erscheint der Ansatz vielversprechend, Phonrepräsentationen hoher Qualität aufzufinden.

Da in einem MPP im Gegensatz zum vollständigen Links-Rechts-HMM kein Datensharing vorhanden ist, wird jede konkurrierende Topologie eine spezifische Phonmodellierung repräsentieren. Abbildung 4.3 stellt ein solches Modell mit zwei parallelen Topologien dar. Hier konkurrieren ein lineares Zwei-Zustands-Modell mit einem linearem Drei-Zustands-Modell miteinander.

Im Gegensatz zur Abbildung 4.1 treten zusätzlich zu den Transitionswahrscheinlichkeiten a_{ij} und den Emissionswahrscheinlichkeiten b_i des oberen Zweiges noch die Wahrscheinlichkeiten a^*_{ij} und b^*_i des unteren Zweiges auf. Die mit $*$ gekennzeichneten Wahrscheinlichkeiten können unkorreliert von den nicht gekennzeichneten berechnet werden und stellen somit eine eigenständige Phonrepräsentation dar. Falls nun einer der beiden Zweige ein Phon besser repräsentiert, wird angenommen, dass in diesem Zweig eine höhere Wahrscheinlichkeitsmasse kumulieren wird. Selbstverständlich könnten MPP-HMMs auch

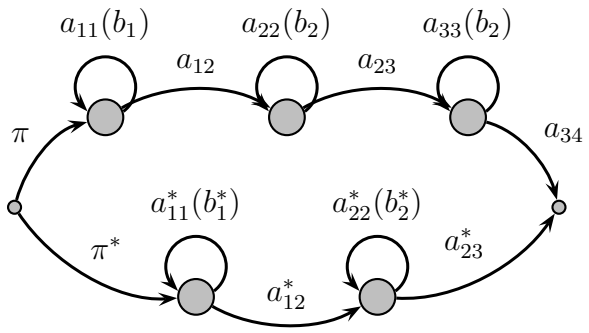

Abbildung 4.3: *MPP-HMM mit 3 und 2 Zuständen*

direkt zur Spracherkennung eingesetzt werden, jedoch widerspricht dies dem Wunsch, kleine und mit wenigen Parametern ausgestattete, handliche Modelle zur Verfügung zu haben.

Separat trainierte Hidden-Markov-Modelle zur Topologieoptimierung: eine weitere Alternative wäre, unabhängig voneinander zwei Topologien auf dem gesamten Trainingsmaterial zu trainieren und sie anschließend zu bewerten. Jedoch würden voraussichtlich die jeweiligen Modelle ungünstig verzerrt werden, wenn sie Beobachtungsfolgen modellieren müssten, für die sie nicht geeignet sind. Dadurch wäre eine Topologieoptimierung im vorher beschriebenen Sinne nicht mehr bzw. nur stark eingeschränkt möglich.

4.2 Qualitätsbestimmung von Hidden-Markov-Modellen

Unter "Qualität" eines stochastischen Modells wird die Zuverlässigkeit verstanden, einen unbekannten Wert durch ein gegebenes Modell zu schätzen. Um eine Aussage über die Zuverlässigkeit einer Schätzung zu erhalten kann, die sog. Konfidenzschätzung (Dallmansn and Elster, 1983) herangezogen werden, die beim Vorhandensein einer bestimmten Irrtumswahrscheinlichkeit

das sog. Vertrauensintervall zurückliefert. Um eine Konfidenzschätzung bezüglich des Erwartungswertes für eine Normalverteilungsfunktion durchzuführen, ist die Kenntnis der Varianz der Verteilung notwendig. Die jeweiligen Verteilungsdichtefunktionen der Zustände eines HMMs bestehen aus M Linearkombinationen (vgl. 2.3), um einen beliebig gestalteten Teil des Merkmalsraums zu beschreiben. Das hat zur Folge, dass die Verteilungsdichtefunktionen innerhalb ihrer Linearkombination stochastisch korreliert sind. Daher ist es unmöglich, eine Gesamtvarianz, die jedoch für eine Konfidenzschätzung notwendig wäre, aus der Linearkombination zu errechnen.

Dennoch ist es möglich, sich den einfachen Zusammenhang zwischen Varianz und Konfidenz zunutze zu machen, die sich zueinander umgekehrt proportional verhalten. Ein "qualitativ hochwertiges" Phon-Modell ist demnach ein Modell, welches sich durch geringe Gesamtvarianzen der einzelnen Zustände auszeichnet und damit in der Lage ist, gegeben eine Merkmalsstichprobe, einen möglichst hohen Gesamtwahrscheinlichkeitswert zu liefern. In wieweit die benötigte Gesamtvarianz der einzelnen Zustände ermittelt werden kann, wird in Abschnitt 4.2.3 genauer diskutiert.

Eine hohe Qualität würde im Kontext der Spracherkennung jedoch auch bedeuten, dass die sog. Robustheit der einzelnen Modelle sinkt. Die Robustheit, unter der i.A. die Unempfindlichkeit gegenüber nicht vorher trainierten Phonvariationen während der Erkennung verstanden wird, ist ein wichtiger Gesichtspunkt beim Training von Spracherkennern. Eine Verminderung der Robustheit wird sich vor allem in der Verschlechterung von Erkennungsergebnissen niederschlagen. Da der Erfolg der Topologieoptimierung durch Erkennungsergebnisse gemessen wird, wird ein Gesamterfolg des Ansatzes nur dann möglich, wenn der Gewinn der Topologieoptimierung größer ist als der gleichzeitige Erkennungsverlust durch die Verminderung der Robustheit.

In dieser Arbeit wird davon ausgegangen, dass einhergehend mit der Topologieoptimierung nicht nur Varianzen kleiner werden, sondern auch die Modellierung der Daten präziser wird. Sollte dies der Fall sein, wäre es denkbar, die

präzisere Topologie unter Beibehaltung der Ortsvektoren künstlich zu verbreitern. Dadurch könnte dem Verlust der Robustheit entgegengewirkt werden. Diese Experimente sind jedoch nicht Bestandteil der vorliegenden Arbeit.

4.2.1 Heuristiken zur Algorithmusspezifikation

Folgende Heuristiken werden zur Algorithmusfindung herangezogen:

- Mit zunehmender Anzahl von emittierenden Knoten eines Phon-HMMs sinkt die Varianz der Verteilungsfunktionen (Duda and Hart, 1973). Die Qualität eines Phon-HMMs nimmt demnach mit seiner Länge zu.

- Die Qualität eines Phon-HMMs nimmt mit der Zahl der zur Verfügung stehenden Trainingsdaten zu, da sich die Konfidenz proportional zur Menge der Trainingsdaten entwickelt.

- Die HMM-Qualität steigt mit zunehmender Anzahl von Mixturen, wobei an dieser Stelle auf ausreichend vorhandenes Trainingsmaterial geachtet werden muss, damit jeder einzelnen Mixtur noch genügend Merkmalvektoren während des Trainings zur Verfügung stehen.

- Der Baum-Welch-Algorithmus modelliert lokal optimal, so dass über die resultierende Varianzinformation der trainierten Modelle eine Aussage über die Qualität der Phon-HMMs abgeleitet werden kann.

4.2.2 Motivation eines geeigneten Qualitätsmaßes

Die Qualität eines Phon-HMMs ist nicht trivial durch die Parameter des HMM-Modells errechenbar. Zum Beispiel ist der Ort im Merkmalsraum, durch den die Dichtewerte errechnet werden, ohne entsprechende Daten nicht auffindbar. Deswegen wird an dieser Stelle ein Algorithmus vorgeschlagen, der über den Umweg einer Erkennungssimulation die Berechnung eines Qualitätsmaßes auch ohne das Vorhandensein von Sprachdaten zulässt.

Dieser Algorithmus soll eine Zustandsfolge durch simulierte Merkmale, gegeben eine Topologie, simulieren und die Berechnung der Gesamtsimulationswahrscheinlichkeit des Modells zulassen. Hiervon ausgehend soll es ebenfalls möglich sein, die simulierten Merkmale so zu wählen, dass die Simulationswahrscheinlichkeit des Modells maximal wird.

Um sich das notwendige Verhalten des Algorithmus zu vergegenwärtigen, ist es notwendig sich genau vor Augen zu führen, wie der Bewertungsprozess bei einer gegebenen Sprachsequenz $\mathcal{O}$ mit Hilfe von HMMs von statten geht. Die Sprachsequenz liegt in Form von Merkmalsvektoren $\vec{X}_i$ vor. Die Merkmalvektoren werden mit Hilfe des Viterbi-Algorithmus den verschiedenen Zuständen q_i der HMMs nach größter Wahrscheinlichkeit zugeordnet, und zwar genau so, dass am Ende die *wahrscheinlichste* Zustandsfolge $q_{P(max)}$ als Ergebnis vorliegt.

In jedem Berechnungsschritt wird die Wahrscheinlichkeit aus dem Produkt des gelernten Transitionskoeffizienten a_{ij} und dem jeweiligen Wahrscheinlichkeitsdichtewert der Verteilung $b_i(\vec{X}_i)$ ermittelt und errechnet, ob es sich um einen eher quasi-statischen oder einen dynamischen Schritt innerhalb des Modells handelt. Diese Berechnung wird solange rekursiv wiederholt, bis alle zur Verfügung stehenden Merkmalvektoren einer Sequenz $\vec{X}_i$ konsumiert wurden. Die Entscheidung, ob das System in einem Zustand q_i verharrt oder dynamisch zum nächsten wechselt, wird über den berechneten Wert der Wahrscheinlichkeitsdichtefunktion ermittelt.

Abbildung 4.4 stellt diesen Prozess stark vereinfacht dar. Es wird angenommen, dass an jedem emittierenden Knoten eines linearen Phon-HMMs nur eine einfache Normalverteilung "angebracht" ist, die die Verteilung in dem jeweiligen Zustand modelliert. In der Abbildung verläuft die zeitliche Richtung von links unten nach rechts oben. In jedem Zeitschritt liegt ein Merkmalsvektor gekennzeichnet durch $\vec{X}_i$ vor, der durch das Beispielmodell bewertet wird. Der jeweilige Wahrscheinlichkeitsanteil $a_{ij} \cdot b_j(\vec{X}_i)$, der in die Gesamtberechnung einfließt, ist ebenfalls eingetragen, wobei die Transitionswahrscheinlichkeiten dem HMM-Formalismus folgend mit a_{ij} und die Eintrittswahrscheinlichkeit mit π gekennzeichnet sind. Die Verteilungen sind in gewohnter Weise durch b_i

gekennzeichnet.

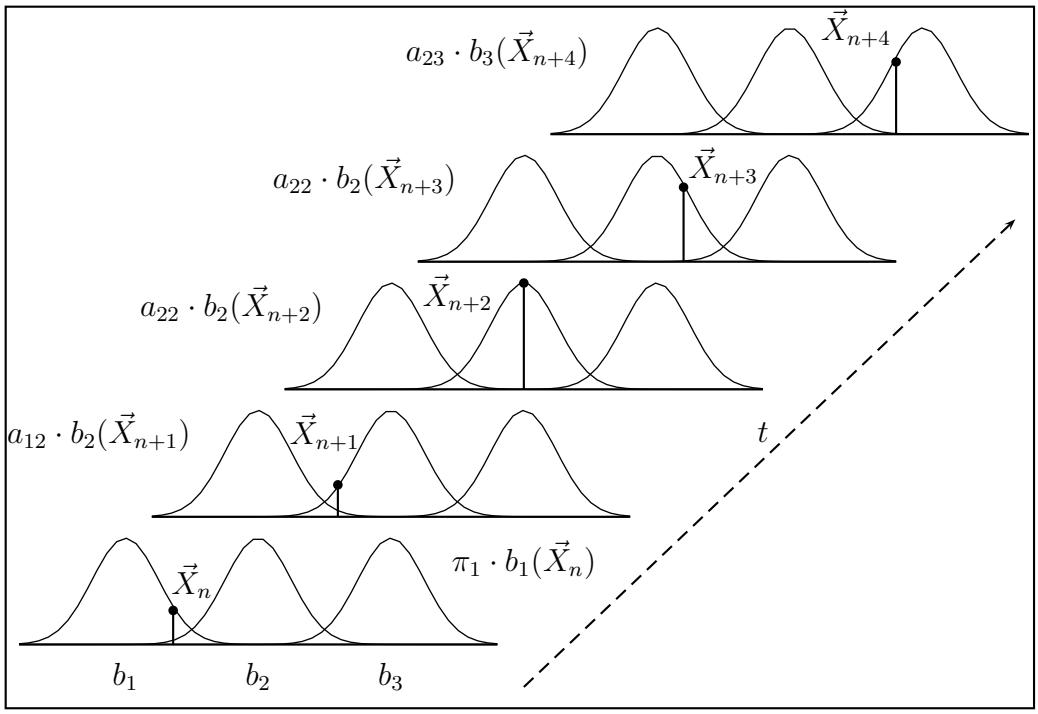

Abbildung 4.4: *Dynamischer Klassifikationsprozess eines Phons*

Um nun auf den Gedanken einer simulierten Zustandfolge zurückzukommen, wird vereinfacht angenommen, dass eine Sequenz $\mathcal{O}_{simul}$ generiert wird, die in Form von simulierten Merkmalsvektoren $\vec{X}_i^{simul}$ vorliegt. Diese Merkmalsvektoren sollen während der Klassifikation durch das vorliegende Modell, immer den optimalen Beitrag zur Modellwahrscheinlichkeit liefern.

Um eine optimale Gesamtsimulationswahrscheinlichkeit zu erreichen, ist es notwendig während jedes Bewertungsschrittes den maximalen Dichtewert der jeweiligen Verteilungsdichtefunktion zu erreichen. Dies ist genau dann der Fall, wenn der simulierte Merkmalvektor mit dem Mittelwertvektor übereinstimmt, also $\vec{X}_i^{simul} = \vec{\mu}$ gilt. Dieser Verlauf ist für eine bestimmte Folge von Merkmalsvektoren $\bar{P}_l$ (hier fünf Frames) in Abbildung 4.5 skizziert und kann während eines Erkennungslaufes mit Echtdaten durchaus eintreten. Typische phonspezifische Merkmalssequenzlängen können aus den Daten nach dem Training er-

mittelt werden. Es ist dadurch möglich die Simulation auf phontypische Merkmalssequenzlängen anzuwenden.

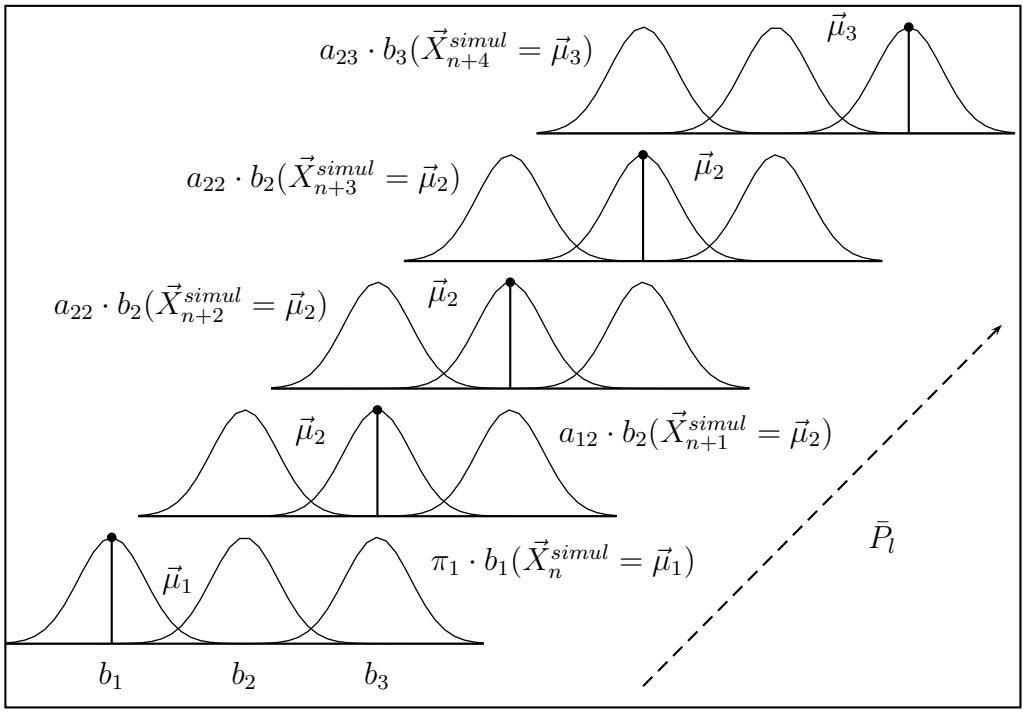

Abbildung 4.5: *Simulierter Klassifikationsprozess eines Phons*

Die Bewertung beginnt modellbedingt im ersten möglichen Modellzustand. Danach wird angenommen, dass die nächsten drei Merkmale optimal durch die Verteilung des zweiten emittierenden Zustands des Beispielmodells beschrieben werden, bevor der letzte Zustand des Modells eingenommen wird. Die Merkmalssequenz, bestehend aus fünf Merkmalsvektoren, ist somit vollständig konsumiert worden.

Wegen dieser speziellen Verlaufsannahme ist es notwendig, auf dem mittleren Zustand eines Dreier-Modells einen 2-fachen Loop zu berücksichtigen. Die Simulation wäre aber auch möglich, wenn die Verteilung der Merkmale anders ist. Es wäre auch möglich, einen Loop auf dem ersten Zustand und auf dem letzten emittierenden Zustand durchzuführen, wodurch auch die angenommene Merkmalssequenzlänge von fünf Merkmalen möglich wäre.

Zur Berechnung der Gesamtsimulationswahrscheinlichkeit einer Topologie für eine gegebene Folge von Merkmalsvektoren müssten alle gültigen Zustandskombinationen errechnet werden und danach über diese Ergebnisse gemittelt werden. Das stellt sich insbesondere bei längeren Merkmalssequenzlängen als sehr aufwendig dar.

Falls wie in diesem Ansatz eine Simulation ohne Kenntnis der realen Daten durchgeführt werden soll, müssten alle Kombinationen, die zu der aus den Daten ermittelten Phonsequenzlängen führen würden, berücksichtigt werden. Wenn das getan würde, müsste noch der Durchschnitt aller Kombinationen berechnet werden, um einen Wert zu erhalten, mit dem weiter gerechnet werden kann.

Aus den Daten lässt sich ermitteln, dass die durchschnittliche Länge eines Phons 9 Frames (90 ms) beträgt. Bei einem linearen Modell mit drei emittierenden Zuständen müssen somit 28 Kombinationen berechnet werden. Sollten später auch Modelle mit Skips berücksichtigt werden, würde sich diese Zahl sogar noch erhöhen. Es wird daher auf die Vereinfachung der arithmetischen Mittelwertsbildung zurückgegriffen, um für die Loopanteile einer Topologie einen durchschnittlichen allg. Loopwahrscheinlichkeitsbeitrag zu errechnen, der aus dem Mittel der Produkte $a_{jj} \cdot b_j$ errechnet wird. Die Kenntnis dieser allg. Loopwahrscheinlichkeit lässt zu, die Wahrscheinlichkeit beliebiger Zustandsfolgen zunächst ohne Wiederholungen zu berechnen. Anschließend kann mit Hilfe des allg. Loopwahrscheinlichkeitsbeitrags die Berechnung der Gesamtsimulationswahrscheinlichkeit bezüglich der gewünschte Merkmalssequenzlänge erfolgen.

Durch diese Vereinfachung wird jede zu untersuchende Topologie auf die vorgegebene Merkmalssequenzlänge normiert, was die Vergleichbarkeit zweier Topologien, ausgedrückt durch ihren jeweiligen Gesamtwahrscheinlichkeitswert, ermöglicht. Dadurch wird es möglich, die oben vorgestellten MPP-Modelle nach dem Training in Zweige aufzuteilen, die Gesamtsimulationswahrscheinlichkeiten zweigweise zu berechnen, und diese Ergebnisse anschließend zu vergleichen.

4.2.3 Behandlung linear kombinierter Dichteverteilungen während der Erkennungssimulation

Die Verteilungsdichtefunktionen eines Phon-HMMs beschreiben durch ihre Linearkombinationen einen Merkmalsraum, der möglicherweise durch mehrere Häufungspunkte gekennzeichnet ist. Es wäre dann unmöglich einen Mittelwertsvektor der Gesamtverteilung zu bestimmen, der alle Häufungspunkte gleichzeitig beschreibt. Ein solcher Mittelwertvektor ist jedoch notwendig, um die oben beschriebene Simulation durchführen zu können.

Folgende alternative Herangehensweisen sind möglich:

- Ermittlung des Mittelwertvektors des größten lokalen Häufungspunktes anhand des größten Koeffizienten der Linearkombination.
- Durch ein Wahrscheinlichkeits-Maximierungsverfahren, welches alle in der Linearkombination auftauchenden Mittelwertvektoren berücksichtigt. Es wird derjenige Mittelwertvektor für die Simulation verwendet, der die Linearkombination der Verteilungen maximiert.
- Über eine Mittelwertbildung aller Einzelwahrscheinlichkeiten $\bar{P}(x_i = \mu_i)$ der Linearkombinationen ohne Berücksichtigung der Gaußschen Koeffizienten.
- Durch eine gewichtete Mittelwertbildung aller Einzelwahrscheinlichkeiten der Linearkombinationen, wobei die Gaußschen Koeffizienten als Gewichte verwendet werden.

Denkbar wäre, diejenige Verteilung der Linearkombination zu betrachten, die den größten Gewichtungskoeffizienten c_k aufweist. Es würde dann angenommen, dass diese Verteilungsfunktion das Zentrum der Gesamtverteilung wesentlich bestimmt und ihr Mittelwertvektor könnte zur Berechnung der Simulation herangezogen werden. Diese Herangehensweise setzt allerdings voraus, dass die Verteilung mit dem größten Koeffizienten und ihrem Mittelwertvektor tatsächlich den Häufungspunkt beschreibt, was keinesfalls gewährleistet ist. So könnte z.B. ein Randteil des Merkmalraumes durch

"eine" Verteilungsfunktion sehr gut modelliert worden sein, was einen großen Koeffizienten ergeben würde. Das Zentrum des Merkmalraumes wird aber vermutlich nur durch sehr viele Funktionen approximiert. An dieser Stelle könnte dann eine Fehlentscheidung getroffen werden.

Ein verbesserter Ansatz einen geeigneten Mittelwertvektor zu finden könnte sein, mit allen verfügbaren Mittelwertvektoren jeweils die Gesamtwahrscheinlichkeit zu berechnen und sich anschließend für denjenigen zu entscheiden, der die Gesamtwahrscheinlichkeit maximiert. Im Fall eines modellierten Gebirges mit kleineren Inseln würde hier immer einer derjenigen Mittelwertvektoren herangezogen werden, der in einem Bereich liegt, wo er von seinen umgebenen Funktionen noch viele Beiträge erhält, unabhängig davon, ob es sich dabei um eine "qualitativ" gute Modellierung handelt. Weiterhin können noch sog. "Ausreißer" auftreten, die diesen Ansatz stark verfälschen würden. Verteilungsfunktionen, die nur durch wenige Trainingsdaten berechnet wurden, können extrem hohe Dichtewerte bei entsprechend kleiner Varianz aufweisen. Die Wahl eines solchen Mittelwertvektors würde die Simulation zur Qualitätsanalyse stark verfälschen.

Die dritte diskutierbare Variante ist, für alle vorhandenen Mittelwertvektoren eines Zustandes, die maximale Gesamtwahrscheinlichkeit zu errechnen, indem das arithmetische Mittel aller Einzelwahrscheinlichkeiten $P_l(x_i = \mu_i)$ gebildet wird. Auch hier besteht die Möglichkeit einer Verzerrung, wenn z.B. außen liegende Teilräume durch viele eher "scharfe" Verteilungen modelliert werden.

Da die oben beschriebenen Ansätze aufgrund der genannten Schwächen und Nachteile nicht geeignet erscheinen, wird in dieser Arbeit eine veränderte varianzbezogene Sichtweise vorgeschlagen und implementiert.

Es wird angenommen, dass sich eine qualitativ hochwertige Gesamtmodellierung eines Zustandes dadurch auszeichnet, dass die einzelnen Verteilungen der Linearkombination geringe Streuungen aufweisen. Jeder einzelne Qualitätsbeitrag einer Funktion wird durch die Koeffizienten c_k aus der Linearkombination gewichtet. Die Gesamtqualität kann somit sehr einfach berechnet werden,

indem für jede einzelne Verteilung $x_i = \mu_k$ gesetzt wird. Durch die Gleichsetzung ergibt sich eine Art Durchschnittsvarianz der Linearkombination, die sich umgekehrt proportional zur Konfidenz verhält.

Um später den Simulationsalgorithmus in Programmcode umsetzen zu können, ist es sinnvoll, ihn in einer kompakten mathematischen Form darzustellen. Als mögliches Modell, welches berechnet werden soll, könnte z.B. das MPP aus Abbildung 5.1 dienen.

Als Einstiegspunkt für die folgende Herleitung dient auch an dieser Stelle wieder der Viterbi-Algorithmus, der über die Maximierung der Wahrscheinlichkeiten die "wahrscheinlichste" Zustandsfolge ermittelt. Der Viterbi-Algorithmus wird nun in der Hinsicht abgewandelt, dass der verallgemeinerte Pfad durch das Modell aufgrund des Simulationsansatzes bereits bekannt ist. Außerdem soll es möglich sein, bei Vorhandensein mehrerer verallgemeinerter Alternativpfade innerhalb eines Zweiges eine Gesamtwahrscheinlichkeit aus allen zusammengehörigen verallgemeinerten Pfaden zu ermitteln, so dass der Simulationsalgorithmus sowohl für "einfache"Topologien, wie der oberen Topologie aus Abbildung 5.1, als auch für Topologien mit Skips angewendet werden kann. Mit Hilfe der simulierte Observationssequenz $\mathcal{O}_{simul}$ lässt sich die Wahrscheinlichkeit bei einem gegebenen HMM λ folgendermaßen darstellen:

$$\mathcal{P}(\mathcal{O}_{simul}|\lambda) = \mathcal{P}(\mathcal{O}_{simul}|\vec{\pi}, \mathbf{A}, \mathbf{B}) \tag{4.1}$$

Wird die Sichtweise auf ein HMM insoweit verändert, dass z.B. ein lineares HMM mit einem Skip, wie es in Abbildung 3.6 dargestellt ist, durch Auflösung des impliziten Datensharings aus zwei Sub-HMMs besteht (Abbildung 4.3), lässt sich 4.1 umformen zu (Jay et al., 2001):

$$\mathcal{P}(\mathcal{O}_{simul}|\lambda) = \sum_{i}^{N} \mathcal{P}(\mathcal{O}_{simul}|\lambda_i) \tag{4.2}$$

wobei die λ_i nun Sub-HMMs darstellen, die innerhalb eines komplexen HMMs eingebettet sind. Die Berechnung des wahrscheinlichsten Topologiezweiges, also die Berechnung des geeignetsten Sub-HMMs λ_p wird durch eine Maximums-

entscheidung ermittelt:

$$p = \arg\max_i \mathcal{P}(\mathcal{O}_{simul}|\lambda_i) \tag{4.3}$$

Der entscheidende Unterschied zum Viterbi-Algorithmus ist, dass die Zustandfolge, die zur maximalen Wahrscheinlichkeit führt, nicht mehr gesucht wird, sondern als Simulationsfolge vorgegeben wird. Dadurch lässt sich die Wahrscheinlichkeit eines Sub-HMMs folgendermaßen berechnen:

$$\mathcal{P}(\mathcal{O}_{simul}|\lambda_l(\pi_i^l, a_{ij}^k, b_j^l(\vec{x} = \vec{\mu})) =$$

$$\frac{1}{M}\sum_{l=1}^{M}\left(\prod_{i,j=1,i\neq j}^{q_N}\pi_i^l a_{ij}^l b_j^l \cdot \frac{L-q_N}{q_N}\sum_{i,j=1;i=j}^{q_N} a_{ij}^l b_j^l\right) \tag{4.4}$$

Eine besondere Erklärung benötigt der Term $\frac{L-q_N}{q_N}\sum_{i,j=1;i=j}^{q_N} a_{ij}^l b_i^l$. Im Abschnitt zur Motivation des Algorithmus wurde bereits angedeutet, dass bei einer gegebenen Phonsequenzlänge nicht bekannt ist, an welcher Stelle während der Simulation die benötigten Loops im Modell durchzuführen sind. Um den Berechnungsaufwand für alle Kombinationen zu verringern, wird an dieser Stelle die Näherung über den arithmetischen Durchschnitt angesetzt, wobei L die durchschnittliche Sequenzlänge eines zu modellierenden Phons und q_N die Anzahl der emittierenden Zustände in dem jeweiligen Sub-HMM beschreibt. Der Term führt zusätzlich eine Längennormierung aus, die für den Vergleich von HMMs unterschiedlicher Länge notwendig ist.

Jeder verallgemeinerte Pfad, der durch ein beliebiges Modell möglich ist, kann einzeln berechnet werden. Anschließend können komplexere, zusammengehörige Strukturen durch Mittelwertbildung zusammengefasst werden.

Wenn der Simulationsalgorithmus auf das Modell 5.1 angewendet wird, ergibt sich folgendes Berechnungsverfahren: das Modell verfügt über zwei zu bewertende Zweige, die im unteren Zweig (alle Wahrscheinlichkeiten ohne $*$), untereinander korrelierte Zustände zeigen. Berechnet werden Simulationswahr-

scheinlichkeiten für vier Topologien, bei einer gegebenen Phonsequenzlänge. Anschließend werden die Ergebnisse für die $*$ Topologien zusammengefasst, so dass es möglich wird, die beiden Zweige qualitativ miteinander zu vergleichen.

Die einzelnen Terme der Formel 4.4 sind:

- Durchschnittliche Loopwahrscheinlichkeit des l-ten Zweiges oder eines Sub-Sub-HMMs

$$\frac{L - q_N}{q_N} \sum_{i,j=1;i=j}^{q_N} a_{ij}^l b_j^l \tag{4.5}$$

- Die Restlängennormierung, die über q_N hinausgeht

$$L - q_N \tag{4.6}$$

- Die Wahrscheinlichkeit für den jeweiligen Zweig $(\cdot)^l$ ohne Berücksichtigung von Loops

$$\prod_{i,j=1,i\neq j}^{q_N} \pi_i^l a_{ij}^l b_j^l \tag{4.7}$$

- Die durchschnittliche Wahrscheinlichkeit der Zweige l eines definierten Sub-HMMs, mit M verallgemeinerten Pfaden

$$\frac{1}{M} \sum_{l=1}^{M} (\cdots) \tag{4.8}$$

Im folgenden Kapitel wird der Simulationsalgorithmus in Programmcode umgesetzt.

Kapitel 5

Implementation des MPP-Simulationsverfahrens

Die Implementation wird in der Programmiersprache Perl durchgeführt. Die Wahl der Programmiersprache ist abhängig von der Problemstellung. Die in der Arbeit vorliegenden Hidden-Markov-Modell-Dateien können im ASCII Format erzeugt werden. Die Programmiersprache Perl stellt für die Verarbeitung von Text-Dateien eine geeignete und vielfältige Umgebung bereit.

Die Implementation gliedert sich in zwei Teile:

1. Die Vorverarbeitung der vom Tool `HTK` erzeugten Metadateien, in denen die Modelle abgelegt sind. Das Modul stellt die Modelle topologieunabhängig zur analytischen Weiterverarbeitung zur Verfügung. Gleichzeitig erlaubt es unter gegebenen Randbedingungen (phonspezifische Topologielänge, Anzahl und Orte von Skips) neue Modelle zu generieren, und diese zur Weiterverarbeitung in `HTK` zu speichern. Dieser Programmcode wird nicht explizit vorgestellt, da er zum wissenschaftlichen Teil dieser Arbeit keinen Beitrag leistet.

2. Die Analyse und anschließende Simulation der trainierten Hidden-Markov-Modelle. Zur Analyse zählen das Auffinden von möglichen gültigen verallgemeinerten Pfaden durch beliebige Modelltopologien und das

anschließende Berechnen der simulierten Gesamtwahrscheinlichkeit einzelner MPP-Zweige.

Das unter 2. beschriebene Programm stellt den Implementationskern dieser Arbeit dar und wird zum besserem Verständnis als *Pseudocode* dargestellt.

Im wesentlichen wird der im vorangegangenen Kapitel beschriebene Simulationsalgorithmus implementiert. Als Beispiel dient das in 5.1 dargestellte Modell. Das Beispielmodell fordert, dass das Programm in der Lage ist, die Berechnung über beliebige parallele HMM-Topologien, die zusätzlich sogar Skips aufweisen können, ausführen zu können.

Praktisch werden jeweils nur zwei parallele Topologien untersucht, da sonst die Gefahr besteht, dass auf einzelnen Zweigen Datenknappheit eintritt und so die Ergebnisse verfälscht werden. Ganz ausschließen lässt sich dieser Fall auch bei zwei parallelen Topologien nicht. Jedoch wird angenommen, dass, wenn zwei ähnliche Topologien, z.B. eine lineare Zwei-Zustands- und eine lineare Drei-Zustands-Topologie, verglichen werden, die Gefahr zu wenig Daten vorrätig zu haben klein bleibt und sich die Wahrscheinlichkeitsmasse eher gleichmäßig auf die Topologien verteilt.

Natürlich sprachlich beschrieben wird ein beliebiges Modell ohne Berücksichtigung der Loops nach gültigen verallgemeinerten Pfaden rekursiv durchsucht. Die rekursive Suche ist nötig, da die einzelnen verallgemeinerten Pfade eines Modells dem Programm unbekannt sind. Verschiedene zu untersuchende Topologiezweige werden im Programm global initialisiert. Anschließend werden alle ermittelten verallgemeinerten Pfade auf die gewünschte Phonsequenzlänge normiert. Nachdem die einzelnen Topologiewahrscheinlichkeiten auf die zu bewertenden Zweige zurückgeführt werden, liegt das Ergebnis vor.

Schematisch lässt sich das Programm in folgende Module unterteilen,

1. Die Hauptfunktion zur Suche gültiger Wege.

(`Function SearchValidPaths`; Abbildung 5.3)

2. Unterfunktion zur Suche gültiger Nachfolger.
   (`Function FindSuccessors`; Abbildung 5.4)

3. Berechnung der simulierten Erkennungswahrscheinlichkeiten.
   (`Function CalculateProbs`; Abbildung 5.5)

4. Ermittlung der vorhandenen Loops eines verallg. Pfades und Normierung der Zweiglängen in Abhängigkeit von dem untersuchten Phon.
   (`Function CalculateLoops`; Abbildung 5.6)

Die rekursive Suche wird als Breitensuche über eine beliebige Transitionsmatrix (5.2) implementiert, indem in jeder Zeile nach gültigen Nachfolgern ($a_{ij} > 0$) gesucht wird. In der Transitionsmatrix sind die Übergangswahrscheinlichkeiten des Modells in Matrixform angegeben.

Der normalerweise größere Speicheraufwand der Breitensuche gegenüber einer Tiefensuche kann hier wegen der geringen Komplexität des Problems vernachlässigt werden. Weiterhin kann der Vorteil der Tiefensuche, nämlich vorzeitig ermittelte verallgemeinerte Pfade zu berechnen und sie dann aus dem Speicher zu entfernen, nicht genutzt werden, da zur Evaluation zweier Topologien nicht nur Gesamtwahrscheinlichkeiten sondern auch die Eigenschaften der optimierten Topologien benötigt werden.

Der Berechnungsaufwand der Suche reduziert sich gegenüber ergodischen Topologien dadurch, dass keine "Rückschritte" in der Spracherkennung zulässig sind. Es müssen demnach nur maximal $Dim([T])^2/2$ Suchschritte durchgeführt werden.

Als Beispiel dient die in Abbildung 5.2 dargestellte Transitionsmatrix, die die Übergangswahrscheinlichkeiten des Modells 5.1 aufweist. Sie zeigt ein paralleles Modell mit drei und zwei Zuständen inklusive eines Skips über den ersten Zustand des 3er-Modells und einem vorzeitigen "Ausstieg" aus dem 3er-Modell vom ersten Zustand aus. Die erste und letzte Spalte kennzeichnen die Zustände ohne Emissionen. Die Matrix ist mit Hilfe der senkrechten Gerade in zwei Modellzweige unterteilt, die berechnet werden sollen. Da der Transitionsmatrix

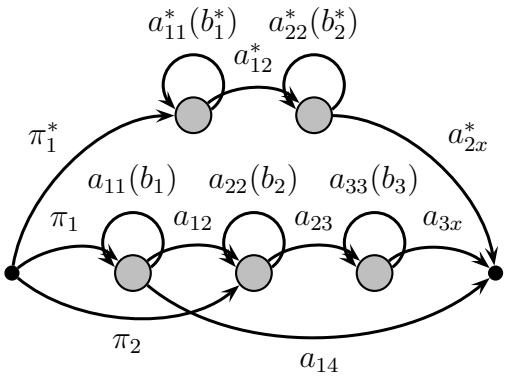

Abbildung 5.1: *MPP-HMM mit zwei Topologien*

nicht zu entnehmen ist, dass es sich um zwei zu evaluierende Topologiezweige handelt, wird die Zustandsanzahl des in der Matrix vorne liegenden Zweiges im Programm global initialisiert.

```
0.0 0.5 0.3 0.0 | 0.2 0.0 0.0
0.0 0.4 0.4 0.0 | 0.0 0.0 0.2
0.0 0.0 0.6 0.4 | 0.0 0.0 0.0
0.0 0.0 0.0 0.6 | 0.0 0.0 0.4
0.0 0.0 0.0 0.0 | 0.6 0.4 0.0
0.0 0.0 0.0 0.0 | 0.0 0.6 0.4
0.0 0.0 0.0 0.0 | 0.0 0.0 0.0
```

Abbildung 5.2: *Typische Transitionsmatrix für zwei parallele Topologien*

5.1 Funktion SearchValidPaths

- **Zeile 2-4**
  Die Datenstrukturen `acc` und `paths` werden mit der leeren Menge initialisiert. Der Akkumulator `acc` dient im Verlauf der Anwendung Zwischenschritte aufzuheben. Die Datenstruktur `paths` nimmt im Laufe der Anwendung relevante Informationen aller möglichen verallgemeinerten

```
Function SearchValidPaths

 1 Begin
 2 var acc ← ∅;
 3 var paths ← ∅;
 4 var T ← [T²]ransitions;
 5 Initialisation
 6 for n ← 0 to Dim(T_x) − 1 do
 7   if a_0n ∈ T > 0 do
 8     acc ← acc ∪ (var i = 0, var j = n, var path ← (0, n));
 9   end if
10 end for
11 End of Initialisation
12 while acc do
18   var successes ← ∅;
19   var x ← ∅;
20   foreach (i, j, path) in acc do
21     x ← FindSuccessors (i, j, path);
22     if x ≠ ∅ do
23       successes ← successes ∪ x;
24     else
25       paths ← paths ∪ (i, j, path);
26     end if else
27   end foreach
28   acc ← successes;
29 end while
30 Return paths
31 End
```

Abbildung 5.3: *Gültige verallgemeinerte Pfade durch das gesamte Modell suchen*

Pfade zunächst unabhängig von der untersuchten Topologie auf. Hierzu gehört die Pfadhistorie, der aktuelle Suchknoten, die Länge des verallg. Pfades in Zuständen und die durch die Simulation erzeugten Wahrscheinlichkeitsdichten. Die Transitionsmatrix für das jeweils zu untersuchende Phonmodell wird in `T` initialisiert.

- **Zeile 6-10**
  Der Akkumulator `acc` wird mit denjenigen nichtverschwindenden Koeffizientenindizes belegt, die aus der ersten Zeile der Transitionsmatrix errechenbar sind. In der ersten Zeile der Transitionsmatrix befinden sich die Eintrittswahrscheinlichkeiten π_i. Weiterhin wird die verallg. Pfadhistorie des gerade untersuchten verallg. Pfades in `path` ebenfalls mit den nichtverschwindenden Koeffizientenindizes initialisiert.

- **Zeile 12-29**
  Solange nichtverschwindende Einträge in der Transitionsmatrix eines verallg. Pfades durch die Funktion `FindSuccessors` gefunden werden, werden diese zunächst in `successes` abgelegt. Durch i, j wird die jeweils letzte nichtverschwindende Indexkombination übergeben. Von dieser Information aus werden anschließend Nachfolger gesucht. Die Suche terminiert, wenn keine weiteren nachfolgenden verallg. Pfadkomponenten gefunden werden. In diesem Fall wird der komplette verallg. Pfad in der Datenstruktur `paths` abgelegt.

5.2 Funktion FindSuccessors

```
Function FindSuccessors  i,j,path

 1 Begin
 2 var T ← [T²]ransitions;
 3 var result ← ∅;
 4 for n ← j + 1 to Dim(Tx) − 1 do
 5   if ajn ∈ T > 0 do
 6     var oldpath ← ∅;
 7     var oldpath ← oldpath ∪ path ⊕ (j,n);
 8     result ← result ∪ (i ← j, j ← n, oldpath);
 9   end if
10 end for
11 Return result
12 End
```

Abbildung 5.4: *Funktion zur Nachfolgebestimmung*

- **Zeile 2-3**
  Um nichtverschwindende Folgekoeffizienten eines verallgemeinerten Pfades aufzufinden, steht der Funktion die Transitionsmatrix in `T` zur Verfügung. Der Rückgabewert `result` der Funktion (eine Datenstruktur, die die verallg. Pfadfolgeinformationen enthält) wird durch die leere Menge initialisiert.

- **Zeile 4-10**
  Während der rekursiven Suche werden mögliche Loops $a_{i=j}$ vorerst nicht

behandelt, da es zu Terminierungsproblemen führen würde. Aus diesem Grund wird in der auf i folgenden Zeile j der Matrix nach den folgenden nichtverschwindenden Einträgen gesucht, die ab $j+1$ beginnen. Die einzelnen Historien `path` werden in `oldpath` abgelegt und um gültige Nachfolger erweitert. Dadurch besteht die Möglichkeit, falls neue Wegvarianten gefunden werden, den alten Anfangszweig zu kopieren und damit eine zusätzliche Historie zu öffnen. Zuletzt wird die erweiterte Gesamtstruktur durch `result` zurückgeliefert.

5.3 Funktion CalculateProbs

Im Folgenden wird der Pseudocode für die Wahrscheinlichkeitsberechnung der einzelnen verallg. Pfade dargestellt. Im realen Code werden die Loop-Wahrscheinlichkeiten durch eine gesonderte Methode eingebettet ermittelt und die Topologien auf die richtige Länge normiert. Für eine übersichtlichere Darstellung werden die beiden Programmteile getrennt dargestellt. Die Funktion CalculateProbs errechnet somit nur den linearen Durchgang durch einen gegebenen verallgemeinerten Pfad ohne Berücksichtigung möglicher Loops.

- **Zeile 2-4**
  Zunächst werden im Initialisierungsteil die Transitionsmatrix `T`, die Parameter der Wahrscheinlichkeitsdichtefunktionen `B` und die zugehörigen Gewichtungskoeffizienten `C` zur Verfügung gestellt.

- **Zeile 5-20**
  Die Struktur `paths` enthält Unterstrukturen, in denen Historie (Variable `path`) und Werte der gefundenen verallgemeinerten Pfade gespeichert sind. Um auf diese Unterstrukturen zugreifen zu können, wird in Zeile 5 die Hilfsvariable `datastruct` jeweils mit einer vorhandenen Unterstruktur belegt. Für jede Unterstruktur `datastruct` des Modells (Zeile 5) wird die temporäre Variable `p`, die die pfadspezifischen Wahrscheinlichkeitsdichtewerte der Simulation aufnehmen wird, mit dem neutralen Element der Multiplikation initialisiert (Zeile 6).

```
Function CalculateProbs  paths

 1 Begin
 2 var T ← [T²]ransitions;
 3 var B ← [D]ensities;
 4 var C ← [M²]ixtures;
 5 foreach datastruct in paths do
 6 var p ← 1;
 7   foreach a_ij ∈ T in path ∈ datastruct do
 8     var Density ← 0;
 9     foreach c_jk ∈ C in path do
10       Density ← Density +c_jk · b_j(x⃗ = μ⃗) ∈ B
11     end foreach
12     if j < Dim(T_x) − 1 do
13       p ← p ·a_ij· Density
14     else
15       p ← p ·a_ij
16     end if else
17   end foreach
18   paths ← datastruct ∪ p
19 end foreach
20 Return paths
21 End
```

Abbildung 5.5: *Wahrscheinlichkeitsberechnung*

– **Zeile 7-8**
Der Dichtewert `density` einer Linearkombination von Gaußschen Verteilungsfunktionen eines gerade untersuchten emittierenden Zustandes (Mixturen) wird mit dem neutralen Element der Summation initialisiert.

– **Zeile 9-11**
Der Dichtewert einer Linearkombination wird in diesen Zeilen für einen emittierenden Zustand errechnet. Falls der Zustand über keine Mixturen $\mathtt{c_{jk}}$ verfügt, ist `c` an dieser Stelle 1.

– **Zeile 12-16**
Solange nichtverschwindende emittierende Zustände innerhalb der verallg. Pfadhistorie existieren, wird iterativ die Gesamtwahrscheinlichkeit `p` des gerade untersuchten verallg. Pfades errechnet. Nachdem der letzte emittierende Zustand erreicht wird, wird die Modellaustrittswahrscheinlichkeit ebenfalls aufmultipliziert (Zeile 15).

– **Zeile 18**
Die errechnete Gesamtwahrscheinlichkeit des untersuchten verallg. Pfades wird in die zugehörige Datenstruktur eingepflegt.

5.4 Funktion CalculateLoops

```
Function CalculateLoops  paths

 1 Begin
 2 var T ← [T²]ransitions;
 3 var B ← [D]ensities;
 4 var C ← [M²]ixtures;
 5 var realduration ← Realdurationof(Phon);
 6 foreach datastruct in paths do
 7   datastruct ← datastruct ∪ var loops ← ∅
 8   var numberOfLoops ← 0;
 9   var loopAvg ← 0;
10   foreach a_ij ∈ T in path ∈ datastruct do
11     if a_jj ∈ T > 0 do
12       loops ← loops ⊕ (jj);
13       var LoopDensity ← 0;
14       foreach c_jk ∈ C in path do
15         LoopDensity ← LoopDensity +c_jk · b_j(x⃗ = μ⃗) ∈B
16       end foreach
17       loopAvg ← loopAvg + a_jj· LoopDensity ;
18       numberOfLoops ← numberOfLoops +1;
19     end if
20   end foreach
21   loopAvg ← loopAvg / numberOfLoops
22   p ∈ datastruct ← p · (realduration-lenghtOfPath) · loopAvg
23 end foreach
24 Return paths
25 End
```

Abbildung 5.6: *Berechnung der Loopdurchschnitte*

- **Zeile 2-5**
Ebenso wie in der Funktion `CalculateProbs` werden anfangs die Transitionsmatrix `T`, die Parameter der Wahrscheinlichkeitsdichtefunktionen `B` und die zugehörigen Gewichtungskoeffizienten `C` zur Verfügung gestellt. Zusätzlich wird die durchschnittliche Sequenzlänge `realduration` eines

Phons initialisiert, um die oben beschriebene Längennormierung durchzuführen.

- **Zeile 6-24**
  Jeder gültigen Unterstruktur `datastruct` wird in Zeile 7 eine Variable `loop` zugeordnet, die gültige Loops aufnimmt. Außerdem werden eine Variable `numberOfLoops` als Loopzähler und die Variable `loopAvg` initialisiert. Die errechnete durchschnittliche Loopwahrscheinlichkeit wird in `loopAvg` gespeichert.

  - **Zeile 10-12**
    Für jeden Transitionskoeffizienten innerhalb eines verallg. Pfades wird überprüft, ob eine Loopwahrscheinlichkeit a_{jj} existiert. Falls dies der Fall ist, wird diese in die Datenstruktur mit aufgenommen (Zeile 12).
  - **Zeile 13-16**
    Für die in Zeile 11 ermittelte Loopwahrscheinlichkeit wird die Gesamtwahrscheinlichkeit errechnet und in `LoopDensity` festgehalten.
  - **Zeile 17-18**
    Während der Loopzähler erhöht wird, wird die Gesamtwahrscheinlichkeit aller Loops aufsummiert.
  - **Zeile 21-22**
    Nachdem alle zugehörigen Loops gefunden und aufsummiert wurden, wird in Zeile 21 die durchschnittliche Loopwahrscheinlichkeit berechnet. Sie wird in Zeile 22 unter Zuhilfenahme der durchschnittlichen Sequenzlänge des gerade untersuchten Phons benutzt, um schließlich die normierte Gesamtwahrscheinlichkeit des untersuchten Zweiges zu errechnen. Die Variable `lenghtOfPath` ist in der Datenstruktur `datastruct` während der Funktion `CalculateProbs` gepflegt worden und beschreibt die lineare der Topologie ohne Loops.

Die Ausgabe des Programms ist sehr flexibel. Es können jeweils die simulierten logarithmischen Wahrscheinlichkeiten der zu vergleichenden Topologien ausgegeben werden. Außerdem ist es möglich, Wahrscheinlichkeitsdichten einzelner Phasen der Simulation quantitativ zu untersuchen.

Das Programm erlaubt, beliebig komplexe Topologien zu evaluieren, die dem Merkmalsfluss von links nach rechts folgen. In jedem Fall werden die Topologien für die Weiterverarbeitung nach einem einfachen k.o.-Kriterium ausgewählt und weiterverarbeitet.

Kapitel 6

Experimente

Alle Experimente, die in diesem Kapitel beschrieben werden, sind in ihrer jeweiligen Experimentierreihe mit jeweils konstanten Parametern betrieben worden. In allen Experimenten werden die Suchparameter Pruning (Wert: 100.0), Worteinfügungsstrafe (Wert: 0.0) und der sog. Grammar-Scale-Faktor (Wert: 6.5) konstant gehalten.

Der Pruning-Faktor begrenzt den Suchraum, indem Suchpfade, bzw. Modelle im Suchraum ignoriert werden, die unterhalb eines bestimmten Schwellwertes fallen. Der Pruning-Faktor macht sich deshalb zunächst in der Performanz der Suche bemerkbar, reduziert jedoch mit zunehmender Größe erheblich die Erkennungsrate.

Liegen aufgrund der Suche schlechte Worterkennungswahrscheinlichkeiten vor, die dem Sprachmodell "unterliegen", werden aufgrund der Wahrscheinlichkeiten des Sprachmodells Worte während der Erkennung eingefügt. Um diesen Effekt zu unterbinden kann in `HTK` die sog. Worteinfügungsstrafe mit einem Wert belegt werden.

Der sog. Grammar-Scale-Faktor balanciert den probabilistischen Einfluss des Sprachmodells gegenüber den ermittelten Wahrscheinlichkeiten der reinen Wortklassifizierung.

Die Werte für das Pruning, die Worteinfügungsstrafe und der den Grammar-Scale-Faktor stammen aus Experimenten, die während des Projektes "Verbmobil" (Wahlster, 2000) ermittelt wurden. Sie werden von den meisten Wissenschaftlern weiterverwendet, auch wenn es möglich wäre, durch Veränderungen dieser Parameter Spracherkennungsergebnisse auf dem Verbmobilkorpus zu verbessern. Dadurch ist die Möglichkeit des wissenschaftlichen Vergleichs gewährleistet.

Die Experimente werden auf dem Teilkorpus Verbmobil I [VM1] und auf dem gesamten Korpus [VM2] des Verbmobil Projektes durchgeführt (Details: siehe Anhang A). Außerdem beschränken sich die experimentellen Untersuchungen auf monophone Hidden-Markov-Modelle.

Auf die Modellierung der Pausenmodelle `sil` (silence) und `sp` (short pause) wird an dieser Stelle nicht weiter eingegangen, da ihre Topologien als konstant festgelegt werden. Die Vorstellung der Pausenmodelle erfolgt im Anhang D.

Als Entwicklungsdatensatz dient in der vorliegenden Arbeit das sog. Dev-Set (Development Set), welches während eines Entwicklungsprozesses herangezogen wird, um nicht in Gefahr zu laufen, auf dem Evaluationsset (Eval-Set) zu optimieren. Beide Daten-Sets sind durch das Verbmobilprojekt vorgegeben. Normalerweise wäre es notwendig am Ende der Arbeit alle Experimente auf dem Eval-Set zu wiederholen, um auszuschließen, dass das Dev-Set gelernt wird und die Erkennungsergebnisse auf dem Eval-Set stark zurückbleiben. Da aber die verschiedenen Topologieexperimente im Wesentlichen untereinander verglichen werden, um das Simulationsverfahren experimentell zu untermauern, spielen Erkennungsergebnisse für die "Machbarkeit" zunächst eine untergeordnete Rolle.

Zusätzlich wird der Sachverhalt dadurch erschwert, dass keine weiteren Eval-Sets zur Verfügung stehen, da das Verbmobil Projekt abgeschlossen ist. Sollen weitere Experimente folgen, tritt der Umstand ein, dass nach einmaliger Überprüfung des Ansatzes auf dem Eval-Set dieses verbraucht ist und kein Ersatz

geschaffen werden kann. Aus diesen Gründen wird auf den Test mit dem Eval-Set in dieser Arbeit verzichtet.

6.1 Bewertung der Spracherkennerergebnisse

Grundsätzlich wurden zur Bewertung der Spracherkennungsergebnisse die sogenannte Wortakkuratheiten herangezogen. Sie stellt das Maß dar, wieviele Worte des Dev- oder Evaluationsset tatsächlich richtig erkannt werden. Dieses Maß wird entweder als "Erkennungsrate" oder als "Fehlerrate" verwendet. In der vorliegenden Arbeit wird die Wortfehlerrate [WFR] in Tabellen und Darstellungen verwendet. Errechnet wird die Wortfehlerrate durch den DTW-Algorithmus unter Verwendung des Referenzmaterials aus dem Dev-Set, indem eine Abbildung des erkannten Satzes auf dem Referenzsatz erzwungen wird. Die für diese Abbildungen notwendigen Einfügungen, Ersetzungen und Wortweglassungen werden von der Anzahl der möglichen Worte abgezogen. Der reziproke Wert ergibt dann die Wortfehlerrate.

Ein weiterer wichtiger Gesichtspunkt ist, mit wievielen Wahrscheinlichkeitsdichteverteilungen die Zustände der einzelnen Modelle modelliert wurden. Die Modellierungsgenauigkeit ist abhängig von der Anzahl der Funktionen einer Linearkombination (Gleichung 2.3).

Es ist nur möglich Ergebnisse miteinander zu vergleichen, die entweder in ihrer Erkennungsleistung konvergierten oder solche, die bis zu einer vereinbarten Anzahl von Mixturen trainiert wurden. Letzteres lässt jedoch nur einen punktuellen Vergleich zu und kann keine Aussage darüber treffen, ob die verglichenen Systeme zu einem Zeitpunkt innerhalb des Trainings andere Ergebnisse liefern würden.

Außerdem ist es notwendig, das sog. Signifikanzniveau zu ermitteln. Es lässt sich mit Hilfe eines von Jeff Bilmes implementierten Tools (Anhang E) errechnen, dass im Bereich um 30% Wortfehlerrate, Differenzen von 0,5% noch als signifikante Veränderungen angesehen werden können.

6.2 Vorbereitung der Sprachdaten

In der internationalen Spracherkennungsgemeinschaft haben sich die folgenden Extraktionsparameter durchgesetzt (Tabelle: 6.1), die auch in dieser Arbeit Verwendung finden. Der extrahierte Merkmalvektor beinhaltet zunächst 12 Mel-Cepstrum-Koeffizienten (MFCC) und einen Energiekoeffizienten **E**. Die zeitliche Änderung des Sprachsignals wird mit Hilfe der jeweils ersten und zweiten Ableitung $\Delta; \Delta^2$ der MFCC-Koeffizienten einschließlich der Energie modelliert. Technisch werden die jeweiligen Ableitungen iterativ mit Hilfe der linearen Regression ermittelt, wobei die Anzahl der Stützstellen per Standardeinstellung mit 2 gewählt wird. Somit werden aus dem gesamten akustischen Material alle 10ms 39 korrelierte Merkmale ermittelt, die als Merkmalvektoren bezeichnet werden. 48 Stunden Sprachmaterial lassen sich so durch ca. 17 Mio. Merkmalvektoren repräsentieren. Die Extraktion erfolgt mittels des Programms `HCopy` [1] (Young et al., 2002).

Akustik	Sample freq.	16 kHz
	Fensterbreite	20 ms
	Fenstervorschub	10 ms
Merkmalsvektor		$(12\ MFCC + 1Energie), +\Delta, +\Delta^2$

Tabelle 6.1: *Extraktionsparameter der Merkmalvektoren*

Für das Teilprojekt VM1 liegen Informationen über phonetische Segmentgrenzen vor, die zur Initialisierung einzelner phonetischer HMMs genutzt werden können. Die Segmentierungen sind mit Hilfe des MAUS Systems (Münchener AUtomatisches Segmentationssystem) (Schiel, 1999, Beringer and Schiel, 2000) im Kontext des Verbmobil-Projekts durchgeführt worden. Mit Hilfe der Programme `HInit, HRest` [2] [3] (Young et al., 2002) werden zunächst für jede benötigte Phontopologie initiale HMMs errechnet.

Es werden lineare HMMs mit jeweils ein bis fünf emittierenden Zuständen

[1] `HCopy -c config -f -T sourcefiles targetfiles`

[2] `HInit -T 1 -i 20 -C configfile -S trainfiles -H prototyp_dir -M out_dir -l HMM_Name -I seg_phonfile prototypfile`

[3] `HRest -T 1 -i 60 -C configfile -S trainfiles -H prototyp -M out_dir -l HMM_Name -I seg_phonfile prototypfile`

errechnet. Aus diesen initialen Modellen können alle benötigten systematischen Referenztopologien und die für die Experimente notwendigen MPPs durch das Programm aus Abschnitt 5.1 erzeugt werden.

Weiterhin werden mehrere Sprachmodelle auf den Transkriptionen der Trainingsdaten von VM1 und VM2 mit Hilfe der Wortlisten aus den Dev-Sets trainiert. Es handelt sich dabei um Unigramme (Einzelwortfrequenzen) und Bigramme (Wortfolgemodelle). Während die Bigramme zur Bewertung aller Experimente herangezogen werden, werden die Unigramme nur stichprobenartig eingesetzt, um sicherzustellen, dass Veränderungen der Spracherkennungsergebnisse nicht aufgrund von verbesserter Anpassung an die Sprachmodelle erfolgen.

Die Perplexitäten der eingesetzten Bigramme sind in Tabelle 6.2 dargestellt.

In anderen Arbeiten, z.B. (Hauenstein, 1996) zur akustischen Topologiemodellierung, wird teilweise auf stochastische Wortfolgemodelle während der Erkennung verzichtet, da der Gewinn in der Spracherkennungsperformanz durch Verbesserung der akustischen Mustererkennung im Vordergrund steht und somit bereits mit einfachen Unigrammen bestimmbar ist. Um aber nicht auf die Erfahrung verzichten zu müssen, wie sich die verschiedenen Modellierungen auf vollständige Spracherkennungssysteme auswirken, werden bei den Experimenten dieser Arbeit Bigramme eingesetzt.

Word-Bigramm VM1	$\wp = 47.3$, $H = 5.56$
Word-Bigramm VM1+II	$\wp = 66.1$, $H = 6.05$

Tabelle 6.2: *Sprachmodellparameter*

6.3 Statistische Vorüberlegungen zur Parameterwahl

Ziel der Arbeit ist es unter anderem verschiedene Spracherkenner, die sich durch verschiedene Phontopologien auszeichnen, bis hin zu voll besetzten Kovarianzmatrizen zu trainieren, um das bestmögliche Erkennungsergebnis einer untersuchten Topologie zu erhalten.

Eine notwendige Voraussetzung hierfür ist es, noch genügend Merkmalvektoren zur Verfügung zu haben, um die dann $\frac{(39^2+3*39)}{2}$ Parameter pro Verteilung (Kovarianzmatrix und Mittelwertvektor) trainieren zu können.

Eine Abschätzung, wieviele Merkmalvektoren pro Mixtur notwendig sind, um die aufwendigen Einzelverteilungen noch trainieren zu können, stellt sich als Herausforderung dar, zumal unbekannt ist, ob die Merkmalvektoren innerhalb eines HMMs während des Trainings eher gleichmäßig auf alle einzelnen Verteilungen verstreut werden oder ob sie z.B. datenbedingt in wenigen speziellen Verteilungen kumulieren.

Um empirisch einen Näherungswert für die benötigte Anzahl von Merkmalvektoren zu erhalten, werden systematisch Experimente mit Hilfe von linearen Topologien mit voll besetzten Kovarianzmatrizen durchgeführt. Nach jedem Trainingsschritt wird die Anzahl der Gaußschen Mixturen pro Zustand systematisch erhöht. Damit erhöht sich ebenfalls die Anzahl der Verteilungen bei konstanter Menge an Merkmalvektoren. Es stellt sich heraus, dass die Zahl von ca. 600 Merkmalvektoren pro Verteilung gerade noch ausreicht, die komplexen Verteilungen zu trainieren. Bei einer durchschnittlichen Phonsequenzlänge von neun Frames sind demnach 200 phonetische Beobachtungen notwendig, um etwa die oben ermittelte Anzahl von Merkmalvektoren zu erreichen, die notwendig ist, um ein HMM mit drei emittierenden Zuständen ohne Berücksichtigung weiterer Mixturen trainieren zu können. Für die Experimente wird eine untere Grenze festgelegt, die fordert, eine bestimmte Anzahl von beobachteten Phonen vorrätig zu haben, um ein Modell unabhängig von seiner Zustandsanzahl um Mixturen erweitern zu können.

Am Beispiel des Diphthongs "OY", welches auf dem VM1-Trainingskorpus 2426-mal beobachtet werden kann, wird vorgeführt, wie während der Experimente das Erhöhen der Mixturen durchgeführt wird. Da die geforderte untere Grenze für Beobachtungen 200 ist, folgt, dass eine beliebige Topologie bis hin zu vier emittierenden Zuständen noch mit voll besetzten Kovarianzmatrizen trainierbar ist, wenn diese Topologie 12 mal gemischt wird. Diese Vorgehensweise hat zur Folge, dass kurze Topologien wie z.B. Zweier-Topologien

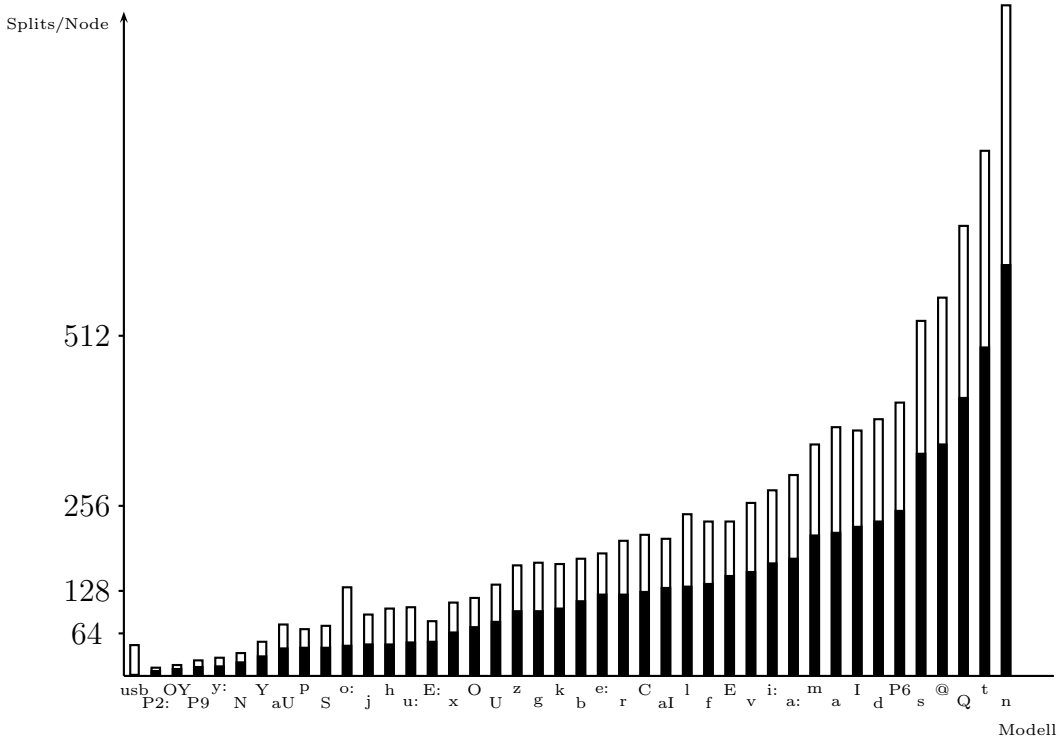

Abbildung 6.1: *Mixturen pro Zustand eines Phonmodells mit drei Zuständen*

insgesamt betrachtet über weniger Mixturen und längere Topologien über mehr Mixturen verfügen werden.

In Abbildung 6.1 wird das Ergebnis aus dieser Abschätzung dargestellt, indem die Gesamtanzahl der zu vergebenen Mixturen pro Zustand eines Dreier-HMMs eingetragen ist. Die schwarzen Balken markieren die maximal zulässige Anzahl von Mixturensplits für VM1, während der darüber liegende Rahmen die maximal mögliche Anzahl von Mixturensplits pro Zustand für VM2 darstellt.

Durch diese Abschätzung lässt sich schon vorab angeben, wieviele Mixturen für die Referenz-Gesamtsysteme maximal möglich sind. Die drei systematischen Baseline-Systeme (lineare Dreier-Topologie, BAKIS-Topologie und reduzierte BAKIS-Topologie) sind aufgrund ihrer Zustandsanzahl-Äquivalenz in

der Tabelle 6.3 zusammengefasst.

Referenzsystem	# VM1	# VM2
Wissensbasiert	12882	22272
Baseline-Systeme	17785	30151

Tabelle 6.3: *Maximal mögliche Mixturen pro Referenzsystem*

6.4 Das HMM-Trainingsverfahren

Alle Experimente werden nach dem nachfolgend beschriebenen Algorithmus systematisch durchgeführt, der zusätzlich in Abbildung 6.2 dargestellt ist.

1. Schritt:
   Alle initialen Phonmodelltopologien für das ausgewählte Experiment werden bereitgestellt. Sie weisen eine einfache diagonale Kovarianzmatrix auf.

2. Schritt:
   Abhängig von der Anzahl der zur Verfügung stehenden Beobachtungen (Abschnitt 6.3) eines Phons, werden die Mixturen je emittierendem Zustand pro Durchgang um Faktor zwei erhöht. Sollte ein Phonmodell über zu wenig Beobachtungen verfügen, wird die bereits erreichte Mixturenanzahl während des restlichen Trainings konstant gehalten.

3. Schritt:
   Ein Zyklus eines Baum-Welch-Trainings [BW-Training] [4]

4. Schritt:
   Evaluation[5] der in Schritt 3 trainierten Modelle. Solange die Wortfehlerraten sinken, weiter mit Schritt 3, sonst zurück zu Schritt 2 und Erhöhung der Anzahl der Mixturen.

[4] `HERest -C configfile -t 250.0 150.0 1000.0 -S trainfiles -H HMM_dir -M HMM_out_dir -I labefiles modelllist`

[5] `HVite -C configfile -t 100.0 -s 6.5 -X ext -S testfiles -l recogfiles_out_dir -w LM -H HMMs DEV_dictionairy modelllist`

5. Schritt:
   Die Erkennungsrate des Gesamtsystems wird global gespeichert, wenn in Schritt 4 das System konvergiert. Sollte in diesen global gespeicherten Erkennungsergebnissen keine Verbesserung mehr zu beobachten sein, werden aus den bis hierhin trainierten Modellen, Modelle mit voll besetzten Kovarianzmatrizen initialisiert.

6. Schritt:
   Ein Zyklus eines Baum-Welch-Trainings [BW-Training].

7. Schritt:
   Evaluation der in Schritt 6 trainierten Modelle. Solange die Erkennungsrate steigt, wird Schritt 6 wiederholt.

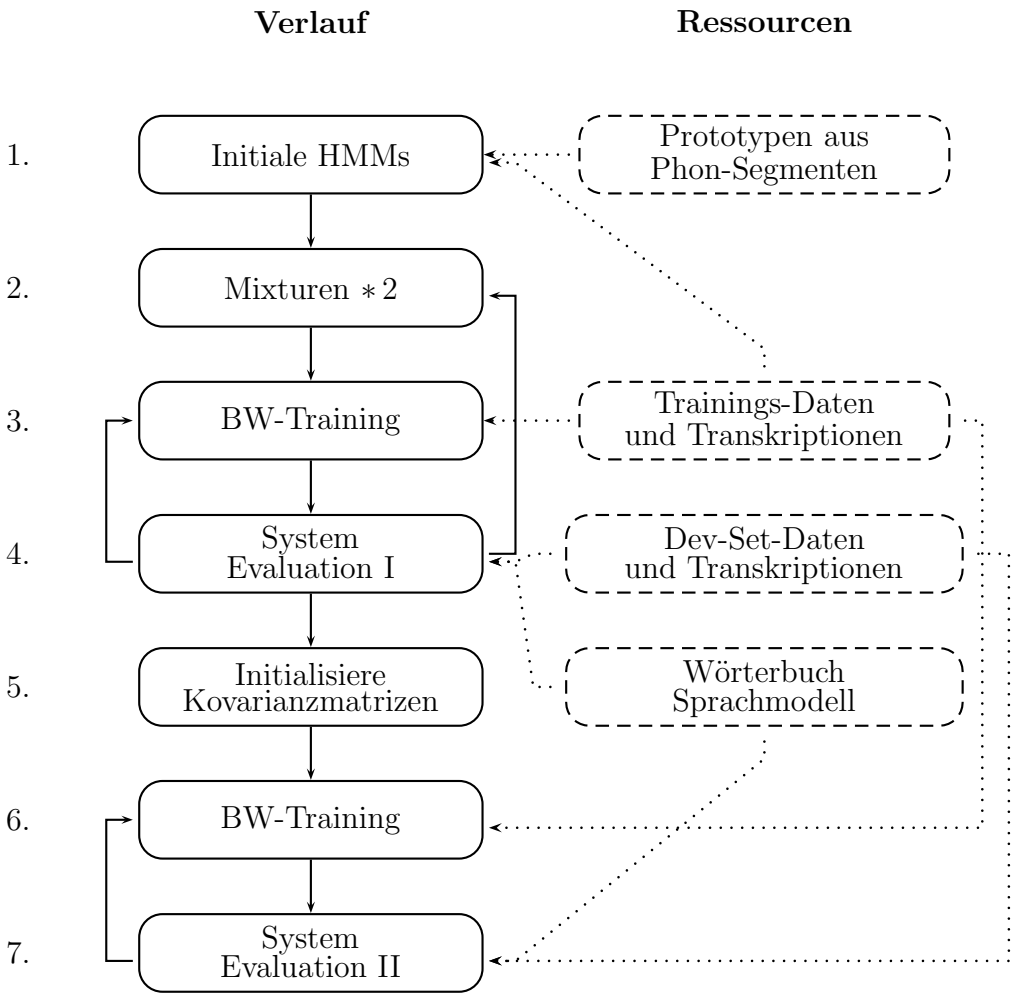

Abbildung 6.2: *Allgemeines Trainingsverfahren für verschiedene Hidden-Markov-Modell basierte Spracherkenner*

Zwischen Schritt 3 und 4 wird zusätzlich einmal ein "forced alignment" durchgeführt um bereinigte Trainingsdaten zu erhalten. Im Gegensatz zu der üblichen Anwendung des "forced alignment", in dem Aussprachevariationen des Wörterbuchs in den Label-Dateien aligniert werden, wird die Technik an dieser Stelle benutzt, um nicht konsistente Trainingsdateien auszusortieren.

Folgende HMM-Parameter werden während des Trainings geschätzt:

- Die Transitionswahrscheinlichkeiten: a_{ij}.
- Die Loopwahrscheinlichkeiten: a_{jj}.
- Die Parameter der Wahrscheinlichkeitsdichtefunktionen: μ_{jk}, Σ.
- Die Koeffizienten der Linearkombination: c_{jk}

Die Möglichkeit, alle Modelle zunächst mit diagonal besetzten Kovarianzmatrizen bis zur Konvergenz zu trainieren und anschließend voll besetzte Kovarianzmatrizen aus diesen Modellen zu initialisieren und zu trainieren, verschafft einen großen Zeitvorteil. Um sicherzustellen, dass die Erkennungsraten auch für voll besetzte Kovarianzmatrizen in den Experimenten dieser Arbeit nach jeder Mixturenerhöhung noch fortlaufend verbessert werden, wird einmalig ein entsprechendes Experiment auf den VM1-Daten durchgeführt. Der experimentelle Ablauf entspricht im Wesentlichen dem aus Abbildung 6.2 bis auf den Unterschied, dass jeder Mixturenschritt bis zur Konvergenz in der Erkennungsrate inklusive voll besetzten Kovarianzmatrizen trainiert wird.

Das Ergebnis ist in Abbildung 6.3 dargestellt. Die schwarzen Balken markieren die Wortfehlerraten der diagonal besetzten Systeme, während die Rahmen die Ergebnisse der Systeme mit Kovarianzmatrizen darstellen. Das Experiment bestätigt die Annahme, dass auch die Erkennungsleistung für voll besetzte Kovarianzmatrizen fortlaufend verbessert wird und weder stagniert noch rückläufig ist.

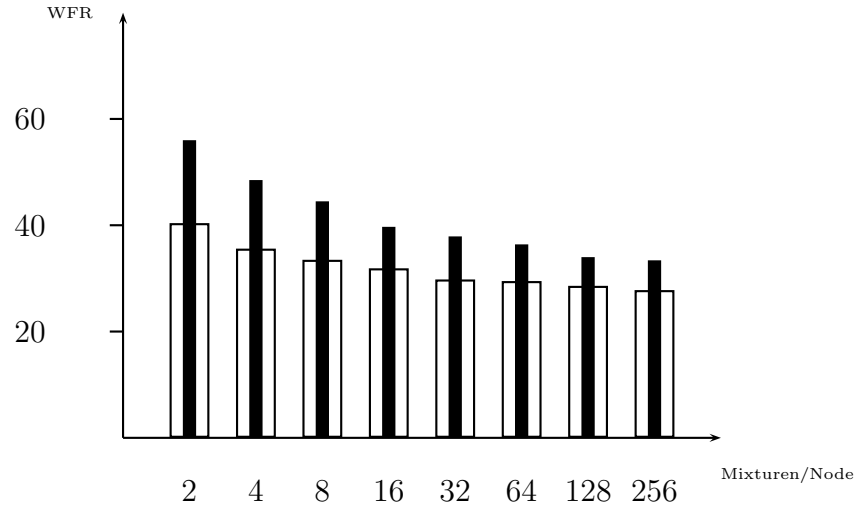

Abbildung 6.3: *Differenz der Erkennungsraten zwischen trainierten diagonal- und vollbesetzten Kovarianzmatrizen je nach Anzahl der Mixturen pro Zustand*

6.5 Referenzexperimente

Um den möglichen Gewinn des vorgestellten Algorithmus gegenüber heutigen Standardsystemen messen zu können ist es notwendig, verschiedene Referenzsysteme zu trainieren, die typische Topologien ausweisen, die heute in der Sprachererkennung eingesetzt werden. Grundsätzlich werden die verschiedenen Baseline-Topologien aus den initialen Phon-HMMs automatisch erzeugt. Es werden folgende Referenztopologien zur Untersuchung herangezogen.

6.5.1 Begründung für die Topologieauswahl

Die Topologieauswahl der Baseline-Systeme erfolgt hauptsächlich aufgrund der heutzutage in der wissenschaftlichen Forschung publizierten Topologien. Am häufigsten findet sich die lineare Dreier-Topologie, z.B. (Schukat-Talamazzini, 1995a) wieder, die nicht über die Fähigkeit verfügt, verschiedene Phonecha-

Arbeitstitel	Topologie
Baseline	drei emitt. Zustände; linear verknüpft
Reduzierte BAKIS	drei emitt. Zustände; linear verknüpft + ein Skip des Zentralphons
BAKIS	drei emitt. Zustände; Topologie nach BAKIS
Wissensbasiert	zwei-vier emitt. Zustände in Abhängigkeit der Phonetik; linear verknüpft
Statistisch	wie das Baseline-System, jedoch drei Längenklassen in Abh. von der Häufigkeit in den Trainingsdaten

Tabelle 6.4: *Topologien der Referenzmodelle*

rakteristika durch eventuell vorhandene implizite Topologien unterschiedlich zu modellieren.

Im Gegensatz hierzu steht das BAKIS-Modell mit drei Skips, welches aufgrund seiner Topologie je nach Datenlage sowohl phoninterne Variationen als auch unterschiedliche phonspezifische Zustandslängen modellieren kann. Es müssen jedoch mehr Parameter errechnet werden. Diese Topologie wird unter anderem beim Spracherkennungssystem der RTWH Aachen eingesetzt, welches im Zuge des Verbmobil-Projektes entwickelt wurde (Kanthak et al., 2000).

Eine Alternative zu den oben beschriebenen Systemen stellt der phonetisch wissensbasierte Topologieansatz dar, der aufgrund des Wissens über typische Phonlängen eine hybride Topologiemodellierung zur Verfügung stellt. Unter typischen Phonlängen werden hier die relativen Längenunterschiede während der sprachlichen Realisierung der Phone verstanden. So können kürzere lineare HMMs für kurze Vokale und Konsonanten gewählt werden. Lange Vokale werden mit einem zusätzlichen Zustand modelliert, Diphthonge mit zwei Zuständen zusätzlich, bis hin zu Geräuschen, die in diesem Fall mit doppelt so vielen Zuständen modelliert werden wie ein kurzer Vokal. In dieser Arbeit werden kurze Vokale und Konsonanten durch ein lineares HMM mit zwei Zuständen modelliert. Lange Vokale werden mit drei emittierenden Zuständen und Diphthonge bzw. Geräusche mit vier emittierenden Zuständen modelliert.

Als Alternative zur BAKIS-Topologie wird eine Topologie herangezogen, deren Herkunft nicht genau bekannt ist. Die einfache lineare Dreier-Topologie wird lediglich um einen systematischen Skip ergänzt, der den quasi-stationären Zustand des Modells überspringt und somit die Möglichkeit zulässt, den Durchlauf durch das Modell um einen emittierenden Zustand zu verkürzen. Im folgenden Text wird dieser Ansatz aufgrund seines Sprunges mit "reduzierter BAKIS-Topologie" bezeichnet. Begründet wird dieser Ansatz in der Literatur (Bechetti and Ricotti, 1999) damit, bei Datenknappheit weniger Parameter trainieren zu müssen als beim BAKIS-Modell. Jedoch wird keine Begründung dafür gegeben, weshalb gerade der mittlere Zustand übersprungen werden soll.

Die letzte Variante basiert auf der Annahme, dass Hidden-Markov-Modelle mit zunehmender Anzahl emittierender Zustände qualitativ verbessert werden können. Da jedoch für Modelle mit mehr Zuständen auch mehr Beobachtungen vorhanden sein müssen, werden drei Modellklassen gebildet. Phonen mit weniger als 30000 Observationen werden lineare Modelle mit zwei Zuständen und solchen mit bis zu 90000 Observationen aufweisen werden Modelle mit drei Zuständen zugewiesen. Darüberliegende werden durch vier linear verknüpfte Zustände modelliert. Die Ergebnisse waren bereits in einem früheren Experiment (Knoblauch, 2004) unbefriedigend, so dass auf eine weitere Untersuchung verzichtet wird.

6.5.2 Ergebnisse der Referenzexperimente mit diagonal besetzten Kovarianzmatrizen

Die Ergebnisse für die Referenztopologien weisen einige Überraschungen auf. Neben der relativ steilen Konvergenz der Systeme mit diagonalen Kovarianzmatrizen (Abbildung 6.4, 6.5) fällt auf, dass die BAKIS-Topologie weder für den VM1 noch für VM2 die besten Erkennungsraten liefert (Tabelle 6.5), obwohl die BAKIS-Topologie die größte Topologievariationsbreite aufweist.

Die BAKIS-Topologie weist zwar deutlich bessere Erkennungsergebnisse als die lineare Dreier-Topologie auf, bleibt aber sowohl hinter der reduzierten BAKIS-Topologie als auch hinter der wissensbasierten Topologie deutlich zurück. Die

wissensbasierte Topologie zeigt nach diesem Trainingsabschnitt die zweitbesten Ergebnisse. Die besten Ergebnisse liefert die reduzierte BAKIS-Topologie mit einer Wortfehlerrate von 33.2% für VM1 und einer Wortfehlerrate von 34.8% für VM2. Die Ergebnisse für die reduzierte BAKIS-Topologie müssen unter der Randbedingung betrachtet werden, dass den Topologien ca. 27% mehr Mixturen zur Verfügung stehen als den Topologien der wissensbasierten Variante und der denen der BAKIS Variante. Dieser Nachteil in der Modellierung könnte durch den nächsten Verfeinerungsschritt (Schritt 5-7), noch ausgeglichen werden, da durch voll besetzte Kovarianzmatrizen eine genauere Modellierung möglich ist.

	Wortfehlerrate [%]			
Corpus	Baseline 14979 mix	Reduzierte BAKIS 14979 mix	BAKIS 11040 mix	Wissensbasiert 11018 mix
VM1	36.9	33.2	35.2	34.2
VM2	37.6	34.8	36.6	35.9

Tabelle 6.5: *Ergebnisse für diagonal besetzte Kovarianzmatrizen der Referenztopologien*

Die Trainingsverläufe der verschiedenen Referenzsysteme sind in Abbildung 6.4 für VM1 und für VM2 in Abbildung 6.5 dargestellt. Die verschiedenen Messpunkte stammen jeweils aus Schritt $4 \rightarrow 3$ des Trainingsverfahrens, in dem durch iterative Anwendung des Trainingsalgorithmus die Wortfehlerate ständig gesenkt wird. In den Abbildungen 6.4 und 6.5 sind auf der Abszisse jeweils die Mixturen pro Zustand angegeben, die maximal aufgrund der Beobachtungsanzahl der verschiedenen Phone erreicht werden können (Abschnitt 6.3). Die den Mixturen zugeordneten Spalten stellen die jeweiligen Schritte $4 \rightarrow 2$ des Experimentes dar. Die Gesamtanzahl der Mixturen des jeweiligen Topologiesystems sind oberhalb der Grafik angegeben. Im Wesentlichen fällt auf, dass die BAKIS-Topologie während des Trainings mit den VM1 Daten mit wenigen Mixturen anfangs bessere Ergebnisse liefert. Mit zunehmender Anzahl Mixturen zeigen die andern Topologien jedoch zunehmend bessere Erkennungsergebnisse. Während des VM2 Trainings fällt auf, dass die reduzierte BAKIS-Topologie von Beginn an die besten Spracherkennungsergebnisse liefert.

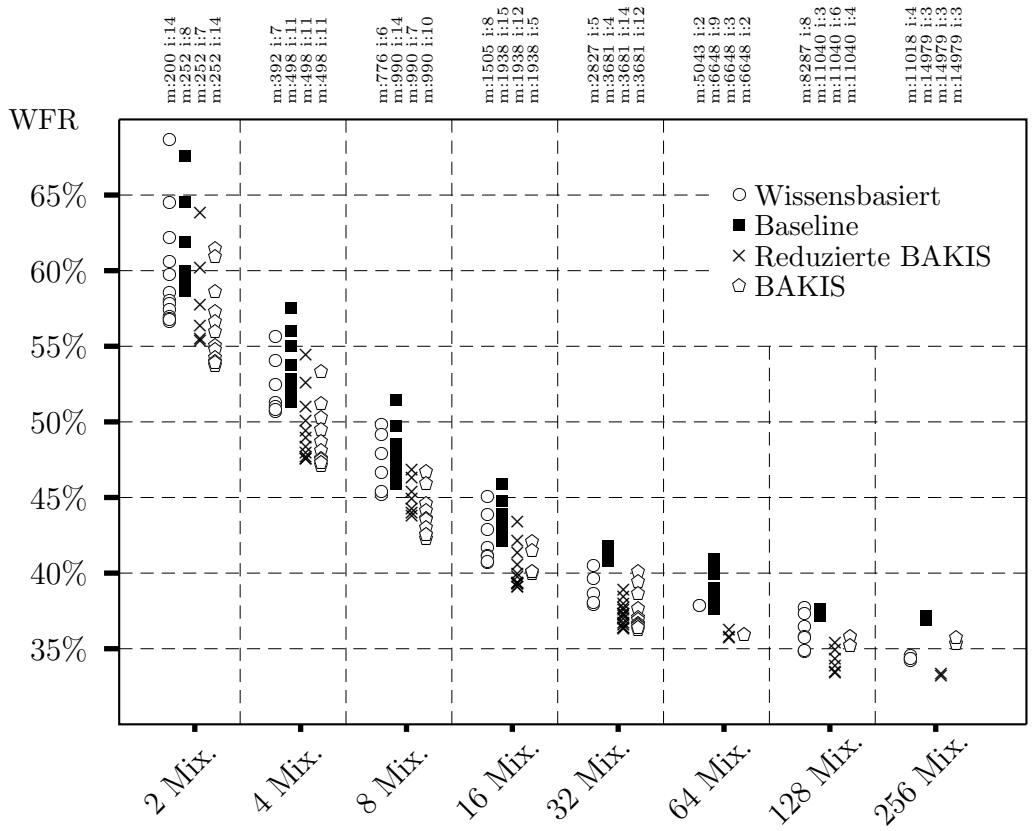

Abbildung 6.4: *Wortfehlerraten der Referenztopologien während des Trainings auf VM1 Daten für diagonal besetzte Kovarianzmatrizen*

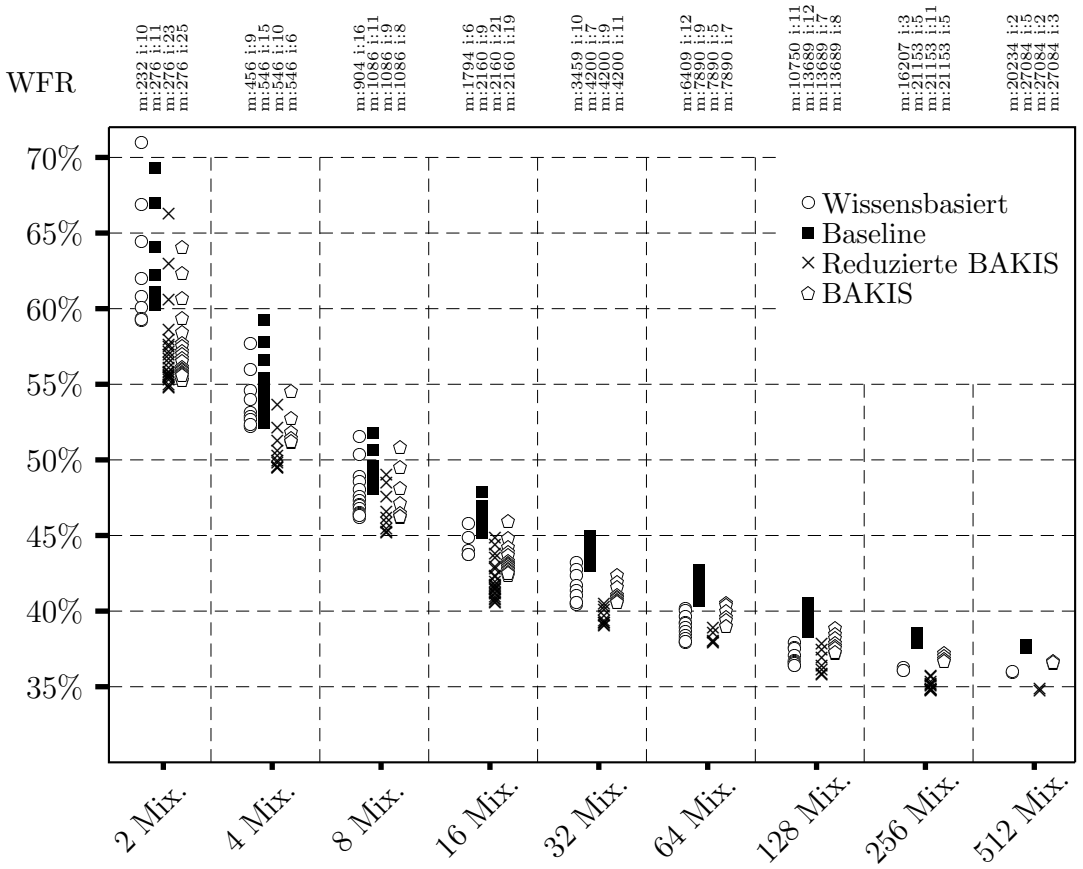

Abbildung 6.5: *Wortfehlerraten der Referenztopologien während des Trainings auf VM2-Daten für diagonal besetzte Kovarianzmatrizen*

6.5.3 Ergebnisse der Referenzexperimente mit voll besetzten Kovarianzmatrizen

Der letzte Verfeinerungsschritt, das Initialisieren und Trainieren voll besetzter Kovarianzmatrizen könnte den Gesamtsystemen mit weniger Mixturen erlauben, ihren Modellierungsnachteil auszugleichen. Tatsächlich profitiert das Wissensbasierte System auch davon (Tabelle 6.6) und kann im Falle des gesamten Trainingsmaterials VM2 die besten Spracherkennungsergebnisse mit einer Wortfehlerrate von 30.9% liefern. Der Unterschied zur varianten BAKIS-Topologie, welche eine Wortfehlerrate von 31.0% für VM2 aufweist, ist jedoch nicht signifikant. Die BAKIS-Topologie kann den Modellierungsnachteil nicht ausgleichen und liefert sowohl für VM1 als auch für VM2 Wortfehlerraten, die nur eine Verbesserung gegenüber dem lineare Baseline-System aufweisen. Das wissensbasierte System zeigt für VM1 nach diesem Trainingsschritt ebenso wie vor diesem Trainingsschritt das zweitbeste Ergebnis, hinter der reduzierten BAKIS-Variante die eine Wortfehlerrate von 28.2% aufweist.

	Wortfehlerrate [%]			
Corpus	Baseline	Reduzierte BAKIS	BAKIS	Wissensbasiert
VM1	31.3	28.2	29.7	28.8
VM2	33.1	31.0	32.3	30.9

Tabelle 6.6: *Ergebnisse für voll besetzte Kovarianzmatrizen der Referenztopologien*

Nach diesen Ergebnissen ist es empfehlenswert die wissensbasierte Topologie bei größeren Datenmengen zu bevorzugen, da sie trotz ca. 27% weniger Mixturen ähnliche Spracherkennungsergebnisse liefert wie die reduzierte BAKIS-Topologie.

6.6 Das MPP-Simulations-Verfahren

Die wesentliche Frage, die anfangs geklärt werden muss, ist, welche Topologien miteinander verglichen werden sollen. Außerdem ist die Berücksichtigung von

verschiedenen experimentellen Randbedingungen notwendig, die im Vorfeld geklärt werden müssen.

In den Referenzexperimenten fällt auf, dass es offensichtlich einen wesentlichen Unterschied in der Erkennungsleistung zwischen Modellen mit zwei und drei linear verbundenen emittierenden Zuständen gibt. Es lässt sich jedoch nicht ohne weiteres sagen, dass die reduzierte BAKIS-Topologie die richtige Lösung für die akustischen Modellierung darstellt. Möglicherweise eignet sich eine phonspezifische Modellierung ähnlich der wissensbasierten Variante eher für eine optimierte akustische Modellierung.

In einem speziellen Vorexperiment (Knoblauch, 2004) wurden verschiedene MPP-Systeme auf ihre Tauglichkeit anhand der VM1-Daten überprüft. Da die Durchführung derartiger Experimente einen enormen Zeitaufwand benötigt, wurden verschiedene Parameter während der Experimente eingeschränkt.

Um den zeitlichen Aufwand zu begrenzen, wurden diese Modelle vorerst bis zu einer Mixturobergrenze von 4800 Mixturen trainiert. Anschließend wurden die Systeme bis hin zu voll besetzten Kovarianzmatrizen verfeinert. Es zeigte sich, dass das System, welches zwischen Zwei-Zustands-Topologien und Drei-Zustands-Topologien wählen konnte, die besten Ergebnisse lieferte (Tabelle 6.7). Die Indizes in der Tabelle beschreiben, wieviele Zustände die jeweiligen Zweige des verwendeten MPPs aufwiesen.

WFR [%] MPP Systeme		
MPP_{2-4}	MPP_{1-4}	MPP_{2-3}
34.2	36.6	32.8

Tabelle 6.7: *Ergebnisse aus Vorexperimenten mit voll besetzten Kovarianzmatrizen auf den VM1-Daten*

Aus diesen Vorexperimenten folgen zwei Untersuchungen, die beide die Topologie aus Abbildung 4.3 verwenden. Das MPP-HMM beinhaltet zwei parallel angeordnete lineare Topologien, von denen der eine Topologiezweig zwei emittierende Zustände und der andere drei emittierende Zustände aufweist. Zusätzlich wird eine dritte experimentelle Variante hinzugenommen, die phonspezifische Eigenschaften berücksichtigt.

- Das folgend mit "MPP Variante I" bezeichnete Experiment trainiert das MPP, indem die Mixturensplits gleichmäßig auf beide Zweige angewendet werden. Das hat zur Folge, dass am Ende des Trainings die einzelnen Zweige über eine verschiedene Anzahl von Mixturen verfügen. Dies kann ein Ungleichgewicht zu Gunsten der längeren Modelle bedeuten.

- Im Gegensatz hierzu wird ein zweites Experiment durchgeführt und nachfolgend mit "MPP Variante II" bezeichnet, bei dem die Mixturensplits auf den verschiedenen Zweigen gleichgewichtet werden.

- Zusätzlich wird ein Simulationsexperiment durchgeführt, welches das Optimierungsverhalten der MPPs in Abhängigkeit der phonspezifischen Normierungslängen untersucht.

6.6.1 Das MPP-Trainingsverfahren

Die initialen MPPs werden genau wie die Referenzexperimente aus den durch die Segmentinformation errechneten Prototypen zweigweise erstellt und zu jeweils einem MPP verknüpft. Existieren keine Segmentinformationen, wie für die Spezialmodelle `noise, anoise, laugh, breath`, wird ein generalisierter Prototyp herangezogen.

Eine weitere Frage ist, bis zu welchem Trainingszustand die MPP-Topologien trainiert werden müssen damit die Zweigauswahl konvergiert. Darunter wird nicht die Konvergenz in Erkennungsergebnissen verstanden, sondern das Auswahlverhalten zwischen den Topologien.

Hierzu wird während des Trainings der ersten MPP-Topologie, die über eine Zweier- und eine Dreier-Topologie verfügt, nach jedem Iterationsschritt geprüft, ob sich das Auswahlverhältnis noch ändert. Bei der Verwendung von diagonalen Kovarianzmatrizen ergibt sich, dass nach $i = 64$ Mixturensplits keine Veränderungen mehr stattfinden.

Nachdem das Training eines MPP-HMMs bis $i = 64$ Mixturensplits durchgeführt wurde, werden die Sprachdaten mit diesem MPP-HMM realigniert um die für die Normierung benötigten durchschnittlichen Phonsequenzlängen zu erhalten. Dieser Schritt ist notwendig, da nicht bekannt ist, ob z.B. durch

den Einsatz einer kürzeren oder längeren Topologie innerhalb des verwendeten MPPs eine Veränderung der phonspezifischen Mittelwerte eintritt.

Dieser Schritt wird nur einmal ausgeführt und daher in der folgenden graphischen Darstellung nicht aufgezeigt. Die Ergebnisse hierzu werden aber in den Abschnitten weiter unten beschrieben, in denen die verschiedenen experimentellen Varianten diskutiert werden.

Das Gesamtverfahren (Abbildung: 6.6) lässt sich nun inklusive der Randbedingungen wie folgt darstellen:

1. Schritt:
   Die einzelnen MPP-Zweige werden aus den initialen Phonmodelltopologien separat erstellt und anschließend zu einem MPP-HMM verknüpft. Sie weisen eine einfache diagonale Kovarianzmatrix auf.

2. Schritt:
   Die Anzahl der Mixturen pro emittierenden Zustand wird pro Durchgang um Faktor zwei in Abhängigkeit von den zur Verfügung stehenden Beobachtungen (Abschnitt 6.3) eines Phons erhöht. Sollte ein Phonmodell über zu wenig Beobachtungen verfügen, wird die bereits erreichte Mixturenanzahl während des restlichen Trainings konstant gehalten. Während im Experiment zur MPP-Variante I die Mixturensplits systematisch auf beiden Zweigen durchgeführt werden, so dass beide Zweige immer eine unterschiedliche Anzahl von Mixturen aufweisen, werden bei den Experimenten MPP-Variante II+III die Mixturensplits so gewichtet, dass beide Zweige in jedem Schritt immer durch gleich viele Mixturen modelliert werden. Das Hinzufügen von Mixturen endet, wenn $i = 64$ Mixturensplits erreicht sind.

3. Schritt:
   Ein Zyklus eins Baum-Welch-Trainingsschrittes.

4. Schritt:
   Evaluation der in Schritt 3 trainierten Modelle. Solange die Erkennungsrate steigt, weiter mit Schritt 3, sonst zurück zu Schritt 2 und Erhöhung der Anzahl der Mixturen.

5. Schritt:
   MPP-Simulation. Der Zweig, der in der phonspezifischen MPP-Simulation die höhere Simulationswahrscheinlichkeit erreicht, wird selektiert.

6. Schritt:
   Aus den qualitativ höher bewerteten Topologiezweigen wird ein neuer Satz phonspezifischer HMM-Prototypen erstellt.

Auf diese Topologieoptimierung folgt ein kompletter Trainingsverlauf, wie er in Abschnitt 6.4 dargestellt ist.

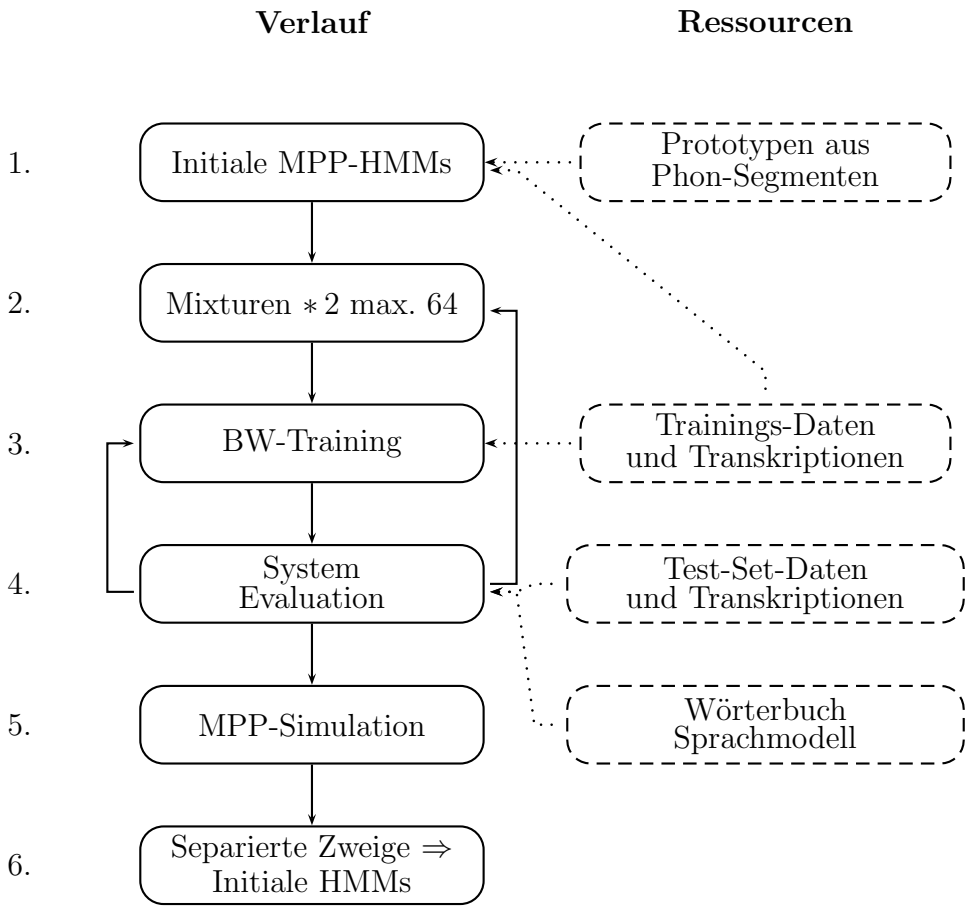

Abbildung 6.6: *Trainingsverfahren der Multi-Parallelen-Modelle und anschließende MPP-Simulation*

6.6.2 Durch das MPP-Modell ermittelte phonspezifische Sequenzlängen

Bevor auf Resultate der experimentellen Varianten der MPP-Modellierung eingegangen wird, soll an dieser Stelle das Ergebnis der Merkmalssequenzlängenanalyse einzelner Phonmodelle vorgestellt werden, welches aus den Trainingsdaten errechnet wird.

Die Frequenzen der Merkmalssequenzlängenklassen werden zunächst gezählt. Anschließend wird das arithmetische Mittel daraus berechnet. Daraus ergeben sich die folgenden in Abbildung 6.7 dargestellten durchschnittlichen Merkmalssequenzlängen pro Phonmodell, die für die Normierung der MPP-Simulation in den Experimenten MPP-Variante I und II herangezogen werden (siehe auch Kapitel 4).

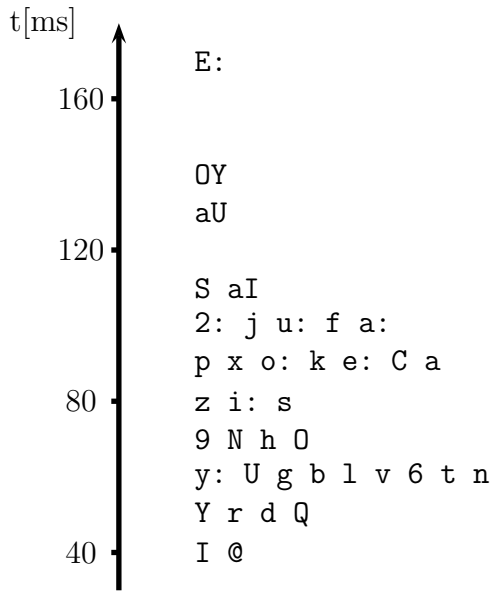

Abbildung 6.7: *Durchschnittliche Merkmalssequenzlängen einzelner Phonmodelle*

Aus Tabelle 6.7 ist zu ersehen, dass sich die phonspeziefischen Merkmalsse-

quenzlängen wie erwartet verhalten. Diphthonge und lange Vokale neigen eher dazu, durch längere Merkmalssequenzen repräsentiert zu werden im Gegensatz zum `@` und zu den kurzen Vokalen, die durch kürzere Merkmalssequenzen repräsentiert werden. Die errechneten Merkmalssequenzlängen unterstützen die phonetische Sichtweise, insbesondere bezüglich des wissensbasierten Ansatzes der Phonetiker.

Die Merkmalssequenzlängen der Spezialmodelle liegt bei 200 ms (20 Frames) und deutlich darüber (Tabelle 6.8).

Modell	t[ms]
`usb`	230
`noise`	200
`anoise`	210
`laugh`	250
`breath`	360

Tabelle 6.8: *Modulationslängen einzelner Spezialmodelle*

6.6.3 Experimentelle MPP-Variante I

Im ersten MPP-Experiment wird die Anzahl der Mixturen pro Zweig gleichmäßig erhöht. Das bedeutet, dass die jeweiligen Topologiezweige, in Abhängigkeit von ihrer Länge eine ungleiche Anzahl von Mixturen aufweisen. Topologien mit weniger emittierenden Zuständen werden somit durch weniger Wahrscheinlichkeitsdichtefunktionen modelliert. Die Daten, die durch einen Zweig mit insgesamt weniger Mixturen modelliert werden, werden vermutlich ungenauer modelliert als der Konkurrenzzweig.

Nach der MPP-Simulation ergibt sich für die neu zu initialisierenden HMM-Topologien folgendes Bild (Tabelle 6.9).

Dieses Ergebnis wird auf beiden Trainingskorpora VM1 und VM2 erzielt. In diesem Experiment ergibt sich zunächst eine ausgewogene Aufteilung zwischen einer Topologie mit zwei Zuständen und einer mit drei Zuständen. Überraschenderweise zeigt sich, dass die Diphthonge `aI`, `aU` und ca. die Hälfte aller langen Vokale durch Topologien mit nur zwei Zuständen optimiert werden. Dies

# Zustände	Modell
2	S N d j y: k g u: v aI b a: z r Q @ 2: aU l p
3	usb a E Y 9 e: E t o: i: s OY I U x 6 h C f n O m

Tabelle 6.9: *Resultierende Modelltopologien nach der MPP-Simulation auf VM1 und VM2*

widerspricht dem wissensbasierten Ansatz, in dem diese Phone durch längere HMM-Topologien modelliert werden.

6.6.4 Experimentelle MPP-Variante II

In der experimentellen MPP-Variante II werden beide Topologiezweige bezüglich der Anzahl der Mixturen gleichgewichtet, so das am Trainingsende jeder Zweig durch gleich viele Mixturen modelliert wird. Durch diese Randbedingung entsteht eine starke Verschiebung in Richtung der Modelle mit zwei Zuständen während der MPP-Simulation. Insgesamt werden nur noch acht Modelle mit drei Zuständen ermittelt (Tabelle 6.10).

# Zustände	Modell
2	S N d j y: k g u: v aI b a: z r Q @ aU O l p a E Y 9 e: o: s OY U x 6 h f n
3	usb 2: C m I i: t E:

Tabelle 6.10: *Modelltopologien nach MPP-Variante II für VM1 und VM2*

6.6.5 Merkmalssequenzlängenabhängige MPP-Variante III

Die phonspezifischen Merkmalsequenzlängen, die für die Normierung der einzelnen Zweige eines Phon-MPPs herangezogen werden, können erheblichen Einfluss auf die simulierten Wahrscheinlichkeiten haben. Zwar wird durch die Normierung gewährleistet, dass die Simulation zwei gleich lange Pfade berechnet. Jedoch findet die Tatsache keine Berücksichtigung, dass das Modell mit mehr emittierenden Zuständen immer mindestens einen Zustandsübergang mehr in der Berechnung aufweist, um auf die Normlänge zu kommen, als

das “kürze” Vergleichsmodell. Weisen nun die HMM-Zweige jeweils höhere Loop-Wahrscheinlichkeiten a_{ii} als Übergangswahrscheinlichkeiten a_{ij} auf, befindet sich der MPP-Zweig mit der größeren Anzahl an emittierenden Zuständen zunächst im Nachteil. Dieser Nachteil kann allerdings bei zunehmender Normierungslänge ausgeglichen werden.

Die folgende Untersuchung soll dieses Verhalten analysieren. Aus den Trainingsdaten für VM1 werden für jedes Phon die Merkmalssequenzlängen errechnet. Gleichzeitig werden die Häufigkeiten der jeweiligen Merkmalssequenzlängen notiert. Die MPP-Simulation wird nun erweitert, indem für das gerade untersuchte Phonmodell alle Normierungslängen in die Berechnung einfließen. Je nachdem welche Topologie die bessere Simulationswahrscheinlichkeit aufgrund der Normierung erreicht, wird der prozentuale Sequenzlängenanteil, der oben errechnet wurde, zugeordnet. Für die oben beschriebene Untersuchung werden die MPP-Modelle der Variante II herangezogen, da diese Variante die besseren Spracherkennungsergebnisse aufweist.

In der folgenden Abbildung 6.8 ist das Ergebnis dargestellt. Auf der Abszisse sind Sequenzlängen in Millisekunden aufgetragen. Der Abszissenwert kennzeichnet die Stelle, ab der das untersuchte Phon die höhere Simulationswahrscheinlichkeit für den längeren Drei-Zustand-Zweig erzielt.

Gleichzeitig gibt der Wert der Ordinate den prozentuale Anteil der jeweiligen MPP-Zweige in Hinblick auf die modellierten Beobachtungen an. Die prozentuale Angabe bezieht sich darauf, wieviel Prozent der jeweiligen Beobachtungen eines Phons durch den jeweiligen Zweig des MPPs während der Simulation die höhere Bewertung erhält. Die Angabe 75/25% bedeutet z.B., dass 75% aller Beobachtungen dieses Phons während der Simulation eine höhere Simulationswahrscheinlichkeit für den Zweig mit drei Zuständen erhalten, und dass nur 25 % der Beobachtungen durch die Zweier-Topologie ”besser” beschrieben werden.

Die MPP-Zweige, die eine eindeutige Zuordnung während der Simulation erreichen, erhalten einen 100 %ige Wert. Diese Modelle sind unabhängig von den Sequenzlängen der Abszisse zu verstehen und aus Platzgründen in Abb.

6.8 zusammengefasst dargestellt.

Die mit * gekennzeichneten Phone weisen das entgegengesetzte Verhalten auf. Sie beginnen bei kurzen Normierungslängen mit dem Drei-Zustands-Modell und enden mit den kürzeren Topologien. Das deutet darauf hin, dass in diesen Phonmodellen die Loopwahrscheinlichkeiten der Zweier-Topologien geringer sind als die der Dreier-Topologie.

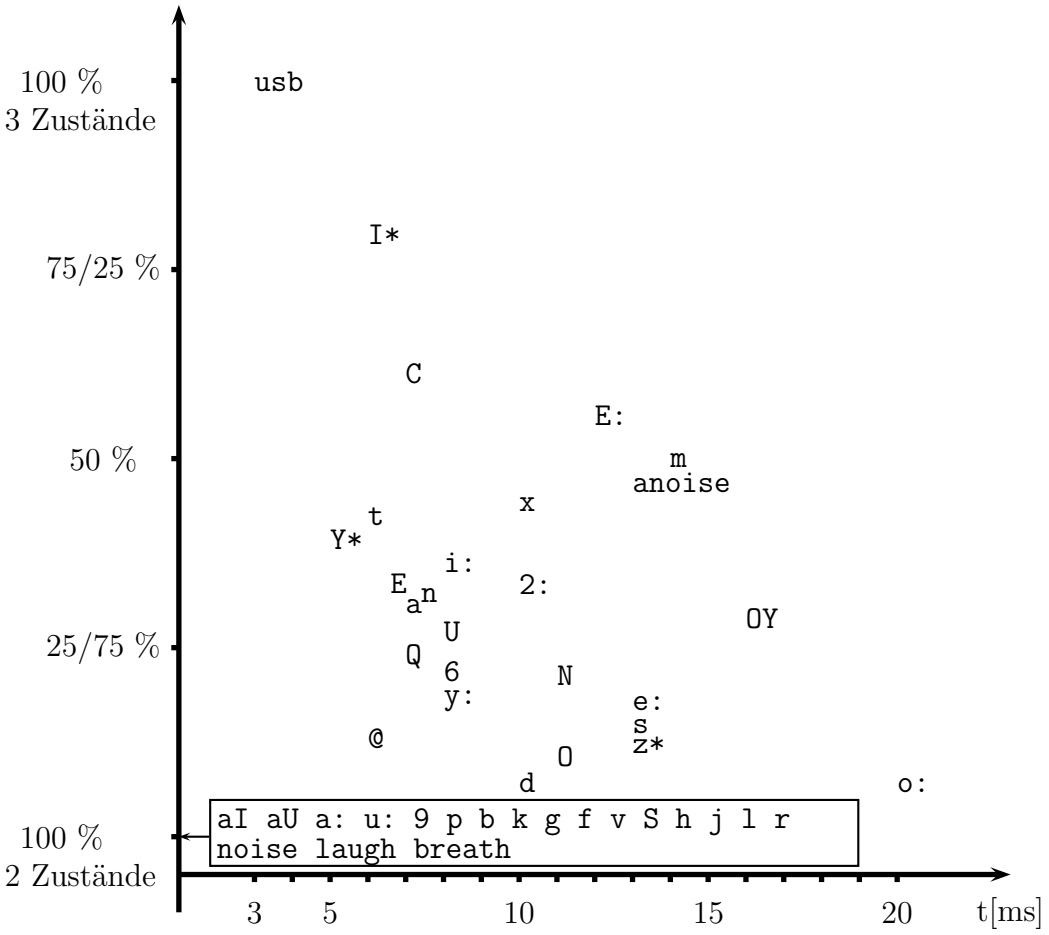

Abbildung 6.8: *Ergebnis der MPP-Simulation in Abhängigkeit von der Normierungslänge*

Dieses Ergebnis wird im Experiment umgesetzt, indem alle 100%-igen Zuordnungen mit einem entsprechenden linearen Links-Rechts-Modell initiali-

siert werden. Die nicht eindeutigen Zuordnungen werden mit der reduzierten BAKIS-Topologie initialisiert, welche aus dem gleichnamigen Referenzexperiment bekannt ist, also über drei linear verknüpfte Zustände verfügt, deren mittlerer Zustand optional ist. Grundsätzlich könnte durch das MPP-Verfahren ermittelt werden, welcher Zustand des Modells für einen Skip geeigneter ist. Von dieser Möglichkeit wird in dieser Arbeit zunächst kein Gebrauch gemacht, da hier im Vordergrund steht, ob die Auswahltechnik der MPP-Variante III Topologien noch weiter optimieren kann als die bereits vorgestellten MPP-Varianten I+II.

6.6.6 MPP-Evaluation für kurze Zweige

Es wird außerdem der Versuch durchgeführt, Topologien mit nur einem Zustand zur MPP-Evaluation zuzulassen, indem ein MPP-Modell mit einer Einer-Topologie und einer Zweier-Topologie trainiert wird. Das Ergebnis ist jedoch unbefriedigend, da durchweg die Einer-Topologien höhere Simulationswahrscheinlichkeiten zeigen und die mit diesen Topologien trainierten Spracherkenner schlechtere Erkennungsraten aufweisen. Es stellt sich an dieser Stelle die Frage nach den Grenzen der MPP-Auswahltechnik, auch Ein-Zustands-Topologien zuzulassen. Eine zusammenhängende Diskussion über die Grenzen der MPP-Verfahren findet sich in Kapitel 7.

6.6.7 Trainingsverläufe der MPP-optimierten Spracherkenner für diagonal besetzte Kovarianzmatrizen

Das Training der diagonal besetzten Topologien wird analog zu den Referenzexperimenten durchgeführt (Abbildung 6.2). Die Trainingsverläufe sind in den Abbildungen 6.9 für VM1 und 6.10 für VM2 dargestellt. Die verschiedenen Messpunkte stammen ebenfalls aus Schritt $4 \rightarrow 3$ des Trainingsverfahrens, indem durch iterative Anwendung des Trainingsalgorithmus die Wortfehlerate ständig gesenkt wird. In den Abbildungen 6.9 und 6.10 sind auf der Abszisse jeweils die Mixturen pro Zustand angegeben, die maximal aufgrund der Beobachtungsanzahl der verschiedenen Phone erreicht werden können (Abschnitt 6.3). Die den Mixturen zugeordneten Spalten stellen die jeweiligen Schritte

$4 \rightarrow 2$ des Experimentes dar. Die Gesamtanzahl der Mixturen des jeweiligen Topologiesystems sind oberhalb der Graphik angegeben.

Aus der Abbildung 6.9 lässt sich erkennen, dass die auf den VM1-Daten trainierte MPP-Variante II, für max. 4-16 Mixturen pro Zustand, die bessere Topologie-Variante darstellt. Beim Training des gesamten Korpus VM2 (Abbildung 6.10) weist die MPP-Variante III fast während des gesamten Trainings bessere Spracherkennungsergebnisse auf. Am Ende des Trainings zeigt jedoch die MPP-Variante II für beide Datenmengen VM1 und VM2 die geringeren Wortfehlerraten.

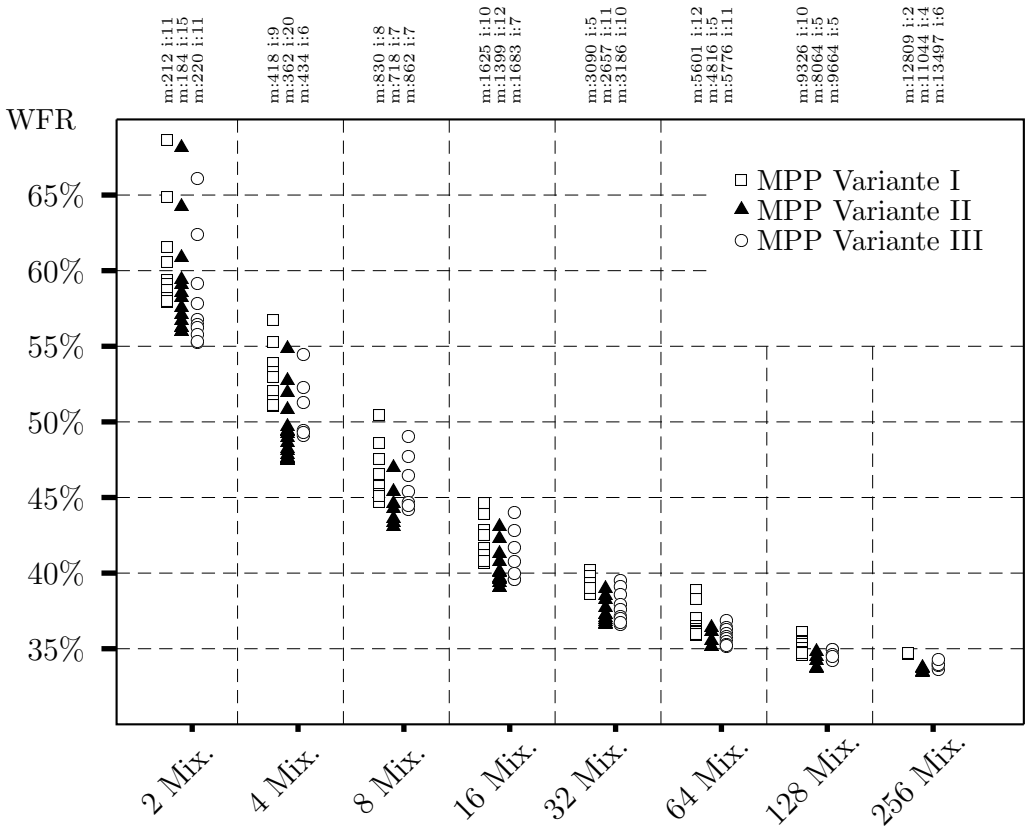

Abbildung 6.9: *Wortfehlerraten der MPP-optimierten Topologien während des Trainings auf VM1-Daten*

6.6.8 Ergebnisse der MPP-Optimierung für diagonale Kovarianzmatrizen

Die Ergebnisse der MPP-Varianten I-III für den Trainingsabschnitt mit diagonal besetzten Kovarianzmatrizen zeichnen sich insgesamt durch geringe Wortfehlerraten aus. Im Gegensatz zur Baseline-Topologie der Referenzsysteme werden die Ergebnisse mit einer geringeren Anzahl von trainierten Mixturen erreicht. Nach diesem Trainingsteil lässt sich vorerst noch kein eindeutiger Trend

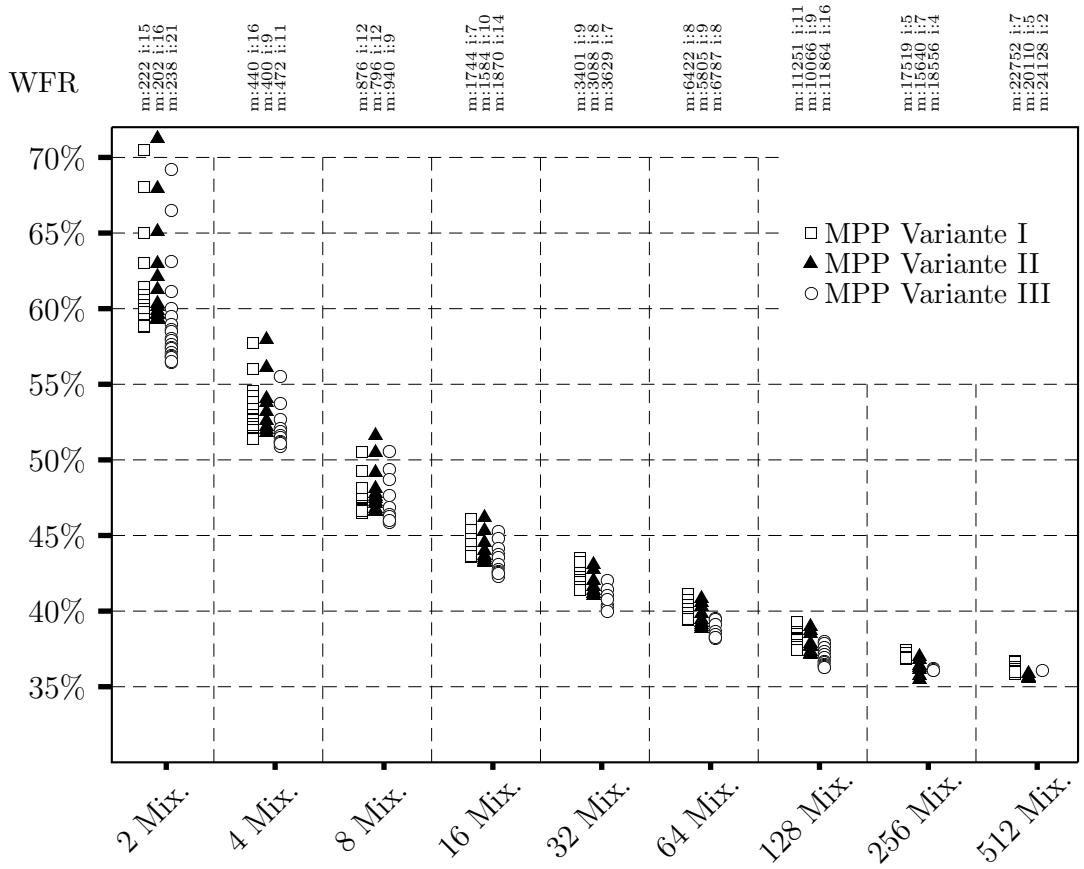

Abbildung 6.10: *Wortfehlerraten der MPP-optimierten Topologien während des Trainings auf VM2-Daten*

erkennen, welcher der Ansätze die besten Erkennungsergebnisse liefert (Tabelle 6.11). Sowohl für die VM1-Daten als auch für die VM2-Daten liefert die MPP-Variante II die geringsten Wortfehlerraten.

Corpus	Wortfehlerraten [%]		
	MPP-Variante I 12809 mx	MPP-Variante II 11044 mx	MPP-Variante III 13497 mx
VM1	34.6	33.4	33.6
VM2	35.8	35.5	36.0

Tabelle 6.11: *Ergebnisse für optimierte Topologien mit diagonal besetzten Kovarianzmatrizen*

6.6.9 Ergebnisse der MPP-Optimierung für voll besetzte Kovarianzmatrizen

Durch das Training voll besetzter, inverser Kovarianzmatrizen lassen sich signifikante Unterschiede zwischen den MPP-optimierten Spracherkennungssystemen feststellen. Die MPP-Variante III erreicht für die beiden Datensätze die geringsten Wortfehlerraten von 27.6 % für VM1 und 30.3 % für VM2 (6.12). Die Topologien der MPP-Variante III verbessern die Wortfehlerraten der MPP-Variante II um weitere 2 % für VM1 und ca. 1.5 % für VM2. Während die Verbesserung der MPP-Variante III gegenüber der MPP-Variante II auf den VM1-Daten noch signifikant ist, sinkt die Verbesserungsdifferenz auf den VM2-Daten unter das Signifikanzniveau. Gegenüber der MPP-Variante I wird bei beiden Datenmengen, VM1 und VM2, eine signifikante Verringerung der Wortfehlerrate erreicht.

Corpus	Wortfehlerraten [%]		
	MPP-Variante I	MPP-Variante II	MPP-Variante III
VM1	28.9	28.2	27.6
VM2	31.2	30.7	30.3

Tabelle 6.12: *Ergebnisse für voll besetzte Kovarianzmatrizen der optimierten Topologien*

6.7 Gesamtergebnis für diagonal besetzte Kovarianzmatrizen

In Tabelle 6.13 werden die Spracherkennungsergebnisse aller untersuchten Topologien relativ zur linearen Links-Rechts-Referenztopologie für den ersten Trainingsabschnitt der diagonal besetzte Kovarianzmatrizen zusammengefasst. Die relative Darstellung erweist sich an dieser Stelle als günstig, da Unterschiede der Erkennungsraten in Abhängigkeit von den jeweiligen Korpora und untersuchten Topologien deutlich werden. Die prozentualen Veränderungen der jeweiligen Topologien werden im Vergleich zum Baseline-System dargestellt, welches für beide Trainingsdatensätze die absolut erreichte Wortfehlerrate aufzeigt. Die verwendeten Mengen von Mixturen werden in der Tabelle mit "mx" angegeben.

	absolute WFR [%]	relative Verbesserung der Wortfehlerrate [%]		
		Referenztopologien		
Korpus	Baseline 14979 mx	Reduzierte BAKIS 14979 mx	BAKIS 11040 mx	Wissensbasiert 11018 mx
VM1	36.9	**10.03**	4.40	7.32
VM2	37.6	**7.45**	2.66	4.52
		Optimierte Topologien		
Korpus		MPP-Variante I 12809 mx	MPP-Variante II 11044 mx	MPP-Variante III 13497 mx
VM1	-	6.23	*9.49*	8.94
VM2	-	4.79	*5.59*	4.26

Tabelle 6.13: *Gegenüberstellung der Ergebnisse für diagonal besetzte Kovarianzmatrizen*

Die reduzierte BAKIS-Topologie erzielt für diagonal besetzte Kovarianzmatrizen die besten Ergebnisse. Die Wortfehlerrate des Baseline-Systems wird für die VM1-Daten um 10.03 % und für die VM2-Daten um 7.45 % verringert. Das durch die MPP-Variante II optimierte Erkennungssystem weist für VM1 ein ähnliches Ergebnis auf, bei dem die Wortfehlerrate des Baseline-Systems um 9.49% verringert wird. Die Verbesserung der Ergebnisse durch das wissensbasierte System bezüglich des Baseline-Systems beträgt 7.32 % für VM1 und

erreicht damit nicht die Verbesserungswerte, die durch die MPP-Variante II und III optimierten Systeme erzielt werden. Das durch die MPP-Variante III optimierte System verringert die Wortfehlerrate des Baseline-Systems um 8.94 % für die VM1-Daten. Das durch die MPP-Variante I optimierte System verringert die Wortfehlerrate des Baseline-Systems für die VM1-Daten um 6.23% und für die VM2-Daten um 4.79%. Das durch die MPP-Variante I optimierte System erzielt ähnliche Ergebnisse wie das wissensbasierte System. Das System, welches die BAKIS-Topologien einsetzt, verbessert die Ergebnisse des Baseline-Systems nur geringfügig.

Wird auch hier, wie in den voherigen Abschnitten, die Anzahl der trainierten Mixturen berücksichtigt, erscheint der Vorteil der reduzierte BAKIS-Topologie für VM1 fragwürdig, da dieses System das Ergebnis mit deutlich mehr trainierten Mixturen erreicht als das wissensbasierte System und das System der MPP-Variante II.

6.8 Gesamtergebnis für voll besetzte Kovarianzmatrizen

Ebenso wie im vorherigen Abschnitt werden in Tabelle 6.14 die Spracherkennungsergebnisse aller untersuchten Topologien relativ zur linearen Links-Rechts-Referenztopologie für den zweiten Trainingsabschnitt, das Training der voll besetzte Kovarianzmatrizen, zusammengefasst. Die relative Darstellung erweist sich ebenfalls als günstig, da Unterschiede der Erkennungsraten in Abhängigkeit von den jeweiligen Korpora und untersuchten Topologien auch hier deutlich werden. Die prozentualen Veränderungen der jeweiligen Topologien werden im Vergleich zum Baseline-System wie im letzten Abschnitt dargestellt, welches für beide Trainingsdatensätze die absolut erreichte Wortfehlerrate aufzeigt. Die verwendeten Mengen von Mixturen sind wiederum mit "mx" in der Tabelle angegeben.

Durch das Training der voll besetzten Kovarianzmatrizen wird der Modellierungsnachteil der Systeme mit wenigen Mixturen teilweise ausgeglichen. Am deutlichsten profitieren die MPP-optimierten Systeme von diesem Verfeine-

	absolute WFR [%]	relative Verbesserung der Wortfehlerrate [%]		
		Referenztopologien		
Korpus	Baseline 14979 mx	Reduzierte BAKIS 14979 mx	BAKIS 11040 mx	Wissensbasiert 11018 mx
VM1	31.3	**9.90**	5.11	7.99
VM2	33.1	6.34	2.42	**6.65**
		Optimierte Topologien		
Korpus		MPP Variante I 12809 mx	MPP Variante II 11044 mx	MPP Variante III 13497 mx
VM1	-	7.67	9.90	*11.82*
VM2	-	5.74	7.25	*8.52*

Tabelle 6.14: *Gegenüberstellung der Ergebnisse für voll besetzte Kovarianzmatrizen*

rungsschritt. Das durch die MPP-Variante III optimierte Spracherkennungssystem verringert die Wortfehlerrate des Baseline-Systems um 11.82 % für die VM1-Daten und um 8.52 % für die VM2-Daten und weist damit die höchsten Verbesserungsraten auf. Das durch die MPP-Variante II optimierte System verbessert die Ergebnisse des Baseline-Systems um 9.90 % für die VM1-Daten und um 7.25 % für die VM2-Daten. Ähnliche Verbesserungen werden auch durch das reduzierte BAKIS-System erreicht. Die Veränderungen der Ergebnisse durch die BAKIS-Topologie fallen wie im ersten Trainingsschritt mit 5.11 % für VM1 und nur 2.42 % für VM2 gering aus.

Kapitel 7

Diskussion und Ausblick

In der Arbeit wurden sieben Spracherkennungssysteme, mit jeweils verschiedenen HMM-Topologien, bis zur Spracherkennungskonvergenz sowohl auf den VM1-Daten und als auch auf den VM2-Daten trainiert und evaluiert. Die Spracherkennungssysteme teilen sich in vier Referenzsysteme und drei topologieoptimierte Systeme auf. Die durch das erstmals in dieser Arbeit vorgestellte Simulationsverfahren optimierten Spracherkennungssysteme zeigen stellenweise deutliche Verbesserungen der Worterkennungsraten gegenüber den Ergebnissen der Referenzsysteme.

Für voll besetzte Kovarianzmatrizen weisen die Referenzsysteme Wortfehlerraten zwischen 28.2 % und 31.3 % für VM1 und zwischen 30.9 % und 33.1 % für VM2 auf.

Wie zu erwarten war, zeigte die lineare Links-Rechts-Topologie mit 31.3 % (VM1) und 33.1 % (VM2) die höchsten Wortfehlerraten. Die starre HMM-Topologie, durch die systematisch alle Phone gleichermaßen modelliert werden, ist nicht die optimale Herangehensweise.

Entgegen der Erwartungen fallen die Wortfehlerraten der Spracherkenner aus, die mit der BAKIS-Topologie modelliert wurden. Es wurde ursprünglich angenommen, dass aufgrund der hohen Topologie-Variationsbreite der BAKIS-Topologie, zumindest bei den verwendeten Datenmengen aus VM2, deutlich geringere Wortfehlerraten erreicht werden als durch die Systeme der linearen Baseline-Topologie. Die Systeme, die durch die BAKIS-Topologie modelliert

werden, konvergieren bereits bei 11040 Mixturen, verbessern die Ergebnisse der lineare Baseline-Topologie jedoch nur unwesentlich um 5.11 % (VM1) und 2.42 % (VM2). Obwohl die BAKIS-Topologie über die Möglichkeit verfügt, verschiedene Aussprachevarianten einer phonetischen Beobachtung innerhalb eines Modells zu modellieren, beschreibt die Topologie die entsprechenden Phone schlecht. Die Tatsache, dass der Ansatz mit zunehmenden Daten nicht besser wird, lässt den Schluss zu, dass dieses Ergebnis nicht ausschließlich an einer zu geringen Datenmenge liegen kann. Möglicherweise ist die schlechte Modellierung auf den ersten Skip der Topologie zurückzuführen, der einen verallgemeinerten Pfad ermöglicht, der nur einen emittierenden Zustand aufweisen kann. Dieser Effekt entsteht ebenfalls bei der MPP-Optimierung und zeigt dort auch mäßige Ergebnisse während der Spracherkennung. Die Ursachen hierfür werden im Zusammenhang mit den Grenzen der MPP-Optimierung weiter unten diskutiert.

Die reduzierte BAKIS-Topologie weist mit der prozentualen Verbesserung der Ergebnisse der Baseline-Systeme um 9.9 % (VM1) und um 6.34 % (VM2) unter Verwendung von 14979 Mixturen sehr gute Ergebnisse auf im Vergleich zu allen Referenztopologien auf.

Das wissensbasierte System zeigt wie erwartet Verbesserungen der Wortfehlerraten gegenüber dem Baseline-System. Sie betragen 7.99 % (VM1) und 6.65 % (VM2) bei nur 11018 Mixturen. Das Ergebnis bestätigt die Annahme, das es möglich ist, ein Datenmodellierungsproblem durch angepasste lineare Topologien geeignet zu lösen.

Die durch das neuartige Simulationsverfahren optimierten Topologien und die damit trainierten Spracherkenner zeigen alle deutliche Verringerungen der Worfehlerraten gegenüber der linearen Baseline-Topologie. Die ermittelten Erkennungsergebnisse für voll besetzte Kovarianzmatrizen liegen zwischen 27.6% und 28.9 % für VM1 bzw. zwischen 30.3 % und 31.2 % für VM2.

Die durch die Simulation der MPP-Variante III entstandenen Topologien, weisen die höchsten Verbesserungsraten gegenüber den Ergebnissen des Baseline-Systems von 11.82 % (VM1) und 8.52 % (VM2) auf. Mit 13497 Mixturen verfügt die Variante über die größte Anzahl Mixturen, verglichen mit den anderen optimierten MPP-Varianten I+II.

Die Topologien, die durch die Simulation der MPP-Variante II, entstanden sind, zeigen Verbeserungen von 9.90 % (VM1) und 7.25 % (VM2) gegenüber dem Baseline-System. Die Ergebnisse sind vergleichbar mit den Verbesserungen, die die reduzierte BAKIS-Topologie erzielte, jedoch bei einer deutlich geringeren Anzahl von Mixturen.

Durch den in dieser Arbeit vorgestellten Simulationsalgorithmus, der HMMs qualitativ bewertet, ergeben sich vielfältige Möglichkeiten der Topologieoptimierung von phonbasierten HMMs. In der Arbeit ist ein kleiner Ausschnitt der experimentellen Möglichkeiten untersucht worden, um die "Machbarkeit" des Algorithmus zu überprüfen. Es konnte gezeigt werden, dass eine Topologiebewertung durch den gewählten MPP-Simulationsalgorithmus möglich ist und zu Verbesserungen von Spracherkennungsergebnissen führt.

Der MPP-Simulationsalgorithmus ist in der Lage, Topologien anhand der simulierten Gesamtwahrscheinlichkeiten zu bewerten und zu optimieren. Für die Bewertung verschiedener Spracherkennungssysteme werden jedoch nicht mathematisch berechnete Topologiequalitäten, sondern Spracherkennungsergebnisse herangezogen, die durch Faktoren beeinflusst werden, die während des Simulationsverfahrens unberücksichtigt bleiben. Daraus ergeben sich verschiedene Grenzen des MPP-Optimierungsverfahrens.

Das Optimierungsverfahren tendiert bei einem Großteil der Phonmodelle dazu, kürzere Modelle mit wenigen emittierenden Zuständen zu bevorzugen. Der Simulationsalgorithmus konvergiert nicht wie erhofft bei jenen Modelltopologien, die auch die besten Spracherkennungsergebnisse liefern, sondern bei den kürzesten Topologien mit nur einem emittierenden Zustand. Diese kurzen Topologien lassen jedoch im Gegensatz zu den längeren Zweier- und Dreier-Topologien keine sinnvolle phonetische Interpretation zu. Lässt die Dreier-Topologie die Interpretation der phonetischen Feinstruktur (vgl. Abschnitt 3.2.3) und die Zweier-Topologie eine mögliche biphonale Interpretation zu, müsste angenommen werden, dass die Einer-Topologie die Reduktion auf die quasi-stationäre Phase der phonetischen Feinstruktur darstellt. Diese Interpretation widerspricht jedoch der spektralen Beobachtung von Phonen, die sehr

wohl dynamische Veränderungen der Formanten während einer sprachlichen Äußerung zeigen können.

Der Simulationsalgorithmus nimmt auf keine phonetische Interpretationen Rücksicht, sondern arbeitet auf rein mathematischer Ebene. Betrachtet man die einzelnen Simulationsschritte, bei denen es möglich ist, die Wahrscheinlichkeiten der einzelnen Zustände während der Simulation zu überprüfen, zeigt sich, dass während der Reduktion der Zustandsanzahl die Varianzen der einzelnen Zustände kontinuierlich abnehmen. Die Loopwahrscheinlichkeiten, die beschreiben, wieviele Merkmalvektoren von der jeweiligen Topologie konsumiert werden, verändern ihren Wert beim Übergang von der Dreier-Topologie zur Zweier-Topologie wenig. Beim Übergang von der Zweier- zur Einer-Topologie steigt der verbleibene Loopwahrscheinlichkeitswert stark an, was vermutlich daran liegt, dass die jeweiligen Merkmalssequenzlängen weiterhin konsumiert werden müssen, aber eben nur von einem Zustand. Diese Erhöhung der Loopwahrscheinlichkeiten ist deutlich größer im Verhältnis zur Veränderung der Simulationswarscheinlichkeiten der einzelnen Zustände und ist dafür verantwortlich, dass während der Berechnung der Gesamtsimulationswahrscheinlichkeiten die Einer-Topologien am besten abschneiden.

Außerdem tendieren einige Phonmodelle (z.B. das `C`) dazu, durch längere Topologien modelliert zu werden. Hier lässt sich ebenfalls beobachten, dass der Simulationsalgorithmus nicht konvergiert und er die Topologien stetig verlängert. Die Begründung hierfür ist möglicherweise, dass an dieser Stelle ein Ganzwortmodell geschätzt wird, das sowohl über kleine Loop- als auch kleine Simulationswahrscheinlichkeiten der einzelnen Zustände verfügt.

Aus diesen Beobachtungen folgt, dass die Topologieoptimierung in der Anzahl der Zustände nach oben und unten begrenzt werden sollte. Ob es sich bei dieser Begrenzung um bestimmte Schwellwerte handelt, z.B. durch Begrenzung der Loopwahrscheinlichkeiten, oder ob, so wie in dieser Arbeit, schlicht begrenzende Zustandslängen vorgegeben werden, lässt sich aufgrund der vorliegenden Ergebnisse nicht abschließend feststellen. An dieser Stelle wären noch weitere systematische Untersuchungen nötig, um Vor- und Nachteile der

begrenzenden Mittel zu bewerten.

In dieser Arbeit konzentrierten sich die Versuche auf die Auswahl von linearen Topologien verschiedener Länge. Interessant wäre auch der Vergleich zwischen linearen Topologien auf einem Zweig und z.B. der reduzierten BAKIS-Topologie auf einem parallelen Zweig. Ob diese MPP-Simulation sinnvoll wäre, müsste jedoch genau überprüft werden, da eine solche komplexere Topologie über Wahrscheinlichkeitsdichten verfügt, die dem Datensharing unterliegen (Abschnitt 4.1). Das Datensharing könnte zu einer Verschmierung des verklebten Zustandes führen. Durch die daraus resultierende größere Varianz der verklebten Wahrscheinlichkeitsdichtefunktion könnte das Simulationsverfahren den einfachen linearen Zweig ohne Skip bevorzugen, so dass ein direkter Qualitätsvergleich nicht möglich ist.

Es ist daher sinnvoll nur Topologien miteinander zu vergleichen, die über jeweils die gleiche Anzahl von übersprungenden Zuständen verfügen. So wäre es ein interessanter Ansatz, die drei möglichen Varianten der reduzierten BAKIS-Topologien (Abschnitt 3.4) miteinander zu vergleichen, die jeweils durch den Skip verschiedene Zustände überspringen. Hiernach könnte eine Aussage darüber getroffen werden, ob es sinnvoll ist, dem Phon-HMM ausschließlich den Skip der quasi-statischen Phase der phonetischen Feinstruktur zu erlauben, oder ob stattdessen andere Teile der Feinstruktur weggelassen werden können, um das Erkennungsergebnis zu verbessern.

Trotz des Erfolges des in dieser Arbeit vorgestellten Ansatzes zur Topologieoptimierung sollte nicht unterschätzt werden, dass die Optimierung von den zu modellierenden Daten abhängig ist. Das vorliegende Korpus weist sehr spezielle umgangsprachliche Mundarten auf und es kann daher nicht ausgeschlossen werden, dass bestimmte Optimierungen nur für dieses Korpus gelten. Ein typisches Beispiel für diese Annahme ist das Phon `C`. Über 60% aller Beobachtungen werden nach der Optimierung mit drei Zuständen modelliert, womit das `C` durch längere Topologien nach der Optimierung modelliert wird (Abschnitt 6.6.5). Da gleichzeitig aus den Transkriptionsdaten errechenbar ist, dass ca. 40% aller phonetischen Beobachtungen des `C`'s aus dem Wort "Ich" `I C` stam-

men, liegt die Vermutung nahe, dass die Topologie für das Phon `C` auf das Ganzwort "Ich" hin optimiert wurde.

Es wäre daher interessant, diese Topologieoptimierung auf weitere Korpora anzuwenden, um das Optimierungsverhalten genauer zu studieren. An dieser Stelle würden sich nicht nur Korpora der deutschen Sprache, sondern auch Korpora anderer Sprachen anbieten.

Bedauerlicherweise konnten keine Vergleichsergebnisse aus anderen Veröffentlichungen herangezogen werden, da sich die verwendeten Systemparameter zu sehr unterscheiden. Die wesentlichen Unterschiede zu den anderen Systemen sind Triphonmodellierungen und z.B. gepoolte diagonale Kovarianzmatrizen. So konnten Veröffentlichungen gefunden werden, die Ergebnisse z.B. für VM1 um WFR 26 % (Beulen et al., 1998) und VM2 um 25 % (Kanthak et al., 2000) zeigen, die aber durch Systeme zustande gekommen sind, die mit andern Parametersätzen versehen waren. Dennoch konnte in dieser Arbeit gezeigt werden, dass es möglich ist, durch optimierte phonspezifische, monophone Topologien in Erkennungsbereiche vorzudringen, die bisher nur mit Triphonmodellierung und Trigrammen erreichbar waren.

Wichtig wäre daher, in weiteren Arbeiten zunächst eine wortinterne Triphonmodellierung algorithmisch vorzubereiten und den Simmulationsalgorithmus darauf anzuwenden. Bevor jedoch die Topologieoptimierung durchgeführt werden könnte, müssten die Eigenschaften der wortinternen Triphonmodellierung auf das Optimierungverfahren angepasst werden. Normalerweise werden wortinterne Triphone aufgrund der viel zu knappen Datenmengen aus der Feinstruktur von Monophonen generiert und anschließend durch geschicktes Verkleben vieler Parameter in ihrer Modellanzahl so stark reduziert, dass nur ein kleiner Teil (etwa 2200) sog. physikalischer Triphonmodelle übrig bleibt. Die benötigten sog. logischen Triphonmodelle werden aus den physikalischen Triphonmodellen durch Konkatenation einzelner Zustände gebildet.

Dieses Verfahren ist in Hinblick auf das in dieser Arbeit entwickelte Simulationsverfahren unbrauchbar, da die Simmulation unabhängig trainierte

physikalische Modelle benötigt. Ein Möglichkeit wäre es, einzelne Phon-HMMs schrittweise mit ihren Triphonkontexten zu untersuchen, während alle anderen Phone durch ihre Monophon-HMMs bereitgestellt werden. Während ein Phon-HMM auf seine Kontexte hin optimiert wird, bräuchten nur gleichzeitig maximal $42^2 + 41$ Modelle trainiert zu werden. Diese Vorgehen könnte dann für jedes einzelne Phon wiederholt werden.

Das zweite Problem während der Triphonmodellierung ist das Verkleben verschiedener Parameter. So können z.B. die einzelnen Zustände eines Dreier-HMMs kontextabhängig miteinander verklebt werden, um die Anzahl der trainierten Zustände zu reduzieren. Handelt es sich bei den HMMs jedoch um verschieden optimierte Topologien, wie es nach der Simulation zu erwarten wäre, lässt sich das Verkleben nicht mehr systematisch durchführen. Es sollte jedoch möglich sein, die Art des Verklebens auf die verschiedenen Längen der entstehenden HMM-Topologien anzupassen.

Schließlich konnte die Vermutung, dass sich ein Spracherkennungsproblem durch eine systematische, aber komplexe Topologie wie z.B. die reduzierte BAKIS-Topologie, besonders effizient lösen lässt, in dieser Arbeit nicht bestätigt werden. So war erwartet worden, dass zumindest die reduzierte BAKIS-Topologie nicht mehr verbesserbar wäre, da hier beide Topologievarianten (Dreier- und Zweier-Topologie) implizit auftreten und der eine zusätzliche Skip die Qualität nicht erheblich verschlechtern sollte. Die Beobachtungen dieser Arbeit zeigen jedoch, dass eine hohe Modellierungsqualität nur mit Topologien möglich ist, die an das jeweilige Problem angepasst sind.

Anhang A

Korpusinformationen

Das Verbmobil-Korpus ist in zwei Projektphasen entstanden, die als Verbmobil I [VM1] (Aufnahmen vor 1997) und als Verbmobil II [VM2] (Aufnahmen ab 1997) bekannt sind (Wahlster, 2000). Das Verbmobil-System erkennt gesprochene Spontansprache, analysiert die Eingabe, übersetzt sie in eine Fremdsprache, erzeugt einen Satz und spricht ihn aus. Für ausgewählte Themenbereiche (z.B. Terminverhandlung, Reiseplanung, Fernwartung) soll Verbmobil Übersetzungshilfe in Gesprächssituationen mit ausländischen Partnern leisten. Unterstützte Sprachen sind Deutsch, Englisch und Japanisch. Das Verbmobil-Projekt war eine über Jahre angelegte Initiative des Bundesministeriums für Bildung und Forschung, (BMBF), an dem mehrere Universitäten, Forschungsgemeinschaften und industrielle Partner teilgenommen haben. Details des Projektes können auf der Homepage (Hompage, 2000) nachgelesen werden. Details eines der Erkenner können u.a. in (Kanthak et al., 2000) nachgelesen werden.

A.1 Verbmobil I

Der deutschsprachige Teil des Verbmobil I-Trainingskorpus beinhaltet Mensch-zu-Mensch Dialoge, die unter geräusch- und störungsfreien Bedingungen aufgenommen wurden. Das Korpus VM1 enthält ca. 28 Stunden Sprachaufnahmen mit 13177 Sätzen. Das zugehörige Wörterbuch umfasst 6732 Einträge. Das Development-Set bzw. das Evaluations-Set (Test-Set) beinhaltet jeweils 342 Sätze.

A.2 Verbmobil II

Der deutschsprachige Teil des Verbmobil II-Trainingskorpus beinhaltet Mensch-zu-Mensch Dialoge, die ebenfalls unter geräusch- und störungsfreien Bedingungen aufgenommen wurden. Es beinhaltet ebenfalls die Aufnahmen und Transkriptionen des Verbmobil I-Projektteils und zusätzlich ca. 30 Stunden Sprachaufnahmen, bestehend aus 12318 Sätzen. Das Wörterbuch umfasst 5314 Wörter. Das Development-Set erhält zusätzlich 349 und das Evaluations-Set zusätzlich 352 Sätze zu den bereits vorhandenen Daten-Sets der ersten Projektphase.

A.3 Systemparameter

Folgende technische Parameter wurden während der Experimente verwendet:

- Fensterbreite: 20ms; Vorschub: 10ms
- Insgesamt 39 Koeffizienten: 12 MFCC Koeffizienten + Energie; davon jeweils die erste und zweite Ableitung.
- Separate Gaußsche Normalverteilungsdichten für jede Mixtur der Linearkombination für einen Zustand mit voll besetzter inverser Kovarianzmatrix.
- Wortfolgemodelle, die ausschließlich auf den Trainingsdaten trainiert werden.
- Die Wortübergangsstrafe wurde nicht verwendet und wurde daher mit 0.0 belegt.
- Die Strahlensuche wird in ihrer Breite begrenzt, um die Suchperformanz zu erhöhen. Der eingestellte Wert liegt bei 100.0.
- Die Skalierung für das Sprachmodell wurde mit 6.5 gewählt.

Anhang B

Liste des Phoninventars

In den folgenden Tabellen werden die in dieser Arbeit verwendeten Phonsymbole nebst Aussprachebeispielen angegeben. Insgesamt beinhaltet das Inventar 43 Phonsymbole und entspricht der erweiterten SAMPA Notation (Gibbon, 1997).

Schwache Vokale	
6	/b U t 6/ "Butter" nichtbetonter zentraler halboffener Vokal mit velarer Färbung (a-Schwa; r-Allophon)
@	/S 2: n @/ "schöne" nichtbetonter zentraler halboffener Vokal (Schwa)

Vokale	
i:	/t E 6 m i: n/ "Termin" gespannter langer ungerundeter geschlossener vorderer Vokal
I	/I n/ "in" ungespannter kurzer ungerundeter vorderer Vokal
y:	/z y: s/ "süß" gespannter langer gerundeter geschlossener vorderer Vokal

Y	/Y p s I l O n/ "Ypsilon" ungespanter kurzer gerundeter geschlossener vorderer Vokal
e:	/g e: n/ "gehen" gespannter langer ungerundeter halbgeschlossener vorderer Vokal
E	/Q a n g @ S t E l t @ n/ "Angestellten" ungespannter kurzer ungerundeter halboffener vorderer Vokal (offenes /E/)
E:	/b @ S t E: t I g U N/ "Bestätigung" gespannter langer ungerundeter halboffener vorderer Vokal (langes offenes E)
2:	/S 2: n/ "schön" gerundeter halbgeschlossener vorderer Vokal
9	/g l 9 k C @ n/ "Glöckchen" gerundeter halboffener vorderer Vokal
a:	/n a: m @/ "Name" langer offener hinterer Vokal
a	/d a x/ "Dach" offener hinterer Vokal
o:	/g r o: s/ "groß" gespannter langer halbgeschlossener hinterer Vokal
O	/Q O f @ n/ "offen" ungespannter kurzer halboffener hinterer Vokal (offenes /O/)
u:	/S u: l @/ "Schule" gespannter langer gerundeter geschlossener hinterer Vokal
U	/h U m @ l/ "Hummel" ungespannter kurzer gerundeter geschlossener hinterer Vokal (kurzes /U/)

Tabelle B.1: *Vokale des Phoninventars*

Diphthonge	
	Folgende Vokalsequenzen kommen als Diphthonge, in ähnlicher Distribution wie die langen Vokale, vor:
OY	"eu", "äu" usw.
aI	"ai", "ei" usw.
aU	"au"

Tabelle B.2: *Diphthonge des Phoninventars*

Konsonanten	
Plosive	
p	/a p/ "ab" stimmloser bilabialer Plosiv
b	/b @ r aI t/ "bereit" stimmhafter bilabialer Plosiv
t	/t E6 m i: n/ "Termin" stimmloser alveolarer Plosiv
d	/Q o: d P6/ "oder" stimmhafter alveolarer Plosiv
k	/b l O k/ "Block" stimmloser velarer Plosiv
g	/n a: g @ l/ "Nagel" stimmhafter velarer Plosiv
Q	/Q a n t r I t/ "Antritt" Glottalverschluß (vor betonten Anlautvokalen)
Frikative	
f	/f o: 6 b @ S p r E C U N/ "Vorbesprechung" stimmloser labiodentaler Frikativ
v	/v O x @/ "Woche" stimmhafter labiodentaler Frikativ
s	/S t r a: s @/ "Straße" stimmloser alveolarer Frikativ
z	/z E p t E m b 6/ "September" stimmhafter alveolarer Frikativ
S	/S t r a: s @/ "Straße" stimmloser postalveolarer Frikativ
Z	/Q a r a N Z i: r @ n/ "arrangieren" stimmhafter postalveolarer Frikativ (in Fremdwörtern)
C	/z I C t/ "Sicht" stimmloser palataler Frikativ (Alternant des /C/-/x/-Morphophonems; Ich-Laut)
x	/t aU x @ n/ "tauchen" stimmloser velarer Frikativ (Alternant des /C/-/x/-Morphophonems; Ach-Laut)
h	/h a l t @ n/ "halten" stimmloser glottaler Frikativ

Tabelle B.3: *Plosive und Frikative des Phoninventars*

Affrikate	
pf	/k n O pf/ "Knopf"
ts	/m E6 ts/ "März"
tS	/k v a tS/ "Quatsch"
	werden in dieser Form nicht modelliert;
	stattdessen werden /ts/ = /t s/ usw. benutzt

Halbvokal	
j	/j e: n 6/ "j e n e r"
	palataler Halbvokal (auch als Frikativ)

Sonoranten, Liquide	
l	/l OY f t/ "läuft"
	lateraler Sonorant
r	/f r I s t/ "Frist"
	zentraler Sonorant (auch als Frikativ)

Nasale	
m	/m e: 6/ "mehr"
	bilabialer Nasal
n	/n e: m @ n/ "nehmen"
	alveolarer Nasal
N	/p Y N k t l I C/ "pünktlich"
	velarer Nasal

Tabelle B.4: *Affrikate, Halbvokale, Sonoranten, Liquide und Nasale des Phoninventar*

Anhang C

Auszüge aus dem Ausprachewörterbuch

C.0.1 Beispiele für Buchstabiereinheiten

$\"A	Q E: sp
$\"O	Q P2: sp
$\"O-$G-$A-$I	Q P2: g e: Q a: Q i: sp
$\"O-$G-$A-$I-Tagung	Q P2: g e: Q a: Q i: t a: g U N sp
$\"U	Q y: sp
$\"U-Umlaut	Q y: Q U m l aU t sp
$A	Q a: sp
$A-$G-$T-$R	Q a: g e: t e: Q E P6 sp
$A-$G-$T-$R-Filiale	Q a: g e: t e: Q E P6 f I l j a: l @ sp
$A-$K	Q a: k a: sp
$A-$K-$D	Q a: k a: d e: sp
$A-$K-$D-$T	Q a: k a: d e: t e: sp
$A-$K-$D-$T-Filiale	Q a: k a: d e: t e: f I l j a: l @ sp
$A-$V-$B-$R	Q a: f aU b e: Q E P6 sp
$A-$V-$B-$R-Treffen	Q a: f aU b e: Q E P6 t r E f @ n sp

C.0.2 Beispiele für Wörter

\"a	Q E: sp
\"ahnlich	Q E: n l I C sp
\"ahnliche	Q E: n l I C @ sp

\"Ahnliches	Q E: n l I C @ s sp
[...]	
\"argerlich	Q E P6 g P6 l I C sp
\"Ather	Q E: t P6 sp
\"au\"serst	Q OY s P6 s t sp
\"au\"sersten	Q OY s P6 s t @ n sp
\"Odipus	Q P2: d i p U s sp
\"Of	Q P9 f sp
[...]	
a	Q a sp
A	Q a: sp
Aachen	Q a: x @ n sp
Ab	Q a p sp
ab	Q a p sp
abbrechen	Q a p b r E C @ n sp
Abend	Q a: b @ n t sp
abend	Q a: b @ n t sp
[...]	

C.0.3 Spezielle Modellierungen

!ANOISE	anoise
!BREATH	breath
!ENTER	sil
!EXIT	sil
!LAUGH	laugh
!NOISE	noise
<HUM>	usb
<NIB>	sil sil sil sil sil
<NIB>	sil sil sil sil
<NIB>	sil sil sil
<NIB>	sil sil
<NIB>	sil
<h\"as>	usb sp
\"ahm	Q E: m sp
hm	m sp

Anhang D

Modellierung akustischer Spezialmodelle

Die Pausenmodelle werden aufgrund einiger Besonderheiten gesondert modelliert und während des gesamten Experiments bzgl. ihrer Topologie konstant gehalten.

D.1 Modellierung größerer Sprechpausen

Sowohl zu Beginn als auch am Ende von Sprachaufnahmen entstehen oft Pausen, die eine durchschnittliche Sequenzlänge von 100ms aufweisen. Obgleich sehr viele Observationen für das Training vorhanden sind, stellt das Training der Pausen ein Problem dar, da sich ihre Merkmale zu wenig unterscheiden. Einzig Hintergrundgeräusche könnten zu einer Diversifizierung der einzelnden Merkmale führen.

Experimentatoren, die für die Aufnahmen eines Korpusses verantwortlich sind fordern gerne Laborbedingungen. Das würde bedeuten, das sich die einzelnen Merkmale der Pausen nicht mehr voneinander unterscheiden. Das hätte jedoch zur Folge, dass eine Wahrscheinlichkeitsdichtefunktion, die eine solche Pause modelliert, nahezu keine Streuung aufweist. Diese Situation wäre sehr ungünstig, da aufgrund der Normierung der Wahrscheinlichkeitsdichtefunktion ein nahezu unendlich hoher Wahrscheinlichkeitsdichtewert erzwungen wird,

der durch das Programm nicht mehr verarbeitbar wäre. Auch dürften keine Mixturen auf dieses Modell angewendet werden, da noch mehr "Genauigkeit" das Problem eher verschlimmern würde.

Hintergrundgeräusche sind aber nicht vollständig auszuschließen. Auch während einer Pause sind mit hoher Wahrscheinlichkeit immer irgendwelche Geräusche zu finden, so dass wir in der Lage sind, die Pausen bis zu einer Obergrenze an Genauigkeit schätzen zu können. Dieser Schätzung sind allerdings Grenzen gesetzt. Zu viele Gaußsche Mixturen können die Geräusche herausmodellieren, weswegen in dieser Arbeit versucht wird, mit wenigen Mixturen (max. 10 pro Zustand) und einer einfachen linearen Links-Rechts-Topologie mit fünf Zuständen auszukommen. In Abbildung D.1 wird das verwendete Modell dargestellt.

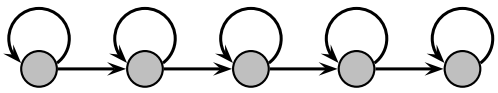

Abbildung D.1: *Das Pausenmodell*

Zusätzlich wird während der Experimente mit der Baseline-Topologie ein Pausenmodell mit nur drei emittierenden Zuständen verwendet. Diese Einschränkung ist notwendig, da sonst zu viele Datensätze aus dem Training herausfallen. Der Grund hierfür ist die unflexible Drei-Zustands-Topologie der Phonmodelle, die eine Mindestlänge der Sequenz fordert, die deutlich länger als die der anderen Topologien ist. Erfüllt die Beobachtungssequenz diese Anforderung nicht, weil sie "zu kurz" ist, wird sie während des Trainings nicht weiter berücksichtigt, was wiederum aufgrund der ohnehin schon knappen Daten ungünstig wäre.

D.2 Modellierung der Wortgrenzen

Selbst in spontan gesprochener Sprache kann es zu Pausen zwischen Worten kommen, die völlig unterschiedliche Längen aufweisen können. Um eventu-

elle Wortgrenzen modellieren zu können, wird in Abbildung D.2 ein Modell vorgeschlagen (Young et al., 2002), welches nur einen emittierenden Zustand aufweist, der zusätzlich noch übersprungen werden kann. Das bedeutet, dass der Zustand dieses Modells während der Viterbi-Suche auch vollständig weggelassen werden kann.

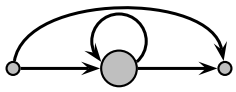

Abbildung D.2: *Modell einer Wortgrenze*

Auftretende Wortgrenzen werden in den Transkriptionen nicht gekennzeichnet. Nur echte Sprechpausen werden i.A. annotiert. Das heißt, dass kein ausreichendes Trainingsmaterial zur Verfügung steht, um dieses Modell zu trainieren. Andererseits sollte eine Wortgrenze sich nicht wesentlich von einem Zustand des *Pausen*-Modells unterscheiden. Aus diesem Grund wird die mittlere Verteilung b_3 des Pausenmodells mit dem einzigen Zustand im Wortgrenzenmodell verklebt und gemeinsam trainiert.

Anhang E

Script zur Signifikanzermittlung

```
#!/usr/bin/perl
#
# $Header: /u/drspeech/src/utils/signif/RCS/signif,
# v 1.5 1996/06/27 00:16:19 bilmes Exp $
# Written by Jeff Bilmes <bilmes@icsi.berkeley.edu>
# Wed Jun 26 1996
# This is a reverse engineered version of the old signif program
# written by Chuck Wooters but for which we have no source.

# signif:
# Usage:
#   signif -a p1 -b p2 -n N

#  Assumption: We've run two experiments each consisting of N
#  independent trials. In the first experiment, p1*N of the trials
#  came out true. In the second experiment, p2*N of the trials came
#  out true.  We want to find out if p2 is significantly (in a
#  statistical sence) better than p1 under N samples.  We do
#  this by assuming a null hypothesis, H0, and seeing how unlikely
#  this null hypothesis is.

#  Since we want to test whether one process (p2) is better than
#  another p1 (as apposed to testing whether one process is either
#  better or worse than another), we use the two
#  hypothesis.
#
#    H0: p1 == p2, i.e., the scores are really identical
#    H1: p1 > p2 , i.e., and p2 is indeed better than p1 at
#                        the current significance level.
#
```

```
#  We do this by assuming H0 and then disproving it at various
#  significance levels.
#
#  We compute the z-score from the statistics of
#  the difference between two random variables with their
#  corresponding distributions. We get:
#      u_diff = p1 - p2                                  (1)
#  and standard deviation
#      sigma_diff = sqrt(p1*q1/N + p2*q2/N)              (2)
#
#  where q1 = 1-p1 and q2 = 1-p2. The distribution of
#  this difference, we assume, is approximately
#  normal with a large enough sample size (N) and therefore
#  we can compute the normalized z-score as:
#   z = (p1-p2)/sigma_diff
#
#  Once we have this z-score and since we are interested in if
#  one process is better than the other, use use a one-tailed
#  rather than a two-tailed significance test.

#  One tailed significance values.  Note: The older version of signif
#  (the one Chuck Wooters wrote and the one we can't find the source
#  for) used only two digits of precision for the significance values
#  (in particular, that program used 2.33 and 1.65 for significance
#  levels 0.05 and 0.01 respectively). We use more digits of precision
#  and therefore get slightly different results.
#  These values came from mapleV using the commands:
#      with(stats):
#           statevalf[icdf,normald](v);
#  where v takes on the significance levels 0.10, 0.05, etc.

%signif_vals = (
# levels   values
  0.10,   1.281551566,
  0.05,   1.644853627,
  0.01,   2.326347874,
  0.005,  2.575829304,
  0.002,  2.878161739,
  0.001,  3.090232306
               );

sub usage {
 printf STDERR (
  "Usage: signif -a p1 -b p2 -n N\n" .
  "   where p1 = the probability of event A (error score of process A)\n" .
  "   where p1 = the probability of event B (error score of process B)\n" .
  "   N is the number trials (number of examples)\n");
```

```
}

sub checkfloat {
    ($opt,$arg) = @_;
    if (( $arg !~ /^[0-9.]+$/ ) || ($arg < 0.0 || $arg > 1.0)) {
        print STDERR ("Error: Value \"", $arg, "\" invalid for option ",
                      $opt, " (real number between 0 and 1 expected)\n");
        &usage; exit(-1);
    }
}

sub checkint {
    ($opt,$arg) = @_;
    if (( $arg !~ /^[0-9]+$/ ) || ($arg < 1)) {
        print STDERR ("Error: Value \"", $arg, "\" invalid for option ",
                      $opt, " (positive integer expected)\n");
        &usage; exit(-1);
    }
}

# parse arguments
if ( $#ARGV < 0 ) {
    &usage;
    exit(0);
}
while ( $#ARGV >= 0 ) {
    $opt = shift(@ARGV);
    $opt =~ tr/A-Z/a-z/; # ignore case
    if ( $opt =~ /^-a/ ) {
        if ( $#ARGV < 0 ) {
            print STDERR ("Error: Option ", $opt,
                          " requires an argument\n");
            &usage; exit(-1);
        }
        $arg = shift(@ARGV);
        &checkfloat($opt,$arg);
        $p1 = $arg;
    } elsif ( $opt =~ /^-b/ ) {
        if ( $#ARGV < 0 ) {
            print STDERR ("Error: Option ", $opt,
                          " requires an argument\n");
            &usage; exit(-1);
        }
        $arg = shift(@ARGV);
        &checkfloat($opt,$arg);
        $p2 = $arg;
    } elsif ( $opt =~ /^-n/ ) {
```

```
        if ( $#ARGV < 0 ) {
            print STDERR ("Error: Option ", $opt,
                          " requires an argument\n");
            &usage; exit(-1);
        }
        $arg = shift(@ARGV);
        &checkint($opt,$arg);
        $N = $arg;
    } elsif ( $opt =~ /^-help/ ) {
        &usage;
        exit(0);
    } else {
        printf STDERR ("Error: Unknown option (%s)\n",$opt);
        &usage;
        exit(-1);
    }
}


$diff = ($p1-$p2);
$diff = -$diff if ($diff < 0);

printf "Number of patterns differing is = %d\n",
    int($diff*$N+0.5);

if ($p1 == $p2) {
    printf "WARNING: p1 = p2 = $p1. Therefore,
            this difference is not significant at any level.\n";
    exit(0);
}

$sigma = sqrt(($p1*(1.0-$p1)+$p2*(1-$p2))/$N);
$z = $diff/$sigma;

printf "This difference is:\n";
foreach $level (sort keys(%signif_vals)) {
    $val = $signif_vals{$level};
    if ($z <= $val) {
        printf "Not significant";
        # i.e., with probability 1.0 - $level, this difference
        # could have been generated by the null hypothesis.
    } else {
        printf "Significant";
        # i.e., with probability $level, this difference
        # could have been generated by the null hypothesis.
        # Since $level is small, we say this is significant.
    }
```

```
    printf " at the %.3f level (%0.2f differences required)\n",
      $level,
      $N*$val*$sigma;
}
```

Tabellenverzeichnis

Abbildungsverzeichnis

Literaturverzeichnis

N. Ahmed and K. R. Nasir. *Orthogonal Transforms for Digital Signal Processing*. Springer Berlin, 1975.

L. Bahl, R.Bakis, P.Cohen, A. Cole, F. Jelinek, B. Lewis, and R. Mercer. Further results on the recognition of a continuously read natural corpus. In *Proc. Int. Conf. on Acoustics, Speech and Signal Processing, Denver, Colorado*, 1980.

R. Bakis. Continuous-speech word spotting via centisecond acoustic states, 1974.

L. Baum and J. Eagon. An Inequality with Applications to Statistical Estimation for Functions of Marcov Processes and to a Model for Ecology. In *Bull. Amer. Math. Soc., Bd 73*, 1967.

L. Baum and T. Petrie. Statistical Inference for Pobabilistic Functions of Finite State Markov Cains. In *Ann. Math. Statist., Bd 37*, 1966.

L. Baum, T. Petrie, G. Soules, and N. Weiss. A Maximization Technique Occuring in the Statistical Analyses of Probabilistic Functions of Markov Chains. In *Ann. Math. Statist., Bd. 41, S. 164-171*, 1970.

L. Baum and G. Sell. Growth Transformations for Functions on Manifolds. In *Pacific Journal of Mathematics, Bd. 27*, 1968.

C. Bechetti and L. P. Ricotti. *Speech Recognition: Theory and C++ Implementation*. John Wiley and Sons, Chichester,West Sussex, England S. xxx, 1999.

N. Beringer and F. Schiel. The Quality of Multilingual Automatic Segmentation Using German MAUS. In *Proc. of the International Conference on Spoken Language Processing*, Beijing, China, 2000.

K. Beulen, S. Ortmanns, A. Eiden, S. Martin, L. Welling, J. Overmann, and H. Ney. Pronunciation Modelling in the RWTH Large Vocabulary Speech Recognizer. In *Proc. ESCA Tutorial and Research Workshop 'Modeling Pronunciation Variation for Automatic Speech Recognition, Kerkrade, Netherlands*, 1998.

H. Bußmann. *Lexikon der Sprachwissenschaft.* Alfred Körner Verlag Stuttgart, 1990.

H. Dallmansn and K.-H. Elster. *Einführung in die höhere Mathematik 3; s. 568ff.* Vieweg, 1983.

S. B. Davis and P. Mermelstein. Comparsion of Parametric Representation for Monosyslabic Word Recognition in Countinousley Spoken Sentences. In *IEEE on Acoustics, Speech and Signal Processing, Bd.28*, 1980.

L. Deg, M. Lennig, V. Gupta, and P. Mermelstein. Modeling acoustic-phonetic details in an hmm-based large vocabulary speech recognizer. In *In Proc. Int. Conf. on Acoustics, Speech and Signal Processing, New York, S. 509-512*, 1988.

A.-M. Derouault. Context-Dependent phonetic Markov Modells for large vocabulary speech recognition. In *Proc. Int. Conf. on Acoustics, Speech and Signal Processing, Dallas,Texas, S. 360-363,*, 1987.

R. Duda and P. Hart. *Pattern Classification and Scene Analysis.* Wiley, 1973.

G. Fant. *The Acoustic Theory of Speech Production.* Mouton & Co, 1960.

G. Fink. *Generalisierte Triphone als Wortuntereinheiten im Akustisch-Phonetischen Netwerk.* Diplomarbeit, Universität Erlangen, 1991.

J. Fourier. *Theorie analytique de la chaleur.* Paris, Chez Firmin Didot, Père et Fils,, 1822.

S. Furui. Speaker-Independent Isolated Word Recognition Using Dynamic Features of Speech Spectrum. In *IEEE Trans. on Acoustics, Speech and Signal Processing, Bd.34*, 1986.

A. Gersho and V. Cuperman. Vector quantization: a pattern-matching technique for speech for speech coding. In *IEEE Communications Magazine, S. 15-23*, 1983.

A. Gersho and R. M. Gray. *Vector Quantization and Signal Compression*. Kluver Academic Publishers, 1992.

D. Gibbon. SAMPA-D-VMlex, 1997. URL `http://coral.lili.uni-bielefeld.de/Documents/sampa-d-vmlex.html`.

A. Hauenstein. *Aussprachewörterbücher zur automatischen Spracherkennung*. Infix, 1996.

V. Hompage. Verbmobil Hompage, 2000. URL `http://verbmobil.dfki.de/`.

J. Hopcroft and J. Ullman. *Introduction to Automata Theory, Languages and Computation*. Addison-Wesley, 1979.

X. Huang, A. Acero, and H.-W. Hon. *Spoken Language Processing*. Prentice Hall PTR, 2001.

X. Huang, Y. Ariki, and M. Jack. Hidden Markov Models for Speech Recognition. *Information Technology Series*, 1990.

X. Huang and M. Jack. Semi-Continous Hidden Markov Models for Speech Signals. *Computer Speech & Language 3*, 1989.

X. Huang, K. Lee, H. Hon, and M. Hwang. Improved Acoustic Modelling with the SPHINX Speech Recognition System. In *Proc. Int. Conf. on Acoustics, Speech and Signal Processing, Toronto, Canada*, 1991.

J. L. Jay, J. Kim, and J. H. Kim. Data-driven Design of HMM Topology for On-line Handwriting Recognition. *World Scientific Series In Machine Perception And Artificial Intelligence Series*, 2001.

F. Jelinek. The development of an experimental discrete dictation recognizer. In *Proceedings of IEEE, Bd. 73, S. 1616-1624*, 1985.

F. Jelinek. *Statistical Methods for Speech Recognition*. The MIT Press, 1997.

F. Jelinek, R. Mercer, and L.Bahl. *Continuous Speech Recognition*. Handbook of Statistics Bd.2 , North-Holland, 1982.

B. Juang, D. Wong, and A. Gray. Distorsion Performance of Vector Quantization for LPC Voice Coding. In *IEEE Trans. on Acoustics, Speech and Signal Processing, Vol. 30, No. 2, S. 294-303*, 1982.

S. Kanthak, A. Sixtus, S. Molau, R. Schlüter, and H. Ney. The RWTH Large Vocabulary Speech Recognition System for Spontaneous Speech. In *KONVENS*, pages 249–254, 2000.

O. Kimball, P. Price, S. Roucos, R. Schwartz, F. Kubala, Y.-L. Chow, A. Haas, M. Kramer, and J. Makhoul. Recognition performance and grammatical constraints. In *In Proceedings of the DARPA Speech Recognifim Workshop, Science Applications International Corporation Report Number SAIC-86/1546, S. 53-59*, 1986.

D. Knoblauch. Data Driven Number-of-States Selection in HMM Topologies. In *Proc. Int. Conf. on Acoustics, Speech and Signal Processing, Jeju, South Korea*, 2004.

C. Lee, E. Giachin, L. Rabiner, R. Pieraccini, and A. Rosenberg. *Improved Acoustic Modeling for Large Vocabulary Continuous Speech Recognition*. Computer, Speech & Language Bd. 6, 1992.

K.-F. Lee. *Automatic Speech Recognition: The Development of the SPHINX SYSTEM*. Kluwer Academic Publishers, Boston, 1989.

S. Levinson, L. Rabiner, and M. Sondhi. An introduction to the application of the theory of probabilistic functions of a markov process to automatic speech recognition. In *Bell Systems Technical Journal, Bd. 62, Nr.4 p.1035-1074*, 1983.

G. Meinhold and E. Stock. *Phonologie der deutschen Gegenwartssprache*. VEB Bibliographisches Institut. Leipzig, 1982.

D. Mergel and H. Ney. Phonetically Guided Clustering for Isolated Word Recognition. In *Proc. Int. Conf. on Acoustics, Speech and Signal Processing, Tampa, Florida*, 1985.

H. Ney. Stochastic grammars and pattern recognition. In *Speech Recognition and Understanding.Recent Advances,Trends and Applications Bd. 75, s.319-344*. F.Springer, 1992.

H. Niemann. *Klassifikation von Mustern*. Springer, Berlin, 1983.

W. Ortmann. *Sprechsilben im Deutschen*. Ghoete-Institut. München, 1980.

L. Rabiner and B. Juang. *Fundamentals of Speech Recognition*. Prentice Hall, 1993.

A. Richter. Modeling of Continuous Speech Observations. In *IBM Advances in Speech Processing Conference*, 1986.

S. Sagayama. Phonem environment clustering for speech recognition. In *Proc. Int. Conf. on Acoustics, Speech and Signal Processing, Glasgow,S. 397-400*, 1989.

A. Salomaa. *Formale Sprachen*. Springer, 1978.

F. Schiel. Automatic Phonetic Transcription of Non-Prompted Speech. In *Proc. of the ICPhS 1999*, San Francisco, 1999.

E. Schukat-Talamazzini. *Automatische Spracherkennung*. Vieweg, 1995a.

E. Schukat-Talamazzini. *Automatische Spracherkennung, S. 133*. Vieweg, 1995b.

R. Schwartz, Y. Chow, O. Kimball, S. Roucos, M. Krasner, and J. Makhoul. Improved Hidden Markov Modeling of Phonemes for Continuous Speech Recognition. In *Proc. Int. Conf. on Acoustics, Speech and Signal Processing, Tampa, Florida*, 1985.

J. Shoup. *Phonetical Aspects of Speech Recognition in Trends in Speech Recognition.* Prentice-Hall Inc., Englewood Cliffs, New Jersey, 1980.

T. Svendsen, K.Paliwal, E. Harborg, and P. Husoy. An improved sub-word based speech recognizer. In *Proc. Int. Conf. on Acoustics, Speech and Signal Processing, Glasgow*, 1989.

A. J. Viterbi. Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm. In *IEEE Transactions on Information Theory*, Beijing, China, 1967.

W. Wahlster. *Verbmobil: Foundations of Speech-to-Speech Translation.* Springer, 2000.

A. Wendemuth. *Grundlagen der stochastischen Sprachverarbeitung.* Oldenburg, 2004.

S. Yakowitz. Unsupervised Learning and the Identification of Finite Mixtures. In *IEEE Trans. on Information Theory Bd. 16*, 1970.

S. Young, G. Evermann, D. Kershaw, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, and P. Woodland. *The HTK Book (Version 3.2).* Microsoft Corp., 2002.

Y. Zhao, H. Wakita, and X. Zhuang. An HMM Based Speaker-Independent Continous Speech Recognition System with Experiments on the TIMIT Database. In *Proc. Int. Conf. on Acoustics, Speech and Signal Processing, Toronto, Canada*, 1991.